# LATENT SEARCH MANIFOLDS: REUSING CONTROL EXPERIENCE FOR PDE-CONSTRAINED OPTIMIZATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

High-dimensional PDE-constrained control in repeated-task settings faces two distinct online burdens: evaluating candidate controls can require expensive PDE solves, while optimization must still explore a high-dimensional control space. Learned operators can reduce evaluation cost, but a cheap evaluator does not determine where the optimizer should search and can become unreliable when search leaves well-supported regions. We exploit a different reusable asset: the controls produced by previous optimization tasks. Treating these controls as optimization experience, we organize their shared structure into a continuous, low-dimensional search manifold. A sliced-Wasserstein autoencoder provides the representation needed to reconstruct controls while shaping a latent space that can be searched directly. New tasks are then solved by optimizing their objectives on this manifold rather than predicting controls directly. Representation, optimizer, and evaluator can therefore be chosen separately: the manifold determines where to search, derivative-free optimization determines how to search, and either a numerical solver or a frozen neural operator evaluates candidate controls, with final solutions numerically verified. Across eight heterogeneous PDE-control benchmarks, controlled comparisons show that search geometry alone can change solver-verified outcomes by orders of magnitude: on NS2inlets, the objective decreases from $2.325$ in the raw control space to $0.0345$ on the learned manifold under the same search budget. The learned search space remains compatible with different derivative-free optimizers and with either exact or surrogate evaluation; when PDE solves are expensive, the latter can further reduce online cost. These results identify reusable control geometry as an independent design axis for repeated PDE optimization, complementary to the choices of optimizer and evaluator.

## 1 INTRODUCTION

Predictive physical models become most consequential when they can inform decisions about how physical systems should be controlled or designed. PDE-constrained optimization provides a principled framework for making such decisions: rather than only asking how a system evolves, it asks how the system should be controlled or designed to achieve a desired outcome (Gunzburger, 2003; Hinze et al., 2009; Biegler et al., 2007). This perspective underlies problems ranging from fluid-flow control and thermal management to physics-based design (Manzoni et al., 2021; Pearson, 2018; Ulbrich & Vexler, 2021). Such decisions can nevertheless be computationally demanding. Spatial or spatiotemporal controls may contain many degrees of freedom after numerical discretization, and evaluating a candidate typically requires solving the governing PDE (Hinze et al., 2009; Manzoni et al., 2021). Classical adjoint methods provide a powerful and scalable route when reliable sensitivities are available (Giles & Pierce, 2000; Gunzburger, 2003). Yet many practical settings require not one optimization, but a sequence of related ones as targets, objectives, or operating conditions change. In this repeated-task regime, the same two costs return with every new task: expensive physical evaluation and search over a high-dimensional control space.

Learning-based methods have increasingly reused information across PDE-control problems to reduce online computation. Neural-operator surrogates replace repeated PDE solves with learned solution maps (Hwang et al., 2022), while physics-informed and derivative-aware methods make sensitivity information more useful for optimization (Hao et al., 2023; O'Leary-Roseberry et al., 2024; Luo et al., 2025; Behroozi et al., 2025; Cheng et al., 2025; Yao et al., 2025; Gong et al.,



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

2026). Learned controllers take a different route by amortizing the task-to-control map across related problems (Guven et al., 2025). These approaches reuse models, derivatives, or direct predictions, but leave another source of repeated effort largely untouched: deciding which regions of a high-dimensional control space are worth exploring. This raises a complementary question: can experience from previous optimization tasks also tell us *where to search*? We discuss the closest connections to PDE-constrained optimization, operator learning, and latent-space search in Appendix A.

Our key observation is that repeated optimization leaves useful information in the controls it produces. Each solved task provides not only a task-specific answer, but also evidence about where high-quality controls tend to lie. Across related tasks, these optimized controls can therefore reveal reusable structure within the much larger feasible space. We treat this structure as optimization experience and organize it into a low-dimensional search manifold, $c \approx D(z)$, so that a new task is solved by optimizing $z$ rather than predicting its control directly. To make this manifold directly searchable, we use a sliced-Wasserstein autoencoder (SWAE) (Kolouri et al., 2019): reconstruction preserves control structure, while distribution-regularized latent coordinates and deterministic decoding provide a continuous search domain that maps back to physical controls. The resulting geometry is learned from the historical task family, which therefore defines its expected transfer regime.

Learning the search space separately also lets us decouple *where to search* from *how to search* and *how candidates are evaluated*. The learned manifold defines the search domain, while the derivative-free optimizer and evaluator remain replaceable: candidates may be scored by either the numerical PDE solver or an optional neural-operator surrogate without relearning the control representation. We use numerical verification as the common reference for final solutions. This makes search geometry, optimizer choice, and evaluation cost distinct design axes rather than a fixed end-to-end pipeline.

We examine these three design choices across eight heterogeneous PDE-control benchmarks, varying the search representation, derivative-free optimizer, and evaluator while numerically verifying the final solutions. Search geometry has a pronounced effect even when the optimizer and budget are fixed: on NS2inlets, for example, latent search reduces the objective from $2.325$ to $0.0345$ relative to raw-space search. Yet the strongest optimizer changes across problems, separating this effect from any particular search algorithm. Surrogate evaluation offers a different advantage when numerical PDE solves dominate online cost, and structured search can further reduce proxy-to-solver mismatch in poorly supported regions. These findings point to a broader opportunity in repeated PDE optimization: rather than starting each new task from the full control space, we can reuse the geometry revealed by solutions to previous tasks—learning not the next control itself, but *where to search for it*.

## 2 Problem Setting

We consider a family of related PDE-constrained control tasks $\tau \in \mathcal{P}$. Each task specifies a governing PDE, an objective, and instance-specific information such as a target, forcing, geometry, or boundary conditions. Within a task family, controls share a compatible parameterization $c \in \mathcal{C} \subseteq \mathbb{R}^{d_c}$; different task families may use different control spaces. Let $\mathcal{S}_\tau$ denote the numerical control-to-state map induced by $\mathcal{F}_\tau(u, c) = 0$. The reduced control problem is:

$$\min_{c \in \mathcal{C}} j_\tau(c), \qquad j_\tau(c) := J_\tau(\mathcal{S}_\tau(c), c)\,. \tag{1}$$

Our setting is repeated optimization. Before a new task is solved, related tasks have produced a collection of high-quality controls:

$$\mathcal{D}_{\mathrm{ctrl}} = \{(c_i, \rho_i)\}_{i=1}^{N}\,, \tag{2}$$

Here $\rho_i$ denotes optional structural context, such as the temporal phase of a trajectory segment, and is omitted when unnecessary. The controls need not be globally optimal; their collective support serves as the reusable optimization experience considered here.

We seek a low-dimensional parameterization of this reusable search support. Let $D_\theta$ be a decoder and $\mathcal{Z}_{\mathrm{search}} \subset \mathbb{R}^{d_z}$, with $d_z \ll d_c$. The resulting control search space is:

$$\mathcal{M}_\theta = \{D_\theta(z; \rho) : z \in \mathcal{Z}_{\mathrm{search}}\} \subseteq \mathcal{C}. \tag{3}$$

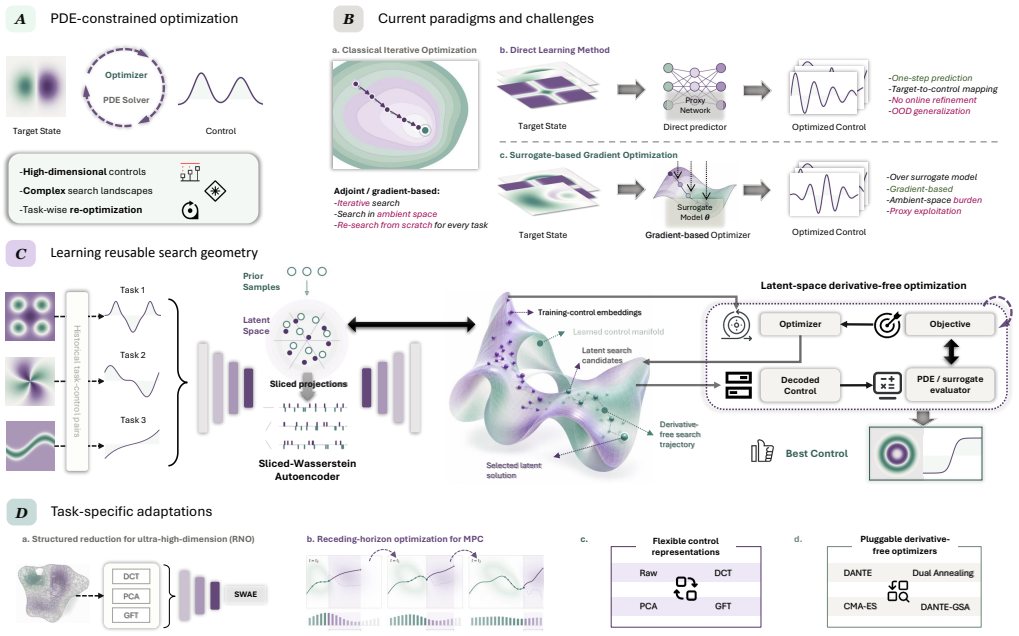


Figure 1: **Learning where to search for PDE-constrained optimization. A,** PDE-constrained control problem. **B,** Existing optimization and learning paradigms. **C,** Our framework learns a reusable control manifold from historical optimized controls and performs derivative-free latent search. **D,** Task-specific adaptations for high-dimensional and trajectory controls.

The online task $\tau$ does not predict a control through $D_\theta$; instead, it enters through the objective used to search $\mathcal{M}_\theta$. Candidate controls may be evaluated either by the numerical PDE objective or by a learned surrogate:

$$\mathcal{E}_\tau(c) = \begin{cases} j_\tau(c), & \text{exact evaluation,} \\ \widehat{j}_\tau(c), & \text{surrogate evaluation.} \end{cases} \tag{4}$$

For a new task, the latent optimization problem is:

$$z_\tau^\star \in \arg\min_{z \in \mathcal{Z}_{\text{search}}} \mathcal{E}_\tau(D_\theta(z;\rho)) \,. \tag{5}$$

This formulation separates three roles: historical controls determine *where* to search, the optimizer determines *how* to search, and the evaluator determines the cost and fidelity of each query. Surrogate-guided solutions are numerically re-evaluated before final performance is reported.

## 3 Method

The central idea is to reuse past optimization not by predicting the next control, but by restricting where the next search takes place. Fig. 1 summarizes this shift. Historical optimized controls are organized into a searchable manifold, on which a derivative-free optimizer solves each new task. The search geometry is learned once, while the optimizer and evaluator remain replaceable; the same principle extends to structured high-dimensional controls and receding-horizon trajectory optimization. All learned components are trained offline and fixed before optimization on a new task.

### 3.1 Learning a Searchable Control Manifold

**From optimization experience to searchable coordinates.** A representation used for optimization must do more than reconstruct historical controls: online search will also query latent points not

observed during training. For each $(c_i, \rho_i) \in \mathcal{D}_{\text{ctrl}}$, we define:

$$\begin{aligned} x_i &= \Psi(\mathcal{N}(c_i)), \qquad z_i = E_\phi(x_i; \rho_i), \qquad \widehat{x}_i = g_\theta(z_i; \rho_i), \\ D_\theta(z; \rho) &= \Pi_{\mathcal{C}} \mathcal{N}^{-1}\left[\Psi^\dagger(g_\theta(z; \rho))\right], \qquad z \in \mathbb{R}^{d_z}, \quad d_z \ll d_x. \end{aligned} \tag{6}$$

Here $\Psi$ is the identity for direct-control benchmarks and a DCT, PCA, or graph-spectral map for irregular controls; $\Psi^\dagger$ is the corresponding synthesis map, and $\Pi_{\mathcal{C}}$ enforces physical constraints. PCA retains leading empirical-variance directions, whereas DCT and graph-spectral parameterizations retain prescribed low-frequency modes. These maps are fitted offline and frozen before optimization; SWAE then defines the latent coordinates used for search.

**Sliced-Wasserstein regularization.** Because these coordinates will be explored by an optimizer, reconstruction alone is insufficient: the aggregate latent geometry must also remain organized away from the encoded training points. We therefore regularize the aggregate code distribution $Q_\phi$ toward a simple prior $P_Z$. For $v \in \mathbb{S}^{d_z - 1}$ and $R_v(z) = v^\top z$, the sliced-Wasserstein distance is:

$$\begin{aligned} \mathrm{SW}_2^2(Q_\phi, P_Z) &= \int_{\mathbb{S}^{d_z-1}} W_2^2((R_v)_\# Q_\phi, (R_v)_\# P_Z)\, d\sigma(v), \\ W_2^2(\mu, \nu) &= \int_0^1 \left|F_\mu^{-1}(t) - F_\nu^{-1}(t)\right|^2 dt. \end{aligned} \tag{7}$$

following the sliced-Wasserstein construction (Kolouri et al., 2019). For a minibatch $\{z_b\}_{b=1}^B$, prior samples $\{\xi_b\}_{b=1}^B \sim P_Z$, and random directions $\{v_\ell\}_{\ell=1}^L$, we use the empirical estimator:

$$\widehat{\mathrm{SW}}_2^2(Q_\phi, P_Z) = \frac{1}{L} \sum_{\ell=1}^{L} \frac{1}{B} \sum_{b=1}^{B} \left[(v_\ell^\top z)_{(b)} - (v_\ell^\top \xi)_{(b)}\right]^2, \tag{8}$$

where $(\cdot)_{(b)}$ is the $b$-th order statistic; for equally weighted samples, sorting computes the projected one-dimensional transport exactly. The manifold is trained with:

$$\mathcal{L}_{\text{man}} = \underbrace{\frac{1}{N} \sum_{i=1}^{N} \|x_i - g_\theta(E_\phi(x_i; \rho_i); \rho_i)\|_2^2}_{\mathcal{L}_{\text{rec}}} + \beta\, \widehat{\mathrm{SW}}_2^2(Q_\phi, P_Z) + \lambda\, \mathcal{R}_{\text{ctrl}}. \tag{9}$$

Reconstruction anchors the representation to historical high-quality controls, while sliced-Wasserstein regularization organizes their aggregate latent geometry. The optional $\mathcal{R}_{\text{ctrl}}$ imposes benchmark-specific spatial or temporal regularity. Importantly, this regularization shapes the coordinate system but does not by itself define the region used for online search.

**From regularized coordinates to a search domain.** We derive the online search support from the encoded optimization experience. Letting $Q_\alpha$ denote an empirical quantile and $m_j \geq 0$ an optional margin, we define:

$$\begin{aligned} \mathcal{Z}_{\text{train}} = \{E_\phi(x_i; \rho_i)\}_{i=1}^N, \qquad \ell_j = Q_{\alpha_j}(z_j) - m_j, \qquad u_j = Q_{1-\alpha_j}(z_j) + m_j, \\ \mathcal{Z}_{\text{search}} = \prod_{j=1}^{d_z} [\ell_j, u_j]. \end{aligned} \tag{10}$$

Thus, the prior regularizes the latent coordinates, whereas historical optimized controls determine the compact region exposed to downstream search.

### 3.2 Derivative-Free Search on the Learned Manifold

Once the manifold is learned, its representation pipeline and $\mathcal{Z}_{\text{search}}$ are fixed for new tasks. For task $\tau$, the latent objective is:

$$F_\tau(z; \rho) := \mathcal{E}_\tau(D_\theta(z; \rho)), \qquad z \in \mathcal{Z}_{\text{search}}, \tag{11}$$

where $\mathcal{E}_\tau$ is the evaluator defined in Sec. 2. We use derivative-free optimization as a common scalar-query interface for both numerical and surrogate evaluators.

Under a budget of $B$ evaluator calls, the returned solution is:

$$\widehat{z}_{\tau,B} \in \arg\min_{1\le b\le B} F_\tau(z^{(b)};\rho), \qquad \widehat{c}_{\tau,B} = D_\theta(\widehat{z}_{\tau,B};\rho). \tag{12}$$

We measure the online budget by actual calls to $\mathcal{E}_\tau$, rather than optimizer iterations, so that different search methods are compared under the same query budget.

### 3.3 Receding-Horizon Latent Search for Trajectory Controls

A single latent code can specify a static control, but using one code for an entire long-horizon trajectory would couple future actions before the states encountered during execution are known. We instead learn from phase-conditioned length-$H$ control windows extracted from historical trajectories and repeatedly apply the same latent-search principle over a shorter horizon. At replanning step $k$, each candidate rollout starts from the current numerical PDE state $u_k$:

$$\widetilde{u}_{k|k} = u_k, \qquad \widetilde{u}_{k+\ell+1|k}(z) = \widetilde{\Phi}_\tau\big(\widetilde{u}_{k+\ell|k}(z), c_{k+\ell|k}(z)\big), \quad \ell = 0,\dots,H_k-1, \tag{13}$$

where $c_{k+\ell|k}(z)$ is decoded from the phase-conditioned window and $\widetilde{\Phi}_\tau$ may be numerical or learned. We optimize the corresponding finite-horizon objective over $z$, initialized from phase-matched historical codes, but execute only the first $s_k \le H_k$ controls with the numerical PDE dynamics before replanning from the resulting state. This repeatedly anchors surrogate-guided search to the evolving numerical PDE state. For Burgers, this replaces a single search over a $200 \times 81 = 16{,}200$-dimensional trajectory with successive 32-dimensional searches over decoded $20 \times 81$ windows. Window extraction, smoothing, and warm-start details are provided in Appendix D.1.

### 3.4 Exact and Surrogate Candidate Evaluation

The evaluator determines how each decoded control is scored while leaving the representation and optimizer unchanged. Let $\widehat{\mathcal{S}}_{\psi,\tau}$ denote an offline-trained, frozen operator surrogate, with $\widehat{j}_\tau(c) = J_\tau(\widehat{\mathcal{S}}_{\psi,\tau}(c), c)$. The exact and surrogate latent objectives are:

$$F^{\mathrm{ex}}_\tau(z;\rho) = j_\tau(D_\theta(z;\rho)), \qquad F^{\mathrm{sur}}_\tau(z;\rho) = \widehat{j}_\tau(D_\theta(z;\rho)). \tag{14}$$

Both evaluators expose the same black-box interface: the latent optimizer receives only scalar objective values from either evaluator, and surrogate derivatives are not used. For trajectory control, the corresponding choice is represented by the numerical or learned $\widetilde{\Phi}_\tau$ in Eq. 13.

Surrogate evaluation reduces the cost of scoring a candidate by replacing the numerical objective with an approximation. Its practical benefit therefore depends jointly on objective fidelity and on the amount of search enabled by its evaluation cost.

**Numerical verification.** For surrogate-guided search, let $\mathcal{I}_K$ index the $K$ retained candidates with the lowest surrogate objectives. These candidates are re-evaluated with the numerical PDE solver, and the final control is:

$$c^{\mathrm{out}}_\tau \in \arg\min_{i\in\mathcal{I}_K} j_\tau\Big(D_\theta(z^{(i)};\rho)\Big). \tag{15}$$

Here $K = 1$ numerically verifies the surrogate-selected candidate, while $K > 1$ reranks multiple retained candidates by their numerical objectives. Candidates evaluated in exact mode already use the numerical objective. For receding-horizon control, each executed block is additionally propagated by the numerical PDE dynamics before replanning.

## 4 Experiments

We evaluate the framework on eight PDE-control benchmarks spanning static and trajectory optimization, distributed and boundary controls, nonlinear dynamics, and regular and irregular geometries (Fig. 2). This diversity allows us to examine the three design choices isolated above: *where to search*, *how to search*, and *how candidates are evaluated*. The appendices provide related work (Appendix A), solver-verified visualizations (Appendix B), complete benchmark specifications (Appendix C), additional algorithmic designs (Appendix D), representation, search, and evaluator details (Appendix E), sensitivity and variability analyses (Appendix F), numerical verification and cost analyses (Appendix G), and limitations (Appendix H).

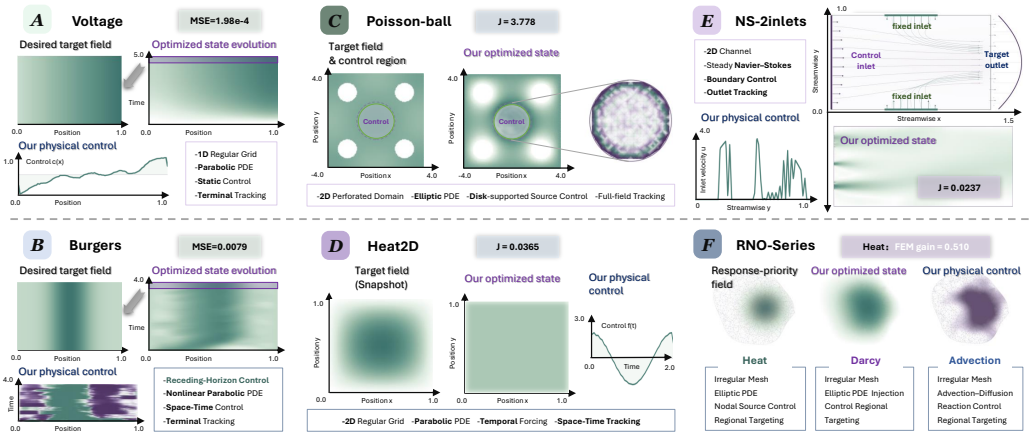


Figure 2: **PDE-control benchmark overview and representative optimized solutions. (A–E)** Regular-grid benchmarks spanning static and receding-horizon control, distributed and boundary actuation, and terminal, full-field, and space–time objectives. Each panel visualizes the target, optimized state, and physical control where applicable. **(F)** Irregular-mesh benchmarks for Heat, Darcy, and Advection, illustrating the corresponding meshes, control settings, and a representative optimized Heat solution.

### 4.1 Experimental Setting

All learned representations and operator surrogates are trained offline and frozen before test-task optimization. Table 1 summarizes the strongest numerically validated quality–efficiency results from the frozen experiments, while Secs. 4.3–4.5 isolate the effects of representation, optimizer, and evaluator by holding the remaining choices fixed. On the regular PDE-OP/BPN benchmarks, every reported final control is numerically evaluated. Surrogate-guided searches retain a short candidate list for numerical verification (top-5 for Voltage, Heat2D, and Poisson; top-3 for NS2inlets), while Burgers executes each selected control block with the numerical PDE dynamics before replanning. Historical PDE-OP and target-specific BPN results are included as reference baselines. The derivative-free searches use dual annealing (DA) (Xiang et al., 1997), the covariance matrix adaptation evolution strategy (CMA-ES) (Hansen & Ostermeier, 2001), and deep active optimization with neural-surrogate-guided tree exploration (DANTE) (Wei et al., 2025); DANTE-GSA is our variant with a generalized-simulated-annealing proposal (Appendix D.2).

For the irregular-geometry benchmarks, RNO-Opt and the FEM adjoint have comparable end-to-end runtimes, so the headline comparison uses matched wall time with the frozen PDE-specific representation for each method. Final objectives are recomputed by FEM, while fixed-representation and count-matched comparisons are reported separately in the following studies. Wall times are compared within each benchmark suite rather than across suites.

### 4.2 Main Results: Quality–Efficiency Trade-offs

Table 1 shows a consistent pattern across the five regular-grid benchmarks: at least one of our configurations lies on the empirical quality–time Pareto frontier of the compared methods. Exact latent search reaches near-physics solution quality at substantially lower online cost. On Heat2D, it obtains an objective of $0.03666$ versus $0.03563$ for the adjoint method while reducing optimization time from $136.78$ to $1.85\,$s. On Burgers, it reaches an MSE of $6.04 \times 10^{-3}$ in $5.12\,$s, compared with $4.14 \times 10^{-3}$ in $201.08\,$s for POD-MPC. Thus, restricting optimization to learned control geometry can substantially reduce online search cost while retaining high-quality numerical solutions.

Surrogate evaluation can extend this trade-off when PDE evaluation is expensive. On Voltage, surrogate-guided search reaches a solver-verified MSE of $6.14 \times 10^{-3}$ after $0.30\,$s of proxy search and five PDE confirmations, compared with $5.24 \times 10^{-3}$ in $2.65\,$s from 197 numerical evaluations. On NS2inlets, it reaches a verified objective of $0.03322$ after $0.65\,$s of proxy search and three FEM confirmations, while exact latent search reaches $0.02718$ in $275.97\,$s. Burgers, Heat2D, and Poisson

Table 1: **Solver-verified solution quality and online cost.** Panel (a) reports the strongest validated results on the PDE-OP/BPN benchmarks, with the optimizer used by our method shown in each cell. Panel (b) reports the irregular-geometry comparison under matched wall time. All reported quality values are evaluated by the corresponding numerical solver; additional numerical checks for surrogate-guided search are listed where applicable.

**(a) PDE-OP and BPN benchmarks.**

| Benchmark | Classical / physics | Prior learned | Ours: exact | Ours: surrogate |
|---|---|---|---|---|
| *Tracking MSE* ↓ | | | | |
| Voltage | $9.37\times10^{-4}$<br>Direct; 17.86 s | $2.17\times10^{-2}$<br>PDE-OP; 0.0019 s | $5.24\times10^{-3}$<br>DANTE; 2.65 s / 197 solver | $6.14\times10^{-3}$<br>DANTE; 0.30 s† / 197P+5S |
| Burgers | $4.14\times10^{-3}$<br>POD-MPC; 201.08 s | $4.43\times10^{-2}$<br>PDE-OP; 0.55 s | $6.04\times10^{-3}$<br>CMA-ES; 5.12 s / 730 solver | $8.28\times10^{-3}$<br>DA; 11.46 s / 730 proxy |
| *Control objective* ↓ | | | | |
| Poisson | 4.756<br>Adjoint-FEM; 41.11 s | 6.648<br>BPN; 412.02 s | 4.848<br>DA; 0.48 s / 200 FEM | 4.916<br>CMA-ES; 1.80 s† / 193P+5F |
| Heat2D | 0.03563<br>Adjoint; 136.78 s | 0.21985<br>BPN; 545.25 s | 0.03666<br>DA; 1.85 s / 500 solver | 0.03589<br>DANTE; 19.51 s / 497P+5S |
| NS2inlets | 0.02354<br>Adjoint-FEM; 146.28 s | 0.13281<br>BPN-500; 15.75 h | 0.02718<br>DA; 275.97 s / 200 FEM | 0.03322<br>DANTE; 0.65 s / 193P+3F |

**(b) Irregular geometry: FEM gain ↑, matched wall time.**

| Benchmark | FEM adjoint | RNO direct | Ours: exact latent | Ours: surrogate latent |
|---|---|---|---|---|
| Irreg. Heat | 0.2555<br>0.208 s | 0.1231<br>DCT; 0.223 s | 0.1564<br>DCT→SWAE; 0.227 s | 0.1507<br>DCT→SWAE; 0.223 s |
| Darcy | 0.3467<br>0.318 s | 0.2002<br>DCT; 0.328 s | 0.2215<br>DCT→SWAE; 0.333 s | 0.2310<br>DCT→SWAE; 0.331 s |
| Advection | 0.1973<br>0.494 s | 0.1588<br>DCT; 0.497 s | 0.1137<br>GFT→SWAE; 0.512 s | 0.1328<br>GFT→SWAE; 0.496 s |

†For the marked surrogate rows, the displayed time is proxy-search time; the listed numerical confirmations are additional. P/S/F denote proxy, solver, and FEM evaluations, respectively.

*Irregular-suite optimizer attribution.* With the PDE-specific latent representation fixed (DCT→SWAE for Heat/Darcy and GFT→SWAE for Advection), a separate matched 65-call study identifies CMA-ES as the strongest optimizer on Heat and Darcy, and DANTE-GSA on Advection. These optimizer-attribution results are not combined numerically with the large-scale matched-wall-time values in panel (b).

instead favor exact evaluation at the tested budgets. On the irregular suite, where RNO gradient optimization and the FEM adjoint have comparable online runtimes, the FEM adjoint remains strongest in mean gain, while latent search improves the learned RNO baseline on Heat and Darcy under matched wall time; Advection instead favors direct DCT. Overall, the results show favorable quality–efficiency trade-offs across distinct PDE-control regimes, with surrogate evaluation providing further savings when numerical candidate evaluation is costly.

### 4.3 Where to Search Matters

Changing the search representation alone produces some of the largest solver-verified gains in the study. In Figure 3A, we fix the optimizer, evaluator, and number of objective calls and vary only the coordinates exposed to search. For the more expensive BPN benchmarks, the comparison uses surrogate-guided search followed by numerical verification. SWAE achieves the lowest mean solver-verified objective on all five regular benchmarks, reducing the final error relative to raw control-space search by $21\%$ on Voltage and up to $98.5\%$ on NS2inlets. DCT and PCA recover a substantial fraction of these gains, showing that structured low-dimensional coordinates already simplify the search; the learned manifold provides the strongest overall search support.

The irregular-mesh experiments separate structured reduction from the learned manifold. We first project the high-dimensional nodal controls into DCT/PCA/GFT coordinates and then learn SWAE within the reduced space. Under the common 80-call RNO budget in Figure 3B, adding SWAE to the PDE-preferred representation increases FEM gain from $0.107$ to $0.131$ on Heat and from $0.139$ to $0.172$ on Darcy using DCT, and from $0.0556$ to $0.0669$ on Advection using GFT. The preferred

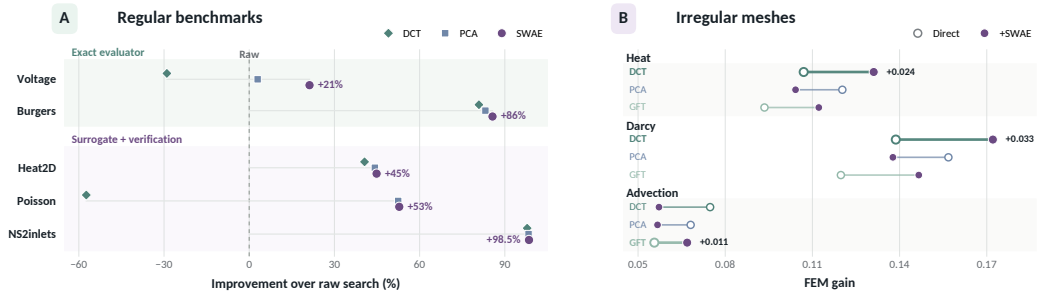


Figure 3: **Where-to-search attribution under fixed search budgets. (A)** Regular benchmarks with the optimizer, evaluator, and benchmark-specific objective-call budget fixed, showing improvement over raw control-space search for DCT, PCA, and SWAE representations. **(B)** Irregular-mesh benchmarks under a fixed 80-call RNO budget, comparing direct search in DCT/PCA/GFT coefficient spaces with SWAE search built on the same upstream representation. Regular-benchmark outcomes are solver verified; irregular-benchmark outcomes report FEM gain.

Table 2: **How-to-search attribution with the search representation fixed.** Panel (a) fixes the SWAE manifold on the regular benchmarks and varies only the derivative-free optimizer; entries are mean solver-verified objectives (lower is better), including benchmark-specific numerical verification after surrogate-guided search. Panel (b) fixes DCT→SWAE for irregular Heat and Darcy and GFT→SWAE for Advection; entries report exact/surrogate FEM gain (higher is better). The irregular comparison uses 65 evaluator calls per search: 65 FEM calls in exact mode or 65 RNO calls followed by one FEM verification in surrogate mode.

**(a) Regular benchmarks: fixed SWAE manifold.**

| Benchmark | Evaluator | DA | CMA-ES | DANTE | DANTE-GSA | Calls |
|---|---|---|---|---|---|---|
| Voltage | Exact | $7.12\times10^{-3}$ | $5.65\times10^{-3}$ | $\mathbf{5.24\times10^{-3}}$ | $1.66\times10^{-2}$ | 197 |
| Voltage | Surrogate | $7.04\times10^{-3}$ | $6.18\times10^{-3}$ | $\mathbf{6.14\times10^{-3}}$ | $1.66\times10^{-2}$ | 197 |
| Burgers | Exact | $1.03\times10^{-2}$ | $\mathbf{6.04\times10^{-3}}$ | $9.95\times10^{-3}$ | $9.92\times10^{-3}$ | 730 |
| Burgers | Surrogate | $\mathbf{8.28\times10^{-3}}$ | $8.54\times10^{-3}$ | $9.18\times10^{-3}$ | $1.22\times10^{-2}$ | 730 |
| Heat2D | Surrogate | $3.626\times10^{-2}$ | $3.591\times10^{-2}$ | $\mathbf{3.589\times10^{-2}}$ | $5.067\times10^{-2}$ | 497 |
| Poisson | Surrogate | 4.930 | **4.916** | 5.127 | 5.350 | 193 |
| NS2inlets | Surrogate | $3.365\times10^{-2}$ | $3.976\times10^{-2}$ | $\mathbf{3.322\times10^{-2}}$ | $4.948\times10^{-2}$ | 193 |

**(b) Irregular meshes: fixed native latent representation.**

| Benchmark | Fixed representation | DA | CMA-ES | DANTE | DANTE-GSA |
|---|---|---|---|---|---|
| Heat | DCT→SWAE | 0.3458 / 0.3431 | **0.4234/0.4123** | 0.3091 / 0.3085 | 0.3972 / 0.3904 |
| Darcy | DCT→SWAE | 0.4837 / 0.4663 | **0.5881/0.5581** | 0.4309 / 0.4275 | 0.5511 / 0.5317 |
| Advection | GFT→SWAE | 0.6087 / 0.5976 | 0.6295 / 0.6057 | 0.5876 / 0.5816 | **0.6299/0.6225** |

Panel (b) reports exact / surrogate FEM gain in each cell.

upstream basis is itself PDE dependent: DCT is strongest for the two diffusion-dominated problems, whereas graph-spectral coordinates are strongest for Advection. Together, these results show that the coordinates and support exposed to the optimizer substantially affect the control reached under the same online budget.

### 4.4 How to Search Is Task Dependent

With the search representation fixed, optimizer choice still changes the solution reached, but the preferred search dynamics vary across PDEs. Table 2(a) compares four derivative-free optimizers on the same frozen SWAE manifold. DANTE performs best on Voltage under both exact and surrogate evaluation and also leads on Heat2D and NS2inlets, whereas CMA-ES performs best on exact Burgers and Poisson. Burgers further shows that the ranking can change with the evaluator: CMA-ES gives

Table 3: **Evaluator attribution.** With the representation and optimizer fixed, we compare exact and surrogate candidate evaluation; all reported quality is recomputed numerically after benchmark-specific final verification.

**(a) Regular: objective ↓, matched search calls**

| Benchmark | Exact | Surrogate |
|---|---|---|
| Voltage | $6.90\times10^{-3}$ (3.07 s) | $7.04\times10^{-3}$ (0.27 s) |
| Burgers | $1.95\times10^{-2}$ (0.67 s) | $2.20\times10^{-2}$ (1.59 s) |
| Heat2D | $3.666\times10^{-2}$ (1.85 s) | $3.645\times10^{-2}$ (13.24 s) |
| Poisson | 4.848 (0.48 s) | 4.930 (0.90 s) |
| NS2inlets | $2.718\times10^{-2}$ (275.97 s) | $3.455\times10^{-2}$ (7.44 s) |

**(b) Irregular: FEM gain ↑, matched time**

| Benchmark | FEM | RNO |
|---|---|---|
| Heat<br>DCT→SWAE | **0.1564**<br>0.227 s | 0.1507<br>0.223 s |
| Darcy<br>DCT→SWAE | 0.2215<br>0.333 s | **0.2310**<br>0.331 s |
| Advection<br>GFT→SWAE | 0.1137<br>0.512 s | **0.1328**<br>0.496 s |

the lowest error under exact rollout, while DA leads under surrogate-guided search. The magnitude of these differences is also problem dependent: relative to DA, DANTE reduces the exact Voltage error by about $26\%$, while CMA-ES reduces the exact Burgers error by about $41\%$; the leading methods on Heat2D and Poisson are much closer.

The irregular benchmarks show the same task dependence after fixing their preferred latent parameterization. As shown in Table 2(b), CMA-ES leads on both Heat and Darcy with DCT→SWAE, whereas DANTE-GSA leads on Advection with GFT→SWAE; this pattern is unchanged between exact and surrogate evaluation. Together with the representation results above, these findings separate two complementary design choices: the learned representation defines the control region exposed to search, while the optimizer determines how that region is explored.

### 4.5 Surrogate Evaluation Helps When PDE Solves Dominate

Surrogate evaluation is most valuable when candidate scoring dominates the online cost. Table 3(a) fixes the search manifold, optimizer, and candidate-call budget and changes only the evaluator. On Voltage, DeepONet retains nearly the same solver-verified accuracy after Top-5 verification while reducing search time from $3.07$ to $0.27$ s. On NS2inlets, surrogate-guided search with three FEM confirmations reaches $0.0345$ in $7.44$ s, compared with $0.0272$ in $275.97$ s for exact search. Burgers, Heat2D, and Poisson instead favor exact evaluation at the tested budgets. Thus, surrogate evaluation provides the clearest benefit when numerical PDE solves dominate the online budget.

Search geometry further affects surrogate-guided selection. In the Voltage Top-5 replay, numerical reranking changes the selected candidate in $77.8\%$ of Raw runs but only $8.3\%$ of SWAE runs, indicating substantially less need for numerical correction on the learned manifold. The irregular benchmarks show the complementary cost-to-search effect under matched wall time: on Heat, $127$ surrogate evaluations plus one FEM verification nearly match the gain from about $99$ FEM evaluations, while on Advection the surrogate search evaluates about $748$ rather than $183$ candidates and improves the gain from $0.1137$ to $0.1328$. Together, these results show that surrogate evaluation is useful when saved PDE solves can support broader search, while numerical verification retains the final decision.

## 5 Conclusion

We introduced a control-centric view of repeated PDE optimization: past optimized controls provide reusable information not only about what good solutions look like, but also about where future optimization should search. Our framework learns this information as reusable low-dimensional search geometry and decouples the resulting representation from both the derivative-free optimizer and the candidate evaluator. Across regular and irregular PDE-control benchmarks, controlled experiments show that the choice of search space materially changes the attainable solution, preferred search dynamics remain task dependent, and surrogate evaluation is most valuable when numerical PDE solves dominate online cost. Together, these results establish where to search, how to search, and how to evaluate as distinct design choices in repeated PDE optimization. Learning reusable control geometry therefore offers an alternative to direct control prediction: learn where to search, then optimize the objective at hand.

### Reproducibility Statement

We provide benchmark definitions, control parameterizations, numerical solver and discretization details, learned-representation and surrogate training settings, optimization budgets, and evaluation procedures in the main paper and appendix. We additionally document the sample counts, search seeds, and fixed experimental conditions used in the controlled attribution studies. Final reported controls are evaluated with the corresponding numerical PDE solvers, with additional numerical verification and solver checks reported in the appendix. Code, configurations, and scripts for reproducing the experiments will be publicly released upon acceptance.

### AI Use Statement

Generative AI tools were used to assist with code development and manuscript editing. All scientific claims and conclusions are based on the reported experiments and numerical verification. AI-assisted code and manuscript content were manually reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

## References

Abdolmehdi Behroozi, Chaopeng Shen, and Daniel Kifer. Sensitivity-constrained Fourier neural operators for forward and inverse problems in parametric differential equations. *arXiv preprint arXiv:2505.08740*, 2025. URL https://arxiv.org/abs/2505.08740.

Lorenz T. Biegler, Omar Ghattas, Matthias Heinkenschloss, David Keyes, and Bart van Bloemen Waanders (eds.). *Real-Time PDE-Constrained Optimization*. Society for Industrial and Applied Mathematics, 2007. doi: 10.1137/1.9780898718935.

George Biros and Omar Ghattas. Parallel Lagrange–Newton–Krylov–Schur methods for PDE-constrained optimization. part II: The Lagrange–Newton solver and its application to optimal control of steady viscous flows. *SIAM Journal on Scientific Computing*, 27(2):714–739, 2005. doi: 10.1137/S1064827502415661.

Gabriele Boncoraglio and Charbel Farhat. Piecewise-global nonlinear model order reduction for PDE-constrained optimization in high-dimensional parameter spaces. *SIAM Journal on Scientific Computing*, 44(4):A2176–A2203, 2022. doi: 10.1137/21M1435343.

Ze Cheng, Zhuoyu Li, Xiaoqiang Wang, Jianing Huang, Zhizhou Zhang, Zhongkai Hao, and Hang Su. Accelerating PDE-constrained optimization by the derivative of neural operators. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 10090–10105. PMLR, 2025. URL https://proceedings.mlr.press/v267/cheng25f.html.

Michael B. Giles and Niles A. Pierce. An introduction to the adjoint approach to design. *Flow, Turbulence and Combustion*, 65:393–415, 2000. doi: 10.1023/A:1011430410075.

Xindi Gong, Dingcheng Luo, Thomas O'Leary-Roseberry, Ruanui Nicholson, and Omar Ghattas. Shape derivative-informed neural operators with application to risk-averse shape optimization. *arXiv preprint arXiv:2603.03211*, 2026. URL https://arxiv.org/abs/2603.03211.

Max D. Gunzburger. *Perspectives in Flow Control and Optimization*, volume 5 of *Advances in Design and Control*. SIAM, 2003. doi: 10.1137/1.9780898718720.

Yusuf Guven, Vincenzo Di Vito, and Ferdinando Fioretto. Learning to solve optimization problems constrained with partial differential equations. *arXiv preprint arXiv:2509.24573*, 2025. URL https://arxiv.org/abs/2509.24573.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi: 10.1162/106365601750190398.

Zhongkai Hao, Chengyang Ying, Hang Su, Jun Zhu, Jian Song, and Ze Cheng. Bi-level physics-informed neural networks for PDE constrained optimization using Broyden's hypergradients. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.07075.

Michael Hinze, René Pinnau, Michael Ulbrich, and Stefan Ulbrich. *Optimization with PDE Constraints*, volume 23 of *Mathematical Modelling: Theory and Applications*. Springer, 2009. doi: 10.1007/978-1-4020-8839-1.

Jiaheng Hu, Julian Whitman, and Howie Choset. GLSO: Grammar-guided latent space optimization for sample-efficient robot design automation. In *Proceedings of the 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 1321–1331. PMLR, 2023. URL https://proceedings.mlr.press/v205/hu23c.html.

Rakhoon Hwang, Jae Yong Lee, Jin Young Shin, and Hyung Ju Hwang. Solving PDE-constrained control problems using operator learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4):4504–4512, 2022. doi: 10.1609/aaai.v36i4.20373.

Soheil Kolouri, Phillip E. Pope, Charles E. Martin, and Gustavo K. Rohde. Sliced Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1804.01947.

Jeffrey Larson, Matt Menickelly, and Stefan M. Wild. Derivative-free optimization methods. *Acta Numerica*, 28:287–404, 2019. doi: 10.1017/S0962492919000060.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.08895.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021. doi: 10.1038/s42256-021-00302-5.

Dingcheng Luo, Thomas O'Leary-Roseberry, Peng Chen, and Omar Ghattas. Efficient PDE-constrained optimization under high-dimensional uncertainty using derivative-informed neural operators. *SIAM Journal on Scientific Computing*, 47(4):C899–C931, 2025. doi: 10.1137/23M157956X.

Andrea Manzoni, Alfio Quarteroni, and Sandro Salsa. *Optimal Control of Partial Differential Equations: Analysis, Approximation, and Applications*, volume 207 of *Applied Mathematical Sciences*. Springer, 2021. doi: 10.1007/978-3-030-77226-0.

Federico Negri, Gianluigi Rozza, Andrea Manzoni, and Alfio Quarteroni. Reduced basis method for parametrized elliptic optimal control problems. *SIAM Journal on Scientific Computing*, 35(5): A2316–A2340, 2013. doi: 10.1137/120894737.

Thomas O'Leary-Roseberry, Peng Chen, Umberto Villa, and Omar Ghattas. Derivative-informed neural operator: An efficient framework for high-dimensional parametric derivative learning. *Journal of Computational Physics*, 496:112555, 2024. doi: 10.1016/j.jcp.2023.112555.

John W. Pearson. PDE-constrained optimization models for scientific processes. *Proceedings in Applied Mathematics and Mechanics*, 18(1):e201800253, 2018. doi: 10.1002/pamm.201800253.

Elizabeth Qian, Martin Grepl, Karen Veroy, and Karen Willcox. A certified trust region reduced basis approach to PDE-constrained optimization. *SIAM Journal on Scientific Computing*, 39(5): S434–S460, 2017. doi: 10.1137/16M1081981.

Yuan Qiu, Nolan Bridges, and Peng Chen. Derivative-enhanced deep operator network. In *Advances in Neural Information Processing Systems*, volume 37, pp. 20945–20981, 2024. doi: 10.52202/079017-0660.

David Sondak and Pavlos Protopapas. Learning a reduced basis of dynamical systems using an autoencoder. *Physical Review E*, 104(3):034202, 2021. doi: 10.1103/PhysRevE.104.034202.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018. URL https://arxiv.org/abs/1711.01558.

Michael Ulbrich and Boris Vexler. OPTE special issue on PDE-constrained optimization. *Optimization and Engineering*, 22(4):1985–1987, 2021. doi: 10.1007/s11081-021-09689-x.

Ye Wei, Bo Peng, Ruiwen Xie, Yangtao Chen, Yu Qin, Peng Wen, Stefan Bauer, Po-Yen Tung, and Dierk Raabe. Deep active optimization for complex systems. *Nature Computational Science*, 5(9): 801–812, 2025. doi: 10.1038/s43588-025-00858-x.

Y. Xiang, D. Y. Sun, W. Fan, and X. G. Gong. Generalized simulated annealing algorithm and its application to the Thomson model. *Physics Letters A*, 233(3):216–220, 1997. doi: 10.1016/S0375-9601(97)00474-X.

Boyuan Yao, Dingcheng Luo, Lianghao Cao, Nikola Kovachki, Thomas O'Leary-Roseberry, and Omar Ghattas. Derivative-informed Fourier neural operator: Universal approximation and applications to PDE-constrained optimization. *arXiv preprint arXiv:2512.14086*, 2025. URL https://arxiv.org/abs/2512.14086.

## A Related Work

**PDE-constrained optimization and model reduction.** PDE-constrained optimization has a mature foundation in first-order optimality systems, adjoint sensitivities, and reduced- or full-space nonlinear optimization, including scalable Newton–Krylov and sequential quadratic programming methods for large discretized PDEs (Giles & Pierce, 2000; Hinze et al., 2009; Biros & Ghattas, 2005; Biegler et al., 2007). Recent overview and special-issue perspectives reflect the breadth of its scientific applications (Pearson, 2018; Ulbrich & Vexler, 2021). For repeated or parametrized problems, POD and reduced-basis methods provide low-dimensional approximations of the PDE or its optimality system, often through an offline–online decomposition; certified variants add error estimators or trust-region mechanisms to retain control of the high-fidelity problem (Negri et al., 2013; Qian et al., 2017). Nonlinear projection and active-space methods can also reduce high-dimensional design problems (Boncoraglio & Farhat, 2022). When derivatives are unavailable, derivative-free optimization remains a general black-box alternative, but its repeated evaluations make the search dimension especially important in simulation-based settings (Larson et al., 2019). Our work is closest to these many-query and dimension-reduction approaches, but reduces a different object: it learns a reusable nonlinear search geometry directly from high-quality controls produced by related tasks, while leaving the PDE evaluator and derivative-free optimizer replaceable.

**Learned PDE surrogates and derivative-aware operator learning.** Neural operators such as DeepONet and the Fourier neural operator (FNO) learn maps between function spaces, enabling one learned model to approximate a family of parametric PDE solutions at lower online cost than repeated numerical solves (Lu et al., 2021; Li et al., 2021). This capability has been used directly in PDE-constrained control, where the learned solution operator acts as a differentiable surrogate (Hwang et al., 2022). For optimization, however, accurate forward predictions do not by themselves guarantee useful sensitivities. Derivative-Informed Neural Operators (DINO) and Derivative-Enhanced DeepONet incorporate derivative information into operator training to improve parameter-sensitivity prediction, including in data-limited regimes (O'Leary-Roseberry et al., 2024; Qiu et al., 2024). Recent work specializes these ideas for optimization: MR-DINO targets PDE-constrained optimization under high-dimensional uncertainty, SC-FNO regularizes parameter sensitivities, and Opt-RNO combines learned operator derivatives with periodic numerical-solver correction (Luo et al., 2025; Behroozi et al., 2025; Cheng et al., 2025). DIFNO and Shape-DINO further develop joint operator–Fréchet-derivative approximation for PDE-constrained and shape optimization (Yao et al., 2025; Gong et al., 2026). Our framework is complementary to this line of work: a neural operator may be used as a fast evaluator, but it is not required to determine the optimization trajectory or to supply gradients. The learned search geometry and the evaluator are separate interfaces, and selected controls are checked with the numerical solver.

**Latent representations and learned search spaces.** Linear bases such as PCA and POD have long exposed low-dimensional structure in high-dimensional physical systems; autoencoders extend this idea to nonlinear manifolds of states and dynamics (Sondak & Protopapas, 2021). In adjacent design domains, latent-space optimization shows how a decoder can turn a difficult design problem

into continuous low-dimensional search and be reused across downstream tasks (Hu et al., 2023). For repeated optimization, reconstruction quality alone is not enough: a standard autoencoder leaves the distribution of latent codes largely unconstrained, so regions between or beyond observed codes have no prescribed search interpretation. Distribution-regularized autoencoders address this issue by matching the aggregated encoded distribution to a prior (Tolstikhin et al., 2018). We use the sliced-Wasserstein autoencoder (SWAE), which performs this matching with sliced-Wasserstein distance and supports readily sampleable priors without adversarial training or an explicit prior likelihood (Kolouri et al., 2019). This choice supplies a low-dimensional coordinate system that can be searched repeatedly and decoded deterministically into candidate controls. Because the representation is trained on high-quality controls from related solved tasks, its learned geometry summarizes reusable optimization experience; the makeup of that control dataset therefore defines the task family over which transfer should be expected.

**Positioning.** Our framework separates three interfaces: *where to search*, *how to search*, and *how to evaluate*. Historical optimized controls define a reusable low-dimensional search manifold, a derivative-free method searches its latent coordinates, and either the numerical PDE solver or a learned surrogate evaluates candidates. New tasks are still solved through optimization rather than direct control prediction, and the surrogate is optional rather than a source of required gradients. The resulting separation lets the representation change the search geometry while keeping the optimizer and evaluator replaceable.

## B SOLVER-VERIFIED FIELD AND CONTROL VISUALIZATIONS

The aggregate metrics in the main text summarize final solution quality, while the visualizations below show the corresponding field and control structure for representative cases from all eight benchmarks. For the regular benchmarks, Figs. 4–8 show the target, optimized response, and control, together with state or terminal error where applicable. For the irregular benchmarks, Figs. 9–11 show the response-priority field, optimized state, and control directly on the native mesh. Method labels and numerical annotations are retained in each figure so that the spatial or temporal solutions can be read alongside their quantitative outcomes.

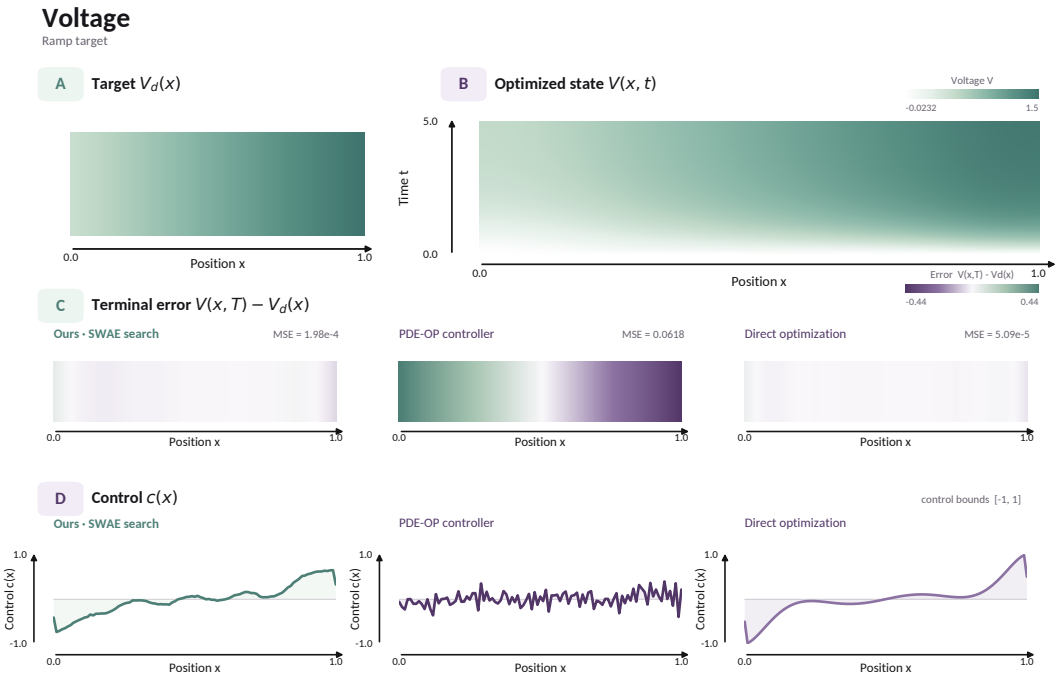


Figure 4: **Voltage: ramp target.** Target voltage, optimized trajectory, terminal error, and spatial control.

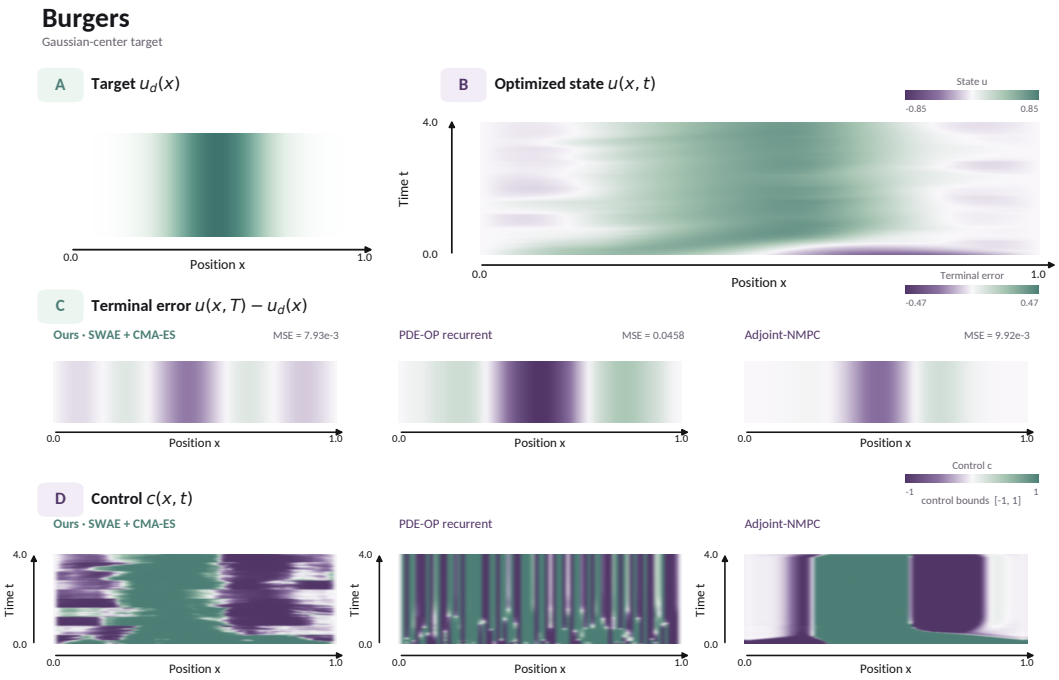


Figure 5: **Burgers: central Gaussian target.** Target state, optimized trajectory, terminal error, and space–time control.

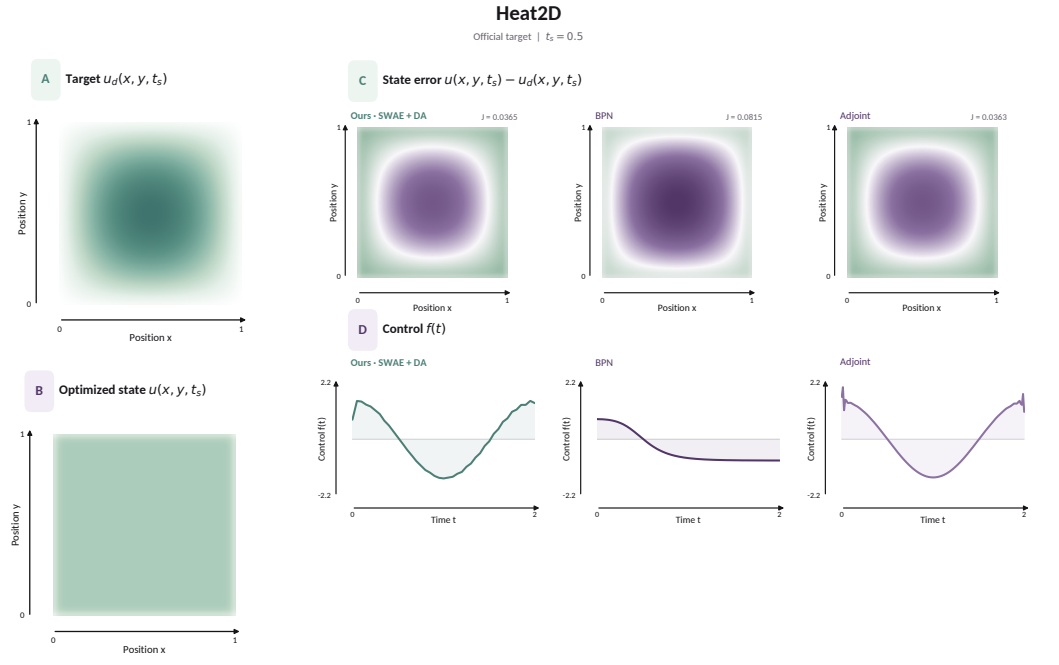


Figure 6: **Heat2D: official target.** Target and optimized temperature at $t_s = 0.5$, state error, and temporal forcing.

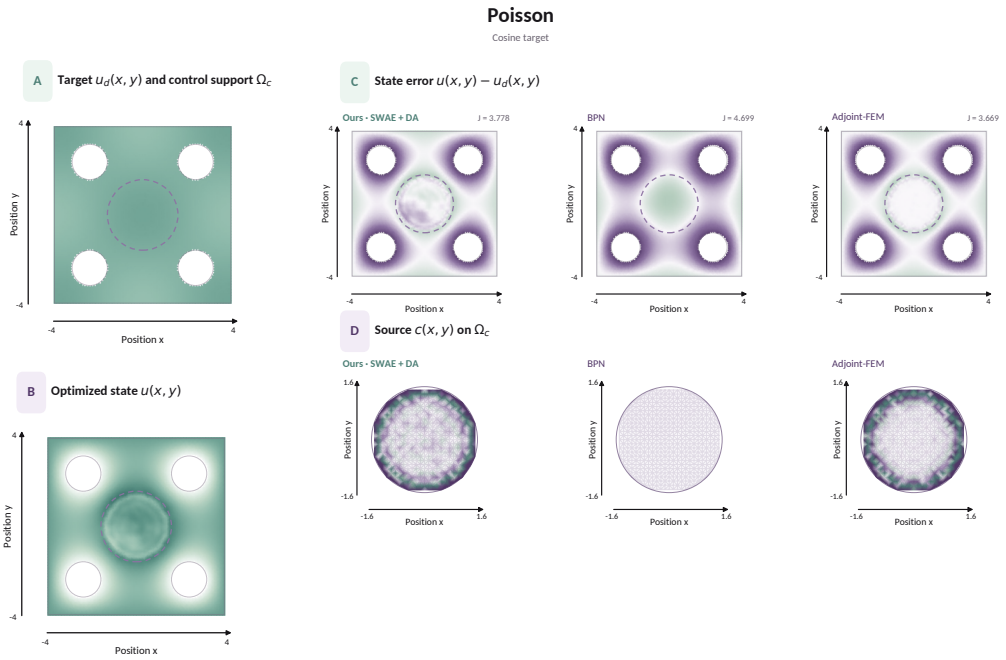


Figure 7: **Poisson: cosine target.** Target and control support, optimized state, state error, and distributed source.

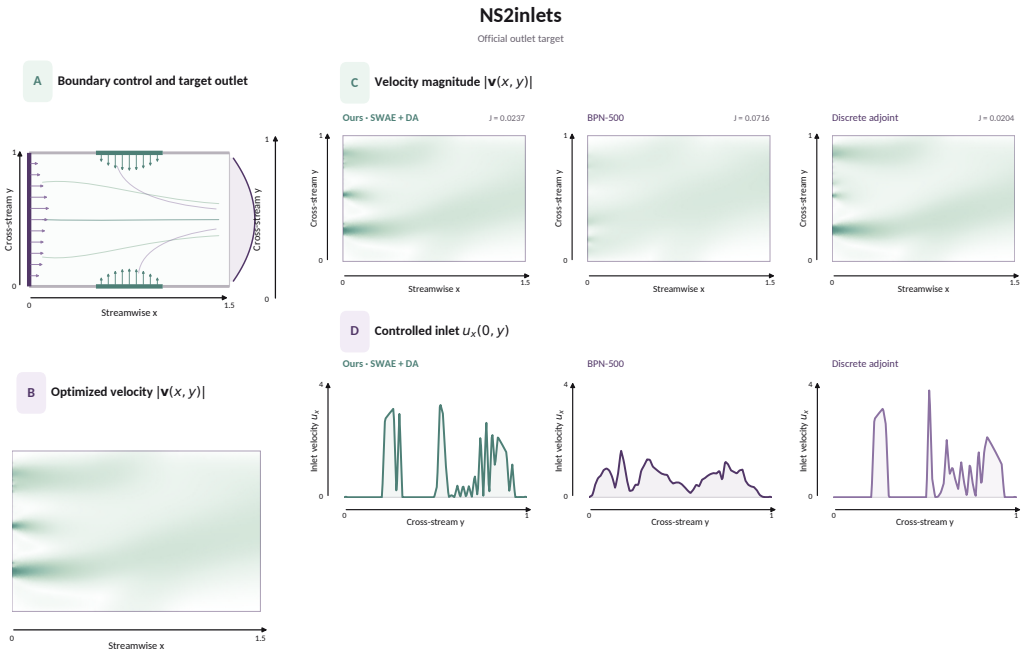


Figure 8: **NS2inlets: official outlet target.** Boundary conditions, optimized velocity magnitude, and controlled inlet profiles.

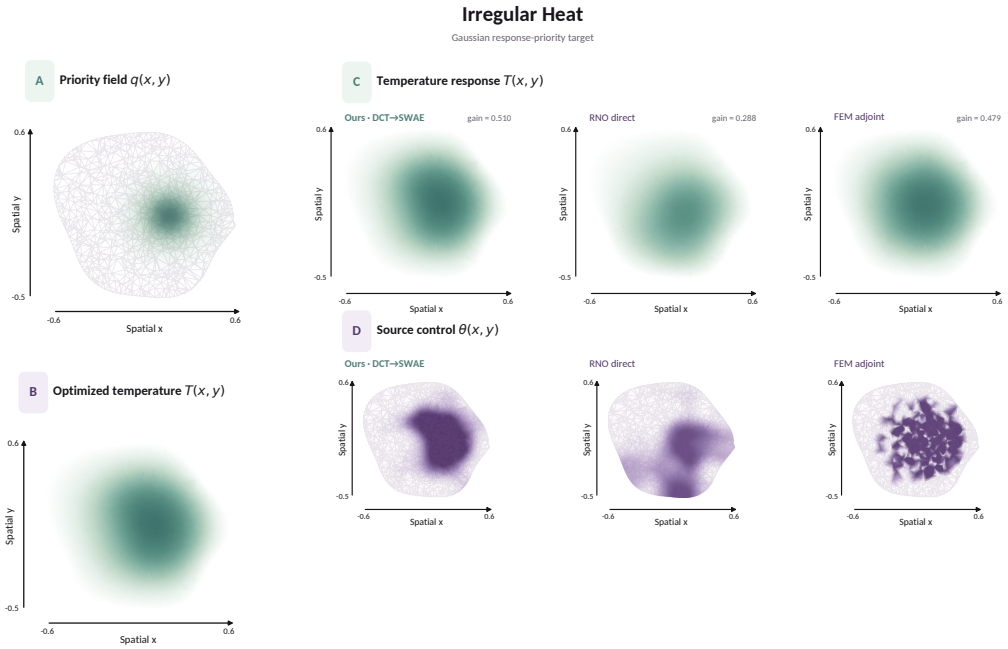


Figure 9: **Irregular Heat.** Gaussian response-priority field, optimized temperature, and source control on the native mesh.

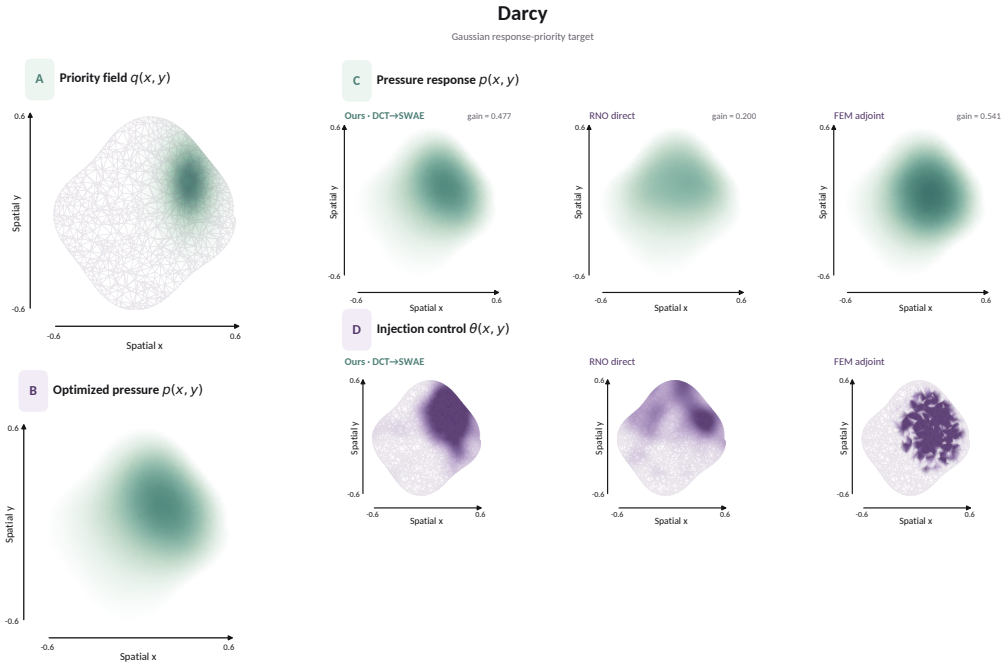


Figure 10: **Darcy.** Gaussian response-priority field, optimized pressure, and injection control on the native mesh.

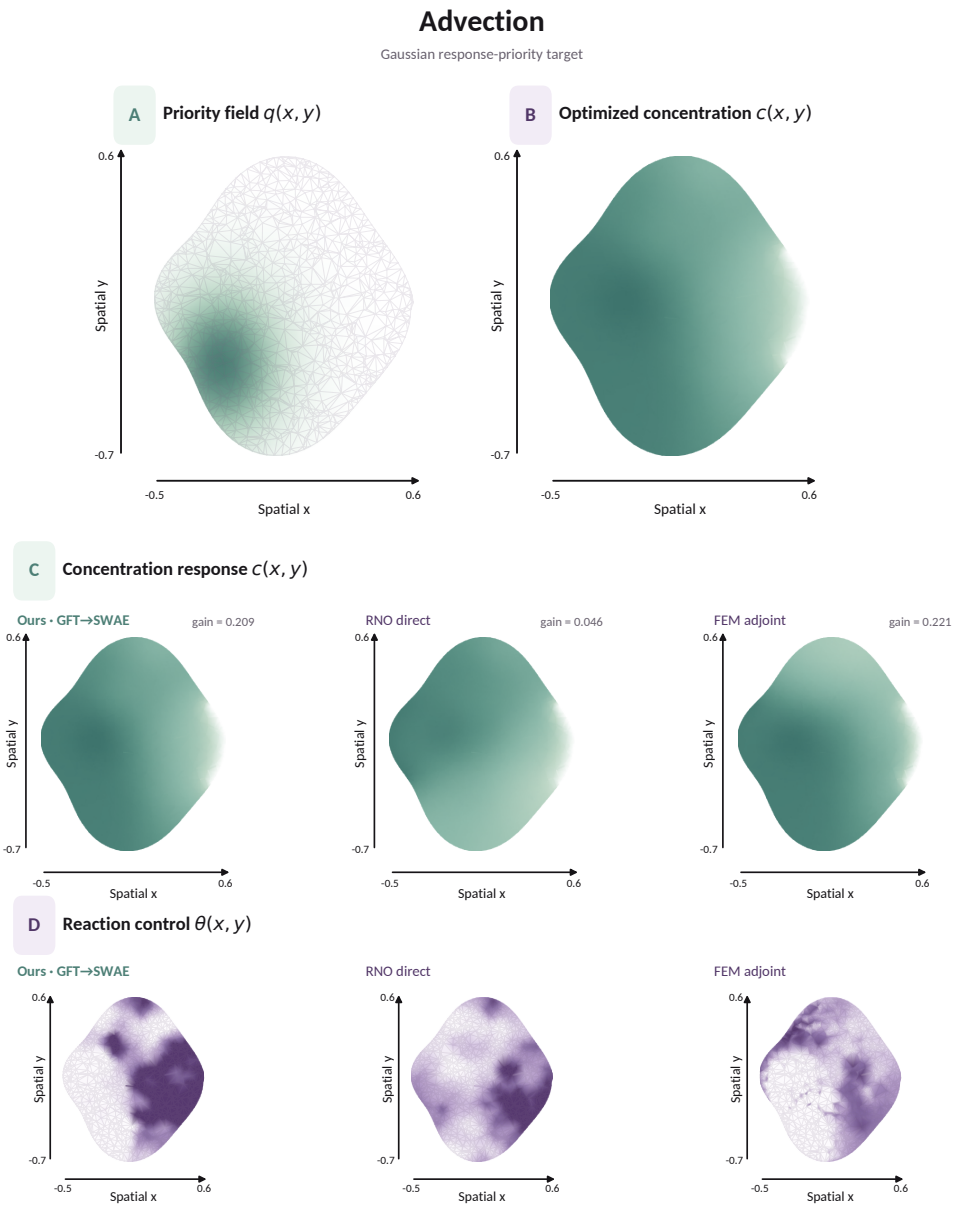


Figure 11: **Advection.** Gaussian response-priority field, optimized concentration, and reaction control on the native mesh.

# C Benchmark Suite Construction, Data Provenance, and Full Problem Specifications

The experimental suite is itself a substantial part of our evaluation infrastructure. We start from problem families and public code released with PDE-OP (the learned PDE-constrained optimization framework) (Guven et al., 2025), bi-level physics-informed neural networks (BPN) (Hao et al., 2023), and Opt-RNO (Cheng et al., 2025), but the final benchmark used in this paper is not a concatenation of those repositories. We reconstruct or retain a numerical ground-truth solver for every problem, regenerate the offline data used by the control representations and neural operators, broaden the target and geometry distributions, and evaluate every reported final control with a numerical PDE solver. The resulting suite contains eight control problems, multiple target or geometry conditions per problem, static and trajectory controls, regular and irregular meshes, and both exact and surrogate-guided online optimization. Taken together, these extensions form a solver-verified multi-task benchmark that can be used independently of the particular search method proposed in this paper.

**Numerical terminology.** We use Crank–Nicolson (CN) time stepping, finite differences (FD), and the finite element method (FEM); CN–FD denotes CN time stepping with an FD spatial discretization. P1 and P2 denote continuous piecewise-linear and piecewise-quadratic finite elements, respectively. Taylor–Hood elements use P2 velocity and P1 pressure. Mean squared error (MSE) is the mean of squared prediction or tracking errors. Independently and identically distributed (IID) test samples follow the training distribution, with disjoint geometries in the irregular suite. Throughout this appendix, an exact evaluator is the fixed numerical solver, rather than an analytic solution. Top-$K$ verification reevaluates the $K$ best surrogate-ranked controls with this solver and selects the best numerical objective.

## C.1 From three public repositories to a unified benchmark suite

Table 4 summarizes the lineage of the suite. The three source repositories play different roles. PDE-OP supplies the Voltage and Burgers control families and their learned-controller and classical reference setting. BPN supplies the Heat2D, Poisson, and two-inlet Navier–Stokes formulations and the corresponding BPN baselines. Opt-RNO supplies the reference neural operator (RNO) evaluator and irregular-operator optimization setting, but we do not reuse its released Microreactor2D, Fuelcell2D, or Inductor2D datasets. Instead, we construct a new irregular-control suite consisting of Heat, Darcy, and Advection problems, generate the FEM data ourselves, and train RNO on these newly generated samples.

Table 4: **Lineage of the final benchmark suite.** "Inherited" denotes the problem family or methodological setting taken from the public source; "rebuilt or extended" lists the experimental infrastructure constructed for this paper.

| Source | Benchmarks used here | Inherited from public source | Rebuilt or extended in this work |
|---|---|---|---|
| PDE-OP | Voltage; Burgers | PDE control families; learned and classical reference methods | regenerated optimized-control banks; own operator data; 12 targets per PDE; exact numerical rescoring |
| BPN | Heat2D; Poisson; NS2inlets | PDE control formulations; released BPN baselines | independent CN–FD and FEM solvers; regenerated operator and control data; 3 targets per PDE; numerical shortlist verification |
| Opt-RNO | Irreg. Heat; Darcy; Advection | RNO methodology and irregular-operator optimization setting | new PDE tasks and geometries; new solver data and control banks; exact FEM verification; count- and time-matched protocols |

The extensions deliberately increase task diversity rather than only adding more random seeds. The PDE-OP-derived experiments expand the three representative Voltage and Burgers targets used in the

source setting to twelve fixed canonical targets for each PDE. Each BPN problem is evaluated on three frozen targets instead of a single canonical target. The irregular suite is a larger redesign: three newly constructed PDE families are evaluated not only on IID meshes but also under node-density, boundary-shape, hole, and L-shaped geometry shifts. Table 5 records this expansion.

Table 5: **Expansion from the source settings to the frozen evaluation suite used in this paper.**

| Family | Source setting | Frozen evaluation in this work | Main generalization axis |
|---|---|---|---|
| Voltage | three representative target types | 12 fixed target profiles $\times$ 3 search seeds | target shape |
| Burgers | three representative terminal targets | 12 fixed terminal targets $\times$ 3 search seeds | nonlinear target shape |
| Heat2D | one canonical tracking target | 3 fixed spatio-temporal targets $\times$ 3 seeds | amplitude, frequency, and phase |
| Poisson | one canonical field target | 3 fixed spatial targets $\times$ 3 seeds | target field |
| NS2inlets | one canonical outlet profile | 3 fixed outlet profiles $\times$ 3 seeds | outlet demand |
| Irregular suite | different Opt-RNO public tasks and datasets | 3 new PDEs $\times$ 5 geometry conditions | PDE and geometry shift |

### C.2 Independent numerical solvers and validation

A common numerical reference is needed to compare direct optimization, learned controllers, latent search, and surrogate-guided search. We therefore distinguish the *search evaluator* from the *reporting evaluator*: DeepONet or RNO may rank candidates online, but every score reported in the paper is recomputed with the fixed numerical solver.

This distinction is particularly important for the BPN-derived tasks. The released BPN implementation optimizes the PDE constraints through its physics-informed neural network (PINN) implementation in the DeepXDE scientific machine-learning library and does not expose an independent solver for rescoring arbitrary controls under a common discretization. We therefore implemented an independent numerical solver for each BPN problem and checked the formulation before using it for headline results. For Heat2D, we checked the implementation against the released `heat2d.py` formulation: the diffusion coefficient is $10^{-3}$, the official target has amplitude $16$, and the released uniform space–time MSE is analytically equivalent to the $0.5$-scaled integral used by our solver on a unit spatial domain over $T = 2$. For Poisson, the FEM implementation fixes the sign convention, outer and hole Dirichlet conditions, central control support, and unhalved tracking integral before deterministically rescoring selected controls. For NS2inlets, the DOLFINx solver includes unit tests for the mesh, control, and objective, a central-finite-difference check of the discrete-adjoint directional derivative, mesh-convergence checks, and nonlinear convergence and flux checks. All NS2inlets controls used in the reported sweeps converge on the fine verification mesh.

For the PDE-OP-derived tasks, the numerical dynamics are already part of the problem pipeline. We use them as the reporting solver and regenerate the training and evaluation data around that solver. Voltage uses a CN–FD reaction–diffusion solver. Burgers uses an explicit upwind treatment of nonlinear convection together with CN diffusion for the latent-search plant. Historical adjoint and POD-MPC baselines use a different nonlinear CN implementation; the continuous equation, grid, boundary conditions, target profiles, and terminal MSE are aligned, so these comparisons are task-level rather than claims of bitwise-identical time integration.

The irregular suite is generated directly from our unstructured FEM implementation. Every RNO-selected control is rescored by FEM, and the strict benchmark reports FEM objective gain rather than the RNO prediction. The same solver stack generates the IID and geometry-shift test cases.

### C.3 Offline data provenance: optimized controls versus solver data

Two different offline datasets are used in the paper and should not be conflated. The SWAE is trained on *optimization experience*: controls produced by numerical optimization on auxiliary PDE-control

Table 6: **Independent numerical solvers and validation checks.** The learned evaluator is never the source of the reported final quality.

| Benchmark | Reference solver | Resolution | Validation checks |
|---|---|---|---|
| Voltage | CN–FD | 101 space $\times$ 101 time nodes | numerical rollout; target-array consistency |
| Burgers | upwind convection + CN diffusion | 81 space $\times$ 201 time nodes | alignment of the PDE, grid, boundary conditions, and metric; numerical execute–replan |
| Heat2D | five-point CN–FD | $45 \times 45$ space, 161 time nodes | agreement with the released formulation; target and MSE scaling checks |
| Poisson | P1 triangular FEM | fixed $65 \times 65$ background resolution | checks of the PDE sign, boundary conditions, and control support; deterministic selected-control rescoring |
| NS2inlets | Taylor–Hood FEM | $64 \times 43$ verification mesh | unit tests; adjoint finite differences; mesh and nonlinear convergence checks |
| Irregular Heat, Darcy, and Advection | unstructured FEM | native irregular mesh | all final gains recomputed by FEM; geometry-shift checks |

tasks,

$$\mathcal{D}_{\mathrm{ctrl}} = \{c_i^\star\}_{i=1}^{N_{\mathrm{ctrl}}}. \tag{16}$$

By contrast, DeepONet and RNO are trained on *solver-generated operator data*,

$$\mathcal{D}_{\mathrm{op}} = \{(\xi_i, c_i, \mathcal{S}(\xi_i, c_i))\}_{i=1}^{N_{\mathrm{op}}}, \tag{17}$$

Here $N_{\mathrm{ctrl}}$ and $N_{\mathrm{op}}$ are the numbers of control and operator samples, respectively; $\xi_i$ denotes the state, geometry, and other conditioning variables and $\mathcal{S}$ is our frozen numerical PDE solver. Thus the representation learns where previous optimization has found useful controls, whereas the surrogate learns how the PDE responds to queried controls. No neural surrogate is used to generate the labels of another neural surrogate.

Table 7 gives the concrete generation routes. The three principal expert-control mechanisms are direct numerical optimization, MPC, including nonlinear MPC (NMPC), and discrete-adjoint optimization. L-BFGS-B denotes the limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm with bound constraints. In particular, the Voltage bank contains 5,000 independently optimized true-PDE controls; Burgers contains 2,000 adjoint-NMPC trajectories; Poisson contains 1,000 FEM+adjoint controls from 200 auxiliary tasks; and NS2inlets contains 450 trajectory snapshots from 30 target-level adjoint L-BFGS-B runs. The irregular SWAE banks are likewise generated from our FEM optimization pipeline rather than from an Opt-RNO dataset.

The operator datasets are regenerated from the same numerical physics used for verification. Voltage DeepONet data come from the CN–FD solver; Burgers propagator data come from the numerical Burgers simulator; the Heat2D, Poisson, and NS2inlets DeepONets use our CN–FD and FEM data; and the irregular RNO is trained on our newly generated irregular-FEM states and sensitivities. Consequently, the public source repositories determine the task lineage, while the actual offline samples used by our SWAE, DeepONet, and RNO models are generated inside our solver-verified pipeline.

### C.4 Benchmark summary and evaluation design

For structured control coordinates, DCT denotes the discrete cosine transform, PCA denotes principal component analysis, and GFT denotes the graph Fourier transform using eigenvectors of the mesh

Table 7: **Offline data provenance.** SWAE data are optimized controls; operator data are state, control, and geometry samples labeled by our numerical solver.

| Benchmark | Representation bank | Neural-evaluator data | Data split |
|---|---|---|---|
| Voltage | 5,000 true-PDE direct L-BFGS-B controls | CN–FD-generated DeepONet data | 4,250 training and 750 validation controls |
| Burgers | 2,000 adjoint-NMPC trajectories | numerical short-horizon propagator rollouts | 1,700 training and 300 validation trajectories; windows never cross the split |
| Heat2D | 500 retained optimized controls from 100 auxiliary target tasks | 4,000 CN–FD-generated operator samples | task-conditioned training and validation sets |
| Poisson | 1,000 FEM+adjoint controls from 200 tasks | 5,000 FEM-generated operator samples | representation fit on frozen training-task subset; task 199 held out |
| NS2inlets | 450 adjoint trajectory snapshots from 30 training targets | FEM-generated DeepONet data from the same training-task pool | target-level split; 360 training and 90 validation operator samples |
| Irregular Heat, Darcy, and Advection | FEM adjoint optimized-control banks generated for our irregular tasks | FEM-generated RNO state and sensitivity data | separate frozen training and test sets |

graph Laplacian. A latent dimension is the number of coordinates optimized after encoding with SWAE.

Table 8: **Benchmark instantiations and final verification.** "Dim." is the online control dimension before the SWAE map; irregular dimensions refer to structured coefficient spaces before the 32-D latent map.

| Benchmark | Control | Dim. | Latent | Exact evaluator | Surrogate | Final verification |
|---|---|---|---|---|---|---|
| Voltage | static spatial field | 101 | 4 | CN–FD | DeepONet | top-5 PDE |
| Burgers | space–time forcing window | $20 \times 81$ | 32 | true rollout | DeepONet | execute–replan |
| Heat2D | temporal forcing | 41 | 4 | CN–FD | DeepONet | top-5 CN–FD |
| Poisson | disk source | 509 | 8 | P1 FEM | DeepONet | top-5 FEM |
| NS2inlets | inlet profile | 62 free; 64 stored | 16 | Taylor–Hood FEM | DeepONet | top-3 FEM |
| Irreg. Heat | nodal control | DCT, PCA: 100; GFT: 97 | 32 | FEM | RNO | FEM rescore |
| Darcy | nodal control | DCT, PCA: 100; GFT: 97 | 32 | FEM | RNO | FEM rescore |
| Advection | nodal control | DCT, PCA: 100; GFT: 97 | 32 | FEM | RNO | FEM rescore |

Voltage and Burgers use the twelve canonical targets in Table 10 with search seeds $\{7, 19, 42\}$. Heat2D uses the official, slow-phase, and fast-phase targets; Poisson uses constant, cosine, and Gaussian targets; NS2inlets uses official, low-flow, and modulated outlet profiles. Their formulas are given below. The strict irregular study reports final FEM gain on pooled IID samples under matched wall time, whereas the irregular optimizer study uses search seeds 2026–2028. For the regular benchmarks, one evaluation case consists of one target paired with one search seed.

Table 9: **Evaluation design and sample counts.** Regular results are averaged over fixed target–seed pairs. The strict irregular study uses pooled test cells, whereas the optimizer study uses 12 fixed tasks and three search seeds per PDE.

| Suite | Test cases | Search seeds | Evaluation design |
|---|---|---|---|
| Voltage | 12 target profiles | 3 | 36 target–seed runs |
| Burgers | 12 terminal targets | 3 | 36 target–seed runs |
| Heat2D | 3 targets | 3 | 9 target–seed runs |
| Poisson | 3 targets | 3 | 9 target–seed runs |
| NS2inlets | 3 outlet profiles | 3 | 9 target–seed runs |
| Irregular strict | 1,000 paired test cells per PDE | – | pooled IID evaluation |
| Irregular optimizer | 12 tasks per PDE | 3 | 36 runs per PDE, optimizer, and evaluator |

### C.5 PDE-OP-DERIVED CONTROL PROBLEMS

#### C.5.1 VOLTAGE: STATIC DISTRIBUTED CONTROL OF REACTION–DIFFUSION DYNAMICS

Let $V(x,t)$ be the state on $\Omega = (0,1)$ and $c(x)$ a time-independent distributed control. The controlled dynamics are

$$\partial_t V(x,t) = D\,\partial_{xx} V(x,t) - \beta\big(V(x,t) - V_{\mathrm{ref}}\big) + \alpha c(x), \tag{18}$$

$$\partial_x V(0,t) = \partial_x V(1,t) = 0, \qquad V(x,0) = 0, \tag{19}$$

with

$$D = 0.1, \qquad \beta = 1, \qquad \alpha = 2, \qquad V_{\mathrm{ref}} = 1, \qquad -1 \le c(x) \le 1.$$

The control is discretized at 101 spatial nodes. The horizon is $T = 5$ with 101 temporal nodes ($\Delta x = 0.01$, $\Delta t = 0.05$). For a prescribed terminal target $V_d(x)$, the reported quantity is the 101-point terminal MSE,

$$J_{\mathrm{V}}(c; V_d) = \frac{1}{101} \sum_{j=1}^{101} \left(V(x_j, T; c) - V_d(x_j)\right)^2 . \tag{20}$$

Some search and data-generation paths additionally use a small control penalty $\gamma \|c\|_2^2$ with $\gamma = 10^{-4}$; all quality numbers reported in the main comparison use the terminal tracking MSE above. The final numerical trajectory is computed by the frozen CN–FD solver.

The source PDE-OP setting uses three representative target types. We freeze a broader set of twelve targets that includes constants, ramps, single- and multi-frequency sinusoids, and localized positive and negative bumps. This expansion is designed to test transfer of the learned control geometry across target shape while keeping the governing PDE and control bounds fixed.

#### C.5.2 BURGERS: NONLINEAR TRAJECTORY CONTROL WITH RECEDING-HORIZON EXECUTION

The Burgers benchmark controls a nonlinear space–time trajectory,

$$\partial_t u(x,t) + u(x,t)\partial_x u(x,t) = \nu\,\partial_{xx} u(x,t) + c(x,t), \tag{21}$$

$$u(0,t) = u(1,t) = 0, \tag{22}$$

$$u(x,0) = \frac{1}{2}\sin(2\pi x), \tag{23}$$

on $x \in [0,1]$, $t \in [0,4]$, with viscosity $\nu = 0.03$ and pointwise control bound $c(x,t) \in [-1,1]$. The control is zero at the two spatial endpoints. The frozen grid has 81 spatial nodes and 201 time nodes ($\Delta x = 0.0125$, $\Delta t = 0.02$), so a fully open-loop control contains $200 \times 81 = 16{,}200$ decisions.

For a terminal target $u_d(x)$, the reported objective is

$$J_{\mathrm{B}}(c; u_d) = \frac{1}{81} \sum_{j=1}^{81} \left(u(x_j, T; c) - u_d(x_j)\right)^2 . \tag{24}$$

Our online parameterization searches over phase-conditioned $20 \times 81$ control windows using a 32-D latent code. Candidate windows may be rolled out by the learned short-horizon propagator, but only the first controls are executed by the numerical Burgers plant before the next replanning step. Hence every realized trajectory is repeatedly re-anchored to the numerical PDE state.

The SWAE bank contains 2,000 adjoint-NMPC trajectories generated on auxiliary tasks with auxiliary initial states and targets. The split is made at the trajectory level (1,700 training and 300 validation trajectories); all 19 phase windows extracted from a trajectory stay on the same side of the split. The final benchmark expands the three representative source targets to twelve fixed terminal profiles.

Table 10: **Canonical Voltage and Burgers target specifications.** Voltage profiles are clipped to $[0.5, 1.5]$. For Burgers, $G(A, \mu, \sigma)$ denotes $A \exp[-(x-\mu)^2/(2\sigma^2)]$ with the linear endpoint interpolant removed so that both boundary values are zero.

| Voltage target | Profile | Burgers target | Profile on $[0, 1]$ |
|---|---|---|---|
| Paper sine | $1 + 0.20\sin(6\pi x)$ | Sine | $0.80\sin(\pi x)$ |
| Paper ramp | linear $0.50 \to 1.50$ | Parabola | $2x(1-x)$ |
| Constant (mid) | 1.00 | Zero | 0 |
| Constant (high) | 1.25 | Sine (low) | $0.45\sin(\pi x)$ |
| Sine (low) | $1 + 0.18\sin(2\pi x)$ | Sine (high) | $1.00\sin(\pi x)$ |
| Sine (mid) | $1 + 0.22\sin(4\pi x + \pi/5)$ | Second harmonic | $0.65\sin(2\pi x)$ |
| Ramp up | linear $0.65 \to 1.35$ | Parabola (low) | $x(1-x)$ |
| Ramp down | linear $1.35 \to 0.65$ | Central Gaussian | $G(0.85, 0.50, 0.13)$ |
| Positive bump | $1 + 0.30e^{-(x-.35)^2/(2(.10)^2)}$ | Bump (left) | $G(0.75, 0.28, 0.10)$ |
| Negative bump | $1 - 0.25e^{-(x-.65)^2/(2(.12)^2)}$ | Bump (right) | $G(0.75, 0.72, 0.10)$ |
| Multi-sine | $1 + .14\sin(2\pi x + .3) + .08\sin(6\pi x - .4)$ | Multi-sine | $.55\sin(\pi x) + .20\sin(2\pi x)$ |
| Smooth mixture | $1 + .12\sin 2\pi x - .10\cos 4\pi x + .06\sin(8\pi x + .7)$ | Antisymmetric | $.55\sin(2\pi x) - .15\sin(3\pi x)$ |

### C.6 BPN-derived control problems and independent numerical solvers

#### C.6.1 Heat2D: temporal forcing for a parabolic PDE

On $\Omega = (0,1)^2$ and $t \in [0, 2]$, the state satisfies

$$\partial_t u(x, y, t) - 10^{-3}\,(\partial_{xx} u + \partial_{yy} u) = f(t), \tag{25}$$

$$u|_{\partial\Omega} = 0, \qquad u(x, y, 0) = 0. \tag{26}$$

The control is a scalar temporal forcing, represented online by 41 samples and clipped to the common search range $[-5, 5]$. The released BPN code represents $f(t)$ by a neural network and imposes no explicit hard box; the common box in our benchmark provides a shared finite search domain and contains the released BPN controls used in the comparison.

The target family is

$$u_d(x, y, t) = A\, x(1-x)y(1-y)\sin(\omega t + \phi). \tag{27}$$

We use the official target $(A, \omega, \phi) = (16, \pi, 0)$, the slow-phase target $(18, 0.75\pi, -0.22)$, and the fast-phase target $(14, 1.25\pi, 0.18)$. The continuous objective used by the independent solver is

$$J_{\mathrm{H}}(f) = \frac{1}{2}\int_0^2 \int_\Omega \left(u(x, y, t; f) - u_d(x, y, t)\right)^2 \, dx\, dy\, dt. \tag{28}$$

Because $|\Omega| = 1$ and $T = 2$, this is the continuous counterpart of the uniform space–time MSE used by the released BPN implementation. We solve the state with a five-point spatial FD stencil and CN time stepping; formal final rescoring uses the $45 \times 45$ spatial grid and 161 time nodes.

#### C.6.2 Poisson: distributed source control on a perforated domain

Let

$$\Omega = [-4, 4]^2 \setminus \bigcup_{\sigma_1, \sigma_2 \in \{-1, 1\}} B_{0.8}(2.4\sigma_1, 2.4\sigma_2) \tag{29}$$

be the square domain with four circular holes, where $B_r(a)$ is the open disk of radius $r$ centered at $a$, and let $\Omega_c = B_{1.6}(0)$ denote the central control disk. The frozen convention matching the BPN formulation is

$$\Delta u(x) = c(x)\mathbf{1}_{\Omega_c}(x), \qquad x \in \Omega, \tag{30}$$

$$u = 1, \qquad x \in \partial[-4,4]^2, \tag{31}$$

$$u = 0, \qquad x \in \partial\Omega \setminus \partial[-4,4]^2. \tag{32}$$

Here $\mathbf{1}_{\Omega_c}$ is the indicator of the control disk. The control is represented by the 509 free P1-FEM source values in the central disk and is bounded by $-50 \le c_i \le 50$. For target field $u_d$ the reported objective is the unhalved tracking integral

$$J_{\mathrm{P}}(c) = \int_\Omega \left(u(x;c) - u_d(x)\right)^2 \, dx. \tag{33}$$

An optional $L^2$ control regularizer exists in the implementation but is disabled for the reported benchmark.

The three frozen target fields are constant (labeled "ones"), cosine, and Gaussian:

$$u_d^{\mathrm{ones}}(x,y) = 1, \tag{34}$$

$$u_d^{\mathrm{cosine}}(x,y) = 0.85 + 0.10\cos\Big(\frac{\pi x}{4}\Big)\cos\Big(\frac{\pi y}{4}\Big), \tag{35}$$

$$u_d^{\mathrm{gaussian}}(x,y) = 0.90 + 0.12\exp\left(-\frac{(x-0.9)^2 + (y+0.7)^2}{2(1.2)^2}\right). \tag{36}$$

The independent solver is a deterministic P1 triangular FEM solver with the fixed $65 \times 65$ background resolution. The SWAE control bank contains 1,000 FEM+adjoint controls generated across 200 auxiliary target tasks; the paper's frozen representation fit uses the designated training-task subset, with task 199 kept outside that fit. The DeepONet data are generated independently by the FEM solver, and surrogate-selected controls are verified by top-5 FEM rescoring.

#### C.6.3 NS2INLETS: BOUNDARY CONTROL OF STEADY INCOMPRESSIBLE FLOW

The state is the velocity–pressure pair $(\mathbf{v}, p)$ on $\Omega = (0, 1.5) \times (0, 1)$. At Reynolds number 100, $\nu = 10^{-2}$, and the steady equations are

$$-\nu\Delta\mathbf{v} + (\mathbf{v}\cdot\nabla)\mathbf{v} + \nabla p = 0, \tag{37}$$

$$\nabla\cdot\mathbf{v} = 0. \tag{38}$$

The controlled left boundary profile is

$$v_x(0,y) = 4\,\theta(y)y(1-y), \qquad v_y(0,y) = 0. \tag{39}$$

The stored control has 64 samples; the two endpoint values are fixed, leaving 62 free optimization variables. The frozen search domain is $0 \le \theta_i \le 4$.

Two additional prescribed inlet segments lie on the central portions of the horizontal boundaries, $x \in (0.5, 1.0)$, with the fixed parabolic profile used by the BPN implementation. The remaining horizontal walls are no-slip. The right boundary is the outlet, with pressure fixed to zero and the corresponding natural viscous-traction closure. We track the horizontal outlet velocity against one of three frozen profiles:

$$v_d^{\mathrm{official}}(y) = 4y(1-y), \tag{40}$$

$$v_d^{\mathrm{low}}(y) = 0.8\, v_d^{\mathrm{official}}(y), \tag{41}$$

$$v_d^{\mathrm{mod}}(y) = v_d^{\mathrm{official}}(y)\left(0.9 + 0.1\cos(2\pi y)\right). \tag{42}$$

The tracking term is

$$J_{\mathrm{track}}(\theta) = \int_0^1 \left(v_x(1.5, y; \theta) - v_d(y)\right)^2 \, dy. \tag{43}$$

Following the released objective, we also retain the soft feasibility penalty

$$J_{\mathrm{pen}}(\theta) = 100\,\mathrm{mean}\big([-\theta]_+^2 + [\theta - 4]_+^2\big) + 100\,[\mathrm{mean}(\theta) - 1.5]_+^2, \tag{44}$$

where $[a]_+ = \max(a, 0)$ and mean averages over control samples, and optimize $J_{\mathrm{NS}} = J_{\mathrm{track}} + J_{\mathrm{pen}}$. The box component is identically zero for our hard-bounded search; the mean-control term is kept. The independent solver uses Taylor–Hood P2 velocity and P1 pressure elements. Training trajectories are generated on a $48 \times 32$ mesh and every formal selected control is rescored on the $64 \times 43$ verification mesh.

The SWAE bank is generated from 30 auxiliary outlet targets sampled by a scrambled five-dimensional Halton design over smooth profile coefficients. A discrete-adjoint L-BFGS-B solve is run for each target and 15 control-space-separated trajectory snapshots are retained, giving 450 controls. The three evaluation targets above are not used to generate these training trajectories. Five-fold target-level validation selects the 16-D SWAE. DeepONet is trained from the same FEM-generated task pool using a target-level split of 360 training and 90 validation samples.

### C.7 Irregular Heat, Darcy, and Advection

The irregular suite extends the evaluation from fixed grids to PDE control on unstructured domains. Its generator is derived from the irregular benchmark infrastructure of Opt-RNO, but the datasets used in this paper are regenerated from the benchmark generators under a common, solver-aligned protocol rather than taken from the released Opt-RNO data. Heat, Darcy, and Advection share the same geometry and mesh generator, while retaining different state equations, coefficients, boundary conditions, and meanings of the control. This section defines the resulting benchmark used throughout the paper.

**Geometry, mesh, and data split.** Each domain is generated as a randomly perturbed star-shaped polygon. The boundary radius is perturbed by three random harmonics, interior points are sampled by rejection sampling inside the polygon, and the combined boundary and interior nodes are triangulated by Delaunay triangulation. Elements whose centroids lie outside the polygon are removed, after which unused nodes are discarded and the connectivity is remapped. The generated polygon is normalized by its area before the PDE instance is constructed. The large datasets use 128 boundary nodes and 896 interior nodes per geometry.

For each PDE, we generate 500 training geometries and 100 held-out test geometries, with 10 control samples per geometry, giving 5,000 training and 1,000 test samples. Training uses base seed 73115 and the test split uses 83115. The two splits are disjoint by geometry and sample identity (the dataset split, geometry identifier, and control-sample index). Thus the operator and representation models are evaluated on geometries and control instances that are not present in their training split.

**Heat.** Heat is a source-controlled diffusion–reaction problem. For a generated geometry $\Omega_g$, the scalar state $T_g$ satisfies

$$-\nabla\cdot(\kappa_g \nabla T_g) + \alpha_g T_g = s_g\, \theta_g \qquad \text{in } \Omega_g, \tag{45}$$

with homogeneous Dirichlet data $T_g = 0$ on $\partial\Omega_g$. Here $\theta_g$ is the bounded nodal source control. Writing $U(a, b)$ for the uniform distribution on $[a, b]$, the coefficients are sampled independently as

$$\kappa_g \sim U(0.035, 0.075), \qquad \alpha_g \sim U(0.015, 0.045), \qquad s_g \sim U(0.8, 1.2). \tag{46}$$

The target is a normalized Gaussian profile whose center coordinates are sampled from $U(0.25, 0.75)$ and whose two widths are sampled from $U(0.10, 0.18)$. The score rewards area-weighted state inside the target and penalizes off-target response,

$$\mathcal{S}_{\mathrm{heat}}(\theta_g) = \langle q_g, T_g\rangle_A - \lambda_g \langle 1 - q_g, T_g\rangle_A, \qquad \lambda_g \sim U(0.20, 0.35), \tag{47}$$

where $\langle a, b\rangle_A = \sum_i A_i a_i b_i / \sum_i A_i$ is the area-weighted nodal mean, with $A_i$ the nodal area weights, and $q_g$ is the target profile. Controls are projected to an area-weighted mean budget of $0.35$.

With linear triangular finite elements, the generated system has the form

$$(\kappa_g K_g + \alpha_g M_g)\, T_g = s_g M_g \theta_g, \tag{48}$$

where $K_g$ and $M_g$ are the stiffness and consistent mass matrices.

**Darcy.** Darcy uses the same irregular geometry and finite-element infrastructure but defines a scalar pressure problem. Importantly, the optimized variable is an *source field*, not the permeability. The pressure $p_g$ satisfies

$$-\nabla\cdot(\kappa_g\nabla p_g) + \alpha_g p_g = s_g\,\theta_g \qquad \text{in } \Omega_g, \qquad p_g = 0 \quad \text{on } \partial\Omega_g, \tag{49}$$

with

$$\kappa_g \sim U(0.020, 0.060), \qquad \alpha_g \sim U(0.001, 0.010), \qquad s_g \sim U(0.8, 1.2). \tag{50}$$

Its P1 finite-element discretization is

$$(\kappa_g K_g + \alpha_g M_g)\, p_g = s_g M_g \theta_g. \tag{51}$$

The target is again a normalized Gaussian, with center coordinates in $U(0.25, 0.75)$ and widths in $U(0.10, 0.22)$. The benchmark score is

$$\mathcal{S}_{\mathrm{darcy}}(\theta_g) = \langle q_g, p_g\rangle_A - 0.30\,\langle 1 - q_g, p_g\rangle_A, \tag{52}$$

and the source control is projected to the same area-weighted mean budget $0.35$.

**Advection.** Advection differs from the previous two problems in both its boundary conditions and the role of the control. The state is a concentration $c_g$ governed by

$$-\nu_g\Delta c_g + \mathbf{u}_g\cdot\nabla c_g + (r_g\theta_g + \ell_g)\, c_g = f_g \qquad \text{in } \Omega_g. \tag{53}$$

Thus $\theta_g$ modulates the *reaction coefficient*; it is not the forcing term. The forcing $f_g$ is an independently generated Gaussian source located near the left side of the normalized domain. Its center satisfies $x_f \sim U(0.12, 0.28)$ and $y_f \sim U(0.35, 0.65)$, with widths $\sigma_{f,x} \sim U(0.03, 0.08)$ and $\sigma_{f,y} \sim U(0.05, 0.12)$.

The velocity field is generated in normalized coordinates $(\xi, \eta)$ as

$$u_x(\xi,\eta) = u_0 + a_{\mathrm{sh}}\sin(\pi\eta) + 0.15\cos(2\pi\xi)\sin(\pi\eta), \tag{54}$$

$$u_y(\xi,\eta) = a_{\mathrm{sw}}\sin(2\pi\xi)\sin(\pi\eta), \tag{55}$$

where

$$u_0 \sim U(0.8, 1.3), \qquad a_{\mathrm{sw}} \sim U(0.05, 0.22), \qquad a_{\mathrm{sh}} \sim U(0.02, 0.18). \tag{56}$$

The remaining coefficients are

$$\nu_g \sim U(0.035, 0.065), \qquad r_g \sim U(0.75, 1.40), \qquad \ell_g \sim U(0.010, 0.030). \tag{57}$$

Boundary nodes in the leftmost and rightmost $10\%$ of the boundary $x$-range define inlet and outlet segments, respectively. We impose $c_g = 1$ at the inlet, $c_g = 0$ at the outlet, and the natural no-flux condition on the remaining wall. The target concentration is an independent normalized Gaussian with center coordinates in $U(0.25, 0.75)$ and widths in $U(0.10, 0.22)$; the benchmark uses a fixed spill penalty of $0.30$.

**Control generation and representation data.** The raw generator uses three primary control families for all three PDEs: graph-spectral controls, signed Gaussian mixtures, and localized patch controls. Before PDE-specific constraints are applied, a raw field $r$ is standardized and mapped to a bounded nodal field by

$$\theta = \mathrm{clip}(0.05 + 0.90\,\sigma(1.2\,\widehat{r}),\, 0,\, 1)\,, \qquad \widehat{r} = \frac{r - \mathrm{mean}(r)}{\mathrm{std}(r) + 10^{-6}}. \tag{58}$$

Here $\sigma(v) = (1 + e^{-v})^{-1}$ is the logistic sigmoid, and $\mathrm{clip}(v, 0, 1)$ clamps each nodal value to $[0, 1]$; $\mathrm{mean}$ and $\mathrm{std}$ are the nodal mean and standard deviation. Heat and Darcy then apply the area-weighted budget projection described above; Advection uses the bounded field directly.

The representation-training bank is generated independently of the held-out test targets and optimized test trajectories. In addition to the three primary families, the revised bank includes independent Gaussian, complement, low- and high-baseline, flow-gradient, and multiscale-local structures to broaden control-space coverage. No test target, target solution, or target-derived deterministic transform is used to construct this bank. For Advection, graph-spectral alignment is a representation step applied after the control field is generated and does not alter the PDE in Eq. (53).

**Serialized data and reproducibility.** Each generated sample is stored as $(x, y, \mathcal{T}, m)$, where $\mathcal{T}$ is the triangular connectivity and $m$ contains the geometry, coefficients, boundary labels, target, control family, objective, and solver metadata. For the large datasets, $x, y \in \mathbb{R}^{1024\times 4}$ and the mesh contains approximately $1.9 \times 10^3$ triangles. The generator configuration also records the data scale, mesh resolution, control families, and random seeds.

The generation protocol checks reproducibility at the geometry, mesh, control, target, and numerical-solution levels. Re-solving stored samples must reproduce the state and objective within floating-point tolerance; control bounds and physical constraints, including area budgets, are checked to $10^{-6}$; and training, validation, and test identifiers are checked for overlap. These generated samples, together with the numerical FEM re-evaluation described in Appendix G, define the irregular benchmark used in the experiments.

# D Additional Algorithmic Designs

The main framework separates *where* to search from *how* to search. Two additional algorithmic constructions are used when the basic static latent-search primitive is insufficient: a receding-horizon composition for long trajectory controls, and a DANTE-GSA proposal mechanism that combines DANTE's tree-structured exploration with a generalized simulated annealing (GSA) proposal. DANTE (deep active optimization with neural-surrogate-guided tree exploration) provides the tree-based derivative-free search used here; GSA supplies its temperature-controlled, heavy-tailed proposal distribution. The first construction is essential for the Burgers trajectory problem; the second is used as an optional derivative-free search variant in the optimizer attribution experiments.

## D.1 Phase-Conditioned Receding-Horizon Latent Search

A static control can be represented by one latent code for the entire decision. For a long trajectory, however, a single open-loop code couples all future actions to the state observed at the beginning of the horizon. The Burgers control contains $200 \times 81 = 16{,}200$ space–time decisions, and the state revealed after executing even a short block can invalidate an open-loop plan. We therefore compose the same latent-search primitive in a receding-horizon loop whose optimization state is repeatedly re-anchored to the numerical PDE.

**Phase-conditioned windows.** Let

$$c^{(i)}_{0:T-1} = \left(c^{(i)}_0, \ldots, c^{(i)}_{T-1}\right)$$

be a high-quality historical control trajectory. In this discrete-time construction, $T$ is the number of control steps, $H$ is the window length, $\Delta$ is the stride between windows, and $P$ is the number of phases. For phase origins $t_p = p\Delta$, we extract overlapping length-$H$ windows

$$W_{i,p} = \left[c^{(i)}_{t_p}, c^{(i)}_{t_p+1}, \ldots, c^{(i)}_{t_p+H-1}\right], \qquad p = 0, \ldots, P-1. \tag{59}$$

The phase index $p$ is used as the conditioning variable of the manifold,

$$z_{i,p} = E_\phi(W_{i,p}; p), \qquad D_\theta(z; p) = [c_{0|p}(z), \ldots, c_{H-1|p}(z)]. \tag{60}$$

We also retain the encoded historical windows as a phase-indexed proposal pool,

$$\mathcal{P}_p = \{E_\phi(W_{i,p}; p)\}_i. \tag{61}$$

Thus, the same historical optimization experience is used twice: first to learn the geometry of admissible trajectory segments, and again to provide phase-matched online proposals.

The frozen Burgers model uses $H = 20$, phase stride $\Delta = 10$, $P = 19$ overlapping phases, and a 32-dimensional latent code. Training and validation splitting is performed at the full-trajectory level, so windows extracted from one trajectory never cross the split.

**State-conditioned finite-horizon objective.** At replanning step $k$, let $u_k$ be the current state produced by the *numerical* Burgers solver. The active phase and within-window offset are

$$p_k = \min\left(\left\lfloor \frac{k}{\Delta} \right\rfloor, P-1\right), \qquad o_k = k - t_{p_k}, \qquad H_k = \min(H - o_k, T - k). \tag{62}$$

For a latent candidate $z$, the controls used from that window are

$$c_{k+\ell|k}(z) = [D_\theta(z; p_k)]_{o_k+\ell}, \qquad \ell = 0, \dots, H_k - 1. \tag{63}$$

Candidate trajectories always start from the current numerical state,

$$\widetilde{u}_{k|k}(z) = u_k, \qquad \widetilde{u}_{k+\ell+1|k}(z) = \widetilde{\Phi}_\tau\big(\widetilde{u}_{k+\ell|k}(z), c_{k+\ell|k}(z)\big)\,, \tag{64}$$

where $\widetilde{\Phi}_\tau$ can be either the numerical one-step map or the frozen learned propagator used for fast candidate scoring.

For state target $u_d$, the finite-horizon latent objective is

$$\begin{aligned} F_k(z) = \sum_{\ell=0}^{H_k-1} \Big[ & q \left\| \widetilde{u}_{k+\ell+1|k}(z) - u_d \right\|_2^2 + r \left\| c_{k+\ell|k}(z) \right\|_2^2 \Big] \\ & + q_T \left\| \widetilde{u}_{k+H_k|k}(z) - u_d \right\|_2^2 + r_\Delta \left\| c_{k|k}(z) - c_{k-1} \right\|_2^2, \end{aligned} \tag{65}$$

where the final switching term is omitted at the first replanning step. The stage, terminal, control-energy, and switching weights $q$, $q_T$, $r$, and $r_\Delta$, respectively, are fixed before test-task optimization.

**Phase-aware warm start and local refinement.** A global latent box is estimated from the encoded training controls. At replanning step $k$, we first re-score a subset of phase-matched historical codes under the *current* state-conditioned objective,

$$z_k^{\mathrm{warm}} \in \arg\min_{z \in \widetilde{\mathcal{P}}_{p_k}} F_k(z), \qquad \widetilde{\mathcal{P}}_{p_k} \subseteq \mathcal{P}_{p_k}. \tag{66}$$

The online optimizer then refines this proposal inside

$$\mathcal{Z}_k = \mathcal{Z}_{\mathrm{search}} \cap \{ z : \|z - z_k^{\mathrm{warm}}\|_\infty \le r_z \}\,. \tag{67}$$

Here $r_z$ is the radius of the local search box. This is more than a conventional initialization. The phase pool selects a historically plausible trajectory segment, while the current numerical state re-ranks those historical segments for the task actually encountered online.

Under a budget of $B_k$ candidate evaluations, let $\widehat{z}_k$ denote the incumbent returned by latent search:

$$\widehat{z}_k \in \arg\min_{z \in \mathcal{Z}_k}^{B_k} F_k(z). \tag{68}$$

Only the first

$$s_k = \min(s, H_k) \tag{69}$$

controls are committed. They are executed with the true numerical dynamics,

$$u_{k+j+1} = \Phi_\tau\big(u_{k+j}, c_{k+j|k}(\widehat{z}_k)\big)\,, \qquad j = 0, \dots, s_k - 1, \tag{70}$$

after which the optimization is rebuilt from the resulting numerical state $u_{k+s_k}$. The frozen Burgers configuration uses $s = 10$.

The resulting computation can be summarized as

$$u_k \xrightarrow[\text{phase-matched warm start}]{\text{32-D latent optimization}} \widehat{c}_{k:k+H_k-1} \xrightarrow[\text{true PDE}]{\text{execute } s_k \text{ controls}} u_{k+s_k} \xrightarrow{\text{replan}} \cdots\,. \tag{71}$$

This construction is the key distinction from a single open-loop latent code: the learned propagator, when used, accelerates only the *candidate evaluation* inside a short horizon, whereas the state from which the next optimization starts is always generated by the numerical PDE. The manifold is therefore reused across the full trajectory without allowing surrogate rollout error to accumulate unchecked over the entire horizon.

### D.2 DANTE-GSA: Tree Search with a Heavy-Tailed Adaptive Proposal

DANTE-GSA is a lightweight modification of the DANTE search kernel rather than a separate optimization pipeline. Vanilla DANTE and DANTE-GSA share the same discretized latent domain, initial point, upper confidence bounds applied to trees (UCT) selection, stopping rule, evaluator-call accounting, and best-so-far selection. The production implementation differs only in the child-proposal mechanism.

**Shared DANTE tree search.** Let the latent search box be $\mathcal{Z} = \prod_{j=1}^{d}[\ell_j, u_j]$. DANTE discretizes each coordinate into a common grid with $M$ intervals,

$$\mathcal{G}_j = \left\{ \ell_j + \frac{m}{M}(u_j - \ell_j) : m = 0, \dots, M \right\}, \qquad \mathcal{G} = \prod_{j=1}^{d} \mathcal{G}_j. \tag{72}$$

The PDE experiments minimize an objective $F(z)$; the DANTE implementation therefore uses the utility

$$U(z) = -F(z) \tag{73}$$

and maximizes $U$. If $n$ is a child of node $p$, selection uses

$$\mathrm{UCT}(n) = U(n) + \omega \sqrt{\frac{\log N(p)}{N(n) + 1}}, \tag{74}$$

where $N(n)$ and $N(p)$ count visits to the child and parent nodes, respectively, and $U(n)$ is the child utility, with an objective-scale-adaptive exploration weight $\omega \propto |U_{\mathrm{best}}|$. Each expansion generates exactly $d$ children, with child $j$ assigned coordinate $j$ as its primary search direction. Vanilla DANTE then perturbs that child using one of six random neighborhood moves: a one-grid-step move, a random reset of the primary coordinate, or a random reset of a small set of coordinates.

**GSA visiting proposal.** DANTE-GSA keeps the tree and UCT rule above but replaces the six-case random proposal by a temperature-controlled generalized Cauchy–Lorentz visitor. The visiting-distribution shape parameter is $q_v = 2.62$, the initial temperature is $T_0 = 5230$, and the temperature floor is $T_{\min} = 0.1$. At expansion $t$, the temperature is

$$T_t = \max\left[ T_{\min},\ T_0 \frac{2^{q_v - 1} - 1}{(t+2)^{q_v - 1} - 1} \right]. \tag{75}$$

The number of perturbed coordinates also decays with the search,

$$m_t = \max\left( 1, \left\lceil \frac{d}{1 + \eta t} \right\rceil \right), \qquad \eta = 0.02. \tag{76}$$

For child $j$, the active set always contains its designated primary coordinate,

$$S_{t,j} = \{j\} \cup R_{t,j}, \qquad |S_{t,j}| = m_t, \tag{77}$$

where $R_{t,j}$ is sampled without replacement from the remaining dimensions. A heavy-tailed visiting vector

$$\delta_{t,j} \sim \mathcal{V}_{q_v}(T_t; S_{t,j}) \tag{78}$$

is drawn only on this active set, using the generalized Cauchy–Lorentz visiting distribution of the dual-annealing implementation of GSA. The continuous proposal is clipped to the latent box and then snapped back to the same DANTE grid,

$$\tilde{z}^{(j)} = \Pi_{\mathcal{Z}}\left( z + P_{S_{t,j}} \delta_{t,j} \right), \qquad z^{(j)} = Q_{\mathcal{G}}\left( \tilde{z}^{(j)} \right), \tag{79}$$

where $P_S$ injects the active coordinates and $Q_{\mathcal{G}}$ denotes nearest-grid projection.

Equations (75)–(79) create a simple global-to-local transition. Early in the search, $T_t$ is high and $m_t \approx d$, so heavy-tailed proposals can move across many latent coordinates. Later, the temperature and active dimension count decrease; in the limiting regime $m_t = 1$, child $j$ perturbs only its designated DANTE coordinate. The construction therefore preserves DANTE's systematic dimension-wise tree expansion while borrowing GSA's long-range visiting behavior in the part of the search where global jumps are most useful.

**What is and is not changed.** The implementation without gradient repair isolates this proposal mechanism. It does *not* use a Metropolis acceptance step, gradient repair, or temperature-modulated UCT. Hence the comparison between DANTE and DANTE-GSA attributes the difference to the heavy-tailed visitor and the adaptive active-dimension schedule rather than to a second acceptance

Table 11: **DANTE and DANTE-GSA differ only in the proposal kernel in the implementation without gradient repair.**

| Component | DANTE | DANTE-GSA |
|---|---|---|
| Latent grid and bounds | same | same |
| UCT tree selection | same | same |
| Primary dimension per child | yes | yes |
| Proposal | six-case random neighborhood | heavy-tailed GSA visitor |
| Perturbed dimensions | random move dependent | $m_t$ with primary dimension fixed |
| Temperature schedule | none | Eq. (75) |
| Metropolis acceptance | no | no |
| Gradient repair | no | no |
| Temperature-modulated UCT | no | no |

or local-search procedure. Because one initial point is followed by complete $d$-child expansions, a nominal budget $B$ yields

$$B_{\mathrm{DANTE}} = 1 + d \left\lfloor \frac{B-1}{d} \right\rfloor \tag{80}$$

actual objective calls; all optimizer-attribution experiments record these actual calls.

DANTE-GSA is therefore used as an optional search variant rather than as the main source of performance gains in the paper. Its purpose is to test whether the same learned search geometry can benefit from a proposal kernel that moves from broad heavy-tailed exploration to increasingly local, dimension-structured refinement.

# E Representations, Search, and Learned Evaluators

This section collects the implementation choices that connect the historical control bank to online optimization. The sequence is the same in every benchmark: a control is represented in a fixed search space, the optimizer proposes candidates in that space, and an exact or learned evaluator scores those candidates. The three parts are described together here because their interfaces determine the actual online problem seen by the optimizer. The search methods include dual annealing (DA) and the covariance matrix adaptation evolution strategy (CMA-ES), alongside DANTE and DANTE-GSA defined in Appendix D.2.

## E.1 Control representations and model selection

**Direct and structured control coordinates.** For the regular-grid benchmarks, SWAE is trained directly on normalized optimized controls. For the irregular problems, the nodal control dimension is first reduced with a structured basis and SWAE is trained on the resulting coefficients. Let $x$ denote the normalized representation presented to SWAE, $z = E_\phi(x)$ the latent code, and $\widehat{x} = D_\theta(z)$ the reconstruction. The training objective is

$$\mathcal{L}_{\mathrm{SWAE}} = \|x - D_\theta(E_\phi(x))\|_2^2 + \beta_{\mathrm{SW}} \widehat{\mathrm{SW}}_2^2(Q_\phi, P_Z) + \mathcal{R}_{\mathrm{sm}}, \tag{81}$$

where $Q_\phi$ is the empirical distribution of encoded controls and $P_Z$ is the latent prior. With random unit directions $\{\vartheta_\ell\}_{\ell=1}^L$, the sliced term is evaluated as

$$\widehat{\mathrm{SW}}_2^2(Q_\phi, P_Z) = \frac{1}{L} \sum_{\ell=1}^{L} W_2^2 \left( \{\vartheta_\ell^\top z_i\}_i, \{\vartheta_\ell^\top \epsilon_i\}_i \right), \qquad \epsilon_i \sim P_Z. \tag{82}$$

Here $W_2$ is the second-order Wasserstein distance, $L$ is the number of projection directions, and $\beta_{\mathrm{SW}}$ weights the sliced penalty. The smoothness penalty $\mathcal{R}_{\mathrm{sm}}$ is used only where the control has an intrinsic spatial or space–time ordering. In Voltage it penalizes neighboring spatial differences; in Burgers it penalizes both temporal and spatial variation of each decoded window. Normalization statistics, basis functions, latent bounds, and network weights are fixed before test-task optimization.

Table 12 lists the actual dimensionality exposed to the online optimizer. The direct tasks reduce from 41–509 control variables (1620 per Burgers window) to 4–32 latent variables. On irregular meshes, the first reduction gives a common coefficient space of roughly one hundred variables; the learned manifold then reduces that space to 32 dimensions.

Table 12: **Control coordinates used by online search.**

| Benchmark | Control dimension | Structured coordinates | SWAE latent | Online latent bounds |
|---|---|---|---|---|
| Voltage | 101 | – | 4 | training latent quantiles |
| Burgers | 1620 per window | – | 32 | phase-matched training codes |
| Heat2D | 41 | – | 4 | frozen training support |
| Poisson | 509 | – | 8 | frozen training support |
| NS2inlets | 62 free | – | 16 | frozen training support |
| Irreg. Heat | nodal | DCT, PCA: 100; GFT: 97 | 32 | 1st and 99th percentiles per latent coordinate |
| Darcy | nodal | DCT, PCA: 100; GFT: 97 | 32 | 1st and 99th percentiles per latent coordinate |
| Advection | nodal | DCT, PCA: 100; GFT: 97 | 32 | 1st and 99th percentiles per latent coordinate |

**Structured bases on irregular meshes.** The DCT representation interpolates each nodal control to a $96 \times 96$ reference grid and retains a 100-dimensional low-frequency block. PCA is fitted only to training controls and keeps 100 components. The graph-spectral basis uses the low-frequency eigenvectors of the mesh graph Laplacian and retains approximately 97 coefficients. These bases provide a compact coordinate system for the geometry-dependent nodal field; SWAE then learns the subset of those coordinates occupied by optimized controls.

**Model selection.** The latent dimension was selected before the online experiments and was not chosen from final optimization performance. Reconstruction quality was used together with preservation of the numerical control objective on held-out optimized controls. Table 13 summarizes the model selection that is most relevant to the final configurations. Voltage uses the 4-D, three-layer 128-wide model trained on 4,250 controls with 750 held out. Burgers uses a 32-D model on phase-conditioned windows, with the training and validation split performed at the trajectory level. Heat2D was swept over latent dimensions $\{4, 8, 16, 32\}$ and several three- or four-layer multilayer perceptrons (MLPs); the online model is 4-D. Poisson was swept over $\{4, 8, 16, 24, 32, 48\}$ with MLP and convolutional neural network (CNN) alternatives; the online model is 8-D. NS2inlets uses a 16-D model selected by five-fold *target-level* cross-validation (CV) over 30 auxiliary targets.

SiLU denotes the sigmoid linear unit activation, $\mathrm{SiLU}(x) = x\sigma(x)$. Network dimensions list successive layer widths; $128 \times 3$, for example, denotes three hidden layers of width 128. In the architecture sweeps, $d_z$ is the latent dimension.

**Poisson masked DCT.** Poisson is the only regular-grid problem whose control lives on an irregular set of FEM nodes inside a disk. A tensor-grid DCT cannot therefore be applied by simply reshaping the control vector. We construct orthonormal 2-D type-II DCT basis fields on the frozen $65 \times 65$ background grid and sample those fields at the 509 control nodes. The eight retained modes are

$$(0,0), (0,1), (1,0), (1,1), (0,2), (2,0), (1,2), (2,1).$$

Coefficient bounds are the 1st and 99th percentiles of least-squares DCT coefficients on the same 800 training controls used for representation selection, and decoded controls are clipped to $[-50, 50]$.

This additional spectral baseline was run after the original fixed-DA representation table had been frozen, so we keep it under the matched 193-proxy-call plus top-5 FEM protocol rather than mixing it into the 200-call table. As shown in Table 14, DCT remains substantially above the matched SWAE objective for all four optimizers.

### E.2 Online search and evaluation budgets

All budgets in the paper count objective evaluations, not optimizer iterations. This distinction matters for the tree-based optimizers because one DANTE expansion evaluates one child per latent dimension.

Table 13: **SWAE model selection for the representations used online.** For Heat2D and Poisson, the table reports the selection sweep because several architectures were compared; for Voltage, Burgers, and NS2inlets, the final training record is also available.

| Benchmark | Latent dim. | Training data | Architecture | Regularization | Selection rule |
|---|---|---|---|---|---|
| Voltage | 4 | 4,250 training and 750 validation controls | 101–128×3–4, SiLU | $\beta_{\mathrm{SW}} = 0.01$, 64 projections, spatial smoothness 0.01 | best validation checkpoint |
| Burgers | 32 | 32,300 training and 5,700 validation windows; trajectory-level split | 1620–512×3–32, SiLU | $\beta_{\mathrm{SW}} = 0.01$, 64 projections, temporal smoothness $10^{-3}$; spatial smoothness $10^{-4}$ | best validation checkpoint |
| Heat2D | 4 | optimized-control bank from auxiliary targets | $d_z \in \{4, 8, 16, 32\}$; MLP width 128 or 256, 3–4 layers | smoothness $\in \{0, 10^{-3}\}$ | reconstruction + held-out true-objective retention |
| Poisson | 8 | 800 training controls from 160 tasks; 200 validation controls from 40 tasks | $d_z \in \{4, 8, 16, 24, 32, 48\}$; MLP or CNN | smoothness $\in \{0, 10^{-3}\}$ | reconstruction + held-out FEM-objective retention |
| NS2inlets | 16 | 30 auxiliary targets, 5-fold target CV | 62–128–64–16–64–128–62 | fixed training recipe | target-level CV, then final refit |

Table 14: **Poisson masked-DCT comparison under 193 proxy evaluations and top-5 FEM verification.** Values are pooled mean±sample standard deviation over three targets and three search seeds; lower is better. The last column is the ratio of the DCT mean objective to the SWAE mean objective.

| Optimizer | DCT | SWAE | Mean ratio |
|---|---|---|---|
| DA | $16.449 \pm 3.664$ | $4.930 \pm 1.082$ | 3.34× |
| CMA-ES | $14.151 \pm 2.295$ | $4.916 \pm 1.113$ | 2.88× |
| DANTE | $14.288 \pm 2.019$ | $5.127 \pm 1.035$ | 2.79× |
| DANTE-GSA | $20.707 \pm 1.037$ | $5.350 \pm 1.219$ | 3.87× |

The exact attainable DANTE count is derived in Appendix D.2; the experiments record the actual calls rather than silently reporting the nominal ceiling.

DA uses the SciPy implementation of dual annealing with local search disabled in the matched comparisons. The irregular CMA-ES runs use `pycma` with population 16 and normalized initial scale $\sigma_0 = 0.25$. DANTE and DANTE-GSA use the same tree, grid, initialization, and UCT rule; their proposal kernels are described in Appendix D.2. The representation, latent bounds, evaluator checkpoint, target, and search seed are held fixed when optimizers are compared.

Table 15 gives the call budgets used in the main controlled comparisons. The slightly unusual values 497 and 193 are therefore intentional: they are the largest common attainable counts under the relevant latent dimension, not dropped or missing evaluations. Final numerical verification is listed separately from proxy-search calls.

Table 15: **Online search budgets used in the controlled comparisons.** P denotes learned-model evaluations. Final verification uses FEM or another numerical solver.

| Comparison | Benchmark | Search evaluator | Actual search calls | Final numerical step |
|---|---|---|---|---|
| Representation | Voltage | exact PDE | 200 | already exact |
| Representation | Burgers | exact rollout | 100 | already exact |
| Representation | Heat2D | DeepONet | 500 P | top-5 solver |
| Representation | Poisson | DeepONet | 200 P | top-5 FEM |
| Representation | NS2inlets | DeepONet | 200 P | top-3 FEM |
| Optimizer | Voltage | exact or DeepONet | 197 | top-5 only for surrogate search |
| Optimizer | Burgers | exact or DeepONet | 730 | numerical execute–replan |
| Optimizer | Heat2D | DeepONet | 497 P | top-5 solver |
| Optimizer | Poisson | DeepONet | 193 P | top-5 FEM |
| Optimizer | NS2inlets | DeepONet | 193 P | top-3 FEM |
| Irregular representation | all | RNO | 80 P | exact FEM rescore |
| Irregular optimizer | all | FEM or RNO | 65 | exact FEM after RNO search |

**Regular benchmarks.** With SWAE fixed, the full four-optimizer comparison is given in Table 16. The table makes the task dependence seen in the main paper explicit. DANTE is strongest on Voltage and NS2inlets, CMA-ES is strongest on Burgers under exact evaluation and is marginally best on Poisson, while Heat2D gives nearly identical results for DANTE and CMA-ES. Burgers also changes its preferred optimizer when the evaluator changes: CMA-ES is best with exact rollouts, whereas DA is best with the learned propagator. These changes are the reason we treat the optimizer as a separate choice from the learned control geometry.

Table 16: **Full optimizer comparison on the regular benchmarks.** The SWAE representation is fixed. Entries are final numerical objectives, mean$\pm$sample standard deviation; lower is better. Voltage surrogate values use the final top-5 numerical selection.

| Benchmark | Evaluator | DA | CMA-ES | DANTE | DANTE-GSA |
|---|---|---|---|---|---|
| Voltage | Exact | $(7.12 \pm 6.28) \times 10^{-3}$ | $(5.65 \pm 8.15) \times 10^{-3}$ | $(5.24 \pm 6.76) \times 10^{-3}$ | $(16.6 \pm 11.3) \times 10^{-3}$ |
| Voltage | Surrogate | $(7.04 \pm 7.24) \times 10^{-3}$ | $(6.18 \pm 8.99) \times 10^{-3}$ | $(6.14 \pm 9.01) \times 10^{-3}$ | $(16.6 \pm 11.3) \times 10^{-3}$ |
| Burgers | Exact | $(10.3 \pm 13.3) \times 10^{-3}$ | $(6.04 \pm 7.68) \times 10^{-3}$ | $(9.94 \pm 9.69) \times 10^{-3}$ | $(9.92 \pm 10.98) \times 10^{-3}$ |
| Burgers | Surrogate | $(8.28 \pm 7.25) \times 10^{-3}$ | $(8.54 \pm 12.37) \times 10^{-3}$ | $(9.18 \pm 8.32) \times 10^{-3}$ | $(12.2 \pm 15.7) \times 10^{-3}$ |
| Heat2D | Surrogate | $(3.626 \pm 0.476) \times 10^{-2}$ | $(3.591 \pm 0.473) \times 10^{-2}$ | $(3.589 \pm 0.465) \times 10^{-2}$ | $(5.067 \pm 1.776) \times 10^{-2}$ |
| Poisson | Surrogate | $4.9295 \pm 1.0817$ | $4.9164 \pm 1.1132$ | $5.1270 \pm 1.0347$ | $5.3502 \pm 1.2190$ |
| NS2inlets | Surrogate | $(3.365 \pm 0.785) \times 10^{-2}$ | $(3.976 \pm 1.321) \times 10^{-2}$ | $(3.322 \pm 0.685) \times 10^{-2}$ | $(4.948 \pm 4.151) \times 10^{-2}$ |

**Irregular meshes.** The irregular optimizer comparison fixes DCT$\rightarrow$SWAE on Heat and Darcy and GFT$\rightarrow$SWAE on Advection, then gives every optimizer 65 actual candidate evaluations. Table 17 reports the final FEM objective for 12 test tasks and three search seeds. CMA-ES is best on Heat and Darcy under both exact and RNO evaluation. On Advection, DANTE-GSA and CMA-ES are essentially tied under exact evaluation, while DANTE-GSA is highest with RNO scoring. The preferred optimizer therefore changes with the PDE even after the latent representation has been fixed.

Table 17: **Full optimizer comparison on irregular meshes.** The representation is DCT→SWAE for Heat and Darcy and GFT→SWAE for Advection. Entries are final exact-FEM objectives, mean±sample standard deviation over 12 tasks×3 seeds; higher is better.

| PDE | Search evaluator | DA | CMA-ES | DANTE | DANTE-GSA |
|---|---|---|---|---|---|
| Heat | Exact | $0.3458 \pm 0.1048$ | $0.4234 \pm 0.1269$ | $0.3091 \pm 0.0980$ | $0.3972 \pm 0.1187$ |
| Heat | RNO | $0.3431 \pm 0.1070$ | $0.4123 \pm 0.1256$ | $0.3085 \pm 0.0995$ | $0.3904 \pm 0.1258$ |
| Darcy | Exact | $0.4837 \pm 0.1737$ | $0.5881 \pm 0.2141$ | $0.4309 \pm 0.1632$ | $0.5511 \pm 0.1969$ |
| Darcy | RNO | $0.4663 \pm 0.1699$ | $0.5581 \pm 0.1967$ | $0.4275 \pm 0.1655$ | $0.5317 \pm 0.2010$ |
| Advection | Exact | $0.6087 \pm 0.0752$ | $0.6295 \pm 0.0704$ | $0.5876 \pm 0.0750$ | $0.6299 \pm 0.0691$ |
| Advection | RNO | $0.5976 \pm 0.0789$ | $0.6057 \pm 0.0796$ | $0.5816 \pm 0.0770$ | $0.6225 \pm 0.0721$ |

### E.3 Learned evaluators and numerical final selection

A learned evaluator is used only to make candidate scoring cheaper. The final quantity reported in the paper is computed by the numerical PDE solver. Voltage uses a DeepONet for the terminal voltage field; Burgers uses a learned short-time propagator inside the receding-horizon objective; Heat2D, Poisson, and NS2inlets use task-specific DeepONets; and the irregular suite uses RNO. The exact role of each learned model and the final numerical step are listed in Table 18.

Table 18: **Learned evaluator and final numerical selection.**

| Benchmark | Learned-model output | Use during search | Numerical final step |
|---|---|---|---|
| Voltage | terminal $V(x, T)$ | candidate objective ranking | top-5 PDE reevaluation |
| Burgers | short-time state | finite-horizon rollout | PDE execute–replan |
| Heat2D | state trajectory | candidate objective ranking | top-5 CN–FD reevaluation |
| Poisson | state or objective proxy | candidate objective ranking | top-5 FEM reevaluation |
| NS2inlets | flow or objective proxy | candidate objective ranking | top-3 FEM reevaluation |
| Irregular optimizer study | state or objective proxy | RNO candidate scoring | one final FEM evaluation |
| Irregular fresh-FEM comparison | state or objective proxy | matched RNO proxy search | direct: 1 FEM; SWAE: top-5 fresh FEM |

**Top-$K$ numerical selection.** Let $\{c_i\}_{i=1}^{B}$ be the controls visited by proxy search and let $\widetilde{J}(c_i)$ be their learned-model scores. For a minimization problem, we form

$$\mathcal{I}_K = \underset{1 \le i \le B}{\arg \operatorname{Kmin}} \widetilde{J}(c_i) \tag{83}$$

and return

$$c^\star = \arg \min_{i \in \mathcal{I}_K} J_{\mathrm{num}}(c_i), \tag{84}$$

where $J_{\mathrm{num}}$ is the numerical objective used by the benchmark. For higher-is-better objectives the ordering is reversed. This is the selection rule behind the Voltage, Heat2D, and Poisson top-5 and NS2inlets top-3 results. In Burgers the analogous safeguard is stronger: the learned propagator can score short-horizon candidates, but the controls are executed by the numerical PDE and the next optimization starts from that numerical state.

**Operator-model validation.** The operator model must preserve the ordering of controls encountered during optimization, not merely predict states accurately on a generic held-out set. For this reason the BPN operator datasets were expanded toward the control regions reached by search. Heat2D increased from roughly 500 to 4,000 solver-generated samples; the relative state error, measured as the error norm divided by the reference-state norm, dropped from $0.07189$ to $0.03207$, while the objective-rank correlation reached $0.99981$. Poisson uses 5,000 FEM-generated operator

samples; the selected width-128 model reaches objective Spearman rank correlation (the correlation of the predicted and numerical objective ranks) $0.99157$, compared with $0.97602$ before the coverage expansion. A larger width-256, depth-3 alternative reaches $0.98402$, so network size alone does not account for the improvement. These numbers are collected in Table 19.

Table 19: **Operator-model validation on BPN search distributions.**

| Benchmark | Earlier model | Final model | Ranking statistic |
|---|---|---|---|
| Heat2D | relative state $L_2 = 0.07189$ | relative state $L_2 = 0.03207$ | objective rank correlation $0.99981$ |
| Poisson | Spearman $0.97602$ | Spearman $0.99157$ | larger alternative: $0.98402$ |

**Fresh-FEM irregular verification.** The corrected irregular comparison between direct RNO and SWAE uses the same 1,000 paired cells per PDE, the same geometries, controls, checkpoints, and matched proxy budgets: 127 calls for Heat, 247 for Darcy, and 656 for Advection. Every FEM verification is rebuilt from scratch for the current geometry and control: the operator is assembled, a new factorization is computed and the system is solved, the true state and objective are computed, and the solve state is discarded before the next call. The direct route performs one such FEM evaluation after proxy search; the SWAE route reevaluates the proxy top five with five independent fresh FEM solves. No assembled operator, sparse factorization, or FEM result is shared between verification calls, and all accepted controls satisfy maximum constraint violation at most $10^{-6}$.

The fresh-FEM comparison and the 65-call optimizer study answer different questions. The former compares the direct and SWAE routes under matched proxy budgets with explicit final reranking; the latter fixes the latent representation and compares optimizers under 65 actual candidate evaluations. Keeping these protocols separate avoids mixing the cost of numerical verification with the number of search evaluations.

# F Sensitivity and Variability

The conclusions in the main paper are based on repeated searches rather than single optimizer runs. We therefore report sample standard deviations together with the corresponding means and examine how the results change with the online evaluation budget, latent dimension, optimizer choice, structured representation, target distribution, and mesh geometry. For the regular BPN sensitivity studies, each budget cell contains three held-out targets and three search seeds ($n = 9$). For the irregular 80-call study, the reported dispersion is the sample standard deviation over the pooled FEM test set.

## F.1 Online evaluation budget

The BPN budget study keeps the target set, SWAE model, operator surrogate, and search hyperparameters fixed while changing only the number of proxy evaluations. Table 20 reports both DA and DANTE under the same three budget levels. DANTE evaluates children in latent-dimension-sized batches, so its actual call counts are the largest attainable values below the nominal ceilings.

Heat2D improves steadily as the budget grows for both optimizers, with the largest gain occurring between the low and medium budgets. Poisson changes much less: the mean objective moves only slightly over a fourfold increase in calls, while the standard deviation remains close to one objective unit because the three target fields have different intrinsic difficulty. NS2inlets is also stable at this resolution. Its mean does not improve monotonically with additional calls, and the changes across budgets are of the same order as the run-to-run dispersion. A single frozen budget is therefore sufficient for the benchmark-level comparisons used in the main paper.

## F.2 Latent dimension and optimizer choice

The latent dimension is selected before online optimization. NS2inlets provides the clearest controlled sweep because its control bank is split at the target level and evaluated with five-fold cross-validation.

Table 20: **Sensitivity to the online proxy-evaluation budget on the BPN benchmarks.** Entries are final numerical objectives, mean±sample standard deviation over 3 targets×3 search seeds; lower is better. DANTE call counts are the actual attainable counts under dimension-sized tree expansions.

| Benchmark | Search | Low | Medium | High |
|---|---|---|---|---|
| Heat2D | DA | 100: $(3.820 \pm 0.571) \times 10^{-2}$ | 200: $(3.699 \pm 0.511) \times 10^{-2}$ | 500: $(3.645 \pm 0.460) \times 10^{-2}$ |
| Heat2D | DANTE | 97: $(3.672 \pm 0.522) \times 10^{-2}$ | 197: $(3.608 \pm 0.476) \times 10^{-2}$ | 497: $(3.589 \pm 0.465) \times 10^{-2}$ |
| Poisson | DA | 50: $4.9511 \pm 1.0919$ | 100: $4.9551 \pm 1.0465$ | 200: $4.9295 \pm 1.0817$ |
| Poisson | DANTE | 49: $5.2424 \pm 1.0782$ | 97: $5.1621 \pm 1.0674$ | 193: $5.1270 \pm 1.0347$ |
| NS2inlets | DA | 50: $(3.110 \pm 0.380) \times 10^{-2}$ | 100: $(3.229 \pm 0.670) \times 10^{-2}$ | 200: $(3.455 \pm 0.788) \times 10^{-2}$ |
| NS2inlets | DANTE | 49: $(3.276 \pm 0.731) \times 10^{-2}$ | 97: $(3.253 \pm 0.666) \times 10^{-2}$ | 193: $(3.322 \pm 0.685) \times 10^{-2}$ |

As shown in Table 21, increasing the latent dimension from 4 to 16 reduces all three held-out reconstruction measures. The selected 16-D model has the lowest reconstruction MSE, relative $L_2$ error (reconstruction-error norm divided by reference-control norm), and slope error, and is then used unchanged in the optimization experiments.

Table 21: **NS2inlets latent-dimension selection by five-fold target-level cross-validation.** All quantities are computed on held-out optimized controls. Here $d_z$ is the latent dimension, and slope MAE is the mean absolute error of the reconstructed control slopes.

| $d_z$ | Validation MSE | Relative $L_2$ | Slope MAE | Selected |
|---|---|---|---|---|
| 4 | 0.01569 | 0.12364 | 0.14743 | no |
| 8 | 0.01092 | 0.09559 | 0.11554 | no |
| 12 | 0.01191 | 0.09761 | 0.11691 | no |
| 16 | 0.01020 | 0.08994 | 0.11116 | yes |

The representation comparison in the main paper uses a fixed optimizer to isolate the effect of the search coordinates. Table 22 repeats the comparison with a second, non-DA optimizer chosen independently for each benchmark. The overall pattern remains similar. On Burgers and the three BPN problems, SWAE is the strongest entry under the alternative optimizer; on Voltage, DCT and SWAE are effectively tied under exact DANTE. The representation advantage is therefore not specific to DA, while the Voltage result also shows that no single coordinate system must dominate every representation–optimizer pair.

Table 22: **Representation comparison under a second optimizer.** Each row fixes the indicated non-DA optimizer and changes only the search coordinates. Entries are final numerical objectives, mean±sample standard deviation; lower is better. Voltage uses exact DANTE. Poisson DCT is omitted here because its completed spectral check uses the separate 193-call plus top-5 protocol.

| Benchmark | Fixed optimizer | Raw | DCT | PCA | SWAE |
|---|---|---|---|---|---|
| Voltage | DANTE | – | $(5.18 \pm 5.20) \times 10^{-3}$ | $(6.59 \pm 5.57) \times 10^{-3}$ | $(5.22 \pm 6.76) \times 10^{-3}$ |
| Burgers | CMA-ES | $(9.77 \pm 8.83) \times 10^{-2}$ | $(8.06 \pm 7.01) \times 10^{-3}$ | $(7.46 \pm 7.78) \times 10^{-3}$ | $(7.03 \pm 9.04) \times 10^{-3}$ |
| Heat2D | DANTE | $(9.29 \pm 2.94) \times 10^{-2}$ | $(3.67 \pm 0.51) \times 10^{-2}$ | $(3.62 \pm 0.48) \times 10^{-2}$ | $(3.59 \pm 0.47) \times 10^{-2}$ |
| Poisson | CMA-ES | $15.8065 \pm 8.6440$ | – | $4.9612 \pm 0.9360$ | $4.9147 \pm 1.1024$ |
| NS2inlets | DANTE | $1.3118 \pm 0.5806$ | $(4.66 \pm 1.10) \times 10^{-2}$ | $(3.99 \pm 0.79) \times 10^{-2}$ | $(3.32 \pm 0.68) \times 10^{-2}$ |

### F.3 Representation regularization under target shift

We compare AE and SWAE on NS2inlets under both the in-distribution official target and a held-out modulated target. Both representations use the same historical train30 control bank, 62–128–64–16–64–128–62 architecture, slope-reconstruction term, and decoder constraints; the AE removes only the sliced-Wasserstein regularizer. In every run, DA receives 200 proxy evaluations, after which the top three candidates are reevaluated by the FEM solver. Table 23 reports the resulting FEM objectives over seeds 0, 1, and 42.

Table 23: **Matched AE/SWAE comparison on NS2inlets in and beyond the training-target distribution.** Entries are final FEM objectives, mean±sample standard deviation over three search seeds; lower is better. All cells use the same control bank, DA optimizer, 200-call proxy budget, and top-3 FEM confirmation setting.

| Representation | ID: official target | OOD: modulated target |
|---|---|---|
| AE | $0.02919 \pm 0.00597$ | $0.04948 \pm 0.01137$ |
| SWAE | $\mathbf{0.02429 \pm 0.00140}$ | $\mathbf{0.04059 \pm 0.00104}$ |

Under this matched comparison, SWAE lowers the mean FEM objective relative to AE on both the official target and the held-out modulated target. Its across-seed dispersion is also smaller in both conditions. Thus, on NS2inlets, the benefit of distribution-regularized latent coordinates is not confined to the shifted target; it appears in-distribution and remains under the tested target shift. Because this comparison uses one benchmark and three seeds, it supports this controlled representation claim rather than a general claim of OOD robustness.

### F.4 Irregular representations and geometry shifts

The irregular suite introduces a second representation choice before SWAE: DCT, PCA, or graph-spectral coordinates. Table 24 reports the complete 80-call IID comparison under a common candidate budget. Heat and Darcy favor the DCT family. Advection behaves differently: direct raw RNO search is strong when every method receives exactly 80 candidate evaluations, and among the learned-manifold routes GFT→SWAE is the strongest. The upstream coordinates therefore remain PDE dependent even when the same 32-D manifold model is used downstream.

Table 24: **Irregular representation sensitivity under the 80-call IID candidate budget.** Entries are exact-FEM gain mean±sample standard deviation over pooled test samples; higher is better.

| Method | Heat | Darcy | Advection |
|---|---|---|---|
| FEM adjoint | $0.2565 \pm 0.0837$ | $0.3454 \pm 0.1332$ | $0.1936 \pm 0.0666$ |
| FEM latent DCT | $0.1488 \pm 0.0557$ | $0.2018 \pm 0.0846$ | $0.0811 \pm 0.0337$ |
| FEM latent PCA | $0.1221 \pm 0.0551$ | $0.1657 \pm 0.0818$ | $0.0761 \pm 0.0316$ |
| FEM latent GFT | $0.1292 \pm 0.0623$ | $0.1759 \pm 0.0916$ | $0.0870 \pm 0.0402$ |
| RNO direct DCT | $0.1070 \pm 0.0371$ | $0.1387 \pm 0.0540$ | $0.0748 \pm 0.0258$ |
| RNO direct PCA | $0.1203 \pm 0.0422$ | $0.1568 \pm 0.0635$ | $0.0681 \pm 0.0256$ |
| RNO direct GFT | $0.0935 \pm 0.0494$ | $0.1198 \pm 0.0680$ | $0.0556 \pm 0.0282$ |
| RNO DCT→SWAE | $0.1311 \pm 0.0537$ | $0.1721 \pm 0.0757$ | $0.0572 \pm 0.0276$ |
| RNO PCA→SWAE | $0.1042 \pm 0.0520$ | $0.1377 \pm 0.0743$ | $0.0567 \pm 0.0262$ |
| RNO GFT→SWAE | $0.1122 \pm 0.0594$ | $0.1466 \pm 0.0831$ | $0.0669 \pm 0.0355$ |
| RNO raw | $0.0720 \pm 0.0449$ | $0.1116 \pm 0.0591$ | $0.1258 \pm 0.0445$ |

The same FEM testbed also lets us change the mesh distribution while holding the PDE family and 80-call exact latent-search protocol fixed. Table 25 uses the strongest structured exact latent basis on IID meshes—DCT for Heat and Darcy, and GFT for Advection—and repeats it under density, boundary-shape, interior-hole, and L-shaped geometry changes. Density and smooth boundary-shape changes leave the mean gain close to the IID value. The interior hole is substantially harder for Heat and Darcy, whereas Advection changes only modestly under the hole and shape shifts. The L-shaped meshes do not produce a uniform degradation: Heat and Darcy improve relative to IID, while Advection decreases moderately. Geometry therefore changes the difficulty of the irregular optimization problem in a PDE-dependent way rather than introducing a single universal shift.

Table 25: **Sensitivity to irregular-mesh geometry.** The search method is exact structured latent search with 80 candidate evaluations: DCT for Heat and Darcy, and GFT for Advection. Entries are exact-FEM gain mean±sample standard deviation over pooled samples; higher is better.

| PDE | IID | Density | Shape | Hole | L-shape |
|---|---|---|---|---|---|
| Heat | $0.1488 \pm 0.0557$ | $0.1466 \pm 0.0522$ | $0.1475 \pm 0.0540$ | $0.0668 \pm 0.0354$ | $0.1621 \pm 0.0748$ |
| Darcy | $0.2018 \pm 0.0846$ | $0.2001 \pm 0.0891$ | $0.1917 \pm 0.0858$ | $0.0973 \pm 0.0537$ | $0.2595 \pm 0.1304$ |
| Advection | $0.0870 \pm 0.0402$ | $0.0819 \pm 0.0394$ | $0.0945 \pm 0.0411$ | $0.0899 \pm 0.0420$ | $0.0749 \pm 0.0278$ |

# G Numerical Verification and Computational Cost

The framework uses learned models in two places: a learned representation may define the search coordinates, and a neural operator may score candidates during optimization. Neither changes the numerical definition of solution quality. Every final control reported in the paper is evaluated by a fixed numerical PDE solver, and the learned evaluator is used only to reduce the cost of proposing or ranking candidates. This section first summarizes the numerical verification path and then accounts for the one-time computation required before repeated online optimization.

## G.1 Numerical solvers and final-control verification

The three source codebases motivate different verification requirements. Voltage and Burgers already contain numerical dynamics that can serve as the final plant. The released BPN optimization path is built around PINN objectives implemented in DeepXDE, so we implemented independent numerical solvers that can rescore arbitrary controls under one common discretization. The irregular suite is rebuilt around our own unstructured FEM solver; RNO is an online evaluator, while the reported objective is always recomputed by FEM. The complete PDE definitions and solver details are given in Appendix C.2; Table 26 summarizes the final scoring path.

Table 26: **Reference solver used for every reported final control.** A neural evaluator may guide the search, but it is never the source of the reported final quality.

| Benchmark | Search may use | Reference solver | Final numerical selection |
|---|---|---|---|
| Voltage | DeepONet | CN–FD | top-5 PDE replay for surrogate search |
| Burgers | learned short-horizon propagator | upwind convection + CN diffusion | numerical execute–replan throughout the trajectory |
| Heat2D | DeepONet | five-point CN–FD | top-5 numerical reevaluation |
| Poisson | DeepONet | P1 triangular FEM | top-5 FEM reevaluation |
| NS2inlets | DeepONet | Taylor–Hood FEM | top-3 fine-mesh FEM reevaluation |
| Irregular Heat, Darcy, and Advection | RNO | unstructured FEM | fresh FEM reevaluation of the selected candidate(s) |

The independent BPN solvers were checked against the released formulations before any comparison was run. For Heat2D, the checks cover the diffusion coefficient, the official target amplitude, and the equivalence between the released uniform space–time MSE and our integral objective. For Poisson, we checked the PDE sign, outer and hole boundary conditions, central control support, and objective convention. For NS2inlets, unit tests cover the mesh, control, and objective; additional checks compare the discrete-adjoint directional derivative with central finite differences and test mesh and nonlinear convergence, including flux behavior. Together, these checks define the numerical reference independently of the learned search model.

**Scope of final-control validation.** The numerical checks extend beyond the headline tables. Table 27 summarizes their scope. For Voltage, the top-5 procedure replays five candidates for each of 792 evaluation runs, giving 3,960 true-PDE candidate evaluations; all 396 exact-evaluator runs retain

the original rank-1 solution. For the BPN benchmarks, we recomputed 297 selected controls from the Heat2D, Poisson, and NS2inlets studies. Burgers executes each of the 36 target–seed trajectories with the numerical PDE rather than accepting an open-loop surrogate rollout. For the irregular suite, the 1,000 paired cells per PDE are evaluated with a newly assembled FEM solve for each final check.

Table 27: **Scope of numerical validation.**

| Benchmark | Validated scope | Validation performed |
|---|---|---|
| Voltage | 792 runs; 3,960 candidate replays | top-5 true-PDE replay; all 396 exact runs retain rank 1 |
| Burgers | 12 targets $\times$ 3 seeds | numerical execution and state re-anchoring at every replanning step |
| Heat2D | 108 retained selected controls | stored fine-solver objective and control-range consistency |
| Poisson | 90 retained selected controls | deterministic $65 \times 65$-background FEM recomputation |
| NS2inlets | 99 retained selected controls | selected control, confirmation record, and fine-FEM objective agree |
| Irregular suite | 1,000 paired cells per PDE | independent fresh FEM solve(s) for every final verification |

**Fresh-FEM comparison on irregular benchmarks.** For the corrected irregular comparison, "FEM verification" means a genuinely fresh numerical call: the operator is assembled for the current geometry and control, a new factorization is computed and the system is solved, the state and objective are computed, and the solve state is discarded before the next call. No sparse factorization, assembled operator, or previous FEM result is reused. Direct RNO with DCT performs one fresh FEM evaluation after proxy search; the SWAE route evaluates the proxy top five with five independent fresh FEM solves. Both routes use the same paired cells, checkpoints, and proxy budgets.

Table 28 reports the resulting wall times. The 65-FEM column is a numerical-solver baseline obtained from the measured single-call cost; the direct and SWAE columns are complete measured online times, including their respective final FEM verification calls. The SWAE route spends additional time on four extra FEM confirmations and obtains a higher verified gain on all three PDEs.

Table 28: **Fresh-FEM timing on the irregular suite.** Direct and SWAE use matched proxy-search budgets: 127 calls for Heat, 247 for Darcy, and 656 for Advection. Direct uses one final FEM call and SWAE uses top-5 FEM verification. Gain is final true-FEM objective minus the paired initial objective; higher is better. The last two columns report gain followed by online time in seconds in parentheses.

| PDE | Single fresh FEM | 65$\times$ fresh FEM | Direct RNO with DCT Gain (seconds) | SWAE top-5 Gain (seconds) |
|---|---|---|---|---|
| Heat | 0.0582 s | 3.7830 s | 0.132697 (1.0676) | $\mathbf{0.166994}$ (1.7408) |
| Darcy | 0.0584 s | 3.7939 s | 0.217110 (1.9665) | $\mathbf{0.262415}$ (2.9047) |
| Advection | 0.1218 s | 7.9165 s | 0.157040 (4.7824) | $\mathbf{0.187714}$ (6.5321) |

This fresh-FEM comparison is separate from the matched-wall-time headline study: it compares direct RNO with DCT with surrogate-guided SWAE under matched proxy budgets and explicit one-versus-five FEM verification. The exact-latent route was not rerun under this fresh-FEM protocol.

### G.2 Offline computational cost

The repeated-task setting gives the offline computation a particularly simple interpretation. The SWAE training set consists of controls produced by *previously solved optimization tasks*. Those controls are the experience being reused; they are not regenerated for every new target. Once the control bank exists, fitting the search geometry is a one-time model-fitting cost. Exact latent search requires no neural PDE surrogate at all. Surrogate-guided search optionally adds solver-generated operator data and one operator-model fit, which are likewise reused across subsequent targets.

It is therefore useful to separate three quantities:

$$C_{\text{offline}} = C_{\text{rep}} + \mathbf{1}_{\text{sur}} \left( C_{\text{op-data}} + C_{\text{op-fit}} \right), \tag{85}$$

where $C_{\text{rep}}$ is the cost of fitting the reusable control representation and the second term is present only when a learned evaluator is used. Historical task solves are the source of the control bank and are shared across future tasks; they are not charged again at test time.

**Hardware and timing convention.** Graphics processing unit (GPU) model fitting was performed on a workstation containing **8 NVIDIA A800-SXM4-80GB GPUs**. Training processes use one visible GPU per job; sweep scripts use the remaining devices to run independent configurations or seeds in parallel rather than using model-parallel training. The wall times in Table 29 are the per-job times recorded for the experiments. This convention makes the reported number a property of one training job rather than the aggregate throughput of the eight-GPU machine.

Table 29: **Recorded one-time model-fitting wall times on the A800-SXM4-80GB workstation.** Selection sweeps and CV are reported when those are the recorded costs used to choose the final representation. The irregular entries include the RNO and latent representation fit.

| Benchmark | Recorded offline component | Seconds | Minutes |
|---|---|---|---|
| Voltage | final SWAE fit | 96.98 | 1.62 |
| Burgers | final phase-conditioned SWAE fit | 2081.89 | 34.70 |
| Heat2D | complete 24-model SWAE selection sweep | 27.55 | 0.46 |
| Heat2D | final DeepONet fit | 441.20 | 7.35 |
| Poisson | complete 48-model SWAE selection sweep | 526.85 | 8.78 |
| NS2inlets | 5-fold SWAE CV + final refit | 263.77 | 4.40 |
| NS2inlets | final DeepONet fit | 211.16 | 3.52 |
| Irreg. Heat | RNO + DCT→SWAE fit | 534.54 | 8.91 |
| Darcy | RNO + DCT→SWAE fit | 501.59 | 8.36 |
| Advection | RNO + GFT→SWAE fit | 515.25 | 8.59 |

The directly timed fitting costs are small relative to the lifetime of a repeated-task benchmark. The largest representation fit is Burgers at 34.7 minutes; it trains on tens of thousands of phase-conditioned $20 \times 81$ windows. Every other timed representation-selection or representation-plus-RNO fit in Table 29 is below nine minutes. In the irregular suite, moving from direct RNO to the latent RNO route adds only about 72–76 seconds of offline fitting per PDE. The timed Heat2D and NS2inlets operator fits take 7.35 and 3.52 minutes, respectively.

The table includes only wall times recorded for the corresponding jobs. Some operator checkpoints, including the final Poisson DeepONet and the PDE-OP surrogate checkpoints, do not retain a comparable training-time value and are therefore omitted from the numerical total rather than estimated. Solver-data generation was also performed in several stages rather than under one common timing procedure. These costs affect only the optional surrogate path; the exact latent method uses the historical control bank and the fitted representation alone.

### G.3 Amortization across repeated tasks

For $M$ future tasks, a one-time investment in the representation and operator contributes only $1/M$ of its cost to the average task:

$$\overline{C}_{\text{ours}}(M) = C_{\text{online}} + \frac{C_{\text{offline}}}{M}. \tag{86}$$

If a reference method costs $C_{\text{base}}$ per task and $C_{\text{base}} > C_{\text{online}}$, the usual break-even task count is

$$M_{\text{break}} = \left\lceil \frac{C_{\text{offline}}}{C_{\text{base}} - C_{\text{online}}} \right\rceil. \tag{87}$$

The important point for our setting is that $C_{\text{rep}}$ is already small, while the control bank comes from optimization work performed on previous tasks. The online savings are therefore amortized over the same stream of related PDE-control problems that supplies the reusable search geometry.

This accounting also separates the two roles of learning in the framework. The representation is the core repeated-task investment and is sufficient for exact latent search. A neural operator is an optional second investment whose value depends on the cost of the numerical PDE solve. When exact evaluation is already inexpensive, the exact latent route avoids this extra offline and online machinery; when PDE evaluation dominates, the operator fit is reused to make many subsequent candidate evaluations cheaper.

## H LIMITATIONS AND OUTLOOK

The framework has a clear scope. It is most natural when related PDE-control problems are solved repeatedly, so that previous optimization runs leave behind a reusable set of high-quality controls. In a genuinely one-off problem with no historical control data, constructing a control bank and fitting a search geometry may not be worth the additional preparation; direct numerical optimization can be the simpler choice. Our target setting is the repeated one: the plant, geometry, or governing physics may remain largely fixed while setpoints, targets, operating conditions, or design requirements change over time. In such settings, yesterday's optimization results need not be discarded after one use.

The method also needs a source of optimized controls. It is therefore not a zero-data method. At the same time, the amount of control data required in our experiments is modest by modern learning standards: useful manifolds were learned from hundreds to a few thousand optimized controls or trajectories, rather than from massive pretraining corpora. These controls can be collected from direct optimization, MPC, NMPC, adjoint methods, or other solvers already used in the application. What matters is that they capture enough successful solutions to reveal reusable structure. Reducing this requirement further is an important direction. In particular, it would be valuable to learn control regularities indirectly from existing PDE solution or trajectory datasets, rather than requiring an explicit bank of optimized controls.

The present method is not zero-shot across unrelated physical systems. A new PDE family currently requires its own control experience, and surrogate-guided search additionally requires operator data generated for that system. We see two longer-term routes beyond this limitation. One is to infer optimization structure from much more widely available PDE state data: if recurring solution patterns can be mapped back to the controls that produce them, useful search geometry may be recoverable without first solving a large collection of control problems. The other is the emergence of physics foundation models or implicit world models that represent many physical systems in a shared model. Such a model would be substantially more valuable for optimization if it exposed not only transferable state prediction, but also transferable coordinates in which control can be searched.

A second open question is theoretical. The numerical results in this paper show that optimized controls often occupy a much more useful low-dimensional region than the ambient control space, but we do not yet have a theory that predicts when such a region should exist, what its intrinsic dimension should be, or how its geometry depends on the PDE, target distribution, constraints, and controllability. The current evidence is therefore primarily empirical. A theory connecting learned control geometry to reachable sets, solution operators, or other structural properties of PDE-constrained optimization would make the idea substantially sharper. We view the experiments here as a starting point for that question rather than its resolution.

Finally, this paper does not attempt to introduce a new universal PDE solver or a new optimization mechanism that replaces existing methods. Its main claim is more specific: *where to search* can be treated as an independent design axis, separate from how candidates are optimized and how they are evaluated. SWAE is one realization of that axis, not the only one. Other representations, online-updated manifolds, task-conditioned geometries, and search methods that co-design the representation and optimizer may prove better in different regimes. Our hope is that making this axis explicit creates room for those alternatives, and that future work treats reusable search geometry as a first-class object in PDE-constrained optimization rather than as an incidental preprocessing choice.