DIAL: ALIGNING LOCAL GRADIENTS AND GLOBAL TRANSPORT FOR FEW-SHOT OFFLINE OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Few-shot offline optimization aims to discover high-performing designs (e.g., materials) from sparse labeled data via two main approaches: surrogate and generative methods. The former learns local improvement directions but might not extrapolate well out of the observed data regime. The latter captures global mass-transport structure between low- and high-performing design regimes but might induce generative paths inconsistent with local improvement directions. Despite operating at different scales, we show that these approaches are coupled. Integrating the differential changes in surrogate prediction along a mass-transport path must recover the oracle value difference between its endpoints. Therefore, violations of this relation provide a cooperative correction signal between the surrogate gradient field and generative transport, allowing each model to expose and correct the other's generalization errors and improve overall performance. Yet, in few-shot tasks, this coupling cannot be learned well from the target task alone because sparse and small labeled data provide little access to the differential information required along such paths. To address this challenge, we introduce a **Differential-to-Integral ALignment** (DIAL) meta optimization framework that learns this coupling across a population of similar functions whose behaviors are consistent with the observed target data. The learned coupling can then be transferred to the target task to enrich its few-shot data with shared improvement structures. Our experiments across eight benchmark tasks demonstrate consistent improvements over SOTA methods.

1 INTRODUCTION

Offline black-box optimization arises in many science and engineering domains where experiments are costly and optimization must therefore rely on previously collected data. The goal is to discover new designs (e.g., materials, proteins, and drugs) that outperform previously evaluated candidates without the budget for additional data collection (Trabucco et al., 2022; Kim et al., 2025; Shin et al., 2025; Fadhel et al., 2026a). However, existing offline optimization methods have largely focused on large-data regimes, with limited attention to few-shot settings (Nguyen et al., 2023; Fadhel et al., 2026b). Existing approaches can be largely categorized into two main groups that adopt different views of the unknown objective function $g(\boldsymbol{x})$. Surrogate-based methods learn a surrogate model $g_\phi(\boldsymbol{x}) \simeq g(\boldsymbol{x})$ and use its local gradients $\nabla g_\phi(\boldsymbol{x})$ to infer local directions of improvement (Hoang et al., 2024; Trabucco et al., 2021; Tan et al., 2025; Chen et al., 2024). Generative methods learn a distribution over global transport paths $\boldsymbol{x}'(t) = \boldsymbol{u}_\theta(\boldsymbol{x}(t), t)$ that move from low-value designs toward high-value regions (Dao et al., 2025; Krishnamoorthy et al., 2022; 2023; Yun et al., 2024).

The former captures fine-grained differential information about how design performance changes under local perturbations while the latter captures global transport structure over the design space. However, both approaches *can perform poorly when experimental data are scarce* (e.g., drug design). The surrogate gradients provide local improvement geometry, but in the small-data regime they can be overly conservative and become trapped in suboptimal regions due to limited knowledge of the global landscape. Generative approaches provide global guidance toward high-value regions, but in the small-data regime their learned transport can be weakly constrained by the objective's local improvement geometry, potentially leading to unreliable behavior.

Research Gap. These local and global views of the same function landscape are often learned independently, overlooking a fundamental relation between them. This disconnect is significant in

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

the small-data regime, where neither view alone has sufficient information to reliably characterize the objective landscape. As shown later, exploiting their underlying relation provides a cooperative mechanism through which the complementary information captured by one view can constrain the other, identifying and correcting inconsistencies between the local and global views. It provides a principled safeguard against unreliable surrogate predictions and generative transport, allowing for a natural integration of their complementary optimization strengths to improve optimization performance (Section 4).

To see this, consider a sampled generative path $\boldsymbol{x}'(t) = \boldsymbol{u}_\theta(\boldsymbol{x}(t), t)$ that transports a design from $\boldsymbol{x}(0)$ to $\boldsymbol{x}(1)$. Along this path, the surrogate predicts the local change in performance as $\mathrm{d}g_\phi(\boldsymbol{x}(t))/\mathrm{d}t = \nabla g_\phi(\boldsymbol{x}(t))\boldsymbol{x}'(t)$. If the local surrogate and global transport faithfully describe the same objective landscape, integrating these local changes along the path should recover the oracle performance difference $g(\boldsymbol{x}(1)) - g(\boldsymbol{x}(0))$:

$$g(\boldsymbol{x}(1)) - g(\boldsymbol{x}(0)) \quad \simeq \quad \int_0^1 \nabla g_\phi(\boldsymbol{x}(t))^\top \boldsymbol{x}'(t)\mathrm{d}t \quad = \quad \int_0^1 \nabla g_\phi(\boldsymbol{x}(t))^\top \boldsymbol{u}_\theta(\boldsymbol{x}(t), t)\mathrm{d}t \,. \tag{1}$$

Remark. Note that via path integration, the RHS of Eq. 1 equals $g_\phi(\boldsymbol{x}(1)) - g_\phi(\boldsymbol{x}(0))$. However, this does not eliminate the coupling with the generative model because the endpoints reached by the learned transport are themselves determined by θ. Although Eq. 1 alone cannot distinguish between paths connecting the same two endpoints, our pre-optimizing stage (as discussed later) fits the coupling across many endpoint pairs and diverse local path segments (Section 3.1), providing additional evidence to disambiguate the gradient–transport interaction before target-task adaptation.

Intuition. The key intuition is that when a surrogate gradient field and a generative transport both faithfully describe the same objective landscape, their local interactions should satisfy the consistency relation in Eq. 1. Violation of this relation therefore exposes an inconsistency between the local differential geometry and global transport structure, suggesting that either the surrogate is extrapolating unreliably or the generative model is routing designs through implausible regions. Minimizing the expected violation encourages the two models to coordinate their local interactions, allowing their complementary local and global information to constrain each other when data are scarce.

Technical Challenge. Estimating this consistency violation in practice requires sampling generative paths whose endpoints belong to the offline dataset so that the oracle value difference between them can be observed. Otherwise, an unconditioned generative path is generally unlikely to terminate at an observed design. Each endpoint-conditioned path however exposes only one local realization of the generative transport and its interaction with the surrogate's gradient field. This is a non-trivial challenge in the few-shot regime, where the small number of labeled designs provides only a limited number of endpoint pairs and therefore sparse coverage of the possible transport behavior.

Solution Approach. Rather than learning the differential coupling solely from scarce target-task data, we introduce a meta-optimization framework that leverages a population of closed-form functions exhibiting similar behavior to the unknown oracle around the observed offline data region. Their closed forms make both the performance differences between sampled design pairs and the corresponding local differential information readily available, enabling differential coupling to be optimized across diverse generative paths and local gradient geometries. The learned population-level structure can then be transferred to the target task, where it supplements the limited offline observations.

Contributions. To realize the above solution vision, we introduce **Differential-to-Integral ALignment** (DIAL), a novel meta-optimization framework for few-shot offline optimization. DIAL minimizes the expected violation of the consistency relation in Eq. 1 through two stages: (i) population-level pre-optimization via meta differential alignment and (ii) target-task transfer optimization via differential-to-integral alignment as described below.

1. Population-level pre-optimization via meta differential alignment. We adapt model-agnostic meta learning (MAML) (Finn et al., 2017b) to a gradient-matching loss that pre-optimizes the surrogate gradient field, while learning the generative transport via meta flow matching, both over a population of related functions constructed by fitting different Gaussian process (Williams & Rasmussen, 2006; Fadhel et al., 2026b; Dao et al., 2025) priors to the same sparse offline data. We then adapt MAML to a simplified step-wise variant of the differential alignment loss in Eq. 1 using the closed-form functions. This avoids the need to estimate the full path-wise integral relation during pre-optimization. Optimizing these meta losses produces a population-level differential coupling that

encodes transferable improvement structure shared across the function population and provides a stronger initialization for subsequent target-task optimization (Section 3.1).

2. Target-task transfer optimization via integral alignment. We transfer the population-level differential coupling to the target task at the integral level, constraining the accumulated predicted differential performance changes along paths connecting observed designs to match the observed performance difference between their endpoints. To make this integral relation observable, we develop a novel endpoint-conditioned sampling method that first samples terminal designs according to the learned transition distribution and then samples generative paths connecting the selected offline endpoints via Doob's H-transform (Corstanje et al., 2023). This enables the pre-optimized differential coupling to be further adapted at the integral level using the scarce target-task data (Section 3.2).

3. Empirical evaluation. We conduct extensive experiments across eight few-shot offline optimization benchmarks (Trabucco et al., 2022; Qian et al., 2025) and demonstrate consistent improvements over strong existing methods. Our ablation studies further demonstrate the complementary benefits of population-level meta-training, differential alignment, and target-task integral alignment (Section 4).

2 Problem Setup and Preliminaries

Offline optimization (Trabucco et al., 2022) aims to discover a high-performing design $\boldsymbol{x}_*$ (e.g., material, drug) that maximizes a black-box function $g(\boldsymbol{x})$ using only an offline dataset $D = \{(\boldsymbol{x}_i, z_i)\}_{i=1}^n$, where $\boldsymbol{x}_i$ denotes a candidate design and $z_i = g(\boldsymbol{x}_i)$ is its oracle evaluation. Existing approaches to offline optimization can be categorized into two main groups as detailed below.

Surrogate Methods aim to learn a surrogate model $g_{\boldsymbol{\phi}}(\boldsymbol{x})$ from the offline data by solving

$$\boldsymbol{\phi}_\star \;\triangleq\; \arg\min_{\boldsymbol{\phi}} \sum_{i=1}^{n} \ell\big(g_{\boldsymbol{\phi}}(\boldsymbol{x}_i), z_i\big) \quad\Rightarrow\quad \boldsymbol{x}_* \;=\; \arg\max_{\boldsymbol{x}} g_{\boldsymbol{\phi}_*}(\boldsymbol{x})\,, \tag{2}$$

where $\ell(\cdot,\cdot)$ denotes the prediction loss and $\boldsymbol{\phi}$ represents the surrogate parameters. A key challenge is that the surrogate may produce inaccurate predictions for out-of-distribution (OOD) designs encountered during optimization. To mitigate this issue, prior work has proposed regularizing surrogate predictions on OOD inputs (Hoang et al., 2024; Tan et al., 2025; Trabucco et al., 2021).

Generative Methods instead learn inverse or generative mappings from desired performance levels to candidate designs (Krishnamoorthy et al., 2023). Recent approaches in this group can be represented by a conditional generative model $\textsc{gen}_\theta(\boldsymbol{x} \mid \gamma)$ defined through a stochastic differential equation (SDE) whose velocity $\boldsymbol{v}_\theta(\boldsymbol{x}(t), t; \gamma)$ is conditioned on a criterion γ such as $\gamma \,:\, z \geq z_o$:

$$\textsc{sde}_\theta(\boldsymbol{x}(t)):\quad \mathrm{d}\boldsymbol{x}(t) \;=\; \boldsymbol{v}_\theta(\boldsymbol{x}(t), t; \gamma)\,\mathrm{d}t \;+\; \sigma(t)\mathrm{d}\boldsymbol{w}(t) \quad \text{where} \quad \mathrm{d}\boldsymbol{w}(t) \sim \mathbb{N}(0, \boldsymbol{I}\mathrm{d}t)\,, \tag{3}$$

where $\sigma(t)$ is a time-dependent diffusion coefficient (Lipman et al., 2022; Song et al., 2020b; Lai et al., 2025). We can then sample a high-performing design $\boldsymbol{x}_*$ from the marginal $p_\theta^{0\to t}(\boldsymbol{x}(t))$ of the above SDE at $t = 1$ starting from $\boldsymbol{x}(0) \sim D$. The same marginal distribution can equivalently be characterized by the probability flow ODE associated with this SDE, as follows from the Fokker-Planck equation (Risken, 1989):

$$\textsc{ode}_\theta(\boldsymbol{x}(t)):\; \mathrm{d}\boldsymbol{x}(t)/\mathrm{d}t \;=\; \boldsymbol{u}_\theta\big(\boldsymbol{x}(t), t; \gamma\big) = \boldsymbol{v}_\theta\big(\boldsymbol{x}(t), t; \gamma\big) - \frac{1}{2}\sigma(t)^2 \nabla_{\boldsymbol{x}(t)} \log p_\theta^{0\to t}(\boldsymbol{x}(t)). \tag{4}$$

Since Eq. 3 and Eq. 4 induce identical state marginals, we employ the ODE trajectory $\boldsymbol{x}(t)$ and its velocity field $\boldsymbol{u}_\theta$ to characterize the generative transport throughout the paper.

Differential Coupling. As discussed in Section 1, surrogate and generative methods provide complementary local and global views of the same objective landscape, but these views are typically learned independently. Equation 1 provides a consistency relation that connects them. Substituting Eq. 4 into Eq. 1 makes explicit the differential interaction between the generative transport parameter θ and surrogate parameter ϕ along the transport trajectory $t \in [0, 1]$.

Few-Shot Optimization Challenge. In the few-shot optimization regime, labeled data for the target task are extremely scarce. A common setting assumes access to only a handful of evaluated designs, together with a larger pool of unevaluated candidates (Nguyen et al., 2023). This poses a severe challenge to learning the differential coupling in Eq. 1 because estimating its consistency violation

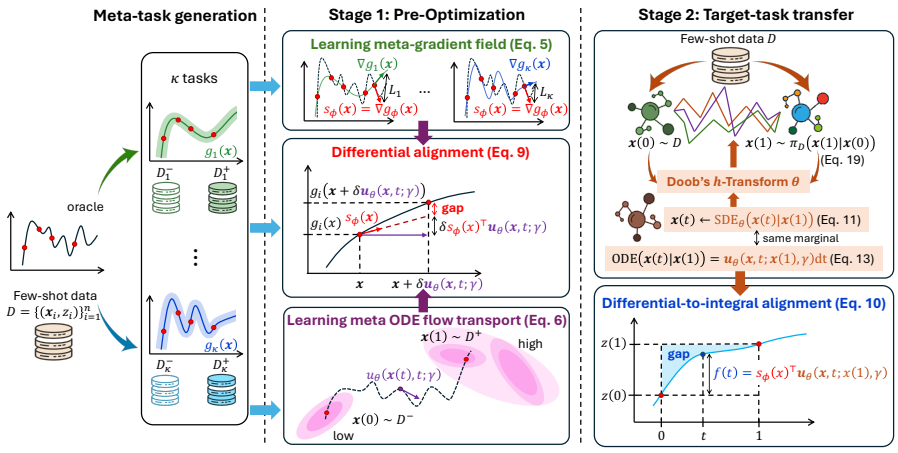


Figure 1: **Overview of our DIAL workflow.** (1) **Pre-optimization:** We construct a population of surrogate tasks from multiple Gaussian processes and jointly train the gradient model and meta-flow across these tasks, coupled through *meta-differential alignment*. (2) **Target transfer:** On the target task, we adapt the meta-flow and gradient model to the offline data using *differential-to-integral alignment*, aligning the learned local gradient information with global transport over the offline data.

requires generative paths whose endpoints are evaluated designs, so that their oracle performance differences are observable. With only a small number of evaluated designs, the number and diversity of such observable endpoint pairs are severely limited, providing sparse coverage of the differential coupling that must be learned.

3 Meta-Optimization via Differential-to-Integral Alignment Loss

We introduce **Differential-to-Integral ALignment** (DIAL), a meta-optimization framework for learning the differential coupling in Eq. 1 from few-shot data. The key challenge is that learning this coupling requires observing local interactions between the surrogate gradient field and generative transport along a diverse range of generative paths, whereas few-shot setting provides only a limited number of evaluated designs from which observable endpoint-conditioned paths can be constructed.

DIAL addresses this challenge using a two-stage approach: (i) population-level pre-optimization and (ii) target task adaptation. For population-level pre-optimization, we construct a population of closed-form functions that are consistent with the target function around the observed data region. As local differential information for these functions is readily accessible throughout the design space, we can use it to learn a shared gradient field and generative transport, and optimize their differential coupling across the function population. To enable target adaptation, we transfer and further fine-tune the pre-optimized coupling by requiring the accumulated differential changes along endpoint-conditioned paths to match the observed performance differences between evaluated designs. Figure 1 illustrates the DIAL workflow, consisting of population-level pre-optimization followed by target-task transfer.

3.1 Pre-Optimization via Meta Differential Alignment

Following prior practice in few-shot offline optimization (Fadhel et al., 2026b; Dao et al., 2025), we fit different GP priors to the same offline dataset which results in multiple GP posteriors. Their posterior means $g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \ldots, g_\kappa(\boldsymbol{x})$ form a population of closed-form functions with similar behavior around the offline data region (Williams & Rasmussen, 2006). We use this function population to learn (i) a shared meta-gradient field $\nabla g_\phi(\boldsymbol{x})$ that captures transferable structure shared across the gradient fields of the individual functions, and (ii) a shared generative transport $\boldsymbol{x}'(t) = \boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma)$ conditioned on γ that captures common low-value to high-value transport structure across the function population. Plugging (i) and (ii) into Eq. 1 consequently enables (iii) meta differential alignment, which pre-optimizes their differential coupling to capture transferable local interactions between gradient and transport structures across the function population. These three steps are detailed below.

(i) Learning meta-gradient field. We parameterize the meta gradient field $s_\phi(\boldsymbol{x}) = \nabla g_\phi(\boldsymbol{x})$ directly to capture shared local differential structure across the function population. For each (differentiable) GP posterior mean function $g_i(\boldsymbol{x})$, its individual gradient field $\nabla g_i(\boldsymbol{x})$ can be distilled into s_ϕ via minimizing the following model-agnostic meta learning (MAML) (Finn et al., 2017a) objective

$$\phi \;=\; \arg\min_{\phi} \frac{1}{\kappa}\sum_{i=1}^{\kappa} L_i\left(\phi - \alpha\nabla_\phi L_i(\phi)\right) \;\text{ where }\; L_i(\phi) \;=\; \mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_i}\left[\left\|\nabla g_i(\boldsymbol{x}) - s_\phi(\boldsymbol{x})\right\|^2\right], \quad (5)$$

where $\mathcal{P}_i$ contains designs obtained by gradient ascent/descent on g_i from offline data.

(ii) Learning meta ODE transport. Let D_i^- and D_i^+ denote the set of low- and high-value designs of $g_i(\boldsymbol{x})$, obtained by performing 100 steps of gradient descent/ascent initialize at the offline data. Let $D^+ = D_1^+ \cup \ldots \cup D_\kappa^+$ and $D^- = D_1^- \cup \ldots \cup D_\kappa^-$. We learn the meta ODE transport $\boldsymbol{x}'(t) = \boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma)$ between the population-level low- and high-value regimes represented empirically by D^- and D^+ by minimizing the following flow matching objective (Lipman et al., 2022; 2024):

$$\theta \;=\; \arg\min_{\theta}\; \mathbb{E}_{(\boldsymbol{x}(0),\boldsymbol{x}(1))\sim(D^-\times D^+)}\mathbb{E}_{t\in\mathbb{U}[0,1]}\left\|\boldsymbol{u}_\theta(\tilde{\boldsymbol{x}}(t), t; \gamma) \;-\; \tilde{\boldsymbol{x}}'(t)\right\|^2, \quad (6)$$

where $\tilde{\boldsymbol{x}}(t)$ is a Gaussian interpolant $\tilde{\boldsymbol{x}}(t) = \psi_t(\boldsymbol{x}_0, \boldsymbol{x}_1) = (1-t)\boldsymbol{x}(0) + t\boldsymbol{x}(1) + \sigma(t)\boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathbb{N}(\mathbf{0}, \boldsymbol{I})$ with instantaneous velocity $\tilde{\boldsymbol{x}}'(t) = \boldsymbol{x}(1) - \boldsymbol{x}(0) + \sigma'(t)\boldsymbol{\epsilon}$.

(iii) Pre-optimizing meta-differential coupling. Substituting $s_\phi(\boldsymbol{x})$ and $\boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma)$ into Eq. 1 allows us to pre-optimize their differential coupling across the function population via the following meta differential alignment objective:

$$(\phi, \theta) \;=\; \arg\min_{(\phi,\theta)} \frac{1}{\kappa}\sum_{i=1}^{\kappa} \mathcal{I}_i(\phi - \alpha\partial_\phi\mathcal{I}_i(\phi,\theta), \theta - \alpha\partial_\theta\mathcal{I}_i(\phi,\theta)), \quad (7)$$

where the differential alignment loss for each function $\mathcal{I}_i(\phi, \theta)$ is defined as

$$\mathcal{I}_i(\phi,\theta) \;\triangleq\; \mathbb{E}_{(\boldsymbol{x}(0),\boldsymbol{x}(1))}\Big(g_i(\boldsymbol{x}(1)) - g_i(\boldsymbol{x}(0)) - \mathbb{E}_{t\sim\mathbb{U}[0,1]}\left[s_\phi(\boldsymbol{x}(t))^\top \boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma)\right]\Big)^2, \quad (8)$$

where $\boldsymbol{x}(0) \sim D^-$ and $\boldsymbol{x}(1) = \textsc{solve}(\textsc{ode}_\theta(\boldsymbol{x}(0)))$, with $\boldsymbol{x}(t)$ denoting the resulting ODE trajectory. Since each closed-form function can be evaluated $g_i(\boldsymbol{x})$ at any input $\boldsymbol{x}$, we can optimize this alignment more directly at the differential level using local δ-step transitions, instead of integrating along the full trajectory from $t = 0$ to $t = 1$ in Eq. 8:

$$\mathcal{I}_i(\phi,\theta) \;\simeq\; \mathbb{E}_{\boldsymbol{x}\sim\mathcal{P}_i}\mathbb{E}^2_{t\sim\mathcal{U}[0,1]}\Big(g_i(\boldsymbol{x} + \delta\boldsymbol{u}_\theta(\boldsymbol{x}, t; \gamma)) - g_i(\boldsymbol{x}) - \delta s_\phi(\boldsymbol{x})^\top \boldsymbol{u}_\theta(\boldsymbol{x}, t; \gamma)\Big). \quad (9)$$

For a sufficiently small δ, $g_i(\boldsymbol{x} + \delta\boldsymbol{u}) - g_i(\boldsymbol{x}) \simeq \delta\nabla g_i(\boldsymbol{x})^\top \boldsymbol{u}$, so Eq. 9 directly aligns the change predicted by the meta-gradient field along the transport direction with the actual local change of g_i.

3.2 Target Transfer via Differential-to-Integral Alignment

The optimized meta differential coupling can then be transferred to the target task by initializing Eq. 8 with the learned (ϕ, θ) and restricting both endpoints to the observed target designs, i.e., $\boldsymbol{x}(0) \sim D$ and $\boldsymbol{x}(1) \sim D$. This confines the alignment loss to the target data regime, where it can be evaluated and further optimized, since the target performance difference $g(\boldsymbol{x}(1)) - g(\boldsymbol{x}(0))$ is only observed for endpoint pairs drawn from D.

However, Eq. 8 require the path $\boldsymbol{x}(t)$ starting from $\boldsymbol{x}(0)$ to terminate at the designated endpoint $\boldsymbol{x}(1)$. This condition was satisfied in Section 3.1, where the expectation was taken only over $\boldsymbol{x}(0)$ and the terminal state was generated directly as $\boldsymbol{x}(1) = \textsc{solve}(\textsc{ode}_\theta(\boldsymbol{x}(0)))$. This does not generally hold when $\boldsymbol{x}(1)$ is constrained to an observed design in D. Therefore, target transfer via Eq. 8 requires two additional components:

1. Construct an endpoint-conditioned path $\textsc{sde}_\theta(\boldsymbol{x}(t) \mid \boldsymbol{x}(1)) \sim \textsc{gen}_\theta(\boldsymbol{x}(t) \mid \gamma, \boldsymbol{x}(1))$ that connects $\boldsymbol{x}(0)$ to the designated observed endpoint $\boldsymbol{x}(1)$.

2. Construct a categorical distribution $\pi_D(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$ over candidate endpoints $\boldsymbol{x}(1) \in D$ according to their likelihood under the unconditioned transport from $\boldsymbol{x}(0)$.

Given these two components, the differential-to-integral alignment objective in Eq. 8 can be adapted to the target task as

$$\mathcal{I}(\phi,\theta) \;\triangleq\; \mathbb{E}\Big(g(\boldsymbol{x}(1)) - g(\boldsymbol{x}(0)) - \mathbb{E}_{t\sim\mathcal{U}[0,1]}\big[s_\phi(\boldsymbol{x}(t))^\top \boldsymbol{u}_\theta(\boldsymbol{x}(t),t;\boldsymbol{x}(1),\gamma)\big]\Big)^2 , \tag{10}$$

where $\boldsymbol{x}(0)\sim D$, $\boldsymbol{x}(1)\sim\pi_D(\boldsymbol{x}(1)\mid\boldsymbol{x}(0))$, and $\boldsymbol{u}_\theta(.;\boldsymbol{x}(1),\gamma)$ is the velocity field of the probability flow ODE associated with the endpoint-conditioned SDE. We construct $\text{SDE}_\theta(\boldsymbol{x}(t)\mid\boldsymbol{x}(1))$ and $\pi_D(\boldsymbol{x}(1)\mid\boldsymbol{x}(0))$ below.

3.2.1 Constructing Endpoint-Conditioned Paths via Doob's h-Transform

As $\text{SDE}_\theta(\boldsymbol{x}(t))$ in Eq. 3 and $\text{ODE}_\theta(\boldsymbol{x}(t))$ in Eq. 4 share the same state marginals (Risken, 1989), the SDE provides an equivalent stochastic representation of the generative transport $\text{GEN}_\theta(\boldsymbol{x}\mid\gamma)$. From this representation, the endpoint-conditioned $\text{SDE}_\theta\big(\boldsymbol{x}(t)\mid\boldsymbol{x}(1)\big)$ can be derived via Doob's h-Transform (Corstanje et al., 2023):

$$\text{SDE}_\theta\big(\boldsymbol{x}(t)\mid\boldsymbol{x}(1)\big):\mathrm{d}\boldsymbol{x}(t)=\big(\boldsymbol{v}_\theta\big(\boldsymbol{x}(t),t;\gamma\big)+\sigma(t)^2\nabla\log p_\theta^{t\to1}\big(\boldsymbol{x}(1)\mid\boldsymbol{x}(t)\big)\big)\,\mathrm{d}t+\sigma(t)\mathrm{d}\boldsymbol{w}(t) \tag{11}$$

which admits a probability flow ODE sharing the same state marginals:

$$\begin{aligned}\text{ODE}_\theta(\boldsymbol{x}(t)\mid\boldsymbol{x}(1)):\ \mathrm{d}\boldsymbol{x}(t)/\mathrm{d}t \;&=\; \boldsymbol{v}_\theta\big(\boldsymbol{x}(t),t;\gamma\big)\;+\;\sigma(t)^2\nabla\log p_\theta^{t\to1}\big(\boldsymbol{x}(1)\mid\boldsymbol{x}(t)\big)\\ &-\;\frac{1}{2}\sigma^2(t)\nabla\log q_\theta^{0\to t}\big(\boldsymbol{x}(t)\mid\boldsymbol{x}(1)\big)\,,\end{aligned} \tag{12}$$

Under the exact Doob transform, this process is conditioned on the terminal state $\boldsymbol{x}(1)$. In the above, $p_\theta^{t\to1}(\boldsymbol{x}(1)\mid\boldsymbol{x}(t))$ denotes the transition density from state $\boldsymbol{x}(t)$ at time t to terminal state $\boldsymbol{x}(1)$ at time 1 under the SDE in Eq. 3, while $q_\theta^{0\to t}(\boldsymbol{x}(t)\mid\boldsymbol{x}(1))$ denotes the state marginal of the endpoint conditioned SDE in Eq. 11. Eq. 12 therefore defines its endpoint-conditioned velocity field as

$$\begin{aligned}\boldsymbol{u}_\theta(\boldsymbol{x}(t),t;\boldsymbol{x}(1),\gamma) \;&=\; \boldsymbol{v}_\theta\big(\boldsymbol{x}(t),t;\gamma\big)+\sigma(t)^2\nabla\log p_\theta^{t\to1}\big(\boldsymbol{x}(1)\mid\boldsymbol{x}(t)\big)\\ &-\;\frac{1}{2}\sigma^2(t)\nabla\log q_\theta^{0\to t}\big(\boldsymbol{x}(t)\mid\boldsymbol{x}(1)\big)\,,\end{aligned} \tag{13}$$

which can be further simplified using Bayes's rule. The endpoint-conditioned marginal $q_\theta^{0\to t}(\boldsymbol{x}(t)\mid\boldsymbol{x}(1))$ is obtained by conditioning the state marginal $p_\theta^{0\to t}(\boldsymbol{x}(t))$ of the unconditioned SDE in Eq. 3 on the terminal state $\boldsymbol{x}(1)$.

$$q_\theta^{0\to t}(\boldsymbol{x}(t)\mid\boldsymbol{x}(1)) \;=\; p_\theta^{t\to1}(\boldsymbol{x}(1)\mid\boldsymbol{x}(t))p_\theta^{0\to t}(\boldsymbol{x}(t))\;/\;p_\theta^{0\to1}(\boldsymbol{x}(1))\,. \tag{14}$$

Taking the log-gradient with respect to $\boldsymbol{x}(t)$ of Eq. 14 gives:

$$\nabla\log q_\theta^{0\to t}(\boldsymbol{x}(t)\mid\boldsymbol{x}(1)) \;=\; \nabla\log p_\theta^{t\to1}(\boldsymbol{x}(1)\mid\boldsymbol{x}(t)) \;+\; \nabla\log p_\theta^{0\to t}(\boldsymbol{x}(t))\,. \tag{15}$$

Substituting Eq. 15 and Eq. 4 into Eq. 13 yields

$$\boldsymbol{u}_\theta(\boldsymbol{x}(t),t;\boldsymbol{x}(1),\gamma) \;=\; \boldsymbol{u}_\theta(\boldsymbol{x}(t),t;\gamma) \;+\; \frac{1}{2}\sigma(t)^2\nabla\log p_\theta^{t\to1}(\boldsymbol{x}(1)\mid\boldsymbol{x}(t))\,, \tag{16}$$

which can be directly used in the adaptation objective in Eq. 10. Thus, the key quantity governing the endpoint-conditioned velocity in Eq. 16 is the transition density $p_\theta^{t\to1}(\boldsymbol{x}(1)\mid\boldsymbol{x}(t))$ of the SDE in Eq. 3, whose score can be learned via denoising score matching (DSM) (Vincent, 2011), as detailed in Appendix B.2. We further augment DSM with a Kolmogorov backward consistency regularizer to improve transition-score estimation. This component is orthogonal to our main contribution and is deferred to Appendix D together with ablation studies for interested readers.

3.2.2 Sampling Endpoint-Conditioned Terminal Designs

Recall that $\pi_D(\boldsymbol{x}(1)\mid\boldsymbol{x}(0))\propto p_\theta^{0\to1}(\boldsymbol{x}(1)\mid\boldsymbol{x}(0))\mathbb{I}(\boldsymbol{x}(1)\in D)$ by construction. Assuming D contain n designs $\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n\}$, this defines the following categorical distribution over D:

$$\forall\boldsymbol{x}\in D:\;\; \pi_D(\boldsymbol{x}(1)=\boldsymbol{x}\mid\boldsymbol{x}(0)) \;=\; p_\theta^{0\to1}(\boldsymbol{x}\mid\boldsymbol{x}(0))\;/\;\left(\sum_{i=1}^{n}p_\theta^{0\to1}(\boldsymbol{x}_i\mid\boldsymbol{x}(0))\right)\,. \tag{17}$$

To estimate Eq. 17, we parameterize a scalar potential $S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0))$ whose gradient defines the score model $s_\zeta(\boldsymbol{x}, \boldsymbol{x}(0)) \triangleq \nabla_{\boldsymbol{x}} S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0))$ where the parameterization applies to $S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0))$. Using score matching (Vincent, 2011), we train $s_\zeta(\boldsymbol{x}, \boldsymbol{x}(0))$ to approximate $\nabla_{\boldsymbol{x}} \log p_\theta^{0\to 1}(\boldsymbol{x} \mid \boldsymbol{x}(0))$. As such, $S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0)) = \log p_\theta^{0\to 1}(\boldsymbol{x} \mid \boldsymbol{x}(0)) + C(\boldsymbol{x}(0))$ where $C(\boldsymbol{x}(0))$ remains constant across different choices of $\boldsymbol{x} \in D$. This can be shown under exact score matching, where

$$\nabla_{\boldsymbol{x}} \left(S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0)) - \log p_\theta^{0\to 1}(\boldsymbol{x} \mid \boldsymbol{x}(0)) \right) \quad = \quad 0 \,. \tag{18}$$

and therefore $S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0)) - \log p_\theta^{0\to 1}(\boldsymbol{x} \mid \boldsymbol{x}(0)) = C(\boldsymbol{x}(0))$. Consequently, the unknown constant cancels when normalizing over D in Eq. 17, yielding the following estimate from which we can sample terminal designs, with $\pi_\zeta = \pi_D$:

$$\forall \boldsymbol{x} \in D: \;\; \pi_\zeta(\boldsymbol{x}(1) = \boldsymbol{x} \mid \boldsymbol{x}(0)) \quad = \quad \exp\left(S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0))\right) \;\; / \;\; \left(\sum_{i=1}^{n} \exp\left(S_\zeta(\boldsymbol{x}_i, \boldsymbol{x}(0))\right) \right) . \tag{19}$$

This completes the target adaptation objective in Eq. 10, with $\boldsymbol{x}(0) \sim D$ and $\boldsymbol{x}(1) \sim \pi_\zeta(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$. We provide full technical details in Appendix B, together with Algorithm 1 for population-level pre-optimization and Algorithm 2 for target-task adaptation. After jointly adapting the meta-flow and gradient field through integral alignment, we further combine them during design generation via Eq. 31. Additional implementation details of the complete pipeline are provided in the appendix.

4 Experiments and Results

Table 1: Experiment results on Design-Bench Tasks. We report the maximum score (100$^{\text{th}}$ percentile) among $Q = 128$ candidates. The best and second-best results are highlighted in **bold red** and blue, respectively. **Mean Rank** is the average rank across all benchmark tasks.

Method	Benchmarks				Mean Rank
	Ant	D'Kitty	TFBind8	TFBind10	
D_o (best)	0.565	0.884	0.439	0.467	-
BO-qEI	0.732 ± 0.049	0.875 ± 0.033	0.861 ± 0.050	0.648 ± 0.024	11/23
CbAS	0.835 ± 0.050	0.891 ± 0.020	0.929 ± 0.051	0.626 ± 0.017	10.5/23
COMs	0.477 ± 0.068	0.842 ± 0.033	0.750 ± 0.082	0.644 ± 0.046	14.75/23
GA	0.221 ± 0.052	0.622 ± 0.192	0.916 ± 0.058	0.642 ± 0.047	13.5/23
MINs	0.886 ± 0.034	0.926 ± 0.009	0.871 ± 0.033	0.639 ± 0.052	7.5/23
REINFORCE	0.246 ± 0.086	0.358 ± 0.133	0.924 ± 0.056	0.628 ± 0.029	15.25/23
RoMA	0.486 ± 0.077	0.776 ± 0.035	0.808 ± 0.086	0.635 ± 0.046	15.25/23
ICT	0.905 ± 0.040	0.928 ± 0.010	0.723 ± 0.153	0.588 ± 0.023	12.25/23
Tri-mentoring	0.624 ± 0.757	0.716 ± 0.011	0.832 ± 0.147	0.635 ± 0.107	15.25/23
Match-Opt	0.869 ± 0.037	0.934 ± 0.010	0.763 ± 0.109	0.615 ± 0.037	11.25/23
DDOM	0.846 ± 0.093	0.924 ± 0.010	0.747 ± 0.065	0.572 ± 0.029	14.75/23
PGS	0.714 ± 0.046	0.905 ± 0.030	0.526 ± 0.000	0.437 ± 0.250	18/23
GTG	0.727 ± 0.044	0.908 ± 0.016	0.865 ± 0.079	0.633 ± 0.053	12/23
GABO	0.855 ± 0.055	0.894 ± 0.014	0.757 ± 0.084	0.622 ± 0.024	13.75/23
BONET	0.779 ± 0.000	0.911 ± 0.000	0.811 ± 0.117	0.578 ± 0.010	14.5/23
SPADE	0.659 ± 0.076	0.886 ± 0.010	0.497 ± 0.109	0.581 ± 0.047	18.5/23
BDI	0.213 ± 0.084	0.719 ± 0.003	0.878 ± 0.063	0.631 ± 0.031	15.5/23
LTR	0.888 ± 0.063	0.917 ± 0.013	0.873 ± 0.074	0.648 ± 0.054	7/23
DEMO	0.458 ± 0.091	0.791 ± 0.047	0.885 ± 0.059	0.638 ± 0.034	12.75/23
ROOT	0.926 ± 0.029	0.947 ± 0.019	0.939 ± 0.033	0.582 ± 0.000	6.75/23
OptBias	0.906 ± 0.029	0.924 ± 0.019	0.876 ± 0.050	0.635 ± 0.034	7.5/23
ExPT	0.940 ± 0.027	**0.967 ± 0.011**	0.874 ± 0.071	0.615 ± 0.036	7.25/23
DIAL	**0.951 ± 0.013**	0.958 ± 0.005	**0.961 ± 0.031**	**0.690 ± 0.084**	**1.25/23**

4.1 Benchmarks, Baselines, and Evaluation

Benchmark Tasks. We evaluate on eight tasks drawn from two benchmarks, Design-Bench Trabucco et al. (2022) and SOO-Bench Qian et al. (2025). Design-Bench includes two discrete tasks, **TF-Bind-8** and **TF-Bind-10**, which optimize DNA sequences for binding affinity to the SIX6 REF R1 transcription factor, and two robot continuous morphology tasks, **Ant** Brockman et al. (2016) and **D'Kitty** Ahn et al. (2020). SOO-Bench contributes four real-world space-mission tasks: **GTOPX2, GTOPX3, GTOPX4**, and **GTOPX6** European Space Agency & Advanced Concepts Team (2020).

Baselines. We chose 22 methods, including BO-qEI Trabucco et al. (2022), CbAS Brookes et al. (2019), COMs Trabucco et al. (2021), MINs Kumar & Levine (2020), REINFORCE Williams

(1992), standard GRADIENT DESCENT (GA), ROMA Yu et al. (2021), ICT Yuan et al. (2023), TRI-MENTORING Chen et al. (2024), MATCH-OPT Hoang et al. (2024), DDOM Krishnamoorthy et al. (2023), PGS Chemingui et al. (2024), GTG Yun et al. (2024), GABO Yao et al. (2024), BONET Krishnamoorthy et al. (2022), SPADE Yang et al. (2026), BDI Chen et al. (2022), LTR Tan et al. (2025), DEMO Yuan et al. (2025), ROOT Dao et al. (2025), OPTBIAS Fadhel et al. (2026b), and EXPT Nguyen et al. (2023).

Experimental Setup. Following (Nguyen et al., 2023), we retain 1% of the offline data as labeled and treat the remainder as unlabeled. Following (Trabucco et al., 2022), each method generates 128 candidates per task for oracle evaluation. We report the 100th- and 80th-percentile scores over 8 runs as mean $\pm$ standard deviation.

Hyper-parameter Configuration. For each baseline, we use the optimized settings from the original papers. For pre-optimization, we run $E = 100$ epochs with learning rate $\eta = 10^{-3}$ and $\kappa = 8$ generated tasks per epoch to optimize Eq. 7. For target-task adaptation, we run $E_{\mathrm{T}} = 25$ epochs using Eq. 10, with 100 endpoint-conditioned trajectories and 50 quadrature points per trajectory. Full key configurations are provided in Appendix B.

4.2 RESULTS COMPARING DIAL AND BASELINES

Results on Design-Bench Tasks. Table 1 shows that **DIAL** achieves state-of-the-art performance on three out of four tasks and the best mean rank of **1.25**. These results demonstrate the strong and consistent performance of **DIAL** across both continuous and discrete design tasks.

Table 2: Experiment results on SOO-Bench Tasks

Method	Benchmarks				Mean Rank
	GTOPX2	GTOPX3	GTOPX4	GTOPX6	
D_o (best)	-195.44 ± 1.47	-151.85 ± 0.71	-215.74 ± 1.25	-112.11 ± 0.35	-
BO-QEI	-72.87 ± 5.83	-47.48 ± 5.66	-86.20 ± 12.55	-51.03 ± 7.59	5/16
REINFORCE	-144.91 ± 33.69	-118.23 ± 41.91	-214.29 ± 5.17	-110.58 ± 3.24	15.75/16
GA	-77.92 ± 16.84	-58.82 ± 17.44	-133.30 ± 31.38	-57.85 ± 9.30	11/16
COMS	-89.84 ± 19.74	-50.29 ± 8.44	-105.32 ± 17.18	-57.06 ± 19.11	9.75/16
CBAS	-73.08 ± 18.46	-44.81 ± 6.15	-81.43 ± 12.23	-51.10 ± 5.99	4.75/16
MINS	-74.99 ± 13.57	-41.88 ± 7.36	**-69.00 ± 16.67**	-47.49 ± 5.40	2.75/16
GABO	-132.18 ± 18.21	-50.93 ± 9.92	-80.96 ± 10.28	-59.72 ± 5.41	9.75/16
DDOM	-82.48 ± 16.78	-42.43 ± 9.09	-80.15 ± 8.13	-47.14 ± 5.01	4.25/16
LTR	-101.47 ± 19.30	-64.16 ± 10.73	-96.43 ± 13.51	-60.52 ± 8.27	12.5/16
BDI	-94.39 ± 9.40	-58.56 ± 11.34	-207.58 ± 59.91	-86.39 ± 12.35	13.25/16
ROOT	-93.90 ± 14.12	-55.80 ± 10.10	-89.90 ± 18.61	-52.42 ± 5.70	9.5/16
MATCH-OPT	-83.12 ± 12.28	-50.38 ± 13.26	-97.87 ± 9.08	-56.28 ± 6.03	9.25/16
GTG	-177.27 ± 26.05	-97.96 ± 24.19	-125.86 ± 23.82	-99.91 ± 8.57	14.75/16
OPTBIAS	-75.33 ± 7.19	-53.60 ± 11.50	-94.78 ± 14.79	-48.42 ± 4.74	7.25/16
EXPT	-79.32 ± 17.58	-48.56 ± 5.83	-84.55 ± 9.38	-44.46 ± 6.07	5.25/16
DIAL	**-67.63 ± 10.75**	**-37.18 ± 4.52**	-70.79 ± 14.54	**-40.95 ± 4.77**	**1.25/16**

Results on SOO-Bench Tasks. Table 2 summarizes the performance of the baselines against **DIAL** on the more recent SOO-Bench benchmark. Our method achieves the best mean rank of **1.25** and performs best on three out of four tasks. These results demonstrate that **DIAL** generalizes well across a broader range of optimization tasks beyond the Design-Bench suite.

In addition to the 100th-percentile results, we report the 80th-percentile results in Appendix C.3, providing additional evidence of the effectiveness of our few-shot optimization method.

4.3 ABLATION ANALYSIS

Effectiveness of DIAL. We ablate meta-differential alignment (DA) (Eq. 9), target-transfer integral alignment (IA) (Eq. 10), and both jointly. As shown in Table 3, removing either loss degrades performance across all tasks, confirming that both contribute to DIAL.

Table 3: Ablation of DIAL loss components

Method	Ant	DKitty	TFBind8	TFBind10
W/O DA	0.925 ± 0.024	0.941 ± 0.015	0.953 ± 0.025	0.650 ± 0.037
W/O IA	0.936 ± 0.027	0.945 ± 0.014	0.955 ± 0.030	0.673 ± 0.042
W/O BOTH	0.919 ± 0.039	0.937 ± 0.018	0.953 ± 0.025	0.652 ± 0.036
DIAL	**0.951 ± 0.013**	**0.958 ± 0.005**	**0.961 ± 0.031**	**0.690 ± 0.084**

Other experiments. We further evaluate the sensitivity and robustness of our framework through several ablations. Appendix C.5 studies the contributions of meta-training and target-task transfer. Appendix C.4 examines different surrogate losses in place of Eq. 5, while Appendix C.6 ablates our design choices for population task generation. We additionally analyze sensitivity to key hyperparameters in Appendix C.7 and ablate the final design-generation strategy in Appendix C.8.

4.4 Experiments on Endpoint-Constrained Posterior Flow Sampling

We consider a controlled 1-D SDE with nonlinear state-dependent drift $\boldsymbol{v}_\theta(\boldsymbol{x}_t) = -2 + 4 \times \mathrm{sigmoid}(6\boldsymbol{x}_t)$ and noise schedule $\sigma(t) = \sigma_0\sqrt{t(1-t)}$ over $t \in [0, 1]$. We train the transition-score potential S_ζ using score distillation together with the backward-consistency constraint (Appendix B.2). We evaluate both the induced endpoint distribution and the resulting Doob-h-guided posterior flow.

Accuracy of the estimated categorical distribution $\pi_\zeta(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$. While Eq. 19 recovers π_D under exact score matching, the learned S_ζ induces an approximation $\pi_\zeta \simeq \pi_D$ in practice. We set $\sigma_0 = 0.6$ and consider four initial states $\boldsymbol{x}_0$ and a terminal set D of 10 evenly spaced points on $[-5, 5]$. Figure 2 compares **(i)** the transition density $p_\theta^{0\to1}(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$ (*grey*), estimated from 60,000 Monte Carlo trajectories with kernel density estimation; **(ii)** the oracle categorical distribution π_D over D (*black*); and **(iii)** the estimate π_ζ induced by S_ζ via Eq. 19 (*red*). Across all cases, π_ζ closely matches π_D, with small total variation distance (TV) over D.

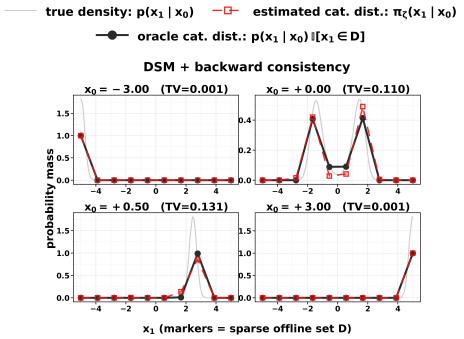


Figure 2: Validation of the estimated categorical distribution π_ζ against the oracle π_D.

Efficiency of Doob-h guidance. We next set $\sigma_0 = 1.2$ and evaluate the estimated Doob-h guidance for 4 endpoint pairs $(\boldsymbol{x}(0), \boldsymbol{x}(1))$. In Figure 3, we show **(i)** raw unconditioned trajectories (*grey*); **(ii)** trajectories rejection-filtered to terminate near $\boldsymbol{x}_1$ (*black*), approximating the posterior flow; and **(iii)** Doob-h-guided rollouts (*red*). The start and target are marked by a *blue* circle and *gold* star, respectively. Using 3,000 trajectories for each flow, we compute the average KL divergence over five timesteps. The guided rollout closely follows the reference posterior flow, with total KL $= 3.306$ across all 4 endpoint pairs. Note that, the Doob-h-guided trajectories were not filtered, as the guidance alone already steers them sufficiently close to the target $\boldsymbol{x}(1)$.

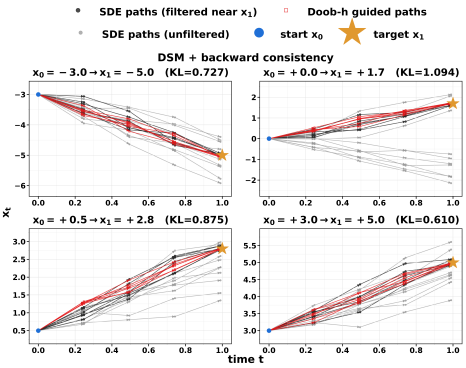


Figure 3: Validation of Doob-h guidance against a reference filtered SDE.

5 Related Work

Prior offline optimization methods broadly fall into *surrogate-based approaches* that optimize learned objective models Trabucco et al. (2021); Hoang et al. (2024); Tan et al. (2025) and *generative approaches* that directly model high-performing designs or transport toward them Kumar & Levine (2020); Krishnamoorthy et al. (2023; 2022); Dao et al. (2025). Most prior work assumes relatively large offline datasets. Few-shot offline optimization remains comparatively underexplored even though it is important in scientific applications. Recent work addresses this regime by leveraging synthetic functions for generative pre-training Nguyen et al. (2023) or surrogate meta-training Fadhel et al. (2026b). In contrast, DIAL jointly couples surrogate gradients and generative transport, exploiting their complementary local and global views to get more value from any given training data. We provide a detailed discussion of related work in Appendix E.

6 Conclusion

We introduced DIAL, a meta-optimization framework for few-shot offline optimization that couples local surrogate gradients with global generative transport through differential-to-integral alignment. DIAL learns transferable gradient-transport consistency across synthetic functions and adapts this coupling to scarce target-task data using endpoint-conditioned paths. Experiments across eight Design-Bench and SOO-Bench tasks demonstrate consistent improvements over strong baselines, with ablations validating the complementary roles of differential and integral alignment. More broadly, differential-to-integral alignment may provide a useful principle for coupling local and global information in other generative optimization and decision-making problems.

AI USE STATEMENT

We used generative AI tools solely to support manuscript polishing, including improvements to organization, presentation, and the clarity of experimental reporting. These tools were not used to define the research problem, develop the core technical contributions, or design the experiments. All AI-assisted content was carefully reviewed and verified by the authors for accuracy, consistency, and fidelity to the intended technical meaning. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. Robel: Robotics benchmarks for learning with low-cost robots. In *Conference on robot learning*, pp. 1300–1313. PMLR, 2020.

Luis A Barrera, Anastasia Vedenko, Jesse V Kurland, Julia M Rogers, Stephen S Gisselbrecht, Elizabeth J Rossin, Jaie Woodard, Luca Mariani, Kian Hong Kock, Sachi Inukai, et al. Survey of variation in human transcription factors reveals prevalent dna binding changes. *Science*, 351 (6280):1450–1454, 2016.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

David Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In *International conference on machine learning*, pp. 773–782. PMLR, 2019.

Yassine Chemingui, Aryan Deshwal, Trong Nghia Hoang, and Janardhan Rao Doppa. Offline model-based optimization via policy-guided gradient search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11230–11239, 2024.

Can Chen, Yingxue Zhang, Jie Fu, Xue Liu, and Mark Coates. Bidirectional learning for offline infinite-width model-based optimization. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_j8yVIyp27Q.

Can Sam Chen, Christopher Beckham, Zixuan Liu, Xue Steve Liu, and Chris Pal. Parallel-mentoring for offline model-based optimization. *Advances in Neural Information Processing Systems*, 36, 2024.

Marc Corstanje, Frank van der Meulen, and Moritz Schauer. Conditioning continuous-time markov processes by guiding. *Stochastics*, 95(6):963–996, 2023.

Manh Cuong Dao, Phi Le Nguyen, Thao Nguyen Truong, and Trong Nghia Hoang. Boosting offline optimizers with surrogate sensitivity. In *Forty-first International Conference on Machine Learning*, 2024a.

Manh Cuong Dao, Phi Le Nguyen, Thao Nguyen Truong, and Trong Nghia Hoang. Incorporating surrogate gradient norm to improve offline optimization techniques. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=ag7piyoyut.

Manh Cuong Dao, The Hung Tran, Phi Le Nguyen, Thao Nguyen Truong, and Trong Nghia Hoang. Root: Rethinking offline optimization as distributional translation via probabilistic bridge. *arXiv preprint arXiv:2509.16300*, 2025.

European Space Agency and Advanced Concepts Team. GTOP database - global optimisation trajectory problems and solutions. https://www.esa.int/gsp/ACT/projects/gtop/, 2020. Archived webpage.

Azza Fadhel, Yassine Chemingui, Minh Hoang, Aryan Deshwal, Trong Nghia Hoang, and Jana Doppa. Nanoporous materials discovery via search bias-guided surrogate modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 38422–38431, 2026a.

Azza Fadhel, The Hung Tran, Trong Nghia Hoang, and Jana Doppa. Black-box optimization from small offline datasets via meta learning with synthetic tasks. *arXiv preprint arXiv:2604.12325*, 2026b.

Clara Fannjiang and Jennifer Listgarten. Autofocused oracles for model-based design. *Advances in Neural Information Processing Systems*, 33:12945–12956, 2020.

Chelsea Finn, P. Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017a. URL https://api.semanticscholar.org/CorpusID:6719686.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proc. ICML*, 2017b.

Justin Fu and Sergey Levine. Offline model-based optimization via normalized maximum likelihood estimation. *arXiv preprint arXiv:2102.07970*, 2021.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Minh Hoang, Azza Fadhel, Aryan Deshwal, Jana Doppa, and Trong Nghia Hoang. Learning surrogates for offline black-box optimization via gradient matching. In *Forty-first International Conference on Machine Learning*, 2024.

Minsu Kim, Jiayao Gu, Ye Yuan, Taeyoung Yun, Zixuan Liu, Yoshua Bengio, and Can Chen. Offline model-based optimization: Comprehensive review. *arXiv preprint arXiv:2503.17286*, 2025.

Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Generative pretraining for black-box optimization. *arXiv preprint arXiv:2206.10786*, 2022.

Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion models for black-box optimization. *arXiv preprint arXiv:2306.07180*, 2023.

Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. *Advances in Neural Information Processing Systems*, 33:5126–5137, 2020.

Chieh-Hsin Lai, Yang Song, Dongjun Kim, Yuki Mitsufuji, and Stefano Ermon. The principles of diffusion models. *arXiv preprint arXiv:2510.21890*, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Tung Nguyen, Sudhanshu Agrawal, and Aditya Grover. Expt: Synthetic pretraining for few-shot experimental design. *Advances in Neural Information Processing Systems*, 36:45856–45869, 2023.

Han Qi, Yi Su, Aviral Kumar, and Sergey Levine. Data-driven offline decision-making via invariant representation learning. *Advances in Neural Information Processing Systems*, 35:13226–13237, 2022.

Hong Qian, Yiyi Zhu, Xiang Shu, Shuo Liu, Yaolin Wen, Xin An, Huakang Lu, Aimin Zhou, Ke Tang, and Yang Yu. SOO-bench: Benchmarks for evaluating the stability of offline black-box optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=bqf0aCF3Dd.

Hannes Risken. Fokker-planck equation. In *The Fokker-Planck equation: methods of solution and applications*, pp. 63–95. Springer, 1989.

Dong-Hee Shin, Young-Han Son, Hyun Jung Lee, Deok-Joong Lee, and Tae-Eui Kam. Offline model-based optimization for real-world molecular discovery. In *Forty-second International Conference on Machine Learning*, 2025.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in artificial intelligence*, pp. 574–584. PMLR, 2020a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020b.

Rong-Xi Tan, Ke Xue, Shen-Huan Lyu, Haopu Shang, yaowang, Yaoyuan Wang, Fu Sheng, and Chao Qian. Offline model-based optimization by learning to rank. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=sb1HgVDLjN.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012.

Brandon Trabucco, Aviral Kumar, Xinyang Geng, and Sergey Levine. Conservative objective models for effective offline model-based optimization. In *International Conference on Machine Learning*, pp. 10358–10368. PMLR, 2021.

Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-bench: Benchmarks for data-driven offline model-based optimization. In *International Conference on Machine Learning*, pp. 21658–21676. PMLR, 2022.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Yonghan Yang, Ye Yuan, Zipeng Sun, Linfeng Du, Bowei He, Haolun Wu, Can Chen, and Xue Liu. Support-proximity augmented diffusion estimation for offline black-box optimization. *arXiv preprint arXiv:2605.11246*, 2026.

Michael S Yao, Yimeng Zeng, Hamsa Bastani, Jacob R. Gardner, James Gee, and Osbert Bastani. Generative adversarial model-based optimization via source critic regularization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=3RxcarQFRn.

Sihyun Yu, Sungsoo Ahn, Le Song, and Jinwoo Shin. Roma: Robust model adaptation for offline model-based optimization. *Advances in Neural Information Processing Systems*, 34:4619–4631, 2021.

Ye Yuan, Can Chen, Zixuan Liu, Willie Neiswanger, and Xue Liu. Importance-aware co-teaching for offline model-based optimization. *arXiv preprint arXiv:2309.11600*, 2023.

Ye Yuan, Youyuan Zhang, Can Chen, Haolun Wu, Melody Zixuan Li, Jianmo Li, James J. Clark, and Xue Liu. Design editing for offline model-based optimization. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=OPFnpl7KiF.

Taeyoung Yun, Sujin Yun, Jaewoo Lee, and Jinkyoo Park. Guided trajectory generation with diffusion models for offline model-based optimization. *arXiv preprint arXiv:2407.01624*, 2024.

A TASK DETAILS

A.1 DESIGN-BENCH TASK

Design-Bench Trabucco et al. (2022) is a widely recognized suite for assessing offline black-box optimization methods. Table 4 summarizes the four tasks we derived from Design-Bench.

Table 4: Overview of tasks in the Design-Bench benchmark. The ground-truth oracle functions for these tasks (unknown to our algorithm) are accessible during evaluation.

Task	Offline Size	Input Dimension	Input Category	Input Type	Oracle
TF Bind 8	32,898	8	4	Discrete	Exact
TF Bind 10	10,000	10	4	Discrete	Exact
Ant Morphology	10,004	60	N/A	Continuous	Exact
D'Kitty Morphology	10,004	56	N/A	Continuous	Exact

TF Bind 8 and TF Bind 10: Optimizing DNA Sequence. The two discrete tasks **TF-Bind-8** and **TF-Bind-10** aim to find the DNA sequence of length 8 and 10, respectively, which maximizes the binding score to a predefined transcription factor (`SIX6_REF_R1`) Barrera et al. (2016). More specifically, each design is a sequence whose element has four possible values that mirror a nucleotide of a DNA sequence. All possible combinations of the sequences (4^8 for **TF-Bind-8** and 4^{10} for **TF-Bind-10**) along with their oracle values are available in the Design-Bench benchmark. Just like the evaluation protocol in Trabucco et al. (2022), only a subset for each task is sampled as offline dataset for that task (i.e., $32,898$ from **TF-Bind-8** and $10,000$ from **TF-Bind-10**).

Ant and D'Kitty Morphology: Robot Morphology Optimization. Both **Ant** and **D'Kitty** focus on optimizing the morphology of simulated robots. More specifically, **Ant Morphology** aims to design the fastest quadrupedal Ant robot from OpenAI Gym Brockman et al. (2016) with 60 continuous parameters, while **D'Kitty Morphology** aims to design a structure most suitable for traversal for ROBEL's D'Kitty robot Ahn et al. (2020) with 56 continuous parameters. The scores for these morphology designs are obtained by simulating a controller trained with Soft Actor-Critic algorithm Haarnoja et al. (2018) in the MuJoCo Todorov et al. (2012) environment. The scoring process consists of 16 independent trials with 100 simulation steps each, and the final score quality is the average of the outputs of these runs.

A.2 SOO-BENCH TASK

SOO-Bench Qian et al. (2025) is a novel benchmark for evaluating offline black-box optimization methods. The suite allows for customization of the distribution of the dataset, suitable for assessing the method's stability and optimality performance. Table 5 summarizes the four tasks we derived from this benchmark.

Table 5: Overview of tasks in the SOO-Bench benchmark. The ground-truth oracle functions for these tasks (unknown to our algorithm) are accessible during evaluation.

Task	Offline Size	Input Dimension	Input Type	Oracle
GTOPX2	11,000	22	Continuous	Exact
GTOPX3	9,000	18	Continuous	Exact
GTOPX4	13,000	26	Continuous	Exact
GTOPX6	11,000	22	Continuous	Exact

GTOPX: Space mission optimization.

These tasks are a set of real-world interplanetary space trajectory design problems. They share a similar goal of minimizing the total velocity variation and differ in settings: **GTOPX2** emulates a rendezvous with Saturn, both **GTOPX3** and **GTOPX4** simulate interplanetary missions to Mercury, with and without resonant of the flybys of the planets respectively, and **GTOPX6** models multi-gravity-assisted missions to Comet 67P/Churyumov-Gerasimenko. The designs of these tasks contain

decision variables crucial to their respective missions, with **GTOPX2** having 22, **GTOPX3** having 18, **GTOPX4** having 26 and **GTOPX6** having 22.

In this work, we follow the few-shot setting in (Nguyen et al., 2023), where only 1% of the data are randomly sampled as labeled observations $D = \{(\boldsymbol{x}_j, z_j)\}_{j=1}^{n}$, while the remaining 99% are treated as unlabeled designs $\Omega = \{\boldsymbol{x}_j\}_{j=1}^{n_u}$ to provide additional support.

B Detailed Algorithmic Description and Implementation of **DIAL**

In this section, we provide further algorithmic and implementation details of **DIAL**.

Algorithm 1 Pre-Optimization via Meta Differential Alignment

Require: Labeled dataset D, unlabeled design pool Ω, epochs $E = 100$, functions per epoch $\kappa = 8$, learning rate $\eta = 10^{-3}$, inner step size $\alpha = 10^{-3}$
Ensure: Meta-gradient parameters ϕ and meta-flow parameters θ

for $e = 1, \dots, E$ **do**
  for $i = 1, \dots, \kappa$ **do**
    Sample RBF kernel parameters (ℓ_i, ρ_i^2)
    Compute K_{DD}^i, $K_{\Omega D}^i$, $K_{D\Omega}^i$, and $K_{\Omega\Omega}^i$ using (ℓ_i, ρ_i^2)
    Sample pseudo-labels $\tilde{\boldsymbol{z}}_i^{\Omega} \sim \mathbb{N}(\boldsymbol{z}_\Omega \mid D)$ via Eq. 20
    Construct $D_i = D \cup \{(\boldsymbol{x}, \tilde{z}_i) : \boldsymbol{x} \in \Omega\}$
    Refit the GP on D_i and obtain its posterior mean g_i
    Construct $\mathcal{P}_i$, D_i^-, and D_i^+ by gradient descent/ascent on g_i
  end for
  $D^- \leftarrow \bigcup_{i=1}^{\kappa} D_i^-$, $D^+ \leftarrow \bigcup_{i=1}^{\kappa} D_i^+$
  Update the meta-gradient field $\nabla g_\phi(\boldsymbol{x})$ according to Eq. 5
  Sample $(\boldsymbol{x}(0), \boldsymbol{x}(1)) \sim D^- \times D^+$ and $t \sim \mathbb{U}[0, 1]$, and set $\gamma = \big(g_i(\boldsymbol{x}(0)), g_i(\boldsymbol{x}(1))\big)$
  Update the meta-flow $\boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma)$ according to Eq. 6
  Jointly update (ϕ, θ) according to meta differential loss Eq. 7 with learning rate η
end for
return (ϕ, θ)

B.1 Pre-Optimization via Meta Differential Alignment

Meta-population task generation. Following common practice in few-shot offline optimization (Fadhel et al., 2026b; Dao et al., 2025), we first construct a population of κ Gaussian process (GP) functions from the labeled dataset $D = \{(\boldsymbol{x}_j, z_j)\}_{j=1}^{n}$. For each function i, we sample the lengthscale ℓ_i and variance ρ_i^2 of an RBF kernel $k_i(\boldsymbol{x}, \boldsymbol{x}') = \rho_i^2 \exp\big(-\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2/(2\ell_i^2)\big)$, and fit the corresponding GP to D. We assume Gaussian observation noise with standard deviation $\sigma_n = 0.01$ to account for noisy observations in the GP regression (Williams & Rasmussen, 2006). For a set of query designs Ω, its posterior is

$$\boldsymbol{z}_\Omega \mid D \sim \mathbb{N}\big(K_{\Omega D}^i (K_{DD}^i + \sigma_n^2 I)^{-1} \boldsymbol{z},\ K_{\Omega\Omega}^i - K_{\Omega D}^i (K_{DD}^i + \sigma_n^2 I)^{-1} K_{D\Omega}^i\big)\,. \tag{20}$$

Unlike prior work, which directly uses the posterior mean of this GP as a generated function, we further densify each function using the unlabeled design pool Ω available in the few-shot setting of (Nguyen et al., 2023). Specifically, we jointly sample pseudo-labels $\tilde{\boldsymbol{z}}_\Omega$ from Eq. 20, construct a densified dataset $D_i = D \cup \{(\boldsymbol{x}, \tilde{\boldsymbol{z}}_{\boldsymbol{i}}) : \boldsymbol{x} \in \Omega\}$, and fit a new GP posterior conditioned on D_i using the same kernel configuration. We then define the generated function g_i as the posterior mean of this densified GP: $g_i(\boldsymbol{x}) = K_{\boldsymbol{x}D_i}^i \left(K_{D_i,D_i}^i + \sigma_n^2 I\right)^{-1} \tilde{\boldsymbol{z}}_i$.

This yields $\{g_i\}_{i=1}^{\kappa}$ with substantially denser support around the available design space; we further evaluate the effectiveness of this construction in Appendix C.6.

Meta-training. For each generated function g_i, we perform gradient descent and ascent from the offline designs to construct $\mathcal{P}_i$, D_i^-, and D_i^+, and use these tasks to meta-train the gradient model $\nabla g_\phi(\boldsymbol{x})$ and the meta ODE flow $\boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma)$ following Section 3.1. In particular, we optimize

$\nabla g_\phi(\boldsymbol{x})$ using Eq. 5, train $\boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma)$ using Eq. 6, and jointly optimize their differential coupling using Eq. 7. For each sampled endpoint pair, the flow condition is $\gamma = (g_i(\boldsymbol{x}(0)), g_i(\boldsymbol{x}(1)))$. We train for $E = 100$ epochs with learning rate 10^{-3} and use the MAML inner-loop step size $\alpha = 10^{-3}$. At each epoch, we sample $\kappa = 8$ GP functions following the procedure above, resulting in a total of 800 generated functions throughout meta-training. Other hyperparameters, including the sampling distributions for the RBF kernel parameters ℓ and ρ and those used for gradient ascent/descent to construct D_i^-, D_i^+, follow the settings in (Dao et al., 2025). Algorithm 1 summarizes this procedure.

Note that, while we learn the generative transport $\mathrm{ODE}_\theta(\boldsymbol{x}(t)) \sim \mathrm{GEN}_\theta(\boldsymbol{x}(t) \mid \gamma)$ through its flow $\boldsymbol{u}_\theta$, target transfer in the next section uses its dual stochastic representation SDE_θ with the same marginals, whose drift $\boldsymbol{v}_\theta$ is recovered via Eq. 4 using the marginal score learned from ODE-induced trajectories before simulating the $\mathrm{SDE}_\theta(\boldsymbol{x}(t))$ trajectories in Appendix B.2.

B.2 Target Transfer via Differential-to-Integral Alignment

Algorithm 2 Target Transfer via Differential-to-Integral Alignment

Require: Target dataset $D = \{(\boldsymbol{x}_i, z_i)\}_{i=1}^n$, pre-optimized parameters (ϕ, θ), adaptation epochs E_T, learning rates η
Ensure: Adapted parameters (ϕ, θ)

for $e = 1, \ldots, E_\mathrm{T}$ **do**

▷ *Transition-score training*
Sample $0 \le s < t \le 1$ and $(\boldsymbol{x}(s), \boldsymbol{x}(t)) \sim \mathrm{SDE}_\theta(\boldsymbol{x}(t))$
Sample $\boldsymbol{\epsilon} \sim \mathbb{N}(\mathbf{0}, \boldsymbol{I})$ and set $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) + \tau\boldsymbol{\epsilon}$
Compute $L_\mathrm{DSM}(\zeta)$ using Eq. 24
Compute $L_\mathrm{BK}(\zeta)$ using Eq. 27
$\zeta \leftarrow \zeta - \eta\nabla_\zeta \left(L_\mathrm{DSM} + \lambda_\mathrm{BK} L_\mathrm{BK}\right)$

▷ *Endpoint sampling*
Sample $(\boldsymbol{x}(0), z_0) \sim D$ and $(\boldsymbol{x}(1), z_1) \sim \pi_D(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$ using Eq. 23 and $\gamma = (z_0, z_1)$

▷ *Endpoint-conditioned ODE velocity*
$\boldsymbol{r}_\zeta(\boldsymbol{x}(t), t) \leftarrow \nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(1), t, 1) - \nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \varnothing, t, \varnothing)$
$\boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \boldsymbol{x}(1), \gamma) \leftarrow \boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma) + \frac{1}{2}\sigma^2(t)\boldsymbol{r}_\zeta(\boldsymbol{x}(t), t)$ as Eq. 16
$\boldsymbol{x}(\cdot) \leftarrow \mathrm{SOLVE}\left(\mathrm{ODE}_\theta(\boldsymbol{x}(t) \mid \boldsymbol{x}(1))\right)$

▷ *Target adaptation*
Compute integral alignment loss $\mathcal{I}(\phi, \theta)$ using Eq. 10
$(\phi, \theta) \leftarrow (\phi, \theta) - \eta\nabla_{(\phi,\theta)}\mathcal{I}(\phi, \theta)$

end for
return adapted parameter (ϕ, θ)

In this section, we provide further implementation details for target-task adaptation. As described in Section 3.2, we fine-tune the meta-gradient model $\nabla g_\phi(\boldsymbol{x})$ jointly with the meta-ODE flow $\boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma)$ using the differential-to-integral alignment loss in Eq. 10. This requires (i) sampling an offline endpoint according to $\pi_D(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$ in Eq. 19, and (ii) constructing the endpoint-conditioned velocity in Eq. 16 toward the sampled endpoint $\boldsymbol{x}(1)$. Both quantities depend on the transition distributions induced by $\mathrm{SDE}_\theta(\boldsymbol{x}(t)) \sim \mathrm{GEN}_\theta(\boldsymbol{x}(t) \mid \gamma)$ in Eq. 3. We first describe the endpoint sampling procedure and then detail how the required transition-score models are learned to construct the corresponding Doob guidance.

B.2.1 Sampling endpoint-conditioned terminal designs

Given a starting design $\boldsymbol{x}(0)$, sampling the terminal design according to Eq. 19 requires evaluating the relative transition likelihood $p_\theta^{0\to1}(\boldsymbol{x} \mid \boldsymbol{x}(0))$ for each candidate $\boldsymbol{x} \in D$. We represent these

transition likelihoods using a scalar score potential $S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(s), t, s)$ parameterized such that

$$\nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(s), t, s) \;\simeq\; \nabla_{\boldsymbol{x}(t)} \log p_\theta^{s\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(s)\big). \tag{21}$$

Note that $\mathrm{SDE}_\theta(\boldsymbol{x}(t)) \sim \mathrm{GEN}_\theta(\boldsymbol{x}(t) \mid \gamma)$ induces different stochastic transition dynamics for different conditions γ; accordingly, in practice we condition the transition-score model on γ, i.e., $S_\zeta(\cdot;\gamma)$, while suppressing this dependence in the subsequent notation for readability.

Then, at $s = 0$ and $t = 1$,

$$S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0), 1, 0) = \log p_\theta^{0\to 1}\big(\boldsymbol{x} \mid \boldsymbol{x}(0)\big) + C(\boldsymbol{x}(0)), \tag{22}$$

where $C(\boldsymbol{x}(0))$ is constant with respect to the candidate endpoint $\boldsymbol{x}$. Therefore, for $D = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, this constant cancels after normalization, and Eq. 19 is implemented as

$$\pi_\zeta\big(\boldsymbol{x}(1) \;=\; \boldsymbol{x} \mid \boldsymbol{x}(0)\big) = \exp\left(S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0), 1, 0)\right) \;/\; \sum_{i=1}^{n} \exp\left(S_\zeta(\boldsymbol{x}_i, \boldsymbol{x}(0), 1, 0)\right). \tag{23}$$

We then sample $\boldsymbol{x}(1) \sim \pi_D(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$ as the terminal design for endpoint-conditioned adaptation. For notational brevity, in the main text we write $S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0)) \equiv S_\zeta(\boldsymbol{x}, \boldsymbol{x}(0), 1, 0)$ when referring to the endpoint transition from $\boldsymbol{x}(0)$ to $\boldsymbol{x}(1) = \boldsymbol{x}$.

Learning transition-score models. To train S_ζ, we sample paired states from trajectories of $\mathrm{SDE}_\theta(\boldsymbol{x}(t))$, perturb the predicted state as $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) + \tau\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathbb{N}(\mathbf{0}, \boldsymbol{I})$, and minimize the denoising score-matching objective (Vincent, 2011)

$$L_{\mathrm{DSM}}(\zeta) = \mathbb{E}_{\boldsymbol{x}(t), \boldsymbol{x}(s)\sim \mathrm{SDE}_\theta(\boldsymbol{x}(t))}\left[\left\| \nabla_{\tilde{\boldsymbol{x}}(t)} S_\zeta\big(\tilde{\boldsymbol{x}}(t), \boldsymbol{x}(s), t, s\big) \;+\; \frac{\tilde{\boldsymbol{x}}(t) - \boldsymbol{x}(t)}{\tau^2} \right\|_2^2\right]. \tag{24}$$

While DSM learns the transition score from sampled trajectories, it does not explicitly enforce that the learned transition potential is consistent with the dynamics of $\mathrm{SDE}_\theta(\boldsymbol{x}(t))$ across different transition intervals. In particular, for a fixed terminal state $(\boldsymbol{x}(t), t)$ and $0 \le s < t \le 1$, the transition density $p_\theta^{s\to t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(s))$ must satisfy the backward Kolmogorov equation with respect to its starting state and time $(\boldsymbol{x}(s), s)$:

$$\partial_s p_\theta^{s\to t} \;+\; \boldsymbol{v}_\theta(\boldsymbol{x}(s), s; \gamma)^\top \nabla_{\boldsymbol{x}(s)} p_\theta^{s\to t} \;+\; \frac{\sigma^2(s)}{2} \Delta_{\boldsymbol{x}(s)} p_\theta^{s\to t} = 0. \tag{25}$$

Since $S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(s), t, s)$ parameterizes the corresponding log-transition density through Eq. 21, we impose the log-form of the backward Kolmogorov equation directly on S_ζ:

$$\partial_s S_\zeta \;+\; \boldsymbol{v}_\theta(\boldsymbol{x}(s), s; \gamma)^\top \nabla_{\boldsymbol{x}(s)} S_\zeta \;+\; \frac{\sigma^2(s)}{2}\left(\Delta_{\boldsymbol{x}(s)} S_\zeta \;+\; \left\|\nabla_{\boldsymbol{x}(s)} S_\zeta\right\|_2^2\right) = 0, \tag{26}$$

where $S_\zeta = S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(s), t, s)$. Accordingly, we define the backward-consistency loss as

$$L_{\mathrm{BK}}(\zeta) \;=\; \mathbb{E}\left[\left| \partial_s S_\zeta \;+\; \boldsymbol{v}_\theta(\boldsymbol{x}(s), s; \gamma)^\top \nabla_{\boldsymbol{x}(s)} S_\zeta \;+\; \frac{\sigma^2(s)}{2}\left(\Delta_{\boldsymbol{x}(s)} S_\zeta + \left\|\nabla_{\boldsymbol{x}(s)} S_\zeta\right\|_2^2\right) \right|^2\right]. \tag{27}$$

The transition-score model is then optimized using $L_{\mathrm{TS}} = L_{\mathrm{DSM}} + \lambda_{\mathrm{BK}} L_{\mathrm{BK}}$. The backward Kolmogorov constraint above applies to the forward transition kernel $p_\theta^{s\to t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(s))$ with $s < t$. Its full derivation and additional ablation studies are provided in Appendix D.

B.2.2 Learning the Doob guidance

Given the sampled endpoint $\boldsymbol{x}(1)$, the endpoint-conditioned velocity in Eq. 16 requires the Doob guidance term $\nabla_{\boldsymbol{x}(t)} \log p_\theta^{t\to 1}(\boldsymbol{x}(1) \mid \boldsymbol{x}(t))$. However, this gradient is taken with respect to the conditioning state $\boldsymbol{x}(t)$ rather than the terminal state $\boldsymbol{x}(1)$. We therefore apply the Bayes relation in Eq. 14 to obtain

$$\nabla_{\boldsymbol{x}(t)} \log p_\theta^{t\to 1}\big(\boldsymbol{x}(1) \mid \boldsymbol{x}(t)\big) \;= \nabla_{\boldsymbol{x}(t)} \log q_\theta^{0\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(1)\big) \;- \nabla_{\boldsymbol{x}(t)} \log p_\theta^{0\to t}\big(\boldsymbol{x}(t)\big). \tag{28}$$

Here, $q_\theta^{0\to t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(1))$ denotes the endpoint-conditioned marginal introduced in Eq. 14, i.e., the distribution of the intermediate state $\boldsymbol{x}(t)$ conditioned on the terminal endpoint $\boldsymbol{x}(1)$.

Table 6: Key hyperparameter settings for **DIAL** .

	Hyperparameter	**Value**
Architecture	Hidden size	128
	Number of layers	4
	Activation	Swish +AdaLN-Zero
Meta-training	Number of epochs (E)	100
	Number of function per epochs κ	8
	Batch size (inner/outer loop)	256
	Learning rate (inner/outer loop) (α, η)	0.001
	Optimizer	Adam
	Conditional dropout for γ	0.15
	Gradient clipping	1.0
	Validation factor	0.01
Target-transfer adaptation	Number of epochs (E_T)	25
	Backward consistency weight (λ_{BK})	1.0
	Number of Euler steps for Doob ODE	50
	Number of Doob ODE trajectories	100
	Guidance scale (λ_s)	0.5
	Sampling steps (T)	50

Recall that S_ζ is parameterized as a conditional score potential over pairs of states and times. Its earlier use in Eq. 21 specializes to the forward transition kernel $p_\theta^{s\to t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(s))$ for $s < t$. The same parameterization can also represent the endpoint-conditioned marginal score by conditioning on the terminal pair $(\boldsymbol{x}(1), 1)$ and training on paired states $(\boldsymbol{x}(t), \boldsymbol{x}(1))$ sampled from the same trajectories. Accordingly, we model

$$\nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(1), t, 1) \;\simeq\; \nabla_{\boldsymbol{x}(t)} \log q_\theta^{0\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(1)\big). \tag{29}$$

For the second term, we apply conditioning dropout by replacing the condition with $\varnothing$, yielding $\nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \varnothing, t, \varnothing) \;\simeq\; \nabla_{\boldsymbol{x}(t)} \log p_\theta^{0\to t}\big(\boldsymbol{x}(t)\big)$, which is jointly trained with the endpoint-conditioned score.

Consequently, the Doob guidance in Eq. 16 is approximated as

$$\nabla_{\boldsymbol{x}(t)} \log p_\theta^{t\to 1}\big(\boldsymbol{x}(1) \mid \boldsymbol{x}(t)\big) \;\simeq \nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(1), t, 1) \;- \nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \varnothing, t, \varnothing). \tag{30}$$

Both score components in Eq. 30 can be learned via DSM analogously to Eq. 24. However, unlike the forward transition kernel in Eq. 21, the first term in Eq. 28 corresponds to the endpoint-conditioned marginal $q_\theta^{0\to t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(1))$, rather than a forward transition kernel $p_\theta^{s\to t}$ with $s < t$. Therefore, the backward-Kolmogorov constraint derived for the latter cannot be applied directly to this score. Instead, we impose consistency directly on the Doob guidance in Eq. 30 through its corresponding forward potential $p_\theta^{t\to 1}(\boldsymbol{x}(1) \mid \boldsymbol{x}(t))$. The full derivation and resulting Doob-guidance consistency loss are provided in Appendix D.2.

Finally, we substitute Eq. 30 into the endpoint-conditioned velocity in Eq. 16. To adapt (ϕ, θ) to the offline target data using the differential-to-integral alignment loss in Eq. 10, we sample $\boldsymbol{x}(0) \sim D$ and $\boldsymbol{x}(1) \sim \pi_\zeta(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$, and then simulate the corresponding Doob-h-conditioned trajectory from $\boldsymbol{x}(0)$ toward the sampled $\boldsymbol{x}(1)$. Since the resulting trajectory depends on θ, it provides the θ-dependent path along which the differential-to-integral alignment loss is evaluated to jointly fine-tune (ϕ, θ) on the target task.

After adapting both the gradient model and the ODE velocity model to the target task, we combine them at inference time to generate the final 128 candidate designs. Specifically, we simulate the adapted flow using the surrogate-guided velocity

$$\boldsymbol{u}_{\mathrm{guide}}(\boldsymbol{x}(t), t; \gamma) \quad \triangleq \quad \boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma) + \lambda_s \nabla_{\boldsymbol{x}(t)} g_\phi(\hat{\boldsymbol{x}}_\theta(1 \mid t)) \;. \tag{31}$$

Here, the first term follows the adapted generative transport, while the second guides the trajectory toward terminal designs with higher predicted objective values. The differentiable one-step estimate $\hat{\boldsymbol{x}}_\theta(1 \mid t)$ of the terminal design is obtained using the average velocity parameterized during meta-flow learning, following Geng et al. (2025). Since $s_\phi(\boldsymbol{x}) = \nabla_{\boldsymbol{x}} g_\phi(\boldsymbol{x})$, the guidance gradient is computed by the chain rule as

$$\nabla_{\boldsymbol{x}(t)} g_\phi(\hat{\boldsymbol{x}}_\theta(1 \mid t)) = \big(\nabla_{\boldsymbol{x}(t)} \hat{\boldsymbol{x}}_\theta(1 \mid t)\big)^\top \, s_\phi(\hat{\boldsymbol{x}}_\theta(1 \mid t)) \,. \tag{32}$$

We simulate Eq. 31 to generate the final 128 candidate designs. The complete target-transfer procedure is summarized in Algorithm 2, with the key hyperparameter settings provided in Table 6.

For reproducibility, our code is available at: `https://anonymous.4open.science/r/DIAL-ICLR`. We provide all code for our main method, together with the 1-D synthetic visualization experiments.

C Additional Experiment Results

C.1 Computation Resource

All our experiments were conducted on a system with the following specifications: Rocky Linux 10.2, a single H100 PCIe 80GB GPU, and CUDA 12.1 . Notably, our method requires less than 5GB of GPU memory per run.

C.2 Empirical Runtime

Empirical Runtime. Table 7 compares the empirical runtime of **DIAL** with representative baselines across multiple tasks. **DIAL** is faster than several widely used methods, including BONET, PGS, and DEMO, while maintaining competitive optimization performance. All experiments are conducted on a single GPU, with approximately 5 GB of memory required per task.

Table 7: Runtime (in seconds) of different models across four tasks.

Model	**Ant**	**D'Kitty**	**TF-Bind-8**	**TF-Bind-10**
GA	122	200	124	420
COMs	154	216	187	329
MINS	103	181	118	165
BONET	16868	21510	4151	4831
DEMO	668	1489	1024	1696
ICT	121	139	81	267
LTR	510	1286	1231	2053
BDI	341	399	497	402
PGS	5494	4868	10014	5869
DIAL	820	904	618	1171

Table 8: Experiment results on Design-Bench Tasks. We report the 80$^{\text{th}}$ percentile score among $Q = 128$ candidates. The best and second-best results are highlighted in **bold red** and blue, respectively. **Mean Rank** is the average rank across all benchmark tasks.

Method	**Benchmarks**				**Mean Rank**
	Ant	**D'Kitty**	**TFBind8**	**TFBind10**	
BO-qEI	0.406 ± 0.066	0.732 ± 0.180	0.520 ± 0.015	**0.521 ± 0.011**	6.25/14
CbAS	0.463 ± 0.041	0.804 ± 0.010	0.600 ± 0.017	0.517 ± 0.014	5.25/14
COMs	0.353 ± 0.015	0.734 ± 0.013	0.431 ± 0.003	0.506 ± 0.008	10.25/14
GA	0.129 ± 0.052	0.434 ± 0.176	0.609 ± 0.028	0.519 ± 0.009	7.75/14
REINFORCE	0.178 ± 0.097	0.338 ± 0.079	0.603 ± 0.025	0.511 ± 0.009	9.25/14
RoMA	0.313 ± 0.018	0.464 ± 0.162	0.437 ± 0.014	0.507 ± 0.007	11.00/14
ICT	0.533 ± 0.036	0.848 ± 0.012	0.500 ± 0.123	0.508 ± 0.041	5.75/14
Tri-mentoring	0.001 ± 0.029	0.617 ± 0.128	0.639 ± 0.102	0.508 ± 0.025	8.50/14
PGS	0.374 ± 0.030	0.783 ± 0.037	0.507 ± 0.036	0.280 ± 0.234	9.50/14
GABO	0.578 ± 0.061	0.833 ± 0.014	0.413 ± 0.011	0.501 ± 0.012	8.00/14
SPADE	0.498 ± 0.100	0.847 ± 0.020	0.492 ± 0.106	0.475 ± 0.044	7.75/14
LTR	0.494 ± 0.042	0.820 ± 0.010	0.565 ± 0.073	0.516 ± 0.028	5.50/14
DEMO	0.291 ± 0.032	0.553 ± 0.181	0.564 ± 0.019	0.517 ± 0.013	8.25/14
DIAL	**0.589 ± 0.033**	**0.853 ± 0.004**	**0.647 ± 0.023**	0.519 ± 0.023	**1.25/14**

C.3 Additional Performance Evaluation of **DIAL**

In addition to the 100th percentile results on the four Design-Bench tasks in the main text, we also report evaluation results at the 80th percentile. As reported in Table 8, **DIAL** achieves the best mean rank of **1.25**, ranking first on Ant, D'Kitty, and TFBind8, and second on TFBind10. These results further demonstrate the effectiveness and robustness of our proposed method.

C.4 Ablation Study on Different Surrogate Loss Choices

We further ablate the loss used to train the gradient model. In our default meta-training setup, we use the **Grad** loss with the MAML-style update in Eq. 5, which directly trains the gradient model to match the GP-generated gradient fields. This is related to the gradient-matching direction explored by Match-Opt (Hoang et al., 2024), but here the gradients are obtained directly from the generated task functions. We further replace this loss with the surrogate objectives from LTR (Tan et al., 2025) and COMs (Trabucco et al., 2021). As shown in Table 9, incorporating **DIAL** consistently improves both surrogate baselines across all four tasks. Among the three variants, the proposed **Grad** objective achieves the strongest overall performance, supporting both the generality of the framework and our choice of gradient-based meta-training.

Table 9: Maximum normalized scores with different surrogate loss choices

Method	**Ant**	**DKitty**	**TFBind8**	**TFBind10**
LTR	0.888 ± 0.063	0.917 ± 0.013	0.873 ± 0.074	0.648 ± 0.054
DIAL (LTR)	**0.944 ± 0.015**	**0.948 ± 0.014**	**0.951 ± 0.025**	**0.657 ± 0.052**
COMs	0.477 ± 0.068	0.842 ± 0.033	0.750 ± 0.082	0.644 ± 0.046
DIAL (COMs)	**0.947 ± 0.023**	**0.949 ± 0.013**	**0.952 ± 0.022**	**0.659 ± 0.070**
Match-Opt	0.869 ± 0.037	0.934 ± 0.010	0.763 ± 0.109	0.615 ± 0.037
DIAL (Grad)	**0.951 ± 0.013**	**0.958 ± 0.005**	**0.961 ± 0.031**	**0.690 ± 0.084**

C.5 Ablation on Meta-Training and Target Transfer

We further ablate the contributions of population-level meta-training and target-task transfer in DIAL. As shown in Table 10, removing either component consistently degrades performance across the benchmark tasks, further highlighting the importance of both stages in the full framework.

Table 10: Maximum normalized scores for ablations of the meta-training and target-transfer stages.

Method	**Ant**	**DKitty**	**TFBind8**	**TFBind10**
w/o meta-training	0.931 ± 0.020	0.950 ± 0.013	0.933 ± 0.039	0.688 ± 0.088
w/o target-transfer	0.936 ± 0.027	0.945 ± 0.014	0.955 ± 0.030	0.673 ± 0.042
DIAL	**0.951 ± 0.013**	**0.958 ± 0.005**	**0.961 ± 0.031**	**0.690 ± 0.084**

C.6 Evaluation of Gaussian-Process Task Generation

We further evaluate the population task-generation strategy introduced in Appendix B.1. Specifically, we compare the GP construction used in prior work with our proposed densified-GP variant. As shown in Table 11, **DIAL** consistently improves over the corresponding ROOT (Dao et al., 2025) baseline under both task-generation settings, demonstrating the benefit of coupling the generative model with the gradient model. Moreover, the densified-GP variant yields stronger performance for **DIAL** on all four tasks, supporting the effectiveness of our proposed population task-generation procedure.

Table 11: Maximum normalized scores with densified versus no densified GPs

Method	Ant	DKitty	TFBind8	TFBind10
ROOT(w/o densified GP)	0.926 ± 0.029	0.947 ± 0.019	0.939 ± 0.033	0.582 ± 0.000
DIAL (w/o densified GP)	**0.941 ± 0.038**	**0.950 ± 0.011**	**0.942 ± 0.038**	**0.663 ± 0.130**
ROOT (densified-GP)	0.946 ± 0.014	0.952 ± 0.017	0.937 ± 0.031	0.636 ± 0.035
DIAL (densified-GP)	**0.951 ± 0.013**	**0.958 ± 0.005**	**0.961 ± 0.031**	**0.690 ± 0.084**

C.7 Hyperparameter Tuning

Backward-consistency weight λ_{BK} and score-matching choice. We tune the weight λ_{BK} used to train the transition-score model by loading a pre-trained meta-ODE flow from the **Ant** task, randomly sampling 100 endpoint pairs $(\boldsymbol{x}(0), \boldsymbol{x}(1))$, and simulating the corresponding Doob-h guided trajectories from $\boldsymbol{x}(0)$ toward $\boldsymbol{x}(1)$. We then measure the mean-squared error between the generated terminal state and the prescribed endpoint $\boldsymbol{x}(1)$. As reported in Table 12, $\lambda_{\mathrm{BK}} = 1.0$ achieves the lowest MSE of 0.0361 when combined with denoising score matching (DSM) in Eq. 24. We also compare DSM (Vincent, 2011) with sliced score matching (SSM) (Song et al., 2020a), and observe that DSM provides better endpoint-reconstruction accuracy in this setting.

Table 12: Terminal MSE between the generated and target $\boldsymbol{x}(1)$ under the endpoint-conditioned SDE, using a pre-trained meta flow on the Ant task.

Score matching loss	$\lambda_{\mathrm{BK}} = 0.0$	0.05	0.5	1.0
DSM	0.2389	0.1297	0.0621	**0.0361**
SSM	0.3120	0.1090	0.0670	0.0684

Table 13: Ablation study on the number of Euler steps T for solving ODE/SDE

T	Ant	TFBind8
25	0.944 ± 0.023	0.960 ± 0.033
50 (ours)	**0.951 ± 0.013**	**0.961 ± 0.031**
100	0.951 ± 0.015	0.960 ± 0.033

Table 14: Ablation study on the guidance scale λ_s in design generation

λ_s	Ant	TFBind8
0.05	0.948 ± 0.015	0.960 ± 0.031
0.1	0.949 ± 0.021	0.957 ± 0.022
0.5 (ours)	**0.951 ± 0.013**	**0.961 ± 0.031**

Number of Euler steps T. Since our method requires solving the ODE/SDE induced by the meta-flow generative transport, we use the standard Euler discretization. We evaluate different numbers of sampling steps and find that $T = 50$ achieves the best performance, while the method remains stable across a range of step counts, as shown in Table 13.

Guidance scale λ_s. During design sampling, λ_s controls the strength of the surrogate guidance in Eq. 31. As shown in Table 14, $\lambda_s = 0.5$ achieves the best performance.

C.8 Ablation on the Design Generation Process

Beyond the design-generation procedure in Eq. 31, we further ablate how the adapted generative transport and surrogate gradient model are combined at inference time. We compare three alternatives: (i) ODE, which samples designs using only the adapted meta-flow ODE, where the ODE has already been fine-tuned using our differential-to-integral alignment framework; (ii) Average velocity, which directly combines the two fields as $\boldsymbol{u}_\theta(\boldsymbol{x}(t), t; \gamma) + \nabla_{\boldsymbol{x}(t)} g_\phi(\boldsymbol{x}(t))$; and (iii) Filter, which first generates 256 designs using the adapted meta-flow ODE and retains the 128 designs with the highest surrogate predictions. As shown in Table 15, the two naive combinations, Average velocity and Filter, substantially degrade performance. The adapted meta-flow ODE alone remains competitive, while the proposed DIAL generation procedure consistently achieves the best performance on both tasks, with a small but consistent improvement over ODE.

Table 15: Ablation study on the design generation process

Method	Ant	TFBind8
ODE	0.946 ± 0.013	0.956 ± 0.034
AVERAGE VELOCITY	0.934 ± 0.026	0.912 ± 0.029
FILTER	0.889 ± 0.036	0.933 ± 0.031
DIAL	**0.951 ± 0.013**	**0.961 ± 0.031**

D DERIVATION OF THE BACKWARD CONSISTENCY CONSTRAINT

In this section, we derive the backward Kolmogorov consistency constraints used throughout the transition-score learning procedure. We first derive the constraint for the forward transition model $p_\theta^{s\to t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(s))$, which yields the regularizer in Eq. 27. We then derive the corresponding consistency condition for the Doob guidance score, obtained from the forward potential $p_\theta^{t\to 1}(\boldsymbol{x}(1) \mid \boldsymbol{x}(t))$ and its Bayes decomposition in Eq. 28.

D.1 BACKWARD CONSISTENCY OF THE TRANSITION-SCORE MODEL

Recall that the transition density $p_\theta^{s\to t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(s))$ is induced by $\mathrm{SDE}_\theta(\boldsymbol{x}(t))$ in Eq. 3.

For a fixed terminal state $(\boldsymbol{x}(t), t)$, define $h(\boldsymbol{x}(s), s) \triangleq p_\theta^{s\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(s)\big)$ $s < t$.

For an infinitesimal step $\delta > 0$, the Markov property of SDE_θ gives

$$h(\boldsymbol{x}(s), s) \;=\; \mathbb{E}\left[h(\boldsymbol{x}(s+\delta), s+\delta) \mid \boldsymbol{x}(s)\right]. \tag{33}$$

Therefore, $\mathbb{E}\left[h(\boldsymbol{x}(s+\delta), s+\delta) - h(\boldsymbol{x}(s), s) \mid \boldsymbol{x}(s)\right] = 0$

Applying Itô's formula to $h(\boldsymbol{x}(s), s)$ gives

$$\mathrm{d}h \;=\; \partial_s h\,\mathrm{d}s \;+\; \nabla_{\boldsymbol{x}(s)} h^\top \mathrm{d}\boldsymbol{x}(s) \;+\; \frac{1}{2}\mathrm{d}\boldsymbol{x}(s)^\top \nabla^2_{\boldsymbol{x}(s)} h\,\mathrm{d}\boldsymbol{x}(s). \tag{34}$$

Substituting Eq. 3 and using $\mathrm{d}\boldsymbol{w}(s)\mathrm{d}\boldsymbol{w}(s)^\top = \boldsymbol{I}\mathrm{d}s$ yields

$$\mathrm{d}h \;=\; \Big[\partial_s h \;+\; \boldsymbol{v}_\theta(\boldsymbol{x}(s), s; \gamma)^\top \nabla_{\boldsymbol{x}(s)} h \;+\; \frac{\sigma^2(s)}{2}\Delta_{\boldsymbol{x}(s)} h\Big]\mathrm{d}s \;+\; \sigma(s)\nabla_{\boldsymbol{x}(s)} h^\top \mathrm{d}\boldsymbol{w}(s). \tag{35}$$

Taking the conditional expectation eliminates the stochastic term. Combining the result with the Markov property gives

$$\partial_s h + \boldsymbol{v}_\theta(\boldsymbol{x}(s), s; \gamma)^\top \nabla_{\boldsymbol{x}(s)} h + \frac{\sigma^2(s)}{2}\Delta_{\boldsymbol{x}(s)} h = 0, \tag{36}$$

which is the backward Kolmogorov equation satisfied by $p_\theta^{s\to t}(\boldsymbol{x}(t) \mid \boldsymbol{x}(s))$.

Since $h(\boldsymbol{x}(s), s) = p_\theta^{s\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(s)\big)$, we have $\nabla_{\boldsymbol{x}(s)} h = h\nabla_{\boldsymbol{x}(s)} \log p_\theta^{s\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(s)\big)$, and

$$\Delta_{\boldsymbol{x}(s)} h \;=\; h\left(\Delta_{\boldsymbol{x}(s)} \log p_\theta^{s\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(s)\big) \;+\; \left\|\nabla_{\boldsymbol{x}(s)} \log p_\theta^{s\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(s)\big)\right\|_2^2\right). \tag{37}$$

Substituting these identities into Eq. 36 and dividing by $h > 0$ gives

$$\begin{aligned}\partial_s \log p_\theta^{s\to t} \;&+\; \boldsymbol{v}_\theta(\boldsymbol{x}(s), s; \gamma)^\top \nabla_{\boldsymbol{x}(s)} \log p_\theta^{s\to t} \\ &+\; \frac{\sigma^2(s)}{2}\left(\Delta_{\boldsymbol{x}(s)} \log p_\theta^{s\to t} + \left\|\nabla_{\boldsymbol{x}(s)} \log p_\theta^{s\to t}\right\|_2^2\right) = 0\end{aligned} \tag{38}$$

We therefore impose the same constraint on S_ζ, yielding

$$\partial_s S_\zeta + \boldsymbol{v}_\theta(\boldsymbol{x}(s), s; \gamma)^\top \nabla_{\boldsymbol{x}(s)} S_\zeta + \frac{\sigma^2(s)}{2}\left(\Delta_{\boldsymbol{x}(s)} S_\zeta + \left\|\nabla_{\boldsymbol{x}(s)} S_\zeta\right\|_2^2\right) \simeq 0, \tag{39}$$

where $S_\zeta = S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(s), t, s)$. Squaring the residual of Eq. 39 and averaging over sampled transitions gives the backward-consistency loss L_{BK} in Eq. 27.

D.2 Backward Consistency of the Doob Guidance

For a fixed endpoint $\boldsymbol{x}(1)$, define $h(\boldsymbol{x}(t), t) = p_\theta^{t\to 1}\big(\boldsymbol{x}(1) \mid \boldsymbol{x}(t)\big), t < 1$.

Since h is the forward transition probability of $\mathrm{SDE}_\theta(\boldsymbol{x}(t))$ from t to 1, it satisfies

$$\partial_t h \;+\; \boldsymbol{v}_\theta(\boldsymbol{x}(t), t; \gamma)^\top \nabla_{\boldsymbol{x}(t)} h \;+\; \frac{\sigma^2(t)}{2} \Delta_{\boldsymbol{x}(t)} h \;=\; 0. \tag{40}$$

Define the Doob guidance score $\boldsymbol{r}(\boldsymbol{x}(t), t) = \nabla_{\boldsymbol{x}(t)} \log h(\boldsymbol{x}(t), t)$. Dividing Eq. 40 by h gives

$$\partial_t \log h \;+\; \boldsymbol{v}_\theta(\boldsymbol{x}(t), t; \gamma)^\top \boldsymbol{r} \;+\; \frac{\sigma^2(t)}{2} \left(\nabla_{\boldsymbol{x}(t)} \cdot \boldsymbol{r} + \|\boldsymbol{r}\|_2^2\right) \;=\; 0. \tag{41}$$

Taking $\nabla_{\boldsymbol{x}(t)}$ on both sides yields

$$\partial_t \boldsymbol{r} \;+\; \nabla_{\boldsymbol{x}(t)} \left[\boldsymbol{v}_\theta(\boldsymbol{x}(t), t; \gamma)^\top \boldsymbol{r}\right] \;+\; \frac{\sigma^2(t)}{2} \nabla_{\boldsymbol{x}(t)} \left[\nabla_{\boldsymbol{x}(t)} \cdot \boldsymbol{r} \;+\; \|\boldsymbol{r}\|_2^2\right] \;=\; 0. \tag{42}$$

From Bayes' rule in Eq. 28,

$$\boldsymbol{r} \;=\; \nabla_{\boldsymbol{x}(t)} \log q_\theta^{0\to t}\big(\boldsymbol{x}(t) \mid \boldsymbol{x}(1)\big) \;-\; \nabla_{\boldsymbol{x}(t)} \log p_\theta^{0\to t}\big(\boldsymbol{x}(t)\big). \tag{43}$$

Accordingly, using the shared score potential S_ζ, we define

$$\boldsymbol{r}_\zeta \;=\; \nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(1), t, 1) \;-\; \nabla_{\boldsymbol{x}(t)} S_\zeta(\boldsymbol{x}(t), \varnothing, t, \varnothing). \tag{44}$$

Substituting $\boldsymbol{r}_\zeta$ into Eq. 42 yields the Doob backward-consistency residual

$$\boldsymbol{R}_{\mathrm{BK}}^{\mathrm{Doob}} = \partial_t \boldsymbol{r}_\zeta \;+\; \nabla_{\boldsymbol{x}(t)} \left[\boldsymbol{v}_\theta(\boldsymbol{x}(t), t; \gamma)^\top \boldsymbol{r}_\zeta\right] \;+\; \frac{\sigma^2(t)}{2} \nabla_{\boldsymbol{x}(t)} \left[\nabla_{\boldsymbol{x}(t)} \cdot \boldsymbol{r}_\zeta \;+\; \|\boldsymbol{r}_\zeta\|_2^2\right]. \tag{45}$$

We regularize the learned Doob guidance using $L_{\mathrm{BK}}^{\mathrm{Doob}}(\zeta) = \mathbb{E}\left[\left\|\boldsymbol{R}_{\mathrm{BK}}^{\mathrm{Doob}}\right\|_2^2\right]$

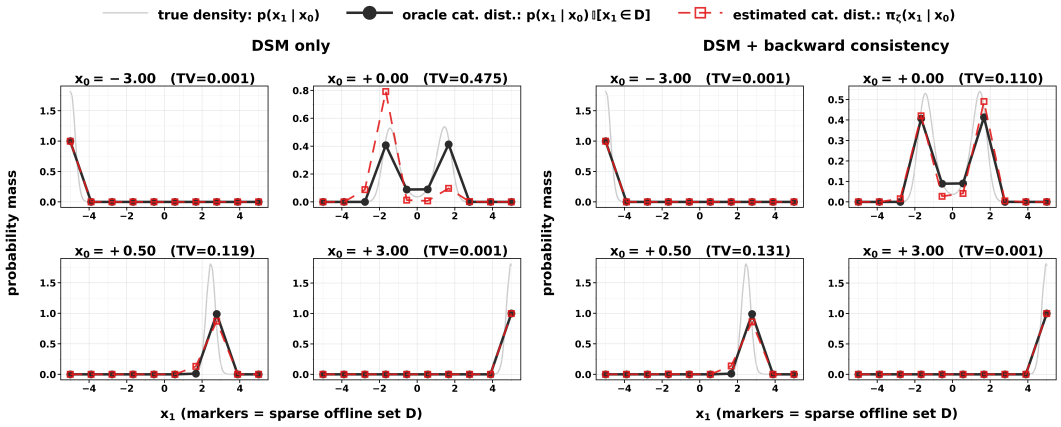


Figure 4: Synthetic 1-D visualization of the estimated endpoint distribution $\pi_D(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$ against the oracle distribution restricted to D: DSM-only training (*left*) versus DSM with backward consistency (*right*). See Sec. D.3 for the experimental protocol.

D.3 Experimental Validation of Backward Consistency

Accuracy of the estimated endpoint distribution $\pi_D(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$.

We instantiate the transition-score model $S_\zeta(\boldsymbol{x}(t), \boldsymbol{x}(s), t, s)$ in Eq. 21 on a controlled 1-D SDE with nonlinear, state-dependent drift $\boldsymbol{v}(\boldsymbol{x}(t)) = -2 + 4 \times \mathrm{sigmoid}(k\boldsymbol{x}(t))$ and noise schedule $\sigma(t) = \sigma_0\sqrt{t(1-t)}$, with $k = 6$ and $\sigma_0 = 0.6$ over $t \in [0, 1]$. For each of four starting points $\boldsymbol{x}(0) \in \{-3, 0, 0.5, 3\}$, Figure 4 plots three curves over a fixed sparse set of terminal designs

$D = \{\boldsymbol{x}^{(j)}\}_{j=1}^{10}$, consisting of 10 evenly spaced points on $[-5, 5]$: **(i)** the true transition density $p_\theta^{0\to1}(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$ (*grey*), estimated by Monte Carlo simulation with kernel density estimation using 60,000 trajectories for each $\boldsymbol{x}(0)$; **(ii)** the corresponding oracle transition mass restricted and normalized over D (*black*); and **(iii)** the estimated categorical distribution $\pi_\zeta(\boldsymbol{x}(1) \mid \boldsymbol{x}(0))$ (*red*), computed from S_ζ using Eq. 23. Panel (a) trains S_ζ using denoising score matching alone via Eq. 24, whereas panel (b) additionally imposes the backward-consistency loss in Eq. 27.

As Figure 4 shows, the estimated π_ζ closely tracks the oracle restricted distribution in both panels, with a clear improvement after backward consistency is introduced. At the bimodal starting point $\boldsymbol{x}(0) = 0$, DSM alone misallocates substantial probability mass toward one mode ($\mathrm{TV}^1 = 0.475$), whereas adding backward consistency reduces the error to $\mathrm{TV} = 0.110$ and recovers the two-mode allocation more accurately. This demonstrates that backward consistency improves the fidelity of the estimated endpoint distribution, particularly in multimodal regimes.

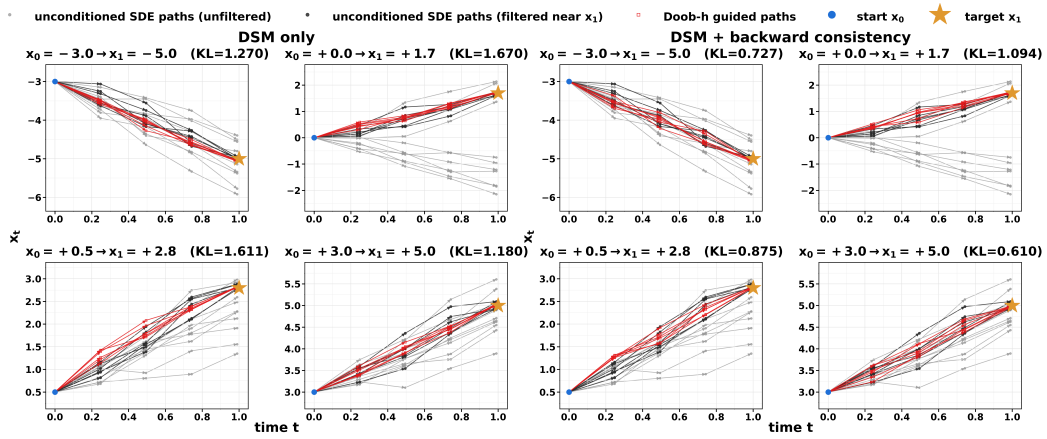


Figure 5: Synthetic 1-D trajectories generated by the learned Doob-guided dynamics against the reference rejection-filtered posterior flow for four endpoint pairs $(\boldsymbol{x}(0), \boldsymbol{x}(1))$. For readability, each panel shows 6 randomly selected trajectories at 5 evenly spaced time points.

Accuracy of the learned Doob guidance. Using the same SDE with $\sigma_0 = 1.2$, we further evaluate the endpoint-conditioned dynamics obtained from the Doob guidance in Eq. 30. We compare trajectories generated by the endpoint-conditioned SDE in Eq. 11 against a reference *filtered SDE*, obtained by simulating unconditioned trajectories from $\boldsymbol{x}(0)$ and retaining only those terminating near $\boldsymbol{x}(1)$. This rejection-filtered ensemble provides a Monte Carlo approximation to the true endpoint-conditioned posterior flow. We evaluate four endpoint pairs $(\boldsymbol{x}(0), \boldsymbol{x}(1))$ in Figure 5. The Doob-guided trajectories are not filtered, since the learned guidance directly steers each rollout toward the prescribed endpoint $\boldsymbol{x}(1)$. For each pair, we simulate 3,000 trajectories from both the filtered SDE and the Doob-guided SDE and compute the average KL divergence between their intermediate-time distributions. Adding backward consistency consistently reduces the discrepancy between the learned and reference posterior flows, with the largest improvement for $\boldsymbol{x}(0) = 3 \to \boldsymbol{x}(1) = 5$, where the KL divergence decreases from 1.180 to 0.610.

E Detailed Related Work

Prior work on offline optimization can be broadly viewed from two complementary perspectives: *surrogate-based methods* and *generative-based methods*. Most of these methods assume relatively large offline datasets and can degrade substantially in the small-data regime.

Surrogate-Based methods. Surrogate-based methods learn forward objective models that provide local improvement signals for optimizing candidate designs. Most methods optimize designs by following the learned surrogate landscape while regularizing predictions in out-of-distribution (OOD) regions Trabucco et al. (2021); Chen et al. (2024); Yuan et al. (2023); Yu et al. (2021); Dao et al.

[1] TV denotes the total variation distance between the oracle and estimated categorical distributions over D.

(2024a;b); Hoang et al. (2024); Qi et al. (2022); Fu & Levine (2021); Fannjiang & Listgarten (2020). Representative examples include (Trabucco et al., 2021; Dao et al., 2024a; Yuan et al., 2023; Chen et al., 2024; Hoang et al., 2024; Tan et al., 2025).

Generative-based Methods. Generative approaches instead learn distributions or transport dynamics that directly produce high-performing designs. Some methods learn conditional mappings from target objective values to candidate designs Kumar & Levine (2020); Nguyen et al. (2023); Krishnamoorthy et al. (2023), while others explicitly model transport from low- to high-performing design regions Krishnamoorthy et al. (2022); Dao et al. (2025). For example, Kumar & Levine (2020) performs model inversion, Krishnamoorthy et al. (2023) learns value-conditioned diffusion dynamics, and Dao et al. (2025) formulates probabilistic transport between low- and high-value design distributions.

Few-Shot Offline Optimization Methods. There is relatively limited work on offline optimization in the small-data regime. Prior work has explored different ways to leverage synthetic data from objective functions sampled from Gaussian processes to address the few-shot optimization challenges. ExPT Nguyen et al. (2023) from the generative family performs pre-training and OptBias Fadhel et al. (2026b) from the surrogate family performs meta-training on this synthetic data.

Our DIAL framework bridges the complementary strengths of surrogate and generative methods to get more value from any given training data. Instead of adapting the surrogate and generative transport independently, we couple them through *differential alignment* during population-level meta-training, aligning the surrogate gradient field with the local dynamics of the generative transport across related functions. During target-task adaptation, we further introduce *differential-to-integral alignment*, which transfers this local consistency to finite transport trajectories using the sparse target offline data. This provides a unified few-shot optimization framework in which local surrogate geometry and global generative transport are jointly adapted and subsequently combined for design generation.