End-to-End Differentiable Diffusion Conditioning for Physics-Informed Optimization

Ricardo Luna Gutierrez[1] , **Vineet Gundecha**[1] , **Rahman Ejaz**[2] , **Varchas Gopalaswamy**[2] , **Riccardo Betti**[2] , **Sahand Ghorbanpour**[1] , **Aarne Lees**[2] , **Soumyendu Sarkar**[1] *
[1]Hewlett Packard Enterprise, [2]University of Rochester
{rluna, vineet.gundecha, sahand.ghorbanpour, soumyendu.sarkar}@hpe.com
{reja, vgop, betti, alees}@lle.rochester.edu

Abstract

Optimizing high-dimensional physical systems under expensive evaluations and complex constraints is a central challenge in science and engineering. In offline model-based optimization (MBO), methods must improve designs using only a fixed dataset and no test-time access to the true objective. However, existing methods typically suffer from conservatism, unconstrained surrogate exploitation, or function as static inference-time samplers that cannot refine candidates against precise physical objectives. In this paper, we propose Differentiable Diffusion Conditioning (D2C) for offline MBO, which leverages the fact that deterministic sampling makes the reverse denoising chain differentiable with respect to the conditioning signal. D2C turns a frozen conditional diffusion model into an end-to-end gradient-based optimizer by directly optimizing conditioning variables at test time via gradients from frozen surrogate objectives and physics-informed penalties backpropagated through the reverse chain. We implement this idea in a composed proposer-evaluator framework and evaluate it on laser pulse optimization for inertial confinement fusion, sustainable data-center workload scheduling, and four Design-Bench tasks. Our method achieves the strongest results on the two real-world tasks and the best average rank across four Design-Bench tasks.

1 Introduction

The optimization of complex physical systems remains one of the most consequential challenges in science and engineering. Problems such as laser pulse shaping for inertial confinement fusion (ICF) and workload scheduling in large-scale data centers share a common structure: objective functions are computationally expensive to evaluate, the design space is high-dimensional with non-linear correlations, and candidate solutions must satisfy strict physical constraints where any violation carries safety-critical consequences.

The offline model-based optimization (MBO) paradigm addresses the evaluation-cost bottleneck by learning entirely from a fixed dataset of previously evaluated designs, with no access to the true objective at test time. Within this paradigm, existing approaches fall into three broad families. Forward (surrogate-based) methods train a neural network to approximate the objective and perform gradient ascent in design space [30, 9]; while effective, optimizing directly in unconstrained design space can move into regions poorly supported by the offline data, where the surrogate is unreliable. Inverse methods learn a conditional mapping from target performance labels to designs [2, 19, 18, 22]; these tend to stay closer to the empirical data distribution but are limited by the performance ceiling of the training labels and struggle to extrapolate beyond observed optima. Trajectory-based methods construct synthetic improvement paths and train generative models to imitate optimization

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

dynamics [3, 39]; while powerful, they remain inference-time samplers whose conditioning targets are fixed prior to generation, with no mechanism to refine designs against downstream objectives.

A common limitation across these families is the lack of a direct, differentiable path from a physics-informed objective back to the generative process. The conditioning signal that steers generation is treated as a static input rather than an optimizable variable. This decoupling is especially limiting in safety-critical physical domains where the objective is not a single scalar score but a composite of performance metrics, conservation laws, and hard feasibility constraints that must be jointly satisfied.

We address these limitations by proposing Differentiable Diffusion Conditioning (D2C) for Physics-Informed Optimization, a novel approach that converts a conditional diffusion model into an end-to-end gradient-based optimizer for constrained physical design. Our key insight for offline MBO is that deterministic diffusion sampling renders the reverse chain differentiable with respect to the conditioning input, enabling end-to-end backpropagation from downstream physics-informed objectives, through a frozen surrogate evaluator and physics-informed penalty terms, back to the conditioning variables. D2C therefore optimizes conditioning directly at test time while keeping both the diffusion proposer and surrogate evaluator fixed, yielding a practical end-to-end gradient path from the downstream objective to the conditioning variables. The diffusion model tends to keep candidates closer to the data distribution, while surrogate and physics-informed losses steer them toward feasible, high-performing designs.

We embed this mechanism within a composed proposer-evaluator architecture: the diffusion model G_θ (proposer) generates candidates, and a surrogate S_ϕ (evaluator) scores them, with both remaining fixed during test-time optimization. Only the conditioning input is updated, which helps keep optimization closer to the proposer's learned data distribution while limiting direct surrogate exploitation. A stochastic multi-objective weighting scheme and validity-first selection further encourage broad exploration of competing objective trade-offs while enforcing strict constraint satisfaction. Figure 1 provides an overview of the full D2C pipeline. Our main contributions are:

- **End-to-End Conditioning Optimization Through Diffusion.** We use the deterministic reverse diffusion chain to backpropagate end-to-end from composite physics objectives to the conditioning inputs, making the conditioning signal itself the optimization variable within the generative process; optimizing it outperforms optimizing the latent noise (Appendix D.7).
- **Frozen Proposer-Evaluator Architecture.** We introduce a composed proposer-evaluator architecture in which a frozen diffusion proposer helps keep candidates closer to the empirical data distribution, while a frozen surrogate evaluator provides optimization gradients, reducing out-of-distribution exploitation.
- **Physics-Informed Multi-Objective Optimization.** We integrate heterogeneous objectives directly into the differentiable sampling chain via a stochastic Dirichlet-weighted loss, enabling principled multi-objective optimization under domain-specific constraints.
- **Strong Empirical Performance.** D2C achieves the best results on the two real-world physical design tasks and the best average rank across four Design-Bench tasks.

2 Preliminaries

2.1 Conditional Denoising Diffusion Models

Denoising Diffusion Probabilistic Models (DDPMs) [15] learn a generative model of a data distribution $p(x)$ by reversing a forward noise process. In the conditional setting, the model learns to sample $x \sim p(x|c)$ given a conditioning signal $c \in \mathbb{R}^{d_c}$. For a continuous design space $x \in \mathbb{R}^d$, the forward process adds Gaussian noise according to a variance schedule $\beta_1, \ldots, \beta_T$:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}) \tag{1}$$

A noise-prediction network $\epsilon_\theta(x_t, c, t)$ is trained by minimizing the denoising objective:

$$\mathcal{L}_{diff} = \mathbb{E}_{x\sim p(x|c),\, \epsilon\sim\mathcal{N}(0,\mathbf{I}),\, t\sim\mathrm{Unif}(\{1,\ldots,T\})}\left[\|\epsilon - \epsilon_\theta(x_t, c, t)\|^2\right] \tag{2}$$

where $x_t = \sqrt{\bar{\alpha}_t}x + \sqrt{1-\bar{\alpha}_t}\epsilon$ and $\bar{\alpha}_t = \prod_{i=1}^{t}(1-\beta_i)$.

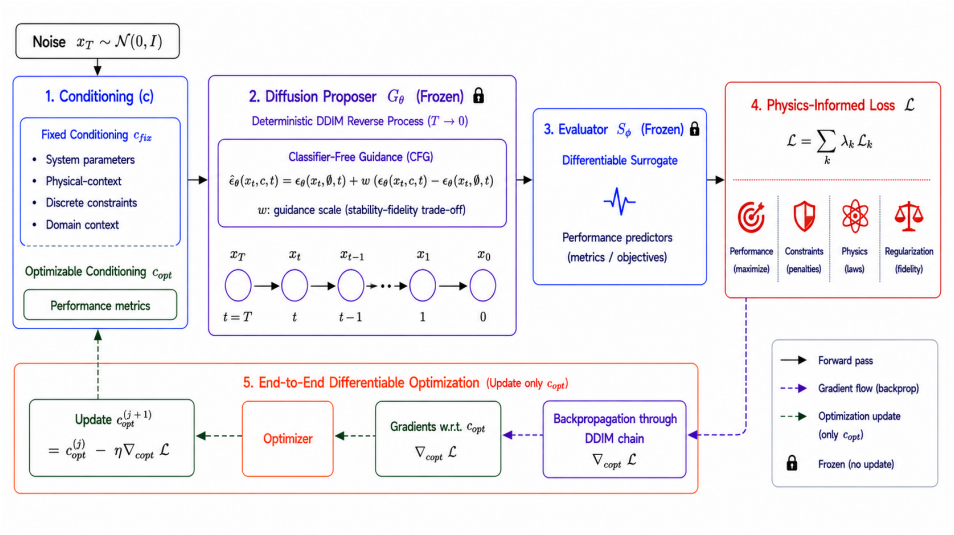

Figure 1: Overview of D2C. A frozen conditional diffusion proposer generates designs from conditioning inputs, a frozen surrogate evaluator and physics-informed penalties define a differentiable test-time objective, and gradients are backpropagated through the deterministic DDIM chain to optimize the conditioning signal. Multiple noise trials and loss-weight samples produce a candidate pool, from which the best valid design is selected after domain-specific constraint checks.

2.2 Deterministic DDIM Sampling

Denoising Diffusion Implicit Models (DDIM) [29] generalize the sampling process by considering a non-Markovian forward process that yields the same marginals as DDPM. Crucially, DDIM allows for a deterministic mapping from a latent noise x_T to a clean sample x_0. The update rule is given by:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, c, t)}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_t, c, t) \tag{3}$$

Eq. (3) gives the fully deterministic DDIM update ($\sigma_t = 0$), in which the stochastic noise term is omitted and the sampling chain becomes a deterministic discrete mapping from x_T to x_0. This update can be viewed as a discrete analogue of the probability-flow ODE perspective, but here we rely only on the finite-step deterministic map induced by the sampler. This determinism ensures that the mapping $G_\theta : (c, x_T) \mapsto x_0$ is differentiable with respect to the conditioning input c.

2.3 Classifier-Free Guidance (CFG)

Classifier-Free Guidance is a technique to improve the adherence of the generated samples to the conditioning signal c. During training, the conditioning c is randomly replaced with a null token $\emptyset$ with probability p_{drop}. At inference time, the noise estimate is interpolated between the conditional and unconditional predictions:

$$\tilde{\epsilon}_\theta(x_t, c, t) = \epsilon_\theta(x_t, \emptyset, t) + w(\epsilon_\theta(x_t, c, t) - \epsilon_\theta(x_t, \emptyset, t)) \tag{4}$$

where $w \geq 0$ is the guidance scale ($w = 0$ yields unconditional sampling, $w = 1$ recovers standard conditional sampling, and $w > 1$ amplifies conditioning adherence). In D2C, we use this guidance not only to bias sampling but also to provide a stable optimization regime in which gradient-based updates of c can proceed.

3 Methodology

We present Differentiable Diffusion Conditioning (D2C), a framework that enables a pretrained conditional diffusion model to function as an end-to-end gradient-based optimizer for physical design. The key idea is a composed proposer-evaluator architecture in which a conditional diffusion model serves as a differentiable proposer and a frozen forward surrogate provides a training-time consistency signal and inference-time gradient guidance. Algorithm 1 summarizes the full pipeline. We describe each component below.

3.1 Problem Formulation

Let $x \in \mathcal{X} \subseteq \mathbb{R}^d$ denote a physical design (e.g. a temporal power profile or a laser pulse shape) and $c \in \mathcal{C} \subseteq \mathbb{R}^{d_c}$ its associated conditioning vector encoding environmental state and performance targets. We decompose the conditioning into a fixed component c_{fix} (immutable physical context such as ambient conditions) and an optimizable component c_{opt} (optimizable performance targets):

$$c = (c_{\text{fix}},\ c_{\text{opt}}), \qquad c_{\text{fix}} \in \mathbb{R}^{d_f},\ \ c_{\text{opt}} \in \mathbb{R}^{d_o},\ \ d_f + d_o = d_c. \tag{5}$$

Given a pretrained conditional diffusion model with frozen parameters θ and a deterministic reverse chain $G_\theta : \mathcal{C} \times \mathbb{R}^d \to \mathcal{X}$ mapping conditioning and a latent noise vector to a design $x_0 = G_\theta(c, x_T)$, we seek, for a *fixed* noise realization x_T:

$$c^*_{\text{opt}}(x_T) = \arg\min_{c_{\text{opt}}}\ \mathcal{L}\big(G_\theta((c_{\text{fix}},\ c_{\text{opt}}),\ x_T)\ ,\ x_{\text{ref}},\ c_{\text{fix}}\big), \qquad x_T \sim \mathcal{N}(0, \mathbf{I}). \tag{6}$$

Each draw x_T induces a different deterministic mapping $G_\theta(\cdot, x_T)$; the optimization is therefore conditional on the noise realization. To cover multiple modes, we solve Eq. (6) independently for N_z draws and select the best feasible solution (Section 3.6). Here x_{ref} is the reference (baseline) design and $\mathcal{L}$ is a composite physics-informed loss detailed in Section 3.5.

3.2 Composed Proposer–Evaluator Architecture

D2C employs a composed proposer-evaluator architecture consisting of a conditional diffusion model G_θ (the *proposer*) that generates designs from conditioning inputs, and a forward surrogate S_ϕ (the *evaluator*) that predicts performance metrics from designs:

$$\hat{y} = S_\phi(x,\ c_{\text{fix}}), \qquad \hat{y} \in \mathbb{R}^{d_y}, \tag{7}$$

where $\hat{y}$ are predicted scalar quantities (e.g., energy consumption, yield, carbon emissions). The surrogate S_ϕ is trained independently on the same dataset and remains frozen throughout.

This architecture serves two roles: providing a surrogate consistency signal during training (Section 3.3) and an objective gradient $\nabla_{x_0}\mathcal{L}_{\text{obj}}$ during inference-time optimization (Section 3.5).

3.3 Training the Conditional Diffusion Proposer

The proposer G_θ is trained offline on a dataset $\mathcal{D} = \{(c_{\text{fix},i},\ x_i,\ y_i)\}_{i=1}^N$, where $y_i \in \mathbb{R}^{d_o}$ are observed performance metrics. The full conditioning is assembled as $c_i = (c_{\text{fix},i},\ y_i)$, so the metrics serve dual roles: as supervised labels for the surrogate S_ϕ and as the optimizable conditioning targets c_{opt} that steer generation. The training objective combines the standard denoising score-matching loss with auxiliary terms that improve generation fidelity:

$$\mathcal{L}_{\text{train}} = \underbrace{\|\epsilon - \epsilon_\theta(x_t, c, t)\|^2}_{\text{denoising}} + \alpha_r \underbrace{\|\hat{x}_0 - x\|^2}_{\text{reconstruction}} + \alpha_a \underbrace{\mathcal{L}_{\text{fid}}(\hat{x}_0, x)}_{\text{physics-based fidelity}} + \alpha_s \underbrace{\|S_\phi(\hat{x}_0, c_{\text{fix}}) - y\|^2}_{\text{surrogate consistency}}, \tag{8}$$

where $x_t = \sqrt{\bar{\alpha}_t}\, x + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$ is the noised sample with $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, and $\hat{x}_0$ is the one-step predicted clean sample:

$$\hat{x}_0 = \frac{x_t - \sqrt{1 - \bar{\alpha}_t}\, \epsilon_\theta(x_t, c, t)}{\sqrt{\bar{\alpha}_t}}. \tag{9}$$

The physics-based fidelity term $\mathcal{L}_{\text{fid}}$ encodes domain-specific structural constraints on the generated design, for example, integral conservation laws, peak-amplitude bounds, or spectral regularity. We express it generically as:

$$\mathcal{L}_{\text{fid}}(\hat{x}_0, x) = \sum_k \big[g_k(\hat{x}_0) - g_k(x)\big]_+, \tag{10}$$

where each g_k is a scalar functional encoding a physical constraint and $[\cdot]_+ = \max(\cdot, 0)$ ensures the penalty activates only when the generated design exceeds the reference on that constraint. In the smooth case, the resulting penalty is differentiable almost everywhere; some instantiated constraints additionally use non-smooth operators such as coordinatewise $\max$ / $\min$, which are handled in practice with automatic-differentiation subgradients. The concrete instantiations of $\mathcal{L}_{\text{fid}}$ and the surrogate consistency term for each application domain are detailed in Appendix E. The surrogate

consistency term leverages the composed architecture of Section 3.2: the frozen evaluator S_ϕ maps generated designs back to the performance-metric space, encouraging the proposer to produce physically consistent outputs.

To enable classifier-free guidance at inference, the conditioning is randomly replaced with a learned null embedding $\emptyset$ with probability p_{drop} during training. Exponential moving averaging (EMA) of model weights with decay β further stabilizes generation quality. Once training converges, all proposer parameters θ are frozen.

3.4 Differentiable DDIM Reverse Chain

The key observation is that when the DDIM sampler uses zero noise ($\sigma_t = 0$), the entire reverse chain becomes a deterministic, differentiable function of its inputs, including the conditioning c. With the proposer frozen, we establish the differentiable path from $\mathcal{L}$ to c_{opt} required by Eq. (6). Each reverse step takes the form:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\,\hat{x}_0(x_t, c, t) + \sqrt{1-\bar{\alpha}_{t-1}}\,\tilde{\epsilon}_\theta(x_t, c, t), \tag{11}$$

where $\hat{x}_0(x_t, c, t) = \big(x_t - \sqrt{1-\bar{\alpha}_t}\,\tilde{\epsilon}_\theta\big)/\sqrt{\bar{\alpha}_t}$ is the predicted clean sample, and $\tilde{\epsilon}_\theta$ is the CFG-guided noise estimate [14]:

$$\tilde{\epsilon}_\theta(x_t, c, t) = \epsilon_\theta(x_t, \emptyset, t) + w\big(\epsilon_\theta(x_t, c, t) - \epsilon_\theta(x_t, \emptyset, t)\big), \tag{12}$$

with $\emptyset$ the learned null embedding and $w \geq 0$ the guidance scale. Because every operation is deterministic and differentiable, the full T-step composition $G_\theta(c, x_T) = x_0$ is differentiable with respect to c. The conditioning c is injected at every timestep, providing gradient shortcut connections that mitigate vanishing gradients across deep chains. The guidance scale w modulates the magnitude of $\partial\tilde{\epsilon}_\theta/\partial c$, controlling both generation fidelity and optimization stability.

3.5 Inference-Time Conditioning Optimization

With G_θ and S_ϕ, we optimize c_{opt} by backpropagating a composite loss through the DDIM chain. The loss is a weighted combination of K differentiable penalty terms:

$$\mathcal{L} = \sum_{k=1}^{K} \lambda_k\,\mathcal{L}_k(x_0,\, x_{\text{ref}},\, c_{\text{fix}}), \tag{13}$$

where each $\mathcal{L}_k$ is a scalar loss computed from the generated design $x_0 = G_\theta(c, x_T)$, the reference design x_{ref}, and/or the fixed conditioning c_{fix} (e.g., the surrogate evaluation $S_\phi(x_0, c_{\text{fix}})$), and the weights $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_K)$ control the trade-off among competing objectives. Crucially, no loss term depends directly on the optimizable component c_{opt}; all losses are functions of the *realized design* x_0 only. The conditioning c_{opt} influences the loss exclusively through its effect on the generated design via $x_0 = G_\theta((c_{\text{fix}}, c_{\text{opt}}), x_T)$, ensuring that any reported improvement reflects a genuine change in the physical design rather than a shift in target labels. In the instantiated losses used here, some penalty terms include non-smooth operators such as coordinatewise $\max$ / $\min$, so optimization is implemented through the deterministic reverse chain with practical automatic-differentiation subgradients where needed.

The loss terms are domain-specific but follow a common taxonomy. A *surrogate objective* term uses the frozen evaluator to maximize a target metric (e.g., $\mathcal{L}_{\text{obj}} = -S_\phi(x_0, c_{\text{fix}})_k$ for target metric k); *constraint* terms penalize violations of physical invariants (integral conservation, amplitude bounds, spectral stability); and a *smoothness* term enforces temporal or spatial regularity. The task-specific loss terms are constructed to be on broadly comparable scales; the concrete loss definitions for each application are given in Appendix F.

At each optimization step, the conditioning is updated via gradient descent:

$$c_{\text{opt}}^{(j+1)} = c_{\text{opt}}^{(j)} - \eta\,\nabla_{c_{\text{opt}}}\mathcal{L}\big(G_\theta(c^{(j)}, x_T),\, x_{\text{ref}},\, c_{\text{fix}}\big), \tag{14}$$

where $j = 0, \ldots, J-1$ indexes optimization steps, η is the learning rate, and x_T is held fixed throughout. In practice, any first-order optimizer (e.g., Adam) can replace vanilla gradient descent; the key requirement is differentiability of the full chain. Crucially, only c_{opt} receives gradient updates;

θ and ϕ remain frozen. This mitigates the risk of surrogate exploitation common in unconstrained model-based optimization [30]: under suitably tuned guidance, the generative model helps keep designs closer to the empirical data distribution induced by the learned prior while the surrogate provides only a directional gradient signal.

3.6 Stochastic Coverage and Candidate Selection

Since the mapping $G_\theta(c, x_T)$ depends on the latent noise x_T, a single noise draw may not expose the best mode of the conditional distribution. To ensure broad coverage without sacrificing differentiability, we adopt a two-level diversification strategy.

Noise trials. For each input, we draw N_z independent noise vectors $\{x_T^{(n)}\}_{n=1}^{N_z}$, each held fixed during optimization. Each trial explores a different region of the latent space, producing structurally distinct candidates.

Weight diversity. Within each noise trial, we instantiate N_λ loss-weight vectors $\boldsymbol{\lambda}^{(m)} \sim \mathrm{Dir}(\boldsymbol{\alpha})$ drawn from a symmetric Dirichlet distribution [8] with $\alpha_k = 1$ for all k (i.e., uniform over the simplex; see Appendix G for details). The Dirichlet distribution ensures each weight vector sums to one and covers the full trade-off simplex uniformly, so no single objective dominates across all trajectories. This yields $B = N_z \times N_\lambda$ parallel trajectories per sample, each emphasizing a different trade-off among competing objectives.

Candidate selection. After J optimization steps, each trajectory produces a candidate $x_0^{(b)}$. Each candidate is then validated against domain-specific hard physical constraints $\mathcal{C}(x_0) \leq \mathbf{0}$ (e.g., temperature limits, integral tolerances; see Appendix C for details). Among candidates that satisfy the hard constraints, the one with the best primary objective is selected; if no candidate passes all constraints, the baseline design x_{ref} is retained (yielding 0% gain).

Algorithm 1 D2C: Differentiable Diffusion Conditioning

Require: Dataset $\mathcal{D} = \{(c_{\mathrm{fix},i},\, x_i,\, y_i)\}$ with $c_i = (c_{\mathrm{fix},i},\, y_i)$; test sample $(c,\, x_{\mathrm{ref}})$
Require: Noise trials N_z, weight samples N_λ, optimization steps J, guidance scale w
Train surrogate S_ϕ on $\{(x_i,\, c_{\mathrm{fix},i}) \to y_i\}$; freeze ϕ
Train ϵ_θ on $\mathcal{D}$ using $\mathcal{L}_{\mathrm{train}}$ (Eq. 8) with CFG dropout p_{drop} and frozen S_ϕ; freeze θ

Decompose $c = (c_{\mathrm{fix}},\, c_{\mathrm{opt}}^{(0)})$
$\mathcal{R} \leftarrow \emptyset$ {candidate pool}
for $n = 1, \ldots, N_z$ **do**
  Sample $x_T^{(n)} \sim \mathcal{N}(0, \mathbf{I})$
  for $m = 1, \ldots, N_\lambda$ **do**
    Sample $\boldsymbol{\lambda}^{(m)} \sim \mathrm{Dir}(\boldsymbol{\alpha})$
    Initialize $c_{\mathrm{opt}} \leftarrow c_{\mathrm{opt}}^{(0)}$
    for $j = 1, \ldots, J$ **do**
      $x_0 \leftarrow G_\theta\big((c_{\mathrm{fix}},\, c_{\mathrm{opt}}),\, x_T^{(n)};\, w\big)$ {Eqs. 11–12}
      $\mathcal{L} \leftarrow \sum_k \lambda_k^{(m)}\, \mathcal{L}_k(x_0,\, x_{\mathrm{ref}},\, c_{\mathrm{fix}})$ {Eq. 13}
      $c_{\mathrm{opt}} \leftarrow c_{\mathrm{opt}} - \eta\, \nabla_{c_{\mathrm{opt}}} \mathcal{L}$ {backprop through G_θ, S_ϕ}
    end for
    $\mathcal{R} \leftarrow \mathcal{R} \cup \big\{G_\theta\big((c_{\mathrm{fix}},\, c_{\mathrm{opt}}),\, x_T^{(n)};\, w\big)\big\}$
  end for
end for

$\mathcal{V} \leftarrow \{x \in \mathcal{R} \mid \mathcal{C}(x) \leq \mathbf{0}\}$ {filter by hard constraints}
if $\mathcal{V} \neq \emptyset$ **then**
  return $\arg\min_{x \in \mathcal{V}} \mathcal{L}_{\mathrm{obj}}(x)$ {surrogate objective}
else
  return x_{ref} {no valid improving candidate; retain baseline (0% gain)}
end if

4 Experiments

We evaluate D2C on two sets of tasks: (i) two real-world physical design problems where constraint satisfaction is safety-critical, and (ii) the standard Design-Bench [31] benchmark suite for offline model-based optimization. Although D2C is motivated by constrained physical domains, we also evaluate it on Design-Bench to test whether the same mechanism remains strong on standard offline MBO benchmarks beyond the application domains that motivate the method. Additional implementation, hyperparameter, and evaluation details are provided in Appendices A and B.4.

4.1 Real-World Physical Design Tasks

Inertial Confinement Fusion (ICF) laser pulse optimization. In ICF, a precisely shaped laser pulse drives the implosion of a deuterium-tritium fuel capsule to achieve thermonuclear ignition. The design variable $x \in \mathbb{R}^{256}$ is a temporal laser power profile discretized into 256 time steps, and the conditioning signal c encodes target specifications (c_{fix}, 7 dimensions) and 20 performance metrics (c_{opt}) derived from LILAC, the radiation-hydrodynamics simulation code used for inertial-confinement-fusion design studies. At inference, D2C optimizes the 20 performance targets to steer the proposer toward higher-yield pulses. The primary objective is to maximize neutron yield, subject to constraints on pulse energy conservation (the integral of the optimized pulse must match the reference), temporal smoothness (to ensure manufacturability), and spectral fidelity (the frequency content must remain physically plausible). Designs that violate these constraints are operationally unsafe and cannot be fielded on a real facility; the concrete validation criteria are detailed in Appendix C.

Data center workload scheduling. For this task we follow the SustainDC [24] framework where the goal is to reduce the carbon footprint of large-scale data centers, which requires scheduling computational workloads to exploit temporal variations in grid carbon intensity and cooling efficiency. Here, $x \in \mathbb{R}^{144}$ is a 24-hour workload power profile at 10-minute resolution, and c encodes environmental state variables (c_{fix}, 73 dimensions: ambient temperatures, cooling set-points, IT demand) and energy metrics (c_{opt}, 10 dimensions: total power, carbon intensity, etc.). At inference, D2C optimizes the 10 energy metric targets to minimize emissions. The objective is to minimize operational carbon emissions while satisfying a hard workload conservation constraint: the total compute delivered must equal the demand (the integral of the optimized profile must match the reference within a tight tolerance). Additional constraints include inlet temperature safety limits. Final designs are validated against the full SustainDC thermal model (Appendix C).

4.2 Design-Bench Benchmarks

To position D2C among established MBO methods, we evaluate on four tasks from Design-Bench [31]: **Ant Morphology** (continuous, 60-d), **D'Kitty Morphology** (continuous, 56-d), **TF Bind 8** (discrete, 8-d), and **TF Bind 10** (discrete, 10-d). These tasks span transcription factor binding (DNA sequence design) and robot morphology optimization, providing a diverse testbed for offline optimization from fixed datasets. The goal in each task is to maximize a black-box oracle score using only the offline dataset.

4.3 Baselines

We compare against 12 baselines, 11 of which are commonly used in offline MBO: Gradient Ascent, COMs [30], BDI [4], CbAS [2], MINs [19], REINFORCE [34], DDOM [18], GTG [39], PGS [3], GP-qEI [28], CMA-ES [13], and Brute Force, a diffusion-based approach that replaces gradient-based conditioning optimization with a forward-only sweep over c_{opt} (Appendix A). For the real-world tasks, all surrogate-based baselines share the same frozen S_ϕ.

4.4 Main Results

Table 1 reports results on the real-world tasks. All results report mean $\pm$ standard deviation over 5 independent runs with different random seeds; the same seeds and test samples are used across all methods. Following the standard MBO candidate-generation protocol [31, 18, 39], all baselines generate 128 candidate designs per test sample; D2C generates 100 candidates per sample. For

Table 1: Real-world physical design tasks. Values are mean percentage improvement (%) over the baseline design ± std over 5 runs. **Bold**: best.

Method	ICF Standard	ICF Top	DC
Gradient Ascent	$2.31\% \pm 0.31$	$1.46\% \pm 0.29$	$0.14\% \pm 0.04$
COMs	$3.14\% \pm 0.41$	$0.43\% \pm 0.11$	$1.33\% \pm 0.24$
BDI	$2.42\% \pm 0.37$	$1.33\% \pm 0.21$	$0.37\% \pm 0.08$
CbAS	$4.49\% \pm 0.63$	$1.86\% \pm 0.29$	$1.85\% \pm 0.31$
MINs	$3.71\% \pm 0.67$	$1.51\% \pm 0.43$	$1.21\% \pm 0.23$
REINFORCE	$3.50\% \pm 0.56$	$0.32\% \pm 0.08$	$1.26\% \pm 0.25$
DDOM	$4.55\% \pm 0.68$	$2.24\% \pm 0.25$	$4.67\% \pm 0.59$
GTG	$5.43\% \pm 0.74$	$2.65\% \pm 0.38$	$5.50\% \pm 0.68$
PGS	$2.47\% \pm 0.42$	$1.73\% \pm 0.27$	$0.42\% \pm 0.10$
GP-qEI	$3.58\% \pm 0.53$	$1.03\% \pm 0.18$	$5.93\% \pm 0.82$
CMA-ES	$4.13\% \pm 0.79$	$1.24\% \pm 0.27$	$3.98\% \pm 0.63$
Brute Force	$4.66\% \pm 0.83$	$3.93\% \pm 0.52$	$1.14\% \pm 0.22$
D2C (Ours)	$\mathbf{17.98\% \pm 1.54}$	$\mathbf{10.06\% \pm 0.88}$	$\mathbf{8.98\% \pm 0.75}$

the real-world tasks, these 100 candidates arise from N_z=20 noise trials × N_λ=5 weight samples; for Design-Bench, we use 100 latent/noise trials (N_λ=1 since single objective) with the Design-Bench-specific setup described in Appendix B.4. For the real-world tasks we report gains in terms of % neutron yield improvement for ICF and % carbon-footprint reduction for the data center task. For ICF, final performance is evaluated by a separate LILAC-validated evaluation oracle (Appendix B.2); for DC, generated workloads are evaluated directly on the SustainDC physics-based simulator. For Design-Bench, we report 100th-percentile normalized score of the best design found as is standard [31, 18, 39]. ICF Standard evaluates optimization over 2,000 randomly sampled pulses from the test set (the same 2,000 for all methods); ICF Top selects the 1,000 highest-yield designs from the test set and optimizes only those, stress-testing each method on the hardest cases where the baseline design is already among the best. For all tasks, only candidates that pass the domain-specific hard constraints (Section 3.6) are counted as improvements; samples where no valid improving candidate is found retain the baseline design and contribute 0% gain.

Real-world tasks. D2C achieves the highest improvement on all three real-world settings (Table 1). On ICF Standard, D2C improves yield by 17.98%, outperforming all baselines by a large margin. On ICF Top, where the baseline designs are already high-yield, D2C achieves 10.06%, more than two and a half times the best baseline, Brute Force (3.93%). This suggests that differentiable guidance through the DDIM chain enables more precise conditioning than fixed-target generation, especially in the high-performance regime. On data center scheduling (DC), D2C reduces carbon emissions by 8.98%, surpassing the best baseline GP-qEI (5.93%) and the best generative baseline GTG (5.50%).

Design-Bench tasks. D2C achieves the best average rank across the four Design-Bench tasks, placing first on TF Bind 8 and TF Bind 10, second on D'Kitty, and third on Ant. We therefore position D2C as robust across standard MBO benchmarks, with the strongest overall performance by average rank.

Additional ablations on guidance scale (including its selection without oracle access), optimization steps, learning rate, DDIM steps, optimizable conditioning dimensionality, conditioning versus latent-noise optimization, constraint-satisfying improvement rate, candidate budget, and computational cost are reported in Section D. In particular, with the proposer, surrogate, and evaluation protocol held fixed, optimizing the conditioning outperforms optimizing the latent noise on both real-world tasks (Section D.7).

5 Related Work

Offline model-based optimization (MBO) seeks high-scoring designs using only a fixed dataset, with no online evaluations [31]. Existing approaches include forward methods that optimize a learned surrogate in design space [30, 9, 36, 37, 5], inverse methods that map target scores to designs with increasingly expressive generators [2, 7, 19, 16, 18], and trajectory-based methods that learn to

Table 2: Design-Bench tasks. 100th-percentile normalized score $\pm$ std over 5 runs. The row $\mathcal{D}$ *(best)* reports the best design present in the offline dataset for each task. Scores above $\mathcal{D}$ *(best)* therefore indicate improvement beyond the best offline example under the evaluation protocol used for that column. Avg. Rank is the mean ordinal rank across all four tasks ($\downarrow$ lower is better). **Bold**: best per column.

Method	TF Bind 8	TF Bind 10	D'Kitty	Ant	Avg. Rank
$\mathcal{D}$ (best)	0.439	0.467	0.884	0.565	–
Gradient Ascent	0.854 ± 0.029	0.575 ± 0.039	0.937 ± 0.014	0.749 ± 0.034	10.38
COMs	0.949 ± 0.018	0.598 ± 0.034	0.935 ± 0.013	0.863 ± 0.021	6.75
BDI	0.969 ± 0.012	0.609 ± 0.026	0.931 ± 0.014	0.717 ± 0.032	7.00
CbAS	0.918 ± 0.027	0.607 ± 0.032	0.908 ± 0.017	0.672 ± 0.035	11.25
MINs	0.923 ± 0.026	0.625 ± 0.033	0.917 ± 0.017	0.806 ± 0.030	8.50
REINFORCE	0.949 ± 0.022	0.594 ± 0.035	0.941 ± 0.013	0.852 ± 0.028	7.00
DDOM	0.964 ± 0.015	0.682 ± 0.025	0.915 ± 0.016	$\mathbf{0.925 \pm 0.019}$	4.38
GTG	0.949 ± 0.020	0.635 ± 0.030	$\mathbf{0.976 \pm 0.009}$	0.905 ± 0.022	3.50
PGS	0.949 ± 0.019	0.575 ± 0.037	0.915 ± 0.016	0.853 ± 0.027	9.00
GP-qEI	0.924 ± 0.024	0.583 ± 0.036	0.948 ± 0.012	0.801 ± 0.029	8.50
CMA-ES	0.949 ± 0.019	0.638 ± 0.019	0.929 ± 0.015	0.843 ± 0.023	6.50
Brute Force	0.968 ± 0.014	0.611 ± 0.028	0.888 ± 0.019	0.874 ± 0.026	6.50
D2C (Ours)	$\mathbf{0.973 \pm 0.011}$	$\mathbf{0.684 \pm 0.023}$	0.974 ± 0.009	0.876 ± 0.020	**1.75**

improve from synthetic optimization paths [23, 3, 39]. Forward methods remain limited by surrogate extrapolation, while inverse and trajectory-based methods typically keep the conditioning target fixed at inference time rather than refining it against downstream objectives.

Recent diffusion-based optimization methods attempt to integrate optimization feedback into generation, including guided warm-starts with Langevin corrections [17], uncertainty-aware conditioning selection [35], editing-based optimization against pseudo-targets [38], and end-to-end gradient propagation through diffusion samplers in image-generation settings [33, 27]. However, these methods either operate outside offline MBO, rely on post-hoc corrections or heuristic target selection, or do not study physics-informed constrained design with a frozen proposer-evaluator architecture. Unlike these methods, D2C targets offline MBO and treats the deterministic reverse diffusion chain as a differentiable map in the conditioning signal itself, optimizing that conditioning directly against surrogate and physics-based penalty losses, with final validity enforced by domain-specific hard constraints.

6 Conclusion and Limitations

We introduced Differentiable Diffusion Conditioning (D2C), a framework that turns a pretrained conditional diffusion model into an end-to-end gradient-based optimizer for physical design. By establishing a practical backpropagation path from composite physics-informed objectives, through a frozen surrogate evaluator, back through the deterministic DDIM reverse chain to the conditioning input, D2C enables gradient-based optimization without retraining or fine-tuning the generative model. The composed proposer-evaluator architecture of D2C helps keep generated designs closer to the empirical data distribution induced by the learned prior under appropriately tuned guidance while the surrogate provides directional gradient signals, helping reduce the out-of-distribution exploitation that plagues direct surrogate optimization. Across two real-world scientific tasks and four Design-Bench tasks, D2C delivers the strongest results on the real-world tasks and the best average rank across the benchmark suite.

6.1 Limitations

- Computational cost. D2C requires backpropagation through a T-step DDIM chain at each optimization iteration, making it roughly 8–17$\times$ slower than single-pass methods (Table 16). While our ablations show that T and J can be reduced with minimal quality loss, the method remains unsuitable for latency-critical applications.
- Surrogate dependence. The quality of the gradient signal is bounded by the accuracy of the frozen surrogate S_ϕ. In regimes where the surrogate is poorly calibrated, the optimization may converge

to high-scoring surrogate designs that perform poorly on the true objective. Future work could incorporate uncertainty quantification into the surrogate to mitigate this.

- Validity-first selection. D2C combines differentiable penalties with a final validity-first selection step. As a result, method quality should be interpreted jointly through both achieved gain and the rate at which valid improving solutions are found.
- Domain-specific tuning. While D2C is broadly applicable, the choice of loss terms, constraint definitions, and optimization hyperparameters requires domain expertise and may involve trial-and-error tuning for new applications. Appendix I provides extended practical guidance.

Acknowledgments

This work was supported by the U.S. Department of Energy, Office of Science, under Award No. DE-SC0024408 and DE-SC0024381.

References

[1] Alibaba Group. Alibaba cluster trace program. `https://github.com/alibaba/clusterdata`, 2026.

[2] David Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In *ICML*, 2019.

[3] Yassine Chemingui, Aryan Deshwal, Trong Nghia Hoang, and Janardhan Rao Doppa. Offline model-based optimization via policy-guided gradient search. In *AAAI*, 2024.

[4] Can Chen, Yingxue Zhang, Jie Fu, Xue Liu, and Mark Coates. Bidirectional learning for offline infinite-width model-based optimization. In *NeurIPS*, 2022.

[5] Can Sam Chen, Christopher Beckham, Zixuan Liu, Xue Steve Liu, and Chris Pal. Parallel mentoring for offline model-based optimization. In *NeurIPS*, 2023.

[6] J. Delettrez, R. Epstein, M. C. Richardson, P. A. Jaanimagi, and B. L. Henke. Effect of laser illumination nonuniformity on the analysis of time-resolved x-ray measurements in uv spherical transport experiments. *Phys. Rev. A*, 36:3926–3934, Oct 1987. doi: 10.1103/PhysRevA.36.3926. URL `https://link.aps.org/doi/10.1103/PhysRevA.36.3926`.

[7] Clara Fannjiang and Jennifer Listgarten. Autofocused oracles for model-based design. In *NeurIPS*, 2020.

[8] Bela A. Frigyik, Amrita Kapila, and Maya R. Gupta. Introduction to the Dirichlet distribution and related processes. Technical Report UWEETR-2010-0006, University of Washington, Department of Electrical Engineering, 2010.

[9] Justin Fu and Sergey Levine. Offline model-based optimization via normalized maximum likelihood estimation. In *ICLR*, 2021.

[10] V Gopalaswamy, R Betti, J P Knauer, N Luciani, D Patel, K M Woo, A Bose, I V Igumenshchev, E M Campbell, K S Anderson, K A Bauer, M J Bonino, D Cao, A R Christopherson, G W Collins, T J B Collins, J R Davies, J A Delettrez, D H Edgell, R Epstein, C J Forrest, D H Froula, V Y Glebov, V N Goncharov, D R Harding, S X Hu, D W Jacobs-Perkins, R T Janezic, J H Kelly, O M Mannion, A Maximov, F J Marshall, D T Michel, S Miller, S F B Morse, J Palastro, J Peebles, P B Radha, S P Regan, S Sampat, T C Sangster, A B Sefkow, W Seka, R C Shah, W T Shmyada, A Shvydky, C Stoeckl, A A Solodov, W Theobald, J D Zuegel, M Gatu Johnson, R D Petrasso, C K Li, and J A Frenje. Tripled yield in direct-drive laser fusion through statistical modelling. *Nature*, 565(7741):581–586, January 2019.

[11] V Gopalaswamy, C A Williams, R Betti, D Patel, J P Knauer, A Lees, D Cao, E M Campbell, P Farmakis, R Ejaz, K S Anderson, R Epstein, J Carroll-Nellenback, I V Igumenshchev, J A Marozas, P B Radha, A A Solodov, C A Thomas, K M Woo, T J B Collins, S X Hu, W Scullin, D Turnbull, V N Goncharov, K Churnetski, C J Forrest, V Yu Glebov, P V Heuer,

H McClow, R C Shah, C Stoeckl, W Theobald, D H Edgell, S Ivancic, M J Rosenberg, S P Regan, D Bredesen, C Fella, M Koch, R T Janezic, M J Bonino, D R Harding, K A Bauer, S Sampat, L J Waxer, M Labuzeta, S F B Morse, M Gatu-Johnson, R D Petrasso, J A Frenje, J Murray, B Serrato, D Guzman, C Shuldberg, M Farrell, and C Deeney. Demonstration of a hydrodynamically equivalent burning plasma in direct-drive inertial confinement fusion. *Nature Physics*, 20(5):751–757, May 2024.

[12] E C Hansen, D H Barnak, R Betti, E M Campbell, P-Y Chang, J R Davies, V Yu Glebov, J P Knauer, J Peebles, S P Regan, and A B Sefkow. Measuring implosion velocities in experiments and simulations of laser-driven cylindrical implosions on the OMEGA laser. *Plasma Physics and Controlled Fusion*, 60(5):054014, apr 2018. doi: 10.1088/1361-6587/aab73f. URL `https://dx.doi.org/10.1088/1361-6587/aab73f`.

[13] Nikolaus Hansen. The CMA evolution strategy: A tutorial. *arXiv:1604.00772*, 2016.

[14] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[16] Minsu Kim, Federico Berto, Sungsoo Ahn, and Jinkyoo Park. Bootstrapped training of score-conditioned generator for offline design of biological sequences. In *NeurIPS*, 2023.

[17] Lingkai Kong, Yuanqi Du, Wenhao Mu, Kirill Neklyudov, Valentin De Bortoli, Dongxia Wu, Haorui Wang, Aaron Ferber, Yi-An Ma, Carla P. Gomes, and Chao Zhang. Diffusion models as constrained samplers for optimization with unknown constraints. In *AISTATS*, 2025. URL `https://arxiv.org/abs/2402.18012`.

[18] Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion models for black-box optimization. In *ICML*, 2023.

[19] Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. In *NeurIPS*, 2020.

[20] A. Lees, R. Betti, J. P. Knauer, V. Gopalaswamy, D. Patel, K. M. Woo, K. S. Anderson, E. M. Campbell, D. Cao, J. Carroll-Nellenback, R. Epstein, C. Forrest, V. N. Goncharov, D. R. Harding, S. X. Hu, I. V. Igumenshchev, R. T. Janezic, O. M. Mannion, P. B. Radha, S. P. Regan, A. Shvydky, R. C. Shah, W. T. Shmyada, C. Stoeckl, W. Theobald, and C. Thomas. Experimentally inferred fusion yield dependencies of OMEGA inertial confinement fusion implosions. *Phys. Rev. Lett.*, 127:105001, Aug 2021. doi: 10.1103/PhysRevLett.127.105001. URL `https://link.aps.org/doi/10.1103/PhysRevLett.127.105001`.

[21] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *ICLR*, 2019.

[22] Ricardo Luna Gutierrez, Vineet Gundecha, Rahman Ejaz, Varchas Gopalaswamy, Riccardo Betti, Sahand Ghorbanpour, Aarne Lees, and Soumyendu Sarkar. IM-LPG: Inverse modeling approach to laser pulse shape generation in inertial confinement fusion. In *AI for Science Workshop at NeurIPS 2025*, 2025. URL `https://openreview.net/forum?id=Wjh3yUUAoS`.

[23] Satvik Mehul Mashkaria, Siddarth Krishnamoorthy, and Aditya Grover. Generative pretraining for black-box optimization. In *ICML*, 2023.

[24] Avisek Naug, Antonio Guillen, Ricardo Luna Gutierrez, Vineet Gundecha, Cullen E. Bash, Sahand Ghorbanpour, Sajad Mousavi, Ashwin Ramesh Babu, Dejan Markovikj, Lekhapriya Dheeraj Kashyap, Desik Rengarajan, and Soumyendu Sarkar. SustainDC: Benchmarking for sustainable data center control. In *NeurIPS*, 2024.

[25] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, Boston, MA, 2004.

[26] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *ICML*, 2021.

[27] Jiachun Pan, Jun Hao Liew, Vincent Y. F. Tan, Jiashi Feng, and Hanshu Yan. AdjointDPM: Adjoint sensitivity method for gradient backpropagation of diffusion probabilistic models. In *ICLR*, 2024.

[28] Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical Bayesian optimization of machine learning algorithms. In *NeurIPS*, 2012.

[29] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[30] Brandon Trabucco, Aviral Kumar, Xinyang Geng, and Sergey Levine. Conservative objective models for effective offline model-based optimization. In *ICML*, 2021.

[31] Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-Bench: Benchmarks for data-driven offline model-based optimization. In *ICML*, 2022.

[32] University of Rochester, Laboratory for Laser Energetics. LLE review, volume 85. Technical Report DOE/SF/19460-378, University of Rochester, Laboratory for Laser Energetics, October–December 2000. URL `https://www.lle.rochester.edu/media/publications/lle_review/documents/v85/85_Review.pdf`. Quarterly Report.

[33] Bram Wallace, Akash Gokul, Stefano Ermon, and Nikhil Naik. End-to-end diffusion latent optimization improves classifier guidance. In *ICCV*, pages 7246–7256, 2023. doi: 10.1109/ICCV51070.2023.00669. URL `https://doi.org/10.1109/ICCV51070.2023.00669`.

[34] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992.

[35] Dongxia Wu, Nikki Lijing Kuang, Ruijia Niu, Yi-An Ma, and Rose Yu. Diffusion-BBO: Diffusion-based inverse modeling for online black-box optimization, 2025. URL `https://arxiv.org/abs/2407.00610`.

[36] Sihyun Yu, Sungsoo Ahn, Le Song, and Jinwoo Shin. RoMA: Robust model adaptation for offline model-based optimization. In *NeurIPS*, 2021.

[37] Ye Yuan, Can Sam Chen, Zixuan Liu, Willie Neiswanger, and Xue Steve Liu. Importance-aware co-teaching for offline model-based optimization. In *NeurIPS*, 2023.

[38] Ye Yuan, Youyuan Zhang, Can Chen, Haolun Wu, Zixuan Li, Jianmo Li, James J. Clark, and Xue Liu. Design editing for offline model-based optimization. *Transactions on Machine Learning Research*, 2025. URL `https://arxiv.org/abs/2405.13964`.

[39] Taeyoung Yun, Sujin Yun, Jaewoo Lee, and Jinkyoo Park. Guided trajectory generation with diffusion models for offline model-based optimization. In *NeurIPS*, 2024.

A Experimental Details

A.1 Computational Resources

All experiments were conducted on a node equipped with four NVIDIA H100 GPUs. Training the diffusion proposer and the surrogate evaluator together on a single GPU takes approximately 20 hours per training run. For the optimization time on this same hardware please refer to Subsection D.10.

A.2 Diffusion Proposer Architecture

The proposer G_θ is an MLP-based epsilon-prediction network. Both ICF and DC tasks use the same architecture design; only the input/output dimensionalities and hidden width differ. The architecture consists of:

- A **data projection** layer: $\texttt{Linear}(d_{\text{out}}, H) \to \texttt{LayerNorm} \to \texttt{SiLU}$, where d_{out} is the design dimensionality and H is the hidden dimension.
- A **time embedding** MLP: sinusoidal positional encoding of the diffusion timestep, followed by three linear layers with SiLU activations, projecting to dimension H.
- A **conditioning MLP**: three linear layers with SiLU activations mapping the full conditioning vector $c \in \mathbb{R}^{d_c}$ to dimension H.
- L **residual blocks**, each containing LayerNorm $\to$ Linear$(H, 4H) \to$ SiLU $\to$ Dropout $\to$ Linear$(4H, H) \to$ Dropout, with a learnable residual scale parameter initialized to 0.1.
- Per-block **FiLM modulation**: at each residual block, the time embedding produces scale and shift vectors via dedicated linear projections, applied as $h \leftarrow \gamma_t \odot h + \beta_t$. The conditioning embedding is added to the hidden state before the residual stack.
- A **null conditioning embedding** (learnable, $\mathbb{R}^{1\times H}$, initialized to $\mathcal{N}(0, 0.01)$) for classifier-free guidance: during training, the conditioning embedding is replaced by the null embedding with probability p_{drop}.
- An **output projection**: LayerNorm $\to$ Linear(H, d_{out}), with weights and biases initialized to zero for stable early training.

A.3 Noise Schedule

Both tasks use a cosine noise schedule [26] with $T_{\text{train}} = 100$ diffusion steps, $s = 0.008$ offset, and betas clipped to $[10^{-4}, 0.999]$:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T + s}{1+s} \cdot \frac{\pi}{2}\right)^2. \tag{15}$$

At inference, we use DDIM with $\sigma_t = 0$ (fully deterministic) and $T = 25$ inference steps (selected via ablation, Section D.5).

A.4 Training Hyperparameters

The proposer is trained with AdamW [21] using a cosine-annealed learning rate with linear warmup. EMA (Exponential Moving Average) of model weights is maintained throughout training with a warmup period of 2,000 steps. Table 3 summarizes the training hyperparameters for both domains.

A.5 Inference-Time Optimization Hyperparameters

At inference, we freeze the trained proposer and evaluator and optimize the conditioning separately for each test sample. Table 4 reports the optimization hyperparameters used for the main results (selected via ablation studies in Section D).

Hyperparameters were selected via grid search on the *validation* set over manually specified value ranges. During inference-time optimization, only the optimizable coordinates of the conditioning vector are updated; the fixed coordinates remain unchanged. After each Adam step, the updated conditioning coordinates are clipped back to the normalized $[0, 1]$ range before the next DDIM rollout, so the frozen proposer is not queried with out-of-range values on those dimensions.

Table 3: Training hyperparameters for the diffusion proposer.

Hyperparameter	ICF	DC
Hidden dimension H	1024	1024
Residual blocks L	6	6
Dropout	0.1	0.1
CFG dropout p_{drop}	0.1	0.1
EMA decay β	0.9999	0.9999
Learning rate	5×10^{-5}	5×10^{-5}
Batch size	1024	1024
Training epochs	400	5,000
Diffusion steps T_{train}	100	100
Noise schedule	Cosine	Cosine
Reconstruction weight α_r	0.5	0.5
Fidelity weight α_a	0.001	0.01
Smoothness weight α_{sm}	0.0001	0.0001
Surrogate weight α_s	1.0	1.0
Conditioning dim d_c	27	83
Design dim d_{out}	256	144

Table 4: Inference-time optimization hyperparameters.

Hyperparameter	ICF	DC
Guidance scale w	2.0	2.0
DDIM inference steps T	25	25
Optimization steps J	100	40
Learning rate η	1×10^{-3}	2×10^{-2}
Noise trials N_z	20	20
Weight samples N_λ per trial	5	5
Total candidates B per sample	100	100
Optimizable conditioning dims d_o	20 (LILAC variables)	10 (energy metrics)
Optimizer	Adam	Adam
Barrier coefficient μ (DC workload loss)	—	100
Validity thresholds		
Minimum gain δ_{gain}	0.1%	0.1%
Spectral tolerance τ_{fft}	0.2	—
Correlation threshold τ_{corr}	0.8	—
Workload deviation tolerance	—	1%

A.6 Baselines Implementations and Hyperparameters

For the **Design-Bench** tasks, baseline implementations and hyperparameters are sourced from the official `design-baselines` repository[2], the GTG authors' implementation[3], and the DDOM authors' implementation[4]. All baselines use the hyperparameters from their respective repositories.

For the **real-world tasks**, we implement all baselines that require a surrogate model using the same frozen S_ϕ to ensure a fair comparison. For approaches that require training a neural network or diffusion model for test-time optimization, we use the same hidden dimension as our proposer when applicable. For gradient-based baselines, we perform grid search over learning rates $\{5 \times 10^{-4}, 1 \times 10^{-3}, 5 \times 10^{-3}, 1 \times 10^{-2}, 5 \times 10^{-2}\}$ and select the best-performing value for each method and task. For the evolutionary baseline CMA-ES, we use the `cma` Python package with default hyperparameters except for population size. For GP-qEI, we use a Gaussian Process surrogate with a Matérn kernel ($\nu = 2.5$).

For the diffusion-based **Brute Force** baseline on the real-world tasks, we do not perform a single one-shot DDIM sample. Instead, we reuse the same frozen proposer as D2C but replace gradient-based conditioning updates with an explicit sweep over hand-chosen conditioning directions. On

[2] `https://github.com/brandontrabucco/design-baselines`

[3] `https://github.com/dbsxodud-11/GTG`

[4] `https://github.com/siddarthk97/ddom`

ICF, the baseline linearly increases the yield-related conditioning coordinate and runs one full DDIM rollout at each sweep point, retaining the best candidate that passes the same gain, area, FFT, and correlation checks used for D2C. On DC, it linearly decreases selected energy-metric conditioning coordinates and runs repeated DDIM rollouts for multiple noise seeds at each sweep point, followed by the same workload-conservation and inlet-temperature checks used for D2C. Thus, Brute Force should be interpreted as a forward-only conditioning sweep with many repeated DDIM generations, not as a single batched generation pass.

For the surrogate-optimized real-world baselines, we do not optimize the primary objective alone. To reduce the chance that differences in constraint-satisfying improvement rate are an artifact of objective-only optimization, Gradient Ascent, REINFORCE, and CMA-ES are augmented with domain-aligned penalties or projections that mirror the main constraint families used by D2C, although these baselines operate directly in design space or in low-dimensional latent parameterizations rather than through the conditioning optimization of Eq. 13. On ICF, Gradient Ascent maximizes surrogate-predicted yield while adding similarity and first/second-order smoothness regularization, REINFORCE uses reward terms for yield together with area and spectral penalties, and CMA-ES searches in a PCA-reduced pulse space with an objective that combines surrogate-predicted yield with similarity, area, and squared first-difference penalties, together with a hard projection that prevents pulse area from exceeding the reference; all three are then validated using the same gain, area, FFT, and correlation thresholds as D2C. On DC, Gradient Ascent minimizes surrogate-predicted carbon emissions with area-conservation and similarity penalties, REINFORCE uses carbon-reduction reward together with an area-conservation penalty, and CMA-ES searches in a PCA-reduced profile space with carbon-reduction objective plus area-conservation, similarity, and squared first-difference penalties; all three are then filtered by the same workload-conservation and inlet-temperature constraints used for D2C. Thus, the lower constraint-satisfying improvement rates of these baselines in Table 14 should not be interpreted as evidence that they ignore constraint-aware optimization entirely.

For all real-world baselines, hyperparameters were selected on the validation split only, and the test split was used exclusively for final reporting.

A.7 Surrogate Evaluator Architecture

The frozen evaluator S_ϕ is a feedforward residual network with LayerNorm and dropout, trained independently via supervised regression. For ICF, the surrogate has 10 hidden layers of width 1024 with residual connections every two layers, predicting 20 LILAC performance variables from a 263-dimensional input (256-d pulse + 7 target parameters). For DC, the surrogate has 6 hidden layers of width 1024 with residual connections every two layers, predicting 10 energy metrics from a 217-dimensional input (144-d workload profile + 73 environmental conditions). Both surrogates use ReLU activations, LayerNorm, 10% dropout during training, and a sigmoid output layer mapping to $[0, 1]$ (targets are MinMax-scaled).

Table 5 summarizes the training hyperparameters for the surrogate evaluator.

Table 5: Training hyperparameters for the surrogate evaluator S_ϕ.

Hyperparameter	ICF	DC
Hidden dimension	1024	1024
Hidden layers	10	6
Residual connections	Every 2 layers	Every 2 layers
Activation	ReLU	ReLU
Dropout (training)	0.1	0.1
Output activation	Sigmoid	Sigmoid
Optimizer	AdamW	AdamW
Learning rate	1×10^{-3}	1×10^{-3}
Weight decay	1×10^{-5}	1×10^{-5}
LR schedule	Cosine warm restarts	Cosine warm restarts
Batch size	1024	1024
Early stopping patience	5 (checked every 100 epochs)	10 (checked every 100 epochs)
Gradient clipping	1.0	1.0
Input dimension	263 (256 + 7)	217 (144 + 73)
Output dimension	20	10

B Data and Evaluation

B.1 ICF

The training dataset comprises $\sim$2,600,000 radiation-hydrodynamics simulations from the LILAC simulation database. Each sample consists of a 256-dimensional laser pulse profile and 27 conditioning dimensions (7 target parameters + 20 LILAC performance variables). The data are split into 80% training, 10% validation, and 10% test ($\sim$260,000 samples each). The validation set was used for hyperparameter tuning; the test set is used exclusively for final evaluation. We randomly sample 2,000 pulses from the test set and use the same 2,000 for all methods. ICF Top selects the 1,000 highest-yield samples from the test set. The surrogate S_ϕ achieves $< 1\%$ WMAPE on all 20 predicted LILAC variables.

B.1.1 ICF Data Generation

The ICF dataset is generated from radiation-hydrodynamics simulations (LILAC code), where each simulation is specified by a target design and a laser pulse shape. LILAC is a prominent one-dimensional Lagrangian hydrodynamics ICF code [12, 32, 6, 20, 10, 11]. The laser pulse shape is prescribed via an internal library that parameterizes it into descriptive features such as inter-pulse time delays between Gaussian-shaped components, maximum power, and total energy. The full input specification thus lives in a 27-dimensional parameter space consisting of target specifications and a parameterized laser-pulse description.

Although this 27-D space permits uniform sampling subject to physical constraints (e.g., maximum laser energy, allowable target thicknesses), such sampling does not reflect how experiments or simulations are actually designed. In practice, high-performing implosions occupy only a small region of this high-dimensional space, guided by physics intent rather than being distributed uniformly across the full admissible hypervolume. The region of desirable inputs therefore lies on a lower-dimensional manifold embedded within the 27-D space, a manifold that is not explicitly known or parameterized but emerges from complex physics couplings between target geometry and laser-pulse features.

To sample this manifold more effectively, we supplement random draws with perturbations of known high-performing implosion designs. Perturbing their laser-pulse features and target parameters produces a family of nearby points that remain close to the underlying physics manifold while still providing variation in implosion performance. This strategy captures the empirical observation that, for a fixed target specification, laser-pulse shapes close to physics-informed designs typically outperform large random perturbations. The resulting dataset therefore reflects a distribution with multiple high-performance clusters concentrated on an unknown manifold within the 27-D input space, rather than a uniform distribution over the full hypervolume.

Note: The fusion dataset and simulation code fall under US export-control regulations. Certain implementation details and the data cannot be publicly released. Interested parties may contact the authors; data may be shared upon reasonable request with parties who have the appropriate permissions and legal authorization to access this class of controlled data.

B.2 ICF Oracle Evaluator

At test time, optimized ICF pulse designs are scored by a separate high-accuracy LILAC-validated evaluation surrogate that is distinct from the frozen optimization-time surrogate. This evaluation oracle is trained on the full available ICF dataset and is used only for final offline scoring, not for optimization. It should therefore be interpreted as a best-available simulator proxy trained on all available offline data for final comparison, rather than as a held-out model used within the optimization loop. Because this evaluation oracle is distinct from the frozen optimization-time surrogate, the final reported scores are not obtained by reusing the same model that provides gradients during optimization.

The oracle achieves $<$1% weighted mean absolute percentage error (WMAPE) on all 20 LILAC performance variables when compared against LILAC radiation-hydrodynamics simulation outputs. This level of accuracy indicates that the reported yield gains are a close proxy for relative trends predicted by the underlying physics code.

Why not evaluate with LILAC directly? Direct online evaluation with the LILAC simulator is infeasible at the scale required in this study because simulator access requires advanced scheduling, computational allocation, and substantial preparation. We therefore use a separate high-accuracy evaluation surrogate for final offline scoring. This introduces a proxy-evaluation limitation, but the same evaluation oracle is used for D2C and all baselines, making the large-scale comparison internally fair under a common scoring rule. Because the oracle has sub-1% error relative to LILAC simulation outputs, it provides a practical and informative approximation for large-scale comparison. This separation between optimization-time surrogate and evaluation-time oracle does not eliminate proxy-evaluation bias, but it reduces the risk that reported gains are merely artifacts of the optimization surrogate itself. In addition to this large-scale oracle-based evaluation, we performed a separate small-scale LILAC validation on 20 high-yield expert-designed pulses. Here, expert-designed refers to pulses manually designed by ICF experts based on expert knowledge. Using the same D2C conditioning-optimization approach, we observed an average yield gain of 9.8% relative to the original expert pulse baselines. Because this direct-simulator check is small-scale, we treat it as supplementary corroboration rather than the primary quantitative basis for the large-scale comparison.

B.3 DC

The training dataset comprises $200{,}000$ data center operating scenarios generated from the SustainDC physics-based thermal and energy simulator [24]. The simulator computes rack-level IT power, HVAC power, cooling tower fan power, compressor load, and thermal state from workload profiles and carbon footprints. We seed the generation with real workload traces from the Alibaba cluster dataset [1] and region-specific carbon intensity and weather profiles for Arizona (AZ), California (CA), and New York (NY), both provided within the SustainDC framework. We augment with synthetic variations of workload patterns, carbon footprint schedules, and CRAC set-point profiles to ensure broad distributional coverage. The three geographic regions provide diverse grid carbon intensities and climate conditions, enabling the model to generalize across different operational regimes. Each sample consists of a 144-dimensional workload power profile (24h at 10-min resolution) and 83 conditioning dimensions (73 environment state variables + 10 energy metrics). The data are split into 80% training, 10% validation, and 10% test ($\sim$20,000 samples each). Following the same protocol as ICF, all hyperparameter selection is performed on the validation set; the test set is reserved for final reporting. We randomly sample 2,000 scenarios from the test set and use the same 2,000 for all methods. Final designs are validated against the full physics-based model (not the surrogate). We use the public SustainDC codebase, whose repository documentation states that the project is primarily released under the MIT License, with some derived portions attributed to CarbonExplorer under CC BY-NC 4.0; we use and cite that code under the stated repository terms.[5]

B.4 Design-Bench

We use the standard offline-subset protocol of Design-Bench [31]: each method trains on the bottom p-th percentile of the full dataset by y-value (50% for TF Bind 8/10, 40% for Ant/D'Kitty Morphology), and y-normalization uses full-dataset bounds so that the best offline design scores below 1. The public Design-Bench codebase used for these tasks is released under the MIT License, and we use and cite the benchmark under that license.[6] For Design-Bench, D2C uses a task-specific variant of the inference-time optimization procedure because the conditional input is only a single scalar target score. In this setting, directly optimizing the conditioning provides only limited control. We therefore fix the conditioning to a high normalized target score (c=1.0) and instead optimize the initial latent/noise vector through the deterministic DDIM chain using gradients from the frozen surrogate. Empirically, this latent-optimization variant performed better on Design-Bench than direct optimization of the one-dimensional conditioning input. In contrast, for ICF and DC we retain direct conditioning optimization because those domains expose multiple optimizable conditioning coordinates; on those tasks, conditioning optimization outperforms latent-noise optimization (Section D.7).

Table 6 summarizes the inference-time Design-Bench hyperparameters used for our method.

For discrete tasks (TF Bind 8/10), we use the native exact Design-Bench oracle (`*-Exact-v0` variants). These two columns in Table 2 are therefore exact-oracle benchmark results, and D2C attains the best result on both of them. For these tasks we do not use a discrete diffusion process or

[5] `https://github.com/HewlettPackard/dc-rl`

[6] `https://github.com/brandontrabucco/design-bench`

Table 6: D2C inference-time hyperparameters for Design-Bench. These settings are shared across TF Bind 8, TF Bind 10, Ant Morphology, and D'Kitty Morphology unless noted otherwise.

Hyperparameter	Value
Optimized variable	Initial latent/noise vector z (same dimension as x)
Fixed conditioning target	$c = 1.0$
Optimizer	Adam
Learning rate	1×10^{-2}
Optimization steps	100
DDIM inference steps	50
Guidance scale w	3.0
Latent/noise trials	100
Returned candidate count	100
Gradient clipping	1.0
LR schedule	Cosine annealing over optimization steps
Latent initialization	$z \sim 0.8\,\mathcal{N}(0, \mathbf{I})$
Task-specific note	TF Bind 10 uses `max_samples=50000`; others use the full offline subset

token embeddings: each sequence is represented as its original length-8 or length-10 integer token vector and then MinMax-scaled coordinatewise to $[0, 1]$, so the four nucleotide categories become the four scalar levels $\{0, 1/3, 2/3, 1\}$. The Gaussian forward process, DDIM reverse process, and gradient-based conditioning optimization are all performed in this relaxed continuous space. After generation, each coordinate is projected to the nearest valid discrete level before oracle evaluation; the normalized design is then inverse-transformed and rounded back to integer tokens for exact lookup. Thus gradients are taken with respect to the relaxed continuous representation, and no gradient is propagated through the final discrete projection step. For continuous tasks (Ant Morphology and D'Kitty Morphology), we use a reproducible evaluation protocol grounded in the released benchmark data rather than relying on the version-sensitive MuJoCo/Gym oracle path. This choice is motivated by two observations: (i) the current Design-Bench continuous-task oracle path has documented reliability issues in the public repository,[7,8] and (ii) published continuous-task score ranges already vary substantially across prior Design-Bench papers such as DDOM and GTG [18, 39], indicating limited cross-paper comparability under differing software environments. To obtain a stable within-paper comparison, we therefore evaluate all methods on these two tasks using the same k-nearest-neighbor (k=5, distance-weighted) regression oracle fitted on the full released dataset. This protocol is grounded entirely in the benchmark data and ensures that all reported Ant and D'Kitty results are directly comparable under a single reproducible evaluation rule. The added row $\mathcal{D}$ (best) in Table 2 reports the best design already contained in the offline dataset for each task. Under this protocol, scores above $\mathcal{D}$ (best) indicate that a method is assigned a value higher than the best offline example for that task. Accordingly, these continuous-task results should be interpreted as a reproducible within-paper comparison under a single stable evaluation rule, rather than as a claim of direct numerical comparability to prior MuJoCo/Gym-based reports.

B.5 Data Normalization

All design variables x and conditioning vectors c are MinMax-scaled to $[0, 1]$ using statistics computed on the training set. The same scalers are applied at inference time, and generated designs are inverse-transformed to physical units for evaluation.

C Domain-Specific Constraints

After conditioning optimization (Section 3.6), each candidate design is validated against a set of domain-specific hard physical constraints before selection. These constraints are not a property of the D2C framework itself but reflect the safety-critical requirements of each application domain. Only candidates that pass all constraints are eligible for selection by primary objective; this ensures that reported improvements correspond to physically realizable and operationally safe designs.

[7] `https://github.com/brandontrabucco/design-bench/issues/11`
[8] `https://github.com/brandontrabucco/design-bench/issues/23`

ICF laser pulse constraints. A candidate pulse $\hat{x}_0$ is deemed *valid* if it satisfies all four of the following conditions simultaneously:

1. **Yield gain.** The surrogate-predicted neutron yield must exceed the baseline yield by at least a minimum threshold: $(\hat{Y} - Y_{\text{ref}})/Y_{\text{ref}} \geq \delta_{\text{gain}}$, with $\delta_{\text{gain}} = 0.1\%$.
2. **Energy conservation.** The integral of the generated pulse must not exceed that of the reference pulse (with a small tolerance $\epsilon = 0.01$): $\int \hat{x}_0(t)\,dt \leq \int x_{\text{ref}}(t)\,dt + \epsilon$. This ensures the optimized pulse does not deliver more laser energy than the original design, which would violate facility power budgets.
3. **Spectral fidelity.** The mean absolute difference between the Fourier amplitude spectra of the generated and reference pulses must remain below a threshold: $\frac{1}{N}\sum_k \big||\hat{X}_k| - |X_k^{\text{ref}}|\big| < \tau_{\text{fft}}$, with $\tau_{\text{fft}} = 0.2$. This prevents the optimizer from introducing high-frequency oscillations that are not physically realizable by the laser system.
4. **Temporal correlation.** The Pearson correlation between the generated and reference pulse waveforms must exceed a minimum: $\rho(\hat{x}_0, x_{\text{ref}}) > \tau_{\text{corr}}$, with $\tau_{\text{corr}} = 0.8$. This ensures structural similarity to the original pulse shape, preventing the optimizer from producing qualitatively different (and potentially unsafe) drive profiles.

Data center workload constraints. A candidate workload profile $\hat{x}_0$ is deemed *valid* if it satisfies all three conditions:

1. **Carbon reduction.** The operational carbon emissions under the optimized profile must be lower than the baseline by at least $\delta_{\text{red}} = 0.1\%$: $(E_{\text{ref}}^{\text{CO}_2} - \hat{E}^{\text{CO}_2})/E_{\text{ref}}^{\text{CO}_2} \geq \delta_{\text{red}}$.
2. **Workload conservation.** The total compute delivered (integral of the workload power profile) must not fall more than 1% below the reference: $(\int \hat{x}_0(t)\,dt - \int x_{\text{ref}}(t)\,dt)/\int x_{\text{ref}}(t)\,dt \geq -1\%$. Under-delivery violates service-level agreements. We make this final hard check asymmetric because under-delivery is the operational failure mode tied to service-level agreements; over-delivery is still discouraged upstream by the inference-time near-equality penalty.
3. **Inlet temperature safety.** The predicted server inlet temperature must remain below a safety limit: $T_{\text{inlet}} \leq T_{\max}$, with $T_{\max} = 50\,^{\circ}\text{C}$. Exceeding this threshold risks hardware damage and triggers emergency shutdowns in real facilities.

In both domains, if no candidate satisfies all constraints, the baseline design is retained unchanged, contributing 0% improvement to the reported metrics (Algorithm 1).

D Ablation Studies

We ablate guidance scale (Section D.1) and its selection without oracle access (Section D.2), optimization steps (Section D.3), learning rate (Section D.4), DDIM steps (Section D.5), optimizable conditioning dimensionality (Section D.6), conditioning versus latent-noise optimization (Section D.7), constraint-satisfying improvement rate (Section D.8), candidate budget (Section D.9), and computational cost (Section D.10).

D.1 Effect of the Guidance Scale

The CFG guidance scale w controls the strength of conditioning adherence during the DDIM reverse chain (Eq. 12). While higher w sharpens the conditional distribution and amplifies $\partial\tilde{\epsilon}_\theta/\partial c$, excessively large values push generated designs away from the empirical data distribution captured by the learned prior, degrading both quality and optimizability. We sweep $w \in \{0, 0.5, 1, 2, 3, 5, 7\}$ on both DC and ICF Top, holding all other hyperparameters fixed.

Table 7: Ablation on the guidance scale w. Values are mean percentage improvement (%) over the baseline design $\pm$ std over 5 runs.

Task	w=0	w=0.5	w=1	w=2	w=3	w=5	w=7
DC	$2.30\% \pm 0.45$	$6.56\% \pm 0.72$	$7.81\% \pm 0.74$	$8.98\% \pm 0.75$	$8.90\% \pm 0.73$	$8.79\% \pm 0.71$	$8.58\% \pm 0.69$
ICF Top	$0.00\% \pm 0.00$	$5.12\% \pm 0.67$	$8.17\% \pm 0.81$	$10.06\% \pm 0.88$	$3.85\% \pm 0.54$	$1.20\% \pm 0.29$	$0.00\% \pm 0.00$

Table 7 reveals a clear optimum at $w = 2$ for both tasks. Without guidance ($w = 0$), the model ignores conditioning entirely and optimization has no steering signal, producing little (2.30% on DC) or no (0.00% on ICF Top) improvement. Moderate guidance ($w = 1$–2) provides strong conditioning gradients while keeping designs closer to the data distribution learned by the prior, yielding peak performance. The two tasks exhibit markedly different sensitivity to over-guidance. DC degrades gracefully: performance drops only 0.40 percentage points from $w = 2$ to $w = 7$, reflecting the relatively smooth DC loss landscape. ICF Top collapses sharply beyond $w = 2$, falling to 0.00% at $w = 7$. This is consistent with the ICF domain's tight physical constraints: over-guidance drives the generated pulses away from the feasible region represented in the offline data, causing validity-first selection to reject most candidates. These results validate the dual role of w described in Section 3.4: it modulates not only generation fidelity but also optimization stability, and must be tuned per domain. Note that the 0.00% entries do not indicate numerical divergence of the optimizer: they mean that no candidate passed the hard constraints of Appendix C, so validity-first selection returned the baseline design rather than an infeasible pulse.

D.2 Selecting the Guidance Scale Without Oracle Access

Table 7 is computed with the evaluation oracle, which is unavailable when D2C is deployed on a new problem. We therefore examine whether w can be selected using only quantities available at optimization time, namely the frozen surrogate S_ϕ and the geometric validity checks of Appendix C. On 100 held-out ICF Top samples, we run D2C for $w \in \{1, 2, 3, 4\}$ with all other hyperparameters fixed and, for each sample, select a single candidate under two surrogate-only rules:

- **Full selection rule.** The selection rule used throughout the paper: candidates must satisfy the energy-conservation, spectral-fidelity (τ_{fft}=0.2), and temporal-correlation (τ_{corr}=0.8) checks together with the surrogate-predicted gain threshold δ_{gain}=0.1%, and the remaining candidate with the highest surrogate-predicted yield gain is chosen.
- **Geometric-only rule.** A more conservative variant that drops the surrogate-gain threshold and keeps only the three geometric checks, again choosing the candidate with the highest surrogate-predicted yield gain.

Table 8 reports the mean surrogate-predicted yield gain of the selected candidates.

Table 8: Surrogate-only selection of the guidance scale w on 100 held-out ICF Top samples. Values are the mean surrogate-predicted yield gain (%) of the candidate selected per sample; no oracle evaluation is used. **Bold**: best per row.

Selection rule	w=1	w=2	w=3	w=4
Full selection rule	8.27%	**11.20%**	4.14%	3.59%
Geometric-only rule	6.14%	**9.44%**	4.01%	3.15%

Both surrogate-only signals peak at w=2, the same value identified with oracle access in Table 7, and the two rules produce identical rankings over w. Because these values are surrogate predictions on a smaller sample rather than oracle evaluations, their magnitudes are not directly comparable to Table 7; what matters for deployment is that the ranking over w is preserved. Thus, although w must be tuned per domain, this tuning can be carried out offline with the frozen surrogate and geometric checks alone, without access to the true objective.

D.3 Effect of the Number of Optimization Steps

We vary the number of conditioning optimization steps $J \in \{10, 20, 30, 40, 50, 75, 100\}$ with all other hyperparameters fixed at their default values ($w = 2$).

Table 9: Ablation on the number of optimization steps J. Values are mean percentage improvement (%) over the baseline design $\pm$ std over 5 runs.

Task	J=10	J=20	J=30	J=40	J=50	J=75	J=100
DC	$7.90\% \pm 0.71$	$8.66\% \pm 0.72$	$8.92\% \pm 0.69$	$8.98\% \pm 0.75$	$8.97\% \pm 0.81$	$8.97\% \pm 0.73$	$8.94\% \pm 0.73$
ICF Top	$0.56\% \pm 0.18$	$1.64\% \pm 0.31$	$3.46\% \pm 0.49$	$5.38\% \pm 0.62$	$6.82\% \pm 0.74$	$8.94\% \pm 0.85$	$10.06\% \pm 0.88$

The two tasks exhibit contrasting convergence profiles (Table 9). DC saturates rapidly: $J = 10$ already captures 88% of the peak improvement, and performance plateaus beyond $J = 40$. This reflects the relatively simple loss landscape of the workload scheduling problem, where a few gradient steps suffice to find a good conditioning point. ICF Top improves monotonically with more steps, reaching its best result at $J = 100$. The ICF loss landscape is substantially more complex: the interplay of yield maximization, energy conservation, spectral stability, and smoothness constraints creates narrow feasible corridors that require more iterations to navigate. In practice, $J = 40$ offers a good efficiency–quality trade-off for DC, while ICF benefits from the full $J = 100$ budget.

D.4 Effect of the Optimization Learning Rate

The learning rate η for the conditioning optimizer controls the step size during gradient-based optimization of the conditioning vector. Too large a learning rate overshoots narrow feasible regions; too small a rate under-explores within the allocated budget J. We sweep $\eta \in \{10^{-1}, 5 \times 10^{-2}, 2 \times 10^{-2}, 10^{-2}, 10^{-3}\}$ on DC and ICF Top with all other hyperparameters fixed.

Table 10: Ablation on the optimization learning rate η. Values are mean percentage improvement (%) over the baseline design $\pm$ std over 5 runs.

Task	η=10^{-1}	η=5×10^{-2}	η=2×10^{-2}	η=10^{-2}	η=10^{-3}
DC	$8.22\% \pm 0.72$	$8.85\% \pm 0.74$	$8.98\% \pm 0.75$	$8.72\% \pm 0.74$	$7.66\% \pm 0.70$
ICF Top	$2.34\% \pm 0.42$	$5.33\% \pm 0.65$	$9.07\% \pm 0.85$	$9.05\% \pm 0.85$	$10.06\% \pm 0.88$

The two tasks exhibit markedly different optimal learning rates (Table 10). DC peaks at $\eta = 2 \times 10^{-2}$: its relatively smooth loss landscape permits aggressive steps that quickly reach the optimum, while smaller rates under-explore within the $J = 40$ budget. ICF Top peaks at $\eta = 10^{-3}$: the complex, tightly constrained ICF landscape requires cautious steps to remain within narrow feasible corridors. Large rates ($\eta = 10^{-1}$) are catastrophic for ICF (2.34%) but only mildly harmful for DC (8.22%), confirming that ICF optimization is far more sensitive to step size.

D.5 Effect of the Number of DDIM Inference Steps

The number of DDIM reverse steps T controls the granularity of the denoising trajectory and, consequently, the quality of the differentiable path through which gradients flow. Fewer steps reduce the computational cost per optimization iteration but make the discrete denoising trajectory coarser, potentially degrading both generation quality and gradient fidelity. We sweep $T \in \{10, 25, 50\}$ on DC and ICF Top with all other hyperparameters fixed.

Table 11: Ablation on the number of DDIM inference steps T. Values are mean percentage improvement (%) over the baseline design $\pm$ std over 5 runs.

Task	T=10	T=25	T=50
DC	$8.84\% \pm 0.73$	$8.98\% \pm 0.75$	$8.97\% \pm 0.74$
ICF Top	$9.90\% \pm 0.86$	$10.06\% \pm 0.88$	$10.05\% \pm 0.87$

Performance is remarkably insensitive to T (Table 11). Even with only $T = 10$ steps, both tasks retain over 95% of the peak improvement, and the optimum is reached at $T = 25$ for both. Increasing to $T = 50$ yields no further gain, indicating that the gradient signal through the DDIM chain is already well-conditioned at moderate step counts. Practically, fewer inference steps translate directly to lower wall-clock time per optimization iteration, making $T = 25$ the default choice.

D.6 Effect of the Number of Optimizable Conditioning Inputs

A key design choice in D2C is which conditioning dimensions to optimize. Optimizing all dimensions of c_{opt} maximizes expressivity but increases the search space; restricting to fewer inputs reduces degrees of freedom but may miss beneficial trade-offs. We vary the number of optimizable conditioning inputs d_o on both DC and ICF Top while keeping all other hyperparameters fixed. For DC, the full conditioning includes $d_o = 10$ optimizable energy metrics; we test subsets of size 1, 2, 4,

and 6. For ICF, the full conditioning includes $d_o = 20$ optimizable LILAC variables; we test subsets of size 1, 4, 9, and 15. These subsets were not randomly re-sampled across trials. Instead, for each task we used one fixed, nested family of task-specific subsets chosen once and then held fixed across all seeds, progressively expanding the optimizable block from the most central target dimensions outward (for ICF beginning with the yield-related LILAC targets, and for DC beginning with the primary energy-metric targets used in the implementation). Thus Table 12 should be interpreted as a controlled dimensionality ablation under a fixed variable ordering, rather than as an average over all subsets of a given size. In both cases, "All" denotes the full set used in the main results.

Table 12: Ablation on the number of optimizable conditioning inputs d_o. Values are mean percentage improvement (%) over the baseline design $\pm$ std over 5 runs.

Task	All	Number of optimizable inputs			
DC	$8.98\% \pm 0.75$	$5.14\% \pm 0.61$ (d_o=1)	$5.67\% \pm 0.64$ (d_o=2)	$5.89\% \pm 0.65$ (d_o=4)	$7.97\% \pm 0.72$ (d_o=6)
ICF	$10.06\% \pm 0.88$	$4.02\% \pm 0.56$ (d_o=1)	$7.97\% \pm 0.79$ (d_o=4)	$8.74\% \pm 0.83$ (d_o=9)	$8.97\% \pm 0.85$ (d_o=15)

Using the full set of optimizable inputs consistently yields the best results (Table 12). Restricting to a single input ($d_o = 1$) cuts performance roughly in half on both tasks, confirming that D2C leverages multi-dimensional conditioning to find better trade-offs. Both tasks improve monotonically with d_o: on ICF, performance rises steadily from 4.02% to 8.97% as more target parameters become available, while DC follows the same trend from 5.14% to 7.97%. The full conditioning set ("All") achieves the best result on both tasks, justifying the default of optimizing all available inputs.

D.7 Conditioning Optimization versus Latent-Noise Optimization

Since the deterministic DDIM chain $G_\theta(c, x_T)$ is differentiable in both of its inputs, the same gradient-based procedure can be applied either to the optimizable conditioning c_{opt} (the default D2C setting on the real-world tasks) or to the latent noise x_T (the variant used on Design-Bench, Appendix B.4). To isolate the effect of this choice, we compare the two variants on ICF Top and DC while holding the diffusion proposer, surrogate, loss, candidate budget, validity-first selection, and evaluation protocol fixed, and changing only the optimized variable. In the latent variant, the conditioning is fixed to its initial value and the gradient updates of Eq. (14) are applied to x_T instead of c_{opt}.

Table 13: Conditioning optimization versus latent-noise optimization on the real-world tasks. Values are mean percentage improvement (%) over the baseline design $\pm$ std over 5 runs. **Bold**: best.

Optimized variable	ICF Top	DC
Latent noise x_T	$7.56\% \pm 0.76$	$3.62\% \pm 0.26$
Conditioning c_{opt} (D2C)	$\mathbf{10.06\% \pm 0.88}$	$\mathbf{8.98\% \pm 0.75}$

Conditioning optimization outperforms latent-noise optimization on both tasks (Table 13), by 2.50 percentage points on ICF Top and 5.36 percentage points on DC. This supports treating the conditioning signal, rather than the initial noise, as the optimizable variable whenever the conditioning exposes several optimizable coordinates (d_o=20 for ICF and d_o=10 for DC). In this regime, c_{opt} acts on the generated design through the semantically meaningful performance targets that the proposer was trained to follow, so updates move designs along directions supported by the learned conditional distribution; perturbing x_T directly offers more raw degrees of freedom but is less aligned with the objectives and more prone to drifting toward regions where candidates fail the hard constraints. Together with Table 12, where restricting ICF to d_o=1 reduces performance to 4.02%, these results also explain the Design-Bench setting: when the conditioning is a single scalar score, it provides too few handles and latent-noise optimization becomes the stronger choice.

D.8 Constraint-Satisfying Improvement Rate

Because final selection is validity-first (Section 3.6), overall performance depends not only on objective improvement but also on how often a method produces a successful test sample. We therefore report the mean constraint-satisfying improvement rate over 5 seeds: the percentage of evaluation samples for which the method produces a final design with nonzero gain, i.e., for which

at least one valid improving candidate is found and selected rather than reverting to the baseline. Higher constraint-satisfying improvement rates indicate that a method succeeds more often on the 2,000-sample ICF Standard/DC evaluations and the 1,000-sample ICF Top evaluation.

Table 14: Constraint-satisfying improvement rate (%) on the real-world tasks. Values are mean constraint-satisfying improvement rate ± std over 5 runs, where the metric is the percentage of evaluation samples for which the final selected design achieves nonzero gain relative to the baseline (equivalently, at least one valid improving candidate is found). Higher is better.

Method	ICF Standard	ICF Top	DC
BDI	27.40 ± 1.28	18.24 ± 1.11	6.00 ± 0.24
Brute Force	32.97 ± 1.36	31.26 ± 1.41	64.50 ± 2.58
CbAS	31.84 ± 1.17	18.52 ± 0.83	39.23 ± 1.61
D2C (Ours)	81.23 ± 2.86	75.12 ± 2.78	100.00 ± 0
CMA-ES	29.97 ± 1.19	19.59 ± 0.97	42.21 ± 1.59
COMs	25.82 ± 0.97	9.12 ± 0.24	40.13 ± 1.80
DDOM	42.57 ± 1.64	33.28 ± 1.28	79.10 ± 2.16
GP-qEI	26.89 ± 0.72	16.48 ± 0.42	100.00 ± 0
Gradient Ascent	31.64 ± 0.97	25.68 ± 0.78	57.50 ± 1.41
GTG	38.50 ± 1.03	21.20 ± 0.69	59.19 ± 1.92
MINs	27.84 ± 0.77	19.27 ± 0.57	45.29 ± 1.70
PGS	21.37 ± 0.71	20.14 ± 0.73	23.50 ± 0.67
REINFORCE	51.74 ± 1.44	19.74 ± 0.53	88.18 ± 1.87

Table 14 shows that D2C attains the highest constraint-satisfying improvement rate on both ICF settings and ties for the highest rate on DC. Importantly, the surrogate-based real-world baselines in this comparison are not optimized against the primary objective alone; their implementation details and constraint-aware penalty terms are summarized in Appendix A. The margin is especially pronounced on the ICF tasks: D2C succeeds on 81.23% of ICF Standard samples and 75.12% of ICF Top samples, far above the next-best methods. This indicates that optimizing conditioning through the differentiable DDIM chain improves not only the quality of the final solution when optimization succeeds, but also the probability of finding any valid improvement at all.

The gap is most informative on ICF Top, where all methods operate on already high-performing baseline designs and the feasible improvement region is especially narrow. Most baselines experience a sharp drop in constraint-satisfying improvement rate in this regime, whereas D2C retains a 75.12% success rate, showing that its advantage is not only better objective optimization but also a higher rate of finding valid improving candidates under strict constraints. On DC, constraint-satisfying improvement rates are generally higher across methods, consistent with the smoother operational landscape of the scheduling problem; D2C and GP-qEI both achieve 100%, while DDOM (79.10%) and REINFORCE (88.18%) also remain relatively robust. Overall, these results explain why validity-first selection benefits D2C so strongly: it not only finds stronger solutions, but also reverts to the baseline much less often.

D.9 Effect of Candidate Budget ($N_z \times N_\lambda$)

The total candidate budget $B = N_z \times N_\lambda$ determines how many parallel optimization trajectories are explored per test sample. N_z controls the number of independent noise realizations (exploring structurally different regions of the latent space), while N_λ controls the number of Dirichlet-sampled weight vectors per noise trial (exploring different objective trade-offs from the same starting point). We sweep seven configurations from B=3 to B=100 on ICF Top and DC, holding all other hyperparameters fixed.

Table 15: Ablation on the candidate budget $N_z \times N_\lambda$. Values are mean ± standard deviation over 5 seeds. Configurations are labeled by their $N_z \times N_\lambda$ values.

Task	1×3	5×3	10×3	20×1	20×2	20×3	20×5
ICF Top	$0.86\% \pm 0.24$	$4.59\% \pm 0.58$	$6.50\% \pm 0.72$	$5.72\% \pm 0.69$	$8.99\% \pm 0.85$	$9.64\% \pm 0.86$	$10.06\% \pm 0.88$
DC	$4.21\% \pm 0.57$	$7.82\% \pm 0.74$	$8.66\% \pm 0.69$	$8.41\% \pm 0.64$	$8.83\% \pm 0.72$	$8.91\% \pm 0.74$	$8.98\% \pm 0.75$

Table 15 reveals two complementary effects. First, increasing noise diversity (N_z) with fixed N_λ=3 produces large gains: ICF Top improves from 0.86% (N_z=1) to 9.64% (N_z=20), confirming that multiple latent-space starting points are essential for discovering high-quality modes in complex landscapes. Second, for a fixed budget of N_z=20 noise trials, increasing weight diversity from N_λ=1 (single loss weighting) to N_λ=3 (Dirichlet-sampled) yields substantial gains on ICF Top ($5.72\% \rightarrow 9.64\%$), validating the multi-objective coverage strategy. The comparison between 20×1 (no Dirichlet diversity) and 20×5 isolates the contribution of stochastic weight sampling: the $+76\%$ relative improvement on ICF Top ($5.72\% \rightarrow 10.06\%$) demonstrates that exploring different trade-offs from the same noise realization uncovers solutions that a single fixed weighting misses. On DC, the landscape is smoother and the method saturates earlier: N_z=10, N_λ=3 already captures 96% of the peak, and further budget increases yield diminishing returns. Overall, N_z=20, N_λ=3 (B=60 total candidates) offers the best efficiency–quality trade-off across both tasks, whereas N_z=20, N_λ=5 (B=100) yields the strongest final performance and is therefore used for the main reported results. Because the optimization trajectories are batched on the GPU, increasing the candidate budget is substantially more efficient than evaluating trajectories sequentially; however, larger budgets still increase memory usage and may increase latency depending on hardware and batch size.

D.10 Computational Cost

Table 16 reports the wall-clock time per sample (in seconds) for each method on ICF Top and DC.

Table 16: Computation time (seconds per sample) for each method. Lower is better.

Method	ICF Top	DC
BDI	2.5	2.9
Brute Force	23.2	35.7
CbAS	14.2	17.5
CMA-ES	35.6	43.7
COMs	18.3	32.9
DDOM	2.3	3.6
GP-qEI	48.6	56.7
Gradient Ascent	25.8	23.4
GTG	14.6	22.8
MINs	2.1	4.8
PGS	2.1	2.3
REINFORCE	3.7	4.6
D2C (Ours)	34.7	39.8

For this evaluation we use the N_z=20, N_λ=5 (B=100 total candidates) configuration (the configuration used for the main results). D2C is among the more expensive methods (34.7s / 39.8s), comparable to CMA-ES and Gradient Ascent, because each optimization step requires a full forward and backward pass through the T-step DDIM chain. The J=100 conditioning updates are sequential, but within each update all B=100 candidate trajectories are executed as a single batched DDIM forward/backward pass on the GPU rather than as 100 separate launches. Lightweight baselines such as MINs, PGS, and DDOM run an order of magnitude faster (2–5s) since they perform a single forward pass without iterative optimization. GP-qEI is the most expensive due to Gaussian process fitting. Importantly, D2C's cost scales linearly with J and T (Sections D.3 and D.5), and the ablations above show that reducing either has minimal impact on quality. For example, reducing T from 25 to 10 would cut wall-clock time by roughly 60% while retaining 98% of the improvement. The Brute Force baseline takes 23–36s despite lacking gradient computation because it is not a single batched forward pass. Instead, it repeatedly reruns the DDIM sampler across an explicit conditioning sweep (100 sweep points per sample in ICF, and 100 sweep points with multiple noise seeds per point in the DC implementation), evaluating each generated candidate afterward. Its runtime is therefore comparable to D2C because both methods spend most of their time repeatedly invoking the DDIM proposer; D2C adds backpropagation, whereas Brute Force spends that budget on many forward-only sweep evaluations.

E Domain-Specific Training Losses

We instantiate the general training objective (Eq. 8) for each application domain. In both cases the total loss follows the same four-term structure: denoising, reconstruction, physics-based fidelity, and surrogate consistency. The differences lie in the specific fidelity constraints and surrogate architecture.

E.1 ICF Laser Pulse Optimization

The design $x \in \mathbb{R}^{256}$ is a laser power pulse profile, conditioned on $c = (T_0, \ldots, T_6, L_1, \ldots, L_{20}) \in \mathbb{R}^{27}$, where T_i are target parameters and L_j are the 20 LILAC-simulated performance variables included in the conditioning vector.

Surrogate consistency (LILAC). The frozen LILAC surrogate S_ϕ^{ICF} takes as input the concatenation $[\hat{x}_0,\, c_{\mathrm{fix}}] \in \mathbb{R}^{256+7}$ (predicted pulse and the 7 target parameters) and predicts 20 LILAC performance variables $\hat{L} \in \mathbb{R}^{20}$. The loss is:

$$\mathcal{L}_{\mathrm{surr}}^{\mathrm{ICF}} = \|\hat{L} - L\|^2, \qquad \hat{L} = S_\phi^{\mathrm{ICF}}(\hat{x}_0,\, c_{\mathrm{fix}}), \tag{16}$$

where $L = (c_8, \ldots, c_{27}) \in \mathbb{R}^{20}$ are the ground-truth LILAC variables from the conditioning vector. This loss is applied with weight $\alpha_s = 1.0$.

Physics-based fidelity. Two one-sided penalties encourage physical realizability of the generated pulse:

$$\mathcal{L}_{\mathrm{fid}}^{\mathrm{ICF}} = \Big[\int \hat{x}_0(t) \cdot s_y \, dt - \int x(t) \cdot s_y \, dt\Big]_+ + \big[\max(\hat{x}_0 \cdot s_y) - V_{\max}\big]_+, \tag{17}$$

where s_y is the data scaler that maps normalized values back to physical units, $V_{\max}$ is the maximum observed pulse amplitude, and $[\cdot]_+ = \max(\cdot, 0)$. The first term prevents the generated pulse from delivering more energy than the reference; the second prevents physically impossible peak powers.

Total ICF training loss.

$$\mathcal{L}_{\mathrm{train}}^{\mathrm{ICF}} = \|\epsilon - \epsilon_\theta\|^2 + \alpha_r \, \|\hat{x}_0 - x\|^2 + \mathcal{L}_{\mathrm{surr}}^{\mathrm{ICF}} + \alpha_a \, \mathcal{L}_{\mathrm{fid}}^{\mathrm{ICF}} + \alpha_{\mathrm{sm}} \, \mathcal{L}_{\mathrm{smooth}}, \tag{18}$$

with weights $\alpha_r = 0.5$, $\alpha_a = 0.001$, $\alpha_{\mathrm{sm}} = 0.0001$.

E.2 Data Center Workload Scheduling

The design $x \in \mathbb{R}^{144}$ is a 24-hour workload power profile at 10-minute resolution, conditioned on $c \in \mathbb{R}^{83}$ comprising 73 environment state dimensions ($C_0, C_1, C_3 \in \mathbb{R}^{24}$ hourly profiles and C_2 scalar IT demand) and 10 energy performance metrics ($E_0, \ldots, E_9$).

Surrogate consistency. The frozen DC surrogate S_ϕ^{DC} takes the concatenation $[\hat{x}_0,\, c_{\mathrm{fix}}] \in \mathbb{R}^{144+73}$ (predicted profile and the 73 environment conditioning dimensions) and predicts 10 energy metrics $\hat{E} \in \mathbb{R}^{10}$:

$$\mathcal{L}_{\mathrm{surr}}^{\mathrm{DC}} = \|\hat{E} - E\|^2, \qquad \hat{E} = S_\phi^{\mathrm{DC}}(\hat{x}_0,\, c_{\mathrm{fix}}), \tag{19}$$

where $E = (c_{74}, \ldots, c_{83}) \in \mathbb{R}^{10}$ are the ground-truth metrics from the conditioning.

Physics-based fidelity. The fidelity term enforces workload integral conservation and amplitude bounds:

$$\mathcal{L}_{\mathrm{fid}}^{\mathrm{DC}} = \Big[\int \hat{x}_0(t) \cdot M \, dt - \int x(t) \cdot M \, dt\Big]_+ + \big[\max(\hat{x}_0) - 1\big]_+ + \big[-\min(\hat{x}_0)\big]_+, \tag{20}$$

where M is the data-range scaling factor. The first term prevents the scheduled workload from exceeding the reference demand; the second and third keep the normalized profile within $[0, 1]$. This training-stage relaxation is intentionally one-sided: the reconstruction term already anchors samples near observed workloads, so the fidelity term mainly suppresses unrealistic demand inflation while preserving coverage of the empirical profile manifold.

Total DC training loss.

$$\mathcal{L}_{\text{train}}^{\text{DC}} = \|\epsilon - \epsilon_\theta\|^2 + \alpha_r \|\hat{x}_0 - x\|^2 + \alpha_a \mathcal{L}_{\text{fid}}^{\text{DC}} + \alpha_{\text{sm}} \mathcal{L}_{\text{smooth}} + \alpha_s \mathcal{L}_{\text{surr}}^{\text{DC}}, \tag{21}$$

with default weights $\alpha_r = 0.5$, $\alpha_a = 0.01$, $\alpha_{\text{sm}} = 0.0001$, $\alpha_s = 1.0$.

Smoothness loss (shared). Both domains use the same squared first-difference penalty (discrete Dirichlet energy):

$$\mathcal{L}_{\text{smooth}}(\hat{x}_0) = \frac{1}{d-1} \sum_{i=1}^{d-1} (\hat{x}_{0,i+1} - \hat{x}_{0,i})^2. \tag{22}$$

F Domain-Specific Inference-Time Losses

We instantiate the general inference-time loss (Eq. 13) for each application domain. In both cases the loss is a weighted sum of a surrogate-based objective, physical constraint penalties, and a smoothness term. We do not apply an additional post-hoc normalization after forming these task-specific losses; instead, comparability is enforced by construction wherever possible. In particular, the objective terms are written as relative improvements over the baseline, conservation penalties are written as relative deviations when appropriate, smoothness and spectral penalties are averaged over dimensions, and all designs and conditioning variables are MinMax-scaled to $[0, 1]$ (Appendix B.5). The Dirichlet-sampled weights therefore act over task-specific losses that are already on broadly comparable scales rather than over raw quantities with incompatible units.

F.1 ICF Laser Pulse Optimization

The inference-time loss for ICF combines four terms with Dirichlet-sampled weights $\boldsymbol{\lambda} = (\lambda_y, \lambda_s, \lambda_e, \lambda_{\text{sm}})$:

$$\mathcal{L}^{\text{ICF}} = \lambda_y \mathcal{L}_{\text{yield}} + \lambda_s \mathcal{L}_{\text{stab}} + \lambda_e \mathcal{L}_{\text{energy}} + \lambda_{\text{sm}} \mathcal{L}_{\text{smooth}}. \tag{23}$$

Yield objective. The frozen LILAC surrogate predicts the log neutron yield from the generated pulse and laser parameters. The loss maximizes relative yield gain over the baseline:

$$\mathcal{L}_{\text{yield}} = -\frac{\exp\big(S_\phi(x_0, c_{\text{fix}})_0\big) - Y_{\text{base}}}{Y_{\text{base}} + \epsilon}, \tag{24}$$

where $S_\phi(\cdot)_0$ is the predicted log yield (inverse-scaled to physical units) and $Y_{\text{base}} = \exp(S_\phi(x_{\text{ref}}, c_{\text{fix}})_0)$.

Spectral stability. The optimized pulse must preserve the frequency content of the reference to ensure physical plausibility:

$$\mathcal{L}_{\text{stab}} = \frac{1}{N} \sum_{i=1}^{N} \big(|\mathcal{F}(x_0)_i| - |\mathcal{F}(x_{\text{ref}})_i|\big)^2, \tag{25}$$

where $\mathcal{F}$ denotes the discrete Fourier transform over N frequency bins.

Energy conservation. The pulse integral must not exceed the reference (one-sided penalty):

$$\mathcal{L}_{\text{energy}} = \Big[\int x_0(t) \cdot s_y \, dt - \int x_{\text{ref}}(t) \cdot s_y \, dt\Big]_+, \tag{26}$$

where s_y maps normalized values to physical units.

Smoothness. Squared first-difference penalty (discrete Dirichlet energy): $\mathcal{L}_{\text{smooth}} = \frac{1}{d-1} \sum_i (x_{0,i+1} - x_{0,i})^2$.

Validity criteria. The hard constraint thresholds used for candidate selection are detailed in Appendix C.

F.2 Data Center Workload Scheduling

The inference-time loss for DC combines three terms with Dirichlet-sampled weights $\boldsymbol{\lambda} = (\lambda_c, \lambda_p, \lambda_{\text{sm}})$:

$$\mathcal{L}^{\text{DC}} = \lambda_c \, \mathcal{L}_{\text{cost}} + \lambda_p \, \mathcal{L}_{\text{power}} + \lambda_{\text{sm}} \, \mathcal{L}_{\text{smooth}}. \tag{27}$$

Cost objective. The frozen DC surrogate predicts energy metrics from the generated profile and environmental conditioning. The loss maximizes relative reduction in the target metric (e.g., carbon footprint E_9 or total power E_0):

$$\mathcal{L}_{\text{cost}} = -\frac{f_{\text{base}} - S_\phi(x_0, c_{\text{fix}})_k}{f_{\text{base}} + \epsilon}, \tag{28}$$

where $S_\phi(\cdot)_k$ is the k-th predicted metric inverse-scaled to physical units and f_{base} is its baseline value.

Workload conservation. The total workload delivered must match the reference within a tight tolerance τ:

$$\mathcal{L}_{\text{power}} = d^2 + \mu \, [\, |d| - \tau \,]_+^2, \qquad d = \frac{\mathcal{I}(x_0) - \mathcal{I}(x_{\text{ref}})}{|\mathcal{I}(x_{\text{ref}})| + \epsilon}, \tag{29}$$

where $\mathcal{I}(x) = \int x(t) \cdot M \, dt$ is the scaled integral and μ is the barrier penalty coefficient ($\mu = 100$ in all DC experiments). At inference, in contrast, we optimize against an external carbon objective, so we use a near-equality penalty around the reference workload to prevent the optimizer from exploiting either over-delivery or under-delivery as an artificial route to lower emissions.

Smoothness. First- and second-order finite differences:

$$\mathcal{L}_{\text{smooth}} = \frac{1}{d-1} \sum_i (\Delta x_i)^2 + \frac{1}{2(d-2)} \sum_i (\Delta^2 x_i)^2, \tag{30}$$

where $\Delta x_i = x_{0,i+1} - x_{0,i}$ and $\Delta^2 x_i = x_{0,i+2} - 2x_{0,i+1} + x_{0,i}$.

Validity criteria. The hard constraint thresholds used for candidate selection are detailed in Appendix C.

G Dirichlet-Weighted Multi-Objective Sampling

The Dirichlet distribution $\text{Dir}(\boldsymbol{\alpha})$ is a continuous probability distribution over the $(K-1)$-dimensional simplex $\Delta^{K-1} = \{\boldsymbol{\lambda} \in \mathbb{R}^K : \lambda_k \geq 0, \ \sum_k \lambda_k = 1\}$. It is parameterized by a concentration vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)$ with $\alpha_k > 0$, and has density:

$$p(\boldsymbol{\lambda} \mid \boldsymbol{\alpha}) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_{k=1}^{K} \lambda_k^{\alpha_k - 1}, \tag{31}$$

where $\Gamma(\cdot)$ is the gamma function. Key properties relevant to our use:

- **Simplex constraint.** Every sample $\boldsymbol{\lambda} \sim \text{Dir}(\boldsymbol{\alpha})$ satisfies $\sum_k \lambda_k = 1$ and $\lambda_k \geq 0$ by construction, making it a natural distribution over loss-weight vectors.
- **Symmetric case ($\alpha_k = \alpha$ for all k).** When all concentration parameters are equal, the distribution is symmetric over all K objectives. With $\alpha = 1$ (our default), the density is *uniform* over the simplex: every possible trade-off among the K loss terms is equally likely.
- **Concentration control.** Setting $\alpha < 1$ concentrates samples near the vertices of the simplex (i.e., each draw emphasizes a single objective strongly), while $\alpha > 1$ concentrates samples near the center $\boldsymbol{\lambda} = (1/K, \ldots, 1/K)$ (i.e., more balanced trade-offs). The choice $\alpha = 1$ provides maximum diversity.

Intuition for D2C. In our framework, each of the N_λ weight vectors $\boldsymbol{\lambda}^{(m)} \sim \mathrm{Dir}(\mathbf{1}_K)$ defines a different scalarization of the multi-objective loss (Eq. 13). Some draws will heavily weight the surrogate yield objective (exploring aggressive performance gains); others will emphasize constraint penalties (producing conservative but highly feasible designs); yet others will balance smoothness against performance. By optimizing N_λ trajectories in parallel with different weight vectors, the candidate pool explores diverse scalarized trade-offs without requiring manual tuning of fixed weights. This uniformity is with respect to the weight simplex over these task-specific scalarizations; it is not a claim that Dirichlet weighting alone removes arbitrary objective-scale mismatches without the loss design described above. The validity-first selection (Stage 3 of Algorithm 1) then picks the best feasible candidate from this diverse pool, combining broad exploration with strict constraint satisfaction.

H Theoretical Analysis: Convergence and Stability

We summarize several local and conditional analytical properties of D2C. The results below establish an exact chain-rule decomposition for gradients through the DDIM reverse chain, a standard gradient-descent bound for a fixed smooth-region subproblem, almost-everywhere differentiability for smooth one-sided penalties, and a simple conditional ranking guarantee for the frozen surrogate objective.

H.1 Gradient Stability Through the DDIM Chain

A primary concern in differentiating through iterative denoising chains is the potential for vanishing or exploding gradients as the number of steps T grows. Rather than claiming a full global stability theorem, we first record the exact chain-rule decomposition that underlies gradient flow through the DDIM reverse process.

Proposition 1 (Conditioning-Jacobian Decomposition). *Let $G_\theta(c, x_T) = x_0$ denote the T-step DDIM reverse chain (Eq. 11) with frozen parameters θ, and assume each reverse step is differentiable in (x_t, c). Then:*

$$\frac{\partial x_0}{\partial c} = \sum_{t=1}^{T} \left(\prod_{s=1}^{t-1} J_s \right) \left. \frac{\partial x_{t-1}}{\partial c} \right|_{x_t}, \tag{32}$$

where $J_s = \frac{\partial x_{s-1}}{\partial x_s}$ is the per-step transition Jacobian, and the direct term $\left. \frac{\partial x_{t-1}}{\partial c} \right|_{x_t}$ captures the local effect of c at step t.

Proof. Each DDIM step (Eq. 11) computes $x_{t-1} = \phi(x_t, c, t)$ where ϕ depends on c through the CFG-guided noise $\tilde{\epsilon}_\theta$ (Eq. 12). Applying the chain rule recursively over the T reverse steps yields Eq. (32). □

Remark. Proposition 1 is only an exact chain-rule decomposition. A separate heuristic interpretation is that, under a clipped schedule and moderate step sizes, each DDIM transition is close to identity, so recent reverse steps may dominate the gradient more than distant ones. We do not promote that heuristic to a theorem here; we mention it only because it is consistent with the empirical finding in Section D.5 that moderate values of T already perform well.

H.2 Convergence of the Weighted-Sum Optimization

We now state a standard smooth-nonconvex gradient-descent bound for a single fixed scalarization inside one smooth region of the objective.

Proposition 2 (Gradient-Descent Bound Within a Fixed Smooth Region). *Let $\mathcal{L}(c) = \sum_{k=1}^{K} \lambda_k \mathcal{L}_k\big(G_\theta(c, x_T)\big)$ be the scalarized objective (Eq. 13) with fixed weights $\boldsymbol{\lambda} > 0$ and fixed latent x_T. Let $\mathcal{U} \subseteq \mathbb{R}^{d_c}$ be an open region on which $\mathcal{L}$ is β-smooth and bounded below by $\mathcal{L}^*$. If gradient descent with step size $\eta \leq 1/\beta$ is initialized in $\mathcal{U}$ and all iterates remain in $\mathcal{U}$, then:*

$$\min_{j=0,\ldots,J-1} \|\nabla_c \mathcal{L}(c^{(j)})\|^2 \;\leq\; \frac{2\big(\mathcal{L}(c^{(0)}) - \mathcal{L}^*\big)}{\eta J}. \tag{33}$$

This is the standard $\mathcal{O}(1/J)$ first-order bound for gradient descent on the restricted smooth-region objective.

Proof. This follows from the standard descent lemma for smooth nonconvex optimization [25]. By β-smoothness: $\mathcal{L}(c^{(j+1)}) \leq \mathcal{L}(c^{(j)}) - \eta(1 - \eta\beta/2)\|\nabla_c \mathcal{L}(c^{(j)})\|^2$. For $\eta \leq 1/\beta$, each step provides descent of at least $(\eta/2)\|\nabla_c \mathcal{L}\|^2$. Telescoping over J steps yields the bound. □

Remark. Proposition 2 is intentionally conditional and local. It applies only to a single trajectory with fixed x_T, fixed weight vector $\boldsymbol{\lambda}$, plain gradient descent, and iterates that remain inside one pre-specified smooth region $\mathcal{U}$. In the actual implementation, Adam replaces vanilla gradient descent, the optimizable conditioning coordinates are clipped to $[0, 1]$ after each step, the loss is only piecewise smooth because of hinge/max/min operators, and different trajectories use independently sampled Dirichlet weights. The proposition therefore should not be read as a global convergence guarantee for the full D2C objective.

Within that idealized smooth-region setting, a stationary point satisfies

$$\sum_{k=1}^{K} \lambda_k \nabla_c \mathcal{L}_k = 0. \tag{34}$$

This is the first-order stationarity condition for the scalarized objective associated with that particular weighting $\boldsymbol{\lambda}$. Sampling multiple weight vectors $\boldsymbol{\lambda}$ therefore provides a practical way to generate a diverse set of scalarized trade-offs, but it does not by itself establish broad Pareto-front coverage or any global Pareto-optimality guarantee.

Lemma 1 (Smooth One-Sided Penalties Are Differentiable Almost Everywhere). *Let $g : \mathbb{R}^d \to \mathbb{R}$ be continuously differentiable, and assume $G_\theta(c, x_T)$ is continuously differentiable in c. Define $\mathcal{L}_{\text{fid}}(c) = [g(G_\theta(c, x_T)) - g(x_{\text{ref}})]_+$ and the boundary set $\partial\mathcal{F} = \{c : g(G_\theta(c, x_T)) = g(x_{\text{ref}})\}$. Then $\mathcal{L}_{\text{fid}}$ is differentiable on $\mathbb{R}^{d_c} \setminus \partial\mathcal{F}$, with:*

$$\nabla_c \mathcal{L}_{\text{fid}}(c) = \begin{cases} \left(\dfrac{\partial x_0}{\partial c}\right)^T \nabla_{x_0} g & \text{if } g(x_0) > g(x_{\text{ref}}), \\ 0 & \text{if } g(x_0) < g(x_{\text{ref}}). \end{cases} \tag{35}$$

Moreover, if $g \circ G_\theta$ is a submersion at every boundary point (i.e., $\nabla_c(g \circ G_\theta)(c) \neq 0$ for all $c \in \partial\mathcal{F}$), then $\partial\mathcal{F}$ is locally a $(d_c{-}1)$-dimensional manifold and in particular has Lebesgue measure zero. Consequently, $\mathcal{L}_{\text{fid}}$ is differentiable almost everywhere.

Proof. The function $h(c) = g(G_\theta(c, x_T)) - g(x_{\text{ref}})$ is continuously differentiable since g and G_θ are. Since $[r]_+ = r \cdot \mathbf{1}_{r>0}$ is differentiable for $r \neq 0$, the composition $\mathcal{L}_{\text{fid}} = [h(c)]_+$ is differentiable wherever $h(c) \neq 0$, i.e., on $\mathbb{R}^{d_c} \setminus \partial\mathcal{F}$.

In the violated regime ($h(c) > 0$): $\nabla_c \mathcal{L}_{\text{fid}} = \nabla_c h = \left(\frac{\partial x_0}{\partial c}\right)^T \nabla_{x_0} g$. If additionally $\nabla_c h(c) \neq 0$, then the negative gradient is a nonzero local descent direction for the penalty.

In the feasible interior ($h(c) < 0$): $\mathcal{L}_{\text{fid}} = 0$ identically, so $\nabla_c \mathcal{L}_{\text{fid}} = 0$ and the penalty is inactive.

For the measure-zero claim: the preimage theorem states that if $\nabla_c h(c) \neq 0$ on the level set $h^{-1}(0)$, then $\partial\mathcal{F} = h^{-1}(0)$ is a smooth $(d_c{-}1)$-dimensional submanifold and hence has d_c-dimensional Lebesgue measure zero. This establishes almost-everywhere differentiability of $\mathcal{L}_{\text{fid}}$, but it does not imply that optimization trajectories cannot reach $\partial\mathcal{F}$; if an iterate lands on the hinge boundary in practice, first-order optimization is interpreted through the corresponding subgradient. □

Remark. Lemma 1 applies only to the smooth one-sided penalties obtained from continuously differentiable constraint functionals g. Some instantiated fidelity terms in Appendix E additionally use coordinatewise $\max / \min$ operators to enforce amplitude bounds; those terms are Lipschitz but not C^1, so the manifold and almost-everywhere differentiability claims above should not be read as applying to those non-smooth pieces. In practice, such terms are handled through the corresponding subgradients supplied by automatic differentiation.

H.3 Limits of the Frozen Surrogate Objective

The preceding results establish differentiability and local first-order properties of the conditioning optimization, but they do not imply that optimizing the frozen surrogate is equivalent to optimizing the true objective. The relevant question is instead whether the surrogate remains accurate on the region of design space reached through conditioning optimization.

Proposition 3 (Pairwise Ordering Reliability Under Uniform Surrogate Error). *Fix a latent noise draw x_T and write $x(c) = G_\theta(c, x_T)$. Let $\hat{f}(x) = S_\phi(x, c_{\text{fix}})$ denote the surrogate objective and let $f(x)$ denote the corresponding true objective. Let $\mathcal{R} \subseteq \{x(c) : c \in \mathbb{R}^{d_c}\}$ be any set that contains the optimization trajectory under consideration. Suppose that*

$$\sup_{x \in \mathcal{R}} |\hat{f}(x) - f(x)| \leq \varepsilon. \tag{36}$$

Then for any $x_1, x_2 \in \mathcal{R}$,

$$\big|\big(\hat{f}(x_1) - \hat{f}(x_2)\big) - \big(f(x_1) - f(x_2)\big)\big| \leq 2\varepsilon. \tag{37}$$

In particular, if $\hat{f}(x_1) \geq \hat{f}(x_2) + 2\varepsilon$, then $f(x_1) \geq f(x_2)$.

Proof. By direct expansion,

$$\big(\hat{f}(x_1) - \hat{f}(x_2)\big) - \big(f(x_1) - f(x_2)\big) = \big(\hat{f}(x_1) - f(x_1)\big) - \big(\hat{f}(x_2) - f(x_2)\big). \tag{38}$$

Applying the triangle inequality and the uniform error bound gives

$$\big|\big(\hat{f}(x_1) - \hat{f}(x_2)\big) - \big(f(x_1) - f(x_2)\big)\big| \leq |\hat{f}(x_1) - f(x_1)| + |\hat{f}(x_2) - f(x_2)| \leq 2\varepsilon. \tag{39}$$

The ordering statement follows immediately. □

Remark. This proposition is intentionally conditional. Freezing θ and ϕ does not, by itself, rule out input-space exploitation: optimizing c can still move $x(c)$ into parts of the generator image where the surrogate is inaccurate. What freezing changes is the search space, not the existence of blind spots. D2C instead relies on a weaker and more defensible claim: the frozen diffusion prior, physics-informed penalties, and validity-first filtering can reduce exploitation in practice by biasing search toward designs that remain closer to the empirical data distribution and satisfy explicit feasibility checks. These mechanisms are practical safeguards, not a general theoretical guarantee against surrogate exploitation.

I Practical Guidance for Design Optimization with D2C

D2C is designed for settings where (i) evaluations of the true objective are expensive (physical simulations, wet-lab experiments, or large-scale facility tests), (ii) hard physical constraints must be satisfied for safety or feasibility, and (iii) offline datasets of moderate size are available. Below we summarize the settings where the method is most compelling, together with regimes where simpler alternatives may sometimes be sufficient.

Settings Well Suited to D2C.

- **High-cost physical systems with expensive evaluations.** D2C is most valuable when each ground-truth evaluation costs hours of compute (e.g., radiation-hydrodynamics simulations) or involves irreversible physical consequences (e.g., laser shots on a fusion facility, equipment management in data centers). The offline training investment is amortized over many test-time optimizations.
- **Multi-objective problems with hard constraints.** The framework naturally accommodates competing objectives and strict feasibility requirements through the composite loss and validity-first selection. If your application has safety-critical constraints that cannot be violated, D2C provides explicit mechanisms to enforce them.
- **High-dimensional or strongly structured designs.** The diffusion proposer excels at generating designs with complex internal structure (temporal profiles, spatial fields) that naive gradient ascent in design space often cannot capture robustly. If the design space has strong correlations or manifold structure, the learned prior is especially beneficial.
- **Settings where a reliable surrogate exists.** The method requires a differentiable surrogate S_ϕ that is accurate in the region of interest. If surrogate error is low ($< 5\%$ relative error), the gradient signal is informative.

Settings Where Simpler Alternatives May Suffice.

- **Low-dimensional problems with weak structure or cheap evaluation.** If the design space is very small, the constraints are mild, and evaluations are cheap enough to support direct search or standard Bayesian optimization, then methods such as GP-qEI, CMA-ES, or even brute-force search may be simpler to deploy and sufficiently effective. This is not a blanket limitation of D2C: even in lower-dimensional settings, the method can still be worthwhile when evaluations are expensive, feasibility is safety-critical, or data-distribution-aware generation and validity-first selection materially improve robustness.
- **Cheap online evaluations available.** If the true objective can be queried cheaply and repeatedly, online methods (reinforcement learning, active learning) are often preferable because they can adapt directly to fresh evaluations. D2C is designed for settings with no online evaluation access at test time.
- **Severely miscalibrated surrogates.** If the surrogate is unreliable in the optimization region, gradient-based methods can exploit those inaccuracies. The diffusion prior and validity checks mitigate this to some extent by constraining designs toward the empirical data distribution, but a fundamentally miscalibrated surrogate can still misdirect optimization.
- **Latency-critical applications.** D2C requires backpropagation through a T-step DDIM chain at each of J optimization iterations. On current hardware, this takes 30–60 seconds per test sample. If sub-second inference is required, single-pass methods (MINs, DDOM) may be a better fit despite their weaker optimization and constraint-handling performance.

Practical considerations.

- **Surrogate investment.** We recommend training the surrogate to convergence with proper validation, achieving $< 2\%$ error before proceeding to diffusion model training.
- **Guidance scale sensitivity.** The guidance scale w is domain-dependent and must be tuned. Domains with tight physical constraints (e.g., ICF) are highly sensitive to $w > 2$; smoother landscapes (e.g., DC) are more tolerant. Start with $w = 2$ and sweep $\{1, 2, 3, 5\}$. The sweep does not require oracle access: ranking w by the surrogate-predicted gain of candidates that pass the geometric validity checks recovers the oracle-selected value (Section D.2).
- **Noise trials vs. optimization steps.** When compute budget is fixed, we find it more effective to increase noise trials N_z (broader coverage) than optimization steps J (deeper optimization per trial). Diminishing returns on J set in after 50–100 steps for most problems.
- **Constraint design.** Hard constraints should be expressed as differentiable penalty terms in the loss wherever possible. Non-differentiable constraints (e.g., discrete feasibility checks) should be handled via the post-hoc validity filter rather than incorporated into the gradient-based loop.

J Broader Impact

As a general offline design-optimization method, D2C could have positive impact in domains where evaluations are expensive and constrained operation matters, such as reducing carbon-footprint in data centers or accelerating scientific design loops while lowering simulation cost. More broadly, methods that keep optimization closer to the empirical data distribution captured by a learned prior while exposing explicit validity checks help reduce unrealistic proposals relative to unconstrained surrogate optimization.

Potential negative impacts are also application-specific. The same optimization machinery could be used in domains where accelerating design search is undesirable or access should be restricted.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: See the abstract, Section 1, and Sections 4–5; the claims are scoped to offline physics-informed optimization and the reported experiments.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Section 5 discusses computational cost, surrogate dependence, validity-first selection, and domain-specific tuning.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Sections 4–6 and the appendix specify datasets, splits, losses, hyperparameters, baselines, evaluation, and ablation settings.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We provide the code used for DC and design-bench attached in the supplementary material. The ICF data and its artifacts are export controlled and cannot be publicly shared (Appendix, ICF data generation).

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: Section 4 and Appendix A report the splits, architectures, optimizers, hyperparameters, and model-selection procedures.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: The main results and ablations report mean $\pm$ standard deviation over 5 runs; see Section 4.3 and the table captions.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix A reports the hardware (four NVIDIA H100 GPUs) and approximate training/optimization costs.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: To the best of our knowledge, yes; the work uses offline data/simulations, involves no human subjects, and discloses relevant release limitations.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The appendix now includes a short broader-impact discussion covering potential benefits, misuse risks, and deployment risks of the method.

Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The submission does not release a high-risk model or scraped dataset.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The paper credits the public code assets used and explicitly states the Design-Bench and SustainDC repository licenses in the appendix data section.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [N/A]

    Justification: The paper does not introduce a new publicly released dataset, model, or code asset.

    Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [N/A]

    Justification: The paper does not involve the use of LLMs as an important, original, or non-standard component of the core methods in this research.

    Guidelines:

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