GENERATIVE REFINEMENT FOR LOW-BUDGET BLACK BOX OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Black-box optimization is a fundamental tool in science and engineering for optimizing objectives when gradient information is unavailable. It becomes especially difficult when the objective function is expensive to evaluate, limiting the evaluation budget to a few tens or hundreds of queries, and when good solutions occupy complex, low-measure regions of the search space. Generative models can supply useful structural priors in such settings, but existing generative BBO approaches bring significant evaluation cost. We identify three design principles for generative optimization under such low-budget conditions: avoid objective learning, optimize in candidate space, and make every evaluation count. Together, these principles motivate separating structural modeling from objective-driven search. We instantiate them in *SPARROW*, a simple sequential optimizer that maintains a persistent, ranked archive of evaluated candidates and uses a fixed, unconditional generative sampler solely as a corruption–refinement operator. *SPARROW* requires only access to the sampler's corruption and refinement processes, and never needs to evaluate the objective to train or guide it. Across three complementary settings, probing thin feasible geometry, disconnected high-performing regions, and failure-prone evaluations, *SPARROW* outperforms classical and generative baselines under strict evaluation budgets. These results demonstrate that separating structural priors from objective-driven search can be effective when evaluations are scarce and the search geometry is challenging.

1 INTRODUCTION

Many problems in science and engineering require optimizing without access to gradient information, as in materials design (Frazier & Wang, 2015), drug discovery (Gómez-Bombarelli et al., 2018; Gruver et al., 2023), and engineering simulation (Diessner et al., 2022; Wu et al., 2021). This is referred to as *Black-Box Optimization* (BBO). It is particularly challenging when evaluations require expensive simulation or physical experimentation, as in high-fidelity engineering and experimental design. In industrial practice, this limits the number of evaluations that can be performed to tens or hundreds, rather than tens of thousands. The difficulty is further magnified when the evaluation feedback is noisy or failure-prone and when good solutions lie on thin, curved or disconnected regions of the search space. We refer to this regime as the very-low-budget (VLB) setting and focus on these scenarios in this paper. In this regime, only limited objective information is available to guide optimization.

Classical approaches can struggle in the VLB-BBO regime: Bayesian optimization (BO) is efficient when a reliable surrogate model is available (Shahriari et al., 2016), but sparse data relative to objective complexity can misdirect acquisition and waste evaluations. Evolutionary strategies (ES) such as CMA-ES (Hansen, 2016) sidestep this surrogate requirement, but can require more evaluations to adapt their search distribution and may waste evaluations when that distribution is poorly matched to the geometry of good solutions.

In recent years, diffusion (Ho et al., 2020; Song et al., 2020; 2021) and flow matching (Lipman et al., 2022) models have emerged as powerful tools for representing complex, high-dimensional distributions. This has motivated generative approaches to BBO when domain data is available (Krishnamoorthy et al., 2023; Wu et al., 2024; Uehara et al., 2025). Such data may provide rich structural information while remaining unlabeled with respect to the objective of interest. Unfortunately,

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

these methods require either a large number of evaluations of the objective function on the training dataset, or at inference time, both of which are incompatible with the VLB setting.

In this paper, we first identify three design principles for generative VLB-BBO. In this scenario, we propose to: avoid objective learning, perform optimization in candidate space, and make every evaluation count. Taken together, these principles suggest a different way to use generative models in BBO, that is, as fixed structural priors, decoupled from the objective signal. We then introduce *SPARROW* (**S**equential **P**roposal via **A**rchival **R**ank-based **R**efinement for **O**ptimization under **W**eak feedback) as a simple instantiation of these principles. It keeps an unconditional generative sampler fixed, performs objective-driven operations directly over fully denoised candidates, and optimizes sequentially over a persistent ranked archive, allowing every evaluation to inform subsequent proposals.

We demonstrate strong empirical performance on complementary problems that stress existing optimization approaches: low-measure feasible subspaces, disconnected high-performing regions, and unreliable evaluations. *SPARROW* consistently outperforms classical and generative baselines under strict evaluation budgets. Our key contribution is therefore a new way to use generative models as fixed structural priors, decoupled from the objective signal as motivated by the design principles, for efficient BBO in challenging scenarios.

2 Related Work

2.1 Classical black-box optimization

Bayesian optimization (BO) repeatedly fits a surrogate of the objective and selects new candidates through an acquisition function (Shahriari et al., 2016). Local methods such as TuRBO (Eriksson et al., 2019) improve scalability by restricting optimization to adaptive trust regions. BO can be highly sample-efficient when the surrogate is informative, but sparse observations relative to objective complexity can make fitting such surrogate difficult. Evolutionary strategies avoid this by guiding the search directly through evaluated candidates. Among them, CMA-ES (Hansen, 2016) is one of the most popular and operates by sampling a Gaussian distribution whose parameters are adapted based on previous evaluations. While often effective, it can become inefficient when the distribution of high-performing candidates is not easily captured by a Gaussian. Differential evolution (Lampinen & Storn, 2004) instead estimates search directions from differences between evaluated candidates, but generally requires larger populations and evaluation budgets for sustained progress. For a more extensive description of classical BBO, we refer the interested reader to Conn et al. (2009).

2.2 Generative black-box optimization

Generative models have recently been used in BBO to provide structure over complex candidate spaces. Existing methods mainly differ in where the objective enters the generative process: during training, at inference time, or externally through evolutionary selection. These choices lead to different trade-offs in sample efficiency and robustness.

Objective-dependent generative optimization. Many approaches condition the generative model on the objective to sample high-performing candidates directly. CbAS (Brookes et al., 2019), DDOM (Krishnamoorthy et al., 2023), Diffusion-BBO (Wu et al., 2024), BONET (Mashkaria et al., 2023), and DiBO (Yun et al., 2025) learn objective-dependent generation from labeled data or online evaluations, while DEMO (Yuan et al., 2024) and RGD (Chen et al., 2024) rely on learned surrogates. Under very small budgets, the available objective evaluations are often insufficient to reliably fit these objective-dependent models, and resulting model errors can bias subsequent proposals.

Inference-time guidance. Other approaches keep the generator fixed and use objective information directly during sampling. DNO optimizes diffusion noise (Tang et al., 2024), while DAS and RERD bias denoising trajectories toward high rewards (Kim et al., 2025; Uehara et al., 2025). This optimization is indirect, as updates are transformed by the remaining generative trajectory before producing a candidate. Prior work reports that this can degrade guidance (Tan et al., 2025; Tang

et al., 2024). Sampling-based methods can additionally require several reward computations during a single generation. Both issues are undesirable in a VLB setting.

Generative evolutionary search. Generative models have also been used directly as mutation operators in evolutionary search, keeping the optimization signal in candidate space. Diffusion-ES (Yang et al., 2024) selects and refines populations of trajectories using an unconditional diffusion model, while EvoDiff-NAS (Tang et al., 2026) uses diffusion-guided mutation for neural architecture search. Both were developed for application-specific settings where population-level evaluations are affordable, and therefore do not target the evaluation constraints of VLB-BBO.

3 Design Principles for Generative VLB-BBO

In very-low-budget black-box optimization, objective evaluations are the main scarce resource. This changes how generative priors should be used for optimization. We identify three desirable design principles.

Avoid objective learning. With only tens to hundreds of evaluations affordable, fitting a surrogate, objective-conditioned generator, or reward-dependent target can produce unreliable approximations whose errors directly bias future proposals. This affects both classical BO and objective-dependent generative methods. Thus, in the VLB regime, it is better to use objective evaluations directly for optimization without approximating the objective.

Optimize in candidate space. Many existing methods use objective information as guidance along the generative trajectory. In the VLB regime, this signal is already weak because it is estimated from very few evaluations. Applying it through intermediate noisy states can further distort its effect before a final candidate is produced (Tan et al., 2025; Tang et al., 2024). Thus, objective-driven operation should be performed in candidate space, leaving the generative trajectory unguided.

Make every evaluation count. When the total budget is small, performing multiple evaluations internally and discarding them after use is wasteful. For example, SMC-based guidance uses many internal evaluations for one single pass, and existing generative evolutionary approaches discard past evaluations when updating the current population. We therefore favor sequential optimization that retains every evaluation and lets it inform future proposals.

Together, these principles suggest a clean separation of roles. The generative model provides structural bias, while objective information drives optimization only in candidate space.

4 Method

SPARROW instantiates the three design principles. At each iteration, it selects a parent from the archive, applies a rank-guided mutation, and uses the fixed sampler to corrupt and refine the resulting candidate.

4.1 Background

We briefly review two of the building blocks used by *SPARROW*: generative corruption–refinement operators, and rank-based comparisons of objective feedback.

Generative Samplers and Refinement via Partial Noising We consider generative models that map noise to data through a trajectory. Let $t \in [0, 1]$ denote a noise level, with $t = 0$ being pure noise and $t = 1$ the data distribution, and let $\mathcal{X}_t$ denote the space of states at noise level t. We take

$$S_{t\to 1} : \mathcal{X}_t \to \mathcal{X}$$

to be the refinement operator that maps a partially noisy state to a sample in data space by following the generative trajectory. Sampling corresponds to applying $S_{0\to 1}$ to a draw from the noise prior, inducing a distribution ν over $\mathcal{X}$.

Let

$$\mathcal{N}_t : \mathcal{X} \to \mathcal{X}_t$$

denote the corruption operator at noise level t, which maps a candidate $x \in \mathcal{X}$ to a noisy state $x_t \sim \mathcal{N}_t(x)$ according to the forward process used during the training of the sampler. The composition of $\mathcal{N}_t$ and $S_{t\to 1}$ induces a family of stochastic transition kernels over the search space, indexed by a controllable corruption level

$$\mathcal{T}_t(x) := S_{t\to 1}(x_t), \quad \text{where } x_t \sim \mathcal{N}_t(x).$$

This construction generalizes stochastic editing procedures such as SDEdit (Meng et al., 2022), where partial corruption followed by denoising biases inputs towards the sampler's data distribution. Here, $\mathcal{N}_t$ and $S_{t\to 1}$ are defined abstractly. In practice, any generative sampler supporting a consistent corruption–refinement decomposition can be used, without access to its internal parametrization.

The noise level t controls the exploration scale: large t produces local refinements, small t induces global modifications. In particular, $\mathcal{T}_0(x) \sim \nu$ for all $x \in \mathcal{X}$, since full corruption erases all input structure and produces samples from the prior.

Rank-based Optimization and Invariance Rank-based optimization replaces raw objective values with their induced ordering, so that all decisions depend on $\mathrm{rank}(x)$ rather than $f(x)$. This yields invariance to strictly monotone transformations of the objective (Ollivier et al., 2017; Hansen, 2016), making the optimization signal more robust to scaling, misspecification, and outcome-dependent noise. The tradeoff is that ranking discards magnitude information, making rank-based updates less efficient than value-based ones when reliable magnitude information is available. In our regime, however, this tradeoff is favorable: robustness and invariance outweigh potential efficiency when evaluations are scarce.

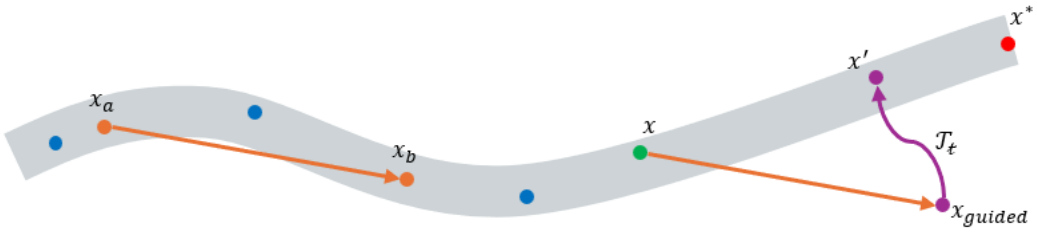

Figure 1: One iteration of *SPARROW* on a thin tube problem, with optimality on one end of the subspace. Archive points lie on the sampler support. A rank-guided directional step moves the parent in a promising direction. The operator $\mathcal{T}_t$ then biases the candidate back towards the sampler support.

4.2 Algorithm and Design Properties

As is described in Algorithm 1, *SPARROW* is a sequential algorithm that iteratively adds single newly evaluated candidates to a persistent archive. At each iteration, as illustrated in Figure 1, *SPARROW* first selects a parent from the ranked archive, favoring high-performing candidates while retaining nonzero probability for all archive members. It then samples two additional archive elements and uses their ranks to orient a differential-evolution direction toward the better candidate, providing a second objective-driven search signal. The displaced parent is corrupted to a rank-dependent noise level and refined through the fixed generative sampler. Better-ranked parents are corrupted less and are therefore refined more locally, while lower-ranked parents are corrupted more and induce broader exploration. The resulting proposal is evaluated once and added to the archive. This implementation follows our three design principles:

No objective learning. The optimization is driven by two simple mechanisms. First, parent selection favors high-ranking candidates in the archive. We show in Appendix A that it follows a maximum-entropy distribution under an expected-rank constraint (Eiben & Smith, 2015), a standard form of rank-based selection in BBO. Second, the directional mutation, inspired by differential evolution (Lampinen & Storn, 2004), provides an additional stochastic bias toward high-performing regions. Neither mechanism requires learning or approximating the objective.

Optimization in candidate space. The optimization engines, parent selection and directional mutation, operate entirely on denoised candidates. The generative sampler is never guided by the objective and only acts afterwards through $\mathcal{T}_t$, pulling proposals toward regions represented by the

Algorithm 1 *SPARROW*

Require: Initial archive $\mathcal{A}_0 = \{(x_i, f(x_i))\}_{i=1}^{n_0}$, fixed corruption–refinement operator $\mathcal{T}_t$, budget B, selection pressure β, corruption exponent γ, step size λ

1: **for** $k = 0, 1, \ldots, B - n_0 - 1$ **do**

2: Compute normalized archive ranks ▷ Lower rank is better

$$r_k(x_i) = \frac{\mathrm{rank}_{\mathcal{A}_k}(x_i)}{n_k}$$

3: Sample parent ▷ Prefer better candidates. Keep non-zero probability on all

$$x_k \sim p_{\mathrm{par}}(\cdot \mid \mathcal{A}_k) \propto \exp(-\beta r_k(\cdot))$$

4: Set rank-dependent corruption level ▷ Good parents: local refinement. Poor parents: stronger corruption

$$t_k \leftarrow 1 - r_k(x_k)^\gamma$$

5: Sample $x_a, x_b \in \mathcal{A}_k$ uniformly at random

6: Form guided candidate ▷ Differential evolution step ▷ Larger step when stronger refinement follows

$$x_{\mathrm{guided}} \leftarrow x_k + \lambda(1 - t_k)\,\mathrm{sign}\big(r_k(x_b) - r_k(x_a)\big)(x_a - x_b)$$

7: Apply the structural prior $x'_{k+1} \sim \mathcal{T}_{t_k}(x_{\mathrm{guided}})$

8: Evaluate and add to the archive $\mathcal{A}_{k+1} \leftarrow \mathcal{A}_k \cup \{(x'_{k+1}, f(x'_{k+1}))\}$

9: **end for**

10: **return** best candidate in the final archive

structured data distribution. The corruption level controls the strength of this refinement, but no objective information is injected along the generative trajectory.

Leveraging every evaluation. *SPARROW* is entirely sequential and performs no evaluations for internal guidance computations. All information obtained from evaluations is retained in the archive and can influence future proposals. Reducing feedback to ranks further improves robustness in poorly understood objective landscapes.

Together, these properties decouple objective-driven search from structural modeling. The archive drives optimization, while the fixed generative sampler acts only as a structural refinement operator.

We show in Appendix B that the proposed search mechanism eventually converges to the global optimum of the sampler's support. Finite-budget performance is evaluated empirically in Section 5.

5 Experiments

5.1 Tasks

We evaluate *SPARROW* on three complementary scenarios chosen to pose distinct challenges to VLB-BBO. First, a synthetic thin-tube benchmark involves a low-measure structured region, highlights the importance of structured priors and enables controlled ablation of the geometric difficulty. Second, HopperController from Design-Bench (Trabucco et al., 2022) requires optimization in a 5126-dimensional landscape with separated high-performing regions. Third, airfoil aerodynamic optimization is of direct industrial relevance and tests the method given failure-prone simulation-based evaluations. Additional results on the remaining continuous Design-Bench tasks are reported in Appendix E. These smoother settings help delineate the regime in which the added structural prior is most useful.

5.2 Evaluation Protocol

All methods are compared under the same budget of objective-function calls, including those used internally. Since exhaustive hyperparameter tuning is itself incompatible with the VLB setting,

we sequentially test five human-informed configurations per tunable method and task, using one run per configuration. The selected configuration is then evaluated over 10 independent runs. We report the median and interquartile range of the best objective value as a function of evaluations consumed. More implementation details are given in Appendix H, and the tuning procedure is detailed in Appendix F.

5.3 BASELINES

We compare *SPARROW* against three classical BBO baselines and three generative BBO baselines spanning unconditional sampling, reward-guided refinement, and surrogate-guided generation.

CMA-ES (Hansen, 2016) is the standard evolution strategy, maintaining an adaptive Gaussian search distribution in ambient space. GP is a Gaussian process surrogate with expected improvement acquisition, a strong standard BO baseline. TuRBO (Eriksson et al., 2019) augments GP with local trust regions, often more adapted as dimensionality scales.

RDS (Random Diffusion Sampling) draws proposals unconditionally from the generative prior without selection or archive, which separates the contribution of guidance from the structural contribution of the prior alone. RERD (Uehara et al., 2025) biases generation trajectories through repeated sampling during inference. We also construct GPGuide, which combines the GP surrogate with the generative prior, doing conditional guidance toward high rewards along the generation trajectory.

We exclude methods that train an objective-conditioned generative model from online evaluations because $B = 100$ observations are insufficient to fit such models reliably and would make the comparison poorly representative of their intended use.

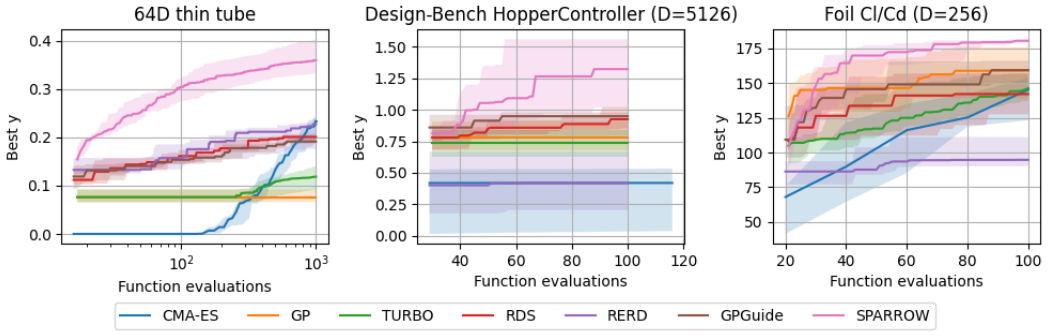

Figure 2: **Objective median value and 25/75 Inter Quartile Range (IQR)** as a function of the number of times the objective function is evaluated, up to $B = 1000$ for the 64-D tube on the left and to $B = 100$ when maximizing the HopperController reward and airfoil C_l/C_d on the middle and right.

Table 1: Best objective value found (median, IQR over 10 runs). The tube is evaluated at two budgets to assess scaling behavior; airfoil at $B = 100$, consistent with realistic aerodynamic design constraints; HopperController at $B = 100$ to probe performance in the low-budget regime on a standard benchmark.

	64D Tube		**HopperController**	**Airfoil C_l/C_d**
Method	$B = 100$	$B = 1000$	$B = 100$	$B = 100$
CMA-ES	$0.00_{[0.00,\,0.00]}$	$0.23_{[0.19,\,0.24]}$	$0.42_{[0.03,\,0.53]}$	$145.30_{[123.26,\,153.68]}$
GP	$0.08_{[0.07,\,0.09]}$	$0.08_{[0.07,\,0.09]}$	$0.78_{[0.68,\,0.91]}$	$158.85_{[139.20,\,175.94]}$
TuRBO	$0.08_{[0.07,\,0.09]}$	$0.12_{[0.09,\,0.14]}$	$0.74_{[0.63,\,0.84]}$	$146.32_{[137.31,\,157.08]}$
RDS	$0.16_{[0.14,\,0.17]}$	$0.20_{[0.19,\,0.21]}$	$0.93_{[0.85,\,1.02]}$	$142.11_{[127.47,\,156.78]}$
RERD	$0.16_{[0.13,\,0.19]}$	$0.23_{[0.18,\,0.24]}$	$0.42_{[0.20,\,0.67]}$	$94.67_{[90.12,\,111.11]}$
GPGuide	$0.15_{[0.15,\,0.16]}$	$0.19_{[0.19,\,0.20]}$	$0.95_{[0.79,\,0.98]}$	$159.37_{[150.04,\,166.12]}$
SPARROW (ours)	$\mathbf{0.30}_{[0.28,\,0.32]}$	$\mathbf{0.36}_{[0.33,\,0.40]}$	$\mathbf{1.32}_{[1.02,\,1.56]}$	$\mathbf{180.47}_{[179.14,\,181.54]}$

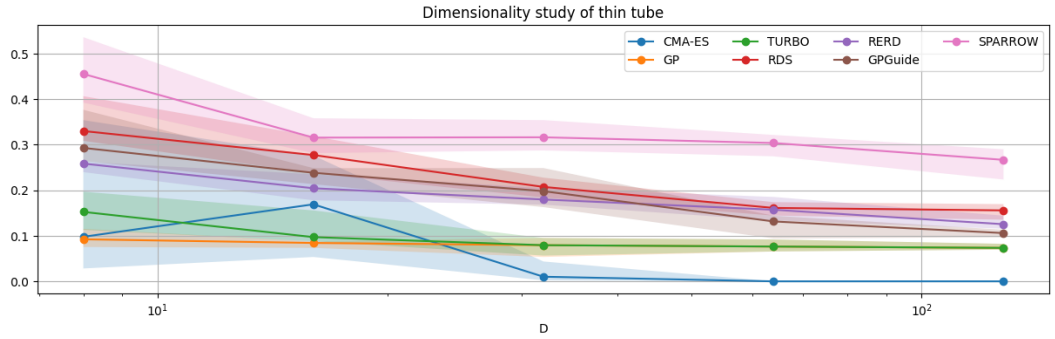

Figure 3: **Objective median value and IQR as a function of the tube dimension** using $B = 100$ evaluations. *SPARROW* again outperforms the baselines across dimensions. The gap between *SPARROW* and RDS widens with increasing D, indicating that rank-based selection becomes increasingly valuable as the geometry grows more complex. Exact values are reported in Appendix G.

5.4 Thin High-Dimensional Tube

We define a synthetic benchmark in which optimal solutions lie near the end of a thin, curved, one-dimensional manifold embedded in a high-dimensional ambient space. Figure 1 illustrates the geometry. While synthetic, this setting shows the core challenge *SPARROW* is designed for: most of the search space yields poor or invalid solutions, and progress requires staying near a structured feasible region.

Experimental Setup. The benchmark consists of a smooth curved tube of radius $r = 0.1$ with a start and finish. The objective rewards points near the finish, but only within the tube. Points outside the tube score worse than points near the start. We report main results at $D = 64$, and vary $D \in \{8, 16, 32, 64, 128\}$ for the ablation. The sampler is trained on points sampled along the tube with a $\mathrm{Beta}(1, 3)$ distribution over arc-length, concentrating mass near the start and away from the optimum. This reflects a realistic scenario where structured data is abundant, but sparse near the optimum. Full formalism and implementation are given in Appendix C.

Results. Fig. 2 (left) and Tab. 1 show reward values as a function of consumed evaluation budget. At $B = 100$, *SPARROW* clearly outperforms all baselines and retains the best performance through $B = 1000$.

CMA-ES makes no progress in the VLB regime, but begins improving after several hundred evaluations. Its delayed improvement is consistent with the Gaussian search distribution requiring many evaluations before concentrating on the thin feasible region. GP remains close to its initial performance, consistent with most ambient-space evaluations providing little information about progress along the thin tube. TuRBO improves modestly at larger budgets, suggesting that localization helps but does not overcome the limited feedback.

RDS makes consistent progress, confirming that the generative prior encodes useful geometric structure. GPGuide slightly underperforms RDS. Early in the search, its GP is fitted to very few evaluations, most of which provide little information about progress along the tube. Its guidance is therefore largely extrapolative and can mislead the sampler rather than improve it. RERD slightly outperforms RDS, showing that reward-guided refinement helps. However, it spends part of the budget on importance weighting and resampling during refinement, rather than directly accumulating new optimization feedback.

SPARROW's guidance adds value beyond the prior. Fig. 3 shows that performance generally degrades as D increases because the tube occupies a smaller fraction of ambient space. Nevertheless, *SPARROW* outperforms all baselines, with a more pronounced relative advantage over unconditional sampling at high dimensions. This suggests that guidance becomes particularly valuable as dimensionality grows.

5.5 HOPPERCONTROLLER

HopperController is a $D = 5126$ continuous control task, in which the search space is the parameter space of a neural network policy. The objective is a combination of the hopper's velocity, control cost, and a bonus for staying upright. The landscape contains many highly disconnected high-performing regions, as shown in Appendix E.

Experimental Setup. We train an unconditional flow matching model on the HopperController training set and initialize all methods from the generative prior. Training points are excluded from initialization because the dataset contains a small number of near-optimal solutions that prevent any method from improving upon the initial archive. The objective is evaluated using the Gaussian process oracle provided by Design-Bench.

Results. Fig. 2 (middle) and Tab. 1 show results at $B = 100$. *SPARROW* substantially outperforms all baselines.

CMA-ES fails to identify an optimization signal, as in the early stages of the tube experiment. Its Gaussian search distribution is poorly matched to a 5126-dimensional space with a structured feasible region and low feedback. GP and TuRBO plateau near the initialization quality, consistent with their surrogates being too poorly calibrated to identify improvement directions.

RDS makes modest progress, confirming that the generative prior encodes useful structure. GPGuide improves slightly further, but remains limited by the sparse feedback available to fit its surrogate. RERD stays close to its initial performance. Unlike the connected tube, Hopper contains disconnected high-performing regions, where progressive reward-guided refinement may be less effective. Its repeated reward queries further limit how many refinement attempts fit within $B = 100$. *SPARROW* instead combines local refinement of good archive candidates with stronger corruption, up to near-unconditional resampling, allowing it to explore new regions while retaining all previously discovered solutions.

5.6 AIRFOIL AERODYNAMIC OPTIMIZATION

We evaluate *SPARROW* on a real-world aerodynamic design task, maximizing the lift-to-drag ratio C_l/C_d of a two-dimensional airfoil computed by XFOIL (Drela, 1989), a widely used panel-method solver. The landscape of aerodynamic objectives is notoriously complex due to turbulence, flow separation, and operation outside the solver's validity range, causing frequent solver failures. As they still consume computational resources, we treat such failures as evaluations: They count against the evaluation budget but are not added to the archive.

Experimental Setup. We use the generative prior of DiffAirfoil (Wei et al., 2024), a diffusion model coupled to an auto-decoder that maps a 256-dimensional latent code to airfoil coordinates. We train it on the UIUC airfoil database (Selig, 1996) and generated NACA profiles, providing a prior over geometrically valid airfoils without aerodynamic supervision. *SPARROW* operates entirely in this latent space, using DiffAirfoil's forward and reverse processes as the mutation operator $\mathcal{T}_t$. All baselines likewise operate in the latent space. The auto-decoder is used only for decoding before evaluation by XFOIL. More details about the implemented pipeline and the behaviour of its latent space are given in Appendix D.

Results. Fig. 2 (right) and Tab. 1 show results at $B = 100$. *SPARROW* improves rapidly and reaches the highest final C_l/C_d with a notably tight IQR.

GP performs strongly early in the search before slowing down. CMA-ES and TuRBO improve more gradually throughout the budget but remain below the strongest methods. This suggests that the aerodynamic objective is sufficiently capturable from sparse observations for traditional methods to be effective.

RDS also makes steady progress, confirming that the generative prior captures useful airfoil structure, while RERD makes little improvement under the limited budget. GPGuide performs similarly to GP. Its surrogate is learned from evaluated latent codes but used to guide partially noised states, where its predictions are less reliable.

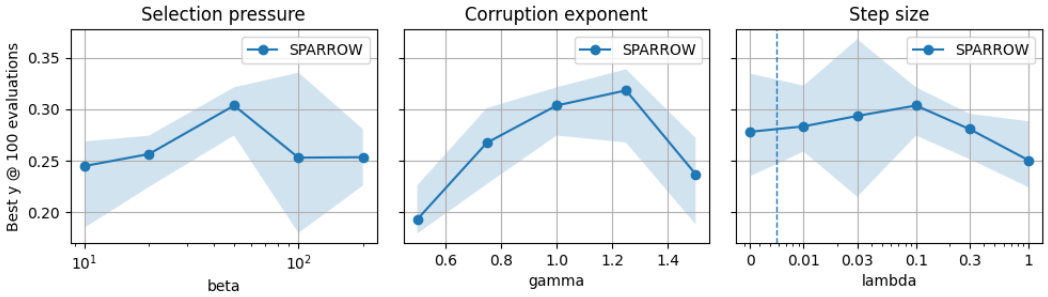

Figure 4: Hyperparameter sensitivity on the 64D thin tube at $B = 100$ evaluations (median and IQR over 5 runs). Each parameter is varied independently with others fixed at their default values $\beta = 50$, $\gamma = 1$, $\lambda = 0.1$. Exact values are reported in Appendix G.

5.7 Hyperparameter Sensitivity

SPARROW has three hyperparameters: selection pressure β, corruption exponent γ, and mutation step size λ. Fig. 4 shows performance at $B = 100$ as each is varied independently around the defaults $\beta = 50$, $\gamma = 1$, $\lambda = 0.1$ on the 64D tube. Low β makes parent selection nearly uniform and rank information ceases to guide the search. At high β, excessive elitism causes stagnation. At $\gamma = 0$, $t_k = 0$ for all candidates, reducing the method to RDS. High γ produces near-identity proposals around top-ranked candidates, causing stagnation around current elites. At $\lambda = 0$, directional mutation is removed entirely. Performance remains strong but below the best nonzero setting, showing that directional guidance is useful but not essential. At large λ, aggressive mutations increasingly move candidates away from good regions before refinement.

In all cases *SPARROW* performs well across a broad intermediate range, though γ is the most sensitive parameter. For our experiments, parameters were derived from these qualitative insights rather than exhaustive search, suggesting that the sensitivity analysis provides useful guidance for selecting practical settings. More details are given in Appendix F.

6 Limitations

SPARROW is designed for low-budget optimization on complex geometric landscapes with unreliable feedback. With simple objectives, reliable feedback, or high evaluation budgets, we expect more established methods to be more effective.

SPARROW's performance is bounded by the sampler quality: Solutions outside the generative prior's support are unreachable. In practice, this means *SPARROW* inherits any biases of the prior.

The three hyper-parameters β, γ, λ require domain-specific initialization. Section 5.7 provides a sensitivity analysis that we found sufficient to transfer across our three domains, but more principled adaptation strategies or less sensitive evolutions of *SPARROW* remain future work.

7 Conclusion

We studied generative black-box optimization in the very-low-budget regime, where objective evaluations are the main scarce resource. We argued for three design principles: avoid objective learning, perform optimization in candidate space, and make every evaluation count. Together, they suggest separating structural modeling from objective-driven search.

Guided by these principles, we introduced *SPARROW*, a simple sequential optimizer combining a persistent ranked archive with a fixed generative refinement operator. Across complementary tasks with thin feasible geometry, disconnected high-performing regions, and failure-prone evaluations, this simple instantiation achieved the strongest final performance under strict evaluation budgets. These results support the broader design principle that generative models can be most useful in VLB-BBO as fixed structural priors rather than objective-dependent optimizers.

AI USE STATEMENT

We used ChatGPT and Claude as research-support tools, primarily for critical review of manuscript drafts, feedback on the presentation of methodology and experiments, language editing and rephrasing, and literature search and discovery. The research ideas, methods, experimental design, mathematical arguments, and reported results were developed by the authors. AI-assisted code was limited to non-algorithmic tasks such as visualization and formatting and did not affect the implementation logic or generation of experimental results. No generative AI was used to generate experimental data. All AI-assisted material was manually reviewed, and technical claims, numerical results, and references were independently checked. The authors take responsibility for the full content of the paper.

ETHICS STATEMENT

This work studies general-purpose black-box optimization and does not involve human subjects, personal data, or sensitive datasets. We are not aware of ethical concerns specific to the methods or experiments beyond those associated with the general use of optimization methods.

REPRODUCIBILITY STATEMENT

We provide the information required to reproduce the reported results throughout the paper and appendices. A clean repository will be released upon publication.

REFERENCES

Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. Botorch: A framework for efficient monte-carlo bayesian optimization. *Advances in Neural Information Processing Systems*, 33:21524–21538, 2020.

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

Can Sam Chen, Christopher Beckham, Zixuan Liu, Xue Liu, and Christopher Pal. Robust guided diffusion for offline black-box optimization. *arXiv Preprint*, 2024.

Andrew R Conn, Katya Scheinberg, and Luis N Vicente. *Introduction to derivative-free optimization*. SIAM, 2009.

Mike Diessner, Joseph O'Connor, Andrew Wynn, Sylvain Laizet, Yu Guan, Kevin Wilson, and Richard D Whalley. Investigating bayesian optimization for expensive-to-evaluate black box functions: Application in fluid dynamics. *Frontiers in Applied Mathematics and Statistics*, 8: 1076296, 2022.

M. Drela. XFOIL: An Analysis and Design System for Low Reynolds Number Airfoils. In *Conference on Low Reynolds Number Aerodynamics*, pp. 1–12, 1989.

Agoston E Eiben and James E Smith. *Introduction to Evolutionary Computing*. Springer, 2015.

David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable global optimization via local bayesian optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Peter I Frazier and Jialei Wang. Bayesian optimization for materials design. In *Information science for materials discovery and design*, pp. 45–75. Springer, 2015.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268, 2018.

Nate Gruver, Samuel Stanton, Nathan Frey, Tim GJ Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew G Wilson. Protein design with guided discrete diffusion. *Advances in Neural Information Processing Systems*, 36:12489–12517, 2023.

Kam Hamidieh. A data-driven statistical model for predicting the critical temperature of a superconductor. *Computational Materials Science*, 154:346–354, 2018.

Nikolaus Hansen. The CMA Evolution Strategy: A Tutorial. *arXiv Preprint*, 2016.

Nikolaus Hansen, yoshihikoueno, Sait Cakmak, Gabi Kadlecová, Guillermo Abad López, Kento Nozawa, Luca Rolshoven, Youhei Akimoto, brieglhostis, and Dimo Brockhoff. Cma-es/pycma: r4.4.0, September 2025. URL `https://doi.org/10.5281/zenodo.17167557`.

J. Ho, A. Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, 2020.

Carl Hvarfner, Erik Orm Hellsten, and Luigi Nardi. Vanilla Bayesian optimization performs great in high dimensions. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 20793–20817, 2024.

Sunwoo Kim, Minkyu Kim, and Dongmin Park. Test-time alignment of diffusion models without reward over-optimization. In *International Conference on Learning Representations*, volume 2025, pp. 87238–87272, 2025.

Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion Models for Black-Box Optimization. In *International Conference on Machine Learning*, 2023.

Jouni Lampinen and Rainer Storn. Differential evolution. In *New Optimization Techniques in Engineering*, pp. 123–166. 2004.

Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow Matching for Generative Modeling. In *arXiv Preprint*, 2022.

Satvik Mehul Mashkaria, Siddarth Krishnamoorthy, and Aditya Grover. Generative pretraining for black-box optimization. In *International Conference on Machine Learning*, pp. 24173–24197. PMLR, 2023.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

Yann Ollivier, Ludovic Arnold, Anne Auger, and Nikolaus Hansen. Information-geometric optimization algorithms: A unifying picture via invariance principles. *Journal of Machine Learning Research*, 18(18):1–65, 2017.

Michael S. Selig. *UIUC airfoil data site*. Department of Aeronautical and Astronautical Engineering, University of Illinois at Urbana-Champaign, 1996.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, {Ryan P.} Adams, and Nando {De Freitas}. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 2016.

Francisco J Solis and Roger J-B Wets. Minimization by random search techniques. *Mathematics of Operations Research*, 6(1):19–30, 1981.

J. Song, C. Meng, and S. Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*, 2021.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*, 2020.

Kim Yong Tan, Yueming Lyu, Ivor Tsang, and Yew-Soon Ong. Fast direct: Query-efficient online black-box guidance for diffusion-model target generation. In *International Conference on Learning Representations*, volume 2025, pp. 19385–19422, 2025.

Zeyu Tang, Zaipeng Xie, Xiaoyi Wen, Bin Tang, and WenZhan Song. Evodiff-nas: Evolutionary neural architecture search with diffusion-guided mutation. *Swarm and Evolutionary Computation*, 107:102442, 2026.

Zhiwei Tang, Jiangweizhi Peng, Jiasheng Tang, Mingyi Hong, Fan Wang, and Tsung-Hui Chang. Inference-time alignment of diffusion models with direct noise optimization. In *International Conference on Machine Learning*, 2024.

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

Masatoshi Uehara, Xingyu Su, Yulai Zhao, Xiner Li, Aviv Regev, Shuiwang Ji, Sergey Levine, and Tommaso Biancalani. Reward-Guided Iterative Refinement in Diffusion Models at Test-Time with Applications to Protein and DNA Design. In *International Conference on Machine Learning*, pp. 60515–60529, 2025.

Zhen Wei, Benoît Guillard, Michaël Bauerheim, Vincent Chapin, and Pascal Fua. Latent Representation of CFD Meshes and Application to 2D Airfoil Aerodynamics. *American Institute of Aeronautics and Astronautics Journal*, 2023.

Zhen Wei, Edouard Dufour, Pascal Fua, and Michaël Bauerheim. DiffAirfoil: An Efficient Novel Airfoil Sampler Based on Latent Space Diffusion Model for Aerodynamic Shape Optimization. In *AIAA AVIATION Forum*, 2024.

Dongxia Wu, Nikki Lijing Kuang, Ruijia Niu, Yi-An Ma, and Rose Yu. Diffusion-bbo: Diffusion-based inverse modeling for online black-box optimization. *arXiv Preprint*, 2024.

Wensi Wu, Christophe Bonneville, and Christopher Earls. A principled approach to design using high fidelity fluid-structure interaction simulations. *Finite Elements in Analysis and Design*, 194: 103562, 2021.

B. Yang, H. Su, N. Gkanatsios, T.-W. Ke, A. Jain, J. Schneider, and K. Fragkiadaki. Diffusion-Es: Gradient-Free Planning with Diffusion for Autonomous and Instruction-Guided Driving. In *Conference on Computer Vision and Pattern Recognition*, pp. 15342–15353, 2024.

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. *arXiv preprint arXiv:2405.13964*, 2024.

Taeyoung Yun, Kiyoung Om, Jaewoo Lee, Sujin Yun, and Jinkyoo Park. Posterior inference with diffusion models for high-dimensional black-box optimization. In *International Conference on Machine Learning*, pp. 73897–73917, 2025.

A Maximum-entropy derivation of rank-based parent selection

We show that the exponential rank-based selection rule

$$p_i = \frac{\exp(-\beta r_i)}{\sum_j \exp(-\beta r_j)}$$

arises as the maximum-entropy distribution over archive elements under a constraint on expected rank.

Let $r_i \in [0, 1]$ denote the normalized rank of candidate x_i. We seek a distribution $p = (p_1, \ldots, p_n)$ that maximizes entropy

$$H(p) = -\sum_i p_i \log p_i$$

subject to

$$\sum_i p_i = 1, \qquad \sum_i p_i r_i = \rho.$$

The Lagrangian is

$$\mathcal{L}(p, \lambda, \beta) = -\sum_i p_i \log p_i + \lambda\left(\sum_i p_i - 1\right) - \beta\left(\sum_i p_i r_i - \rho\right).$$

Setting $\partial\mathcal{L}/\partial p_i = 0$ gives

$$\log p_i = \lambda - 1 - \beta r_i,$$

hence

$$p_i \propto \exp(-\beta r_i).$$

The parameter β controls the expected rank: $\beta = 0$ yields uniform selection, while larger β increases preference for better-ranked candidates.

B Asymptotic Global Consistency over the Sampler Support

We show that *SPARROW* is asymptotically globally consistent over the support of the unconditional sampler. The proof follows the classical framework of Solis & Wets (1981). The key point is that *SPARROW* naturally satisfies its conditions: the exponential selection rule assigns positive probability to the worst-ranked candidate at every iteration, which produces an unconditional sample from ν, thereby ensuring persistent exploration of the sampler support without requiring explicit random restarts.

Let ν denote the sampler's resulting distribution and assume that full corruption produces an unconditional sample, i.e.,

$$\mathcal{T}_0(x) \sim \nu \quad \text{for all } x \in X.$$

Let $S = \operatorname{supp}(\nu)$, assume evaluations are noiseless, and let $M = \sup_{x \in S} f(x)$. Assume that the supremum is approachable under ν, i.e., for all $\delta > 0$,

$$\nu(\{x \in S : f(x) > M - \delta\}) > 0.$$

Proposition 1. *Under the above assumptions,*

$$\max_{0 \le j \le k} f(x_j) \xrightarrow[k \to \infty]{a.s.} M.$$

Proof. At iteration k, the worst-ranked element has normalized rank $r_k = 1$, hence $t_k = 1 - 1^\gamma = 0$, and sampling from this parent yields a draw from ν. The probability of selecting the worst-ranked element is

$$p_k^{\text{worst}} = \frac{e^{-\beta}}{\sum_{i=1}^{n_k} e^{-\beta r_k(x_i)}} \ge \frac{e^{-\beta}}{n_k}.$$

Thus, for any measurable set $A \subseteq S$,

$$\mathbb{P}(x_{k+1} \in A \mid \mathcal{A}_k) \geq \frac{e^{-\beta}}{n_k}\,\nu(A).$$

Since $n_k = n_0 + k$, we have $\sum_{k=0}^{\infty} n_k^{-1} = \infty$, so any set A with $\nu(A) > 0$ is visited infinitely often almost surely.

For any $\delta > 0$, let $A_\delta = \{x \in S : f(x) > M - \delta\}$. By assumption, $\nu(A_\delta) > 0$, so proposals enter A_δ infinitely often almost surely, implying

$$\liminf_{k\to\infty} \max_{0\leq j\leq k} f(x_j) \geq M - \delta.$$

Since $\delta > 0$ is arbitrary and $\max_{j\leq k} f(x_j) \leq M$ for all k, the result follows. □

C THIN, HIGH-DIMENSIONAL TUBE

For an even ambient dimension D, let $c : [0, 1] \to \mathbb{R}^D$ be a smooth centerline defined by

$$\begin{aligned} c_1(s) &= s, \\ c_{2j-1}(s) &= a_j \sin(2\pi j s + \phi_j), \quad j = 2, \ldots, D/2, \\ c_{2j}(s) &= a_j \cos(2\pi j s + \phi_j), \quad j = 1, \ldots, D/2. \end{aligned}$$

where $a_j = 1/j$ and phases $\phi_j \sim \mathrm{Uniform}[0, 2\pi]$ are sampled once and fixed across all methods and runs. Let

$$d(x, \mathcal{C}) = \min_{s\in[0,1]} \|x - c(s)\|_2, \qquad s^\star(x) \in \arg\min_{s\in[0,1]} \|x - c(s)\|_2. \tag{1}$$

The objective rewards both proximity to the tube and progress along it:

$$f(x) = s^\star(x)\left(1 - \frac{d(x, \mathcal{C})}{r}\right), \tag{2}$$

which is positive only inside the tube $d(x, \mathcal{C}) \leq r$, increases toward the endpoint $c(1)$, and penalizes deviation from the centerline continuously. Out-of-tube evaluations return a negative value; we clip reported rewards at zero.

To train the sampler, we sample $D * 1000$ arc lengths $s \sim \mathrm{Beta}(1, 3)$, place points near $c(s)$ with isotropic Gaussian noise, and retain only samples inside the tube. The $\mathrm{Beta}(1, 3)$ distribution concentrates mass near $s = 0$. The prior captures the tube geometry without carrying any information about the objective direction.

D DIFFAIRFOIL

D.1 ARCHITECTURE

DiffAirfoil (Wei et al., 2024) parameterizes airfoil geometry through an autodecoder originally introduced in Wei et al. (2023). Each airfoil is represented by a latent vector $z \in \mathbb{R}^{256}$ and the autodecoder $\mathcal{D}_\phi$ maps z to a set of surface coordinates by learning a deformation field applied to a fixed template airfoil profile. Geometrically valid airfoils therefore correspond to a continuous deformation of the template, which enforces smoothness and prevents degenerate geometries by construction.

The diffusion model originally proposed in Wei et al. (2024) operates entirely in this latent space. In our implementation, we replace the diffusion sampler with a flow matching model (Appendix H), trained on the same latent codes. This substitution is inconsequential for SPARROW: the method requires only access to corruption and refinement operators $\mathcal{N}_t$ and $\mathcal{S}_{t\to1}$, which flow matching provides directly. The autodecoder $\mathcal{D}_\phi$ is used solely for decoding proposals to airfoil coordinates before XFOIL evaluation, and is never updated.

D.2 Training Data

The autodecoder and generative prior are trained on the UIUC airfoil database (Selig, 1996), supplemented with generated NACA profiles, yielding $N = 2{,}648$ geometrically valid airfoils (Table 8). No aerodynamic labels are used at any stage: the prior is purely geometric, encoding the distribution of plausible airfoil shapes without any information about lift, drag, or the objective direction.

D.3 Latent Space Structure

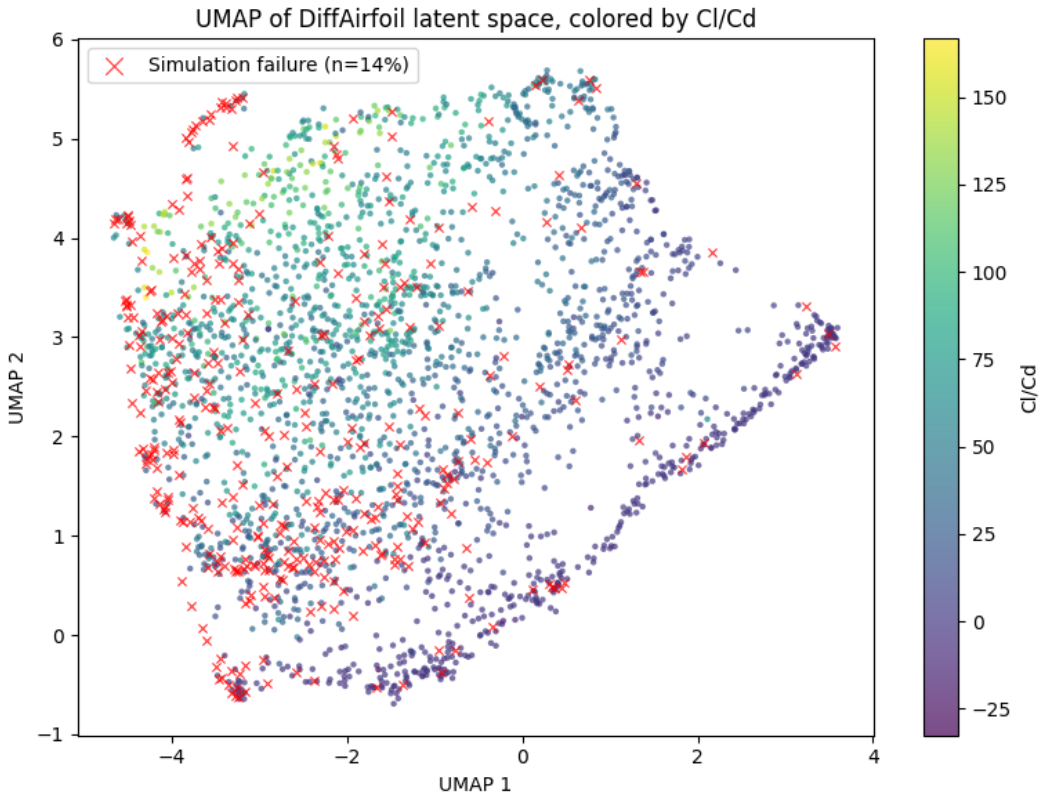

Figure 5: 2-D UMAP of the foil training data in latent space. Colored by XFOIL CL/CD value.

The 256-dimensional latent space exhibits two key properties.

Geometric structure without aerodynamic alignment. Figure 5 shows a UMAP projection of the training latent codes, colored by C_l/C_d evaluated by XFOIL. The distribution of Cl/Cd values is spatially structured: a compact region of the latent space concentrates high-performing airfoils, while the training data mass occupies a broader, aerodynamically mediocre region. The prior therefore encodes geometric validity across the full training distribution, but is not aligned with the aerodynamic objective.

Uniform, objective-independent solver failures. XFOIL is a panel-method solver whose convergence depends on flow conditions and airfoil geometry: turbulence, flow separation, and operation outside the solver's validity range cause frequent, unpredictable failures. Figure 5 shows the C_l/C_d landscape exhibits a smooth objective gradient, while failures appear irregularly throughout the space. Crucially, solver failures are not concentrated near the optimum and do not reflect proximity to a geometric boundary, but constitute pure evaluation noise distributed across the latent space.

Together, these properties define the challenge: the generative prior concentrates away from the optimum, and a significant number of evaluations are wasted regardless of where the search focuses. SPARROW addresses both simultaneously. Rank-based guidance steers proposals toward better-ranked regions. Rank-based invariance makes the optimization signal more robust to missing

feedback, so that failed evaluations do not corrupt the search. Critically, the generative prior itself reduces the baseline failure rate by orienting the search towards neighborhoods of known valid airfoils. Classical methods operating directly in latent space more easily sample regions that exceed XFOIL's convergence domain.

E DESIGN-BENCH

Design-Bench (Trabucco et al., 2022) is a standard offline BBO benchmark: For each task, methods receive a fixed dataset of evaluated candidates and a surrogate oracle, a proxy trained on that dataset, to score proposals. We use it in an online fashion, querying the oracle sequentially as optimizations generate candidates.

E.1 HOPPERCONTROLLER LANDSCAPE

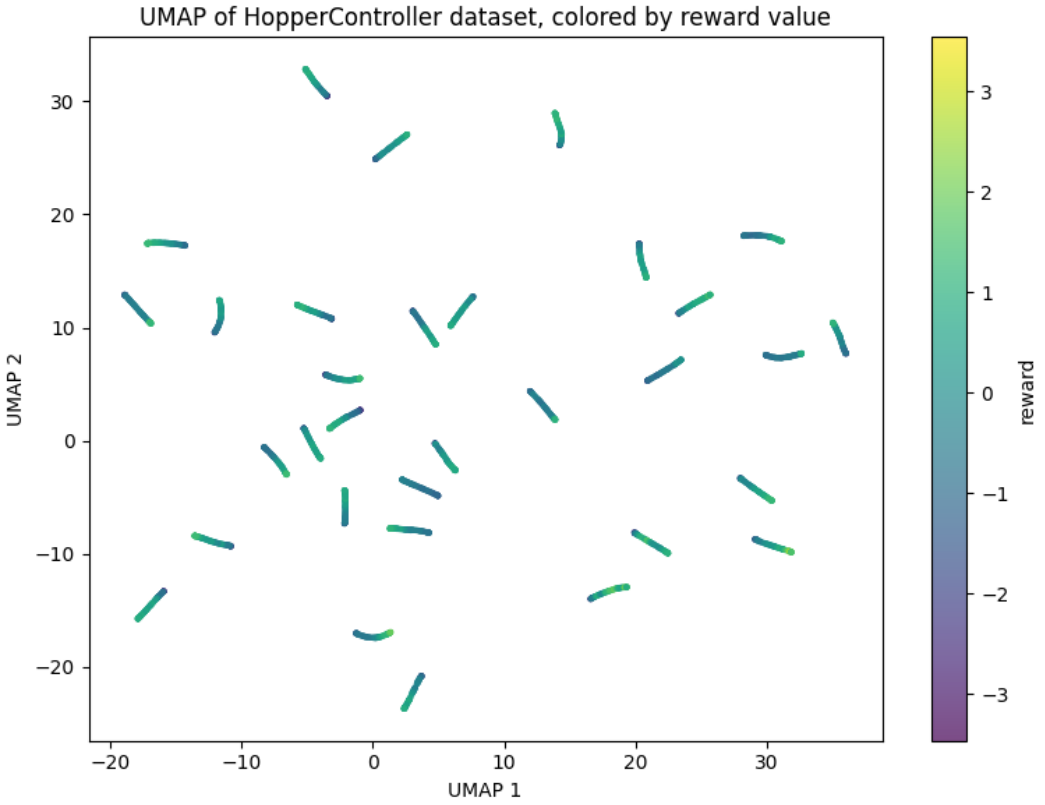

Figure 6: 2-D UMAP of the HopperController dataset in search space. Colored by reward value.

The HopperController search space, shown on Figure 6 exhibits two key properties. First, it is significantly higher-dimensional than the other tasks and experiments, which makes traditional methods struggle. Second, the used oracle, trained on the available dataset, has multiple highly disconnected high-performing regions, that are relatively steep compared to the global structure. Search therefore has to balance local exploitation and global exploration.

E.2 OTHER TASKS

As noted in Trabucco et al. (2022), three of Design-Bench's tasks incorporate a key bias. The AntMorphology, DKittyMorphology, and HopperController datasets were collected using CMA-ES. This biases the results in two ways: First, the oracle is trained on CMA-ES trajectories, which densely cover regions reachable by ambient-space perturbations and are sparse elsewhere, rewarding

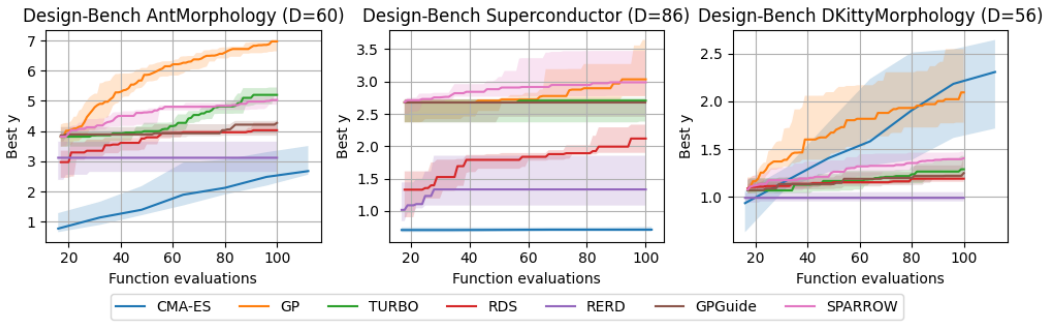

Figure 7: **Objective median value and 25/75 Inter Quartile Range (IQR)** as a function of the number of times the objective function is evaluated.

Table 2: Best objective value found (median, IQR over 10 runs)

	AntMorphology	**DKittyMorphology**	**SuperConductor**
Method	$B = 100$	$B = 100$	$B = 100$
CMA-ES	$2.67_{[2.55,\,3.50]}$	$\mathbf{2.30}_{[1.71,\,2.64]}$	$0.71_{[0.68,\,0.75]}$
GP	$\mathbf{6.96}_{[6.63,\,7.01]}$	$2.09_{[1.77,\,2.54]}$	$\mathbf{3.03}_{[2.77,\,3.64]}$
TuRBO	$5.19_{[5.15,\,5.43]}$	$1.28_{[1.20,\,1.42]}$	$2.70_{[2.37,\,2.74]}$
RDS	$4.02_{[3.93,\,4.27]}$	$1.18_{[1.15,\,1.23]}$	$2.11_{[1.89,\,2.33]}$
SPARROW (ours)	$5.01_{[4.38,\,5.32]}$	$1.40_{[1.28,\,1.47]}$	$2.98_{[2.77,\,3.47]}$

ambient-space methods independently of solution quality. Second, our sampler is trained on the biased dataset, biasing its support towards CMA-ES friendly regions, reducing *SPARROW*'s structural advantage. This bias is severe at moderate dimensionality but negligible at $D = 5126$, where CMA-ES itself collapses and its trajectories offer no meaningful coverage advantage. Superconductor uses real experimental data (Hamidieh, 2018) and is free of collection bias. However, its oracle retains a smoothness bias, with a comparatively low dimensionality $D = 86$.

Results on AntMorphology, DKittyMorphology, and Superconductor, shown in Figure 7 and Table 2 reflect two compounding effects: the tasks' smoothness and relatively low complexity, combined with the benchmark bias, favor GP over *SPARROW*. Disentangling the two effects is difficult, but both make the tasks fall outside the regime *SPARROW* is built to solve. On HopperController in Section 5.5 and the airfoil task in Section 5.6, where the geometry is complex and the feedback is unreliable, *SPARROW* clearly outperforms competing approaches. Additionally, even on simpler smoother tasks, *SPARROW* outperforms the generative baselines, highlighting that this smoothness impacts all generative approaches and is not specific to *SPARROW*. This helps define the setting where generative approaches to BBO make sense: highly structured and complex spaces where traditional search struggles.

F Tuning Procedures

The protocol: Hyperparameter tuning also consumes objective evaluations and is therefore itself costly in the VLB setting. We consequently use a small and equal tuning allowance for all tunable methods rather than exhaustive hyperparameter optimization. For each task and method, we evaluate five hand-chosen candidate configurations, each for one full optimization run with budget $B = 100$. The configuration is chosen considering the result of the previous tuning runs, the known structure of the search space and the known behaviour of the tunable hyperparameters. All candidates within a method–task pair use the same random seed. We select the configuration with the highest terminal best-so-far objective value. For the 64D tube experiment at $B = 1000$, we reuse the configuration selected at $B = 100$ without further tuning. The selected configurations are then used unchanged for the runs reported in the main paper. Tables 3, 4 and 5 report all candidate configurations considered.

The parameters: For CMA-ES, σ_0 controls the initial search scale. In our tables it is expressed in normalized coordinates. For GP, K caps the number of archive points used to fit the surrogate. When

Table 3: Hyperparameter tuning on the 64D tube. Each configuration is evaluated once with $B = 100$ using the same tuning seed. The configuration with the highest final best-so-far objective is selected. Selected configurations are shown in bold.

Method	Configuration	Best f	Method	Configuration	Best f
CMA-ES	$\sigma_0 = 1$	-0.088	GP	$(K, m) = (100, 0.1)$	0.075
	$\sigma_0 = 0.5$	-0.277		$(K, m) = (50, 0.1)$	0.075
	$\sigma_0 = 0.25$	-0.097		$(K, m) = (100, 0.03)$	0.075
	$\sigma_0 = 0.1$	-0.007		$(K, m) = (100, 0.3)$	0.075
	$\sigma_0 = \mathbf{0.05}$	**0.013**		$(K, m) = (20, 0.1)$	0.075
TuRBO	$l = 0.2$	0.075	RERD	$(L, N, K) = (3, 10, 4)$	0.138
	$l = 0.4$	0.075		$(L, N, K) = (\mathbf{2}, \mathbf{10}, \mathbf{3})$	**0.165**
	$l = 0.8$	0.075		$(L, N, K) = (2, 5, 3)$	0.150
	$l = 0.1$	0.075		$(L, N, K) = (3, 5, 2)$	0.139
	$l = 0.05$	0.075		$(L, N, K) = (3, 1, 4)$	0.085
GPGuide	$g = 1$	0.088	SPARROW	$(\beta, \gamma, \lambda) = (\mathbf{50}, \mathbf{1.0}, \mathbf{0.1})$	**0.256**
	$g = 0.5$	0.088		$(\beta, \gamma, \lambda) = (100, 2.0, 0.05)$	0.153
	$g = 0.25$	0.088		$(\beta, \gamma, \lambda) = (25, 0.5, 0.2)$	0.165
	$g = 0.1$	0.178		$(\beta, \gamma, \lambda) = (75, 1.5, 0.1)$	0.235
	$g = \mathbf{0.03}$	**0.186**		$(\beta, \gamma, \lambda) = (40, 0.8, 0.15)$	0.218

Table 4: Hyperparameter tuning on HopperController. Each configuration is evaluated once with $B = 100$ using the same tuning seed. The configuration with the highest final best-so-far objective is selected. Selected configurations are shown in bold.

Method	Configuration	Best f	Method	Configuration	Best f
CMA-ES	$\sigma_0 = 1$	-0.025	GP	$(K, m) = (100, 0.1)$	0.787
	$\sigma_0 = 0.5$	-0.024		$(K, m) = (50, 0.1)$	0.787
	$\sigma_0 = 0.25$	0.030		$(K, m) = (100, 0.03)$	0.787
	$\sigma_0 = 0.1$	0.165		$(K, m) = (100, 0.3)$	0.787
	$\sigma_0 = \mathbf{0.05}$	**0.189**		$(K, m) = (20, 0.1)$	0.787
TuRBO	$l = 0.2$	0.787	RERD	$(L, N, K) = (\mathbf{3}, \mathbf{10}, \mathbf{4})$	**0.812**
	$l = 0.4$	0.787		$(L, N, K) = (2, 10, 3)$	0.812
	$l = 0.8$	0.787		$(L, N, K) = (2, 5, 3)$	0.579
	$l = 1.2$	0.787		$(L, N, K) = (3, 5, 2)$	0.579
	$l = 1.6$	0.787		$(L, N, K) = (3, 1, 4)$	0.018
GPGuide	$g = 1$	0.878	SPARROW	$(\beta, \gamma, \lambda) = (50, 1.0, 0.1)$	0.847
	$g = 0.5$	0.878		$(\beta, \gamma, \lambda) = (10, 1.0, 0.1)$	0.854
	$g = 0.25$	0.878		$(\beta, \gamma, \lambda) = (5, 1.0, 0.1)$	0.942
	$g = 0.1$	0.878		$(\beta, \gamma, \lambda) = (\mathbf{5}, \mathbf{0.1}, \mathbf{0.1})$	**0.953**
	$g = 0.03$	0.878		$(\beta, \gamma, \lambda) = (5, 0.1, 0.01)$	0.944

the archive exceeds K, we retain 80% high-scoring points and 20% randomly sampled points. The margin m expands the acquisition-search domain beyond the coordinate-wise range of the evaluated archive. For TuRBO, ℓ sets the initial trust-region side length, which is subsequently adapted during optimization. For RERD, N is the number of parallel samples, L is the duplication factor used during reward-guided refinement, and K is the number of refinement levels. Together they determine the number of objective calls in one complete refinement cycle, $Q = N((K-1)L+1)$. For GPGuide, g scales the surrogate guidance applied along the generative trajectory. RDS has no tunable optimization hyperparameters.

An analysis of the chosen configurations: For HopperController, the latent space contains disconnected feasible regions, making this the hardest structural setting among our tasks. Very low $\beta = 5$ encourages broad exploration across disconnected components rather than over-exploiting a single region. Very low $\gamma = 0.1$ delegates most structural burden to the generative prior, whose corruption–refinement operators can bridge disconnected regions in ways that rank signals alone cannot. $\lambda = 0.1$ was set by qualitative transfer from the tube experiment.

Table 5: Hyperparameter tuning on the airfoil task. Each configuration is evaluated once with $B = 100$ using the same tuning seed. The configuration with the highest final best-so-far objective is selected. Selected configurations are shown in bold.

Method	Configuration	Best f	Method	Configuration	Best f
CMA-ES	$\sigma_0 = 1$	144.526	GP	$(K, m) = (\mathbf{100}, \mathbf{0.1})$	$\mathbf{189.475}$
	$\sigma_0 = 0.5$	102.925		$(K, m) = (50, 0.1)$	176.812
	$\sigma_0 = 2$	142.276		$(K, m) = (100, 0.03)$	177.273
	$\sigma_0 = \mathbf{3}$	$\mathbf{151.551}$		$(K, m) = (100, 0.3)$	186.037
	$\sigma_0 = 5$	143.219		$(K, m) = (20, 0.1)$	186.054
TuRBO	$l = \mathbf{0.2}$	$\mathbf{144.343}$	RERD	$(L, N, K) = (3, 10, 4)$	99.790
	$l = 0.4$	132.844		$(L, N, K) = (\mathbf{2}, \mathbf{10}, \mathbf{3})$	$\mathbf{104.559}$
	$l = 0.8$	117.389		$(L, N, K) = (2, 5, 3)$	61.395
	$l = 0.1$	134.741		$(L, N, K) = (3, 5, 2)$	63.305
	$l = 0.05$	120.339		$(L, N, K) = (3, 1, 4)$	57.570
GPGuide	$g = \mathbf{1}$	$\mathbf{146.122}$	SPARROW	$(\beta, \gamma, \lambda) = (50, 1.0, 0.1)$	119.115
	$g = 0.5$	139.138		$(\beta, \gamma, \lambda) = (20, 0.75, 0.5)$	153.264
	$g = 0.25$	131.913		$(\beta, \gamma, \lambda) = (10, 0.5, 1.0)$	170.440
	$g = 0.1$	127.323		$(\beta, \gamma, \lambda) = (\mathbf{10}, \mathbf{0.5}, \mathbf{2.0})$	$\mathbf{175.284}$
	$g = 0.03$	123.056		$(\beta, \gamma, \lambda) = (5, 0.25, 2.0)$	162.706

For the airfoil experiment, lower $\beta = 10$ reflects the importance of diversity in a multimodal aerodynamic landscape. Lower $\gamma = 0.5$ increases reliance on the structural prior, which is useful in the failure-prone aerodynamic landscape. Larger $\lambda = 2$ accounts for the greater characteristic distances in the DiffAirfoil latent space.

G Raw Ablation Results

G.1 Tube dimensionality

Table 6: Dimensionality ablation on the thin-tube benchmark at $B = 100$. Values are median best objective with IQR over 10 runs.

Method	$\mathbf{D = 8}$	$\mathbf{D = 16}$	$\mathbf{D = 32}$	$\mathbf{D = 64}$	$\mathbf{D = 128}$
CMA-ES	$0.10_{[.03,.35]}$	$0.17_{[.05,.28]}$	$0.01_{[.00,.04]}$	$0.00_{[.00,.00]}$	$0.00_{[.00,.00]}$
GP	$0.09_{[.08,.12]}$	$0.08_{[.07,.09]}$	$0.08_{[.05,.10]}$	$0.08_{[.07,.09]}$	$0.07_{[.07,.08]}$
TuRBO	$0.15_{[.11,.20]}$	$0.10_{[.09,.16]}$	$0.08_{[.06,.10]}$	$0.08_{[.07,.09]}$	$0.07_{[.07,.08]}$
RDS	$0.33_{[.31,.41]}$	$0.28_{[.24,.32]}$	$0.21_{[.18,.23]}$	$0.16_{[.14,.17]}$	$0.16_{[.14,.17]}$
RERD	$0.26_{[.24,.26]}$	$0.20_{[.18,.24]}$	$0.18_{[.17,.21]}$	$0.16_{[.13,.19]}$	$0.13_{[.12,.14]}$
GPGuide	$0.29_{[.26,.38]}$	$0.24_{[.21,.25]}$	$0.20_{[.16,.25]}$	$0.13_{[.09,.14]}$	$0.11_{[.10,.11]}$
SPARROW (ours)	$\mathbf{0.46}_{[.39,.54]}$	$\mathbf{0.32}_{[.28,.36]}$	$\mathbf{0.32}_{[.29,.35]}$	$\mathbf{0.30}_{[.28,.32]}$	$\mathbf{0.27}_{[.22,.29]}$

G.2 HYPERPARAMETERS

Table 7: Hyperparameter sensitivity on the 64D thin-tube benchmark at $B = 100$. Each parameter is varied independently while the other two are fixed at $\beta = 50$, $\gamma = 1$, and $\lambda = 0.1$, shown in bold. Values report median best objective and IQR.

β		γ		λ	
Value	Best f	Value	Best f	Value	Best f
10	$0.25_{[.19,.27]}$	0.5	$0.19_{[.18,.23]}$	0	$0.28_{[.24,.34]}$
20	$0.26_{[.23,.27]}$	0.75	$0.27_{[.23,.30]}$	0.01	$0.28_{[.26,.32]}$
50	$\mathbf{0.30}_{[.28,.32]}$	**1**	$\mathbf{0.30}_{[.28,.32]}$	0.03	$0.29_{[.22,.37]}$
100	$0.25_{[.18,.34]}$	1.25	$0.32_{[.27,.34]}$	**0.1**	$\mathbf{0.30}_{[.28,.32]}$
200	$0.25_{[.23,.28]}$	1.5	$0.24_{[.19,.27]}$	0.3	$0.28_{[.25,.30]}$
				1	$0.25_{[.22,.29]}$

H EXPERIMENTAL IMPLEMENTATION

H.1 GENERATIVE MODEL ARCHITECTURE

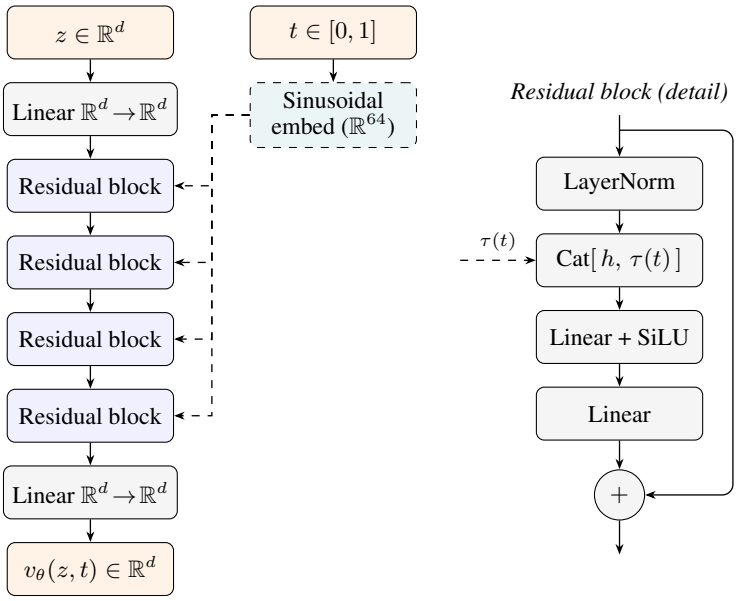

Figure 8: Flow matching model architecture, shared across all experiments. The ambient dimension d varies per task (See Appendix H.3). The noise level t is encoded by a 64-dimensional sinusoidal embedding $\tau(t)$ and injected into each residual block by concatenation with the hidden state. Each block applies LayerNorm, two linear layers with SiLU activation, and a residual connection. The network maps $(z, t) \in \mathbb{R}^d \times [0, 1]$ to a velocity $v_\theta(z, t) \in \mathbb{R}^d$.

All generative priors share a common architecture (Figure 8): a four-block residual network with sinusoidal time conditioning, trained with a flow matching objective. The hidden dimension equals the ambient dimension d in all experiments.

H.2 FLOW MATCHING TRAINING

All models are trained with a linear interpolation bridge using the procedure described in Algorithm 2. Noise samples are drawn from $\mathcal{N}(\mu_{\mathcal{D}}, \sigma^2_{\mathcal{D}} I)$, fitted from the training dataset, and are reordered via minibatch optimal transport (Hungarian algorithm on pairwise squared ℓ_2 cost) to reduce training variance without altering marginal distributions. Time is sampled from $\mathrm{Beta}(0.5, 0.5)$, which upweights the endpoints of the trajectory.

Algorithm 2 Flow Matching Training Step

Require: Training set $\mathcal{D}$, model v_θ, batch size B

1: Sample minibatch $\{z_1^{(i)}\}_{i=1}^{B} \sim \mathcal{D}$
2: Sample noise $z_0^{(i)} \sim \mathcal{N}(\mu_{\mathcal{D}},\, \sigma_{\mathcal{D}}^2 I)$ for each i
3: Reorder $\{z_0^{(i)}\}$ to minimise $\sum_i \|z_0^{(i)} - z_1^{(i)}\|^2$ ▷ minibatch OT
4: Sample $t^{(i)} \sim \mathrm{Beta}(0.5, 0.5)$ for each i
5: $z_t^{(i)} \leftarrow (1 - t^{(i)})\, z_0^{(i)} + t^{(i)}\, z_1^{(i)}$
6: $v^{\star(i)} \leftarrow z_1^{(i)} - z_0^{(i)}$
7: Update θ on $\mathcal{L} = \frac{1}{B} \sum_i \|v_\theta(z_t^{(i)},\, t^{(i)}) - v^{\star(i)}\|^2$

All models are trained for 100'001 steps with Adam (lr $= 10^{-4}$), cosine annealing with warm restarts ($T_0 = 10'000$, $T_{\mathrm{mult}} = 2$, $\eta_{\min} = 10^{-7}$), and mixed-precision training. Batch size is 512.

At sampling time, initial noise is drawn from $\mathcal{N}(\mu_{\mathcal{D}}, \sigma_{\mathcal{D}}^2 I)$, consistent with training. Integration uses RK4 with $\max(5, \lfloor (1 - t_{\mathrm{start}}) \times 100 \rfloor)$ steps.

H.3 Training Datasets

Table 8: Training set sizes and dimensionality for each generative prior.

Task	Source	D	N
Thin tube	Synthetic	D	$1'000 \times D$
Airfoil	UIUC database + NACA profiles	256	$2'648$
AntMorphology	Design-Bench	60	$10'004$
DKittyMorphology	Design-Bench	56	$10'004$
HopperController	Design-Bench	5126	$3'200$
Superconductor	Design-Bench	86	$17'014$

Dataset sizes for each task are given in Table 8. The thin tube prior is explained in Section 5.4 and Appendix C. The airfoil prior is described in Appendix D. All Design-Bench priors are trained directly on the provided training sets.

H.4 Baseline Implementations

CMA-ES. We use the pycma (Hansen, 2016) library with default parameters. x_0 is initialized with a random training data point. σ_0 is chosen by the tuning.

GP. We use the BoTorch (Balandat et al., 2020) implementation of vanilla Bayesian optimisation with a Matérn-5/2 kernel and Expected Improvement acquisition, following Hvarfner et al. (2024) with their dimension-scaled length-scale priors.

TuRBO. We implement TuRBO following the official BoTorch tutorial, with default parameters and the same archive cap of $K = 200$ points (80% top-ranked, 20% random).

In all methods, failed evaluations (e.g. solver crashes in the airfoil task) are excluded from the archives/populations but counted against the budget B.

RERD. We follow Uehara et al. (2025) with the hyperparameters and budget accounting specified in Appendix F.

GPGuide. We guide the generative trajectory using the gradient of a GP's expected improvement, evaluated at an estimate of the final candidate and added to the sampler's velocity with bounded strength g.

H.5 XFOIL EVALUATION SETTINGS.

All aerodynamic evaluations use XFOIL (Drela, 1989) at a fixed angle of attack $\alpha = 0^\circ$ and Reynolds number $Re = 10^6$, applied during both the latent space analysis of Appendix D and the optimization experiments of Section 5.6.

H.6 COMPUTE

All experiments were run on a single consumer GPU (NVIDIA RTX 4060, 8GB). Generative model training took at most a few hours per task. Optimization runs completed in at most a few hours. Consistent with our low-budget framing, we consider generative model inference as negligible relative to the cost of objective evaluation: the dominant expense in all experiments is querying f, not running the sampler.

H.7 LICENSES

Design-Bench (Trabucco et al., 2022) and BoTorch (Balandat et al., 2020) are MIT licensed. pycma (Hansen et al., 2025) is BSD licensed. XFOIL (Drela, 1989) is distributed under the GNU GPL. The airfoil coordinates of the UIUC Airfoil Database (Selig, 1996) are publicly available with no stated license. All are used in accordance with their respective licenses.