FROM FIXED ARCHIVES TO NEW OPERATING CONDITIONS: OFFLINE PARAMETRIC MULTI-OBJECTIVE OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Many engineering systems must operate under varying conditions, such as turbines facing different wind speeds or vehicles carrying different payloads, and the designs that are best for one condition are rarely best for another. Because cost, safety, and performance must be balanced at once, each condition poses its own multi-objective problem with its own Pareto set, and repeating the optimization for every new condition is too slow for timely deployment. In practice, substantial historical evaluation data are often available, which motivates reusing them offline to reconstruct trade-off solutions for previously unobserved conditions without new evaluations. The challenge in this offline parametric multi-objective optimization (Offline PMOO) setting is to generalize to unseen conditions and to recover diverse trade-offs without exploiting model errors in regions with sparse data, which neither current offline optimization nor parametric multi-objective methods address. We therefore formalize Offline PMOO and construct a benchmark of 27 tasks, ranging from synthetic test functions to engineering applications. After investigating the limitations of task-conditioned surrogate models in this regime, we propose a generative refinement strategy, instantiated via Parametric Scalarized Diffusion (PSD) and Parametric Pareto Flow Matching (PPFM), which learns a conditional prior from archived elite designs and uses surrogate models purely to steer refinement, so that candidates do not drift into unsupported regions. PPFM and PSD rank first and second on the synthetic and shared-component engineering suites. Our evaluation also maps the limits of learned solvers: surrogate-only models are preferable on a conditional simulator suite, and zero-learning archive reuse suffices for many-objective problems where Pareto dominance loses its discriminative power.

1 INTRODUCTION

Many design problems in robotics and engineering are not solved once. A walking robot that must keep moving after damage draws on a repertoire of behaviours evaluated before deployment, and the best behaviour changes with the damage it suffers (Cully et al., 2015). A soft robot co-designed for locomotion has a different optimal body for every environment it is placed in (Wang et al., 2023), and a mechanical component optimized across varying contexts has a Pareto set that moves with the context (Makatura et al., 2021; Cheng et al., 2025). In each case the objective vector $F(\boldsymbol{x}, \boldsymbol{\theta})$ depends on an externally specified task parameter $\boldsymbol{\theta}$ in addition to the design $\boldsymbol{x}$, so that every value of $\boldsymbol{\theta}$ defines one member of a related family of multi-objective problems. This structure is studied as parametric multi-objective optimization (PMOO) (Makatura et al., 2021; Cheng et al., 2025). It also appears as dynamic multi-objective optimization when the parameter is time (Deb et al., 2007; Zhou et al., 2013; Muruganantham et al., 2015; Jiang et al., 2018), as contextual and multi-task Bayesian optimization when evaluations can be allocated across conditions (Krause & Ong, 2011; Swersky et al., 2013; Pearce & Branke, 2018; Chowdhury & Gopalan, 2021), as multi-scenario and multi-parametric optimization in engineering design and process systems (Deb et al., 2018; Pistikopoulos et al., 2007), and as parametric Pareto-set learning when one model is trained to answer any condition (Makatura et al., 2021; Weaver-Rosen & Malak, 2023; Cheng et al., 2025; 2026; Wei et al., 2025).

These settings share one requirement: the objectives can be evaluated, or at least sampled, while the optimization runs. In practice that requirement often expires. An evaluation campaign ends, a

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

simulator is retired, or an experimental budget is exhausted, and what remains is a historical archive of designs, the conditions under which they were evaluated, and the objective values observed (Liao et al., 2008; Cully et al., 2015; Wang et al., 2023). Later design requests then arrive for conditions the campaign never covered. Offline optimization studies exactly this situation for a fixed objective, proposing good designs from a static dataset without querying the objective again (Trabucco et al., 2021; Kim et al., 2026), and offline multi-objective optimization extends it to recovering a set of trade-offs (Yuan et al., 2025a; Shrestha et al., 2026; Annadani et al., 2025). What has not been studied is the case in which both constraints hold at once: recovering a Pareto set at a *new* condition from an archive collected at *other* conditions, with no evaluation at the new one.

Existing tools do not transfer cleanly to this offline parametric setting. Reusing the nondominated designs of the nearest archived condition provides a starting point, but those designs need not remain nondominated once the condition changes. Fitting a surrogate over the joint design–parameter space and optimizing it at the new condition adapts the search, yet a surrogate optimized freely is known to exploit its own errors and to propose designs that look good to the model and fail under the true objectives (Trabucco et al., 2021; Skalse et al., 2022). Parametric Pareto-set learning amortizes the whole family in one model, but was developed for settings that permit new evaluations, and its inverse map inherits every surrogate error when it cannot (Cheng et al., 2026). This work addresses the whole problem: an archive-anchored solver that generalizes across conditions without any new evaluation, together with the benchmark and the conditions needed to judge it.

We therefore study *offline parametric multi-objective optimization* (Offline PMOO) systematically. Given a fixed archive of design, task-parameter, and objective tuples, the optimizer must return a finite approximation to the Pareto set at a queried $\boldsymbol{\theta}$ without further evaluations. A task parameter determines the objectives under which a design is judged, and a preference vector selects one trade-off within that condition. We make four contributions.

- We formulate Offline PMOO, state the conditions under which it is well posed, and prove that no zero-query method admits an unconditional guarantee. To make solvers comparable, we introduce 27 benchmark instances built from widely used synthetic test functions and common engineering applications, with a held-out-task evaluation protocol.
- We systematically study how task-conditioned surrogate models affect offline Pareto reconstruction, covering concatenation, hypernetwork, and low-rank hypernetwork conditioning, end-to-end and objective-wise outputs, and standard and robust multi-objective training, against three oracle-free controls: nearest-archive reuse, warm-start-only search, and parametric Pareto-set learning.
- We develop warm-started generative Pareto refinement, which learns a task-conditioned prior over archived elite designs and uses the surrogate only to start and steer refinement, instantiated with diffusion and flow-matching models as Parametric Scalarized Diffusion (PSD) and Parametric Pareto Flow Matching (PPFM).
- We characterize solver performance across the three suites and under changes in archive coverage: the proposed methods rank first and second on the synthetic and shared-component suites, a surrogate solver ranks first on the conditional simulator suite, and nearest-archive reuse ranks first on the nine-objective problems, where dominance no longer discriminates among archived designs.

2 Preliminaries and Problem Setup

We consider minimization throughout. Parametric multi-objective optimization (PMOO) concerns a family of multi-objective problems indexed by an exogenous task parameter $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p$,

$$\min_{\boldsymbol{x}\in\mathcal{X}} F(\boldsymbol{x}, \boldsymbol{\theta}) = \big(f_1(\boldsymbol{x}, \boldsymbol{\theta}), \ldots, f_m(\boldsymbol{x}, \boldsymbol{\theta})\big), \tag{1}$$

where $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ is the design variable and $F(\boldsymbol{x}, \boldsymbol{\theta}) \in \mathbb{R}^m$ the objective vector. The parameter $\boldsymbol{\theta}$ is not optimized. It specifies the operating condition, environment, or task instance under which a design is evaluated, and the feasible domain $\mathcal{X}$ is shared across task parameters. For a fixed $\boldsymbol{\theta}$, a design $\boldsymbol{x}_a$ Pareto dominates $\boldsymbol{x}_b$, written $\boldsymbol{x}_a \prec_{\boldsymbol{\theta}} \boldsymbol{x}_b$, if $f_i(\boldsymbol{x}_a, \boldsymbol{\theta}) \le f_i(\boldsymbol{x}_b, \boldsymbol{\theta})$ for all $i \in \{1, \ldots, m\}$ with strict inequality for at least one objective. The task-conditioned Pareto set and Pareto front are

$$\mathcal{X}^\star(\boldsymbol{\theta}) = \{\boldsymbol{x} \in \mathcal{X} : \nexists \boldsymbol{x}' \in \mathcal{X} \text{ such that } \boldsymbol{x}' \prec_{\boldsymbol{\theta}} \boldsymbol{x}\}, \qquad \mathcal{Y}^\star(\boldsymbol{\theta}) = F\big(\mathcal{X}^\star(\boldsymbol{\theta}), \boldsymbol{\theta}\big). \tag{2}$$

Each $\boldsymbol{\theta}$ thus defines its own Pareto set, and the goal of PMOO is to recover or approximate $\mathcal{X}^\star(\boldsymbol{\theta})$ across the task domain rather than for a single fixed instance.

Offline PMOO. This work studies PMOO in the offline setting. The optimizer is given only a fixed archive

$$\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{\theta}_i, \boldsymbol{y}_i)\}_{i=1}^{N}, \qquad \boldsymbol{y}_i = F(\boldsymbol{x}_i, \boldsymbol{\theta}_i), \tag{3}$$

where the archive may contain dominated, suboptimal, and unevenly distributed samples over the joint design–parameter space $\mathcal{X} \times \Theta$. No additional calls to the true objective function F are allowed during training or inference. At test time, for a held-out task parameter $\boldsymbol{\theta}_{\text{eval}}$, a method must return a finite approximation

$$\mathcal{A}(\mathcal{D}, \boldsymbol{\theta}_{\text{eval}}) = S_{\boldsymbol{\theta}_{\text{eval}}} = \left\{\hat{\boldsymbol{x}}^{(1)}, \ldots, \hat{\boldsymbol{x}}^{(K)}\right\} \subseteq \mathcal{X} \tag{4}$$

to the unknown Pareto set $\mathcal{X}^\star(\boldsymbol{\theta}_{\text{eval}})$. The ground-truth oracle is used only by the benchmark evaluator to score the returned designs. It is not available to the optimization method itself.

Offline PMOO is challenging because the method must generalize simultaneously across designs and task parameters: the support mismatch lives in the joint space $(\boldsymbol{x}, \boldsymbol{\theta})$, so a design that is well supported for one parameter value may be unsupported or poor for another, and no new evaluation can repair sparse regions of Θ or validate uncertain candidates.

Deployment regime. The zero-query constraint describes a deployment regime. The oracle *was* available when the archive was collected, through past optimization campaigns, legacy simulators, or a one-off compute budget, and is unavailable at deployment, when a system must answer many new operating conditions $\boldsymbol{\theta}$ without launching a new campaign for each. Offline PMOO thus stands to online and expensive PMOO as offline model-based optimization stands to Bayesian optimization (Appendix A).

2.1 Applicability Conditions and What Is Achievable

Because no method can be evaluated against an oracle at deployment, it matters precisely which conditions a solver relies on. Throughout, let $G_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{\theta})$ denote the augmented Tchebycheff scalarization of the *true* objectives F under preference $\boldsymbol{w}$, and let $\widehat{G}_{\boldsymbol{w}}$ denote its surrogate counterpart formed from the learned model $\hat{F}_\psi$. The distinction matters below.

- **A1 (In-domain interpolation.)** The queried $\boldsymbol{\theta}_{\text{eval}}$ lies inside the training parameter domain, with a moderate nearest-task distance $h(\boldsymbol{\theta}) = \min_i \|\boldsymbol{\theta} - \boldsymbol{\theta}_i\|$.
- **A2 (Local task regularity.)** For each preference $\boldsymbol{w}$, the true scalarization $G_{\boldsymbol{w}}(\boldsymbol{x}, \cdot)$ is locally $L_{\boldsymbol{\theta}}$-Lipschitz in $\boldsymbol{\theta}$ over the supported neighbourhood. The conditional Pareto set may be disconnected, multimodal, or change topology, provided the relevant branches are witnessed somewhere in the archive.
- **A3 (Archive support.)** Near $\boldsymbol{\theta}_{\text{eval}}$, the archive covers the main trade-off regions of the front, so that each requested preference direction has a usable archived starting design.

A fourth condition, local surrogate reliability (A4), concerns only the refinement stage: usable surrogate values and gradients are needed near the archived starting designs and along the refinement path, not globally on $\mathcal{X} \times \Theta$. It enters the error decomposition below and is stated with the formal versions of A1–A3 in Appendix B. The following statement, proved in Appendix B, shows that some such restriction is unavoidable for *any* zero-query method.

Theorem 1 (Unidentifiability without queries)**.** *Let $\mathcal{D}$ be any finite archive whose task parameters exclude $\boldsymbol{\theta}^\star$. Then there exist two objective families $F^{(1)}$ and $F^{(2)}$, both continuous on $\mathcal{X} \times \Theta$, that agree on every archived tuple, and indeed on all of $\mathcal{X}$ at every archived parameter, yet whose conditional Pareto sets at $\boldsymbol{\theta}^\star$ are arbitrarily far apart.*

Consequently every guarantee must be conditional on assumptions of the A1–A4 type, and within them Appendix B decomposes the true scalarized gap of the design $\hat{\boldsymbol{x}}$ returned for preference $\boldsymbol{w}$,

$$\begin{aligned} & G_{\boldsymbol{w}}(\hat{\boldsymbol{x}}, \boldsymbol{\theta}) - G^{\star}_{\boldsymbol{w}}(\boldsymbol{\theta}) \\ & \lesssim \underbrace{\varepsilon_{\mathrm{arch}}}_{\text{starting-design quality}} + \underbrace{2L_{\boldsymbol{\theta}}\, h(\boldsymbol{\theta})}_{\text{task distance (A1–A2)}} + \underbrace{2\varepsilon_{\mathrm{sur}}}_{\text{surrogate value/gradient (A4)}} \\ & \quad + \underbrace{\varepsilon_{\mathrm{ideal}}}_{\text{ideal-point estimate}} + \underbrace{\varepsilon_{\mathrm{opt}}}_{\text{optimization residual}} + \underbrace{L_{\boldsymbol{x}}\, \delta_{\mathrm{ref}}}_{\text{elite-support displacement (A3)}} . \end{aligned} \tag{5}$$

The bound follows one candidate through the solver. Its first two terms are fixed by the archive, namely the suboptimality $\varepsilon_{\mathrm{arch}}$ of the best archived design at the nearest archived parameter and the cost $2L_{\boldsymbol{\theta}}h(\boldsymbol{\theta})$ of transferring it to the query, so no zero-query method can reduce them, in line with Theorem 1. The terms set by the solver pull against each other: a longer surrogate search lowers $\varepsilon_{\mathrm{opt}}$ but leaves the region where $\varepsilon_{\mathrm{sur}}$ is small (A4), and returning to the archive's support costs $L_{\boldsymbol{x}}\delta_{\mathrm{ref}}$. The bound thus favours improving a good archived design only within that support, which is how Section 5 assigns one mechanism to each term and what the diagnostics of Section 6 probe.

3 Benchmark, Datasets, and Evaluation

We construct a benchmark for offline PMOO that covers controlled synthetic functions, engineering design problems, and real-world conditional design tasks. Each problem instance defines a continuum or finite family of task-parameterized MOO problems indexed by $\boldsymbol{\theta}$, and methods receive only a fixed offline archive and must return Pareto-set approximations for queried task parameters without accessing the true objective during inference.

Benchmark suites. Table 1 summarizes the three benchmark suites. The DF suite (Cheng et al., 2026; Jiang et al., 2018) provides analytic synthetic problems with controlled parametric changes in the Pareto set and Pareto front. The RE suite (Tanabe & Ishibuchi, 2020) converts real-world engineering MOO problems into shared-component PMOO tasks by fixing selected structural design variables as task parameters. The REAL suite (Makatura et al., 2021) contains real-world conditional design problems where $\boldsymbol{\theta}$ represents an exogenous operating condition rather than a subset of the design variables.

Table 1: Overview of the offline PMOO benchmark suites. Here d is the design dimension, p is the task-parameter dimension, and m is the number of objectives.

Suite	#Problem instances	d	p	m	Role in the benchmark
DF (Jiang et al., 2018)	14	10	1	2/3	Controlled synthetic tasks with analytic Pareto-front structure and parameter-dependent PS/PF variation.
RE w/ SC (Tanabe & Ishibuchi, 2020; Cheng et al., 2025)	9	2–6	1–3	3/4/9	Engineering shared-component tasks testing conditional generalization under increasing parameter and objective dimensionality.
REAL (Makatura et al., 2021; Wei et al., 2025)	4	3–32	1/2	2/3	Real-world conditional design tasks with exogenous operating contexts and simulator-based objectives.

Offline archives. For each problem, we generate static archives before training any method: task parameters are sampled from Θ with structured and randomized designs, a data-collection MOEA is run at each sampled parameter, and population snapshots are stored throughout the evolutionary trajectory rather than only the final nondominated population, so the archives contain near-optimal, dominated, and intermediate-quality designs in the form of eq. (3) (Appendix C).

Train and held-out task parameters. For each problem and random seed, the archive is constructed only at training task parameters Θ_{arch}. Held-out evaluation parameters are sampled independently from the task-parameter domain and are disjoint from Θ_{arch}, so a method must transfer across conditions rather than recall. The held-out set is fixed across all methods and seeds and is used only by the benchmark evaluator, and train-parameter evaluation uses $\Theta_{\mathrm{eval}} = \Theta_{\mathrm{arch}}$ (sizes and sampling schemes in Appendix C.2).

Evaluation. For each evaluation parameter $\boldsymbol{\theta} \in \Theta_{\text{eval}}$, a method returns a finite candidate set $S_{\boldsymbol{\theta}} \subset \mathcal{X}$ without querying the ground-truth objective. The oracle is used only after inference to score the returned candidates. We report the empirical expected hypervolume $\widehat{\text{EHV}} = |\Theta_{\text{eval}}|^{-1} \sum_{\boldsymbol{\theta} \in \Theta_{\text{eval}}} \text{HV}(S_{\boldsymbol{\theta}}, \boldsymbol{r}_{\boldsymbol{\theta}})$, the average hypervolume of the returned sets over the evaluation parameters with a task-specific reference point $\boldsymbol{r}_{\boldsymbol{\theta}}$ fixed before method comparison, and refer to it as EHV in the rest of the paper.

Objective normalization. Hypervolume is not invariant to rescaling an individual objective, and the archived ranges of different objectives within one problem differ by up to three orders of magnitude in this benchmark (Appendix C.2), which would let one objective decide the comparison on its own. We therefore evaluate on *normalized* objectives: for each problem and task parameter we map $\boldsymbol{y} \mapsto (\boldsymbol{y} - \boldsymbol{z}^{\text{ideal}})/(\boldsymbol{z}^{\text{nadir}} - \boldsymbol{z}^{\text{ideal}})$ and place the reference point at $\boldsymbol{r} = 1.1 \cdot \mathbf{1}$, so every objective contributes on a common $[0, 1]$ scale and the reference point no longer depends on the archive, which keeps hypervolumes comparable across experiments that modify the archive (Appendix H.5). The bounds are used for evaluation only and are obtained per suite as described in Appendix C.2.

4 Task-Conditioned Surrogate Pareto Solvers

A natural approach to offline PMOO is to learn a differentiable surrogate $\hat{F}_\psi(\boldsymbol{x}, \boldsymbol{\theta}) \approx F(\boldsymbol{x}, \boldsymbol{\theta})$ of the task-conditioned objective landscape by supervised regression on the archive, using normalized objective values and a per-objective weighting of the squared error, and then to optimize the surrogate for each queried task parameter. Unlike fixed-task offline MOO, where the surrogate only needs to model $F(\boldsymbol{x})$, offline PMOO requires the surrogate to represent how the objective landscape deforms across Θ, so the central modeling question is how $\boldsymbol{\theta}$ should modulate the map from designs to objectives. We study three parameter-conditioning strategies that trade off simplicity, expressiveness, and statistical stability.

Concatenation conditioning (Cat) feeds the concatenated input $[\boldsymbol{x}; \boldsymbol{\theta}]$ to a single MLP. All network parameters are shared across tasks, which is statistically stable when the dependence on $\boldsymbol{\theta}$ is smooth and moderate, but because $\boldsymbol{x}$ and $\boldsymbol{\theta}$ interact only through standard hidden layers, Cat models may underrepresent sharp task-dependent changes in the landscape.

Hypernetwork conditioning (HN) maps the task parameter to the weights of a target network, $\mathbf{W}(\boldsymbol{\theta}) = H_\psi(\boldsymbol{\theta})$ and $\hat{F}^{\text{HN}}_\psi(\boldsymbol{x}, \boldsymbol{\theta}) = T_{\mathbf{W}(\boldsymbol{\theta})}(\boldsymbol{x})$ (Ha et al., 2017), so that every $\boldsymbol{\theta}$ defines its own objective predictor. This suits problems whose Pareto set moves, bends, or changes topology with the task parameter, at the cost of predicting a large number of target-network weights, which can be unstable under sparse offline archives.

Low-rank hypernetwork adaptation (HN-LoRA) keeps a shared base weight W^0_ℓ in every target layer and lets the hypernetwork predict only a rank-r residual (Hu et al., 2022),

$$W_\ell(\boldsymbol{\theta}) = W^0_\ell + \frac{\alpha}{r} A_\ell(\boldsymbol{\theta}) B_\ell(\boldsymbol{\theta}), \qquad A_\ell(\boldsymbol{\theta}) \in \mathbb{R}^{h^{\text{out}}_\ell \times r}, \quad B_\ell(\boldsymbol{\theta}) \in \mathbb{R}^{r \times h^{\text{in}}_\ell}, \quad r \ll \min(h^{\text{in}}_\ell, h^{\text{out}}_\ell), \tag{6}$$

so the task parameter induces a low-dimensional deformation of a global predictor. Compared with Cat it provides explicit task-dependent modulation, and compared with a full hypernetwork it generates far fewer parameters and is therefore more stable in the offline setting.

Objective coupling and training. The surrogate must also handle the coupling among objectives. End-to-end (E2E) models predict all objectives with one network, which encourages shared representations but exposes training to gradient conflict, whereas multiple-model (MM) surrogates train one predictor per objective, which removes the interference and provides stable gradients for scalarized guidance at the cost of m predictors. We combine both output organizations with vanilla MSE training and with three robust multi-objective training variants, GradNorm (Chen et al., 2018), PCGrad (Yu et al., 2020), and IOM (Qi et al., 2022). Implementation details of the complete surrogate family are provided in Appendix F.1.

Surrogate Pareto search. After training, a surrogate is a standalone offline PMOO solver: for each evaluation parameter $\boldsymbol{\theta}_{\text{eval}}$ we freeze $\hat{F}_\psi$ and run a multi-objective evolutionary optimizer on the predicted objectives, $S^{\text{sur}}_{\boldsymbol{\theta}_{\text{eval}}} = \text{MOEA}(\hat{F}_\psi(\cdot, \boldsymbol{\theta}_{\text{eval}}), \mathcal{X})$, without querying the true objective. These

solvers play two roles in our study. They are strong standalone baselines, and their differentiable predictions provide the scalarized guidance used by the generative refinement of Section 5.

5 GENERATIVE PARAMETRIC REFINEMENT

Surrogate Pareto search has two limitations in offline PMOO. It requires a new optimization run for every test parameter, and direct optimization of a learned surrogate can move candidates into regions poorly supported by the archive, where surrogate errors are more likely to be exploited. We therefore develop a complementary generative approach: learn a task-conditioned prior over high-quality designs, and use the surrogate only for test-time preference steering and warm-start refinement.

Elite-set training. In fixed-task offline optimization, restricting training to the best archived samples mainly trades away coverage of the design space. The parametric setting is different. Archives are collected by running a search per task parameter and retaining population snapshots, so a large part of every slice consists of early, near space-filling designs whose distribution hardly depends on $\boldsymbol{\theta}$, whereas the designs whose location moves with $\boldsymbol{\theta}$ are the converged, nondominated ones. A generator fitted to the full archive spends its capacity on this task-invariant bulk and learns a conditional distribution that is over-smoothed in $\boldsymbol{\theta}$. We therefore train the proposed generator on the part of the archive that carries the task-conditional signal, the first nondominated front of every archived slice,

$$\mathcal{D}_{\text{elite}} = \bigcup_{\boldsymbol{\theta}\in\Theta_{\text{arch}}} \{(\boldsymbol{x},\boldsymbol{\theta}) : (\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{y}) \in \mathcal{D},\ \boldsymbol{x} \in \text{ND}(\mathcal{D}_{\boldsymbol{\theta}})\}, \tag{7}$$

and model the task-conditioned distribution $p_\phi(\boldsymbol{x}^\star \mid \boldsymbol{\theta})$ of high-quality designs. Appendix H.6 measures the resulting conditioning gain directly and identifies where it stops applying. The generator is deliberately not conditioned on a preference vector: the archive records no preference labels and one design may serve several scalarizations, so preferences enter only at inference through surrogate guidance.

Task-conditional backbone and scalarized guidance. Both instantiations use a time-dependent network N_ϕ whose input is $[\boldsymbol{x}_t;\, t;\, \boldsymbol{\theta}]$ and whose output is read either as a diffusion noise prediction or as a flow-matching velocity. At inference, a preference $\boldsymbol{w} \in \Delta^{m-1}$ is injected through the augmented Tchebycheff scalarization of the surrogate,

$$J_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{\theta}) = \max_{1\le j\le m} w_j \left(\hat{f}_{\psi,j}(\boldsymbol{x},\boldsymbol{\theta}) - \hat{z}_j^\star(\boldsymbol{\theta})\right) + \rho \sum_{j=1}^{m} w_j \left(\hat{f}_{\psi,j}(\boldsymbol{x},\boldsymbol{\theta}) - \hat{z}_j^\star(\boldsymbol{\theta})\right), \tag{8}$$

where $\rho > 0$ is a small augmentation coefficient and $\hat{\boldsymbol{z}}^\star(\boldsymbol{\theta})$ is an oracle-free estimate of the task-specific ideal point computed from the surrogate and the initialization pool. Since all objectives are minimized, guidance follows the descent direction $-\nabla_{\boldsymbol{x}} J_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{\theta})$.

5.1 PARAMETRIC SCALARIZED DIFFUSION

Our first proposed method, Parametric Scalarized Diffusion (PSD), instantiates the generator as a conditional denoising diffusion model (Ho et al., 2020; Song et al., 2021). Given $(\boldsymbol{x}_0,\boldsymbol{\theta}) \sim \mathcal{D}_{\text{elite}}$, noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, and diffusion time t, the corrupted sample is $\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}$ and the network is trained to predict the injected noise, $\mathcal{L}_{\text{PSD}}(\phi) = \mathbb{E}\,\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta})\|_2^2$. During sampling, the reverse process is guided by the scalarized surrogate objective,

$$\tilde{\boldsymbol{\epsilon}}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}, \boldsymbol{w}) = \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}) + s\,\sigma_t \nabla_{\boldsymbol{x}_t} J_{\boldsymbol{w}}(\boldsymbol{x}_t, \boldsymbol{\theta}), \tag{9}$$

where s controls the guidance strength. PSD is used as a denoising refinement model: rather than sampling from pure noise, it partially corrupts a surrogate warm start and runs the guided reverse process back toward clean designs, which keeps guidance near the learned elite manifold, where surrogate gradients are more reliable.

5.2 PARAMETRIC PARETO FLOW MATCHING

Our second proposed method, Parametric Pareto Flow Matching (PPFM), is a deterministic alternative based on conditional flow matching (Lipman et al., 2023). With an elite design $\boldsymbol{x}_1 \sim p_{\text{elite}}(\cdot \mid \boldsymbol{\theta})$

Table 2: Average rank of all 16 methods on *held-out* task parameters under normalized hypervolume (Section 3), by benchmark suite (left) and by number of objectives m (right, instance counts in parentheses). Suffixes denote training variants: B = base MSE, G = GradNorm, P = PCGrad, I = IOM. Best per column in bold, second best underlined.

	By benchmark suite			*By number of objectives*			
Method	Synthetic	RE w/ SC	REAL	$m = 2$ (12)	$m = 3$ (9)	$m = 4$ (3)	$m = 9$ (3)
Oracle-free controls							
Nearest-Archive	11.90	3.14	13.91	12.73	9.94	**1.78**	**1.03**
Warm-Start-Only	7.70	3.28	8.30	7.60	6.81	<u>2.56</u>	3.46
PPSL-MOBO	9.55	12.52	11.88	7.99	11.35	16.00	15.99
Task-conditioned surrogate solvers							
Cat-E2E-B	11.13	10.64	8.72	9.12	12.57	11.76	9.58
Cat-E2E-G	8.27	8.97	**6.27**	9.16	6.97	7.38	8.96
Cat-E2E-P	7.90	9.28	<u>6.34</u>	8.87	6.82	7.48	9.77
Cat-MM-B	10.37	12.75	9.04	8.21	12.73	14.63	13.03
Cat-MM-I	9.69	10.22	6.72	10.07	8.16	9.34	10.73
HN-E2E-B	9.60	11.23	10.36	9.02	11.04	13.41	9.72
HN-E2E-G	7.27	8.42	6.54	8.91	5.87	5.35	9.33
HN-E2E-P	6.70	9.31	7.70	8.90	5.51	8.08	9.26
HN-LoRA-B	9.47	11.24	10.21	7.77	12.45	13.96	9.19
HN-LoRA-G	8.04	9.69	6.95	8.57	7.69	8.57	9.91
HN-LoRA-P	7.27	10.39	8.17	8.10	7.67	10.04	10.54
Generative refinement (ours)							
PSD	<u>5.85</u>	<u>3.02</u>	6.97	<u>5.69</u>	<u>5.43</u>	3.04	3.54
PPFM	**5.26**	**1.91**	7.91	**5.30**	**5.01**	2.62	<u>1.97</u>

and a base sample $\boldsymbol{x}_0 \sim \mathcal{N}(0, I)$, the linear path $\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$ has velocity $\boldsymbol{x}_1 - \boldsymbol{x}_0$, and a task-conditioned velocity field is trained with $\mathcal{L}_{\mathrm{PPFM}}(\phi) = \mathbb{E}\,\|\boldsymbol{v}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}) - (\boldsymbol{x}_1 - \boldsymbol{x}_0)\|_2^2$. At inference, the velocity is modified by surrogate guidance,

$$\tilde{\boldsymbol{v}}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}, \boldsymbol{w}) = \boldsymbol{v}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}) - \gamma_t \nabla_{\boldsymbol{x}_t} J_{\boldsymbol{w}}(\boldsymbol{x}_t, \boldsymbol{\theta}), \tag{10}$$

with a schedule γ_t that increases guidance near the data end of the trajectory, so early transport is dominated by the learned elite prior and later transport emphasizes the target trade-off direction.

Warm-started oracle-free reconstruction. For an unseen $\boldsymbol{\theta}_{\mathrm{eval}}$, the proposed PSD and PPFM act as warm-started refinement operators (Algorithm 2 in Appendix H.1). For each of K sampled preferences, a starting design is taken from a candidate pool derived from the archive, improved by projected gradient descent on the surrogate scalarization, and then refined: PSD partially corrupts it and runs the guided reverse diffusion, and PPFM mixes it with a base sample at an intermediate flow time and integrates the guided ODE to the data endpoint. The surrogate thus supplies task- and preference-specific directions while the generator regularizes the proposals toward the learned high-quality manifold, which is what offline PMOO needs, since unrestricted surrogate optimization exploits model error and unconditional generation alone does not cover preference-specific trade-offs.

6 Experiments

6.1 Experimental Protocol

We compare three classes of methods on the three suites of Section 3. All are oracle-free at test time and share the same normalization bounds and held-out task parameters. **(i) Task-conditioned surrogate solvers** learn $\hat{F}(\boldsymbol{x}, \boldsymbol{\theta})$ from the archive and run NSGA-II (Deb et al., 2002) on the frozen surrogate for each test parameter. This class covers Cat-E2E, Cat-MM, HN-E2E, and HN-LoRA surrogates trained with vanilla MSE, GradNorm, PCGrad, or IOM. **(ii) Warm-started generative refinement**, the proposed PSD and PPFM, uses the surrogate only for oracle-free warm starts and test-time guidance while regularizing proposals toward the elite-design distribution. **(iii) Three controls** isolate what the setting requires. *Nearest-Archive* returns the nondominated set of the nearest training parameter's archive slice, involves no learning, and measures how far plain reuse gets. *Warm-Start-Only* returns the projected-gradient warm start without the generative stage. *PPSL-MOBO* is the parametric Pareto-set-learning baseline: a map $h(\boldsymbol{\theta}, \boldsymbol{w}) \to \boldsymbol{x}$ (Lin et al., 2022; Cheng et al., 2026; 2025) trained on the same frozen surrogate, ideal point, and budget as PSD and PPFM

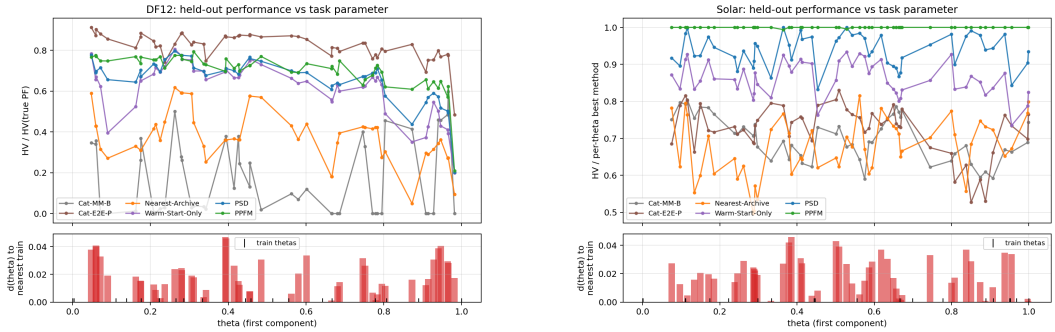


Figure 1: Held-out performance versus task parameter, with distance to the nearest training parameter beneath. **Left:** DF12, an analytic problem whose front changes topology. The vertical axis is hypervolume relative to the true front. **Right:** Solar, a real-world conditional simulator. The vertical axis is relative to the per-$\boldsymbol{\theta}$ best method. Dips in the surrogate baselines do not line up with the $d(\boldsymbol{\theta})$ bars beneath them.

(Appendix H.2), which optimizes the learned model with nothing anchoring it to the archive's support and thus probes the δ_{ref} term of eq. (5) directly. A single pre-specified configuration, selected without an oracle from objective labels already in the archive (Appendix G.8), is used unchanged on all 27 instances. Architectures and hyperparameters are in Appendix F.5 and per-problem results in Appendix J.

6.2 Main Results

Table 2 reports the held-out ranking of all 16 methods. The proposed PPFM and PSD rank first and second on both the Synthetic and the RE w/ SC suites, ahead of the best surrogate solver by a clear margin, so learning an archived elite prior and refining surrogate warm starts recovers held-out Pareto sets better than optimizing the same surrogate directly, whereas on REAL a concatenation surrogate trained with GradNorm ranks first and the generative solvers sit in the middle of the field. Among the surrogate solvers, gradient-balancing and conflict-reduction training improve on vanilla training, but no single architecture leads on every suite. The controls locate what the setting requires. Nearest-Archive, which performs no learning, ranks near the bottom on Synthetic and last on REAL, so those suites measure task-conditional generalization rather than recall, yet it is the best of all 16 methods at nine objectives (right block), where archived fronts transfer almost losslessly and dominance no longer discriminates, with most of each RE91 slice already on the first nondominated front (Appendix H.6). PPSL-MOBO amortizes the whole task-parameter family in one map trained on the same frozen surrogate, which is exactly what the setting rewards, yet it ranks near the bottom on both engineering suites, because a map that descends the learned model freely inherits the model's errors, which the elite prior prevents (Appendix H.8).

6.3 Behaviour Across the Task Domain

Figure 1 plots held-out hypervolume against the task parameter for two contrasting problems, with the distance $d(\boldsymbol{\theta})$ to the nearest training parameter beneath (remaining problems and predicted-front snapshots in Appendix G.5). On DF12 the surrogate baseline oscillates between 0.89 and 0.95 of the true-front hypervolume while the generative solvers stay near 1.00 across the whole domain, and on Solar PPFM is best at essentially every test parameter, so the average ranks reflect persistent gaps rather than a few bad parameters. The dips in the surrogate curves are not aligned with large $d(\boldsymbol{\theta})$: within the training domain, variation is driven by problem geometry and surrogate fidelity rather than by distance to the nearest observed parameter, the empirical counterpart of condition A1.

6.4 Ablations and Diagnostics

Ablation. Removing one component of the refinement procedure at a time on four representative tasks (Appendix H.4) shows that the two solvers rely on different parts of it. For PPFM, scalarized guidance is decisive, costing 1–10% on every task when removed, and the elite prior is second,

costing up to 14%. For PSD, the warm start is the component that matters, moving the result by 6–14% in a direction set by the quality of the archived starting designs, while guidance and elite selection change it by under 1% on three of four tasks. Neither solver acts as an unconditional sampler.

Archive quality and coverage (A3). Retraining every model on archives that keep only the best, a random, the worst, or a sector-deleted half of each slice (Appendix H.5) shows that which half is kept matters far less for the generative solver than for the surrogate: an elite-only archive costs Cat-MM-B 85% on DF9 and 14% on UAV32 against 12% and 1% for PSD, because a surrogate needs contrast to learn what makes a design poor. The failures that do occur are coverage failures, with a low-quality-only archive the worst variant on all four tasks and a deleted preference sector costing 10–16% on three of them, which is the mode Theorem 1 shows no zero-query method can repair.

Unwitnessed transitions (A2). On a controlled family whose Pareto set switches between two branches with tunable sharpness, with the transition region deleted from every archive (Appendix H.7), all methods hedge between the branches rather than committing to the wrong one, and the in-gap penalty grows with sharpness as Theorem 1 predicts. PPFM is markedly less sensitive than PSD, with its branch deviation flat at 0.044–0.059.

Surrogate exploitation. Two oracle-free diagnostics over six problems (Appendix H.8), the optimism gap between hypervolume under the surrogate's predictions and under the oracle, and the distance from returned designs to the nearest archived design, single out the two methods that optimize the learned model to convergence: Cat-MM-B and PPSL-MOBO are the only ones with a positive mean gap, while the archive-anchored solvers stay about 35% closer to the archive. This is why a stronger Pareto-set-learning baseline does not displace the design: offline, a map trained purely against a learned model inherits its errors.

Further diagnostics. A task-conditional signal-to-noise ratio of the training distribution rises by up to $1.87\times$ under elite selection where nondominated sorting is selective, and is inert on the nine-objective RE91-S1, where 86.5% of every slice is already nondominated (Appendix H.6), the boundary at which Nearest-Archive becomes strongest. Recomputing the ranking on the 14 analytic DF problems with IGD+, IGD, and spacing preserves the advantage of the generative solvers and shows PPSL-MOBO attaining the best spacing while ranking 9.84 on IGD+, a uniform front placed away from the true one (Appendix G.3). Refinement cuts the box-boundary saturation of the returned designs by 2.5–$6.8\times$ where surrogate gradients push against the box (Appendix G.6). PSD gives the fastest inference on all four diagnostic tasks, PPFM's guided ODE can be cut from 100 to 25 steps at no loss in quality, the generative solvers keep their lead at every candidate budget down to $K = 10$ on DF4 and RE34-S1 and on all four diagnostic tasks under 25% and 50% parameter sparsity, and held-out EHV moves by at most 3.4% across a $30\times$ range of the guidance strength (Appendices I and G.7).

7 Conclusion

We studied offline parametric multi-objective optimization, in which Pareto sets at unseen task parameters must be reconstructed from a fixed archive without further evaluations, and made the setting precise through applicability conditions, an unidentifiability theorem showing that some such conditions are unavoidable for any zero-query method, and an error decomposition that ties each solver component to one source of error. On 27 benchmark instances, a systematic study of task-conditioned surrogate solvers showed that their quality depends strongly on architecture and training strategy, and the proposed warm-started generative refinement, instantiated as PSD and PPFM, obtained the best held-out ranks on the synthetic and shared-component suites by combining an archived elite prior with surrogate-started, preference-guided refinement. The same protocol marked the limits of the approach: a surrogate solver ranks first on the conditional simulator suite, and plain archive reuse is preferable on many-objective problems, where dominance no longer separates archived designs. Archive-composition, transition-sharpness, and surrogate-exploitation studies located where the conditions bind: the generative solvers are far less sensitive to archive composition than surrogate search, while unsupported preference sectors and unwitnessed transitions remain the binding failure modes. Constrained and mixed-variable domains, calibrated guidance, and refinement strength chosen without held-out labels are natural next steps.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript editing and LaTeX formatting.

REFERENCES

Yashas Annadani, Syrine Belakaria, Stefano Ermon, Stefan Bauer, and Barbara E Engelhardt. Preference-guided diffusion for multi-objective offline optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Ian Char, Youngseog Chung, Willie Neiswanger, Kirthevasan Kandasamy, Andrew O Nelson, Mark Boyer, Egemen Kolemen, and Jeff Schneider. Offline contextual bayesian optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

Can Chen, Yingxueff Zhang, Jie Fu, Xue Steve Liu, and Mark Coates. Bidirectional learning for offline infinite-width model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 29454–29467, 2022.

Can Chen, Christopher Beckham, Zixuan Liu, Xue Liu, and Christopher Pal. Robust guided diffusion for offline black-box optimization. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International Conference on Machine Learning (ICML)*, pp. 794–803. PMLR, 2018.

Ji Cheng, Xi Lin, Bo Xue, and Qingfu Zhang. Parametric pareto set learning: Amortizing multi-objective optimization with parameters. *IEEE Transactions on Evolutionary Computation*, 2025.

Ji Cheng, Bo Xue, and Qingfu Zhang. Parametric pareto set learning for expensive multi-objective optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 36829–36837, 2026.

Sayak Ray Chowdhury and Aditya Gopalan. No-regret algorithms for multi-task bayesian optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1873–1881. PMLR, 2021.

Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

Manh Cuong Dao, Phi Le Nguyen, Thao Nguyen Truong, and Trong Nghia Hoang. Boosting offline optimizers with surrogate sensitivity. In *International Conference on Machine Learning (ICML)*, pp. 10072–10090. PMLR, 2024.

Manh Cuong Dao, The Hung Tran, Phi Le Nguyen, Thao Nguyen Truong, and Trong Nghia Hoang. ROOT: Rethinking offline optimization as distributional translation via probabilistic bridge. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Kalyanmoy Deb and Himanshu Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, 2013.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2): 182–197, 2002.

Kalyanmoy Deb, Udaya Bhaskara Rao N, and Sindhya Karthik. Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling. In *International Conference on Evolutionary Multi-criterion Optimization*, pp. 803–817. Springer, 2007.

Kalyanmoy Deb, Ling Zhu, and Sandeep Kulkarni. Handling multiple scenarios in evolutionary multiobjective numerical optimization. *IEEE Transactions on Evolutionary Computation*, 22(6): 920–933, 2018.

Justin Fu and Sergey Levine. Offline model-based optimization via normalized maximum likelihood estimation. In *International Conference on Learning Representations (ICLR)*, 2021.

Edgar Galvan, Richard J Malak, Darren J Hartl, and Jeffrey W Baur. Performance assessment of a multi-objective parametric optimization algorithm with application to a multi-physical engineering system. *Structural and Multidisciplinary Optimization*, 58(2):489–509, 2018.

David Ginsbourger, Jean Baccou, Clément Chevalier, Frédéric Perales, Nicolas Garland, and Yann Monerie. Bayesian adaptive reconstruction of profile optima and optimizers. *SIAM/ASA Journal on Uncertainty Quantification*, 2(1):490–510, 2014.

David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/forum?id=rkpACe1lx.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 6840–6851, 2020.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yaru Hu, Jinhua Zheng, Shouyong Jiang, Shengxiang Yang, and Juan Zou. Handling dynamic multiobjective optimization environments via layered prediction and subspace-based diversity maintenance. *IEEE Transactions on Cybernetics*, 53(4):2572–2585, 2021.

Desi R Ivanova, Joel Jennings, Tom Rainforth, Cheng Zhang, and Adam Foster. Co-bed: information-theoretic contextual optimization via bayesian experimental design. In *International Conference on Machine Learning*, pp. 14445–14464. PMLR, 2023.

Himanshu Jain and Kalyanmoy Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. *IEEE Transactions on Evolutionary Computation*, 18(4):602–622, 2013.

Min Jiang, Zhongqiang Huang, Liming Qiu, Wenzhen Huang, and Gary G Yen. Transfer learning-based dynamic multiobjective optimization algorithms. *IEEE Transactions on Evolutionary Computation*, 22(4):501–514, 2017.

Shouyong Jiang and Shengxiang Yang. A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 21(1): 65–82, 2016.

Shouyong Jiang, Shengxiang Yang, Xin Yao, Kay Chen Tan, Marcus Kaiser, and Natalio Krasnogor. Benchmark functions for the CEC'2018 competition on dynamic multiobjective optimization. Technical report, Newcastle University, 2018.

Minsu Kim, Jiayao Gu, Ye Yuan, Taeyoung Yun, Zixuan Liu, Yoshua Bengio, and Can Chen. Offline model-based optimization: Comprehensive review. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=QcSZWo1TLl. Survey Certification.

Andreas Krause and Cheng Ong. Contextual gaussian process bandit optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 24, 2011.

Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion models for black-box optimization. In *International Conference on Machine Learning (ICML)*, pp. 17842–17857. PMLR, 2023.

Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 5126–5137, 2020.

Seul Lee, Jaehyeong Jo, and Sung Ju Hwang. Exploring chemical space with score-based out-of-distribution generation. In *International Conference on Machine Learning (ICML)*, pp. 18872–18892. PMLR, 2023.

Zhengping Liang, Tiancheng Wu, Xiaoliang Ma, Zexuan Zhu, and Shengxiang Yang. A dynamic multiobjective evolutionary algorithm based on decision variable classification. *IEEE Transactions on Cybernetics*, 52(3):1602–1615, 2020.

Xingtao Liao, Qing Li, Xujing Yang, Weigang Zhang, and Wei Li. Multiobjective optimization for crash safety design of vehicles using stepwise regression model. *Structural and multidisciplinary optimization*, 35(6):561–569, 2008.

Xi Lin, Zhiyuan Yang, Xiaoyuan Zhang, and Qingfu Zhang. Pareto set learning for expensive multi-objective optimization. In *Advances in Neural Information Processing Systems*, volume 35, pp. 19231–19247, 2022.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Xiaoliang Ma, Yanan Yu, Xiaodong Li, Yutao Qi, and Zexuan Zhu. A survey of weight vector adjustment methods for decomposition-based multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 24(4):634–649, 2020.

Liane Makatura, Minghao Guo, Adriana Schulz, Justin Solomon, and Wojciech Matusik. Pareto gamuts: exploring optimal designs across varying contexts. *ACM Transactions on Graphics*, 40(4), 2021.

Arrchana Muruganantham, Kay Chen Tan, and Prahlad Vadakkepat. Evolutionary dynamic multiobjective optimization via kalman filter prediction. *IEEE Transactions on Cybernetics*, 46(12): 2862–2873, 2015.

Leonard Papenmeier and Petru Tighineanu. Smog: Scalable meta-learning for multi-objective bayesian optimization. *arXiv preprint arXiv:2601.22131*, 2026.

Michael Pearce and Juergen Branke. Continuous multi-task bayesian optimisation with correlation. *European Journal of Operational Research*, 270(3):1074–1085, 2018.

Michael Pearce, Janis Klaise, and Matthew Groves. Practical bayesian optimization of objectives with conditioning variables. *arXiv preprint arXiv:2002.09996*, 2020.

Bo Peng, Ye Wei, Yu Qin, Jiabao Dai, Yue Li, Aobo Liu, Yun Tian, Liuliu Han, Yufeng Zheng, and Peng Wen. Machine learning-enabled constrained multi-objective design of architected materials. *Nature Communications*, 14(1):6630, 2023.

Zhou Peng, Jinhua Zheng, Juan Zou, and Min Liu. Novel prediction and memory strategies for dynamic multiobjective optimization. *Soft Computing*, 19(9):2633–2653, 2015.

Efstratios N. Pistikopoulos, Michael C. Georgiadis, and Vivek Dua (eds.). *Multi-Parametric Programming: Theory, Algorithms and Applications*. Wiley-VCH, Weinheim, 2007.

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

Miao Rong, Dunwei Gong, Yong Zhang, Yaochu Jin, and Witold Pedrycz. Multidirectional prediction approach for dynamic multiobjective optimization problems. *IEEE Transactions on Cybernetics*, 49(9):3362–3374, 2018.

Gan Ruan, Guo Yu, Jinhua Zheng, Juan Zou, and Shengxiang Yang. The effect of diversity maintenance on prediction in dynamic multi-objective optimization. *Applied Soft Computing*, 58:631–647, 2017.

Adriana Schulz, Jie Xu, Bo Zhu, Changxi Zheng, Eitan Grinspun, and Wojciech Matusik. Interactive design space exploration and optimization for cad models. *ACM Transactions on Graphics*, 36(4): 1–14, 2017.

Jatan Shrestha, Santeri Heiskanen, Kari Hepola, Severi Rissanen, Pekka Jääskeläinen, and Joni 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.

Nilotpal Sinha, Peyman Rostami, Abd El Rahman Shabayek, Anis Kacem, and Djamila Aouada. Multi-objective hardware aware neural architecture search using hardware cost diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8032–8039, 2024.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 9460–9471, 2022.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task bayesian optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 26, 2013.

Ryoji Tanabe and Hisao Ishibuchi. An easy-to-use real-world multi-objective optimization problem suite. *Applied Soft Computing*, 89:106078, 2020.

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

Chunfeng Wang, Gary G Yen, and Min Jiang. A grey prediction-based evolutionary algorithm for dynamic multiobjective optimization. *Swarm and Evolutionary Computation*, 56:100695, 2020.

Tsun-Hsuan Wang, Pingchuan Ma, Andrew Everett Spielberg, Zhou Xian, Hao Zhang, Joshua B. Tenenbaum, Daniela Rus, and Chuang Gan. Softzoo: A soft robot co-design benchmark for locomotion in diverse environments. In *International Conference on Learning Representations (ICLR)*, 2023.

Yanjiao Wang, Tianlin Du, Tingting Liu, and Lei Zhang. Dynamic multiobjective squirrel search algorithm based on decomposition with evolutionary direction prediction and bidirectional memory populations. *IEEE Access*, 7:115997–116013, 2019.

Jonathan M Weaver-Rosen and Richard J Malak. An algorithm for multi-objective efficient parametric optimization. *Journal of Mechanical Design*, 145(3):031709, 2023.

Tingyang Wei, Jiao Liu, Abhishek Gupta, Chin Chun Ooi, Puay Siew Tan, and Yew-Soon Ong. Parametric expensive multi-objective optimization via generative solution modeling. *arXiv preprint arXiv:2511.09598*, 2025.

Zhenxing Wu, Odin Zhang, Xiaorui Wang, Li Fu, Huifeng Zhao, Jike Wang, Hongyan Du, Dejun Jiang, Yafeng Deng, Dongsheng Cao, et al. Leveraging language model for advanced multiproperty molecular optimization via prompt engineering. *Nature Machine Intelligence*, 6(11):1359–1369, 2024.

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

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5824–5836, 2020.

Ye Yuan, Can Sam Chen, Zixuan Liu, Willie Neiswanger, and Xue Steve Liu. Importance-aware co-teaching for offline model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 55718–55733, 2023.

Ye Yuan, Can Chen, Christopher Pal, and Xue Liu. Paretoflow: Guided flows in multi-objective optimization. In *International Conference on Learning Representations (ICLR)*, 2025a.

Ye Yuan, Youyuan Zhang, Can Chen, Haolun Wu, Melody Zixuan Li, Jianmo Li, James J. Clark, and Xue Liu. Design editing for offline model-based optimization. *Transactions on Machine Learning Research*, 2025b. ISSN 2835-8856.

Taeyoung Yun, Sujin Yun, Jaewoo Lee, and Jinkyoo Park. Guided trajectory generation with diffusion models for offline model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 83847–83876, 2024.

Aimin Zhou, Yaochu Jin, and Qingfu Zhang. A population prediction strategy for evolutionary dynamic multiobjective optimization. *IEEE Transactions on Cybernetics*, 44(1):40–53, 2013.

Juan Zou, Qingya Li, Shengxiang Yang, Jinhua Zheng, Zhou Peng, and Tingrui Pei. A dynamic multiobjective evolutionary algorithm based on a dynamic evolutionary environment model. *Swarm and Evolutionary Computation*, 44:247–259, 2019.

APPENDIX CONTENTS

A Extended Related Work and Scope **17**
A.1 Offline Optimization and Offline MOO . . . 17
A.2 Parametric Optimization and Contextual Bayesian Optimization . . . 18
A.3 Parametric Multi-Objective Optimization . . . 18
A.4 Positioning of Offline PMOO . . . 19
A.5 Limitations and Future Work . . . 20
A.6 Broader Impact . . . 21

B Applicability Conditions: Statements and Proofs **22**
B.1 Unidentifiability Without Queries . . . 22
B.2 Conditional Error Decomposition . . . 23

C Benchmark Protocol and DF Suite **24**
C.1 Shared Archive Construction Protocol . . . 24
C.2 Evaluation Protocol . . . 24
C.3 DF Problem Statistics . . . 25
C.4 Mixed-Quality Survival Operator . . . 25
C.5 Data-Collection Hyperparameters . . . 27
C.6 Task-Parameter Sampling Details . . . 28

D RE Benchmark Suite **28**
D.1 Shared-Component Formulation . . . 29
D.2 Problem Selection . . . 29
D.3 Shared-Component Configurations . . . 29
D.4 Dataset Statistics . . . 31

E Real-World Benchmark Suite **31**
E.1 Problem Selection . . . 31
E.2 Problem Definitions . . . 32
E.3 Dataset Statistics . . . 33

F Method and Implementation Details **34**
F.1 Surrogate Architecture Details . . . 34
F.2 Multi-Objective Surrogate Training . . . 34
F.3 Surrogate Search and Warm-Start Construction . . . 34
F.4 Generative Model Details . . . 36
F.5 Training and Inference Hyperparameters . . . 37

G Additional Result Tables **39**
G.1 Train-Parameter Ranking . . . 39
G.2 Breakdown by Objective Count . . . 39
G.3 Distance-Based and Diversity Metrics . . . 40
G.4 Matched-Budget Many-Objective Search . . . 40
G.5 Performance Across the Task Domain: Remaining Problems . . . 41
G.6 Boundary Saturation Under Guided Refinement . . . 41
G.7 Inference-Time Hyperparameter Sensitivity . . . 42
G.8 Configuring the Method Without an Oracle . . . 43

H Detailed Experimental Analyses **44**
H.1 Warm-Start Construction . . . 44
H.2 PPSL-MOBO Baseline . . . 44
H.3 Archive-Only Diagnostic . . . 45
H.4 Ablation of the Refinement Procedure . . . 45
H.5 When Does Archive Quality and Coverage Matter? . . . 45
H.6 Why Elite Selection Helps Here but Not in Fixed-Task Offline MOO . . . 46
H.7 Unwitnessed Transitions: Where A2 Binds . . . 47
H.8 Diagnosing Surrogate Exploitation . . . 48

I Additional Efficiency, Robustness, and Qualitative Analyses **48**
I.1 Test-Time Efficiency . . . 48
I.2 Generation-Budget Robustness . . . 49
I.3 Parameter-Sparsity Robustness . . . 50

J Detailed Per-Suite Results **52**

K Licenses for used assets **55**

A Extended Related Work and Scope

A.1 Offline Optimization and Offline MOO

In applications such as protein engineering (Wu et al., 2024), materials design (Peng et al., 2023), and hardware design (Sinha et al., 2024), evaluating new candidates often requires costly wet-lab experiments or simulations. Offline optimization studies how to use the resulting archives to propose further designs.

Offline Model-Based Optimization (MBO) addresses the challenge of optimizing a black-box objective function $f : \mathcal{X} \to \mathcal{Y}$ using only a fixed, static dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, where $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^d$ represents a design and $y_i = f(\boldsymbol{x}_i)$ represents its evaluated property or properties (Kim et al., 2026). Unlike online optimization or active learning, where the algorithm can iteratively query the ground-truth oracle to reduce uncertainty, offline MBO must propose candidate solutions solely based on the information contained within $\mathcal{D}$.

The fundamental difficulty in this setting stems from the need to estimate the objective landscape beyond the support of the training data. As noted by Trabucco et al. (2021), extrapolating into these unseen regions introduces significant epistemic uncertainty. Naive optimization of a learned surrogate model often leads to "objective hacking" (or reward hacking), where the optimizer exploits inaccuracies in the model to identify designs that appear high-performing but fail under the true oracle (Skalse et al., 2022). Below we separate prior work into single-objective and multi-objective offline optimization, since the latter must additionally recover a diverse set of trade-off solutions rather than a single optimum.

Single-Objective Optimization. In the SOO setting ($y \in \mathbb{R}$), the goal is to find a design $\boldsymbol{x}^\star = \arg\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x})$. Recent approaches typically fall into two complementary categories: surrogate modeling and generative modeling.

Surrogate-based methods focus on learning a robust function approximator $f_\phi(\boldsymbol{x})$ that is resistant to OOD overestimation. A prominent strategy involves conservative estimation. For instance, COMs (Trabucco et al., 2021) regularizes the surrogate by penalizing predictions on adversarial samples generated via gradient ascent. Similarly, NEMO (Fu & Levine, 2021) employs a normalized maximum-likelihood formulation to quantify uncertainty, while ROMA (Yu et al., 2021) enforces smoothness priors on the manifold to stabilize gradient-based optimization.

Generative approaches view optimization as a conditional generation problem, aiming to model $p(x|y)$. MINs (Kumar & Levine, 2020) learns an inverse mapping using GANs to generate designs conditioned on high target values. More recent advances leverage deep generative models to better capture the data distribution. ICT (Yuan et al., 2023) and BDI (Chen et al., 2022) utilize generative models with specialized training strategies, such as triplet losses and bidirectional learning, to ensure generated candidates remain close to the data manifold.

Furthermore, diffusion models have been extensively adapted for this task, primarily through conditional generation mechanisms. A prominent direction involves classifier-free guidance, where models are conditioned on high-performance values (Krishnamoorthy et al., 2023), a technique further refined by utilizing surrogates to adjust guidance parameters (Chen et al., 2024) or by incorporating full trajectories (Yun et al., 2024). Beyond standard conditioning, recent works have employed diffusion to map low-quality samples to high-performing designs (Dao et al., 2024; 2025) or to steer generation toward specific biochemical properties in molecular discovery (Lee et al., 2023). In contrast to these approaches, Yuan et al. (2025b) implement a variant of classifier guidance by subjecting gradient-optimized designs to a noise-injection and denoising process to maintain distributional validity. Generally, however, such classifier-guided methods are less favored than classifier-free approaches due to the additional training burden of auxiliary models and the potential for reward hacking.

Multi-Objective Optimization. In the MOO setting, the output is a vector $\boldsymbol{y} \in \mathbb{R}^m$, and the objective is to approximate the Pareto set, the set of designs that are not dominated by any other design. In offline MOO, the archive contains vector-valued evaluations, and the optimizer must approximate the Pareto set using only this fixed archive.

While SOO methods are well-explored, offline MOO remains less developed. Traditional approaches often reduce the problem to SOO via scalarization (e.g., weighted sum) (Ma et al., 2020), but this fails to capture the complexity of non-convex Pareto fronts. Recent works have begun to address this by integrating generative modeling with multi-objective guidance. For instance, ParetoFlow (Yuan et al., 2025a) employs flow matching guided by multiple surrogates to steer generation toward the Pareto front. Expanding on this generative direction, recent work on Preference-Guided Diffusion (Annadani et al., 2025) proposes a classifier-based guidance mechanism where a preference model is trained to predict the probability of one design dominating another. Crucially, this approach introduces a diversity-aware preference guidance that augments standard Pareto dominance with diversity criteria. The additional diversity criterion encourages generated solutions to cover a broader region of the objective space, although generalization outside the archive support remains dependent on the learned model and data coverage. Shrestha et al. (2026) propose Pareto-Conditioned Diffusion (PCD), an end-to-end framework that formulates offline multi-objective optimization as a conditional sampling problem, utilizing dominance-aware reweighting and reference-direction mechanisms to achieve high-quality and diverse Pareto solutions without relying on explicit surrogate models.

A.2 Parametric Optimization and Contextual Bayesian Optimization

Bayesian optimization (BO) provides a classical framework for expensive black-box optimization, where a probabilistic surrogate is updated sequentially and an acquisition function selects the next evaluation. A closely related setting to PMOO is contextual or parametric BO, where the response depends jointly on a controllable design variable and an exogenous task, context, or environmental parameter. Contextual Gaussian-process bandits model the reward function over the joint context–action space and select actions after observing the context (Krause & Ong, 2011). Multi-task BO and continuous multi-task BO instead exploit correlations across a collection or continuum of related tasks, so that evaluations at one task can improve optimization at other tasks (Swersky et al., 2013; Pearce & Branke, 2018). Related profile-optimization methods aim to learn the optimizer or optimum-value function as a function of nuisance or task variables, for example through profile expected improvement (Ginsbourger et al., 2014). More recent conditional BO frameworks further formalize optimization under conditioning variables and design acquisition rules that share information across conditional objectives (Char et al., 2019; Pearce et al., 2020; Ivanova et al., 2023).

These works are highly relevant because they already recognize that the optimum should be viewed as a function of a task or context parameter, rather than as a single fixed design. However, they differ from Offline PMOO in three important ways. First, contextual and multi-task BO methods generally assume sequential access to new function evaluations, whereas Offline PMOO receives a fixed archive and cannot query the oracle after training. Second, most contextual BO methods are single-objective and aim to identify one optimizer per context, while Offline PMOO must reconstruct a set of Pareto-optimal trade-off designs for each test parameter. Third, BO acquisition functions decide where to sample next, while Offline PMOO must learn an oracle-free generator or solver that directly returns candidate Pareto sets for unseen task parameters. Thus, contextual BO provides an important online and usually single-objective precursor, but it does not directly address fixed-archive, task-conditioned Pareto-set reconstruction.

A.3 Parametric Multi-Objective Optimization

PMOO considers a family of optimization problems parameterized by an exogenous task parameter $\boldsymbol{\theta} \in \Theta$ (Eq. 1). Existing work can be broadly categorized into four complementary research threads.

Parametric MOO in Time-Varying Environments. When the parameter $\boldsymbol{\theta}$ varies dynamically over time, the problem transforms into Dynamic Multi-Objective Optimization (DMOO), where the core challenge is to rapidly track the moving Pareto optimal set. Existing research primarily encompasses three strategies. *Prediction-based strategies* exploit historical information to mine the temporal patterns of $\boldsymbol{\theta}$ to generate high-quality initial populations. While early population prediction (Zhou et al., 2013) and Kalman filtering methods (Muruganantham et al., 2015) mainly addressed linear changes, subsequent works introduced grey prediction models (Wang et al., 2020) for abrupt changes, multi-directional prediction (Rong et al., 2018) and transfer learning (Jiang et al., 2017) for set deformation and cross-environment knowledge reuse, and decision variable classification (Liang et al., 2020) for precise allocation of predictive resources. *Diversity-based strategies* focus on

reactivating the population's exploration capability after parameter shifts, with representative works including dynamic NSGA-II (Deb et al., 2007), hybrid evolutionary strategies (Jiang & Yang, 2016), and subspace partition compensation mechanisms (Ruan et al., 2017; Hu et al., 2021). *Memory-based strategies* are tailored for periodically changing environments by constructing memory pools to store and reuse historical elite solutions (Peng et al., 2015; Zou et al., 2019), further enhanced by bidirectional memory strategies that utilize both forward and backward evolutionary information to significantly improve convergence efficiency (Wang et al., 2019).

Multi-Task Collaborative Optimization in Discrete Spaces. When computational resources are limited and prohibit traversing the entire parameter space, Θ is typically discretized via sampling. The core challenge then becomes handling topological changes between discrete samples and mining correlations across different parameter configurations (i.e., tasks). To address topological alterations, adaptive grids and interpolation models have been utilized for effective design space exploration (Schulz et al., 2017). To enhance optimization efficiency, Multi-Task Bayesian Optimization (MTBO) has emerged as a mainstream approach. Methods in this vein leverage multi-task kernels and improved acquisition functions to exploit latent inter-task correlations, enabling collaborative optimization with limited samples (Chowdhury & Gopalan, 2021). Furthermore, modular meta-learning frameworks like SMOG (Papenmeier & Tighineanu, 2026) have been proposed to achieve linear computational scalability with respect to the number of tasks.

Pareto Set Manifold Learning in Continuous Domains. In contrast to discrete sampling, research in continuous parameter domains aims to construct a continuous mapping from $\boldsymbol{\theta}$ to the Pareto set, realizing a "solve once, respond everywhere" paradigm. Predictive dominance mechanisms have been employed to enable direct evolutionary search in continuous domains, overcoming the precision limitations of discretization (Galvan et al., 2018). When analytical gradients are available, the concept of *Pareto Gamuts* (Makatura et al., 2021) utilizes homotopy continuation based on generalized KKT conditions to trace the optimal solution manifold. However, such numerical continuation methods struggle with non-convex or disconnected manifolds. To fit complex nonlinear mappings, deep learning-based parametric Pareto set learning frameworks (Cheng et al., 2025) combine hypernetworks and Low-Rank Adaptation (LoRA) to construct a unified mapping from the joint parameter-preference space to the decision space, enabling real-time inference of the Pareto front under arbitrary parameter configurations.

Expensive Parametric MOO. The aforementioned methods generally rely on a substantial number of objective function evaluations. However, in scenarios with strictly constrained computational resources, such as aerospace design, the focus shifts to utilizing small samples to build high-fidelity models for decision support. Surrogate-based strategies emphasize designing efficient acquisition functions to guide sampling; for instance, the MO-EPO algorithm (Weaver-Rosen & Malak, 2023) proposes a parameterized expected hypervolume improvement criterion to balance exploration and exploitation via Gaussian processes. Alternatively, generative learning-based strategies directly mine the mapping between parameters and solution sets to exploit inter-task synergies. Wei et al. (2025) construct an inverse model by alternating between task-aware Gaussian process-guided search and conditional generative models (such as VAEs or Diffusion Models). Similarly, Cheng et al. (2026) introduce hypernetworks and LoRA to establish a joint parameter-preference mapping, providing a new paradigm for real-time solution generation in high-dimensional continuous parameter spaces.

A.4 Positioning of Offline PMOO

Offline PMOO differs from both fixed-task offline MOO and existing forms of PMOO. The distinction is not only whether the objectives are parameterized, but also whether the optimizer is allowed to acquire new evaluations after observing the archive. Section 2 contrasts Offline PMOO with its neighbouring settings; here we expand on the two comparisons that are most often conflated with it.

Versus dynamic MOO. Dynamic MOO tracks a Pareto set that drifts with time, detecting changes and reusing the incumbent population across successive environments. Three differences make it a distinct problem rather than a special case. The conditioning variable $\boldsymbol{\theta}$ is an arbitrary, possibly multi-dimensional operating condition, not a scalar time index, so there is no notion of a "previous" environment from which to warm-start. Queries arrive in arbitrary order and are answered independently, so no population is carried over. Most importantly, a dynamic-MOO algorithm evaluates the objective *after* the environment changes, which is precisely the capability the zero-query protocol

removes; such algorithms therefore cannot be run in our setting at all, rather than merely performing poorly in it.

Versus per-parameter fixed-task offline MOO. Offline PMOO is not equivalent to running fixed-task offline MOO independently for each observed parameter value. In many cases, the test parameter $\boldsymbol{\theta}_{\text{eval}}$ may have no exact counterpart in the archive, so the method must transfer information across nearby or related task parameters. It is also not merely a low-budget variant of online PMOO: after the archive is fixed, the method cannot use acquisition functions, uncertainty estimates, or evolutionary search to request additional ground-truth evaluations. This zero-query constraint makes archive coverage, task-parameter generalization, and robustness to surrogate error central to the problem.

A.5 Limitations and Future Work

This work focuses on continuous offline parametric multi-objective optimization with bounded design and task-parameter spaces. Although the benchmark includes synthetic, engineering, and real-world design tasks, it does not fully cover discrete or mixed-integer variables, categorical design choices, hard safety constraints, or highly irregular feasible regions. Box constraints are handled by projection; general nonlinear constraints are outside the present scope. Extending offline PMOO to such settings is an important direction, especially for manufacturing, robotics, and scientific design problems where feasibility constraints may dominate objective trade-offs.

Failure modes tied to the applicability conditions. Section 2.1 states four conditions, and each has a corresponding failure mode that our experiments locate rather than merely acknowledge. **F1, task-space gap** (A1): the queried parameter lies far from any archived parameter, or outside the archived range entirely. Our evaluation measures interpolation within the observed domain, the regime that condition A1 covers. **F2, Pareto-mode omission** (A3): a branch of the conditional Pareto set is absent from every neighbouring archive slice. Theorem 2 shows this case is unidentifiable in principle, and Appendix H.5 quantifies the cost: deleting an f_1-specialized preference sector costs 10% held-out EHV on Solar and stays within seed noise on the other three diagnostic tasks, while an archive of only low-quality designs costs up to 23%. Notably, restricting the archive to *elite* designs is not a failure mode for the generative solvers, though it costs a surrogate baseline up to 25%; what matters is coverage of the trade-off surface, not the Pareto rank of the retained designs. **F3, local surrogate failure** (A4): surrogate gradients are unreliable along the refinement path, so the warm start is misled. Appendix H.8 shows this is the mode that separates support-anchored from unanchored solvers. A fourth regime combines a sharp transition in the conditional Pareto set with no archive coverage of the transition itself (Appendix H.7); there all methods hedge between branches rather than recovering the correct one.

Six indicators computable with *zero* oracle queries flag these regimes at deployment. The nearest-task distance $h(\boldsymbol{\theta})$, the local archive density, and the local elite count signal F1 and F2, and the candidate support distance, surrogate ensemble disagreement, and denoising residual signal F3. Each indicator targets one documented mode, and calibrating detection thresholds for them is a natural next step.

Where the setting is easier than a learned solver assumes. Our controls delimit the algorithmic claim as well as the setting. Nearest-archive reuse is competitive on the RE shared-component suite (3.14, against 3.02 for PSD and 1.91 for PPFM), and on many-objective instances it is outright preferable: it ranks 1.78 at four objectives and 1.03 at nine, first of all sixteen methods (Section 6, Appendix G.2). Not every Offline PMOO instance requires a learned conditional solver, and practitioners should run the zero-learning control before adopting a more complex method. The pattern correlates with objective count, but m is confounded with problem family in our benchmark, so identifying the true driver requires many-objective instances outside the RE family.

Metrics. The benchmark emphasizes hypervolume-based evaluation. While HV captures both convergence and diversity, it depends on the choice of reference point and may hide local coverage failures on disconnected or highly non-uniform Pareto fronts. Appendix G.3 therefore reports IGD+, IGD, and spacing on the analytic DF suite, where a true front is available; the distance metrics agree with HV for our solvers, and the combination exposes a uniform-but-misplaced front that neither metric alone would reveal. Constructing trustworthy reference sets for the RE and REAL suites, where any such set is itself oracle-derived, remains open.

Promising directions include calibrated surrogate guidance, robust elite-set construction under noisy archives, constrained and discrete offline PMOO, adaptive per-task refinement strength selected without held-out labels, many-objective guidance, active data acquisition under limited query budgets, and stronger protocols for sparse-data and extrapolation regimes.

A.6 Broader Impact

Offline parametric multi-objective optimization can reduce the need for repeated expensive evaluations by reusing historical design archives across varying operating conditions. This may benefit scientific and engineering domains such as materials design, energy systems, robotics, transportation, and industrial product design, where physical experiments or high-fidelity simulations are costly. By explicitly modeling trade-offs under changing task parameters, offline PMOO may also support more transparent decision making than single-objective optimization, since users can inspect alternative Pareto-optimal designs rather than receiving a single prescribed solution.

At the same time, the deployment of offline optimization methods carries risks. Models trained on historical archives may reproduce biases in data collection, fail in poorly covered regions of the parameter space, or exploit inaccuracies in learned surrogates. In safety-critical applications, generated designs should therefore not be deployed without independent validation using domain-specific simulators, experiments, or expert review. The benchmark and algorithms introduced here are intended as research tools rather than autonomous decision-making systems. We recommend that future applications document objective definitions, constraints, parameter ranges, data coverage, and validation procedures, and that code and datasets be released when possible to support reproducibility and independent auditing.

B Applicability Conditions: Statements and Proofs

This appendix supports Section 2.1. We first prove that no zero-query method admits an unconditional guarantee (Appendix B.1), then state and prove the conditional error decomposition that organizes our solver design (Appendix B.2). Throughout, $\mathcal{X} \subseteq \mathbb{R}^d$ is compact, $\Theta \subseteq \mathbb{R}^p$ is compact, and $G_{\boldsymbol{w}}$ denotes the augmented Tchebycheff scalarization of the *true* objectives,

$$G_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{\theta}) = \max_{j \in [m]} w_j \big(f_j(\boldsymbol{x}, \boldsymbol{\theta}) - z_j^\star(\boldsymbol{\theta})\big) + \rho \sum_{j=1}^{m} w_j \big(f_j(\boldsymbol{x}, \boldsymbol{\theta}) - z_j^\star(\boldsymbol{\theta})\big), \tag{11}$$

with $\boldsymbol{z}^\star(\boldsymbol{\theta})$ the true ideal point and $\rho > 0$ the augmentation coefficient. We write $\widehat{G}_{\boldsymbol{w}}$ for the same expression formed from the learned surrogate $\hat{F}_\psi$ and the estimated ideal point $\hat{\boldsymbol{z}}(\boldsymbol{\theta})$. Keeping these two symbols distinct is essential: a bound on the surrogate objective alone says nothing about true Pareto quality.

B.1 Unidentifiability Without Queries

Theorem 2 (Restatement of Theorem 1)**.** *Let $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{\theta}_i, \boldsymbol{y}_i)\}_{i=1}^N$ be a finite archive and let $\Theta_{\mathrm{arch}} = \{\boldsymbol{\theta}_i\}_{i=1}^N$ be its set of task parameters. Fix any $\boldsymbol{\theta}^\star \in \Theta \setminus \Theta_{\mathrm{arch}}$ and any $R > 0$. Then there exist two objective families $F^{(1)}, F^{(2)} : \mathcal{X} \times \Theta \to \mathbb{R}^m$, both continuous (indeed C^∞ in $\boldsymbol{\theta}$), such that*

(i) $F^{(1)}(\boldsymbol{x}, \boldsymbol{\theta}) = F^{(2)}(\boldsymbol{x}, \boldsymbol{\theta})$ for every $\boldsymbol{x} \in \mathcal{X}$ and every $\boldsymbol{\theta} \in \Theta_{\mathrm{arch}}$; in particular the two families are consistent with every tuple in $\mathcal{D}$, and

(ii) the conditional Pareto sets satisfy $d_{\mathrm{H}}\Big(\mathcal{X}^\star_{(1)}(\boldsymbol{\theta}^\star),\, \mathcal{X}^\star_{(2)}(\boldsymbol{\theta}^\star)\Big) \geq R'$, where d_{H} is the Hausdorff distance and R' is the diameter of $\mathcal{X}$ whenever R exceeds a threshold depending only on $F^{(1)}$.

Proof. Since Θ_{arch} is finite and $\boldsymbol{\theta}^\star \notin \Theta_{\mathrm{arch}}$, the quantity $r = \frac{1}{2} \min_i \|\boldsymbol{\theta}^\star - \boldsymbol{\theta}_i\|$ is strictly positive. Let B_r denote the open ball of radius r around $\boldsymbol{\theta}^\star$; by construction $B_r \cap \Theta_{\mathrm{arch}} = \emptyset$.

Let $\phi : \Theta \to [0, 1]$ be the standard smooth bump function supported on B_r,

$$\phi(\boldsymbol{\theta}) = \begin{cases} \exp\!\left(1 - \dfrac{r^2}{r^2 - \|\boldsymbol{\theta} - \boldsymbol{\theta}^\star\|^2}\right), & \|\boldsymbol{\theta} - \boldsymbol{\theta}^\star\| < r, \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

which is C^∞, satisfies $\phi(\boldsymbol{\theta}^\star) = 1$, and vanishes identically outside B_r.

Take $F^{(1)}$ to be any continuous objective family consistent with $\mathcal{D}$, and let $\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathcal{X}$ be two designs realizing the diameter of $\mathcal{X}$. Choose a continuous $g : \mathcal{X} \to \mathbb{R}^m$ that is minimized on a neighbourhood of $\boldsymbol{x}_b$ and strictly increasing in every coordinate away from it, and set

$$F^{(2)}(\boldsymbol{x}, \boldsymbol{\theta}) = F^{(1)}(\boldsymbol{x}, \boldsymbol{\theta}) + R\, \phi(\boldsymbol{\theta})\, g(\boldsymbol{x}). \tag{13}$$

Property (i) is immediate: for $\boldsymbol{\theta} \in \Theta_{\mathrm{arch}}$ we have $\phi(\boldsymbol{\theta}) = 0$, so $F^{(2)}(\cdot, \boldsymbol{\theta}) \equiv F^{(1)}(\cdot, \boldsymbol{\theta})$ on all of $\mathcal{X}$, and the two families are therefore indistinguishable from *any* amount of data collected at archived parameters, not merely from the N tuples actually in $\mathcal{D}$.

For property (ii), at $\boldsymbol{\theta}^\star$ we have $\phi(\boldsymbol{\theta}^\star) = 1$, so $F^{(2)}(\boldsymbol{x}, \boldsymbol{\theta}^\star) = F^{(1)}(\boldsymbol{x}, \boldsymbol{\theta}^\star) + R\, g(\boldsymbol{x})$. As $R \to \infty$ the term $R\, g$ dominates, and every design outside the minimizing neighbourhood of g becomes dominated by $\boldsymbol{x}_b$; hence $\mathcal{X}^\star_{(2)}(\boldsymbol{\theta}^\star)$ concentrates near $\boldsymbol{x}_b$. Choosing $F^{(1)}$ so that $\mathcal{X}^\star_{(1)}(\boldsymbol{\theta}^\star)$ lies near $\boldsymbol{x}_a$ gives a Hausdorff separation of order $\|\boldsymbol{x}_a - \boldsymbol{x}_b\|$, i.e. the diameter of $\mathcal{X}$. □

Three consequences are worth recording. First, the obstruction is informational rather than statistical: it does not vanish with more samples, more seeds, or a better model class, because the two families are identical everywhere the archive can look. Second, it applies to every zero-query method, so the appropriate question is not whether a method makes assumptions but whether it states them and whether they hold in the deployment regime of interest. Third, the construction is exactly the unwitnessed-discontinuity case that condition A2 excludes, which is why we test that case empirically with the transition-sharpness family of Appendix H.7: the observed outcome there, models hedging *between* the two branches rather than committing to the unwitnessed one, is the finite-sample signature of this proposition.

B.2 Conditional Error Decomposition

We now bound the true scalarized gap of a returned design by a sum of interpretable terms under A1–A4, one per source of error.

Assumption 1 (Formal versions of A1–A4)**.** *Fix a preference $\boldsymbol{w}$ and a queried parameter $\boldsymbol{\theta}$ with nearest archived parameter $\boldsymbol{\theta}_{i(\boldsymbol{\theta})}$ at distance $h(\boldsymbol{\theta})$.*

(A1) *$\boldsymbol{\theta}$ lies in the convex hull of Θ_{arch} and $h(\boldsymbol{\theta})$ is finite.*

(A2) *$|G_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{\theta}) - G_{\boldsymbol{w}}(\boldsymbol{x},\boldsymbol{\theta}')| \le L_{\boldsymbol{\theta}}\|\boldsymbol{\theta}-\boldsymbol{\theta}'\|$ for all $\boldsymbol{x}\in\mathcal{X}$ and all $\boldsymbol{\theta},\boldsymbol{\theta}'$ within the supported neighbourhood.*

(A3) *The archive slice at $\boldsymbol{\theta}_{i(\boldsymbol{\theta})}$ contains a design $\tilde{\boldsymbol{x}}$ with $G_{\boldsymbol{w}}(\tilde{\boldsymbol{x}},\boldsymbol{\theta}_{i(\boldsymbol{\theta})}) - G^{\star}_{\boldsymbol{w}}(\boldsymbol{\theta}_{i(\boldsymbol{\theta})}) \le \varepsilon_{\mathrm{arch}}$.*

(A4) *$\sup|\widehat{G}_{\boldsymbol{w}} - G_{\boldsymbol{w}}| \le \varepsilon_{\mathrm{sur}}$ over the tube traversed by the warm start and refinement, and the estimated ideal point satisfies $\|\hat{\boldsymbol{z}}(\boldsymbol{\theta}) - \boldsymbol{z}^{\star}(\boldsymbol{\theta})\|_{\infty} \le \varepsilon_{\mathrm{ideal}}$.*

Additionally, $G_{\boldsymbol{w}}(\cdot,\boldsymbol{\theta})$ is $L_{\boldsymbol{x}}$-Lipschitz in $\boldsymbol{x}$, the surrogate optimization leaves residual $\varepsilon_{\mathrm{opt}}$ in $\widehat{G}_{\boldsymbol{w}}$, and the generative refinement displaces the candidate by at most δ_{ref} in design space.

Proposition 1 (Conditional error decomposition)**.** *Under Assumption 1, the design $\hat{\boldsymbol{x}}$ returned for preference $\boldsymbol{w}$ at parameter $\boldsymbol{\theta}$ satisfies*

$$G_{\boldsymbol{w}}(\hat{\boldsymbol{x}},\boldsymbol{\theta}) - G^{\star}_{\boldsymbol{w}}(\boldsymbol{\theta}) \;\le\; \varepsilon_{\mathrm{arch}} + 2L_{\boldsymbol{\theta}}h(\boldsymbol{\theta}) + 2\varepsilon_{\mathrm{sur}} + 2(1+\rho m)\,\varepsilon_{\mathrm{ideal}} + \varepsilon_{\mathrm{opt}} + L_{\boldsymbol{x}}\,\delta_{\mathrm{ref}}. \tag{14}$$

Proof. Write $\boldsymbol{\theta}_i = \boldsymbol{\theta}_{i(\boldsymbol{\theta})}$ and let $\tilde{\boldsymbol{x}}$ be the archive seed of (A3). We track the candidate through the three stages of the solver.

Starting design. By (A3) and (A2),

$$\begin{aligned} G_{\boldsymbol{w}}(\tilde{\boldsymbol{x}},\boldsymbol{\theta}) &\le G_{\boldsymbol{w}}(\tilde{\boldsymbol{x}},\boldsymbol{\theta}_i) + L_{\boldsymbol{\theta}}h(\boldsymbol{\theta}) \le G^{\star}_{\boldsymbol{w}}(\boldsymbol{\theta}_i) + \varepsilon_{\mathrm{arch}} + L_{\boldsymbol{\theta}}h(\boldsymbol{\theta}) \\ &\le G^{\star}_{\boldsymbol{w}}(\boldsymbol{\theta}) + \varepsilon_{\mathrm{arch}} + 2L_{\boldsymbol{\theta}}h(\boldsymbol{\theta}), \end{aligned} \tag{15}$$

where the last step applies (A2) once more to transfer the optimal value between $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}$.

Warm start. The projected-gradient stage minimizes $\widehat{G}_{\boldsymbol{w}}$ and terminates with residual $\varepsilon_{\mathrm{opt}}$, so its output $\boldsymbol{x}^{\mathrm{ws}}$ obeys $\widehat{G}_{\boldsymbol{w}}(\boldsymbol{x}^{\mathrm{ws}},\boldsymbol{\theta}) \le \widehat{G}_{\boldsymbol{w}}(\tilde{\boldsymbol{x}},\boldsymbol{\theta}) + \varepsilon_{\mathrm{opt}}$ (the stage never increases the surrogate objective beyond its residual). Converting to the true scalarization costs $\varepsilon_{\mathrm{sur}}$ on each side by (A4):

$$G_{\boldsymbol{w}}(\boldsymbol{x}^{\mathrm{ws}},\boldsymbol{\theta}) \le G_{\boldsymbol{w}}(\tilde{\boldsymbol{x}},\boldsymbol{\theta}) + 2\varepsilon_{\mathrm{sur}} + \varepsilon_{\mathrm{opt}}. \tag{16}$$

Refinement. The generative stage moves the candidate by at most δ_{ref} in design space, so by $L_{\boldsymbol{x}}$-Lipschitzness of $G_{\boldsymbol{w}}(\cdot,\boldsymbol{\theta})$,

$$G_{\boldsymbol{w}}(\hat{\boldsymbol{x}},\boldsymbol{\theta}) \le G_{\boldsymbol{w}}(\boldsymbol{x}^{\mathrm{ws}},\boldsymbol{\theta}) + L_{\boldsymbol{x}}\delta_{\mathrm{ref}}. \tag{17}$$

Ideal point. Both $G_{\boldsymbol{w}}$ and $\widehat{G}_{\boldsymbol{w}}$ are built on a reference point. A perturbation of $\boldsymbol{z}$ by $\varepsilon_{\mathrm{ideal}}$ in ℓ_∞ changes the max-term of eq. (11) by at most $\varepsilon_{\mathrm{ideal}}$ (since $\sum_j w_j = 1$) and the augmentation term by at most $\rho m\,\varepsilon_{\mathrm{ideal}}$; the factor 2 accounts for the estimate entering both the evaluated design and the optimum.

Chaining eqs. (15) to (17) and adding the ideal-point term gives eq. (14). □

Reading the decomposition. Each solver component of Section 5 is associated with exactly one term, which is how we use eq. (14) in the design discussion: the nearest-archive seed controls $\varepsilon_{\mathrm{arch}}$; in-domain interpolation controls $L_{\boldsymbol{\theta}}h(\boldsymbol{\theta})$; partial rather than full corruption keeps the trajectory inside the region where (A4) holds, which controls $\varepsilon_{\mathrm{sur}}$; the projected-gradient warm start supplies the preference direction and drives $\varepsilon_{\mathrm{opt}}$; and the elite prior bounds δ_{ref} by pulling proposals back toward supported designs.

Two remarks complete the result. First, eq. (14) bounds a *scalarized* gap for a fixed preference, and turning a family of such bounds into a Pareto-front distance such as IGD requires a coverage condition on the preference set. Second, the experiments of Section 6 probe the failure modes behind the constants $L_{\boldsymbol{\theta}}$, $\varepsilon_{\mathrm{sur}}$, and δ_{ref} rather than estimating the constants themselves: archive coverage via the sweep of Appendix H.5, task regularity via the transition family of Appendix H.7, and surrogate reliability via the optimism-gap and support-distance diagnostics of Appendix H.8.

C BENCHMARK PROTOCOL AND DF SUITE

This appendix specifies the shared benchmark protocol used throughout the paper and then provides the detailed setup of the DF benchmark suite. The shared protocol includes archive construction, task-parameter sampling, reference-point construction, EHV evaluation, and rank-based aggregation. The RE and REAL suites follow the same protocol unless explicitly stated otherwise in Appendices D and E.

C.1 SHARED ARCHIVE CONSTRUCTION PROTOCOL

For each benchmark problem, the offline archive is generated before training or evaluating any optimization method. Let $\Theta_{\mathrm{arch}} \subset \Theta$ denote the set of task parameters used for archive construction. For each $\boldsymbol{\theta} \in \Theta_{\mathrm{arch}}$, we run a data-collection MOEA and retain population snapshots at a fixed collection interval. The resulting archive for one random seed is

$$\mathcal{D}^{(s)} = \bigcup_{\boldsymbol{\theta}\in\Theta_{\mathrm{arch}}} \bigcup_{g\in\mathcal{G}} \left\{ \big(\boldsymbol{x}, \boldsymbol{\theta}, F(\boldsymbol{x},\boldsymbol{\theta})\big) : \boldsymbol{x} \in P_g^{(s,\boldsymbol{\theta})} \right\}, \tag{18}$$

where $P_g^{(s,\boldsymbol{\theta})}$ is the population at generation g for seed s and task parameter $\boldsymbol{\theta}$, and $\mathcal{G}$ is the set of generations at which snapshots are retained. All objective values in the archive are computed before method training. During optimization and inference, methods are not allowed to query the true objective function.

The archive is intentionally not restricted to nondominated solutions. Retaining dominated and intermediate-quality designs creates a more realistic offline optimization setting and allows us to test whether methods can learn from imperfect historical data without exploiting surrogate errors. To diversify archive quality levels, we use a mixed-quality survival operator that occasionally reverses the nondominated-front processing order, which is described in Appendix C.4.

C.2 EVALUATION PROTOCOL

For each evaluation task parameter $\boldsymbol{\theta} \in \Theta_{\mathrm{eval}}$, a method returns an ordered list or finite set of K candidate designs

$$S_{\boldsymbol{\theta}} = \{\hat{\boldsymbol{x}}^{(1)}, \ldots, \hat{\boldsymbol{x}}^{(K)}\} \subseteq \mathcal{X}.$$

The method may use the offline archive, learned surrogates, and its own internal optimization procedure, but it cannot query the true objective function F during inference. After inference, the benchmark evaluator computes $F(\hat{\boldsymbol{x}}^{(k)}, \boldsymbol{\theta})$ for reporting only.

Held-out task parameters. For every problem the held-out set is drawn once and shared by all methods and seeds. Its size follows the dimension of the task-parameter space: 50 parameters for one-dimensional Θ (all DF problems, the RE S1 instances, Turbine, Solar, and Lamp21), 100 for two-dimensional Θ (the RE S12 instances and UAV32), and 200 for three-dimensional Θ (the RE S123 instances). One-dimensional domains are sampled uniformly; higher-dimensional domains use a half-and-half mixture of uniform sampling and a Latin-hypercube design, so that both the interior and the extremes of the domain are represented. Held-out parameters are excluded from the archive by construction: no archived design was evaluated at any of them, and every method is scored on exactly the same set. The ablation study of Section 6 uses an earlier, smaller held-out set of 20 parameters per problem drawn by the same rule; it is an internal comparison at fixed parameters and does not enter the main ranking.

For minimization problems, we compute hypervolume using the nondominated subset of the returned candidates:

$$\mathrm{HV}\big(S_{\boldsymbol{\theta}}, \boldsymbol{r}_{\boldsymbol{\theta}}\big) = \lambda\left(\bigcup_{\boldsymbol{x}\in \mathrm{ND}(S_{\boldsymbol{\theta}})} \prod_{j=1}^{m} [f_j(\boldsymbol{x},\boldsymbol{\theta}), r_{\boldsymbol{\theta},j}] \right), \tag{19}$$

where $\lambda(\cdot)$ is the Lebesgue measure and $\boldsymbol{r}_{\boldsymbol{\theta}}$ is a task-specific reference point shared by all methods.

Objective normalization and reference point. All hypervolumes are computed on normalized objectives. For each problem and task parameter we fix an ideal point $\boldsymbol{z}_{\boldsymbol{\theta}}^{\mathrm{ideal}}$ and a nadir point $\boldsymbol{z}_{\boldsymbol{\theta}}^{\mathrm{nadir}}$, map every objective vector to

$$\tilde{y}_j = \frac{y_j - z_{\boldsymbol{\theta},j}^{\mathrm{ideal}}}{\max\left(z_{\boldsymbol{\theta},j}^{\mathrm{nadir}} - z_{\boldsymbol{\theta},j}^{\mathrm{ideal}},\ \epsilon\right)}, \qquad \boldsymbol{r}_{\boldsymbol{\theta}} = 1.1 \cdot \mathbf{1}, \tag{20}$$

and evaluate eq. (19) on $\tilde{\boldsymbol{y}}$ with the reference point $\boldsymbol{r}_{\boldsymbol{\theta}}$. The bounds are used for evaluation only; training, warm starts, and guidance use the archive-derived ideal-point estimate and make no oracle call. For the DF suite the analytic Pareto front is available at every task parameter, and the ideal and nadir points are its coordinate-wise extremes, so the bounds are exact and $\boldsymbol{\theta}$-dependent. For the RE suite we use the reference front distributed with the RE benchmark, which is defined per problem rather than per task parameter, so one pair of bounds covers all task parameters of a problem. For the REAL suite no closed-form front exists; we use a best-known estimate, namely the coordinate-wise extremes over the oracle-evaluated candidate sets of every method and seed at that task parameter, which includes archived designs because the Nearest-Archive control returns archived designs scored at the test parameter. All bounds are frozen and released with the benchmark, so the evaluation is reproducible and does not shift when the method set changes.

Normalization matters because hypervolume is not invariant to rescaling individual objectives, and the archived objective ranges within one problem are very unequal on the engineering suites. Taking the ratio between the widest and the narrowest archived range as a measure, it is $1002\times$ on RE91-S1, $193\times$ on RE34-S1, $39\times$ on UAV32, and $31\times$ on RE41-S1, whereas it stays below $7\times$ on the DF problems and below $2\times$ on the other REAL problems. Without normalization the comparison on the engineering suites would be decided almost entirely by a single objective. A second effect concerns the reference point: a margin taken as a fixed multiple of the raw archive maximum lands far outside the objective range whenever that range is narrow relative to its offset (on RE34-S1 the first objective spans $[1660, 1700]$, so a 10% multiplicative margin sits four spans beyond the maximum), whereas the construction in eq. (20) places the margin relative to each objective's own range. Because the bounds do not depend on the archive, hypervolumes also remain comparable across experiments that modify the archive, such as the archive-composition sweep of Section H.5.

The empirical expected hypervolume is

$$\widehat{\mathrm{EHV}} = \frac{1}{|\Theta_{\mathrm{eval}}|} \sum_{\boldsymbol{\theta} \in \Theta_{\mathrm{eval}}} \mathrm{HV}\big(S_{\boldsymbol{\theta}}, \boldsymbol{r}_{\boldsymbol{\theta}}\big). \tag{21}$$

Since the scale of HV can differ substantially across task parameters and benchmark problems, we use rank-based aggregation for the main cross-problem summaries. For each problem, seed, and task parameter, methods are ranked by $\mathrm{HV}(S_{\boldsymbol{\theta}}, \boldsymbol{r}_{\boldsymbol{\theta}})$, with rank 1 assigned to the best method and ties assigned average ranks. We then average ranks over task parameters, random seeds, and problems within each benchmark suite.

We do not report the standard deviation of raw HV over task parameters as a primary summary. In PMOO, $\mathrm{HV}(S_{\boldsymbol{\theta}}, \boldsymbol{r}_{\boldsymbol{\theta}})$ may vary sharply with $\boldsymbol{\theta}$ even for the true Pareto front, so the standard deviation over i.i.d. test parameters conflates intrinsic task-parameter variation with method instability. Figure 2 illustrates this effect for the analytical DF fronts. We therefore report empirical EHV and use rank-based aggregation over paired methods; per-problem rank variability over random seeds is provided in Appendix J.

C.3 DF Problem Statistics

The DF suite contains 14 synthetic dynamic multi-objective problems. Following the CEC dynamic MOO benchmark, we reinterpret the original time variable as a continuous task parameter $\theta \in [0, 1]$. All DF problems use $d = 10$ decision variables and a single task parameter $p = 1$. DF1–DF9 are bi-objective problems, while DF10–DF14 are tri-objective problems. Table 3 reports the decision-variable bounds, task-parameter range, average archive size, and reference IGD for each problem.

C.4 Mixed-Quality Survival Operator

Standard NSGA-II survival retains nondominated and high-crowding-distance solutions, which biases the archive toward near-optimal regions. For offline optimization, however, a useful archive should

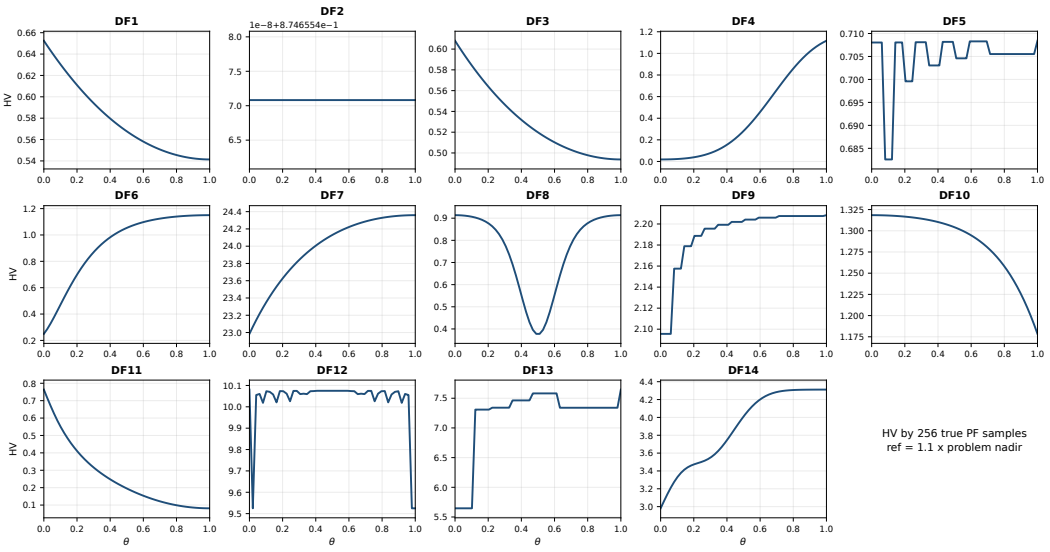


Figure 2: Hypervolume variation of the analytical Pareto fronts across the parameter space for DF1–DF14. For each problem, 50 uniformly spaced parameter values are evaluated.

Table 3: Problem setup for the DF benchmark suite. All problems have d=10, p=1, and $|\Theta|$=20. DF1–DF9 are bi-objective; DF10–DF14 are tri-objective. The columns for x_1, x_2, $x_3, \ldots, x_{10}$, and θ report their ranges.

Problem	m	x_1	x_2	$x_3, \ldots, x_{10}$	θ	$\bar{N}$	IGD↓
DF1	2	$[0, 1]$	$[0, 1]$	$[0, 1]$	$[0, 1]$	26 912	0.1974
DF2	2	$[0, 1]$	$[0, 1]$	$[0, 1]$	$[0, 1]$	26 968	0.0994
DF3	2	$[0, 1]$	$[-1, 2]$	$[-1, 2]$	$[0, 1]$	26 720	0.7156
DF4	2	$[-2, 2]$	$[-2, 2]$	$[-2, 2]$	$[0, 1]$	27 179	1.8979
DF5	2	$[0, 1]$	$[-1, 1]$	$[-1, 1]$	$[0, 1]$	26 791	0.4990
DF6	2	$[0, 1]$	$[-1, 1]$	$[-1, 1]$	$[0, 1]$	26 440	5.4849
DF7	2	$[1, 4]$	$[0, 1]$	$[0, 1]$	$[0, 1]$	26 259	0.3945
DF8	2	$[0, 1]$	$[-1, 1]$	$[-1, 1]$	$[0, 1]$	26 604	0.2138
DF9	2	$[0, 1]$	$[-1, 1]$	$[-1, 1]$	$[0, 1]$	26 448	0.5909
DF10	3	$[0, 1]$	$[0, 1]$	$[-1, 1]$	$[0, 1]$	29 409	0.2464
DF11	3	$[0, 1]$	$[0, 1]$	$[0, 1]$	$[0, 1]$	30 036	0.4852
DF12	3	$[0, 1]$	$[0, 1]$	$[-1, 1]$	$[0, 1]$	29 815	0.6158
DF13	3	$[0, 1]$	$[0, 1]$	$[-1, 1]$	$[0, 1]$	29 735	1.0535
DF14	3	$[0, 1]$	$[0, 1]$	$[-1, 1]$	$[0, 1]$	30 242	0.5226

also contain dominated and intermediate-quality designs. We therefore introduce a mixed-quality survival operator that occasionally reverses the nondominated-front processing order during survival selection. At each generation, with probability p_{ama}, the standard front-processing order is reversed: survivors are first drawn from the worst front $\mathcal{F}_{K-1}$, then $\mathcal{F}_{K-2}$, and so on. Crowding-distance tie-breaking is applied on the critical front regardless of the processing direction, ensuring tournament selection remains well-defined. Population snapshots are collected every Δg generations throughout the evolutionary run, capturing solutions at varying quality levels.

Algorithm 1 gives the pseudocode. Setting $p_{\text{ama}} = 0$ recovers standard NSGA-II (expert-only), while larger values produce an increasingly exploratory mix; we use $p_{\text{ama}} = 0.3$ on the Synthetic and RE suites and $p_{\text{ama}} = 0.5$ on REAL (Table 5). Section H.5 quantifies what this operator buys. Its effect is asymmetric across method families: restricting an archive to elite designs costs a surrogate baseline up to 25% held-out EHV, since a model that must support global search needs to see poor designs too, while leaving the archive-anchored generative solvers essentially unchanged. The operator therefore matters most for the predict-then-optimize baselines this benchmark also hosts.

Algorithm 1 Mixed-Quality Rank-and-Crowding Survival

Require: Combined population P of size 2μ; survival size μ; exploratory probability p_{ama}
Ensure: Survivor population S of size μ

Compute non-dominated fronts $\mathcal{F}_0, \mathcal{F}_1, \ldots, \mathcal{F}_{K-1}$ from P
if $\text{rand}() < p_{\text{ama}}$ **then**
  order $\leftarrow (K{-}1,\ K{-}2,\ \ldots,\ 0)$ {Exploratory: worst first}
else
  order $\leftarrow (0,\ 1,\ \ldots,\ K{-}1)$ {Expert: best first}
end if
$S \leftarrow \emptyset$
for k **in** *order* **do**
  Compute crowding distance d_i for each $i \in \mathcal{F}_k$
  Set $\text{rank}(i) \leftarrow k$, $\text{crowding}(i) \leftarrow d_i$ for all $i \in \mathcal{F}_k$
  if $|S| + |\mathcal{F}_k| \leq \mu$ **then**
    $S \leftarrow S \cup \mathcal{F}_k$
  else
    Sort $\mathcal{F}_k$ by d_i descending; take first $\mu - |S|$ individuals into S
    break
  end if
end for
return S

C.5 Data-Collection Hyperparameters

Table 4 lists the variation operators, which are shared by all suites, and Table 5 lists the per-suite collection budgets. The budgets differ by suite because the evolutionary horizon needed to populate an archive slice with a useful spread of quality levels depends on the difficulty of the underlying problem: the analytic DF problems and the RE engineering problems converge quickly, so a short horizon with frequent snapshots and a moderate exploratory probability already yields a mixed-quality slice, whereas the REAL conditional simulators need a substantially longer run.

Table 4: NSGA-II variation operators used for archive construction. These settings are shared by all three benchmark suites.

Parameter	Symbol	Value
Population size	μ	100
SBX crossover prob.	p_c	0.9
SBX distribution index	η_c	15
PM distribution index	η_m	20
PM mutation prob.	p_m	$1/d$

Table 5: Per-suite data-collection budgets. n_{gen} is the evolutionary horizon per task parameter, Δg the snapshot interval, and p_{ama} the exploratory-survival probability of Appendix C.4. The REAL suite uses a long horizon for grid parameters and a short randomized horizon for the additional random parameters.

Suite	Problems	n_{gen}	Δg	p_{ama}
Synthetic	DF1–DF9	20	2	0.3
	DF10–DF14	30	2	0.3
RE w/ SC	RE34, RE41	30	2	0.3
	RE91	40	2	0.3
REAL	Turbine, Solar, Lamp21, UAV32	200 / 40	5	0.5

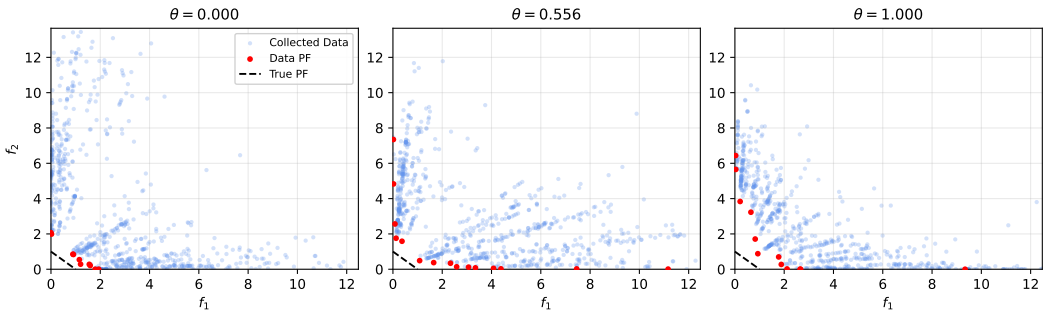


Figure 3: Offline data distribution on DF9 under three representative parameter values. Blue points denote the collected offline objective vectors, red points denote the nondominated front extracted from the offline dataset, and the black dashed curves denote the analytical Pareto fronts.

C.6 Task-Parameter Sampling Details

C.6.1 Stratified Grid Sampling

Given the parameter space $\Theta = [\theta^L, \theta^U]$ (scalar for the DF suite) and n_g grid points per dimension, we construct a uniform grid

$$\theta^{(k)} = \theta^L + \frac{k-1}{n_g - 1}(\theta^U - \theta^L), \quad k = 1, \ldots, n_g. \tag{22}$$

For multi-dimensional $\Theta \subseteq \mathbb{R}^p$, the full grid is the Cartesian product of per-dimension grids, yielding n_g^p points. For each grid point, a complete NSGA-II run is performed and snapshots are collected every Δg generations. The default is $n_g = 10$.

Grid points can be iterated sequentially or sampled in random order. For a query parameter $\boldsymbol{\theta} \notin \Theta_{\text{grid}}$, the k nearest grid points in Euclidean distance can be retrieved for local interpolation.

C.6.2 Randomized Sampling

To complement the fixed grid and improve inter-grid interpolation, we additionally sample parameters from the continuous space using one of the following strategies:

Uniform sampling. Each parameter is drawn independently as $\theta_i \sim \mathcal{U}(\theta_i^L, \theta_i^U)$.

Latin Hypercube Sampling (LHS). For each dimension, the interval $[\theta_i^L, \theta_i^U]$ is divided into n_s equal strata; exactly one sample is drawn uniformly within each stratum, and the per-dimension samples are randomly permuted. This guarantees marginal uniformity with fewer samples than full factorial designs.

Sobol sequence. A scrambled Sobol quasi-random sequence of length n_s is generated in $[0, 1]^p$ and affinely mapped to Θ, providing low-discrepancy coverage.

Local region sampling. For targeted exploration around a specific θ_0, samples are drawn uniformly within a hypersphere of radius ρ: $\theta = \theta_0 + \rho\, r\, \mathbf{d}/\|\mathbf{d}\|$ with $\mathbf{d} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $r \sim \mathcal{U}(0, 1)$, clipped to Θ.

For one-dimensional task spaces, we use a mixture of uniformly spaced grid points and randomized samples. For higher-dimensional task spaces, we use sparse grids together with LHS or Sobol samples to avoid the exponential growth of full Cartesian grids.

D RE Benchmark Suite

This appendix describes the RE-based shared-component PMOO tasks. Unless otherwise specified, all RE tasks follow the archive construction and evaluation protocol in Appendices C.1 and C.2. We use the original RE-suite problem identifiers RE34, RE41, and RE91, and construct three shared-component variants for each problem: S1, S12, and S123.

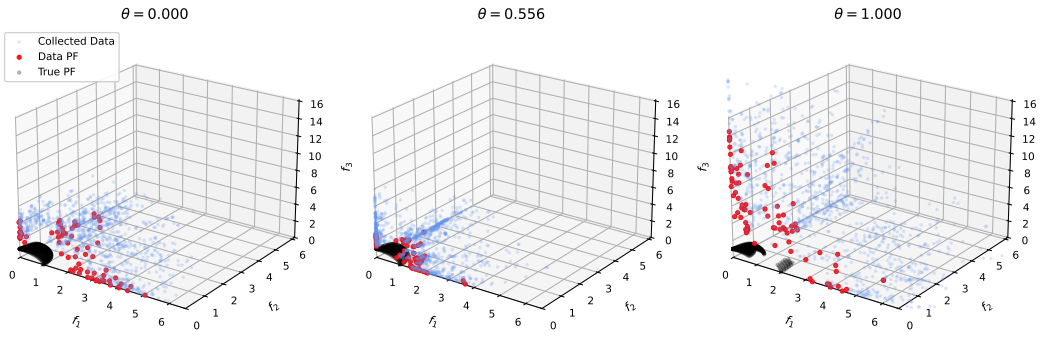


Figure 4: Offline data distribution on DF12 under three representative parameter values. Black points denote samples from the analytical Pareto front.

Table 6: Selected RE problems for shared-component PMOO. We use the original RE-suite identifiers throughout the paper.

Problem	D	m	Engineering context
RE34	5	3	Vehicle frontal crashworthiness; reinforced component thicknesses
RE41	7	4	Car side-impact design; load-path member thicknesses
RE91	7	9	Car cab design; structural thicknesses and crash/NVH objectives

D.1 Shared-Component Formulation

Let the original RE problem be defined on a full design vector $\boldsymbol{z} \in \mathcal{Z} \subseteq \mathbb{R}^D$ with objective vector $G(\boldsymbol{z}) \in \mathbb{R}^m$. For a shared-component configuration, we partition the design indices into a task-parameter index set I_θ and an optimizable-design index set I_x. The task parameter and design variable are

$$\boldsymbol{\theta} = \boldsymbol{z}_{I_\theta}, \qquad \boldsymbol{x} = \boldsymbol{z}_{I_x}.$$

Given $(\boldsymbol{x}, \boldsymbol{\theta})$, the full RE design is reconstructed by inserting the shared components into their original coordinates:

$$\boldsymbol{z} = \Pi(\boldsymbol{x}, \boldsymbol{\theta}).$$

The corresponding parametric objective is

$$F(\boldsymbol{x}, \boldsymbol{\theta}) = G\big(\Pi(\boldsymbol{x}, \boldsymbol{\theta})\big). \tag{23}$$

Thus, each fixed value of $\boldsymbol{\theta}$ defines a distinct engineering MOO instance in which selected structural components are shared across task variants.

D.2 Problem Selection

We select three problems from the RE suite (Tanabe & Ishibuchi, 2020) based on two criteria: (i) all decision variables correspond to *structural thickness variables* of physical components, providing a natural and industrially motivated shared-component interpretation (e.g., thickness standardization, part reuse, and manufacturing simplification); and (ii) the problems span a range of objective dimensionalities ($m \in \{3, 4, 9\}$), allowing us to study the interaction between parametric conditioning and many-objective complexity.

Table 6 summarizes the three selected problems.

D.3 Shared-Component Configurations

For each RE problem, we identify candidate shared components through engineering analysis of the original design context. In automotive structural design, component standardization across vehicle variants or production generations is primarily driven by three considerations: (i) the component serves as a *primary load-bearing structure* whose geometry is commonly inherited across platforms; (ii) standardizing the component yields significant *tooling and manufacturing cost reduction* due

Table 7: Shared-component configurations for the RE-based PMOO tasks. p is the task-parameter dimension and d is the number of remaining optimizable design variables.

Problem	Task ID	Shared $\boldsymbol{\theta}$	p	d	m
RE34	RE34-S1	$\{z_1\}$	1	4	3
RE34	RE34-S12	$\{z_1, z_2\}$	2	3	3
RE34	RE34-S123	$\{z_1, z_2, z_3\}$	3	2	3
RE41	RE41-S1	$\{z_1\}$	1	6	4
RE41	RE41-S12	$\{z_1, z_2\}$	2	5	4
RE41	RE41-S123	$\{z_1, z_2, z_3\}$	3	4	4
RE91	RE91-S1	$\{z_1\}$	1	6	9
RE91	RE91-S12	$\{z_1, z_2\}$	2	5	9
RE91	RE91-S123	$\{z_1, z_2, z_3\}$	3	4	9

to high part commonality; and (iii) the component simultaneously *influences multiple conflicting objectives*, making it a critical design parameter whose value is often fixed early in the development process. Based on these criteria, we rank candidate variables by standardization priority for each problem and construct three sharing levels (S1, S12, and S123) by progressively including the next highest-priority candidate. This naturally results in a nested structure that also allows us to study how increasing parametric dimensionality p affects offline generalization.

RE41 (Car side-impact design). The seven decision variables represent the thicknesses of structural members along the lateral load path of a passenger vehicle (Jain & Deb, 2013): z_1 (B-pillar inner), z_2 (B-pillar reinforcement), z_3 (floor side inner), z_4 (cross member), z_5 (door beam), z_6 (door belt-line reinforcement), and z_7 (roof rail). The B-pillar is the *primary energy-absorbing member* in a side collision and its cross-section is commonly carried over across vehicle platforms that share the same body-in-white architecture; its inner panel (z_1) is therefore the highest-priority candidate for standardization. The B-pillar reinforcement (z_2) works in direct structural conjunction with z_1 and is typically designed and manufactured as a matching pair, making it the natural second candidate. The floor side inner (z_3) serves as a secondary load-path member connecting the B-pillar to the vehicle floor structure; it is frequently shared across sedan and SUV variants built on a common platform. We therefore define the three sharing levels as $\boldsymbol{\theta} = \{z_1\}$, $\{z_1, z_2\}$, and $\{z_1, z_2, z_3\}$.

RE34 (Vehicle frontal crashworthiness). This problem involves five thickness variables of reinforced components in the frontal structure of a passenger vehicle (Liao et al., 2008), with three objectives related to weight, acceleration characteristics, and toe-board intrusion. In frontal-impact design, the foremost structural members, those closest to the point of impact, are the first to be standardized because they define the initial crush zone and are shared across powertrain variants (e.g., ICE vs. hybrid) that differ primarily in rear-cabin packaging. Following this front-to-rear priority, z_1 is the outermost frontal reinforcement and the highest-priority candidate, z_2 is the adjacent supporting member, and z_3 is the connecting cross-reinforcement. The sharing levels are thus $\boldsymbol{\theta} = \{z_1\}$, $\{z_1, z_2\}$, and $\{z_1, z_2, z_3\}$.

RE91 (Car cab design). The seven decision variables represent thicknesses of structural components governing the cab of a heavy vehicle (Deb & Jain, 2013), with nine objectives capturing diverse crash and NVH (noise, vibration, and harshness) performance metrics. Because the cab is a self-contained module that is often mounted onto different chassis configurations, components that form the cab's primary skeleton are natural standardization candidates: they remain identical when the same cab module is reused across truck variants. Variable z_1 corresponds to the main cab frame thickness and has the strongest influence on the largest number of objectives, making it the first candidate. Variables z_2 and z_3, representing the adjacent frame reinforcement and the front-panel thickness respectively, are the next most influential structural members as indicated by sensitivity analysis in the original study. The sharing levels follow as $\boldsymbol{\theta} = \{z_1\}$, $\{z_1, z_2\}$, and $\{z_1, z_2, z_3\}$.

Table 7 summarizes all nine resulting task instances. The nested structure $\{z_1\} \subset \{z_1, z_2\} \subset \{z_1, z_2, z_3\}$, which coincides across the three problems, follows from the per-problem engineering analysis described above.

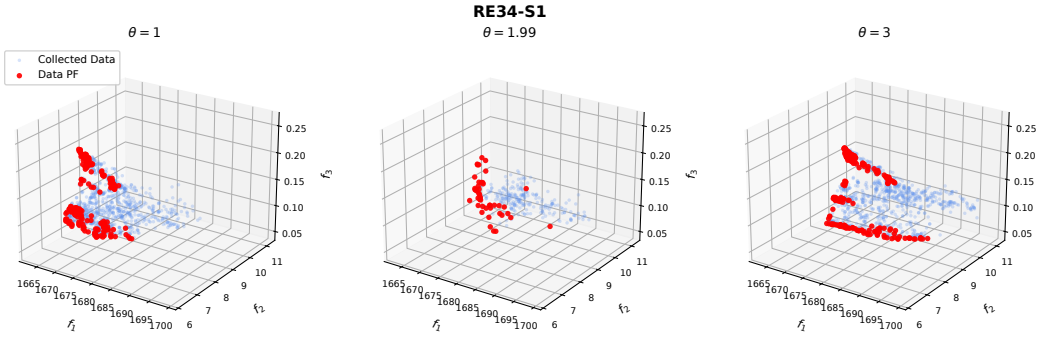


Figure 5: Objective-space distribution of the offline dataset for RE34-S1 under three representative parameter values. Blue points denote collected objective vectors, and red points denote the Data PF obtained by nondominated sorting in the full three-objective space.

D.4 Dataset Statistics

We follow the shared archive construction protocol in Appendix C.1, using the mixed-quality survival operator in Appendix C.4 and the task-parameter sampling strategies in Appendix C.6. Because the task parameter $\boldsymbol{\theta}$ now lives in the original variable bounds rather than a unit interval, we adapt the stratified grid to sample n_g points uniformly within the feasible range of each shared variable.

Table 8 reports the task-parameter bounds, number of sampled task parameters, and average archive size for each RE task. Since the true Pareto fronts are unknown, IGD is not reported.

Table 8: Dataset statistics for the RE benchmark suite. $|\Theta|$ is the number of sampled parameter values per dataset. $\bar{N}$ is the average number of evaluated solutions per seed.

Task ID	m	$\boldsymbol{\theta}$ bounds	$\lvert\Theta\rvert$	$\bar{N}$
RE34-S1	3	$[1, 3]$	20	7,083
RE34-S12	3	$[1, 3]^2$	28	11,051
RE34-S123	3	$[1, 3]^3$	34	15,330
RE41-S1	4	$[0.5, 1.5]$	20	7,271
RE41-S12	4	$([0.5, 1.5], [0.45, 1.35])$	28	11,149
RE41-S123	4	$([0.5, 1.5], [0.45, 1.35], [0.5, 1.5])$	34	15,997
RE91-S1	9	$[0.5, 1.5]$	20	7,720
RE91-S12	9	$([0.5, 1.5], [0.45, 1.35])$	28	11,238
RE91-S123	9	$([0.5, 1.5], [0.45, 1.35], [0.5, 1.5])$	34	16,088

E Real-World Benchmark Suite

This appendix describes the REAL benchmark suite. Unlike the DF suite, these tasks do not have closed-form Pareto fronts. Unlike the RE shared-component suite, the task parameter $\boldsymbol{\theta}$ is an exogenous operating condition rather than a fixed subset of the design vector. Unless otherwise specified, archive construction and evaluation follow the shared protocol in Appendices C.1 and C.2.

Each task defines a family of multi-objective optimization problems

$$\min_{\boldsymbol{x}\in\mathcal{X}} F(\boldsymbol{x}, \boldsymbol{\theta}) = \big(f_1(\boldsymbol{x}, \boldsymbol{\theta}), \ldots, f_m(\boldsymbol{x}, \boldsymbol{\theta})\big), \qquad \boldsymbol{\theta} \in \Theta. \tag{24}$$

All design and context variables are represented in normalized coordinates before being mapped to their physical ranges inside the evaluator. All objectives are formulated as minimization objectives.

E.1 Problem Selection

We select four real-world PMOO tasks that span different sources of parametric variation and different levels of design complexity: wind-turbine design, solar-rooftop geometry, lamp geometry generation,

and UAV controller design. The first three-dimensional task, Turbine, provides a low-dimensional but physically interpretable sanity check. Solar introduces a structured geometric surface with smoothness regularization. Lamp21 is a higher-dimensional tri-objective CAD-style design task. UAV32 is a high-dimensional trajectory control task with a two-dimensional context, and is therefore the most challenging real-world instance in the suite.

Table 9 summarizes the selected problems. The task definitions are implemented through a unified evaluator interface that maps normalized designs and task parameters to physical variables before computing objective values.

Table 9: Real-world PMOO problem summary. d is the design dimension, p is the context dimension, and m is the number of objectives. All variables are represented in normalized coordinates in the learning interface.

Problem	d	p	m	Context	Objectives
Turbine	3	1	2	wind speed	mass and negative power output
Solar	9	1	2	building orientation	morning and evening solar losses
Lamp21	21	1	3	focal height	stability, mass, and focal-point error
UAV32	32	2	2	fuselage length and density	terminal error and energy

E.2 Problem Definitions

Turbine. The Turbine problem optimizes the geometry of a wind turbine under varying wind conditions. The design vector $\boldsymbol{x} = (x_1, x_2, x_3) \in [0,1]^3$ controls the turbine radius, turbine height, and blade pitch angle after rescaling to physical units. The scalar context $\theta \in [0,1]$ controls the wind speed. We use two objectives: (i) a normalized mass/volume proxy, and (ii) the negative of the estimated power output, so that maximizing generated power is converted into minimization. This problem is low-dimensional and smooth, and is used primarily to test whether an offline PMOO method can learn a basic conditional trade-off manifold without relying on excessive data.

Solar rooftop. The Solar problem optimizes a fixed-boundary rooftop surface for photovoltaic collection. We use a 5×5 grid and optimize only the interior 3×3 heights, giving $d = 9$ decision variables. Boundary heights are fixed, while $x_i \in [0,1]$ is rescaled to a physical height used by the geometric evaluator. The context $\theta \in [0,1]$ represents the building orientation and is mapped to an angle in $[0, 2\pi]$. The two objectives measure morning and evening collection losses, each regularized by a surface smoothness term:

$$f_{\text{morning}}(\boldsymbol{x}, \theta) = L_{\text{morning}}(\boldsymbol{x}, \theta) + \lambda_s S(\boldsymbol{x}), \qquad f_{\text{evening}}(\boldsymbol{x}, \theta) = L_{\text{evening}}(\boldsymbol{x}, \theta) + \lambda_s S(\boldsymbol{x}), \tag{25}$$

where $S(\boldsymbol{x})$ is a second-difference smoothness penalty and $\lambda_s = 0.2$ in the evaluator. This task tests whether an offline method can interpolate Pareto fronts over a continuous geometric context.

Lamp21. The Lamp21 problem optimizes a three-arm desk-lamp geometry. The decision vector has 21 normalized variables encoding the base vector, three arm vectors, and three hand/end-effector vectors. The scalar context is the desired focal height. We evaluate three objectives: a stability proxy based on the center of mass relative to the base support region, a material/mass proxy, and a focal quality objective measuring how well the lamp heads reach the desired focal location. Compared with Turbine and Solar, Lamp21 is both higher-dimensional and tri-objective, making it a stronger test of objective-conflict modeling and conditional surrogate accuracy.

UAV32. The UAV32 problem is a twin-rotor controller design task. The decision vector contains 32 thrust commands, corresponding to two rotor controls across 16 time steps. The two-dimensional context $\boldsymbol{\theta} = (\theta_1, \theta_2) \in [0,1]^2$ controls the vehicle length and structural density after rescaling. The evaluator simulates the controlled trajectory and reports two objectives: terminal navigation error relative to the target state and cumulative control energy. This task is the most challenging real-world problem in the suite because it combines high-dimensional controls, stateful dynamics, and a two-dimensional continuous context.

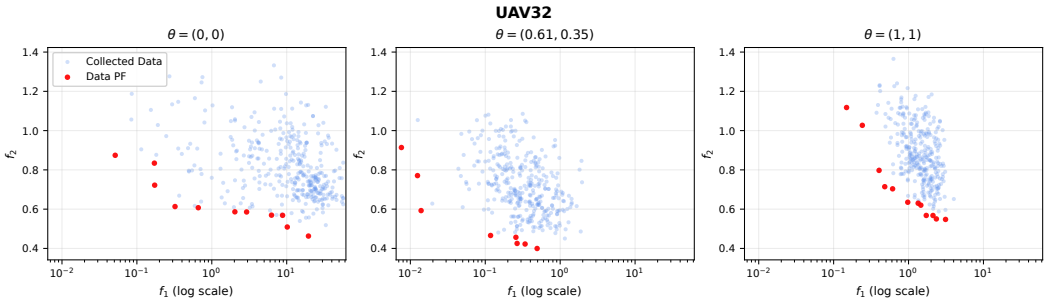


Figure 6: Objective-space distribution of the offline dataset for UAV32 under three representative parameter values. Blue points denote collected objective vectors, and red points denote the Data PF obtained by nondominated sorting in the two-objective space.

E.3 Dataset Statistics

We use the same offline data-collection principle as in Appendix C.3: expert evolutionary search is run at sampled context values and population snapshots are retained to form a static offline archive. To avoid making the real-world datasets unrealistically easy, the raw archives are downsampled at the level of each context value. This preserves coverage across Θ while limiting the number of highly correlated samples from the same evolutionary trajectory.

For each problem we generate five independent datasets with different random seeds. Table 10 reports the number of sampled context values, the per-seed dataset-size range, and the average number of retained samples per seed. Since these tasks do not provide analytic Pareto fronts, we do not report IGD for this suite.

Table 10: Dataset statistics for the REAL benchmark suite. Each problem uses 5 independent dataset seeds. $|\Theta|$ is the number of sampled training context values per seed. $\bar{N}$ is the average number of retained samples per seed after context-level downsampling.

Problem	$\lvert\Theta\rvert$	$N_{\min}$–$N_{\max}$	$\bar{N}$
Turbine	16	919–972	952.2
Solar	20	4,916–5,062	5,008.4
Lamp21	16	2,000–2,000	2,000.0
UAV32	28	9,647–10,020	9,759.6

The retained dataset sizes deliberately increase with problem difficulty. Turbine is smooth and low-dimensional, so a small archive is sufficient and prevents the baselines from saturating the task. Solar requires denser samples because the decision variables describe a coupled surface. Lamp21 uses more samples per context than Turbine because it is high-dimensional and tri-objective, but we cap the total size to avoid over-representing a small number of focal-height values. UAV32 receives the largest archive because it has both the highest decision dimension and a two-dimensional context space.

F METHOD AND IMPLEMENTATION DETAILS

F.1 SURROGATE ARCHITECTURE DETAILS

All surrogate models are trained on normalized design variables, task parameters, and objective values. The normalization constants are computed from the training archive only and are reused for validation and evaluation. For a task parameter $\boldsymbol{\theta}$, each surrogate predicts a normalized objective vector. Objective predictions are transformed back to the original objective scale only for reporting and benchmark evaluation. All objectives are treated as minimization objectives.

Cat. The Cat surrogate uses a standard MLP with input $[\boldsymbol{x}; \boldsymbol{\theta}]$. This is the most direct conditioning mechanism and provides a parameter-efficient baseline when the objective landscape varies smoothly over the task-parameter space Θ. We use ReLU activations and a linear output layer.

HN. The HN surrogate uses a hypernetwork to generate task-specific target-network parameters from $\boldsymbol{\theta}$. In our implementation, the target network maps $\boldsymbol{x}$ to the objective vector, while the hypernetwork maps $\boldsymbol{\theta}$ to the target-network weights. This makes the predictor explicitly task-adaptive, but the full generated-parameter space can be large when the target network has multiple hidden layers.

HN-LoRA. HN-LoRA reduces the number of task-dependent parameters generated by the hypernetwork. For each generated layer, the task-dependent weight is decomposed into a shared component and a low-rank residual:

$$W_\ell(\boldsymbol{\theta}) = W_\ell^0 + \frac{\alpha}{r} A_\ell(\boldsymbol{\theta}) B_\ell(\boldsymbol{\theta}).$$

The base weights W_ℓ^0 are learned jointly with the hypernetwork. The hypernetwork outputs only the low-rank factors $A_\ell(\boldsymbol{\theta})$ and $B_\ell(\boldsymbol{\theta})$, reducing the number of generated parameters from $O(h_\ell^{\text{out}} h_\ell^{\text{in}})$ to $O(r(h_\ell^{\text{out}} + h_\ell^{\text{in}}))$ per layer. This reduction is useful in Offline PMOO because task-parameter coverage is finite and sparse, making full task-specific weight generation prone to overfitting.

Network sizes. Cat-E2E, Cat-MM, and the HN-LoRA target networks use two hidden layers with dimensions $[256, 256]$. The HN-LoRA hypernetwork uses hidden dimensions $[128, 128]$. For Cat-MM-IOM, the representation network, forward predictor, and discriminator use hidden dimensions $[512, 512]$, $[256, 256]$, and $[256, 256]$, respectively. Surrogate networks use ReLU activations with linear output layers. The IOM discriminator uses LeakyReLU with negative slope 0.1.

F.2 MULTI-OBJECTIVE SURROGATE TRAINING

For E2E surrogates, all objectives share the same hidden representation and are optimized jointly. The base objective uses uniform weights $\lambda_j = 1/m$ in the regression loss $\mathcal{L}_{\text{sur}}(\psi) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{m} \lambda_j \left(\hat{f}_{\psi,j}(\boldsymbol{x}_i, \boldsymbol{\theta}_i) - y_{i,j}\right)^2$. GradNorm dynamically adjusts λ_j so that different objective losses train at comparable rates. PCGrad modifies the per-objective gradients before the shared parameters are updated: when two objective gradients have negative cosine similarity, one gradient is projected onto the normal plane of the other to reduce destructive interference.

For MM surrogates, each objective has an independent predictor and optimizer. This eliminates gradient conflict by construction and provides the objective-wise prediction interface used by the generative guidance wrapper. We also include IOM as an offline-optimization-oriented training variant. IOM regularizes the learned representation so that optimized out-of-distribution candidates become less distinguishable from archive-supported samples, reducing the tendency of surrogate optimization to exploit unsupported high-confidence regions. We use IOM regularization coefficient $\lambda_{\text{IOM}} = 0.5$.

F.3 SURROGATE SEARCH AND WARM-START CONSTRUCTION

Surrogate-only search. For standalone surrogate solvers, we run NSGA-II on the frozen surrogate $\hat{F}_\psi(\cdot, \boldsymbol{\theta}_{\text{eval}})$ for each evaluation task parameter. The optimizer operates only on surrogate predictions

and never queries the true objective during inference. The true objective oracle is used only after inference for benchmark scoring.

Guidance surrogate. PSD and PPFM use the Cat-MM-B surrogate, which consists of independently trained objective predictors

$$\hat{\boldsymbol{F}}_\psi(\boldsymbol{x}, \boldsymbol{\theta}) = \left[\hat{f}_{\psi,1}(\boldsymbol{x}, \boldsymbol{\theta}), \ldots, \hat{f}_{\psi,m}(\boldsymbol{x}, \boldsymbol{\theta})\right]^\top.$$

Using objective-wise predictors avoids shared-representation gradient interference during test-time guidance.

We use Cat-MM-B as a fixed guidance surrogate for all generative experiments to avoid selecting a different guidance model per benchmark suite and to keep the comparison between PSD and PPFM controlled. The MM structure provides objective-wise gradients without shared-representation interference, which is important for scalarized test-time guidance. The same guidance surrogate is used for all PSD/PPFM variants and ablations.

Preference vectors and scalarization. For each evaluation task parameter, the main experiments generate $K = 256$ preference vectors on every suite. The generative solvers and the oracle-free controls return K candidates per task parameter, and the surrogate solvers return the final NSGA-II population of 100 designs. Preference vectors are drawn as

$$\boldsymbol{w}^{(k)} \sim \mathrm{Dirichlet}(1, \ldots, 1), \qquad \boldsymbol{w}^{(k)} \in \Delta^{m-1}.$$

In the generation-budget study, we set $K = q$ to match the requested generation budget. For a preference vector $\boldsymbol{w}$, inference uses the augmented Tchebycheff scalarization

$$J_{\boldsymbol{w}}(\boldsymbol{x}, \boldsymbol{\theta}) = \max_{1 \le j \le m} w_j \left(\hat{f}_{\psi,j}(\boldsymbol{x}, \boldsymbol{\theta}) - \hat{z}^\star_j(\boldsymbol{\theta})\right) + \rho \sum_{j=1}^{m} w_j \left(\hat{f}_{\psi,j}(\boldsymbol{x}, \boldsymbol{\theta}) - \hat{z}^\star_j(\boldsymbol{\theta})\right),$$

with augmentation coefficient $\rho = 10^{-4}$. The ideal point estimate is computed without oracle queries from the guidance surrogate and the initialization pool $\mathcal{C}_{\boldsymbol{\theta}_{\mathrm{eval}}}$ defined below:

$$\hat{z}^\star_j(\boldsymbol{\theta}_{\mathrm{eval}}) = \min_{\boldsymbol{x} \in \mathcal{C}_{\boldsymbol{\theta}_{\mathrm{eval}}}} \hat{f}_{\psi,j}(\boldsymbol{x}, \boldsymbol{\theta}_{\mathrm{eval}}).$$

All scalarization and guidance computations are performed in normalized objective space.

Archive-supported seed pool. Let $\boldsymbol{\theta}_{\mathrm{nn}}$ denote the matched training parameter if $\boldsymbol{\theta}_{\mathrm{eval}}$ is present in the archive, and otherwise the nearest archived training parameter in normalized task-parameter space. For each preference vector $\boldsymbol{w}^{(k)}$, we select an archive-supported seed from the data slice associated with $\boldsymbol{\theta}_{\mathrm{nn}}$ by minimizing the same augmented Tchebycheff scalarization under that matched or nearest training parameter. In this seed-selection step, the scalarization uses the normalized archived objectives in the nearest archive slice and the corresponding archive-slice ideal point

$$z^\star_{\mathrm{arch},j}(\boldsymbol{\theta}_{\mathrm{nn}}) = \min_{(\boldsymbol{x}, \boldsymbol{\theta}_{\mathrm{nn}}, \boldsymbol{y}) \in \mathcal{D}_{\boldsymbol{\theta}_{\mathrm{nn}}}} y_j.$$

The archive-supported seed is selected as

$$\boldsymbol{x}^{(k)}_{\mathrm{seed}} \in \arg \min_{(\boldsymbol{x}, \boldsymbol{\theta}_{\mathrm{nn}}, \boldsymbol{y}) \in \mathcal{D}_{\boldsymbol{\theta}_{\mathrm{nn}}}} \left[\max_{1 \le j \le m} w^{(k)}_j (y_j - z^\star_{\mathrm{arch},j}(\boldsymbol{\theta}_{\mathrm{nn}})) + \rho \sum_{j=1}^{m} w^{(k)}_j (y_j - z^\star_{\mathrm{arch},j}(\boldsymbol{\theta}_{\mathrm{nn}}))\right].$$

If no valid archive-supported seed is available, the implementation falls back to a uniformly sampled feasible design. This produces an oracle-free initialization pool $\mathcal{C}_{\boldsymbol{\theta}_{\mathrm{eval}}} = \{\boldsymbol{x}^{(k)}_{\mathrm{seed}}\}_{k=1}^{K}$ with the same size as the generation budget.

Projected-gradient warm start. Before local refinement, we add a small jitter of magnitude 0.01 times the box span to each selected seed and project it back to the feasible box. Starting from the jittered seed, we run 25 projected-gradient steps on the scalarized Cat-MM-B surrogate:

$$\boldsymbol{g}_t = \nabla_{\boldsymbol{x}} J_{\boldsymbol{w}}(\boldsymbol{x}_t, \boldsymbol{\theta}_{\mathrm{eval}}), \qquad \bar{\boldsymbol{g}}_t = \frac{\boldsymbol{g}_t}{\|\boldsymbol{g}_t\|_2 + 10^{-12}},$$

$$\boldsymbol{x}_{t+1} = \Pi_{\mathcal{X}}\left(\boldsymbol{x}_t - \eta \bar{\boldsymbol{g}}_t\right), \qquad \eta = 0.05.$$

Here $\Pi_{\mathcal{X}}$ denotes projection to the box constraints $[\boldsymbol{x}_l, \boldsymbol{x}_u]$, equivalently to $[0,1]^d$ in normalized design coordinates. We do not apply an additional clipping threshold to guidance gradients. The projected warm-start routine uses unit-normalized gradients, while generative model training uses gradient clipping with norm 1.0.

F.4 Generative Model Details

Elite-set construction. The generator is trained on an elite subset of the archive rather than on all collected designs. For each archived task parameter, we perform nondominated sorting within the corresponding data slice and retain the first Pareto front. Equivalently,

$$\mathcal{D}_{\text{elite}} = \bigcup_{\boldsymbol{\theta} \in \Theta_{\text{arch}}} \{(\boldsymbol{x}, \boldsymbol{\theta}) : (\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{y}) \in \mathcal{D},\ \boldsymbol{x} \in \text{ND}(\mathcal{D}_{\boldsymbol{\theta}})\}.$$

The resulting elite set concentrates training on designs that are empirically associated with strong trade-offs under each task parameter. The generator is trained to model a task-conditioned elite-design distribution $p_\phi(\boldsymbol{x}^\star \mid \boldsymbol{\theta})$.

Preference-agnostic generator. The generator is not conditioned on the preference vector during training. The offline archive contains tuples $(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{y})$ but does not provide reliable labels indicating which preference vector produced each design. Moreover, the same design can be compatible with multiple scalarizations. We therefore train a preference-agnostic but task-conditional generator and introduce preferences only at inference time through surrogate warm starts and surrogate guidance.

Unified task-conditional backbone. Both PSD and PPFM use a time-dependent MLP backbone N_ϕ that receives the current state, continuous time, and task parameter:

$$\boldsymbol{h}_t = [\boldsymbol{x}_t; t; \boldsymbol{\theta}].$$

The output of N_ϕ is interpreted either as a diffusion noise prediction for PSD or as a velocity field for PPFM. Generative models are trained on normalized design variables and task parameters. The generator does not receive $\boldsymbol{w}$ as an input.

Parametric Scalarized Diffusion. PSD is implemented as a conditional DDPM on the elite set. We use a linear DDPM beta schedule with $\beta_1 = 10^{-4}$, $\beta_T = 0.02$, and $T = 100$ diffusion steps. Given $(\boldsymbol{x}_0, \boldsymbol{\theta}) \sim \mathcal{D}_{\text{elite}}$, noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I)$, and a sampled diffusion step t, the forward process is written as

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon},$$

where α_t and σ_t are determined by the DDPM schedule. The network predicts the injected noise using the standard noise-matching objective:

$$\mathcal{L}_{\text{PSD}}(\phi) = \mathbb{E}_{t, \boldsymbol{x}_0, \boldsymbol{\theta}, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta})\|_2^2\right].$$

At inference time, PSD starts from a partially corrupted warm start with corruption level $\tau = 0.15$:

$$\boldsymbol{x}_\tau = \alpha_\tau \boldsymbol{x}_{\text{ws}} + \sigma_\tau \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I).$$

We then run the ancestral DDPM reverse sampler from the partially corrupted state to the clean endpoint. The reverse process is guided by the scalarized surrogate objective:

$$\tilde{\boldsymbol{\epsilon}}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}, \boldsymbol{w}) = \boldsymbol{\epsilon}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}) + s\, \sigma_t \nabla_{\boldsymbol{x}_t} J_{\boldsymbol{w}}(\boldsymbol{x}_t, \boldsymbol{\theta}),$$

with guidance scale $s = 1.0$. The final design is projected to $\mathcal{X}$ before evaluation. Under the epsilon-parameterized DDPM sampler, adding $s\sigma_t \nabla_{\boldsymbol{x}_t} J_{\boldsymbol{w}}$ to the predicted noise shifts the corresponding clean-sample estimate toward the descent direction $-\nabla_{\boldsymbol{x}_t} J_{\boldsymbol{w}}$.

Parametric Pareto Flow Matching. PPFM is implemented using conditional flow matching on the elite set. Let $\boldsymbol{x}_1 \sim p_{\text{elite}}(\cdot \mid \boldsymbol{\theta})$ be an elite design and $\boldsymbol{x}_0 \sim \mathcal{N}(0, I)$ be a base sample. We use the linear probability path

$$\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1, \qquad \boldsymbol{u}_t = \boldsymbol{x}_1 - \boldsymbol{x}_0,$$

with training objective

$$\mathcal{L}_{\text{PPFM}}(\phi) = \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{x}_1,\boldsymbol{\theta}} \left[\| \boldsymbol{v}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}) - (\boldsymbol{x}_1 - \boldsymbol{x}_0) \|_2^2 \right].$$

At inference time, PPFM uses a corruption fraction $\tau = 0.15$ and initializes at flow time $t_0 = 1 - \tau$:

$$\boldsymbol{x}_\tau = \tau \boldsymbol{z} + (1-\tau)\boldsymbol{x}_{\text{ws}}, \qquad \boldsymbol{z} \sim \mathcal{N}(0, I).$$

The guided ODE is then integrated from $t = t_0$ to $t = 1$ using the Euler solver with 100 inference steps:

$$\frac{d\boldsymbol{x}_t}{dt} = \boldsymbol{v}_\phi(\boldsymbol{x}_t, t, \boldsymbol{\theta}) - \gamma(t) \nabla_{\boldsymbol{x}_t} J_{\boldsymbol{w}}(\boldsymbol{x}_t, \boldsymbol{\theta}), \tag{26}$$

where the guidance schedule increases linearly in flow time,

$$\gamma(t) = \gamma t, \qquad \gamma = 10.0.$$

Early integration is therefore dominated by the learned elite-design transport, while later integration applies stronger preference-specific surrogate guidance near the data endpoint. The final design is projected to $\mathcal{X}$ before evaluation.

Warm-started oracle-free reconstruction. For an evaluation parameter $\boldsymbol{\theta}_{\text{eval}}$, PSD and PPFM generate one refined candidate for each sampled preference vector. The output set is

$$\mathcal{S}^{\text{gen}}_{\boldsymbol{\theta}_{\text{eval}}} = \{\hat{\boldsymbol{x}}^{(1)}, \ldots, \hat{\boldsymbol{x}}^{(K)}\}.$$

All steps used to construct this set are oracle-free: the procedure uses the offline archive, the trained Cat-MM-B surrogate, and the trained generator, but does not query the true objective at $\boldsymbol{\theta}_{\text{eval}}$ during inference. The benchmark oracle is used only after the candidate set has been returned.

Figure 7 provides a qualitative view of the warm-started generative refinement process on two representative DF tasks. The gray points denote archive samples, the orange points denote oracle-free warm-start candidates obtained from surrogate scalarization, and the blue/green points denote the final candidates after PSD and PPFM refinement, respectively. The oracle Pareto front is shown only for visualization and is never used during inference.

The visualization illustrates two properties of the proposed generative solvers. First, the warm-start candidates already provide task- and preference-dependent initialization near promising objective regions, but they can still be scattered or biased by the learned surrogate. Second, PSD and PPFM further concentrate the candidates toward the Pareto-front region while remaining close to the archive-supported high-quality manifold. This supports the intended role of the generator: it does not replace preference guidance, but regularizes surrogate-driven search and reduces the risk of unconstrained surrogate exploitation. On DF4, both refinement methods produce candidates closely aligned with the smooth front. On DF9, where the objective-space geometry is more irregular, the refined candidates still move toward the lower-front region but exhibit less uniform coverage, reflecting the greater difficulty of recovering irregular conditional Pareto fronts from a fixed offline archive.

F.5 Training and Inference Hyperparameters

Table 11 summarizes the main hyperparameters used for surrogate training, generative model training, warm-start construction, and surrogate-only NSGA-II inference. Unless otherwise specified for an ablation, the same standard configuration is used across benchmark suites and random seeds. PSD and PPFM are trained once on the offline archive and then reused for all queried task parameters. At inference time, both methods use the same projected-gradient warm-start procedure, followed by either guided denoising for PSD or guided ODE integration for PPFM.

All experiments were run on an NVIDIA RTX A4000 GPU with 16 GB memory, an Intel Core i7-13700 CPU with 24 logical threads, and 64 GB RAM. Surrogate training took approximately 1-3 minutes per problem-seed; PSD and PPFM training took about 1 minute and 0.5-1 minute, respectively.

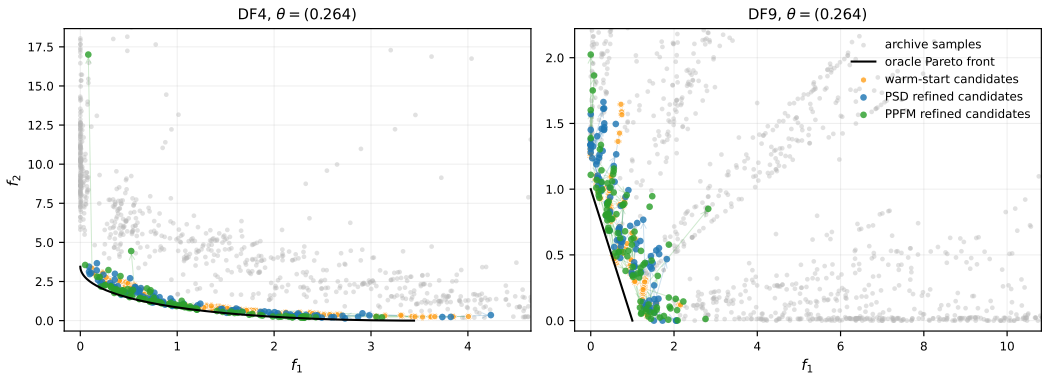


Figure 7: Warm-started generative refinement on DF4 and DF9 at $\theta = 0.264$. PSD and PPFM refine oracle-free surrogate warm starts toward the Pareto-front region while remaining close to archive-supported high-quality samples. The oracle Pareto front is shown only for visualization and is not used during inference.

Table 11: Training and inference hyperparameters used in the standard configuration.

Component	Hyperparameter	Value
Surrogates	optimizer	Adam
Surrogates	learning rate	10^{-3}
Surrogates	batch size / epochs	256 / 100
Surrogates	early stopping patience	20
Surrogates	hidden dimensions	Cat-E2E, Cat-MM, HN-LoRA target: $[256, 256]$; HN-LoRA hypernet: $[128, 128]$
Cat-MM-IOM	rep. / fwd. / disc. hidden dimensions	$[512, 512]$ / $[256, 256]$ / $[256, 256]$
Surrogates	activation	ReLU; IOM discriminator uses LeakyReLU(0.1); output layers are linear
HN-LoRA	rank and scaling	$r = 8$, $\alpha = 16$; hence $\alpha/r = 2$
IOM	regularization coefficient	$\lambda_{\mathrm{IOM}} = 0.5$
Guidance	surrogate	Cat-MM-B
Guidance	scalarization	augmented Tchebycheff with $\rho = 10^{-4}$
Guidance	PSD guidance scale	$s = 1.0$
Guidance	PPFM guidance schedule	$\gamma(t) = \gamma t$, $\gamma = 10.0$
Guidance	gradient clipping	no explicit clipping for guidance gradients; warm start uses unit-normalized gradients; generative training uses clip norm 1.0
Preference sampling	number and rule	$K = 256$ per task on every suite; $\boldsymbol{w} \sim \mathrm{Dirichlet}(1, \ldots, 1)$; budget study uses $K = q$
Warm start	candidate-pool size	same as K
Warm start	pool construction	per preference, select the archive candidate minimizing the same scalarization under the matched/nearest training parameter; fallback is uniform random in bounds
Warm start	jitter, PGD, and projection	jitter 0.01 times box span; 25 projected-gradient steps; learning rate 0.05; projection to $[\boldsymbol{x}_l, \boldsymbol{x}_u]$
PSD	backbone / learning rate	$[1024, 1024, 1024]$ / 2×10^{-4}
PSD	noise schedule	linear DDPM beta schedule, $\beta_1 = 10^{-4}$ to $\beta_T = 0.02$, $T = 100$
PSD	partial corruption and sampler	$\tau = 0.15$; ancestral DDPM reverse sampler from partially corrupted warm start
PPFM	backbone / learning rate	$[1024, 1024]$ / 2×10^{-3}
PPFM	warm-start time and solver	$\tau = 0.15$; Euler ODE solver; 100 inference steps
Generative training	optimizer / schedule	AdamW with cosine learning-rate schedule
Generative training	batch size / epochs	256 / 200
PSD/PPFM inference	number of refinement steps	100
NSGA-II (test-time surrogate search)	population size / generations	100 / 100
NSGA-II (test-time surrogate search)	initialization	uniform random in $[\boldsymbol{x}_l, \boldsymbol{x}_u]$; *not* archive-warm-started
NSGA-II (test-time surrogate search)	variation operators	SBX $p_c = 0.9$, $\eta_c = 15$; polynomial mutation $\eta_m = 20$, $p_m = 1/d$

G Additional Result Tables

This appendix collects the supporting tables referenced from Section 6: the train-parameter ranking (Appendix G.1), the breakdown by objective count (Appendix G.2), the distance-based and diversity metrics (Appendix G.3), and the matched-budget many-objective study (Appendix G.4).

G.1 Train-Parameter Ranking

Table 2 in the main text reports held-out task parameters, which is the quantity of interest in Offline PMOO. Table 12 adds the ranking on training parameters for the same 16 methods, together with the archive-only diagnostic $\mathcal{D}_{\text{best}}$ (Appendix H.3), which exists only at training parameters. Ranks are computed exactly as in Table 2, so the held-out columns reproduce its left block. At a training parameter, Nearest-Archive returns the nondominated designs of that parameter's own archive slice.

Table 12: Average rank on training and held-out task parameters by benchmark suite under normalized hypervolume, for the methods of Table 2 and the archive-only diagnostic $\mathcal{D}_{\text{best}}$, which exists only on training parameters. Best per column in bold, second best underlined.

	Training parameters			*Held-out parameters*		
Method	Synthetic	RE w/ SC	REAL	Synthetic	RE w/ SC	REAL
Archive-only diagnostic						
$\mathcal{D}_{\text{best}}$	12.33	3.84	14.42	-	-	-
Oracle-free controls						
Nearest-Archive	11.77	<u>3.17</u>	14.34	11.90	3.14	13.91
Warm-Start-Only	7.88	4.21	8.57	7.70	3.28	8.30
PPSL-MOBO	9.84	13.78	11.56	9.55	12.52	11.88
Task-conditioned surrogate solvers						
Cat-E2E-B	11.25	11.29	9.18	11.13	10.64	8.72
Cat-E2E-G	8.63	10.25	6.97	8.27	8.97	**6.27**
Cat-E2E-P	8.30	10.30	7.00	7.90	9.28	<u>6.34</u>
Cat-MM-B	10.79	13.35	9.47	10.37	12.75	9.04
Cat-MM-I	10.04	10.77	<u>6.85</u>	9.69	10.22	6.72
HN-E2E-B	9.99	11.99	10.24	9.60	11.23	10.36
HN-E2E-G	7.56	9.85	7.42	7.27	8.42	6.54
HN-E2E-P	7.41	10.38	7.77	6.70	9.31	7.70
HN-LoRA-B	9.57	12.01	9.10	9.47	11.24	10.21
HN-LoRA-G	7.88	10.56	**6.84**	8.04	9.69	6.95
HN-LoRA-P	7.20	11.06	7.07	7.27	10.39	8.17
Generative refinement (ours)						
PSD	<u>6.52</u>	3.72	7.66	<u>5.85</u>	<u>3.02</u>	6.97
PPFM	**6.04**	**2.47**	8.53	**5.26**	**1.91**	7.91

The strongest methods are the same on both kinds of parameters: PPFM and PSD lead the Synthetic suite, PPFM leads RE w/ SC, and surrogate solvers lead REAL. At a training parameter the archive's own nondominated designs are available, as $\mathcal{D}_{\text{best}}$ and as the output of Nearest-Archive, yet they rank near the bottom on Synthetic and REAL and behind PPFM on RE w/ SC, so the learned solvers do more than recover archived solutions. The exception is the nine-objective RE91 family, where the archived fronts are the best entries even at training parameters, the same regime in which Nearest-Archive leads at held-out parameters.

G.2 Breakdown by Objective Count

The benchmark spans $m \in \{2, 3, 4, 9\}$, and the right block of Table 2 groups the held-out ranking of all 16 methods by objective count.

Two patterns are visible. Against every method that relies on the learned objective model, the advantage of the proposed solvers does not degrade with m and is largest at $m = 4$ and $m = 9$, where PPSL-MOBO falls to last place. Against the zero-learning control the picture is regime-dependent: Nearest-Archive improves monotonically with m ($12.73 \to 9.94 \to 1.78 \to 1.03$), is beaten by a wide margin at two and three objectives, overtakes PSD at $m = 4$, and is the single best of the 16 methods at $m = 9$. The effect does not come from objective scaling, since the ranking uses normalized hypervolume and the control's rank at $m = 9$ is the same under unnormalized hypervolume (1.04).

In this benchmark m is confounded with problem family, because the $m = 4$ and $m = 9$ groups are drawn entirely from the RE suite and contain three instances each, so the pattern is that archive reuse becomes stronger on the higher-objective RE tasks. Many-objective instances outside the RE family would separate the two factors, and many-objective guidance for offline parametric problems is the open problem these results locate.

G.3 Distance-Based and Diversity Metrics

Hypervolume conflates convergence and diversity and depends on the reference point. We therefore recompute the ranking on the 14 analytic DF problems using IGD+, IGD, and spacing, normalized per (problem, $\boldsymbol{\theta}$) by the range of the true front. We do not construct pseudo-fronts for the RE and REAL suites, where any reference set would itself be oracle-derived and would make the comparison circular.

Table 13: Average rank of 16 methods on the 14 analytic DF problems under distance-based and diversity metrics, with the Synthetic HV rank repeated for reference. Lower is better for all four columns.

Method	IGD+	IGD	spacing	HV (Synthetic)
PPFM	**4.81**	**4.52**	8.44	**5.26**
PSD	5.31	5.39	7.29	5.85
HN-E2E-G	6.27	6.29	8.44	7.27
Warm-Start-Only	7.74	7.74	6.35	7.70
PPSL-MOBO	9.84	9.59	**1.44**	9.55
Nearest-Archive	12.14	11.71	13.29	11.90

The distance metrics agree with hypervolume for the generative solvers, which addresses the concern that the observed advantage might be specific to HV. More interestingly, the two families of metrics separate sharply for PPSL-MOBO: it attains the best spacing of any method (1.44) while ranking 9.84 on IGD+ and 9.55 on HV. This is the signature of a front that is perfectly uniform but placed away from the true one, a failure that neither a convergence metric nor a diversity metric would reveal on its own, and it illustrates why we report both.

G.4 Matched-Budget Many-Objective Search

A natural question is whether the surrogate baselines are understated because NSGA-II is not designed for many objectives. We therefore repeat the surrogate search with NSGA-III at matched population size on the six many-objective RE tasks, across four surrogate configurations and five seeds. Reference directions are sized to the population budget rather than by a fixed partition count, so that the two algorithms receive the same number of candidate evaluations.

Table 14: Held-out normalized EHV of the best surrogate configuration under NSGA-II and NSGA-III at matched population. Higher is better.

Task	NSGA-II (ours)	NSGA-III
RE41-S1	**0.2287**	0.2253
RE91-S1	**0.0209**	0.0164
RE91-S12	**0.0200**	0.0140
RE91-S123	**0.0197**	0.0150

NSGA-III never beats NSGA-II beyond seed noise on any problem or surrogate configuration tested, and it is clearly weaker on the nine-objective tasks. A many-objective-specialized optimizer therefore does not strengthen the surrogate baselines under this protocol, and the NSGA-II results reported in the main text already represent that class at full strength. This also means the strength of Nearest-Archive at high m (Appendix G.2) cannot be attributed to a weak surrogate searcher.

G.5 Performance Across the Task Domain: Remaining Problems

Figure 1 in the main text shows DF12 and Solar. Figure 8 gives the remaining four problems, and Figure 9 shows predicted fronts at four held-out task parameters, which make the aggregate hypervolume numbers concrete.

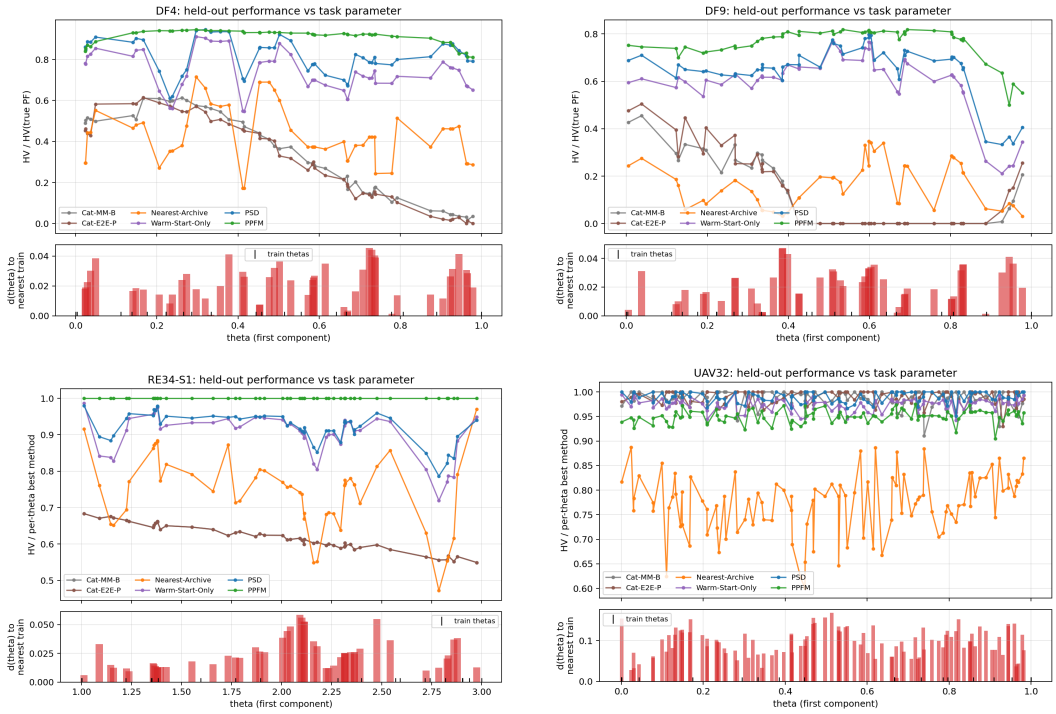


Figure 8: Held-out performance versus task parameter for the remaining four diagnostic problems, with distance to the nearest training parameter beneath each panel. DF4 and DF9 are normalized by the true front; RE34-S1 and UAV32 by the per-$\boldsymbol{\theta}$ best method.

G.6 Boundary Saturation Under Guided Refinement

Guiding a diffusion or flow model with surrogate gradients could in principle drive states out of the feasible box, so that the final projection dominates the output. We measure this with the *boundary saturation* rate, the fraction of design coordinates lying exactly on x_l or x_u, over every returned design on seven problems, approximately 0.7M designs in total. Returned sets are projected onto the box and restricted to designs with finite objective values before they are scored, so the audit concerns the designs a practitioner receives. Plain projected-gradient warm starts on the same surrogate provide the reference: a guided process that keeps pushing the state out of the feasible region should show an elevated rate relative to that reference.

Table 15: Box-boundary saturation of the returned designs, approximately 0.7M in total. "PGD" is the projected-gradient warm start on the same surrogate, with no generative refinement.

Problem	PGD	PSD	PPFM
DF4	0.05%	0.14%	0.16%
DF9	2.44%	3.25%	6.32%
DF12	4.03%	6.48%	7.81%
RE34-S1	73.47%	**29.67%**	74.82%
RE91-S1	64.92%	**12.36%**	**9.55%**
Solar	11.69%	**4.06%**	**3.61%**
UAV32	0.07%	0.15%	0.80%
mean	22.38%	**8.02%**	**14.73%**

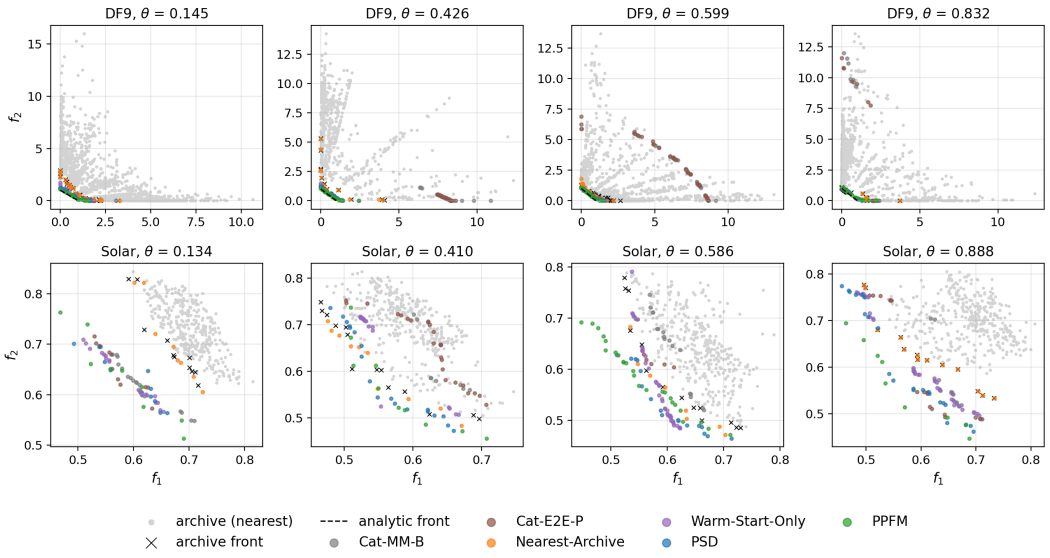


Figure 9: Predicted fronts at four held-out task parameters on DF9 (top row, with the analytic front) and Solar (bottom row), shown against the archive slice of the nearest training parameter and its nondominated front. The generative solvers track the front as $\boldsymbol{\theta}$ changes, whereas the surrogate baselines lose coverage at the extremes of the trade-off.

Boundary contact is not by itself evidence of instability, since some optima genuinely lie on the box, which is why we report it relative to projected gradient under identical conditions.

The saturation pattern should be read per problem rather than only in the mean. Where the surrogate strongly drives designs toward the box (RE34-S1, RE91-S1, and Solar, the three problems with double-digit PGD saturation), generative refinement reduces saturation substantially, by a factor of 2.5 to 6.8 for PSD and, on RE91-S1 and Solar, for PPFM as well. On the four problems where saturation is low for every method (DF4, DF9, DF12, UAV32, all below 8%), the guided processes sit slightly *above* the projected-gradient reference in absolute terms. The mean is therefore dominated by the high-saturation problems, and PPFM's overall mean is held up by RE34-S1, where it matches rather than improves on the reference. The consistent statement is narrower than the mean alone suggests: *when surrogate gradients push hard against the feasible boundary, the elite prior restrains them*, and when they do not, refinement neither helps nor meaningfully hurts on this measure.

G.7 Inference-Time Hyperparameter Sensitivity

All main-table results use a single pre-specified configuration. To characterize how sharp that choice is, we sweep the inference-time settings while reusing the trained checkpoints, so that only test-time behaviour varies. Table 16 reports held-out EHV as a percentage change from the default under the same sweep protocol.

Three observations. First, the defaults are not knife-edge: across a $30\times$ range of the guidance strength γ, held-out EHV moves by at most 3.4%, and PSD's guidance scale s moves it by at most 0.4% anywhere. Second, the sensitivity that does exist is concentrated in a single mechanism, the strength of PSD's refinement: a weaker corruption ($\tau = 0.05$) or fewer refinement steps costs 6–8% on DF4, DF10, and Solar, a stronger corruption ($\tau = 0.30$) gains up to 6.5%, and RE34-S1 moves by at most 1.2% on the same axes. PPFM is markedly less sensitive: its corruption strength stays within $\pm 0.9\%$, and fewer ODE steps never hurt it (up to $+2.9\%$ at 25 steps). Third, the direction of the γ effect is problem-dependent, decreasing on RE34-S1 and increasing on the other three tasks, which is precisely why we do not tune it per task.

Every comparison in the main text uses the single pre-specified default, and no per-task setting is derived from this sweep, since it is scored with held-out labels that an offline practitioner would not have. Selecting the refinement strength per task without held-out labels is a natural next step.

Table 16: Inference-time sensitivity: percentage change in held-out normalized EHV relative to the default configuration. All entries reuse the same trained checkpoints; entries beyond $\pm 4\%$ are in bold.

Group	Setting	DF4	DF10	RE34-S1	Solar
PPFM guidance scale	$\gamma = 1$	-0.8	-3.2	$+1.0$	-1.5
	$\gamma = 3$	-1.0	-1.2	$+1.1$	-1.2
	$\gamma = 10$ (default)	$+0.0$	$+0.0$	$+0.0$	$+0.0$
	$\gamma = 30$	$+3.4$	$+1.3$	-1.4	$+2.4$
PPFM schedule	constant vs. linear	$+0.6$	$+0.2$	-0.8	$+1.2$
PSD guidance scale	$s = 0.3$	$+0.1$	$+0.0$	-0.2	$+0.0$
	$s = 3.0$	-0.3	-0.0	$+0.4$	-0.0
PSD corruption τ	$\tau = 0.05$	$\mathbf{-6.3}$	$\mathbf{-7.1}$	-1.2	$\mathbf{-8.3}$
	$\tau = 0.30$	$\mathbf{+6.5}$	$+2.7$	$+1.0$	$\mathbf{+5.3}$
PPFM corruption τ	$\tau = 0.05$	$+0.0$	-0.1	$+0.4$	-0.3
	$\tau = 0.30$	$+0.0$	$+0.9$	-0.8	$+0.1$
PSD refinement steps	25	$\mathbf{-5.8}$	$\mathbf{-7.0}$	-1.1	$\mathbf{-7.7}$
	50	-3.0	-3.8	-0.5	-3.7

G.8 Configuring the Method Without an Oracle

A method for a zero-query setting must also be *configurable* without queries, or the setting is only nominally offline. We therefore state our own selection protocol explicitly.

The single configuration reported throughout the paper was fixed as follows. Candidate settings were compared using only objective labels already recorded in the archive at *training* task parameters, values the archive contains by construction, requiring no new evaluation. The configuration was then frozen and applied unchanged to all 27 problem instances. Held-out test parameters and the ground-truth oracle were never consulted during this process, and no new oracle calls were made at any point. This is why no per-problem tuning appears anywhere in the main results, and why reproducing our numbers requires no search.

For a practitioner deploying on a new archive, we recommend the same principle in its cleanest form: withhold a subset of your archived task parameters from model fitting, select hyperparameters against those parameters' already-recorded labels, freeze, and then deploy. Because the withheld parameters are archived, their objective values are already known, so the procedure consumes zero oracle budget while still measuring generalization to parameters the model did not fit.

Three oracle-free indicators additionally let a user judge whether a given query falls inside the supported regime before trusting the output: the nearest-task distance $h(\boldsymbol{\theta})$, the local elite count near $\boldsymbol{\theta}$, and the support distance from returned candidates to the nearest archived design. High values of the first two flag the F1 and F2 modes of Appendix A.5, and a large support distance flags F3. Each indicator targets one documented failure mode, and calibrating their detection thresholds is a natural next step.

H Detailed Experimental Analyses

This appendix gives the full tables and discussion behind the ablations and diagnostics summarized in Section 6, together with the warm-start construction, the PPSL-MOBO baseline, and the archive-only diagnostic used in the per-problem tables.

H.1 Warm-Start Construction

For an unseen task parameter $\boldsymbol{\theta}_{\text{eval}}$, the proposed PSD and PPFM are used as warm-started refinement operators. We first sample K preference vectors $\{\boldsymbol{w}^{(k)}\}_{k=1}^{K} \subset \Delta^{m-1}$. For each preference, we construct an oracle-free warm start by minimizing the surrogate scalarization,

$$\boldsymbol{x}_{\text{ws}}^{(k)} \approx \arg\min_{\boldsymbol{x}\in\mathcal{X}} J_{\boldsymbol{w}^{(k)}}(\boldsymbol{x}, \boldsymbol{\theta}_{\text{eval}}), \tag{27}$$

using only the learned surrogate and a candidate pool derived from the offline archive: the pool is the union of the archived slices nearest to $\boldsymbol{\theta}_{\text{eval}}$, the starting design for preference $\boldsymbol{w}^{(k)}$ is the pool member with the smallest surrogate scalarization, and projected gradient descent on eq. (27) is run from it for a fixed number of steps with projection onto $\mathcal{X}$ after every step. The warm start is then refined by PSD or PPFM. For PSD, we partially corrupt $\boldsymbol{x}_{\text{ws}}^{(k)}$ to an intermediate noise level τ and run the guided reverse diffusion process. For PPFM, we mix $\boldsymbol{x}_{\text{ws}}^{(k)}$ with a base sample at an intermediate flow time and integrate the guided ODE toward the data endpoint. The final output is

$$S_{\boldsymbol{\theta}_{\text{eval}}}^{\text{gen}} = \left\{\hat{\boldsymbol{x}}^{(1)}, \ldots, \hat{\boldsymbol{x}}^{(K)}\right\}, \tag{28}$$

which is evaluated only after inference by the benchmark oracle. The resulting procedure lies between pure surrogate optimization and pure generation. The surrogate supplies task- and preference-specific directions, while the generator regularizes the proposals toward the learned high-quality design manifold. The corruption level, the number of refinement steps, the guidance schedule, and the warm-start step count are listed with the other hyperparameters in Appendix F.5.

Algorithm 2 Warm-started generative Pareto reconstruction

Require: Task parameter $\boldsymbol{\theta}_{\text{eval}}$, surrogate $\hat{F}_\psi$, generator G_ϕ, candidate pool $\mathcal{C}_{\boldsymbol{\theta}_{\text{eval}}}$, preference set $\{\boldsymbol{w}^{(k)}\}_{k=1}^{K}$

1: Estimate $\hat{\boldsymbol{z}}^\star(\boldsymbol{\theta}_{\text{eval}})$ from $\hat{F}_\psi$ and $\mathcal{C}_{\boldsymbol{\theta}_{\text{eval}}}$ without oracle queries.
2: **for** $k = 1, \ldots, K$ **do**
3: Select the starting design $\boldsymbol{x}_0^{(k)} \leftarrow \arg\min_{\boldsymbol{x}\in\mathcal{C}_{\boldsymbol{\theta}_{\text{eval}}}} J_{\boldsymbol{w}^{(k)}}(\boldsymbol{x}, \boldsymbol{\theta}_{\text{eval}})$.
4: Obtain the warm start $\boldsymbol{x}_{\text{ws}}^{(k)}$ by projected gradient descent on Eq. (27), initialized at $\boldsymbol{x}_0^{(k)}$.
5: Refine $\boldsymbol{x}_{\text{ws}}^{(k)}$ using PSD reverse denoising in Eq. (9) or PPFM guided ODE integration in Eq. (10).
6: Project the refined design to $\mathcal{X}$, yielding $\hat{\boldsymbol{x}}^{(k)}$.
7: **end for**
8: **return** $S_{\boldsymbol{\theta}_{\text{eval}}} = \{\hat{\boldsymbol{x}}^{(k)}\}_{k=1}^{K}$

H.2 PPSL-MOBO Baseline

Pareto set learning maps a preference vector to a Pareto-optimal design and was introduced for expensive multi-objective optimization by Lin et al. (2022); its parametric form, which conditions the map jointly on a task parameter and a preference, $h(\boldsymbol{\theta}, \boldsymbol{w}) \to \boldsymbol{x}$, is due to PPSL-MOBO (Cheng et al., 2026; 2025). That parametric form is what the Offline PMOO setting calls for, so we adopt it here, replacing the Gaussian process with the neural surrogate this benchmark trains on the archive. The map is trained to minimize the expected augmented-Tchebycheff value of its own prediction over sampled task parameters and Dirichlet preferences, on the same frozen surrogate, ideal point, and candidate budget used by PSD and PPFM, and makes no oracle call at any stage. We implement the task conditioning by concatenation rather than the hypernetwork with low-rank adaptation used in the original; the hypernetwork and LoRA conditioning structures appear separately in the HN-E2E and HN-LoRA surrogate families, neither of which changes the qualitative picture.

Table 17: Ablation of the proposed refinement procedure under normalized hypervolume on four representative tasks. Each block gives the mean held-out value of the full method and the relative change when one component is removed (five seeds). All rows share the two-front elite setting and the ablation study's own fixed held-out parameters (Appendix C.2).

	DF4	DF10	RE34-S1	Solar
PSD (full)	0.845	1.024	0.727	0.779
w/o warm start	$+14.4\%$	-12.2%	$+6.3\%$	-5.9%
w/o guidance	$+0.2\%$	$+0.0\%$	-0.3%	$+0.0\%$
w/o elite selection	$+2.1\%$	-0.4%	-0.7%	$+0.6\%$
PPFM (full)	0.972	0.992	0.794	0.802
w/o warm start	$+0.0\%$	$+0.6\%$	$+0.7\%$	-1.4%
w/o guidance	-1.2%	-10.3%	-4.8%	-1.8%
w/o elite selection	-0.3%	-13.8%	-0.9%	-7.0%

H.3 Archive-Only Diagnostic

The per-problem tables of Appendix J additionally report an archive-only diagnostic $\mathcal{D}_{\mathrm{best}}$ on *training* parameters. For each $\boldsymbol{\theta} \in \Theta_{\mathrm{arch}}$, $\mathcal{D}_{\mathrm{best}}$ is constructed from the corresponding archive slice $\mathcal{D}_{\boldsymbol{\theta}}$ by extracting at most 64 designs by nondominated sorting with crowding-distance truncation in objective space. It uses only objective values already contained in the offline archive, and is not defined for held-out task parameters because no archive slice exists at unseen $\boldsymbol{\theta}$.

H.4 Ablation of the Refinement Procedure

Table 17 removes one component at a time, and the two solvers turn out to rely on different parts of the procedure. For PPFM, scalarized guidance is the decisive component: removing it costs between 1% and 10% on every task, and the elite prior is the second, with losses of 14% on DF10 and 7% on Solar once the generator is trained on the full archive. Its warm start is close to neutral, which is what one expects from a deterministic flow that transports mass onto the same elite manifold from either starting point. For PSD, guidance and elite selection change the result by well under 1% on three of four tasks, in line with the small guidance sensitivity reported in Appendix G.7, and the warm start is the component that matters. Its effect follows the quality of the archived starting designs at the queried parameter: with the warm start, DF10 and Solar are 12% and 6% better than from noise, whereas DF4 and RE34-S1 are 14% and 6% worse. PSD corrupts its starting design only partially ($\tau = 0.15$) and therefore stays close to it, so a starting design near the front is a strong start and one far from it is an anchor; the sensitivity study shows the same mechanism from the other side, with Solar rewarding heavier corruption. The ablation therefore identifies what to strengthen in each solver, guidance and the prior for PPFM and starting-design quality or corruption strength for PSD, and confirms that neither solver acts as an unconditional sampler.

H.5 When Does Archive Quality and Coverage Matter?

Condition A3 requires the archive to support the main trade-off regions near the query. We test how performance degrades when it does not, holding the per-slice sample count fixed and retraining every model from scratch for each variant, so that quality is decoupled from quantity: *expert* keeps only the best fronts ($p_{\mathrm{ama}} = 0$), *mixed* is the setting used throughout the paper, *low-quality* keeps only the worst fronts, and *sector-deleted* removes the f_1-specialized third of each slice. Four tasks $\times$ four variants $\times$ five seeds, with full retraining per variant.

All five variants are scored with the normalized protocol of Section 3, whose ideal and nadir bounds are frozen per task parameter and therefore do not depend on the archive a model was trained on. This matters more than it may appear: a reference point derived from each variant's own archive would give an elite-only archive, which by construction contains no poor designs, an objective box 0.49–$0.60\times$ the size of the full archive's on two of these tasks, and a mechanically smaller hypervolume regardless of design quality. Archive-independent bounds are required for any cross-variant comparison to mean anything.

Three readings follow from Table 18.

Table 18: Held-out normalized EHV under five archive variants, all scored with the same archive-independent bounds. *full* is the unfiltered archive used by the main experiments; the other four retain half of each slice and differ only in which half. Cat-MM-B is a surrogate baseline that trains on the whole archive; PSD trains its prior on the elite subset only.

Task	Method	full (100%)	expert	mixed	low-quality	sector-deleted
DF9	Cat-MM-B	**0.0707**	0.0103	0.0398	0.0507	0.0537
	PSD	**0.3951**	0.3465	0.3352	0.1087	0.3332
RE34-S1	Cat-MM-B	0.4932	0.4932	0.4932	0.4932	0.4932
	PSD	**0.7326**	0.7322	0.7322	0.6588	0.6616
Solar	Cat-MM-B	**0.5796**	0.5604	0.5348	0.4433	0.5053
	PSD	**0.7475**	0.7373	0.6908	0.5267	0.6542
UAV32	Cat-MM-B	**1.1280**	0.9650	0.9572	0.8906	0.9617
	PSD	**1.1310**	1.1190	1.1230	1.0540	1.1100

First, *which half of the archive is kept matters far less for the generative solver than for the surrogate*. On the three engineering tasks, keeping only the better half of every slice (expert) changes PSD's held-out value by at most 1.4%, and keeping a random half (mixed) by at most 7.6%. DF9 is the exception: its front is thin under the normalized metric, so every halving costs PSD 12–16%, which is still a fraction of what the surrogate loses. Since the main experiments train on the full archive, this says the method does not depend on a carefully curated one.

Second, *the surrogate baseline is not equally robust, and fails in the direction one would expect*. On DF9 and UAV32 the elite-only archive costs Cat-MM-B 85% and 14% respectively, while costing PSD 12% and 1%. A surrogate must model the objective landscape well enough for NSGA-II to search it, and an archive containing only near-optimal designs gives it no contrast from which to learn what makes a design poor. The generative prior is fitted to the elite subset in any case, and its warm start is anchored to archived designs, so removing the archive's poor half removes little it was using. Elite-biased data therefore penalizes precisely the method that must learn the whole landscape, and touches the archive-anchored method far less, which supports the design.

Third, the failures that do occur are *coverage* failures, matching condition A3. An archive of only low-quality designs is the worst variant on all four tasks for PSD, from 7% on UAV32 to 72% on DF9, and deleting an f_1-specialized preference sector costs 10–16% on three of the four. Both are cases where a region of the trade-off surface has no archived support near the query, which is the mode Theorem 1 shows no zero-query method can repair. What does *not* matter is the Pareto rank of the retained designs in itself. Separately, the elite set's *definition* also matters little: first front, first three fronts, or top 10% land within about one seed standard deviation of each other on every task (RE34-S1 PSD 0.735 / 0.722 / 0.736, s.d. ± 0.011).

H.6 Why Elite Selection Helps Here but Not in Fixed-Task Offline MOO

Section 5 argued that elite selection plays a different role under task conditioning than it does for a single fixed objective. We test that argument directly, with no oracle call and no retraining, by measuring how much information a design carries about *which* task parameter produced it. For each archived parameter we take the slice of designs, compute its centroid in min–max normalized design space, and form a Fisher-style task-conditional signal-to-noise ratio

$$\mathrm{SNR} = \frac{\text{between-}\boldsymbol{\theta}\text{ dispersion of slice centroids}}{\text{mean within-}\boldsymbol{\theta}\text{ dispersion about each centroid}}. \tag{29}$$

A higher SNR means the training distribution is more sharply separated by $\boldsymbol{\theta}$, which is exactly the structure a conditional generator must learn. We compute it twice per problem: over the full slice, and over its elite subset.

Elite selection raises the task-conditional SNR on five of seven problems, by up to $1.87\times$. More informative than the average is what the gain tracks: *how much the elite filter actually removes*. Where nondominated sorting retains under 2% of a slice (DF4, DF9) the SNR nearly doubles; where it retains a sixth or more (Solar, RE34-S1) the gain shrinks to 1.1–$1.2\times$. Note that Solar and UAV32 retain two fronts rather than one, following the published configuration, which is why their elite fractions exceed those of the other low-objective problems.

Table 19: Task-conditional signal-to-noise of the training distribution, mean over five seeds. "Elite %" is the fraction of each slice retained by nondominated sorting.

Problem	m	Elite %	SNR (full slice)	SNR (elite subset)	ratio
DF9	2	1.6%	0.451	0.842	**1.87**×
DF4	2	1.9%	0.313	0.561	**1.79**×
DF12	3	6.9%	0.263	0.335	1.27×
Solar	2	13.6%	0.640	0.758	1.18×
RE34-S1	3	16.3%	0.320	0.352	1.10×
UAV32	2	7.2%	0.632	0.595	0.94×
RE91-S1	9	86.5%	0.101	0.093	0.93×

The extreme case is instructive. On the nine-objective RE91-S1, 86.5% of each slice already lies on the first front: dominance ceases to discriminate when the objective count is large, so "elite selection" removes almost nothing, and the SNR ratio duly falls to 0.93. The design choice therefore has a well-defined scope: it is valuable exactly when nondominated sorting is selective enough to strip away the task-invariant bulk of the archive, and is expected to be inert when it is not.

That boundary coincides with an independent observation from Table 2: the zero-learning Nearest-Archive control becomes strongest in the many-objective regime, where a slice is already almost entirely nondominated and returning it wholesale yields a large, well-spread candidate set. The two findings share one cause, which makes the many-objective regime a documented boundary of the approach.

The SNR ratio identifies where elite selection can help and where it cannot, while the size of the resulting gain is set by other factors: DF4 shows the largest SNR gain, yet removing elite selection changes its normalized hypervolume by under 2%, whereas Solar shows a modest $1.18\times$ gain and a 7% drop for PPFM (Table 17). We therefore use the ratio to locate the regime in which the argument applies, and the ablation study to quantify what it buys there.

H.7 Unwitnessed Transitions: Where A2 Binds

Theorem 1 says an unwitnessed change in the conditional Pareto set is unidentifiable. To see what methods actually do in that regime, we construct a controlled family that interpolates between two well-separated branches,

$$F_\kappa(\boldsymbol{x}, \theta) = \big(1 - s_\kappa(\theta)\big) F_A(\boldsymbol{x}) + s_\kappa(\theta) F_B(\boldsymbol{x}), \tag{30}$$

where s_κ is a logistic of sharpness $\kappa \in \{2, 10, 50, \infty\}$ ($\kappa = \infty$ is a hard switch at $\theta = 0.5$) and F_A, F_B place their Pareto sets in well-separated design regions. We delete $\theta \in (0.4, 0.6)$ from every training archive, so no method observes the switch. Since the two branches have comparable hypervolume, an objective-space metric is weakly diagnostic here. We therefore measure *branch recovery* in design space, i.e. the deviation of returned designs from the branch the true transition selects (0 = correct branch, ≈ 0.6 = opposite branch).

Table 20: In-gap branch deviation as the transition sharpens. Lower is better; ≈ 0.6 would indicate committing to the wrong branch.

Method	$\kappa = 2$	$\kappa = 10$	$\kappa = 50$	$\kappa = \infty$
Cat-MM-B	0.027	0.029	0.104	0.119
Nearest-Archive	0.062	0.087	0.094	0.121
PSD	0.042	0.044	0.066	0.097
PPFM	0.052	0.059	**0.044**	**0.057**

The degradation is real and monotone: PSD's in-gap deviation at $\kappa = \infty$ is $2.7\times$ its own out-of-gap deviation, and the in-gap normalized-HV penalty grows with sharpness as predicted ($0.001 \to 0.102$ for PSD, $0.002 \to 0.040$ for PPFM). Two further observations matter for interpretation. The failure mode is *hedging, not collapse*: deviations reach ≈ 0.12 rather than ≈ 0.6, meaning the models interpolate between the two branches rather than committing to the unwitnessed one, the finite-sample signature of Theorem 1. And PPFM is markedly less sensitive to sharpness than PSD, staying flat at 0.044–0.059, which is the concrete benefit of releasing the flow solver alongside the diffusion solver. The binding regime is thus narrow and now measured: a sharp transition *together with* no archive coverage of the transition region.

H.8 Diagnosing Surrogate Exploitation

The decomposition in eq. (5) predicts that methods which optimize the learned model to convergence without support anchoring pay for it through $\varepsilon_{\mathrm{sur}}$. We test this with two oracle-free-to-compute diagnostics evaluated post hoc over six problems: the *optimism gap*, the normalized hypervolume a method attains under the surrogate's own predictions minus that under the oracle, and the *support distance*, the mean distance from a returned design to the nearest archived design.

Table 21: Surrogate-exploitation diagnostics, averaged over six problems. The optimism gap is in normalized hypervolume units; a positive value means the method's candidates look better to the surrogate than they are.

Method	optimism gap	support distance
Cat-MM-B (surrogate + NSGA-II)	$+0.15$	0.479
PPSL-MOBO (surrogate, differentiable map)	$+0.24$	0.413
Warm-Start-Only (surrogate PGD)	-0.05	0.303
PSD	-0.09	0.306
PPFM	-0.11	0.327
Nearest-Archive	-0.09	0.218

The only two positive means belong to the two methods that optimize the learned model to convergence, and on Solar they reach $+0.16$ and $+0.22$ against -0.22 (PSD) and -0.31 (PPFM). The archive-anchored methods also stay about 35% closer to the nearest archived design. We read this as evidence that the unanchored methods exploit the surrogate and that the elite prior restrains surrogate-driven updates. Because a negative gap on its own could also come from a globally pessimistic surrogate, we pair it with the support distance, which moves in the same direction. It does, however, explain why a stronger Pareto-set-learning baseline does not displace our design: offline, a preference-conditioned map trained purely against a learned model inherits that model's errors.

I Additional Efficiency, Robustness, and Qualitative Analyses

This appendix reports additional diagnostics beyond the main EHV comparison. We focus on two practical questions: (i) whether the proposed generative refinements provide a favorable test-time cost profile compared with repeatedly solving task-specific surrogate optimization problems, and (ii) whether their performance depends on returning a large number of generated candidates.

I.1 Test-Time Efficiency

Table 22 reports the average wall-clock inference time per held-out task parameter. All timings exclude one-time model training and include only oracle-free test-time inference. For surrogate-only methods, inference requires solving a fresh NSGA-II problem on the learned surrogate $\hat{F}_\psi(\cdot, \boldsymbol{\theta})$ for each queried $\boldsymbol{\theta}$. Thus, even after the surrogate has been trained, the optimization cost is repeatedly incurred across test parameters. In contrast, PSD and PPFM reuse a trained task-conditional generator and perform preference-conditioned refinement from oracle-free warm starts. The warm-start and guidance steps still introduce additional computation, but refinements for different preferences and different task parameters are naturally batched or parallelizable.

Table 22: Efficiency comparison. Each cell reports average wall-clock seconds per held-out task parameter, excluding one-time model training.

Method	DF4	DF10	RE34-S1	Solar
Cat-E2E-P	0.469s	0.462s	0.482s	0.450s
Cat-MM-I	0.507s	0.530s	0.562s	0.541s
HN-LoRA-P	0.514s	0.511s	0.543s	0.541s
PSD	**0.128**s	**0.204**s	**0.193**s	**0.127**s
PPFM	0.368s	0.531s	0.523s	0.706s

The observed runtime shows a clear advantage for PSD in the current implementation. PSD is the fastest method on all four diagnostic tasks, taking 0.128s, 0.204s, 0.193s, and 0.127s on DF4, DF10, RE34-S1, and Solar, respectively. This is substantially lower than the surrogate-only NSGA-II

baselines, whose inference times are around 0.45–0.56s per held-out task on the same problems. Therefore, PSD provides both amortization across task parameters and lower realized wall-clock cost in this setup.

PPFM exhibits a more nuanced profile. Although flow matching provides a deterministic refinement path and avoids stochastic reverse diffusion sampling, the current guided ODE implementation still requires repeated velocity-field and surrogate-gradient evaluations. As a result, PPFM is faster than the surrogate baselines on DF4, comparable on DF10 and RE34-S1, and slower on Solar. PPFM's advantage therefore lies in reconstruction quality, especially on the Synthetic and RE w/ SC suites, while PSD is the fast option. Reducing the number of ODE function evaluations or improving batched ODE execution is a natural implementation-level direction for improving PPFM's test-time efficiency.

Refinement steps. Reducing PPFM's guided ODE from 100 to 25 steps changes held-out EHV by between $+0.1\%$ and $+2.9\%$ on the four diagnostic tasks while reducing its end-to-end inference wall-clock to $0.49\times$ on DF4, $0.56\times$ on RE34-S1, and $0.90\times$ on Solar. The residual on Solar reflects the fixed cost of oracle-scoring the candidate set, which does not scale with step count. These are back-to-back ratios rather than absolute seconds, which are load-dependent and measure a different quantity from the oracle-free per-parameter timings of Table 22.

I.2 Generation-Budget Robustness

Table 23: Generation-budget robustness under $K = 10$. Each cell reports held-out normalized EHV.

Method	DF4	DF10	RE34-S1	Solar
Cat-E2E-P	0.1634	0.2234	0.4927	0.5027
Cat-MM-I	0.1683	0.2566	0.4927	**0.5863**
HN-LoRA-P	0.1913	0.3017	0.4927	0.5771
PSD	0.6968	**0.3755**	0.6270	0.5856
PPFM	**0.8571**	0.3310	**0.7077**	0.5078

We next evaluate whether the methods rely on returning a large candidate pool. The main experiments generate a larger candidate pool, while Tables 23–25 vary $K \in \{10, 50, 100\}$ and report held-out normalized EHV.

Table 24: Generation-budget robustness under $K = 50$. Each cell reports held-out normalized EHV.

Method	DF4	DF10	RE34-S1	Solar
Cat-E2E-P	0.3310	0.7901	0.4927	0.5787
Cat-MM-I	0.3301	**0.8171**	0.4927	**0.7121**
HN-LoRA-P	0.3693	0.7359	0.4927	0.6733
PSD	0.8066	0.7313	0.6986	0.6851
PPFM	**0.9295**	0.7182	**0.7648**	0.6795

The results show that PSD and PPFM do not rely on producing a large number of redundant samples. On DF4 and RE34-S1, both generative methods outperform the surrogate-only baselines at every budget, and the gap is largest at the smallest budget $K = 10$, where a truncated NSGA-II population covers little of the front. This indicates that the warm-started generative refinement concentrates candidates in high-quality trade-off regions rather than requiring a very large pool to compensate for poor ranking or excessive redundancy.

Increasing the generation budget from $K = 10$ to $K = 50$ gives the largest improvement for every method, and the surrogate-only baselines gain the most because their NSGA-II populations are truncated hardest at small K: on DF10 they rise from below 0.31 to above 0.73 and overtake the generative solvers from $K = 50$ onward. The gain from $K = 50$ to $K = 100$ is small on DF4 and RE34-S1, where the generative solvers are already close to their full-budget values, while PPFM still improves on RE34-S1 (from 0.765 to 0.782), which suggests that additional preferences help cover more trade-off regions in the three-objective engineering task.

The budget sweep also reveals limitations. On DF10, the surrogate baselines are the strongest methods once $K \geq 50$, so this task rewards a large, well-spread population rather than candidate quality and is not diagnostic of the generative solvers' advantage. On Solar, PPFM is sensitive to the generation

Table 25: Generation-budget robustness under $K = 100$. Each cell reports held-out normalized EHV.

Method	DF4	DF10	RE34-S1	Solar
Cat-E2E-P	0.3442	**0.9462**	0.4927	0.5744
Cat-MM-I	0.3405	0.9370	0.4927	0.7178
HN-LoRA-P	0.3842	0.8326	0.4927	0.6766
PSD	0.8335	0.8960	0.7159	**0.7249**
PPFM	**0.9484**	0.8849	**0.7822**	0.7157

budget: it is weak at $K = 10$ but becomes competitive at $K = 50$ and $K = 100$. This suggests that deterministic flow refinement may require more preference samples to cover irregular real-world conditional Pareto fronts. PSD is more stable on Solar, matches the best surrogate at $K = 10$, and obtains the best score at $K = 100$, consistent with the main results where diffusion-style denoising is stronger on REAL tasks.

Overall, the generation-budget sweep supports a cautious interpretation of the main results. The advantages of PSD and PPFM are not merely an artifact of returning a large candidate pool: their top candidates already provide strong Pareto coverage on DF4 and RE34-S1. At the same time, DF10 and Solar remain budget-sensitive, so large-pool results should be interpreted together with the smaller-budget diagnostics reported here.

I.3 Parameter-Sparsity Robustness

A central difficulty in offline PMOO is limited coverage of the task-parameter space. Unlike online PMOO, where the optimizer can request additional evaluations at poorly covered parameters, offline PMOO must reconstruct Pareto sets for held-out $\boldsymbol{\theta}$ values from a fixed archive. We therefore evaluate parameter-sparsity robustness by retaining only a fraction $\alpha_{\boldsymbol{\theta}} \in \{0.25, 0.50, 1.00\}$ of the training task parameters and retraining each method. The held-out evaluation parameters are kept unchanged. This experiment isolates whether a method can transfer across Θ rather than relying on a dense grid of observed task parameters.

Table 26: Parameter-sparsity robustness with 25% of training task parameters retained. Values are held-out normalized EHV means over the same five seeds and the same held-out parameter set at every ratio; the 100% row is the full-data model of the main experiments.

Method	DF4	DF10	RE34-S1	Solar
Cat-E2E-P	0.1818	0.5397	0.4927	0.3961
Cat-MM-I	0.2874	1.0457	0.4927	0.4754
HN-LoRA-P	0.2489	0.5802	0.4927	0.4323
PSD	0.8828	1.0537	0.7332	**0.7621**
PPFM	**0.9560**	**1.0538**	**0.8021**	0.7383

Tables 26–28 show that PSD and PPFM are substantially more robust than surrogate-only search under sparse parameter coverage. With only 25% of training task parameters retained, PPFM obtains the best value on DF4 and RE34-S1, PSD and PPFM tie on DF10, and PSD is best on Solar. This indicates that the learned elite-design prior and amortized refinement operator transfer useful structure across task parameters, instead of merely memorizing observed $\boldsymbol{\theta}$ values.

Table 27: Parameter-sparsity robustness with 50% of training task parameters retained. Values are held-out normalized EHV means over the same five seeds and the same held-out parameter set at every ratio; the 100% row is the full-data model of the main experiments.

Method	DF4	DF10	RE34-S1	Solar
Cat-E2E-P	0.2753	0.6845	0.4927	0.4452
Cat-MM-I	0.3296	1.0168	0.4927	0.6074
HN-LoRA-P	0.3000	0.7399	0.4927	0.5129
PSD	0.9153	**1.0276**	0.7329	**0.7599**
PPFM	**0.9659**	1.0236	**0.8010**	0.7583

The contrast with surrogate-only search is most pronounced on RE34-S1. Across all sparsity levels, the surrogate-only baselines remain at 0.493, whereas PSD stays near 0.73 and PPFM near 0.80.

This suggests that direct optimization of $\hat{F}_\psi(\cdot, \boldsymbol{\theta})$ can become unreliable when the training parameter grid is sparse: the optimizer may exploit artifacts of the learned surrogate in regions where the joint $(\boldsymbol{x}, \boldsymbol{\theta})$ support is weak. PSD and PPFM mitigate this failure mode by regularizing candidate designs toward an elite-design distribution learned from the archive, while still using surrogate guidance for preference-specific refinement.

Table 28: Parameter-sparsity robustness with 100% of training task parameters retained. Values are held-out normalized EHV means over the same five seeds and the same held-out parameter set at every ratio; the 100% row is the full-data model of the main experiments.

Method	DF4	DF10	RE34-S1	Solar
Cat-E2E-P	0.3442	0.9462	0.4927	0.5744
Cat-MM-I	0.3405	0.9370	0.4927	0.7178
HN-LoRA-P	0.3842	0.8326	0.4927	0.6766
PSD	0.8576	**1.0208**	0.7340	0.8056
PPFM	**0.9689**	1.0133	**0.7970**	**0.8617**

The results also reveal an important limitation. On Solar, performance improves with denser task-parameter coverage, especially for PPFM, which increases from 0.738 under 25% parameter coverage to 0.862 under full coverage. This indicates that generative refinement is not coverage-free: when the conditional Pareto geometry is irregular, the learned elite prior itself depends on having sufficiently representative elite samples across Θ. Thus, PSD and PPFM are more robust than surrogate-only search under sparse parameter coverage, but their advantage still relies on adequate elite coverage in the offline archive.

Overall, the parameter-sparsity sweep supports the main interpretation of the proposed methods. Offline PMOO is difficult because the optimizer must generalize across task parameters without additional oracle queries. Surrogate-only search repeatedly optimizes a learned model at each queried $\boldsymbol{\theta}$ and is therefore vulnerable to surrogate exploitation under weak parameter support. Warm-started generative refinement provides a stronger default by combining surrogate guidance with an amortized elite prior, while the Solar result shows that sufficiently broad elite coverage remains important for real-world conditional design tasks.

J Detailed Per-Suite Results

The following tables report per-problem results on training and held-out task parameters for the 16 methods of Table 2, grouped and ordered as there, and, on training parameters, for the archive-only diagnostic $\mathcal{D}_{\text{best}}$ (Appendix H.3). EHV is the mean over seeds of the per-run average normalized hypervolume (Section 3). Ranks are computed per task parameter on seed-averaged hypervolume, exactly as in Table 2, and reported as mean $\pm$ standard deviation over task parameters. The Avg columns give the mean $\pm$ standard deviation over the problems of a suite, so the held-out Avg values reproduce Table 2. At a training parameter, Nearest-Archive returns the nondominated designs of that parameter's own archive slice. Suffixes denote training variants: B = base MSE, G = GradNorm, P = PCGrad, and I = IOM. Best per column in bold, second best underlined.

Table 29: **Synthetic** suite, *training parameters*: average normalized EHV.

Method	DF1	DF2	DF3	DF4	DF5	DF6	DF7	DF8	DF9	DF10	DF11	DF12	DF13	DF14
Archive-only diagnostic														
$\mathcal{D}_{\text{best}}$	0.290	0.626	0.029	0.302	0.167	0.008	0.494	0.403	0.081	0.693	0.087	0.278	0.214	0.251
Oracle-free controls														
Nearest-Archive	0.291	0.627	0.029	0.302	0.167	0.008	0.494	0.403	0.081	0.814	0.095	0.291	0.245	0.256
Warm-Start-Only	0.528	0.754	0.198	0.527	0.444	0.023	0.782	0.596	0.299	0.822	0.179	0.550	0.536	0.537
PPSL-MOBO	**0.555**	0.625	0.161	0.749	0.483	0.000	0.652	0.663	0.319	0.000	0.062	0.021	0.061	0.042
Task-conditioned surrogate solvers														
Cat-E2E-B	0.550	0.694	0.115	0.259	0.490	0.047	0.080	0.650	0.094	0.335	0.054	0.143	0.148	0.233
Cat-E2E-G	0.550	0.688	0.110	0.254	0.498	0.041	0.080	0.650	0.090	0.640	0.218	0.778	0.736	0.733
Cat-E2E-P	0.551	0.692	0.125	0.254	0.490	0.043	0.080	0.653	0.099	0.729	0.217	**0.809**	0.713	0.729
Cat-MM-B	0.553	0.701	0.149	0.242	0.518	0.078	0.080	0.658	0.071	0.036	0.056	0.155	0.199	0.263
Cat-MM-I	0.520	0.661	0.050	0.272	0.532	0.014	0.061	0.653	0.104	0.962	0.203	0.678	0.502	0.450
HN-E2E-B	0.535	0.797	0.151	0.227	**0.595**	0.125	0.080	0.637	0.122	0.513	0.078	0.323	0.402	0.390
HN-E2E-G	0.538	0.802	0.151	0.213	0.589	0.119	0.084	0.603	0.132	0.932	0.218	0.697	**0.763**	0.877
HN-E2E-P	0.522	0.790	0.146	0.222	0.591	**0.135**	0.082	0.648	0.133	0.958	0.219	0.736	0.761	**0.881**
HN-LoRA-B	0.544	0.795	0.142	0.332	0.542	0.104	0.085	0.668	0.104	0.462	0.039	0.219	0.226	0.325
HN-LoRA-G	0.493	**0.803**	0.111	0.296	0.545	0.094	0.084	0.665	0.105	0.709	**0.229**	0.633	0.712	0.804
HN-LoRA-P	0.546	0.788	0.142	0.322	0.538	0.093	0.088	**0.671**	0.101	0.752	0.222	0.680	0.709	0.783
Generative refinement (ours)														
PSD	0.535	0.780	0.226	0.647	0.485	0.017	**0.816**	0.639	0.351	0.941	0.185	0.584	0.569	0.580
PPFM	0.534	0.791	**0.304**	**0.865**	0.563	0.001	0.811	0.646	**0.443**	**0.972**	0.183	0.652	0.538	0.671

Table 30: **Synthetic** suite, *training parameters*: average rank by normalized EHV.

Method	DF1	DF2	DF3	DF4	DF5	DF6	DF7	DF8	DF9	DF10	DF11	DF12	DF13	DF14	Avg
Archive-only diagnostic															
$\mathcal{D}_{\text{best}}$	15.7±1.9	12.7±3.9	13.9±2.8	10.2±4.6	15.2±2.0	9.8±2.0	5.5±1.5	15.0±3.5	11.6±4.1	10.0±2.7	12.6±3.1	13.4±2.0	13.9±2.0	13.2±3.2	12.3±2.6
Oracle-free controls															
Nearest-Archive	14.8±3.7	11.4±5.3	13.9±2.8	10.2±4.6	15.1±2.1	9.8±2.0	5.5±1.5	14.9±3.6	11.6±4.1	7.4±3.0	11.7±3.8	12.9±2.4	13.0±2.9	12.6±4.1	11.8±2.7
Warm-Start-Only	8.1±4.8	9.1±3.7	6.1±3.7	6.5±4.2	9.8±4.8	9.0±2.8	2.8±0.9	10.2±4.7	5.6±3.8	7.4±3.4	8.0±2.9	9.5±2.9	8.7±3.0	9.4±3.1	7.9±2.0
PPSL-MOBO	**3.8±3.1**	12.7±4.4	8.2±5.2	3.7±4.2	9.0±4.2	10.2±1.8	3.9±2.8	**6.2±3.7**	5.3±4.5	16.2±0.6	12.3±2.4	15.5±1.3	15.6±1.4	15.1±1.8	9.8±4.6
Task-conditioned surrogate solvers															
Cat-E2E-B	6.9±3.2	11.5±3.2	11.0±3.0	10.8±3.0	9.0±4.3	9.2±2.2	12.3±2.9	7.9±4.4	10.7±2.7	13.5±2.8	13.0±3.1	14.2±2.0	14.2±2.4	13.2±2.9	11.3±2.2
Cat-E2E-G	7.4±3.2	11.9±3.2	11.5±2.9	10.6±3.2	8.9±4.2	9.3±1.4	12.0±2.4	8.7±4.5	11.2±2.5	10.3±3.2	5.8±3.1	2.8±2.1	3.9±2.1	6.5±3.1	8.6±2.9
Cat-E2E-P	6.6±3.2	11.6±3.2	10.0±3.4	10.7±3.5	9.5±4.5	9.3±2.1	12.1±3.0	7.7±4.3	10.8±2.6	8.6±3.4	6.0±2.8	**1.9±1.5**	4.7±3.2	6.7±4.2	8.3±2.8
Cat-MM-B	5.6±3.5	11.5±3.5	8.7±3.9	10.5±3.3	6.7±3.7	8.9±2.5	11.1±3.0	7.3±3.7	12.0±2.6	15.9±0.6	12.6±2.7	13.8±2.4	13.8±2.2	12.6±3.1	10.8±2.9
Cat-MM-I	12.9±3.1	13.1±4.2	13.5±3.6	10.8±2.9	10.2±5.2	9.8±2.0	13.8±3.8	8.4±4.6	10.4±3.6	4.4±3.5	7.5±2.1	6.3±2.4	9.3±2.5	10.2±2.8	10.0±2.7
HN-E2E-B	9.9±3.3	6.5±3.8	7.9±3.8	12.5±3.5	**6.2±3.6**	7.6±3.0	12.5±2.9	10.3±4.2	9.7±4.0	12.0±3.3	11.6±2.9	11.6±3.2	10.6±2.8	11.0±3.2	10.0±2.1
HN-E2E-G	9.4±3.6	5.6±3.3	7.9±4.1	12.5±3.3	6.6±3.6	8.0±3.1	11.9±3.0	12.1±3.6	8.7±3.7	4.7±2.9	6.5±2.7	5.4±2.5	**3.1±2.2**	**3.4±2.2**	7.6±3.0
HN-E2E-P	10.8±3.9	6.8±3.9	8.3±4.3	12.5±3.2	6.3±4.3	**7.2±3.3**	11.9±2.7	9.3±4.2	9.2±3.8	4.5±3.2	6.0±2.8	4.1±2.1	3.5±2.1	3.5±2.4	7.4±2.9
HN-LoRA-B	5.3±3.9	5.6±3.8	7.8±3.4	7.9±3.4	7.7±4.2	8.4±2.7	11.1±2.7	6.4±4.0	10.0±2.9	12.8±2.7	13.2±3.1	13.1±3.5	12.8±3.3	12.0±3.5	9.6±2.8
HN-LoRA-G	11.2±4.8	**4.4±3.8**	10.9±3.5	8.7±3.3	7.2±4.5	8.5±2.8	11.3±2.7	6.7±4.0	10.3±2.9	9.4±3.1	**5.1±3.1**	7.3±2.8	4.8±2.0	4.4±2.4	7.9±2.5
HN-LoRA-P	5.2±4.1	5.6±4.4	7.8±3.2	8.4±3.6	7.8±4.3	8.5±2.5	10.9±3.0	**6.2±3.8**	10.4±3.0	8.3±3.4	5.9±2.8	6.0±2.5	4.7±2.4	5.1±3.0	7.2±1.9
Generative refinement (ours)															
PSD	9.3±3.5	6.8±3.7	4.0±2.7	4.3±3.9	9.0±4.8	9.4±2.2	**1.8±0.8**	7.2±5.0	3.9±2.8	3.9±2.8	7.3±3.3	8.4±2.8	7.7±3.1	8.2±3.5	6.5±2.3
PPFM	9.9±3.8	6.2±3.5	**1.4±1.1**	**2.2±1.4**	8.8±4.7	10.2±2.0	2.4±1.0	8.5±3.6	**1.6±1.0**	**3.8±2.0**	8.0±2.5	7.0±2.3	8.6±2.4	5.9±3.7	**6.0±3.1**

Table 31: **Synthetic** suite, *held-out parameters*: average normalized EHV.

Method	DF1	DF2	DF3	DF4	DF5	DF6	DF7	DF8	DF9	DF10	DF11	DF12	DF13	DF14
Oracle-free controls														
Nearest-Archive	0.288	0.598	0.019	0.329	0.167	0.003	0.489	0.335	0.076	0.847	0.111	0.330	0.233	0.269
Warm-Start-Only	0.526	0.724	0.197	0.552	0.461	0.010	0.790	0.540	0.318	0.849	0.191	0.594	0.540	0.548
PPSL-MOBO	**0.551**	0.611	0.163	0.763	0.450	0.000	0.656	0.611	0.360	0.000	0.077	0.017	0.071	0.039
Task-conditioned surrogate solvers														
Cat-E2E-B	0.545	0.688	0.100	0.257	0.459	0.012	0.084	0.588	0.066	0.332	0.055	0.158	0.153	0.226
Cat-E2E-G	0.546	0.684	0.090	0.240	0.470	0.023	0.084	0.591	0.062	0.645	0.237	0.811	0.750	0.729
Cat-E2E-P	0.547	0.692	0.104	0.242	0.454	0.022	0.084	0.592	0.069	0.754	0.235	**0.842**	0.732	0.725
Cat-MM-B	0.548	0.692	0.136	0.244	0.494	0.044	0.084	0.608	0.052	0.046	0.070	0.164	0.199	0.203
Cat-MM-I	0.514	0.644	0.038	0.253	0.519	0.009	0.062	0.598	0.063	0.977	0.219	0.719	0.500	0.465
HN-E2E-B	0.524	0.744	0.128	0.198	0.574	0.088	0.084	0.583	0.088	0.536	0.082	0.358	0.404	0.389
HN-E2E-G	0.532	0.757	0.127	0.187	0.565	0.071	0.086	0.549	0.087	0.948	0.236	0.738	0.786	**0.888**
HN-E2E-P	0.510	0.747	0.126	0.199	0.562	**0.096**	0.085	0.599	0.094	0.970	0.237	0.776	**0.788**	0.885
HN-LoRA-B	0.526	0.736	0.124	0.318	0.521	0.043	0.088	0.618	0.079	0.483	0.049	0.249	0.231	0.261
HN-LoRA-G	0.461	0.753	0.097	0.292	0.524	0.041	0.087	0.612	0.076	0.739	**0.241**	0.697	0.724	0.781
HN-LoRA-P	0.522	0.733	0.124	0.318	0.518	0.048	0.088	**0.619**	0.074	0.788	0.236	0.707	0.728	0.774
Generative refinement (ours)														
PSD	0.535	0.757	0.223	0.665	0.511	0.007	**0.826**	0.585	0.371	0.979	0.202	0.646	0.573	0.598
PPFM	0.532	**0.792**	**0.298**	**0.872**	**0.575**	0.001	0.811	0.591	**0.455**	**1.005**	0.201	0.693	0.546	0.685

Table 32: **Synthetic** suite, *held-out parameters*: average rank by normalized EHV.

Method	DF1	DF2	DF3	DF4	DF5	DF6	DF7	DF8	DF9	DF10	DF11	DF12	DF13	DF14	Avg
Oracle-free controls															
Nearest-Archive	15.8±0.4	13.1±3.1	15.8±0.5	9.5±4.7	16.0±0.1	9.7±2.4	4.8±0.4	16.0±0.0	9.9±5.3	6.8±1.3	11.3±2.3	12.0±1.3	13.3±1.9	12.8±2.2	11.9±3.4
Warm-Start-Only	9.4±3.3	8.3±3.8	4.2±2.0	5.7±3.1	9.2±4.5	8.3±3.0	2.9±0.5	13.7±1.7	4.5±1.3	6.8±1.3	8.1±3.0	9.9±1.7	8.5±1.4	8.4±2.3	7.7±2.6
PPSL-MOBO	**3.2±2.8**	13.6±3.3	5.9±4.2	2.2±0.7	10.8±3.4	10.4±2.2	4.2±0.5	5.1±2.7	3.7±3.2	15.8±0.2	11.6±2.2	15.7±0.4	15.7±0.5	15.8±0.4	9.5±5.1
Task-conditioned surrogate solvers															
Cat-E2E-B	6.1±1.5	10.3±3.4	11.4±2.0	10.0±2.4	10.5±3.6	9.6±1.1	10.3±3.3	8.8±4.1	10.6±1.9	14.0±0.1	12.9±2.9	14.1±1.3	14.4±1.1	12.8±2.0	11.1±2.2
Cat-E2E-G	5.6±1.9	10.8±2.9	13.0±1.5	10.9±2.3	8.4±4.4	9.4±0.8	10.5±2.7	8.9±3.0	11.1±2.3	10.7±0.9	4.6±3.0	2.2±0.7	3.4±1.5	6.3±2.8	8.3±3.2
Cat-E2E-P	4.4±1.9	10.3±3.7	10.7±1.7	10.5±2.5	10.6±3.8	9.5±1.0	10.4±2.9	8.1±3.8	10.5±2.1	8.9±1.4	5.2±2.7	**1.2±0.7**	4.0±2.6	6.3±3.2	7.9±3.0
Cat-MM-B	3.9±1.9	11.0±3.4	7.5±3.8	10.3±2.4	6.8±3.4	8.8±1.8	10.2±3.4	6.8±3.7	12.1±1.7	15.2±0.2	12.2±2.5	13.9±1.8	13.4±0.8	13.1±2.3	10.4±3.1
Cat-MM-I	12.1±1.9	12.4±3.6	15.0±0.7	11.0±2.7	7.6±4.9	9.9±1.6	13.9±4.0	7.8±3.7	11.1±2.3	3.3±1.7	7.5±1.0	5.8±1.7	8.9±1.9	9.3±2.7	9.7±3.1
HN-E2E-B	10.4±2.6	6.7±3.0	7.5±2.9	13.3±3.9	**4.4±2.9**	4.4±3.1	12.3±2.8	9.8±3.4	9.5±3.0	12.1±0.5	11.2±2.2	11.5±1.1	10.8±0.9	10.7±1.4	9.6±2.7
HN-E2E-G	9.2±1.5	4.8±2.9	7.4±3.0	14.0±2.4	5.9±3.5	6.2±3.4	11.5±1.9	13.5±1.8	9.9±3.1	3.8±1.2	6.0±2.7	5.3±1.8	2.3±1.3	**2.1±1.3**	7.3±3.7
HN-E2E-P	12.5±1.4	6.8±2.2	7.7±2.3	13.1±2.9	5.4±3.2	**3.7±3.1**	11.8±2.7	7.4±3.6	9.0±3.5	3.3±1.7	5.8±2.4	3.0±0.7	**2.1±0.9**	2.2±1.2	6.7±3.6
HN-LoRA-B	6.1±5.1	7.0±3.7	7.5±1.8	6.4±1.5	8.8±3.3	8.8±1.9	9.1±2.0	3.6±2.6	9.8±2.4	12.8±0.5	13.4±3.4	13.1±0.9	13.2±0.6	13.1±1.4	9.5±3.1
HN-LoRA-G	13.3±3.7	4.9±3.7	11.3±1.5	8.3±2.2	8.5±3.7	8.7±2.1	10.6±2.6	4.9±3.2	10.3±1.7	9.4±1.3	**4.2±2.9**	7.1±1.6	5.6±1.7	5.5±1.8	8.0±2.7
HN-LoRA-P	6.5±5.4	7.3±4.4	7.8±2.2	6.5±1.9	9.1±3.2	8.8±2.0	10.3±3.1	**3.3±2.2**	10.4±1.9	8.2±1.3	6.0±1.9	6.4±1.8	5.5±1.7	5.7±2.6	7.3±1.9
Generative refinement (ours)															
PSD	8.2±3.3	5.2±4.0	2.5±1.3	3.3±1.6	7.6±4.9	9.5±1.9	**1.2±0.4**	9.5±3.5	2.7±0.6	2.9±1.4	7.3±3.0	8.3±1.9	7.0±1.8	6.8±2.4	5.8±2.7
PPFM	9.2±3.4	**3.5±2.9**	**1.0±0.3**	**1.1±0.2**	6.2±4.8	10.3±2.0	2.0±0.6	8.7±2.9	**1.0±0.0**	**2.1±1.3**	8.7±1.3	6.8±1.8	7.8±1.6	5.0±3.3	**5.3±3.3**

Table 33: **RE w/ SC** and **REAL** suites, *training parameters*: average normalized EHV.

	RE w/ SC									REAL			
	RE34			RE41			RE91						
Method	S1	S12	S123	S1	S12	S123	S1	S12	S123	Turbine	Solar	Lamp21	UAV32
Archive-only diagnostic													
$\mathcal{D}_{\text{best}}$	0.632	0.553	0.401	0.573	0.428	0.367	**0.088**	0.064	**0.056**	1.006	0.540	0.571	0.833
Oracle-free controls													
Nearest-Archive	0.633	0.555	0.408	**0.602**	**0.441**	**0.374**	**0.088**	**0.065**	**0.056**	1.006	0.540	0.572	0.833
Warm-Start-Only	0.734	0.596	0.344	0.560	**0.441**	0.361	0.056	0.034	0.030	1.134	0.690	0.818	1.127
PPSL-MOBO	0.739	0.545	0.334	0.016	0.044	0.054	0.002	0.011	0.012	1.151	0.470	0.757	1.018
Task-conditioned surrogate solvers													
Cat-E2E-B	0.494	0.545	0.334	0.222	0.182	0.159	0.020	0.021	0.019	**1.165**	0.577	0.610	1.120
Cat-E2E-G	0.494	0.545	0.334	0.225	0.184	0.162	0.021	0.021	0.019	**1.165**	0.563	0.863	1.134
Cat-E2E-P	0.494	0.545	0.334	0.225	0.184	0.162	0.021	0.021	0.019	**1.165**	0.574	0.863	1.125
Cat-MM-B	0.494	0.545	0.334	0.185	0.155	0.128	0.021	0.016	0.015	**1.165**	0.581	0.516	1.115
Cat-MM-I	0.494	0.545	0.334	0.224	0.184	0.162	0.021	0.020	0.019	1.164	0.749	0.842	1.033
HN-E2E-B	0.494	0.545	0.334	0.206	0.168	0.143	0.022	0.020	0.018	1.158	0.698	0.713	0.567
HN-E2E-G	0.494	0.545	0.334	0.226	0.185	0.162	0.021	0.021	0.019	**1.165**	0.713	0.891	0.582
HN-E2E-P	0.494	0.545	0.334	0.225	0.184	0.161	0.021	0.021	0.019	1.160	0.695	**0.893**	0.549
HN-LoRA-B	0.494	0.545	0.334	0.175	0.165	0.154	0.021	0.021	0.019	1.152	0.698	0.516	1.046
HN-LoRA-G	0.494	0.545	0.334	0.224	0.184	0.161	0.021	0.021	0.019	1.162	0.675	0.853	1.049
HN-LoRA-P	0.494	0.545	0.334	0.224	0.183	0.161	0.021	0.021	0.018	1.156	0.673	0.853	1.049
Generative refinement (ours)													
PSD	0.743	0.609	0.371	0.556	0.433	0.363	0.053	0.033	0.030	1.127	0.767	0.812	**1.144**
PPFM	**0.802**	**0.639**	**0.416**	0.573	0.438	0.367	0.072	0.051	0.045	1.146	**0.808**	0.767	1.081

Table 34: **RE w/ SC** and **REAL** suites, *training parameters*: average rank by normalized EHV.

	RE w/ SC										REAL				
	RE34			RE41			RE91								
Method	S1	S12	S123	S1	S12	S123	S1	S12	S123	Avg	Turbine	Solar	Lamp21	UAV32	Avg
Archive-only diagnostic															
$\mathcal{D}_{\text{best}}$	7.4±4.3	8.1±6.1	4.2±4.9	3.1±1.2	3.9±1.1	3.2±1.2	**1.5±0.5**	1.6±0.4	**1.5±0.3**	3.8±2.3	16.3±0.8	13.4±4.1	14.3±1.8	13.6±1.3	14.4±1.2
Oracle-free controls															
Nearest-Archive	7.2±4.5	7.7±6.4	3.4±5.2	**1.6±1.1**	**2.3±1.3**	**1.8±1.2**	**1.5±0.5**	**1.5±0.4**	**1.5±0.4**	3.2±2.3	16.3±0.8	13.4±4.1	14.0±2.0	13.6±1.3	14.3±1.2
Warm-Start-Only	3.1±1.3	3.9±2.0	8.0±3.2	3.5±1.1	2.4±1.2	3.6±1.3	4.2±0.6	4.5±0.6	4.7±0.9	4.2±1.5	13.2±1.6	7.9±3.4	8.0±2.7	5.2±2.7	8.6±2.9
PPSL-MOBO	3.6±1.3	9.4±3.4	11.1±3.5	17.0±0.0	17.0±0.0	17.0±0.0	17.0±0.0	16.6±1.2	15.4±3.2	13.8±4.5	12.1±1.6	14.6±3.4	9.7±4.6	9.8±2.9	11.6±2.0
Task-conditioned surrogate solvers															
Cat-E2E-B	11.6±0.8	10.9±1.0	10.9±0.8	12.3±1.2	12.0±1.8	11.1±2.7	11.6±3.5	10.6±2.8	10.5±3.1	11.3±0.6	6.8±2.5	11.9±2.8	13.1±3.1	4.9±2.9	9.2±3.4
Cat-E2E-G	11.6±0.8	10.9±1.0	10.9±0.8	9.0±2.0	8.7±2.1	8.3±2.1	10.7±3.1	10.8±3.0	11.2±3.0	10.2±1.2	6.8±2.5	12.5±2.9	4.8±2.0	3.8±2.5	7.0±3.3
Cat-E2E-P	11.6±0.8	10.9±1.0	10.9±0.8	9.2±2.2	8.8±2.1	8.6±2.1	11.0±3.0	10.2±2.9	11.4±3.0	10.3±1.1	6.5±2.7	12.1±2.7	4.9±2.1	4.4±2.7	7.0±3.1
Cat-MM-B	11.6±0.8	10.9±1.0	10.9±0.8	15.2±0.5	15.6±0.6	15.6±0.5	11.0±3.1	15.3±2.4	13.9±3.6	13.3±2.0	5.4±2.9	11.6±3.2	15.4±1.2	5.6±2.9	9.5±4.2
Cat-MM-I	11.6±0.8	10.9±1.0	10.9±0.8	10.3±2.1	10.2±2.1	9.7±2.4	11.4±3.0	10.9±3.2	10.9±3.1	10.8±0.6	6.8±3.4	4.7±3.0	6.6±2.3	9.3±2.5	6.9±1.7
HN-E2E-B	11.6±0.8	10.9±1.0	10.9±0.8	13.8±1.3	13.8±1.9	14.6±1.6	8.8±2.7	11.8±3.2	11.6±3.2	12.0±1.7	6.5±3.8	7.3±4.3	11.4±3.9	15.7±1.3	10.2±3.6
HN-E2E-G	11.6±0.8	10.9±1.0	10.9±0.8	7.8±2.0	7.4±1.7	8.0±2.3	11.3±3.2	10.3±3.0	10.4±3.1	9.9±1.6	5.7±3.5	6.3±3.5	2.3±1.6	15.4±1.4	7.4±4.8
HN-E2E-P	11.6±0.8	10.9±1.0	10.9±0.8	8.8±2.2	9.0±2.0	10.9±2.1	10.9±3.1	9.8±2.7	10.5±3.1	10.4±0.9	6.0±3.8	7.1±3.9	**2.2±1.4**	15.8±1.3	7.8±5.0
HN-LoRA-B	11.6±0.8	10.9±1.0	10.9±0.8	15.5±0.7	14.8±0.8	12.9±2.0	11.4±2.9	10.2±2.9	9.8±3.1	12.0±1.9	6.6±4.6	6.8±3.7	14.7±2.7	8.2±3.6	9.1±3.3
HN-LoRA-G	11.6±0.8	10.9±1.0	10.9±0.8	9.2±2.2	9.4±1.9	10.3±2.2	11.3±3.1	10.6±2.8	10.7±2.9	10.6±0.8	**5.1±3.1**	8.2±3.7	5.9±2.2	8.1±3.5	**6.8±1.3**
HN-LoRA-P	11.6±0.8	10.9±1.0	10.9±0.8	9.8±2.1	11.3±1.8	10.9±2.1	11.5±3.1	10.9±2.8	11.5±3.1	11.1±0.5	5.9±4.3	7.9±3.3	5.8±2.1	8.6±3.1	7.1±1.2
Generative refinement (ours)															
PSD	2.9±0.7	2.6±0.9	3.7±0.7	3.8±1.0	3.5±1.1	3.3±1.2	4.7±0.5	4.6±0.6	4.5±0.8	3.7±0.7	14.2±1.0	4.4±3.4	8.9±2.1	**3.1±2.1**	7.7±4.3
PPFM	**1.0±0.0**	**1.2±0.5**	**2.3±0.9**	3.0±1.5	2.9±1.6	3.1±1.5	3.0±0.4	2.9±0.5	2.9±0.4	**2.5±0.8**	12.6±2.1	**2.7±2.0**	11.0±1.4	7.8±2.2	8.5±3.8

Table 35: **RE w/ SC** and **REAL** suites, *held-out parameters*: average normalized EHV.

	RE w/ SC									REAL			
	RE34			RE41			RE91						
Method	S1	S12	S123	S1	S12	S123	S1	S12	S123	Turbine	Solar	Lamp21	UAV32
Oracle-free controls													
Nearest-Archive	0.607	0.519	0.381	**0.605**	0.436	**0.380**	**0.077**	**0.053**	**0.049**	1.053	0.572	0.612	0.884
Warm-Start-Only	0.725	0.572	0.332	0.558	**0.441**	0.371	0.055	0.033	0.031	1.154	0.704	0.829	1.132
PPSL-MOBO	0.739	0.522	0.324	0.016	0.043	0.053	0.002	0.011	0.013	1.169	0.483	0.767	1.039
Task-conditioned surrogate solvers													
Cat-E2E-B	0.493	0.522	0.324	0.224	0.180	0.161	0.020	0.020	0.019	**1.179**	0.593	0.615	1.130
Cat-E2E-G	0.493	0.522	0.324	0.227	0.183	0.164	0.021	0.020	0.019	**1.179**	0.581	0.868	1.143
Cat-E2E-P	0.493	0.522	0.324	0.227	0.183	0.164	0.020	0.020	0.019	**1.179**	0.592	0.866	1.132
Cat-MM-B	0.493	0.522	0.324	0.187	0.154	0.129	0.021	0.015	0.015	**1.179**	0.589	0.508	1.126
Cat-MM-I	0.493	0.522	0.324	0.226	0.182	0.164	0.020	0.020	0.019	1.177	0.743	0.843	1.060
HN-E2E-B	0.493	0.522	0.324	0.209	0.168	0.143	0.022	0.019	0.019	1.171	0.714	0.713	0.649
HN-E2E-G	0.493	0.522	0.324	0.229	0.184	0.164	0.021	0.020	0.020	1.178	0.726	0.890	0.671
HN-E2E-P	0.493	0.522	0.324	0.228	0.183	0.163	0.021	0.020	0.019	1.174	0.708	**0.892**	0.631
HN-LoRA-B	0.493	0.522	0.324	0.177	0.164	0.157	0.020	0.020	0.020	1.162	0.704	0.523	1.064
HN-LoRA-G	0.493	0.522	0.324	0.227	0.183	0.164	0.020	0.020	0.019	1.177	0.691	0.854	1.066
HN-LoRA-P	0.493	0.522	0.324	0.227	0.181	0.163	0.020	0.020	0.019	1.167	0.683	0.855	1.067
Generative refinement (ours)													
PSD	0.738	0.584	0.358	0.550	0.434	0.373	0.053	0.033	0.031	1.151	0.774	0.824	**1.146**
PPFM	**0.797**	**0.613**	**0.400**	0.571	0.433	0.374	0.070	0.049	0.046	1.165	**0.812**	0.786	1.094

Table 36: **RE w/ SC** and **REAL** suites, *held-out parameters*: average rank by normalized EHV.

	RE w/ SC										REAL				
	RE34			RE41			RE91								
Method	S1	S12	S123	S1	S12	S123	S1	S12	S123	Avg	Turbine	Solar	Lamp21	UAV32	Avg
Oracle-free controls															
Nearest-Archive	5.4±2.2	10.4±6.0	4.0±5.3	**1.1±0.3**	2.5±1.1	**1.7±0.9**	**1.0±0.0**	**1.0±0.2**	**1.1±0.2**	3.1±3.0	16.0±0.0	13.3±2.1	13.5±1.8	12.9±0.3	13.9±1.2
Warm-Start-Only	3.6±0.6	3.0±0.2	4.8±2.4	2.8±0.6	**1.7±0.8**	3.1±0.9	3.2±0.4	3.6±0.5	3.6±0.5	3.3±0.8	13.7±1.2	7.0±2.1	8.2±1.3	4.4±1.6	8.3±3.4
PPSL-MOBO	2.7±0.9	4.5±0.5	9.6±4.3	16.0±0.0	16.0±0.0	16.0±0.0	16.0±0.0	16.0±0.0	16.0±0.2	12.5±5.2	10.4±2.0	15.9±0.3	10.8±2.7	10.4±1.7	11.9±2.3
Task-conditioned surrogate solvers															
Cat-E2E-B	11.0±0.2	10.5±0.5	10.3±0.6	12.0±0.1	11.9±0.3	11.4±1.2	12.1±2.6	7.9±2.5	8.8±2.6	10.6±1.4	4.8±1.6	12.5±1.0	13.2±1.5	4.4±2.2	8.7±4.1
Cat-E2E-G	11.0±0.2	10.5±0.5	10.3±0.6	8.0±1.4	7.4±1.3	6.7±1.3	8.6±2.5	9.2±2.7	9.1±2.7	9.0±1.4	5.2±2.1	13.6±1.3	3.8±1.2	2.5±1.4	**6.3±4.3**
Cat-E2E-P	11.0±0.2	10.5±0.5	10.3±0.6	7.6±1.4	7.6±1.5	7.3±1.6	10.8±2.7	9.1±2.6	9.5±2.9	9.3±1.4	4.8±1.8	12.6±1.2	4.0±1.1	4.0±2.0	**6.3±3.6**
Cat-MM-B	11.0±0.2	10.5±0.5	10.3±0.6	14.0±0.1	15.0±0.0	14.9±0.3	9.3±2.6	15.0±0.1	14.8±1.1	12.7±2.3	**3.3±2.6**	13.0±1.5	15.1±0.7	4.8±1.7	9.0±5.1
Cat-MM-I	11.0±0.2	10.5±0.5	10.3±0.6	10.3±0.9	9.7±1.0	8.0±1.8	11.5±2.8	10.9±2.8	9.8±2.8	10.2±0.9	6.1±2.2	4.3±2.5	6.8±1.2	9.6±1.6	6.7±1.9
HN-E2E-B	11.0±0.2	10.5±0.5	10.3±0.6	13.0±0.0	13.1±0.3	14.1±0.3	5.4±0.8	11.7±2.6	12.1±2.4	11.2±2.4	9.0±3.3	5.9±1.7	11.5±2.1	15.1±0.8	10.4±3.4
HN-E2E-G	11.0±0.2	10.5±0.5	10.3±0.6	5.1±0.5	5.3±0.7	5.6±0.9	9.7±2.7	9.6±2.5	8.6±2.9	8.4±2.3	5.2±2.9	4.9±1.5	1.6±0.5	14.5±0.6	6.5±4.8
HN-E2E-P	11.0±0.2	10.5±0.5	10.3±0.6	7.0±1.3	7.3±1.2	10.0±1.2	9.8±3.0	8.3±2.7	9.7±2.6	9.3±1.4	7.5±3.6	6.4±1.9	**1.4±0.6**	15.4±0.8	7.7±5.0
HN-LoRA-B	11.0±0.2	10.5±0.5	10.3±0.6	15.0±0.1	13.9±0.3	13.0±0.1	10.8±2.8	8.7±2.6	8.1±2.6	11.2±2.2	10.6±4.9	6.3±1.8	14.7±1.5	9.2±2.3	10.2±3.0
HN-LoRA-G	11.0±0.2	10.5±0.5	10.3±0.6	8.6±1.5	8.3±1.6	8.8±1.5	11.6±2.7	9.2±2.6	9.0±2.7	9.7±1.1	4.9±3.0	7.8±2.4	5.7±1.3	9.4±2.0	6.9±1.8
HN-LoRA-P	11.0±0.2	10.5±0.5	10.3±0.6	9.4±1.4	10.5±1.3	10.2±1.3	10.4±2.8	10.6±2.8	10.6±2.6	10.4±0.4	9.0±3.8	9.1±1.0	5.4±1.2	9.2±2.1	8.2±1.6
Generative refinement (ours)															
PSD	2.7±0.6	2.0±0.2	2.7±0.5	3.5±0.6	2.9±0.9	2.7±1.0	3.8±0.4	3.4±0.5	3.4±0.5	3.0±0.5	14.2±0.7	2.3±1.0	9.2±0.9	**2.1±1.3**	7.0±5.1
PPFM	**1.0±0.0**	**1.0±0.1**	**1.4±0.5**	2.5±1.0	2.8±1.2	2.5±1.2	2.0±0.0	2.0±0.2	1.9±0.2	**1.9±0.6**	11.5±1.6	**1.1±0.3**	11.1±0.9	8.0±1.6	7.9±4.2

K LICENSES FOR USED ASSETS

Table 37 lists the used assets and their licenses. Our code is licensed under the MIT License.

Table 37: List of sources and licenses for the datasets and problem suites used in this work.

Resource	Type	Link	License
DF suite (Jiang et al., 2018)	Problem Suite / Dataset Source	https://chang88ye.github.io/homepage/publication/2018-01-01-Benchmark-Functions-for-the-CEC2018-Competition-on-Dynamic-Multiobjective-Optimization	None specified
RE problems (Tanabe & Ishibuchi, 2020)	Problem Suite / Dataset Source	https://ryojitanabe.github.io/reproblems/	None specified
Pareto Gamuts real problems (Makatura et al., 2021)	Code / Problem Suite	https://github.com/LianeMakatura/ParetoGamuts	MIT License
Offline-PMOO code/datasets (Ours)	Code/Dataset	https://anonymous.4open.science/r/offline-pmoo	MIT License