Conservative Pareto Set Amortization for Offline Multi-Objective Optimization

Ji Cheng
City University of Hong Kong
jicheng9617@gmail.com

Abstract

Offline multi-objective optimization (MOO) seeks high-quality trade-off designs from a static dataset of previously evaluated candidates. Existing approaches typically rely on guided diffusion or flow sampling to generate an unordered candidate pool. We instead formulate offline MOO as deterministic Pareto set amortization. Our method, Offline Pareto Set Learning (OffPSL), learns a preference-conditioned map $h_\pi : \Delta^{m-1} \to \mathcal{X}$ that returns a design for each requested trade-off direction in a single forward pass. In the absence of preference–solution labels, OffPSL updates h_π using two offline-only signals: a pessimistic surrogate scalarization and a multi-noise support regularizer parameterized by a frozen single-step denoising autoencoder. We complement the method with a conditional analysis that decomposes scalarized regret into surrogate error, surrogate suboptimality, and a coverage gap, and establishes small-noise support-distance consistency of the denoising residual under unique projection and controlled denoiser error. On standard offline MOO benchmarks, OffPSL significantly outperforms guided generative and Bayesian baselines on hypervolume and matches the strongest methods on IGD, while reducing inference cost by two orders of magnitude.

1 Introduction

Many scientific and engineering design problems require optimizing several conflicting objectives from limited historical data. Examples include molecular and protein design, hardware and architecture search, and mechanical or aerodynamic design, where objectives such as performance, cost, robustness, safety, and manufacturability cannot usually be optimized by unrestricted online experimentation. In the offline multi-objective optimization (offline MOO) setting, the learner is given a fixed dataset

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N, \qquad \boldsymbol{y}_i = f(\boldsymbol{x}_i) \in \mathbb{R}^m, \tag{1}$$

and must return high-quality trade-off designs without further objective evaluations [21, 23]. Its central difficulty is surrogate exploitation: the learned objective model is least reliable in regions that gradient-based optimization tends to explore under distribution shift [20].

Recent offline MOO methods typically use learned generative models as finite candidate-set generators. Diffusion- and flow-based optimizers first learn a distribution over offline designs and then guide generation toward Pareto-preferred or surrogate-predicted high-value regions [9, 1, 26, 19]. This paradigm can produce strong finite candidate sets, but it gives an indirect interface to the Pareto set: the output is an unordered pool, and a preference vector $\boldsymbol{\lambda} \in \Delta^{m-1}$ specified by a decision maker is not necessarily associated with a unique returned design.

We instead study deterministic Pareto set amortization for offline MOO. The goal is to learn a preference-conditioned map $h_\pi : \Delta^{m-1} \to \mathcal{X}, \boldsymbol{x} = h_\pi(\boldsymbol{\lambda})$, which returns one design for each requested trade-off direction. This interface is natural for interactive decision making because users

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

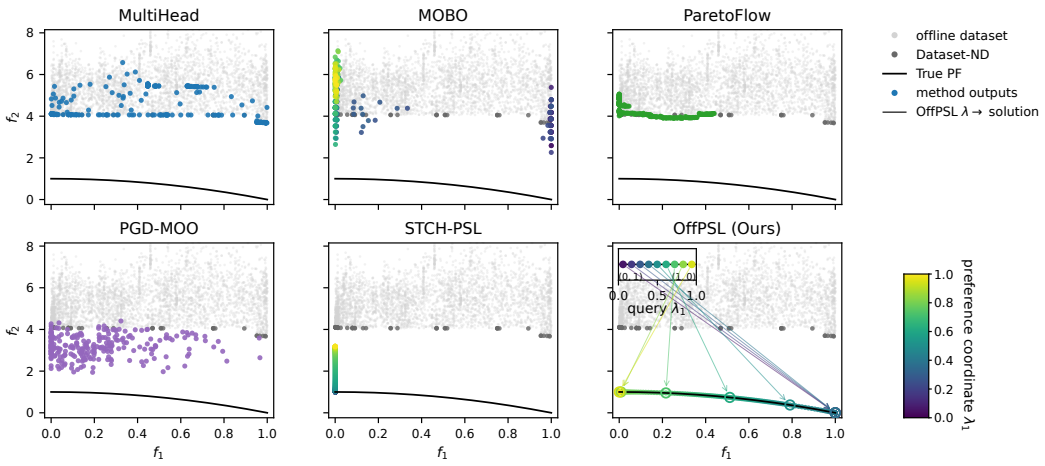

Figure 1: **ZDT2 objective-space comparison.** Gray points show the offline data and its nondominated subset, black shows the true Pareto front, and colored points show oracle-evaluated method outputs. Colors denote queried λ_1 for preference-conditioned methods. OffPSL shows a more coherent preference ordering and broader coverage on this example.

can query, interpolate, or sweep preferences, and candidate generation reduces to one forward pass per preference. Pareto set learning has been studied in online or model-assisted settings, including Pareto hypernetworks, expensive multi-objective Bayesian optimization, and neural combinatorial optimization [16, 12, 11]. The offline-only setting is substantially harder: the dataset contains no preference–solution labels $(\boldsymbol{\lambda}, \boldsymbol{x}^\star_\lambda)$, and the only available training signal comes from learned models whose errors can be amplified by optimization.

OffPSL addresses these challenges by combining objective-space pessimism with design-space support regularization. It trains h_π using a pessimistic surrogate scalarization alongside a frozen single-step denoising autoencoder that provides a local support signal during training. At test time, candidate generation reduces to evaluating $h_\pi(\boldsymbol{\lambda})$ in a single forward pass per reference preference.

Our contributions are fourfold.

- We formulate offline MOO as deterministic Pareto set amortization and evaluate it with both set-quality metrics and interface-aware diagnostics.
- We introduce OffPSL, an offline-only training objective that combines pessimistic smooth Tchebycheff scalarization with multi-noise denoising support regularization. This supplies the training signal for Pareto set learning from fixed design–objective data.
- We establish a finite-data coverage impossibility result, decompose conditional scalarized regret into surrogate error, surrogate suboptimality, and a comparison-region gap, and prove small-noise support-distance consistency under unique projection and controlled denoiser error.
- We evaluate OffPSL under a fixed candidate budget against surrogate-based offline MOO, multi-objective Bayesian optimization (MOBO), guided generative offline MOO including PCD, and prior PSL baselines, reporting absolute normalized-regret statistics with significance tests in addition to ranks, together with preference-consistency, duplicate-rate, support, and clipping diagnostics.

2 Preliminaries and Problem Setting

2.1 Multi-objective optimization

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the design space and $f : \mathcal{X} \to \mathbb{R}^m$ a vector-valued objective, stated in minimization form. A design $\boldsymbol{x}_1$ Pareto-dominates $\boldsymbol{x}_2$ ($\boldsymbol{x}_1 \prec \boldsymbol{x}_2$) if $f_r(\boldsymbol{x}_1) \leq f_r(\boldsymbol{x}_2)$ for all r, with strict inequality for some r; the non-dominated designs form the *Pareto set* $\mathcal{P}_s$ and their image the *Pareto front* $\mathcal{P}_f$ [15].

Scalarization converts a vector objective into a scalar objective indexed by a preference vector $\boldsymbol{\lambda} \in \Delta^{m-1}$. Linear scalarization is simple but may fail to recover non-convex regions of the Pareto front [2]. Tchebycheff scalarization can represent broader Pareto-front geometries and is commonly used in decomposition-based MOO:

$$g_{\mathrm{tch}}(\boldsymbol{x}; \boldsymbol{\lambda}, \boldsymbol{z}) = \max_{r \in [m]} \lambda_r |f_r(\boldsymbol{x}) - z_r|, \tag{2}$$

where $\boldsymbol{z}$ is an ideal point. In gradient-based learning, the maximum can be replaced by the smooth log-sum-exp approximation [13]. When online objective evaluations are available, population-based algorithms such as NSGA-II [7] and MOEA/D [29], and Bayesian-optimization methods based on expected hypervolume improvement [5, 6], provide classical finite-set approximations to the Pareto front.

2.2 Offline model-based optimization

Offline model-based optimization seeks high-performing designs from a static dataset of evaluated examples [21], typically by training a surrogate $\hat{f}_\theta$ on $\mathcal{D}$ and optimizing through it. Its main failure mode is surrogate exploitation: optimization drifts off the data distribution, where the surrogate may overestimate performance [20]. Conservative methods therefore regularize the surrogate, the optimized designs, or the optimization trajectory, trading predicted improvement against support [20, 18, 25, 24, 4].

2.3 Pareto set learning

Pareto set learning replaces a finite approximation set with a continuous preference-conditioned model $h_\pi : \Delta^{m-1} \to \mathcal{X}$. Given a scalarizer g and access to an objective model $\hat{f}$, a standard PSL training objective has the form

$$\min_{\pi} \ \mathbb{E}_{\boldsymbol{\lambda} \sim p(\Delta^{m-1})} \big[g(\hat{f}(h_\pi(\boldsymbol{\lambda})); \boldsymbol{\lambda}) \big]. \tag{3}$$

After training, h_π produces a design for any preference in a single forward pass. Existing PSL methods have been developed for multi-task learning, expensive multi-objective Bayesian optimization, neural multi-objective combinatorial optimization, hypervolume-based geometric formulations, and smooth-Tchebycheff or Stein-variational variants [16, 12, 11, 30, 13, 22, 17]. These settings typically assume an oracle, an actively refined surrogate, or task-specific online feedback. Offline MOO instead learns from a fixed dataset.

2.4 Offline Pareto set amortization

We consider a fixed-dataset offline MOO problem. The learner receives a dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$ with $\boldsymbol{y}_i = f(\boldsymbol{x}_i)$ and cannot query f during surrogate training, support-model training, Pareto-set training, checkpoint selection, hyperparameter selection, or candidate generation. For a test-time budget K, the learner must submit $X_K = \{h_\pi(\boldsymbol{\lambda}_k)\}_{k=1}^K, \boldsymbol{\lambda}_k \in \Delta^{m-1}$, where the reference preferences are fixed independently of the oracle. The submitted set is evaluated only after candidate generation by querying the true objective on X_K and computing metrics such as hypervolume and, when available, inverted generational distance.

The target is a family of trade-off designs indexed by $\boldsymbol{\lambda}$. Shared parameters couple the solutions across preferences, making both set quality and the coherence of the preference map relevant evaluation criteria. For fixed reference directions, OffPSL generates the submitted set through exactly K deterministic forward passes of h_π. Its training objective combines conservative scalarization with a support regularizer learned from the offline designs.

3 Related Work

Offline model-based optimization. Offline MBO optimizes an unknown black-box objective from a static dataset [21]; its central issue is distribution shift, since surrogate optimization tends to leave the data support, where the surrogate can be confidently wrong [20]. Conservative Objective Models penalize overoptimistic predictions on optimized designs [20], and related methods use invariant

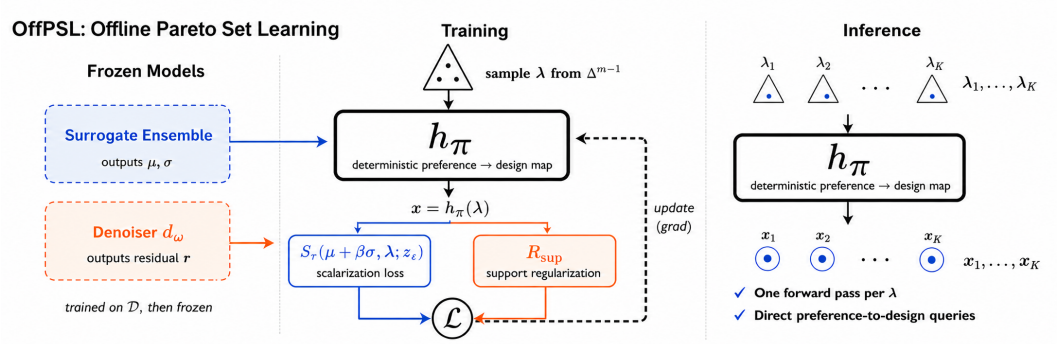

Figure 2: **Overview of OffPSL.** A surrogate ensemble provides pessimistic smooth Tchebycheff scalarization $S_\tau(\mu + \beta\sigma, \boldsymbol{\lambda}; \boldsymbol{z}_\epsilon)$, and a frozen denoising model provides multi-noise support regularization. These training signals update the preference-conditioned map h_π. At inference, each reference direction $\boldsymbol{\lambda}$ yields one design $\boldsymbol{x} = h_\pi(\boldsymbol{\lambda})$ through a deterministic forward pass.

representations, model adaptation, co-teaching, or mentoring to reduce off-support exploitation [18, 24, 25, 4]. A complementary line learns inverse or generative maps from target values to designs, including model inversion networks, bidirectional learning, generative pretraining, and diffusion-based black-box optimization [10, 3, 14, 9]. These motivate conservative training signals, but target a single scalar objective.

Offline multi-objective optimization. Offline MOO extends static-dataset optimization to vector-valued objectives. Recent benchmarks emphasize that offline MOO methods should be evaluated not only by the quality of the best returned design, but also by coverage of the trade-off surface under a fixed candidate budget [23]. Existing approaches include adaptations of single-objective offline MBO with multi-head or multiple surrogate models, offline variants of MOBO acquisition optimization, and guided generative methods. These methods can produce finite candidate sets, but they do not directly learn a continuous map from preferences to designs.

Generative modeling for offline optimization. Diffusion and flow models have become prominent tools for offline optimization. In single-objective settings, diffusion models can generate or edit designs toward high predicted objective values [9, 28, 27, 31]. In offline MOO, PGD-MOO guides diffusion sampling using preference and dominance signals, ParetoFlow uses flow matching with predictor guidance, and PCD conditions a diffusion model on Pareto-reweighted targets to generate candidate sets [1, 26, 19]; we compare against all three under a shared protocol in Section 5. These methods treat offline MOO as guided candidate generation. OffPSL combines a frozen denoising support prior during training with a deterministic Pareto-set map for candidate generation.

Pareto set learning. Pareto set learning amortizes the Pareto set as a neural map conditioned on a preference vector. Pareto hypernetworks introduced this interface for multi-task learning [16], and PSL-MOBO extended it to expensive multi-objective Bayesian optimization [12]. Subsequent work studies PSL for neural combinatorial optimization, hypervolume maximization, Pareto manifold learning, multi-sample hypernetworks, and smooth Tchebycheff scalarization [11, 30, 8, 13, 22]. These methods share the preference-queryable interface adopted here, but they rely on oracle access, online surrogate refinement, or task-specific feedback. The closest methods to OffPSL are PGD-MOO and ParetoFlow on the offline MOO side, and STCH-PSL and PSL-MOBO on the amortized PSL side. PGD-MOO and ParetoFlow satisfy the offline restriction but return finite generated pools. STCH-PSL and PSL-MOBO provide a preference-queryable map but assume oracle or actively refined surrogate information. OffPSL combines the offline restriction of the former with the preference-queryable interface of the latter.

Positioning of our work. OffPSL combines fixed-dataset learning with a directly queryable Pareto-set map. It shares the offline setting of PGD-MOO, ParetoFlow, and PCD [1, 26, 19], and the preference-conditioned interface of PSL-MOBO and STCH-PSL [12, 13]. Pessimistic surrogate

scalarization and a frozen denoising prior provide its training signals. This conservative offline amortization interface complements pool-based generative optimizers by mapping each requested trade-off direction directly to a design.

4 Method

OffPSL learns a deterministic map $\widetilde{h}_\pi : \Delta^{m-1} \to \mathbb{R}^d$ from offline data only. Since $\mathcal{D}$ contains no labels of the form $(\boldsymbol{\lambda}, \boldsymbol{x}^\star_\lambda)$, the map is trained by differentiating through two frozen offline-trained models: a conservative objective surrogate and a denoising support model. All neural components operate on normalized representations derived from the offline dataset. The Pareto-set map and denoising support model use the normalized design space, while the surrogate ensemble predicts dataset min–max normalized objectives. The surrogate ensemble and denoising model are trained once on the offline dataset and then frozen; during Pareto-set training, gradients through these frozen models are used only to update $\widetilde{h}_\pi$.

4.1 Surrogate-aligned objective space

All surrogate targets are normalized using only offline dataset statistics,

$$\boldsymbol{y}_i^{\mathrm{mm}} = \frac{\boldsymbol{y}_i - \boldsymbol{y}^{\mathcal{D}}_{\min}}{\boldsymbol{y}^{\mathcal{D}}_{\max} - \boldsymbol{y}^{\mathcal{D}}_{\min}}. \tag{4}$$

We train an ensemble $\{g_{\theta_j}\}_{j=1}^J$ on $(\boldsymbol{x}_i, \boldsymbol{y}_i^{\mathrm{mm}})$. For a generated design $\boldsymbol{x}$, let $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ denote the coordinate-wise ensemble mean and standard deviation. For minimization, OffPSL uses the pessimistic prediction

$$\hat{\boldsymbol{y}}^{\mathrm{cons}}(\boldsymbol{x}) = \mu(\boldsymbol{x}) + \beta\sigma(\boldsymbol{x}), \tag{5}$$

so uncertainty is retained separately for each objective before scalarization. We adopt this coordinate-wise formulation as a simple default. Factorial ablations (Appendix I) show comparable performance when scalarization precedes uncertainty penalization across the five evaluated tasks, supporting robustness to this ordering choice.

Given a preference $\boldsymbol{\lambda}$, we optimize a smooth Tchebycheff scalarizer in the same normalized surrogate space:

$$S_\tau(\hat{\boldsymbol{y}}, \boldsymbol{\lambda}; \boldsymbol{z}) = \tau \log \sum_{r=1}^{m} \exp\left(\frac{\lambda_r |\hat{y}_r - z_r|}{\tau}\right). \tag{6}$$

A subtle offline issue is the ideal point. If $\boldsymbol{z}$ is set to the best observed dataset value, predictions that moderately improve beyond the dataset minimum can be penalized again by the absolute value. We therefore use an outside ideal point

$$\boldsymbol{z}_\epsilon = -\epsilon\mathbf{1}, \qquad \epsilon > 0, \tag{7}$$

computed in dataset min–max space. This dataset-based offset reduces the spurious penalty associated with moderate improvements beyond the observed minimum.

4.2 Multi-noise denoising support regularization

The conservative surrogate penalizes objective-space uncertainty, while a single-step denoising autoencoder d_ω provides a local support regularizer. We train the autoencoder across multiple noise scales on offline designs:

$$\mathcal{L}_{\mathrm{den}}(\omega) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_x,\, s,\, \xi}\left[\|d_\omega(\boldsymbol{x} + s\xi, s) - \boldsymbol{x}\|_2^2\right], \qquad \xi \sim \mathcal{N}(0, I). \tag{8}$$

During Pareto-set training, d_ω remains frozen. For each generated design $\boldsymbol{x} = \widetilde{h}_\pi(\boldsymbol{\lambda})$, we draw P perturbations and define

$$r_p(\boldsymbol{x}) = \boldsymbol{x} - d_\omega(\boldsymbol{x} + s_p\xi_p, s_p). \tag{9}$$

The residual support penalty is

$$R_{\mathrm{sup}}(\boldsymbol{x}) = \frac{1}{P}\sum_{p=1}^{P} \|r_p(\boldsymbol{x})\|_2^2. \tag{10}$$

We also penalize instability of the denoising residuals across perturbations:

$$R_{\mathrm{cov}}(\boldsymbol{x}) = \frac{1}{d}\,\mathrm{tr}\left(\frac{1}{P-1}\sum_{p=1}^{P}(r_p(\boldsymbol{x})-\bar{r}(\boldsymbol{x}))(r_p(\boldsymbol{x})-\bar{r}(\boldsymbol{x}))^\top\right), \qquad \bar{r}(\boldsymbol{x}) = \frac{1}{P}\sum_{p=1}^{P} r_p(\boldsymbol{x}). \tag{11}$$

Large or unstable residuals indicate that noisy neighborhoods around $\boldsymbol{x}$ are not denoised back consistently, which empirically marks regions with weak support under the offline design distribution and higher risk of surrogate exploitation.

4.3 Pareto-set objective

For a mini-batch of preferences $\{\boldsymbol{\lambda}_i\}_{i=1}^{B}$, OffPSL minimizes

$$\begin{aligned}\mathcal{L}(\pi) = &\frac{1}{B}\sum_{i=1}^{B} S_\tau\Big(\hat{\boldsymbol{y}}^{\mathrm{cons}}(\widetilde{h}_\pi(\boldsymbol{\lambda}_i)), \boldsymbol{\lambda}_i; \boldsymbol{z}_\epsilon\Big) \\ &+ \alpha_{\mathrm{sup}}\frac{1}{B}\sum_{i=1}^{B} R_{\mathrm{sup}}(\widetilde{h}_\pi(\boldsymbol{\lambda}_i)) + \alpha_{\mathrm{cov}}\frac{1}{B}\sum_{i=1}^{B} R_{\mathrm{cov}}(\widetilde{h}_\pi(\boldsymbol{\lambda}_i)) + \alpha_{\mathrm{div}} R_{\mathrm{div}}.\end{aligned} \tag{12}$$

The optional diversity term discourages preference collapse:

$$R_{\mathrm{div}} = \frac{2}{B(B-1)}\sum_{i<j}\exp\left(-\frac{\|\boldsymbol{u}_i-\boldsymbol{u}_j\|_2^2}{\eta}\right), \tag{13}$$

where $\boldsymbol{u}_i$ is the conservative objective-space representation $\hat{\boldsymbol{y}}^{\mathrm{cons}}(\widetilde{h}_\pi(\boldsymbol{\lambda}_i))$ and η is an adaptive bandwidth. This repulsive term promotes spread across distinct trade-off regions. Together, the conservative Tchebycheff loss drives preference-aligned improvement, the denoising residuals promote local support stability, and the diversity regularizer discourages preference collapse.

4.4 Inference

For each preference, one deterministic network pass produces a raw design $\widetilde{h}_\pi(\lambda)$. The evaluated map is $h_\pi^{\mathrm{feas}}(\lambda) = \Pi_{\mathcal{X}}(\widetilde{h}_\pi(\lambda))$, where $\Pi_{\mathcal{X}}$ is the legal-box projection in Appendix C. Thus K preferences require K forward passes. The notation $h_\pi : \Delta^{m-1} \to \mathcal{X}$ denotes this feasible interface; training losses are computed on raw network designs. For the bounded-output variant, the projection is the identity.

4.5 Conditional Analysis of Offline Amortization

Three results characterize the setting; Appendix F gives complete assumptions and proofs.

Proposition 1 (Finite-data coverage impossibility; summary)**.** *On a compact design domain with nonempty interior, fix a finite dataset compatible with a continuous objective. Any dataset-only algorithm returning a nonempty set of at most $K < \infty$ candidates has arbitrarily large expected IGD on some compatible continuous objective with a singleton Pareto front, when the function class has no common amplitude or regularity bound.*

Proposition 2 (Conditional scalarized regret; summary)**.** *Let $\mathcal{X}$ and $S \subseteq \mathcal{X}$ be nonempty compact sets, with continuous f on $\mathcal{X}$ and $\hat{f}$ on S. Fix a common reference z, $\tau > 0$, and $\lambda \in \Delta^{m-1}$. If the surrogate error on S is at most ε_N and the final output $x_h = h_\pi^{\mathrm{feas}}(\lambda) \in S$, then*

$$T_z(f(x_h);\lambda) - \min_{x\in\mathcal{X}} T_z(f(x);\lambda) \le 2\varepsilon_N + \eta_\lambda + \tau\log m + \Delta_S(\lambda). \tag{14}$$

Here $T_z(y;\lambda) = \max_r \lambda_r|y_r - z_r|$, η_λ is the surrogate smooth-scalarization gap at x_h, and Δ_S is the cost of restricting the true scalarized optimum to S (Appendix F).

The gap includes optimization, representation, and regularization effects; all terms can vary with the data and training. Support regularization encourages proximity to offline designs, while uniform surrogate accuracy and $x_h \in S$ are explicit conditions. The bound concerns the fixed-reference scalar score; a Pareto-coverage interpretation requires additional reference-point and objective conditions.

Table 1: Average rank (↓) under HV, and OffPSL's win/tie/loss against each method over the 23 continuous tasks (task-mean HV, tie tolerance 0.01). OffPSL, PCD, and PGD-MOO are our runs over ten seeds; ParetoFlow, MOBO, STCH-PSL, and PSL-MOBO are our runs over five seeds, all under the shared evaluator and legal-box protocol. Rows marked ‡ are as reported by [1] (not rerun by us). Ranks use task-mean HV with average tie-breaking over learned methods; Dataset-ND-256 is an unranked reference. Avg. is the equally weighted mean of the three family ranks, computed before rounding. Per-task results are in Appendix D.

Method	ZDT	DTLZ	RE	Avg.	W/T/L
Dataset-ND-256 (ref.)	–	–	–	–	–
MultiHead‡	5.80	11.07	7.14	8.00	15/3/5
MultiHead-PCGrad‡	7.00	7.86	7.00	7.29	16/3/4
MultiHead-GradNorm‡	11.00	12.21	13.45	12.22	21/1/1
MultipleModels‡	5.60	9.64	5.05	6.76	15/2/6
MultipleModels-COM‡	10.00	10.36	8.91	9.76	18/4/1
MultipleModels-IOM‡	5.90	8.50	5.18	6.53	12/6/5
MultipleModels-ICT‡	4.80	10.50	4.91	6.74	12/5/6
MultipleModels-RoMA‡	7.70	8.36	8.95	8.34	16/3/4
MultipleModels-TriMentoring‡	10.60	10.57	5.64	8.94	16/4/3
MOBO	10.70	5.50	6.41	7.54	15/6/2
ParetoFlow	12.60	8.43	13.14	11.39	22/0/1
PGD-MOO	6.50	6.50	12.41	8.47	16/5/2
PCD	10.50	5.29	10.68	8.82	17/5/1
STCH-PSL	7.20	7.07	10.05	8.11	17/3/3
PSL-MOBO	15.80	10.86	13.23	13.29	21/1/1
OffPSL (Ours)	**4.30**	**3.29**	**3.86**	**3.82**	–

Proposition 3 (Support-distance consistency; summary)**.** *Let $M = \operatorname{supp}\mu$ be nonempty and compact, and let x have a unique nearest point in M. For the Gaussian MMSE denoiser d_s^* and an independent bounded positive scale factor $0 < a_{\min} \le A \le a_{\max} < \infty$,*

$$\mathbb{E}_{A,\xi}\|x - d^*_{tA}(x + tA\xi)\|^2 \longrightarrow \operatorname{dist}(x, M)^2 \quad (t \downarrow 0), \qquad \xi \sim \mathcal{N}(0, I_d).$$

The same pointwise limit holds for learned denoisers whose L^2 approximation error under these candidate-centered perturbations tends to zero.

Appendix F separates denoiser approximation and finite-perturbation errors and derives the covariance penalty's variance interpretation. The small-noise result supplies a conditional support-distance interpretation; the fixed noise settings are assessed by the empirical diagnostics in Section 5.

5 Experiments

5.1 Experimental Protocol

We evaluate all methods under the same offline protocol: the oracle is queried only after each method has submitted its fixed candidate set. Unless stated otherwise, each method submits $K = 256$ candidates per task and seed; for OffPSL these are 256 deterministic forward passes, one per reference preference. We evaluate 23 tasks from ZDT, DTLZ, and RE over five seeds; OffPSL, PCD, and PGD-MOO are additionally replicated at ten seeds on all continuous tasks (Appendix G), which leaves every aggregate conclusion unchanged. The benchmark evaluator normalizes objectives using offline dataset statistics and reports 100th-percentile HV under the reference-point convention of [23, 1]. All post-hoc diagnostic evaluations (unprojected, support-proxy, surrogate-error, and pool-retrieval analyses) use fixed candidate or diagnostic point sets after training concludes. The oracle-optimization study in Appendix J separately measures the effect of replacing the learned objective model with the true objective.

Table 2: Condensed interface-aware comparison on ZDT1–ZDT4, ZDT6, RE21, RE22; objectives are scale-normalized first. *Mono. viol.*: violations of the expected adjacent trade-off ordering; *Duplicate*: near-repeated designs or objective vectors; *Occupied bins*: breadth of coverage in normalized objective space; *Interp.*: local midpoint-preference consistency. Interpolation is interpreted jointly with duplicate rate and occupied-bin coverage.

Method	Protocol	Mono. viol. ↓	Duplicate ↓	Occupied bins ↑	Interp. ↑
OffPSL (Ours)	A	0.141 ± 0.120	0.072 ± 0.110	30.6 ± 19.7	0.974 ± 0.024
STCH-PSL	A	0.124 ± 0.212	0.164 ± 0.349	26.9 ± 22.7	0.976 ± 0.036
PSL-MOBO	A	0.157 ± 0.265	0.332 ± 0.381	8.1 ± 16.2	1.000 ± 0.001
MOBO	A	0.445 ± 0.093	0.001 ± 0.002	26.6 ± 16.0	0.218 ± 0.164
PGD-MOO	B	0.000 ± 0.000	0.446 ± 0.227	3.2 ± 1.6	1.000 ± 0.000
ParetoFlow	B	0.000 ± 0.000	0.488 ± 0.330	6.6 ± 6.4	1.000 ± 0.000

5.2 Baselines and Implementation

We compare four baseline groups: surrogate-based offline MBO methods from Off-MOO-Bench (MultiHead, MultipleModels, and COM/IOM/ICT/RoMA/TriMentoring variants) [23]; offline GP-style MOBO variants based on acquisition optimization [23, 6]; guided generative offline MOO methods PGD-MOO, ParetoFlow, and PCD [1, 26, 19]; and offline-adapted deterministic PSL baselines STCH-PSL and PSL-MOBO [13, 12]. PCD is evaluated using its official implementation and recommended hyperparameters under the shared 256-candidate budget. The benchmark evaluator applies legal-box projection uniformly to all methods before oracle evaluation (Appendix C.3). Appendix E.2 additionally reports unprojected results and clipping diagnostics for PCD and OffPSL.

Dataset-ND-256 summarizes offline data quality using up to 256 nondominated dataset designs. OffPSL hyperparameters and checkpointing rules are listed in Table 3.

5.3 Main Results

Table 1 summarizes family-level average ranks computed from task-mean HV with average tie-breaking over learned methods. The full per-task results are provided in Tables 4–6. OffPSL achieves the best average rank across all three benchmark families.

Absolute comparisons and significance. Scale-free normalized regret quantifies the performance gaps across tasks (Appendix G). Defining per-task normalized regret as $(\mathrm{HV}_t^\star - \mathrm{HV}_{m,t})/(\mathrm{HV}_t^\star - \mathrm{HV}_t^{\min})$ so that 0 is the best method on task t, the mean regret over the 23 continuous tasks is 0.043 for OffPSL versus 0.117 for the next-best method (MOBO), a $2.7\times$ gap. The W/T/L column of Table 1 reports OffPSL's per-method win/tie/loss on task-mean HV; paired Wilcoxon signed-rank tests across tasks are significant for all six key comparisons after Holm correction (adjusted $p \leq 0.0065$: PCD 0.0034, PGD-MOO 0.0030, ParetoFlow 0.0002, MOBO 0.0065, STCH-PSL 0.0031, PSL-MOBO < 0.0001). Appendix G reports the corresponding comparison using the best available per-task HV from published and rerun results.

IGD. On true-Pareto-front IGD, OffPSL attains the best average rank (3.13 of 7 methods, compared with 3.26 for MOBO and 3.30 for PCD). Paired tests detect no significant difference against PCD, MOBO, PGD-MOO, or ParetoFlow, and detect a significant improvement over PSL-MOBO (Appendix G.1). OffPSL thus combines statistically significant HV gains with competitive IGD and a deterministic, preference-queryable map at two orders of magnitude lower inference cost (Table 15).

5.4 Coverage and Preference-Map Diagnostics

Coverage and preference-map diagnostics evaluate the ordering, diversity, and continuity of the designs returned as preferences vary. Appendix E.6 visualizes two-objective results alongside the offline dataset, Dataset-ND-256, and true-front samples. Coloring OffPSL designs by λ_1 reveals how the learned map traverses the trade-off surface.

Table 2 compares user-facing interface metrics under two protocols. **Protocol A** evaluates the native returned preference sequence. **Protocol B** evaluates pool-based methods by post-hoc oracle retrieval

under the target scalarizer, providing a reference for the preference-sweep quality attainable from a fixed pool. Appendix C.4 defines the retrieval rule.

Table 2 shows that, among the three PSL methods, OffPSL has the lowest duplicate rate and highest occupied-bin count, with comparable monotonicity and interpolation consistency. PSL-MOBO is locally smooth but collapses to fewer bins, while Protocol-B retrieval for PGD-MOO and ParetoFlow attains perfect local monotonicity by repeatedly selecting from small pool regions. Appendix E.1 reports the full metrics.

Appendix E.2 measures the effect of output feasibility through unprojected, feasible-only, and invalid-as-reference evaluations of fixed outputs. A bounded-output variant directly parameterizes designs inside the legal box. Together with design-level clipping rates, these results characterize the relationship between output parameterization and HV.

5.5 Ablation and Component Analysis

Support prior at full scale. We compare no-support, kNN-penalty, denoising-residual, and residual-plus-covariance variants on *all* 23 continuous tasks with five seeds each (460 runs), under one fixed default configuration with offline-only support-weight calibration (Appendix H). The denoising residual outperforms the kNN penalty on 17/23 tasks in HV (mean $\Delta\mathrm{HV}{=}{+}1.32$) and 18/23 in true-PF IGD, at $+6\%$ training cost. The kNN penalty provides stronger clipping suppression, while the denoising residual provides higher set quality across these tasks. Relative to no support, the denoising prior is HV-neutral (median per-task $\Delta\mathrm{HV}{=}{-}0.013$) while reducing the duplicate rate (0.008 vs. 0.019), reducing design-level clipping (0.23 vs. 0.29), and improving occupied-bin coverage. Consistent with the conditional analysis of Section 4.5, its largest gains appear on tasks with smooth connected support (e.g., ZDT6: HV 3.98 vs. 3.73, IGD 0.10 vs. 0.37 against kNN).

Pessimism versus support. A factorial ablation crossing conservatism and support variants on five representative tasks (Appendix I) identifies support regularization as the primary stabilizing component: design-level clipping decreases from 0.52 to 0.33 and duplicate rate from 0.057 to 0.020 relative to the plain variant, with HV essentially unchanged. Coordinate-wise pessimism and scalarize-then-penalize perform comparably across all five tasks ($\max|\Delta\mathrm{HV}| = 0.013$), supporting robustness to the conservative parameterization.

DTLZ3 root cause. A post-hoc diagnostic crossing learned and true objective models with amortized and independent per-preference optimization (Appendix J) identifies surrogate representation error as the main source of the DTLZ3 gap. With independent per-preference search held fixed, replacing the frozen surrogate with the true objective reduces IGD from 0.126 to 0.012. Replacing the amortized map with independent search produces smaller, task-dependent changes. HV is nearly saturated on DTLZ3 (9.900 for oracle search and 9.891 for OffPSL), making IGD more informative for this comparison. Support-proxy validation (Appendix E.3) further shows strong correlations with kNN distance on ZDT6, moderate correlations on ZDT1, and weak correlations on DTLZ3 and RE42, characterizing the support signal at the finite noise settings used in practice.

Data size and coverage structure. Retraining the full pipeline on $10/25/50/100\%$ subsamples shows performance saturates early (at 50% the mean true-PF IGD ratio to the full data is 0.98; at 10% it is $1.37\times$). Holding the sample count fixed and removing coverage of specific trade-off regions is far more damaging: on RE22, keeping only the central region degrades IGD by $28.5\times$ relative to a uniform subsample of the same size, which explains the RE22 partial-collapse case and highlights the practical importance of coverage (Appendix K). Full-pipeline retraining changes all learned components; the IGD measurements do not isolate the terms of Proposition 2.

Scalarizer choice and output parameterization. Replacing the smooth Tchebycheff scalarizer with linear, augmented-Tchebycheff, or PBI alternatives trains stably in all cases; augmented-Tchebycheff matches STCH throughout, linear degrades on the disconnected ZDT3 front as theory predicts, and PBI is geometry-dependent (Appendix L). Training the bounded-output parameterization on all 23 tasks leaves 19/23 tasks unchanged within 0.01 HV under no-projection evaluation, with a genuine validity–performance trade-off on RE34, ZDT6, RE61, and DTLZ6 (Appendix E.2).

5.6 Computation Cost

Table 15 quantifies the amortization benefit. OffPSL fits its surrogate ensemble, denoising prior, and Pareto-set map offline, then generates candidates by direct evaluation of the learned map. Producing 256 preference-conditioned designs takes 60 ms on the reported workstation.

6 Conclusion

We introduced OffPSL, a deterministic Pareto set learning method for offline multi-objective optimization. A pessimistic surrogate scalarization and a frozen multi-noise denoising prior train a preference-conditioned map from fixed offline data. Each requested trade-off direction then yields a design in one forward pass. Across 23 tasks, OffPSL achieves the lowest normalized HV regret, with significant paired improvements after multiplicity correction, and competitive IGD at two orders of magnitude lower inference cost. The conditional analysis connects per-preference performance to surrogate accuracy, amortization error, and Pareto-relevant data coverage.

Limitations and Future Work. The diagnostics identify surrogate representation and data coverage as priorities for extending OffPSL. On DTLZ3, improved objective information yields the largest reduction in IGD (Appendix J). Removing a trade-off region at fixed sample count increases IGD by up to $28\times$ (Appendix K). These findings motivate coverage-aware training, richer surrogates for multimodal landscapes, and extensions to mixed design spaces.

References

[1] Y. Annadani, S. Belakaria, S. Ermon, S. Bauer, and B. E. Engelhardt. Preference-guided diffusion for multi-objective offline optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=moiVS9AY0q.

[2] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[3] C. Chen, Y. Zhang, J. Fu, X. S. Liu, and M. Coates. Bidirectional learning for offline infinite-width model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 29454–29467, 2022.

[4] C. S. Chen, C. Beckham, Z. Liu, X. S. Liu, and C. Pal. Parallel-mentoring for offline model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 76619–76636, 2023.

[5] S. Daulton, M. Balandat, and E. Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective Bayesian optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9851–9864, 2020.

[6] S. Daulton, M. Balandat, and E. Bakshy. Parallel Bayesian optimization of multiple noisy objectives with expected hypervolume improvement. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 2187–2200, 2021.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[8] N. Dimitriadis, P. Frossard, and F. Fleuret. Pareto manifold learning: Tackling multiple tasks via ensembles of single-task models. In *International Conference on Machine Learning (ICML)*, pages 8015–8052. PMLR, 2023.

[9] S. Krishnamoorthy, S. M. Mashkaria, and A. Grover. Diffusion models for black-box optimization. In *International Conference on Machine Learning (ICML)*, pages 17842–17857. PMLR, 2023.

[10] A. Kumar and S. Levine. Model inversion networks for model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 5126–5137, 2020.

[11] X. Lin, Z. Yang, and Q. Zhang. Pareto set learning for neural multi-objective combinatorial optimization. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=QuObT9BTWo.

[12] X. Lin, Z. Yang, X. Zhang, and Q. Zhang. Pareto set learning for expensive multi-objective optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. URL https://openreview.net/forum?id=vriLTB2-O0G.

[13] X. Lin, X. Zhang, Z. Yang, F. Liu, Z. Wang, and Q. Zhang. Smooth Tchebycheff scalarization for multi-objective optimization. In *International Conference on Machine Learning (ICML)*, pages 30479–30509. PMLR, 2024.

[14] S. M. Mashkaria, S. Krishnamoorthy, and A. Grover. Generative pretraining for black-box optimization. In *International Conference on Machine Learning (ICML)*, pages 24173–24197. PMLR, 2023.

[15] K. Miettinen. *Nonlinear multiobjective optimization*, volume 12. Springer Science & Business Media, 1999.

[16] A. Navon, A. Shamsian, G. Chechik, and E. Fetaya. Learning the Pareto front with hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2021.

[17] M.-D. Nguyen, P. M. Dinh, Q.-H. Nguyen, L. P. Hoang, and D. D. Le. Improving Pareto set learning for expensive multi-objective optimization via Stein variational hypernetworks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 19677–19685, 2025.

[18] H. Qi, Y. Su, A. Kumar, and S. Levine. Data-driven offline decision-making via invariant representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 13226–13237, 2022.

[19] J. Shrestha, S. Heiskanen, K. Hepola, S. Rissanen, P. Jääskeläinen, and J. Pajarinen. Pareto-conditioned diffusion models for offline multi-objective optimization. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=S2Q00li155.

[20] B. Trabucco, A. Kumar, X. Geng, and S. Levine. Conservative objective models for effective offline model-based optimization. In *International Conference on Machine Learning (ICML)*, pages 10358–10368. PMLR, 2021.

[21] B. Trabucco, X. Geng, A. Kumar, and S. Levine. Design-bench: Benchmarks for data-driven offline model-based optimization. In *International Conference on Machine Learning (ICML)*, pages 21658–21676. PMLR, 2022.

[22] T. A. Tuan, L. P. Hoang, D. D. Le, and T. N. Thang. A framework for controllable Pareto front learning with completed scalarization functions and its applications. *Neural Networks*, 169: 257–273, 2024.

[23] K. Xue, R. Tan, X. Huang, and C. Qian. Offline multi-objective optimization. In *International Conference on Machine Learning (ICML)*, pages 55595–55624. PMLR, 2024.

[24] S. Yu, S. Ahn, L. Song, and J. Shin. RoMA: Robust model adaptation for offline model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 4619–4631, 2021.

[25] Y. Yuan, C. S. Chen, Z. Liu, W. Neiswanger, and X. S. Liu. Importance-aware co-teaching for offline model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 55718–55733, 2023.

[26] Y. Yuan, C. Chen, C. Pal, and X. Liu. ParetoFlow: Guided flows in multi-objective optimization. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=mLyyB4le5u.

[27] Y. Yuan, Y. Zhang, C. Chen, H. Wu, M. Z. Li, J. Li, J. J. Clark, and X. Liu. Design editing for offline model-based optimization. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

[28] T. Yun, S. Yun, J. Lee, and J. Park. Guided trajectory generation with diffusion models for offline model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pages 83847–83876, 2024.

[29] Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[30] X. Zhang, X. Lin, B. Xue, Y. Chen, and Q. Zhang. Hypervolume maximization: A geometric view of Pareto set learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 38902–38929, 2023.

[31] T. Zhou, Z. Chen, W. Lyu, Z. Chen, D. H. Tsang, and J. Zhang. Learning design-score manifold to guide diffusion models for offline optimization. *npj Artificial Intelligence*, 2(1):4, 2026.

A Broader Impact and Societal Considerations

OffPSL studies offline multi-objective optimization, where candidate designs are optimized using a fixed dataset of previously evaluated examples. This setting is relevant to scientific and engineering domains in which unrestricted online experimentation is expensive, slow, unsafe, or otherwise impractical. Potential positive applications include more efficient exploration of design spaces for materials, molecules, hardware configurations, and engineering systems, where decision makers must balance performance, cost, robustness, and other competing criteria.

The offline formulation can reduce the need for new physical experiments or simulator calls, but it does not eliminate risk. The quality of the returned designs depends on the representativeness of the offline dataset and on the calibration of learned surrogate models. If the dataset is biased, incomplete, or collected under narrow operating conditions, an offline optimizer may amplify these biases by producing designs that appear favorable under the learned models but fail under real deployment conditions. This risk is particularly important in high-stakes domains such as drug discovery, medical decision support, energy systems, structural design, and autonomous systems.

OffPSL is intended as a decision-support tool rather than an autonomous deployment mechanism. The method returns candidate trade-off designs indexed by user-specified preferences, but these candidates should be validated using domain-specific safety checks, simulation, expert review, and, where appropriate, controlled experimental evaluation before deployment. The deterministic preference-to-design interface may improve inspectability relative to unordered candidate pools, because users can examine how generated designs change as preferences vary. However, this interface should not be interpreted as a guarantee of correctness, fairness, or safety.

Because OffPSL can be applied to general black-box design problems, it could also be misused in domains where automated optimization creates harmful artifacts or circumvents safety constraints. We therefore recommend using the method only under application-specific governance procedures, with explicit constraints on allowable design spaces and with independent validation of any generated candidate before real-world use.

B Detailed Method Specification

This appendix expands the method description in the main text. We focus on the information flow, the deterministic Pareto-set map, the conservative surrogate scalarizer, and the denoising-residual support prior. All quantities used during training are computed from the offline dataset unless stated otherwise.

B.1 Training Information Flow

Let

$$\mathcal{D} = \{(x_i^{\mathrm{raw}}, y_i^{\mathrm{raw}})\}_{i=1}^{N}, \qquad y_i^{\mathrm{raw}} = f(x_i^{\mathrm{raw}}),$$

be the fixed dataset in Section 2.4. It supplies the training inputs and objective targets for the three learned components below.

The implementation uses three learned components:

1. a surrogate ensemble $\{g_{\theta_j}\}_{j=1}^{J}$ trained on offline design–objective pairs;
2. a denoising support model d_ω trained only on offline designs;
3. a deterministic Pareto-set model $\widetilde{h}_\pi : \Delta^{m-1} \to \mathbb{R}^d$ trained through the frozen surrogate ensemble and frozen support model.

The selected model is the final exponential-moving-average version of $\widetilde{h}_\pi$ after a fixed number of training iterations.

B.2 Data Representations and Normalization

Neural networks operate in a normalized design space. We denote normalized designs by x_i and raw task-domain designs by x_i^{raw}. Objective targets are normalized coordinate-wise using only offline

dataset statistics:

$$y_i^{\mathrm{mm}} = \frac{y_i^{\mathrm{raw}} - y_{\min}^{\mathcal{D}}}{y_{\max}^{\mathcal{D}} - y_{\min}^{\mathcal{D}}},$$

where $y_{\min}^{\mathcal{D}}$ and $y_{\max}^{\mathcal{D}}$ are computed over the offline dataset. Small denominators are handled using the same numerical safeguard as the data loader. This normalization is used for surrogate training, conservative prediction, and scalarization.

Dataset-ND-256 is constructed by applying the benchmark dataset-reference procedure and retaining at most 256 nondominated dataset designs.

B.3 Deterministic Pareto Set Amortization

For a preference vector $\lambda \in \Delta^{m-1}$, define the corresponding scalarized optimum as

$$x_\lambda^\star \in \arg\min_{x \in \mathcal{X}} g(f(x); \lambda),$$

where g is a scalarizer. In online Pareto set learning, the map $\widetilde{h}_\pi(\lambda) \approx x_\lambda^\star$ can be improved using oracle evaluations or actively refined surrogates. In the offline setting, neither $x_\lambda^\star$ nor a label pair $(\lambda, x_\lambda^\star)$ is available. OffPSL therefore trains $\widetilde{h}_\pi$ by differentiating through frozen learned models.

The model is a preference-conditioned neural map

$$\widetilde{h}_\pi : \Delta^{m-1} \to \mathbb{R}^d.$$

For two-objective evaluation, we use a deterministic grid

$$\lambda_k = \left(\frac{k-1}{K-1}, 1 - \frac{k-1}{K-1} \right), \qquad k = 1, \ldots, K.$$

For $m \geq 3$, reference preferences are generated as deterministic reference directions on the simplex. During training, preferences are sampled from the simplex, and the same learned map is updated for all sampled trade-off directions. This amortized formulation is different from optimizing one design per preference independently: parameters are shared across preferences, so the model must learn a coherent representation of the Pareto set rather than a collection of unrelated candidate points.

B.4 Coordinate-Wise Conservative Surrogate

The surrogate ensemble consists of J independently initialized predictors

$$g_{\theta_j} : \mathcal{X} \to \mathbb{R}^m, \qquad j = 1, \ldots, J.$$

Each member is trained on the same offline dataset by minimizing

$$\min_{\theta_j} \frac{1}{N} \sum_{i=1}^{N} \left\| g_{\theta_j}(x_i) - y_i^{\mathrm{mm}} \right\|_2^2.$$

For a generated design x, the coordinate-wise ensemble mean and standard deviation are

$$\mu(x) = \frac{1}{J} \sum_{j=1}^{J} g_{\theta_j}(x),$$

and

$$\sigma(x) = \left(\frac{1}{J-1} \sum_{j=1}^{J} \left(g_{\theta_j}(x) - \mu(x) \right)^2 \right)^{1/2},$$

where the square and square root are applied coordinate-wise.

For minimization, OffPSL uses the pessimistic prediction

$$\hat{y}^{\mathrm{cons}}(x) = \mu(x) + \beta \sigma(x).$$

This construction preserves uncertainty separately for each objective before combining objectives through the preference vector. Appendix I compares this default with penalizing uncertainty after scalarization and finds comparable performance across the five evaluated tasks.

B.5 Smooth Tchebycheff Scalarization in Surrogate Space

Given a conservative prediction $\hat{y}^{\mathrm{cons}}(x)$ and a preference λ, OffPSL uses the smooth Tchebycheff scalarizer

$$S_\tau(\hat{y}^{\mathrm{cons}}, \lambda; z) = \tau \log \sum_{r=1}^{m} \exp\left(\frac{\lambda_r |\hat{y}_r^{\mathrm{cons}} - z_r|}{\tau}\right).$$

As $\tau \to 0$, this approaches the standard Tchebycheff maximum

$$\max_{r \in [m]} \lambda_r |\hat{y}_r^{\mathrm{cons}} - z_r|.$$

The scalarizer is applied in the same dataset min–max space as the surrogate outputs. This avoids mixing raw objective scales and prevents a single large-scale objective from dominating gradients.

The ideal point is set outside the observed dataset range:

$$z_\epsilon = -\epsilon \mathbf{1}, \qquad \epsilon > 0.$$

This design avoids treating the observed dataset minimum as the true ideal point. If $z = 0$ in dataset min–max space, then a generated design whose surrogate prediction moderately improves beyond the observed minimum can be penalized again because $|\hat{y}_r - z_r|$ increases when $\hat{y}_r < 0$. The outside ideal point reduces this spurious penalty while using only offline dataset statistics.

B.6 Relation to Diffusion-Based Offline Optimization

Diffusion models are commonly used in offline optimization by learning a distribution over offline designs and then steering generation toward high predicted objective values. A typical guided diffusion optimizer alternates between denoising steps and objective guidance, using gradients of a learned predictor or a preference-conditioned scalarization to bias samples toward better regions. In multi-objective offline optimization, this guidance can be conditioned on a preference vector, a dominance signal, or a predicted Pareto-improvement objective. Such methods use diffusion as a sampler: the final candidate set is produced by running an iterative reverse process and then selecting or filtering generated designs.

In OffPSL, the denoising model supplies a differentiable training-time support penalty. Its parameters are fitted to offline designs and frozen while the Pareto-set map learns preference-conditioned solutions. The resulting interface maps each preference to the feasible design $h_\pi^{\mathrm{feas}}(\lambda)$, coupling the training-time support model with deterministic candidate generation.

B.7 Denoising Support Model

Let $q_s(\tilde{x} \mid x)$ be a Gaussian perturbation of an offline design:

$$\tilde{x} = x + s\xi, \qquad \xi \sim \mathcal{N}(0, I),$$

where s is a noise scale. The denoising model $d_\omega(\tilde{x}, s)$ is trained to reconstruct the clean offline design:

$$\mathcal{L}_{\mathrm{den}}(\omega) = \mathbb{E}_{x \sim \mathcal{D}_x,\, s,\, \xi}\left[\|d_\omega(x + s\xi, s) - x\|_2^2\right].$$

The resulting denoiser is frozen before Pareto-set training, when its local reconstruction behavior supplies the support signal.

Intuitively, if a candidate x lies near the offline design manifold, then a small perturbation around x should denoise back close to x. If x lies in an unsupported region, the denoiser is more likely to map the perturbed point toward a different nearby training-supported region, producing a larger residual. This residual is used as a differentiable proxy for off-support extrapolation risk.

B.8 Multi-Noise Denoising Support Regularization

For a generated design $x = \widetilde{h}_\pi(\lambda)$, OffPSL draws P perturbations

$$\tilde{x}_p = x + s_p \xi_p, \qquad \xi_p \sim \mathcal{N}(0, I),$$

and computes denoising residuals

$$r_p(x) = x - d_\omega(\tilde{x}_p, s_p).$$

The residual support penalty is

$$R_{\text{sup}}(x) = \frac{1}{P}\sum_{p=1}^{P} \|r_p(x)\|_2^2.$$

This term penalizes generated designs whose noisy neighborhoods are not denoised back to the original point.

We also use a residual-covariance penalty:

$$R_{\text{cov}}(x) = \frac{1}{d}\,\text{tr}\left(\frac{1}{P-1}\sum_{p=1}^{P}(r_p(x) - \bar{r}(x))(r_p(x) - \bar{r}(x))^\top\right),$$

where

$$\bar{r}(x) = \frac{1}{P}\sum_{p=1}^{P} r_p(x).$$

While R_{sup} penalizes the average residual magnitude, R_{cov} penalizes instability of the denoising direction across noise samples. This is useful because some off-support points can have moderate average residual but highly inconsistent denoising behavior. The covariance term therefore encourages generated designs to lie in locally stable regions of the offline design distribution.

B.9 Diversity Regularization

A deterministic Pareto-set map can collapse by sending many preferences to the same design or to the same small region of objective space. To discourage this behavior, OffPSL optionally uses a batch-level repulsive term

$$R_{\text{div}} = \frac{2}{B(B-1)}\sum_{i<j}\exp\left(-\frac{\|u_i - u_j\|_2^2}{\eta}\right),$$

where u_i is the representation used for diversity. In the main setting, u_i is the conservative objective-space representation of $\widetilde{h}_\pi(\lambda_i)$. The bandwidth η is set adaptively from the current batch. Objective-space diversity is preferred over decision-space diversity when multiple distant designs can map to similar objective trade-offs, because the goal is to cover the Pareto front rather than to maximize design-space spread for its own sake.

B.10 Overall Pareto-Set Training Objective

For a mini-batch of preferences $\{\lambda_i\}_{i=1}^B$, let

$$x_i = \widetilde{h}_\pi(\lambda_i), \qquad \hat{y}_i^{\text{cons}} = \mu(x_i) + \beta\sigma(x_i).$$

The full OffPSL objective is

$$\begin{aligned}\mathcal{L}(\pi) = \frac{1}{B}\sum_{i=1}^{B} S_\tau\left(\hat{y}_i^{\text{cons}}, \lambda_i; z_\epsilon\right) \\ + \alpha_{\text{sup}}\frac{1}{B}\sum_{i=1}^{B} R_{\text{sup}}(x_i) + \alpha_{\text{cov}}\frac{1}{B}\sum_{i=1}^{B} R_{\text{cov}}(x_i) + \alpha_{\text{div}} R_{\text{div}}.\end{aligned}$$

The scalarization term pushes each preference toward predicted improvement under a conservative surrogate. The denoising-residual terms encourage generated designs to remain near stable offline support. The diversity term discourages preference collapse. The surrogate ensemble and denoising model are frozen during this stage; only π is updated.

Algorithm 1 OffPSL Training and Inference

Require: Offline dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, candidate budget K
Normalize designs and objectives using offline dataset statistics.
Train surrogate ensemble $\{g_{\theta_j}\}_{j=1}^{J}$ on (x_i, y_i^{mm}).
Train denoising support model d_ω on offline designs x_i.
Freeze $\{g_{\theta_j}\}_{j=1}^{J}$ and d_ω.
Initialize Pareto-set map $\widetilde{h}_\pi : \Delta^{m-1} \to \mathbb{R}^d$.
for $t = 1, \ldots, T$ **do**
    Sample preferences $\{\lambda_i\}_{i=1}^{B}$ from the simplex.
    Generate designs $x_i = \widetilde{h}_\pi(\lambda_i)$.
    Compute conservative predictions $\hat{y}_i^{\mathrm{cons}} = \mu(x_i) + \beta\sigma(x_i)$.
    Compute smooth Tchebycheff losses $S_\tau(\hat{y}_i^{\mathrm{cons}}, \lambda_i; z_\epsilon)$.
    Compute denoising residual penalties $R_{\mathrm{sup}}(x_i)$ and $R_{\mathrm{cov}}(x_i)$.
    Update π by minimizing the OffPSL objective.
end for
Construct K reference preferences $\{\lambda_k\}_{k=1}^{K}$.
return $X_K = \{h_\pi^{\mathrm{feas}}(\lambda_k)\}_{k=1}^{K}$, with legal-box projection.

Table 3: Main OffPSL hyperparameters used for the reported experiments.

Component	Setting
Pareto-set model	Deterministic MLP hypernetwork
Hidden width	256
Depth	3 linear layers
Activation	SiLU
Surrogate ensemble size J	5
Surrogate target space	dataset min–max normalized objectives
Surrogate training epochs	200
Conservative weight β	1.0
Scalarizer	smooth Tchebycheff
Scalarizer temperature τ	0.05
Outside ideal shift ϵ	0.2
Support weight α_{sup}	0.1
Support covariance weight α_{cov}	0.05
Denoising noise range	$s \in [0.05, 0.20]$
Support perturbations P	4
Diversity weight α_{div}	0.05

B.11 Training and Inference Algorithm

The returned set contains one design per reference preference and requires exactly K forward passes through $\widetilde{h}_\pi$.

C Experimental Protocol and Implementation Details

C.1 Candidate Budget and Offline Evaluation

Under the protocol in Section 5.1, each method submits $K = 256$ candidates per task and seed. For OffPSL, the submitted set is

$$X_{256} = \{h_\pi(\lambda_k)\}_{k=1}^{256}.$$

The candidate budget equals the number of deterministic map evaluations.

C.2 Evaluation Metrics

The primary metric is HV under the compatible evaluator in [23, 1]. Hypervolume is computed on the submitted candidate set using the benchmark reference-point convention. We report mean and sample standard deviation across seeds.

When true Pareto-front samples are available, we also report true-PF IGD:

$$\mathrm{IGD}(P^\star, Y_{\mathrm{nd}}) = \frac{1}{|P^\star|} \sum_{p \in P^\star} \min_{y \in Y_{\mathrm{nd}}} \|p - y\|_2,$$

where Y_{nd} is the non-dominated subset of the oracle-evaluated candidates. For two-objective tasks, we additionally report spacing and coverage diagnostics. The spacing metric is

$$\mathrm{Spacing}(Y_{\mathrm{nd}}) = \left[\frac{1}{n-1} \sum_{i=1}^{n} (d_i - \bar{d})^2 \right]^{1/2}, \qquad d_i = \min_{j \neq i} \|y_i - y_j\|_2.$$

Lower spacing indicates more uniform local gaps among the returned non-dominated points.

Table 3 reports the fixed training configuration.

C.3 Projection Diagnostics

For continuous tasks, let $\tilde{x}_k$ denote the raw design returned by a method after denormalization to the native task coordinates, and let

$$x_k^{\mathrm{clip}} = \Pi_{\mathcal{B}}(\tilde{x}_k), \qquad \mathcal{B} = [l, u],$$

where $\Pi_{\mathcal{B}}$ is coordinate-wise projection onto the legal box. The main benchmark evaluator uses x_k^{clip} whenever a raw candidate lies outside the legal box. For OffPSL, this is $h_\pi^{\mathrm{feas}}(\lambda_k)$ expressed in native task coordinates. The following diagnostics quantify feasibility and the effect of this projection.

Post-clip HV. This is the main benchmark value:

$$\mathrm{HV}_{\mathrm{clip}} = \mathrm{HV}\Big(\{ f(x_k^{\mathrm{clip}}) \}_{k=1}^{K}; r \Big),$$

where r is the benchmark reference point. This is the value used in the main HV tables.

No-projection HV. When the task oracle can be evaluated on the raw output $\tilde{x}_k$ and all returned objective values are finite, we compute

$$\mathrm{HV}_{\mathrm{raw}} = \mathrm{HV}\big(\{ f(\tilde{x}_k) \}_{k=1}^{K}; r \big).$$

If any raw oracle objective is non-finite, we report NA and separately report the finite raw-oracle fraction.

Feasible-only HV. Let

$$\mathcal{I}_{\mathrm{feas}} = \{ k : \tilde{x}_k \in \mathcal{B},\ f(\tilde{x}_k) \in \mathbb{R}^m \}.$$

Feasible-only HV computes HV using only the raw candidates in $\mathcal{I}_{\mathrm{feas}}$. This diagnostic asks how much performance is obtained from candidates that were already legal before projection.

Invalid-as-reference HV. For the most conservative diagnostic, invalid or non-finite raw candidates are assigned the reference-point value:

$$\bar{y}_k = \begin{cases} f(\tilde{x}_k), & \tilde{x}_k \in \mathcal{B} \text{ and } f(\tilde{x}_k) \in \mathbb{R}^m, \\ r, & \text{otherwise.} \end{cases}$$

We then compute $\mathrm{HV}(\{\bar{y}_k\}_{k=1}^{K}; r)$. Assigning invalid candidates to the reference point makes them dominated and therefore gives them no positive hypervolume contribution.

Finite raw-oracle fraction. For tasks where raw oracle calls can be non-finite, we report

$$\mathrm{NoProjFiniteFrac} = \frac{1}{K} \sum_{k=1}^{K} \mathbb{I}[f(\tilde{x}_k) \in \mathbb{R}^m].$$

Coordinate- and design-level clipping. We report both coordinate-level and design-level clipping using tolerance $\tau_{\text{clip}} = 10^{-6}$:

$$\text{ClipFrac}_{\text{coord}} = \frac{1}{Kd} \sum_{k=1}^{K} \sum_{\ell=1}^{d} \mathbb{I}\Big[|\tilde{x}_{k,\ell} - x_{k,\ell}^{\text{clip}}| > \tau_{\text{clip}}\Big],$$

and

$$\text{ClipFrac}_{\text{design}} = \frac{1}{K} \sum_{k=1}^{K} \mathbb{I}\Big[\|\tilde{x}_k - x_k^{\text{clip}}\|_\infty > \tau_{\text{clip}}\Big].$$

The design-level statistic is more directly interpretable: it is the fraction of submitted designs that required any box projection.

Bounded-output OffPSL. As a stronger robustness check, we also train a bounded-output parameterization

$$h_\pi^{\text{box}}(\lambda) = l + (u - l) \odot [\gamma + (1 - 2\gamma)\sigma(g_\pi(\lambda))], \qquad \gamma = 10^{-5},$$

where g_π is an unconstrained neural network. This maps every output strictly inside the legal box. Bounded-output OffPSL is evaluated without post-hoc projection.

C.4 Interface-Aware Diagnostics

For two-objective tasks, we sort queried preferences by λ_1 and evaluate the corresponding objective sequence in task-normalized objective space. The diagnostics characterize the ordering, continuity, and coverage of the preference sweep.

Protocols. Protocol A evaluates the native preference-conditioned output sequence. For pool-based methods, Protocol B retrieves a design from the oracle-evaluated pool $\mathcal{P}$ for each queried preference λ:

$$x_\lambda = \arg\min_{x \in \mathcal{P}} S_\tau(f(x), \lambda; z),$$

and computes the same interface metrics on the retrieved sequence. The oracle minimizer gives a best attainable scalarized score within the fixed pool, enabling a common assessment of its preference-sweep structure.

Spearman consistency. We report $\rho(\lambda_1, f_1)$ and $\rho(\lambda_1, f_2)$, the Spearman rank correlations between the queried preference coordinate and each oracle objective. For minimization, the expected signs depend on the objective convention, so we report signed correlations rather than collapsing them into an absolute value.

Monotonicity-violation rate. After sorting by λ_1, let $y_k = f(x_k)$ be the normalized objective vector. We count adjacent pairs whose objective movement violates the expected trade-off ordering:

$$\text{MonoViol} = \frac{1}{K-1} \sum_{k=1}^{K-1} \mathbb{I}[(y_{k+1,1} - y_{k,1} > 0) \ \vee \ (y_{k+1,2} - y_{k,2} < 0)],$$

with the sign convention adjusted consistently with the reported objective normalization. Lower values indicate a more ordered preference sweep.

Duplicate rate. Duplicate rate measures preference collapse. We report the fraction of returned designs whose nearest neighbor among the other returned designs is within a small tolerance in design or objective space:

$$\text{Dup} = \frac{1}{K} \sum_{k=1}^{K} \mathbb{I}\left[\min_{j \neq k} \|x_k - x_j\|_2 < \epsilon_x \ \vee \ \min_{j \neq k} \|y_k - y_j\|_2 < \epsilon_y\right].$$

Largest gap and occupied bins. We bin the normalized objective-space sweep along the primary trade-off coordinate. Occupied bins count how many bins contain at least one returned candidate. Largest gap is the largest empty interval between neighboring sorted candidates. Higher occupied bins and smaller largest gaps indicate broader coverage.

Smoothness and max jump. We compute

$$\mathrm{MaxJump} = \max_{k=1,\dots,K-1} \|y_{k+1} - y_k\|_2.$$

Lower values indicate a smoother user sweep, but this metric must be interpreted together with duplicate rate and occupied bins because a collapsed map can have a very small max jump.

Interpolation consistency. For adjacent queried preferences, we evaluate whether the midpoint-preference output falls inside the neighboring trade-off interval in normalized objective space. We report the fraction of adjacent intervals satisfying this condition. High interpolation consistency indicates local smoothness. Occupied bins and largest gaps quantify the complementary aspect of front coverage.

C.5 Support-Proxy Validation Protocol

The denoising residual penalties are used during training as frozen support signals. To validate whether they behave like support proxies, we compute post-hoc diagnostics on fixed point sets: offline training points, held-out offline points when available, random box points, OffPSL generated points, no-support ablation outputs, and residual-only ablation outputs when available, following Section 5.1.

Let $\mathcal{D}^x_{\mathrm{train}}$ be the offline training designs in the same normalized design space used by the support model. For a diagnostic point x, define

$$d_{\mathrm{kNN}}(x) = \frac{1}{k} \sum_{j=1}^{k} \|x - x_j^{\mathrm{NN}}\|_2,$$

where x_j^{NN} are the k nearest neighbors of x in $\mathcal{D}^x_{\mathrm{train}}$. We report Spearman rank correlations between $d_{\mathrm{kNN}}(x)$ and the frozen penalties $R_{\mathrm{sup}}(x)$ and $R_{\mathrm{cov}}(x)$. Since larger d_{kNN} means farther from the offline data and larger R_{sup} or R_{cov} means weaker support, positive correlations indicate that the penalty increases away from the offline data.

When oracle objectives are available for the fixed diagnostic points, we also compute a post-hoc surrogate error

$$e_{\mathrm{surr}}(x) = \|\hat{f}(x) - f(x)\|_2,$$

in the normalized objective space, and report Spearman correlations between $R_{\mathrm{sup}}(x)$ and $e_{\mathrm{surr}}(x)$.

For clipped/off-support classification, we use $R_{\mathrm{sup}}(x)$ as the scalar score and report AUC. Positive examples are diagnostic points marked as clipped or off-support under the saved diagnostic labels; offline and held-out offline points are used as in-support negatives. This diagnostic tests whether the residual penalty can separate unsupported or projection-heavy points from points drawn from the offline data support.

D Full Results Tables

This section reports complete per-task HV tables under the 256-design evaluation budget in Tables 4, 5, and 6. OffPSL, PCD, and PGD-MOO are our runs over ten seeds; ParetoFlow, MOBO, STCH-PSL, and PSL-MOBO are our runs over five seeds, all under the shared evaluator and legal-box protocol. Rows marked ‡ are as reported by [1] and were not rerun by us. Values are mean ± sample standard deviation. Dataset-ND-256 summarizes the quality of the offline data.

E Additional Analyses

E.1 Comparative Interface-Aware Diagnostics

The main HV tables evaluate submitted candidate sets as unordered sets. In contrast, the interface claim of deterministic Pareto set amortization concerns whether a user can sweep preferences and

Table 4: HV results on ZDT tasks under a 256-design budget. OffPSL, PCD, and PGD-MOO are our runs over ten seeds; ParetoFlow, MOBO, STCH-PSL, and PSL-MOBO are our runs over five seeds, all under the shared evaluator and legal-box protocol. Rows marked ‡ are as reported by [1] (not rerun by us). Dataset-ND-256 is an unranked dataset reference. Ranks use task-mean HV with average tie-breaking over learned methods; W/T/L is OffPSL's win/tie/loss against each method over this family's tasks (tie tolerance 0.01).

Method	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	Avg. rank	W/T/L
Dataset-ND-256 (ref.)	4.17	4.67	5.15	5.45	4.61	–	–
MultiHead‡	4.80±0.03	5.57±0.07	5.58±0.20	4.59±0.26	4.78±0.01	5.80	2/1/2
MultiHead-PCGrad‡	4.84±0.01	5.55±0.11	5.51±0.03	3.68±0.70	4.67±0.10	7.00	3/0/2
MultiHead-GradNorm‡	4.63±0.15	5.37±0.17	5.54±0.20	3.28±0.90	3.81±1.20	11.00	5/0/0
MultipleModels‡	4.81±0.02	5.57±0.07	5.48±0.21	5.03±0.19	4.78±0.01	5.60	3/0/2
MultipleModels-COM‡	4.52±0.02	4.99±0.12	5.49±0.07	5.10±0.08	4.41±0.21	10.00	5/0/0
MultipleModels-IOM‡	4.68±0.12	5.45±0.11	5.61±0.06	4.99±0.21	4.75±0.01	5.90	3/0/2
MultipleModels-ICT‡	4.82±0.01	5.58±0.01	5.59±0.06	4.63±0.43	4.75±0.01	4.80	2/1/2
MultipleModels-RoMA‡	4.84±0.01	5.43±0.35	**5.89±0.04**	4.13±0.11	1.71±0.10	7.70	3/0/2
MultipleModels-TriMentoring‡	4.64±0.10	5.22±0.11	5.16±0.04	5.12±0.12	2.61±0.01	10.60	5/0/0
MOBO	4.59±0.01	5.03±0.06	5.43±0.04	4.60±0.06	4.58±0.07	10.70	5/0/0
ParetoFlow	4.12±0.04	4.69±0.09	5.19±0.09	4.89±0.12	4.27±0.06	12.60	5/0/0
PGD-MOO	4.48±0.10	5.41±0.05	5.51±0.08	**5.42±0.04**	**4.81±0.02**	6.50	3/0/2
PCD	4.22±0.03	4.88±0.03	5.28±0.04	5.25±0.06	4.58±0.13	10.50	5/0/0
STCH-PSL	**4.85±0.02**	5.53±0.06	5.87±0.04	3.62±0.15	2.85±0.49	7.20	3/0/2
PSL-MOBO	2.86±0.10	2.97±0.00	4.63±0.09	3.54±0.15	1.57±0.02	15.80	5/0/0
OffPSL (Ours)	4.70±0.21	**5.61±0.11**	5.59±0.15	5.28±0.20	4.60±0.44	4.30	–

Table 5: HV results on DTLZ tasks under a 256-design budget. OffPSL, PCD, and PGD-MOO are our runs over ten seeds; ParetoFlow, MOBO, STCH-PSL, and PSL-MOBO are our runs over five seeds, all under the shared evaluator and legal-box protocol. Rows marked ‡ are as reported by [1] (not rerun by us). Dataset-ND-256 is an unranked dataset reference. Ranks use task-mean HV with average tie-breaking over learned methods; W/T/L is OffPSL's win/tie/loss against each method over this family's tasks (tie tolerance 0.01).

Method	DTLZ1	DTLZ2	DTLZ3	DTLZ4	DTLZ5	DTLZ6	DTLZ7	Avg. rank	W/T/L
Dataset-ND-256 (ref.)	10.60	9.91	10.00	10.76	9.35	8.88	8.56	–	–
MultiHead‡	10.51±0.23	9.03±0.56	10.48±0.23	6.73±1.40	8.41±0.15	8.72±1.07	**10.66±0.09**	11.07	5/1/1
MultiHead-PCGrad‡	10.64±0.01	9.64±0.33	10.55±0.12	9.95±1.93	9.02±0.24	9.90±0.25	10.61±0.03	7.86	5/1/1
MultiHead-GradNorm‡	10.64±0.01	8.86±1.27	10.26±0.28	7.45±0.75	7.87±1.06	8.16±2.21	10.31±0.22	12.21	5/1/1
MultipleModels‡	10.64±0.01	9.03±0.80	10.58±0.03	7.66±1.30	7.65±1.39	9.58±0.31	10.61±0.16	9.64	5/1/1
MultipleModels-COM‡	10.64±0.01	8.99±0.97	10.27±0.37	9.72±0.39	9.44±0.41	9.37±0.35	10.09±0.36	10.36	5/1/1
MultipleModels-IOM‡	10.64±0.01	10.10±0.27	10.24±0.13	10.03±0.53	9.77±0.18	9.30±0.31	10.60±0.05	8.50	5/1/1
MultipleModels-ICT‡	10.64±0.01	8.68±0.88	10.25±0.42	10.33±0.24	9.25±0.28	9.10±1.16	10.29±0.05	10.50	5/1/1
MultipleModels-RoMA‡	10.64±0.01	10.04±0.05	**10.61±0.03**	9.25±0.11	8.71±0.47	9.84±0.25	10.53±0.04	8.36	5/1/1
MultipleModels-TriMentoring‡	10.64±0.01	9.39±0.35	10.48±0.12	10.21±0.06	7.69±1.03	9.00±0.48	10.12±0.09	10.57	5/1/1
MOBO	**10.65±0.00**	12.13±0.02	9.87±0.01	17.50±0.01	9.94±0.01	10.87±0.01	10.44±0.02	5.50	5/1/1
ParetoFlow	10.62±0.01	12.28±0.02	9.84±0.02	16.63±0.66	10.63±0.03	10.70±0.05	8.88±0.22	8.43	6/0/1
PGD-MOO	**10.65±0.00**	12.37±0.03	9.89±0.00	14.73±0.83	10.40±0.11	10.52±0.11	9.86±0.15	6.50	5/2/0
PCD	**10.65±0.00**	12.42±0.00	9.89±0.00	**17.65±0.02**	10.66±0.01	10.87±0.05	9.35±0.08	5.29	4/2/1
STCH-PSL	**10.65±0.00**	11.71±0.10	9.89±0.00	14.44±0.61	9.58±0.16	10.36±0.03	10.40±0.09	7.07	5/2/0
PSL-MOBO	10.59±0.02	11.81±0.23	6.86±0.76	12.15±0.18	8.35±0.23	10.56±0.00	9.74±0.00	10.86	7/0/0
OffPSL (Ours)	**10.65±0.00**	**12.44±0.00**	9.89±0.00	15.37±1.14	**10.75±0.00**	**11.08±0.01**	**10.66±0.09**	3.29	–

receive a coherent sequence of trade-off designs. We therefore compare OffPSL against preference-conditioned PSL baselines and pool-based baselines using scale-normalized interface metrics on ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, RE21, and RE22.

We use the native-sequence and pool-retrieval protocols defined in Appendix C.4. Both are evaluated with the same scale-normalized interface metrics.

The main pattern is that OffPSL reduces preference collapse while preserving a stable local sweep. Among the three PSL methods, OffPSL has the lowest duplicate rate and the largest occupied-bin count. PSL-MOBO has very small max jump and nearly perfect interpolation, but this is paired with high duplication and low occupied-bin coverage, indicating that local smoothness can be caused by collapse. Under Protocol B, PGD-MOO and ParetoFlow achieve zero monotonicity violations because the retrieval step directly selects scalarization-favorable pool elements, but both methods repeatedly retrieve from a small portion of the pool, producing high duplicate rates and low occupied-

Table 6: HV results on RE tasks under a 256-design budget. OffPSL, PCD, and PGD-MOO are our runs over ten seeds; ParetoFlow, MOBO, STCH-PSL, and PSL-MOBO are our runs over five seeds, all under the shared evaluator and legal-box protocol. Rows marked ‡ are as reported by [1] (not rerun by us). Dataset-ND-256 is an unranked dataset reference. Ranks use task-mean HV with average tie-breaking over learned methods; W/T/L is OffPSL's win/tie/loss against each method over this family's tasks (tie tolerance 0.01).

Method	RE21	RE22	RE25	RE32	RE35	RE37	RE41	RE61	RE34	RE36	RE42	Avg. rank	W/T/L
Dataset-ND-256 (ref.)	4.10	4.78	4.79	10.56	10.08	5.57	18.27	97.49	9.30	7.61	14.52	–	–
MultiHead-GradNorm‡	4.28±0.39	4.70±0.44	4.52±0.50	10.54±0.15	9.76±1.30	5.67±1.41	17.06±3.82	108.01±1.00	8.47±1.87	9.67±0.43	18.77±2.99	13.45	11/0/0
MultiHead-PCGrad‡	4.59±0.01	4.73±0.36	4.78±0.14	10.63±0.01	10.51±0.05	6.68±0.06	20.66±0.10	108.54±0.23	10.11±0.00	10.17±0.08	22.57±0.26	7.00	8/2/1
MultiHead‡	**4.60±0.00**	**4.84±0.00**	4.74±0.20	10.60±0.05	10.49±0.07	6.67±0.05	20.62±0.11	108.92±0.22	10.10±0.01	10.23±0.03	22.38±0.35	7.14	8/1/2
MultipleModels-COM‡	4.38±0.09	**4.84±0.00**	4.83±0.01	10.64±0.01	10.55±0.02	6.35±0.10	20.37±0.06	107.99±0.48	9.96±0.09	9.82±0.35	17.44±0.71	8.91	8/3/0
MultipleModels-ICT‡	**4.60±0.00**	**4.84±0.00**	**4.84±0.00**	10.64±0.00	10.50±0.01	**6.73±0.00**	20.58±0.04	108.68±0.27	10.10±0.00	**10.29±0.03**	22.27±0.15	4.91	5/3/3
MultipleModels-IOM‡	4.58±0.02	**4.84±0.00**	4.83±0.01	**10.65±0.00**	10.57±0.01	6.71±0.02	20.66±0.05	107.71±0.50	10.11±0.01	**10.29±0.04**	22.43±0.10	5.18	4/5/2
MultipleModels-RoMA‡	4.57±0.00	4.61±0.51	4.83±0.01	10.64±0.00	10.53±0.03	6.67±0.02	20.39±0.09	108.47±0.28	9.91±0.01	9.72±0.28	21.41±0.37	8.95	8/2/1
MultipleModels-TriMentoring‡	**4.60±0.00**	**4.84±0.00**	**4.84±0.00**	10.62±0.01	**10.59±0.00**	**6.73±0.01**	20.68±0.04	108.61±0.29	10.08±0.02	9.64±1.42	21.60±0.19	5.64	6/3/2
MultipleModels‡	**4.60±0.00**	**4.84±0.00**	4.63±0.25	10.62±0.02	10.55±0.01	**6.73±0.03**	20.77±0.08	108.96±0.06	10.11±0.00	10.24±0.03	22.59±0.11	5.05	7/1/3
MOBO	4.57±0.00	**4.84±0.00**	4.81±0.00	10.64±0.00	10.57±0.00	6.72±0.00	**20.90±0.03**	108.53±0.10	10.05±0.00	10.05±0.03	21.84±0.03	6.41	5/5/1
ParetoFlow	4.08±0.05	4.79±0.04	4.82±0.01	10.60±0.04	10.41±0.03	5.53±0.28	17.59±0.28	95.78±3.40	9.49±0.15	8.40±0.14	16.21±0.55	13.14	11/0/0
PGD-MOO	3.05±0.00	**4.84±0.00**	**4.84±0.00**	10.64±0.00	3.20±0.00	6.07±0.12	11.08±0.58	83.97±7.27	8.21±0.00	6.60±0.00	15.56±0.00	12.41	8/3/0
PCD	4.34±0.01	**4.84±0.00**	**4.84±0.00**	**10.65±0.00**	10.33±0.01	5.66±0.01	17.73±0.03	91.80±0.08	9.78±0.02	8.15±0.08	15.81±0.06	10.68	8/3/0
STCH-PSL	4.58±0.00	**4.84±0.00**	4.65±0.25	10.49±0.07	10.49±0.07	6.47±0.38	20.30±0.15	109.27±0.13	9.69±0.02	9.07±0.10	19.31±1.88	10.05	9/1/1
PSL-MOBO	4.58±0.00	4.57±0.61	4.37±0.00	10.63±0.00	10.29±0.02	6.66±0.00	19.46±0.05	18.95±0.08	9.52±0.01	3.52±0.01	15.00±0.05	13.23	9/1/1
OffPSL (Ours)	4.51±0.04	**4.84±0.00**	**4.84±0.01**	10.64±0.00	10.58±0.01	6.72±0.01	**20.90±0.64**	**109.31±3.30**	**11.50±2.00**	10.16±0.13	**22.80±0.03**	3.86	–

Table 7: Scale-normalized interface-aware comparison on ZDT1, ZDT2, ZDT3, ZDT4, ZDT6, RE21, and RE22. All entries are mean±std over evaluated task–seed pairs. Protocols A (native sequence) and B (pool retrieval) are defined in Appendix C.4.

Method	Protocol	$\rho(\lambda_1, f_1)$	$\rho(\lambda_1, f_2)$	Mono. viol. ↓	Dup. ↓	Largest gap ↓	Occupied bins ↑	Max jump ↓	Interp. ↑
OffPSL (Ours)	A	−0.83±0.39	0.67±0.54	0.141±0.120	0.072±0.110	0.307±0.336	30.6±19.7	0.174±0.328	0.974±0.024
STCH-PSL	A	−0.77±0.50	0.58±0.70	0.124±0.212	0.164±0.349	0.024±0.027	26.9±22.7	0.026±0.032	0.976±0.036
PSL-MOBO	A	−0.68±0.71	0.57±0.74	0.157±0.265	0.332±0.381	0.018±0.093	8.1±16.2	0.002±0.004	1.000±0.001
MOBO	A	−0.70±0.27	0.45±0.35	0.445±0.093	0.001±0.002	0.473±0.279	26.6±16.0	0.636±0.374	0.218±0.164
PGD-MOO	B	−0.88±0.09	0.88±0.09	0.000±0.000	0.446±0.227	0.384±0.310	3.2±1.6	0.384±0.310	1.000±0.000
ParetoFlow	B	−0.85±0.18	0.85±0.18	0.000±0.000	0.488±0.330	0.203±0.200	6.6±6.4	0.203±0.200	1.000±0.000

bin counts. These results show how OffPSL balances ordered preference response, low duplication, and broad objective-space coverage within a directly queryable map.

Per-task OffPSL diagnostics. For two-objective tasks, OffPSL returns one design for each scalar preference $\lambda_1 \in [0, 1]$. A useful diagnostic is whether the oracle-evaluated objectives change coherently as λ_1 varies. We report Spearman correlations between λ_1 and each oracle objective:

$$\rho_r = \mathrm{Spearman}\left(\{\lambda_{k,1}\}_{k=1}^{K}, \{f_r(h_\pi(\lambda_k))\}_{k=1}^{K}\right).$$

For minimization, increasing λ_1 should typically improve objective 1 and worsen the competing objective, although the exact sign depends on the objective convention. We therefore report the signed correlations together with objective-space scatter plots colored by λ_1.

Table 8: Preference-map diagnostics on two-objective tasks. Higher ordered coverage and smaller largest gaps indicate a more coherent learned Pareto-set map.

Task	$\rho(\lambda_1, f_1)$	$\rho(\lambda_1, f_2)$	Occupied bins ↑	Largest gap ↓
ZDT1	-0.796 ± 0.452	0.997 ± 0.004	19.0 ± 2.2	0.036 ± 0.020
ZDT2	-0.592 ± 0.773	0.971 ± 0.050	19.8 ± 0.4	0.040 ± 0.016
ZDT3	-0.996 ± 0.002	0.327 ± 0.713	20.0 ± 0.0	0.026 ± 0.008
ZDT4	-0.457 ± 0.546	-0.245 ± 0.514	20.0 ± 0.0	0.017 ± 0.007
ZDT6	-0.876 ± 0.099	0.998 ± 0.002	20.0 ± 0.0	0.034 ± 0.009
RE21	-0.988 ± 0.002	0.994 ± 0.001	20.0 ± 0.0	0.013 ± 0.002
RE22	-0.948 ± 0.016	-0.481 ± 0.168	10.2 ± 0.4	0.210 ± 0.000

Table 8 shows that OffPSL usually learns a coherent preference-to-design map on two-objective tasks, but the quality of this map is task dependent. On ZDT1, ZDT2, ZDT6, and RE21, the oracle-evaluated objectives vary monotonically with the queried preference, with high-magnitude rank correlations, nearly full occupied-bin counts, and small largest gaps. These results indicate that OffPSL not only returns high-HV sets, but also preserves a relatively ordered and well-spread parameterization of the Pareto set.

The more difficult cases help clarify the limitations of these diagnostics. On ZDT3, the occupied-bin count and largest-gap statistic remain strong, while the correlation with one objective becomes weak

Table 9: No-projection and clipping robustness diagnostics. Diagnostic post-clip HV summarizes five seeds (1000–5000). For all tasks except RE21, these are the first five seeds of the ten-seed benchmark in Tables 4–6. RE21 uses a separate box-corrected diagnostic configuration. No-projection HV evaluates raw outputs when all raw oracle objectives are finite; NA means at least one raw oracle objective is non-finite. Feasible-only HV keeps only raw candidates that are feasible before projection. Invalid-as-reference HV assigns invalid or non-finite raw candidates the benchmark reference-point value. ClipFrac is reported both at coordinate level and design level. Values are mean±sample std across seeds.

Task	Diag. post-clip HV	No-proj. HV	Feas.-only HV	Invalid-ref HV	No-proj. finite frac.	ClipFrac coord.	ClipFrac design
ZDT1	4.737±0.285	NA	4.292±0.531	4.292±0.531	0.621±0.124	0.094±0.015	0.766±0.130
ZDT2	5.668±0.090	5.668±0.090	5.405±0.178	5.405±0.178	1.000±0.000	0.090±0.006	0.640±0.124
ZDT3	5.635±0.211	NA	4.983±0.364	4.983±0.364	0.841±0.026	0.065±0.016	0.637±0.138
ZDT4	5.447±0.054	5.447±0.054	5.447±0.054	5.447±0.054	1.000±0.000	0.013±0.028	0.125±0.280
ZDT6	4.604±0.463	NA	4.459±0.435	4.459±0.435	0.464±0.050	0.000±0.000	0.000±0.000
DTLZ3	9.891±0.004	9.891±0.004	9.890±0.003	9.890±0.003	1.000±0.000	0.008±0.016	0.083±0.164
DTLZ4	14.831±0.001	14.831±0.001	14.831±0.001	14.831±0.001	1.000±0.000	0.004±0.009	0.039±0.087
RE21	4.355±0.034	4.355±0.034	4.335±0.028	4.335±0.028	1.000±0.000	0.017±0.010	0.068±0.038
RE22	4.840±0.000	4.840±0.000	4.836±0.008	4.836±0.008	1.000±0.000	0.104±0.075	0.312±0.224
RE42	22.825±0.030	22.830±0.026	22.738±0.022	22.738±0.022	1.000±0.000	0.090±0.004	0.331±0.014

and variable. This is consistent with the disconnected geometry of the front: global rank correlation can become less informative even when the returned solutions still cover multiple segments. ZDT4 shows a related phenomenon: ordered coverage remains high in terms of occupied bins and gap size, but the objective-wise rank correlations are less stable, suggesting that multimodality can distort the global preference ordering without causing complete collapse. RE22 is the clearest failure case. Its occupied-bin count is much lower and its largest-gap value is substantially higher than for the other tasks, indicating that the learned map covers only part of the attainable trade-off surface. Together, the preference-map diagnostics and HV characterize both the structure and set quality of the returned designs.

E.2 Projection and Bounded-Output Robustness

We quantify output feasibility and projection sensitivity using saved candidate sets. For each run, we compare benchmark post-clip HV with unprojected HV, feasible-only HV, and invalid-as-reference HV, as defined in Appendix C.3. Coordinate-level and design-level clipping fractions describe the extent of box violations. Synthetic tasks use the task legal box, while RE tasks use dataset min–max bounds matching the saved evaluation runs.

Across ZDT2, ZDT4, DTLZ3, DTLZ4, RE21, RE22, and RE42, unprojected HV matches or closely tracks post-clip HV when raw oracle evaluations are finite. Feasible-only HV also remains close to the post-clip value on ZDT4, DTLZ3, DTLZ4, RE21, RE22, and RE42, showing that feasible raw candidates sustain set quality on these tasks. On RE42, unprojected HV (22.830 ± 0.026) and invalid-as-reference HV (22.738 ± 0.022) remain close to post-clip HV (22.825 ± 0.030) despite moderate design-level clipping.

ZDT1 and ZDT3 show larger projection sensitivity under the conservative raw-output diagnostics, so we further train a bounded-output variant whose outputs are constrained to lie inside the legal box by construction. ZDT6 is a different case: its design-level clip fraction is zero, but the raw oracle has non-finite values for a subset of raw outputs, so the NA no-projection HV reflects oracle-domain or numerical non-finiteness rather than box projection.

Bounded-output OffPSL generates designs inside the legal box and is evaluated directly. On ZDT1, ZDT2, ZDT3, and RE42, it achieves HV close to or above the original post-clip values. On RE42, bounded HV is 22.829 ± 0.026, with a paired mean difference of $+0.004\pm0.012$. These results demonstrate that a feasible output parameterization preserves HV on these four tasks.

Full-suite bounded-output study. Table 11 extends the bounded-output parameterization to all 23 continuous tasks (5 seeds each, no post-hoc projection anywhere). The median per-task change in HV relative to the five-seed controls is 0.00%: 19 of 23 tasks are unchanged to within 0.01 HV and ZDT1 improves by $+1.7\%$, while four tasks show a genuine validity–performance trade-off (RE34 -24.2%, ZDT6 -7.9%, RE61 -5.4%, DTLZ6 -0.3%): the sigmoid map cannot place designs on the box boundary, and on these tasks part of the attainable front lies exactly there. Design-level clipping

Table 10: Bounded-output OffPSL evaluated without post-hoc projection over five seeds (1000–5000). The post-clip and invalid-reference controls use these same five seeds; the post-clip control is the first-five-seed subset of the ten-seed benchmark in Tables 4–6. Bounded runs use an interior sigmoid with margin 10^{-5}, task-box bounds on ZDT, and dataset min–max bounds on RE42. ΔHV is the difference between the bounded no-clip HV mean and the five-seed post-clip control mean. For RE42, the paired per-seed difference is $+0.004 \pm 0.012$.

Task	5-seed post-clip HV	5-seed invalid-ref HV	Bounded no-clip HV	ΔHV	Bounded ClipFrac design
ZDT1	4.737 ± 0.285	4.292 ± 0.531	4.816 ± 0.109	$+0.079$	0.000 ± 0.000
ZDT2	5.668 ± 0.090	5.405 ± 0.178	5.668 ± 0.090	$+0.000$	0.000 ± 0.000
ZDT3	5.635 ± 0.211	4.983 ± 0.364	5.634 ± 0.212	-0.001	0.000 ± 0.000
RE42	22.825 ± 0.030	22.738 ± 0.022	22.829 ± 0.026	$+0.004$	0.000 ± 0.000

Table 11: Bounded-output OffPSL on all 23 continuous tasks (5 seeds: 1000–5000), evaluated with no post-hoc projection. The baseline HV is the first-five-seed mean of the original OffPSL runs; Tables 4–6 report the ten-seed benchmark. ΔHV is the bounded mean minus this five-seed baseline mean. Design-level clipping is exactly zero on 114 of 115 runs (single exception: ZDT4 seed 3000 at 0.60; the task aggregate is unaffected).

Task	5-seed baseline HV	Bounded no-clip HV	ΔHV
ZDT1	4.737	4.816	+0.078
ZDT2	5.668	5.668	-0.000
ZDT3	5.635	5.634	-0.000
ZDT4	5.447	5.449	+0.002
ZDT6	4.604	4.243	-0.361
DTLZ1	10.646	10.646	-0.000
DTLZ2	12.436	12.436	-0.000
DTLZ3	9.891	9.892	+0.001
DTLZ4	14.831	14.831	+0.000
DTLZ5	10.748	10.748	+0.000
DTLZ6	11.076	11.044	-0.033
DTLZ7	10.617	10.617	-0.000
RE21	4.498	4.498	-0.000
RE22	4.840	4.840	-0.000
RE25	4.838	4.838	-0.000
RE32	10.640	10.640	-0.000
RE34	13.081	9.922	-3.159
RE35	10.576	10.576	-0.000
RE36	10.213	10.212	-0.001
RE37	6.721	6.721	+0.000
RE41	20.691	20.691	+0.000
RE42	22.825	22.824	-0.001
RE61	112.394	106.284	-6.109

is exactly zero on 114 of 115 bounded runs; the single exception (ZDT4, seed 3000, 0.60 against the training box) does not affect the ZDT4 aggregate (ΔHV $+0.002$). We therefore recommend the bounded parameterization as the default wherever boundary violations have physical meaning, and retain the unconstrained variant with projection diagnostics for comparability.

For PCD, the mean design-level clip fraction across 23 tasks and five seeds is 0.60. More than half of the raw designs fall outside the legal box on 13 tasks (ZDT6 0.997, ZDT2 0.993, and ZDT1 0.975; the lowest rate is 0.066 on RE36). PCD's original study evaluates unprojected candidates. The comparisons here use uniform legal-box projection, with the raw-output diagnostics reported separately.

E.3 Support-Proxy Validation and Support-Penalty Comparison

We assess the frozen support penalties using correlations with kNN distance and surrogate error, together with clipped/off-support classification AUC. The point sets and evaluation procedure are specified in Appendix C.5.

The residual provides a task-dependent local support signal. Its correlation with kNN distance is strong on ZDT6, moderate on ZDT1, and weak on DTLZ3 and RE42. The covariance penalty is similarly informative on ZDT6. Classification AUC follows the same pattern: 0.995 on ZDT6, 0.648 on ZDT1, and values near 0.5 on DTLZ3 and RE42. These measurements characterize finite-noise

Table 12: Support-proxy validation on representative tasks. R_{sup} is the frozen denoising-residual penalty and R_{cov} is the frozen denoising-covariance penalty, so larger values indicate weaker support. Positive Spearman correlations with kNN distance d_{kNN} indicate that the penalty increases with distance from the offline data. AUC uses R_{sup} as the clipped/off-support classifier score. All entries are mean±std across seeds. The evaluation follows Appendix C.5.

Task	Point sets	Spearman$(R_{\mathrm{sup}}, d_{\mathrm{kNN}})\uparrow$	Spearman$(R_{\mathrm{cov}}, d_{\mathrm{kNN}})\uparrow$	Spearman$(R_{\mathrm{sup}}, e_{\mathrm{surr}})\uparrow$	AUC clipped/off-support ↑
ZDT1	dataset / held-out / random OffPSL / no-support	0.372 ± 0.111	-0.005 ± 0.008	0.372 ± 0.163	0.648 ± 0.049
ZDT6	dataset / held-out / random OffPSL	0.830 ± 0.010	0.736 ± 0.020	0.233 ± 0.026	0.995 ± 0.001
DTLZ3	dataset / held-out / random OffPSL / no-support / residual-only	0.164 ± 0.043	-0.011 ± 0.025	0.065 ± 0.027	0.483 ± 0.039
RE42	dataset / held-out / random OffPSL / no-support / residual-only	0.181 ± 0.040	-0.008 ± 0.010	-0.002 ± 0.052	0.522 ± 0.015

Table 13: Single-seed support-penalty comparison on representative tasks, characterizing coverage, clipping, and duplication. IGD is measured against the reference front.

Task	Variant	HV ↑	IGD ↓	ClipFrac design ↓	Duplicate ↓	Occupied bins ↑
DTLZ3	Denoising + kNN	9.89	0.216	0.000	0.000	25.0
DTLZ3	Denoising residual	9.89	0.156	0.000	0.000	24.0
DTLZ3	kNN support penalty	9.89	0.122	0.000	0.000	28.0
DTLZ3	L2-nearest support penalty	9.88	0.163	0.000	0.000	29.0
DTLZ3	No support	9.89	0.183	0.000	0.000	29.0
RE42	Denoising + kNN	22.61	0.087	0.957	0.000	45.0
RE42	Denoising residual	22.85	0.086	0.977	0.007	43.0
RE42	kNN support penalty	22.62	0.088	0.965	0.000	43.0
RE42	L2-nearest support penalty	22.45	0.087	0.938	0.000	47.0
RE42	No support	22.79	0.106	0.941	0.006	32.0
ZDT1	Denoising + kNN	4.72	NA	0.289	0.000	50.0
ZDT1	Denoising residual	4.86	NA	0.801	0.039	48.0
ZDT1	kNN support penalty	4.74	NA	0.320	0.000	49.0
ZDT1	L2-nearest support penalty	4.61	NA	0.500	0.000	49.0
ZDT1	No support	4.86	NA	1.000	0.004	48.0
ZDT2	Denoising + kNN	5.71	0.006	0.473	0.000	39.0
ZDT2	Denoising residual	5.71	0.005	0.816	0.162	42.0
ZDT2	kNN support penalty	5.71	0.006	0.473	0.000	40.0
ZDT2	L2-nearest support penalty	5.64	0.053	0.441	0.000	42.0
ZDT2	No support	5.71	0.008	0.859	0.005	31.0
ZDT6	Denoising + kNN	4.80	0.023	0.000	0.000	46.0
ZDT6	Denoising residual	4.81	0.009	0.000	0.000	45.0
ZDT6	kNN support penalty	4.79	0.026	0.000	0.000	41.0
ZDT6	L2-nearest support penalty	2.22	0.910	0.000	0.000	3.0
ZDT6	No support	3.78	0.127	0.000	0.000	15.0

behavior on the evaluated tasks; the small-noise consistency result provides a conditional mechanism interpretation.

We also evaluate alternative support penalties within the same training pipeline on representative tasks to characterize their trade-offs across coverage, clipping, and duplication.

The support penalties produce distinct trade-offs. kNN penalties reduce clipping on ZDT1 and ZDT2. On ZDT6, denoising residuals achieve HV 4.81, IGD 0.009, and 45 occupied bins, while L2-nearest penalties concentrate the returned set in three bins. Relative to the no-support variant, denoising residuals expand occupied-bin coverage from 15 to 45 on ZDT6 and from 32 to 43 on RE42. The joint metrics characterize how each penalty shapes the preference map.

E.4 Sensitivity Analysis

We sweep key coefficients one factor at a time on representative tasks to evaluate operational stability across hyperparameter regimes.

Table 14: Sensitivity analysis on representative tasks. We vary one factor at a time and report hypervolume, true-PF IGD, and clip fraction.

Sweep	Values	Task	HV ↑	true-PF IGD ↓	Clip fraction ↓
ϵ	0	ZDT2	5.48 ± 0.02	0.170 ± 0.188	0.670 ± 0.237
ϵ	0.1	ZDT2	5.67 ± 0.09	0.105 ± 0.222	0.600 ± 0.097
ϵ	0.2	ZDT2	5.67 ± 0.09	0.105 ± 0.222	0.607 ± 0.047
ϵ	0.3	ZDT2	5.67 ± 0.09	0.105 ± 0.222	0.612 ± 0.052
ϵ	0	ZDT6	4.18 ± 0.55	0.082 ± 0.057	0.000 ± 0.000
ϵ	0.1	ZDT6	4.39 ± 0.56	0.057 ± 0.063	0.000 ± 0.000
ϵ	0.2	ZDT6	4.19 ± 0.56	0.080 ± 0.061	0.000 ± 0.000
ϵ	0.3	ZDT6	4.19 ± 0.56	0.079 ± 0.062	0.000 ± 0.000
β	0	ZDT6	4.39 ± 0.57	0.064 ± 0.066	0.000 ± 0.000
β	0.5	ZDT6	4.40 ± 0.57	0.059 ± 0.061	0.000 ± 0.000
β	1	ZDT6	4.60 ± 0.46	0.032 ± 0.052	0.000 ± 0.000
β	2	ZDT6	4.40 ± 0.57	0.057 ± 0.066	0.000 ± 0.000
β	0	RE42	22.83 ± 0.02	0.088 ± 0.005	0.951 ± 0.006
β	0.5	RE42	22.83 ± 0.02	0.087 ± 0.004	0.947 ± 0.008
β	1	RE42	22.82 ± 0.03	0.088 ± 0.005	0.331 ± 0.014
β	2	RE42	22.81 ± 0.03	0.088 ± 0.004	0.969 ± 0.012
$\alpha_{\text{sup}}, \alpha_{\text{cov}}$	no	DTLZ3	9.89 ± 0.01	0.138 ± 0.035	0.260 ± 0.188
$\alpha_{\text{sup}}, \alpha_{\text{cov}}$	residual	DTLZ3	9.89 ± 0.00	0.165 ± 0.027	0.083 ± 0.164
$\alpha_{\text{sup}}, \alpha_{\text{cov}}$	residual+cov	DTLZ3	9.89 ± 0.00	0.169 ± 0.027	0.083 ± 0.164
$\alpha_{\text{sup}}, \alpha_{\text{cov}}$	no	RE42	22.80 ± 0.01	0.095 ± 0.004	0.368 ± 0.014
$\alpha_{\text{sup}}, \alpha_{\text{cov}}$	residual	RE42	22.82 ± 0.03	0.088 ± 0.005	0.330 ± 0.013
$\alpha_{\text{sup}}, \alpha_{\text{cov}}$	residual+cov	RE42	22.82 ± 0.03	0.088 ± 0.005	0.331 ± 0.014

Table 14 shows stable performance over moderate hyperparameter ranges. On ZDT2, a positive scalarizer offset improves HV and true-PF IGD over $\epsilon = 0$, and performance remains stable for $\epsilon \in [0.1, 0.3]$. This supports the offset ideal-point construction across that range.

On ZDT6, $\beta = 1$ provides the strongest HV/IGD trade-off among the evaluated values. On RE42, HV varies little across β, while design-level clipping decreases to 0.331 at $\beta = 1$, compared with 0.947–0.969 at the other settings. Adding support regularization reduces clipping on DTLZ3 while preserving HV, and improves HV, IGD, and clipping on RE42. These sweeps support the default configuration as a stable operating point across the evaluated task geometries.

E.5 Computation Cost

All experiments were conducted on a desktop workstation equipped with an Intel Core i5-14600KF CPU and an NVIDIA GeForce RTX 4070 Ti SUPER GPU with 16 GB VRAM.

We report training and inference costs separately. Training includes the cost of fitting the surrogate ensemble, the denoising support model, and the Pareto-set map. Inference is measured as the wall-clock time required to produce the 256 candidate designs used by the evaluator.

Table 15: Computation cost. Inference time is the time required to generate the 256 candidates.

Method	Model training (s)	Inference for 256 designs (s)
MultipleModels-ICT	748.26	7.95
MultiHead-PCGrad	239.38	7.61
MOBO	0.78	3309.40
ParetoFlow	4404.78	32.62
PGD-MOO	2961.81	7.83
STCH-PSL	12.37	0.04
PSL-MOBO	15.20	0.03
OffPSL (Ours)	262.97	0.06

OffPSL amortizes optimization into the offline training phase. Fitting the surrogate ensemble, denoising model, and Pareto-set map takes approximately 260 seconds, after which 256 preference-

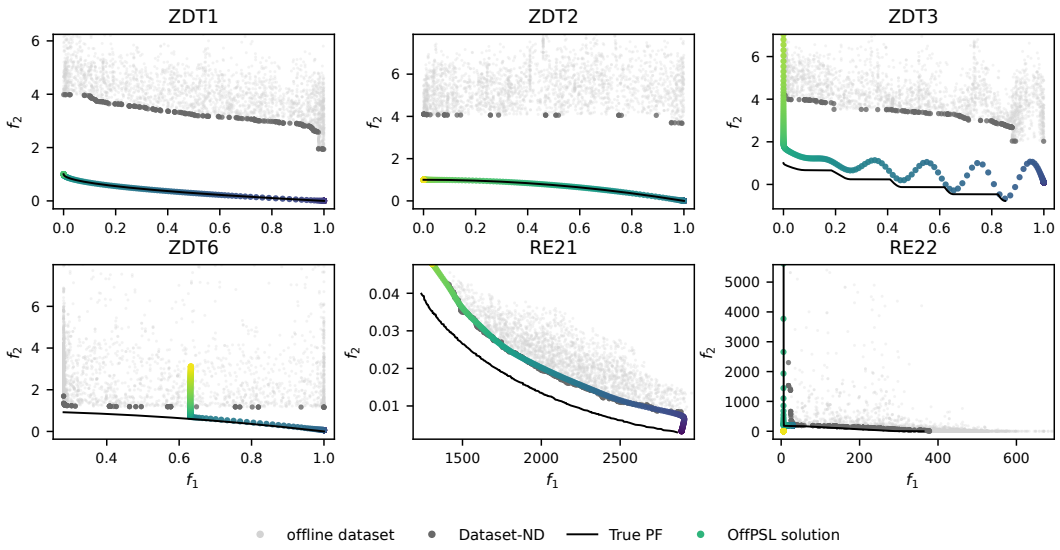

Figure 3: Qualitative objective-space diagnostics on two-objective tasks. The offline dataset is shown in light gray, Dataset-ND in dark gray, the true Pareto front in black, and OffPSL solutions in color. On ZDT1, ZDT2, and RE21, OffPSL produces solutions that closely follow the true front and vary smoothly with the queried preferences, indicating a coherent preference-to-design map. ZDT3 and ZDT6 illustrate harder cases where the learned solutions cover useful regions of the front but exhibit stronger non-uniformity due to disconnected or highly non-uniform front geometry. RE22 shows the clearest partial-collapse behavior: many returned solutions concentrate near a narrow region of objective space, which is consistent with the larger largest-gap and lower occupied-bin diagnostics reported in Table 8.

conditioned designs are generated in 60 milliseconds. MOBO, ParetoFlow, and PGD-MOO perform acquisition optimization or iterative generation at inference, accounting for their higher candidate-generation times in Table 15.

E.6 Qualitative Objective-Space Diagnostics

Objective-space visualizations complement the quantitative diagnostics in Appendix E.1 by showing how the preference map traverses the trade-off surface. Each panel displays the offline dataset, its nondominated subset, the benchmark true front when available, and OffPSL candidates colored by reference-preference order.

Figure 3 shows that OffPSL usually learns an ordered and front-seeking map on the two-objective tasks, but the degree of coverage depends on the geometry and support of the benchmark. On ZDT1, ZDT2, and RE21, the returned candidates trace the reference front with a clear preference ordering and improve substantially over the nondominated subset of the offline dataset. This supports the central use case of deterministic Pareto set amortization: a user can sweep preferences and obtain a corresponding ordered sequence of trade-off designs.

The same figure also exposes failure modes that are less visible from HV alone. For ZDT3, the discontinuous Pareto front makes global preference ordering more difficult to summarize by a single monotonic correlation, even though several front segments are recovered. For ZDT6, OffPSL returns a mixture of front-aligned solutions and a nearly vertical branch, suggesting that some preferences are pulled toward a narrow supported region instead of uniformly covering the full front. RE22 is the most pronounced failure case: the generated set occupies only a limited portion of the trade-off surface, confirming that the learned map can partially collapse when the attainable front and the offline support are highly irregular.

Figure 4 shows structured nondominated candidates on three-objective engineering tasks. On RE32 and RE35, the candidates concentrate near narrow, high-performing regions of the nondominated

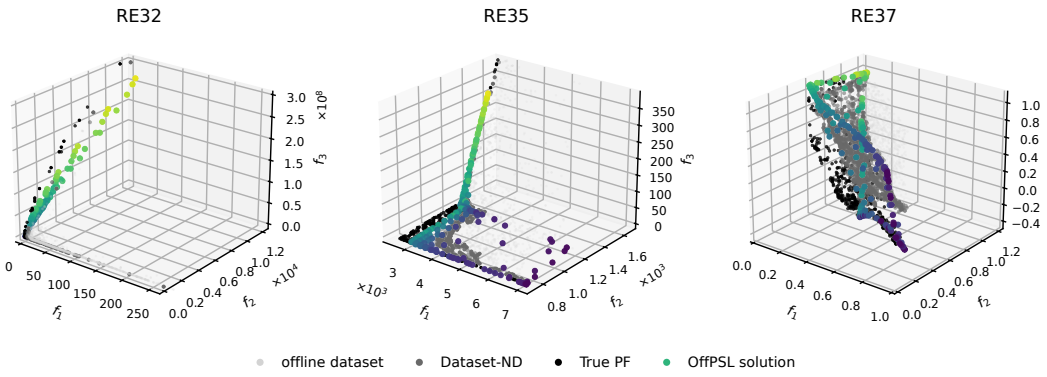

Figure 4: Qualitative objective-space diagnostics on three-objective RE tasks. The offline dataset is shown in light gray, Dataset-ND in dark gray, the benchmark true Pareto-front samples in black, and OffPSL solutions in color. These plots show the deterministic amortized map on three-objective fronts. On RE32, RE35, and RE37, OffPSL returns solutions that lie near high-quality nondominated regions, although the visualized coverage can be concentrated along lower-dimensional structures depending on the geometry of the offline data and the engineering task.

boundary. On RE37, they span a broader three-dimensional trade-off region. These views show the geometry of the returned sets, with coverage quantified by the accompanying set and interface metrics.

Together with the interface metrics, these plots characterize the learned preference-indexed Pareto-set approximation: ordered front traversal on supported regions, and non-uniform coverage or partial collapse on more irregular geometries.

F Conditional Analysis of Strict-Offline Pareto Set Amortization

This appendix gives formal statements and complete proofs of the three results in Section 4.5. Objectives are minimized coordinate-wise. All expectations in the first result are conditional on the fixed observed dataset.

F.1 Impossibility of unconditional coverage

Proposition 4 (Finite-data coverage impossibility)**.** *Let $\mathcal{X} \subset \mathbb{R}^d$ be compact with nonempty interior, and let the finite dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ be compatible with a continuous objective. An algorithm observes only $\mathcal{D}$ and returns a random nonempty set $X_K \subset \mathcal{X}$ with $|X_K| \le K$ for a positive integer $K < \infty$. For every $C > 0$, there is a continuous $f : \mathcal{X} \to \mathbb{R}^m$ compatible with $\mathcal{D}$, with singleton Pareto front $P_f^\star = \{v\}$, such that*

$$\mathbb{E}\,\mathrm{IGD}(\{v\}, f(X_K)) \ge C, \qquad \mathrm{IGD}(\{v\}, Y) := \min_{y \in Y} \|v - y\|_2.$$

The function class imposes no common objective-amplitude bound or common modulus of continuity.

Proof. Choose $2K$ pairwise disjoint open balls $U_1, \dots, U_{2K}$ inside $\mathrm{int}(\mathcal{X})$, avoiding all dataset points. Each realization of X_K intersects at most K of them. Taking expectations gives

$$\sum_{j=1}^{2K} \Pr(X_K \cap U_j \neq \varnothing) \le K.$$

Hence one ball U satisfies $\Pr(X_K \cap U = \varnothing) \ge 1/2$. Take a compatible continuous objective $\bar{f}$ and set $L = \min_{x \in \mathcal{X},\, r \in [m]} \bar{f}_r(x)$, which exists by compactness. Choose a continuous $\phi : \mathcal{X} \to [0, 1]$ supported inside U with $\phi(x_0) = 1$ at some $x_0 \in U$. For $a > 0$, define

$$v_a = (L - a)\mathbf{1}, \qquad f_a(x) = (1 - \phi(x))\bar{f}(x) + \phi(x) v_a.$$

This objective is continuous and agrees with every observation. Moreover, $f_a(x_0) = v_a$ and $f_a(x) \geq v_a$ coordinate-wise for every x; any distinct objective vector is dominated by v_a. Thus $P^\star_{f_a} = \{v_a\}$. On $X_K \cap U = \varnothing$, every candidate satisfies $f_a(x) = \bar{f}(x) \geq L\mathbf{1}$, so its distance to v_a is at least $a\sqrt{m}$. The output law depends only on the unchanged dataset, giving

$$\mathbb{E}\,\mathrm{IGD}(P^\star_{f_a}, f_a(X_K)) \geq a\sqrt{m}/2.$$

Set $a = 2C/\sqrt{m}$. The pair f_a, f_{2a} also gives indistinguishable observations with singleton fronts separated by $a\sqrt{m}$. □

This is a worst-case statement over an unrestricted continuous function class. Uniform guarantees require structure linking unobserved objectives to the data. A globally bounded objective range instead bounds IGD; dataset min–max normalization alone imposes no such global range constraint.

F.2 Conditional scalarized-regret decomposition

Fix a reference point z in the same normalized objective space as f and $\hat{f}$, and write

$$T_z(y;\lambda) = \max_r \lambda_r |y_r - z_r|, \qquad S_{\tau,z}(y;\lambda) = \tau \log \sum_{r=1}^{m} \exp\left(\frac{\lambda_r |y_r - z_r|}{\tau}\right).$$

Here T_z is the Tchebycheff score used in the main text, with its reference explicit.

Proposition 5 (Conditional scalarized regret)**.** *Let $\mathcal{X}$ and $S \subseteq \mathcal{X}$ be nonempty compact sets, with f continuous on $\mathcal{X}$ and $\hat{f}$ continuous on S. Fix $\lambda \in \Delta^{m-1}$, $\tau > 0$, and z, and assume $\sup_{x\in S} \|\hat{f}(x) - f(x)\|_\infty \leq \varepsilon_N$. Let $x_h = h^{\mathrm{feas}}_\pi(\lambda) \in S$ be the final evaluated output, including legal-box projection when used. Define*

$$\eta_\lambda = S_{\tau,z}(\hat{f}(x_h);\lambda) - \min_{x\in S} S_{\tau,z}(\hat{f}(x);\lambda),$$
$$\Delta_S(\lambda) = \min_{x\in S} T_z(f(x);\lambda) - \min_{x\in\mathcal{X}} T_z(f(x);\lambda).$$

Then

$$T_z(f(x_h);\lambda) - \min_{x\in\mathcal{X}} T_z(f(x);\lambda) \leq 2\|\lambda\|_\infty \varepsilon_N + \eta_\lambda + \tau \log m + \Delta_S(\lambda).$$

Proof. Continuity and compactness ensure that all minima exist. For $L_\lambda = \|\lambda\|_\infty$, the reverse triangle inequality coordinate-wise gives

$$|T_z(u;\lambda) - T_z(v;\lambda)| \leq L_\lambda \|u - v\|_\infty.$$

Factoring the largest exponential out of the smooth maximum gives $T_z \leq S_{\tau,z} \leq T_z + \tau \log m$. Choose $x^\star_S \in \arg\min_{x\in S} T_z(f(x);\lambda)$ and suppress the fixed z, λ, τ in the following chain:

$$\begin{aligned} T(f(x_h)) &\leq S(\hat{f}(x_h)) + L_\lambda \varepsilon_N \\ &= \min_{x\in S} S(\hat{f}(x)) + \eta_\lambda + L_\lambda \varepsilon_N \\ &\leq S(\hat{f}(x^\star_S)) + \eta_\lambda + L_\lambda \varepsilon_N \\ &\leq T(f(x^\star_S)) + \tau \log m + \eta_\lambda + 2L_\lambda \varepsilon_N. \end{aligned}$$

Subtracting the global minimum and using the definition of Δ_S proves the claim. □

Interpretation and scope. Δ_S is the cost of restricting the comparison to S. The surrogate gap η_λ includes representation, optimization, and regularization effects of training the shared map. All three quantities ε_N, η_λ, and Δ_S can vary with data size, coverage, and training. For the conservative surrogate $\hat{f} = \mu + \beta\sigma$, the uniform-error bound is an explicit assumption; ensemble disagreement alone does not establish it. Support penalties encourage proximity to offline designs, while membership of the final output in an error-controlled S remains a separate condition. In particular, the gap is evaluated after any projection.

The conclusion concerns a scalar score at a fixed reference point. The observed-data offset $z_\epsilon = -\epsilon\mathbf{1}$ need not be an ideal point for all unobserved objectives, so a Pareto-coverage conclusion requires additional conditions. The IGD studies in Appendix K examine the practical importance of data coverage through full-pipeline retraining; they do not separately estimate these bound terms.

F.3 Small-noise support-distance consistency

Proposition 6 (Consistency of the ideal denoising residual)**.** *Let μ be a probability measure on $\mathbb{R}^d$ with nonempty compact support $M = \operatorname{supp}\mu$. For $Z \sim \mu$ and independent $\xi_0 \sim \mathcal{N}(0, I_d)$, use the everywhere-defined Gaussian posterior mean*

$$d_s^*(y) = \mathbb{E}[Z \mid Z + s\xi_0 = y] = \frac{\int_M z \exp(-\|y-z\|^2/(2s^2))\,\mu(dz)}{\int_M \exp(-\|y-z\|^2/(2s^2))\,\mu(dz)}, \qquad s > 0.$$

Fix x with unique nearest support point $p = \pi_M(x)$ and distance $\delta = \|x - p\|$. Let A be independent of $\xi \sim \mathcal{N}(0, I_d)$, with $0 < a_{\min} \le A \le a_{\max} < \infty$. Then

$$R_t^*(x) := \mathbb{E}_{A,\xi}\|x - d_{tA}^*(x + tA\xi)\|^2 \longrightarrow \operatorname{dist}(x, M)^2 \quad \text{as } t \downarrow 0.$$

Proof. For $e > 0$, let $K_e = \{z \in M : \|z - p\| \ge e\}$. When K_e is nonempty, compactness and uniqueness give $\gamma_e = \min_{z \in K_e} \|x - z\|^2 - \delta^2 > 0$. Choose an open neighborhood V_e of p such that $\sup_{z \in V_e \cap M} \|x - z\|^2 \le \delta^2 + \gamma_e/4$. Its mass $b_e = \mu(V_e)$ is positive because $p \in \operatorname{supp}\mu$. Uniform continuity on the compact support gives a neighborhood of x such that, for every y in it and every $z \in M$,

$$\left|\|y - z\|^2 - \|x - z\|^2\right| \le \gamma_e/8.$$

On K_e the squared distance is at least $\delta^2 + 7\gamma_e/8$; on $V_e \cap M$ it is at most $\delta^2 + 3\gamma_e/8$. Bounding the posterior numerator by the former and its denominator using the latter yields

$$\mu_s(K_e \mid y) \le b_e^{-1} \exp(-\gamma_e/(4s^2)).$$

With $D = \operatorname{diam}(M)$, the posterior mean therefore satisfies

$$\|d_s^*(y) - p\| \le e + Db_e^{-1} \exp(-\gamma_e/(4s^2)).$$

If K_e is empty, the bound $\|d_s^*(y) - p\| \le e$ follows directly by averaging. Since $e > 0$ is arbitrary, $d_s^*(y) \to p$ whenever $s \to 0$ and $y \to x$. For each fixed finite (A, ξ), $tA \to 0$ and $x + tA\xi \to x$, so the residual converges pointwise to $\|x - p\|^2$. The posterior mean lies in the convex hull of M, which gives the global bound

$$\|x - d_s^*(y)\| \le \max_{z \in M} \|x - z\| < \infty.$$

Dominated convergence proves the expectation limit, including the Gaussian tail events. □

Support and uniqueness. The result concerns the actual support of μ. For an empirical measure it describes distance to the finite observed design set. A compact manifold of positive reach is one example: by the definition of reach, points at distance below the reach have unique nearest points. The theorem leaves points with nonunique projections outside its scope.

Learned-denoiser error. For a family of learned denoisers d_{ω_t}, define, at the same fixed x,

$$\epsilon_t(x)^2 = \mathbb{E}_{A,\xi}\|d_{\omega_t}(x + tA\xi, tA) - d_{tA}^*(x + tA\xi)\|^2,$$
$$R_t^\omega(x) = \mathbb{E}_{A,\xi}\|x - d_{\omega_t}(x + tA\xi, tA)\|^2.$$

When $\epsilon_t(x) < \infty$, the reverse triangle inequality in L^2 gives

$$\left|\sqrt{R_t^\omega(x)} - \sqrt{R_t^*(x)}\right| \le \epsilon_t(x).$$

Multiplying by the sum of the square roots, which is at most $2\sqrt{R_t^*(x)} + \epsilon_t(x)$, gives

$$|R_t^\omega(x) - R_t^*(x)| \le 2\sqrt{R_t^*(x)}\,\epsilon_t(x) + \epsilon_t(x)^2.$$

Consequently $\epsilon_t(x) \to 0$ implies $R_t^\omega(x) \to \operatorname{dist}(x, M)^2$. The required error is measured under perturbations around the candidate x; training error averaged over offline designs does not by itself control this quantity. These are pointwise results; uniform conclusions over an output set require uniform conditions.

Finite perturbation estimator. For fixed x, t and a fixed trained denoiser, take i.i.d. (A_p, ξ_p) and set

$$U_p = \|x - d_\omega(x + tA_p\xi_p, tA_p)\|^2, \qquad \widehat{R}^\omega_{t,P}(x) = P^{-1}\sum_{p=1}^{P} U_p.$$

If $\mathbb{E}U_1 < \infty$, linearity gives $\mathbb{E}\widehat{R}^\omega_{t,P}(x) = R^\omega_t(x)$. If the fourth residual moment is finite, write $v_t(x) = \mathrm{Var}(U_1)$. Independence eliminates cross-covariances, yielding

$$\mathrm{Var}(\widehat{R}^\omega_{t,P}(x)) = v_t(x)/P, \qquad \Pr\left(|\widehat{R}^\omega_{t,P}(x) - R^\omega_t(x)| > u\right) \le \frac{v_t(x)}{Pu^2} \quad (u > 0),$$

where the last step is Chebyshev's inequality. This connects the expectation to the Monte Carlo support penalty. The experimental scales $s \in [0.05, 0.20]$ and $P = 4$ are fixed operating settings evaluated empirically; the asymptotic result supplies a conditional mechanism interpretation, with finite-setting accuracy depending on denoiser error, noise scale, and residual variance.

Covariance penalty. For any $P \ge 2$ residual vectors $r_p \in \mathbb{R}^d$ with mean $\bar{r}$, the definitions of R_{sup} and R_{cov} give the exact identity

$$R_{\mathrm{sup}} = \|\bar{r}\|^2 + \frac{d(P-1)}{P} R_{\mathrm{cov}}.$$

Indeed, expanding $r_p = \bar{r} + (r_p - \bar{r})$ and summing cancels the cross terms, so $\sum_p \|r_p\|^2 = P\|\bar{r}\|^2 + \sum_p \|r_p - \bar{r}\|^2$. For i.i.d. residuals with finite second moments,

$$\mathbb{E}\sum_p \|r_p - \bar{r}\|^2 = P\mathbb{E}\|r_1\|^2 - P\mathbb{E}\|\bar{r}\|^2 = (P-1)\,\mathrm{tr}\,\mathrm{Cov}(r_1).$$

Thus $\mathbb{E}R_{\mathrm{cov}} = d^{-1}\,\mathrm{tr}\,\mathrm{Cov}(r_1)$. Adding this penalty increases the weight on residual variability; the surrogate-error and output-membership assumptions of Proposition 5 remain explicit.

G Extended Standard-Metric Comparisons

Normalized regret. For each task t let $\mathrm{HV}^\star_t$ and $\mathrm{HV}^{\min}_t$ be the best and worst task-mean HV over all compared methods, and define the normalized regret of method m as $(\mathrm{HV}^\star_t - \mathrm{HV}_{m,t})/(\mathrm{HV}^\star_t - \mathrm{HV}^{\min}_t) \in [0, 1]$. Mean normalized regret over the 23 continuous tasks (methods as sourced in Table 1): OffPSL 0.043; MOBO 0.117; MultipleModels-IOM 0.157; MultipleModels-ICT 0.194; MultiHead-PCGrad 0.215; MultipleModels 0.222; PCD 0.244; STCH-PSL 0.253; PGD-MOO and ParetoFlow follow.

Significance tests and replication. Table 16 reports win/tie/loss counts and paired Wilcoxon signed-rank tests with tasks as the paired unit and seeds averaged first. Holm correction is applied across six baseline comparisons, all significant at adjusted $p \le 0.0065$. Increasing OffPSL's replication from five to ten seeds changes task-mean HV by 1.15% on average and preserves the aggregate findings. The right half uses the best available per-task HV for PGD-MOO and ParetoFlow, defined as the maximum of each published value and its corresponding rerun value. All six comparisons remain significant under this definition ($p \le 0.0135$).

G.1 IGD comparison

Table 17 reports true-PF IGD for seven methods across 23 tasks. OffPSL has the best average rank (3.13). Paired Wilcoxon tests yield $p = 0.67$ against PCD, $p = 0.89$ against MOBO, $p = 0.25$ against PGD-MOO, and $p = 0.20$ against ParetoFlow, with a significant improvement over PSL-MOBO after correction. PSL-MOBO achieves the lowest IGD on DTLZ6, while PCD leads on DTLZ7. Across the suite, OffPSL combines significant HV gains with competitive IGD.

H Full-Scale Support-Prior Comparison

Tables 18 and 19 compare no-support, kNN, denoising-residual, and residual-plus-covariance variants on all 23 continuous tasks (5 seeds each; 460 runs) under one fixed default configuration; support

Table 16: Win/tie/loss of OffPSL on task-mean HV over the 23 continuous tasks (tie tolerance 0.01) with paired Wilcoxon signed-rank tests and Holm correction. Left: all methods from our runs under the shared evaluator (OffPSL/PCD/PGD-MOO at ten seeds; ParetoFlow/MOBO/STCH-PSL/PSL-MOBO at five). Right: best available per-task HV for PGD-MOO and ParetoFlow, taking the maximum of published and rerun values.

	Our runs		Best available HV	
OffPSL vs	W/T/L	Holm-adj. p	W/T/L	Holm-adj. p
PCD	17/5/1	0.0034	17/5/1	0.0020
PGD-MOO	16/5/2	0.0030	16/4/3	0.0082
ParetoFlow	22/0/1	0.0002	18/0/5	0.0135
MOBO	15/6/2	0.0065	15/6/2	0.0097
STCH-PSL	17/3/3	0.0031	17/3/3	0.0034
PSL-MOBO	21/1/1	$<$0.0001	22/0/1	$<$0.0001

Table 17: True-PF IGD (task means; DTLZ5–7 use the dataset-referenced IGD as no true-front reference exists). Bold marks the best method per task. OffPSL has the best average rank, but paired differences against PCD, MOBO, PGD-MOO, and ParetoFlow are not statistically significant.

Task	OffPSL	PCD	PGD-MOO	ParetoFlow	MOBO	STCH-PSL	PSL-MOBO
ZDT1	0.263	0.304	0.197	0.333	0.166	**0.004**	1.046
ZDT2	0.255	0.366	**0.136**	0.474	0.366	0.405	1.321
ZDT3	0.194	0.295	0.244	0.336	0.229	**0.032**	0.720
ZDT4	0.368	0.242	**0.150**	0.512	0.674	1.050	0.985
ZDT6	0.035	0.158	**0.019**	0.545	0.173	0.779	1.026
DTLZ1	**0.067**	0.077	0.073	0.104	0.098	0.084	0.076
DTLZ2	**0.068**	0.096	0.122	0.117	0.399	0.639	0.573
DTLZ3	0.165	0.174	0.165	0.189	0.241	**0.121**	0.549
DTLZ4	0.388	**0.232**	0.461	0.298	0.329	0.779	0.688
DTLZ5	0.083	**0.034**	0.073	0.046	0.409	0.535	0.720
DTLZ6	0.494	0.086	0.132	0.130	0.166	0.094	**0.076**
DTLZ7	0.459	**0.090**	0.168	0.144	0.221	0.772	0.799
RE21	0.061	0.059	0.618	0.116	**0.008**	0.019	**0.008**
RE22	0.089	**0.002**	0.168	0.004	0.063	0.197	0.228
RE25	0.001	**0.000**	**0.000**	0.009	0.151	0.084	0.213
RE32	0.125	0.258	**0.046**	0.198	0.114	0.417	0.436
RE34	0.304	**0.073**	0.503	0.077	0.094	0.126	0.308
RE35	0.093	0.093	1.154	0.066	**0.034**	0.238	0.485
RE36	**0.061**	0.397	0.545	0.285	0.076	0.258	0.904
RE37	0.064	0.141	0.105	0.162	**0.056**	0.108	0.074
RE41	0.147	0.227	0.325	0.212	**0.119**	0.197	0.213
RE42	0.090	0.385	0.506	0.326	**0.087**	0.187	0.484
RE61	0.242	0.342	0.496	0.305	0.139	**0.127**	1.453
Avg. rank	3.13	3.30	4.04	4.13	3.26	4.30	5.83

weights are calibrated offline only. These independently trained five-seed variants use the unified default configuration; the main benchmark aggregates ten seeds under its task configurations. The denoising residual outperforms the kNN penalty on 17/23 tasks in HV (mean ΔHV $+1.32$, median $+0.33$) and 18/23 in IGD at $+6\%$ training cost, while the kNN penalty suppresses clipping most strongly. Relative to no support, the denoising prior is HV-neutral (median per-task ΔHV -0.013) but reduces the duplicate rate by more than half, reduces clipping, and improves coverage. Structurally, the kNN penalty requires retaining the dataset during Pareto-set training and has piecewise-constant gradients in the neighbor set, whereas the denoiser is a fixed-size smooth surrogate for support with no neighbor hyperparameter.

I Factorial Ablation of Conservatism and Support

Table 20 crosses conservatism variants {none, coordinate-wise, scalarize-then-penalize} with support variants {none, denoising, kNN} on five representative tasks under the same unified default configuration (five seeds per variant). The Full row is the within-study control. Support regularization is the primary stabilizing component: design-level clipping decreases from 0.52 to 0.33 and duplicate rate from 0.057 to 0.020 relative to the plain variant, with HV essentially unchanged. Coordinate-wise pessimism and scalarize-then-penalize yield comparable performance across all

Table 18: Full-scale support-prior comparison: 23 continuous tasks $\times$ 5 seeds per variant (460 runs), one fixed default configuration, offline-only support-weight calibration ($\epsilon = 0.2$, map width 256). This is an independent controlled study using seeds 1000–5000, with Residual+cov as its within-study reference; Tables 4–6 report the ten-seed benchmark. Ranks are average ranks over the 23 tasks.

Variant	HV rank ↓	IGD rank ↓	Clip rank ↓	Duplicate ↓	Bins ↑	Train (s)
No support	1.83	2.43	2.96	0.019	28.5	317
kNN penalty	3.70	3.35	1.87	0.002	25.6	327
Denoising residual	2.04	2.00	2.52	0.008	29.4	348
Residual+cov	2.43	2.22	2.65	0.008	29.4	335

Table 19: Support-prior comparison: per-task HV (mean±std over 5 seeds) under the unified offline-calibrated default configuration ($\epsilon = 0.2$, map width 256), using seeds 1000–5000. Residual+cov is the reference for this independent ablation; Tables 4–6 report the ten-seed benchmark. Bold marks the best variant per task.

Task	No support	kNN penalty	Denoising residual	Residual+cov
ZDT1	4.48±0.45	**4.73±0.03**	4.72±0.13	4.72±0.13
ZDT2	**5.55±0.09**	**5.55±0.09**	**5.55±0.09**	**5.55±0.09**
ZDT3	**5.65±0.10**	5.30±0.28	5.63±0.21	5.63±0.21
ZDT4	4.99±0.12	**5.44±0.07**	5.08±0.05	5.06±0.04
ZDT6	3.78±0.00	3.73±1.10	**3.98±0.46**	**3.98±0.45**
DTLZ1	**10.65±0.00**	10.64±0.00	**10.65±0.00**	**10.65±0.00**
DTLZ2	12.32±0.15	12.43±0.00	**12.44±0.00**	**12.44±0.00**
DTLZ3	**9.89±0.01**	9.89±0.01	**9.89±0.00**	**9.89±0.00**
DTLZ4	**16.22±1.27**	14.70±0.05	14.83±0.00	14.83±0.00
DTLZ5	10.73±0.01	10.73±0.01	**10.75±0.00**	**10.75±0.00**
DTLZ6	**11.12±0.03**	10.54±0.02	11.08±0.01	11.08±0.01
DTLZ7	**10.65±0.09**	10.22±0.05	10.62±0.08	10.62±0.08
RE21	**4.59±0.02**	3.82±0.41	4.36±0.03	4.35±0.03
RE22	**4.84±0.00**	4.47±0.23	**4.84±0.00**	**4.84±0.00**
RE25	**4.83±0.01**	4.76±0.03	4.79±0.00	4.79±0.00
RE32	**10.65±0.00**	10.24±0.28	10.56±0.03	10.56±0.03
RE34	9.57±0.61	8.94±0.31	**9.92±0.01**	**9.92±0.01**
RE35	**10.58±0.01**	10.16±0.11	10.44±0.04	10.44±0.04
RE36	**10.22±0.14**	7.54±0.15	9.30±0.36	9.29±0.39
RE37	6.46±0.38	5.88±0.11	**6.54±0.02**	**6.54±0.02**
RE41	**20.69±0.90**	18.34±0.42	20.43±0.04	20.42±0.03
RE42	**22.80±0.01**	18.08±0.42	21.69±0.24	21.68±0.24
RE61	**109.13±0.04**	87.93±7.24	106.30±0.53	106.29±0.54

five tasks ($\max |\Delta\text{HV}| = 0.013$, with IGD differences in the fourth decimal place). This supports robustness to the conservative parameterization and the coordinate-wise formulation as a simple default.

Table 21 additionally sweeps the scalarizer offset, the support prior, and the diversity term on the five representative tasks used above, ranking variants within each group; it is the earlier, smaller-scale companion to the full-benchmark study of Appendix H.

J DTLZ3 Root-Cause Diagnostic

Table 22 compares three diagnostic configurations on DTLZ3 and ZDT4 over five seeds: the learned surrogate with the amortized map (A), the learned surrogate with independent per-preference search (B), and the true objective with independent search (C). Holding independent search fixed, replacing the surrogate with the true objective reduces IGD from 0.126 to 0.012 on DTLZ3 and from 0.533 to 0.018 on ZDT4. These reductions identify objective-model accuracy as the main determinant of the gap in this diagnostic. The A–B differences are smaller and task dependent. On DTLZ3, HV is nearly saturated (9.900 for oracle search and 9.891 for OffPSL), so IGD better resolves the changes in front coverage.

Cell C uses 5.2×10^5 true evaluations per seed and retains 43% distinct objective vectors under independent optimization without diversity regularization. Cell B omits the support and diversity terms, so the A–B comparison jointly changes parameter sharing and regularization. These configurations characterize the objective-model and optimization contributions to the observed results.

Table 20: Factorial ablation of conservatism and support (5 tasks $\times$ 5 seeds per variant, seeds 1000–5000) under the unified offline-calibrated default configuration ($\epsilon = 0.2$, map width 256). Full (coordinate-wise) is the reference for this independent ablation; Tables 4–6 report the ten-seed benchmark. Per-task columns report mean HV; the last three columns aggregate over all runs. Coordinate-wise pessimism and scalarize-then-penalize yield comparable results (max $|\Delta\text{HV}| = 0.013$ per task).

Variant	ZDT2	ZDT6	DTLZ3	RE22	RE42	IGD	Clip	Dup.
Plain (no pess., no supp.)	5.547	3.772	9.894	4.840	22.800	0.149	0.52	0.057
Pessimism only	5.548	3.777	9.889	4.840	22.800	0.152	0.51	0.039
Support only	5.547	3.984	9.895	4.840	21.731	0.158	0.37	0.020
Full (coordinate-wise)	5.547	3.979	9.892	4.840	21.682	0.169	0.33	0.020
Scalarize-then-penalize	5.547	3.979	9.891	4.840	21.695	0.168	0.33	0.018
Pessimism + kNN	5.547	3.728	9.886	4.473	18.082	0.261	0.30	0.009

Table 21: Component ablation on five representative tasks. Ranks are computed within each ablation group over task–seed results. ΔHV is measured relative to the corresponding full-objective control within this ablation study.

Group	Variant	Avg. HV rank ↓	Avg. ΔHV vs Full ↑	Avg. true-PF IGD rank ↓	Avg. clip frac. ↓
Scalarizer	Raw dataset	4.60	-1.086	4.48	0.081
	Minmax $\epsilon = 0$	3.52	-0.001	3.24	0.224
	Minmax $\epsilon = 0.1$	3.80	-0.072	3.36	0.214
	Minmax $\epsilon = 0.2$	2.48	+0.000	3.56	0.256
	Minmax $\epsilon = 0.3$	3.00	+0.212	2.88	0.236
	Minmax $\epsilon = 0.5$	3.60	+0.060	3.48	0.250
Support prior	No support	2.36	+0.143	2.48	0.305
	Residual only	2.08	-0.074	1.96	0.235
	Residual + covariance	1.56	+0.000	1.56	0.262
Diversity	No diversity	2.04	-0.087	1.72	0.254
	Objective diversity	1.84	+0.000	2.12	0.262
	Decision diversity	2.12	+0.395	2.16	0.319

K Data Size and Coverage Structure

Table 23 reports two controlled studies with the full pipeline (surrogate, support model, and Pareto-set map) retrained per condition. *Data-size scaling*: random subsampling at $10/25/50/100\%$ shows early saturation – at 50% the mean true-PF IGD ratio to the full data is 0.98; at 10% it degrades to $1.37\times$ (ZDT3 $1.97\times$, ZDT6 $1.69\times$). *Coverage structure*: holding the point count fixed at the 50% level and changing only which region of the nondominated objective space is covered degrades IGD far more – on RE22, keeping only the central region gives $28.5\times$ the uniform-subsample IGD, a contiguous hole gives $14.7\times$, and keeping only extremes gives $4.8\times$. RE22 is precisely the partial-collapse failure case identified in the main text; these results explain it and highlight the importance of Pareto-relevant coverage alongside sample count. These IGD measurements assess full-pipeline retraining rather than estimating the individual terms in Proposition 5. Hole effects are task-dependent (negligible on ZDT2/DTLZ3 at this deletion radius, where the removed neighborhoods contain few points), and we report the deletion metadata with each run.

L Scalarizer Study

Table 24 swaps the smooth Tchebycheff scalarizer for linear, augmented-Tchebycheff, and PBI alternatives with all other components fixed (ZDT2, ZDT3, RE21; 5 seeds each). All variants train stably through the same pipeline, confirming the framework is scalarizer-agnostic. Augmented-Tchebycheff is indistinguishable from STCH throughout; the linear scalarizer degrades exactly where decomposition theory predicts, most visibly on the disconnected ZDT3 front (HV -0.30, IGD $2.4\times$ worse than STCH); PBI is geometry-dependent – best on the well-behaved RE21 (HV 4.52, IGD 0.060, zero clipping) but weaker on ZDT3. STCH and augmented-Tchebycheff remain the safest defaults across front geometries.

Table 22: Objective-model and optimization diagnostic on DTLZ3 and ZDT4 (5 seeds: 1000–5000), measured by true-PF IGD. Cell A re-evaluates saved OffPSL candidates with the diagnostic evaluator shared by B and C; Table 17 reports the ten-seed benchmark. Cell B uses independent surrogate optimization without support or diversity regularization. ‡Cell C uses direct true-objective optimization with 5.2×10^5 evaluations per seed.

Cell	DTLZ3	ZDT4
A: surrogate + amortized map (OffPSL)	0.166	0.388
B: surrogate + per-preference search	0.126	0.533
C: true oracle + per-preference search‡	0.012	0.018

Table 23: Data-size scaling (left; random subsampling with full retraining) and coverage-structure stress test (right; fixed point count, different covered regions), true-PF IGD averaged over 5 seeds. Which region is covered matters far more than how many points are kept.

Task	10%	25%	50%	100%	Uniform	Hole	Extremes	Central
ZDT2	0.403	0.403	0.403	0.403	0.403	0.403	0.403	0.403
ZDT3	0.322	0.234	0.149	0.164	0.173	0.116	0.173	0.173
ZDT6	0.171	0.137	0.133	0.101	0.132	0.129	0.112	0.199
DTLZ3	0.186	0.143	0.130	0.137	0.126	0.135	0.118	0.119
RE22	0.005	0.004	0.005	0.006	0.004	0.056	0.018	0.108

M Licenses for Used Assets

Table 25 lists the external assets used in our experiments and their licenses. Our released implementation will be licensed under the MIT License. Third-party assets are used only for benchmarking, baseline comparison, evaluation, or standard software dependencies.

Table 24: Scalarizer ablation over 3 tasks (ZDT2, ZDT3, RE21): the smooth Tchebycheff scalarizer of the training objective is swapped for a linear weighted sum, augmented Tchebycheff (ρ=0.1, same smooth-max temperature τ), or PBI (θ=5.0), with all other components fixed at the unified default configuration (ϵ=0.2, β=1.0, α_{sup}=0.1, α_{cov}=0.05, α_{div}=0.05, width 256), seeds 1000, 2000, 3000, 4000, 5000. The STCH row reuses the denoising-residual-plus-covariance runs from the support-prior comparison (Appendix H) with identical base configurations. Values are mean $\pm$ standard deviation across task–seed runs. IGD is true-PF IGD. Duplicate is the pairwise model-space design duplicate rate, and occupied bins counts f_1 bins after dataset min–max objective normalization.

Scalarizer	HV ↑	IGD ↓	ClipFrac-design ↓	Duplicate ↓	Occupied bins (50) ↑
STCH	5.18±0.62	0.226±0.186	0.443±0.294	0.031±0.057	33.9±20.9
Linear (weighted sum)	5.02±0.65	0.308±0.183	0.396±0.285	0.037±0.063	18.7±19.8
Augmented Tchebycheff ($\rho = 0.1$)	5.19±0.61	0.224±0.188	0.466±0.306	0.036±0.062	33.3±20.7
PBI ($\theta = 5.0$)	5.16±0.47	0.201±0.130	0.348±0.381	0.003±0.010	34.1±14.3

Table 25: Used assets and their licenses.

Type	Asset	License	URL	Usage
Code	OffPSL (ours)	MIT License	https://github.com/jicheng9617/offPSL	Released implementation
Benchmark / Code	Off-MOO-Bench / `offline-moo`	Not explicitly specified in the public repository; verify before redistribution	https://github.com/lamda-bbo/offline-moo	Benchmark, evaluator, and baseline reference
Dataset	ZDT, DTLZ, and RE offline datasets from Off-MOO-Bench	Same as the Off-MOO-Bench release; verify upstream licenses before redistribution	https://github.com/lamda-bbo/offline-moo	Training and evaluation
Code	PGD-MOO	MIT License	https://github.com/yannadani/pgd_moo	Baseline comparison
Code	ParetoFlow	MIT License	https://github.com/mila-iqia/ParetoFlow	Baseline comparison
Code	STCH / STCH-PSL	MIT License	https://github.com/Xi-L/STCH	Pareto-set-learning baseline
Code	PSL-MOBO	MIT License	https://github.com/Xi-L/PSL-MOBO	Pareto-set-learning baseline
Code / Library	BoTorch	MIT License	https://github.com/meta-pytorch/botorch	Offline MOBO and acquisition optimization
Code / Library	`pymoo`	Apache-2.0 License	https://github.com/anyoptimization/pymoo	MOO utilities, test problems, and reference fronts
Code / Benchmark	Design-Bench	MIT License	https://github.com/brandontrabucco/design-bench	Offline optimization benchmark reference

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction state the scope as deterministic Pareto set amortization for offline multi-objective optimization. Section 5 and Appendices D and G report the HV comparisons on 23 ZDT, DTLZ, and RE tasks. Appendix G.1 reports the IGD comparisons, and Section 4.5 states the conditional theoretical results.

2. **Limitations**

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

   Answer: [Yes]

   Justification: The conclusion and Appendix F discuss the limits imposed by surrogate error and offline data coverage. Appendix J examines multimodal landscapes, Appendix K examines missing Pareto-relevant data, and Appendix E.5 reports the training–inference trade-off. Mixed design spaces are identified as future work.

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: [Yes]

   Justification: Appendix F provides formal statements, explicit assumptions, and complete proofs for all theoretical results summarized in Section 4.5: finite-data coverage impossibility, conditional scalarized-regret decomposition, and small-noise support-distance consistency. It distinguishes the ideal denoiser, its learned approximation, and the finite-perturbation estimator, and specifies the scope of each result.

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: Section 5 and Appendices B, C, and D disclose the offline evaluation protocol, candidate budget, benchmark families, normalization, metrics, hyperparameters, checkpointing rule, and per-task results. Appendix M lists the implementation and external assets.

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: Appendix M lists the OffPSL implementation repository, its intended MIT license, and the external benchmark and baseline repositories. Appendices B and C specify the training and evaluation procedures.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Section 5 describes the benchmark suites, candidate budget, baselines, oracle-evaluation protocol, and reporting convention. Appendix C gives the metrics, projection diagnostics, and hyperparameters, while Appendix B specifies normalization and the training and inference algorithm.

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: Appendix D reports mean $\pm$ sample standard deviation over ten seeds for OffPSL, PCD, and PGD-MOO and five seeds for the other rerun methods. Appendix G reports paired task-level Wilcoxon tests with Holm correction. Published baseline results are marked separately.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix E.5 specifies the CPU, GPU, and GPU memory and reports wall-clock training and inference times, including the time required to generate 256 candidates.

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research uses public benchmark tasks and offline datasets and does not involve human subjects or personally identifiable research data. Appendix A discusses safety, misuse, validation, and governance considerations for responsible use of offline optimization methods.

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: Appendix A discusses potential positive impacts in scientific and engineering design, including reduced need for expensive or unsafe online experimentation. It also discusses negative impacts, including dataset bias, surrogate miscalibration, unsafe deployment in high-stakes domains, and possible misuse for harmful design optimization.

11. **Safeguards**

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

    Answer: [N/A]

    Justification: The paper does not release high-risk pretrained models, image generators, language models, scraped datasets, or other assets that require special controlled-release safeguards. The released asset is an optimization implementation evaluated on benchmark tasks, while Appendix A discusses application-level validation and governance safeguards.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Appendix M lists the external code, benchmark, dataset, and library assets used in the experiments, together with URLs, usage descriptions, and licenses where available. For assets whose upstream licenses are not explicitly specified, the paper notes this uncertainty and states that the licenses should be verified before redistribution.

13. **New assets**

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

    Answer: [Yes]

    Justification: Appendix M lists the OffPSL implementation repository and its intended MIT license. The method, training objective, inference procedure, hyperparameters, evaluation protocol, and limitations are documented in the main text and Appendices B, C, and F.

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, user studies, or experiments with human subjects. All experiments are conducted on offline optimization benchmarks and previously evaluated datasets.

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 work does not involve crowdsourcing or research with human subjects, so IRB approval or equivalent human-subjects review is not applicable.

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 core method, experiments, and scientific contribution do not use LLMs as an important, original, or non-standard component. Any use of LLMs solely for writing, editing, or formatting would not require declaration under the checklist instruction.