AIMS Mathematics, 11(7): 22166–22204.
DOI:10.3934/math.2026898
Received: 14 April 2026
Revised: 30 June 2026
Accepted: 07 July 2026
Published: 24 July 2026

https://www.aimspress.com/journal/Math

Research article

Pareto-based selection strategies for batch Bayesian optimization of expensive black-box functions

Kittisak Chaiyotha[1] **and Tipaluck Krityakierne**[1,2,*]

[1] Department of Mathematics, Faculty of Science, Mahidol University, Bangkok 10400, Thailand
[2] Centre of Excellence in Mathematics, MHESI, Bangkok 10400, Thailand

* **Correspondence:** Email: tipaluck.kri@mahidol.ac.th.

Abstract: Optimizing computationally expensive black-box functions is challenging because each function evaluation incurs a high computational cost. Bayesian optimization (BO) addresses this problem by using Gaussian process surrogate models to guide the selection of new evaluation points. Within the Bayesian optimization with Pareto-based selection (BOPS) framework, the predictive mean and standard deviation define a bi-objective optimization problem that balances exploitation and exploration. A batch of solutions is then selected from its approximate Pareto set. This paper presents a systematic empirical study of Pareto front approximation and Pareto-based solution selection within the BOPS framework. We investigated three space-filling designs (Sobol sequences, Latin hypercube sampling, and uniform random sampling) in combination with five Pareto-based solution selection strategies. Experiments were conducted on the 20-dimensional CEC2022 benchmark suite (12 functions) and a real-world robotic manipulation problem using a batch size of $q = 4$, an evaluation budget of 800 function evaluations, and 30 independent runs. BOPS variants were compared with standard acquisition-based batch BO methods, including q-expected improvement, q-upper confidence bound, and q-knowledge gradient. The results showed that optimization performance depends primarily on the solution selection mechanism rather than the Pareto approximation strategy once a sufficiently accurate Pareto set is obtained. Moreover, several BOPS variants using simple space-filling-based Pareto approximation achieve optimization performance comparable to or better than standard acquisition-based BO. These findings provide empirical insights into Pareto-based batch Bayesian optimization, demonstrate the effectiveness of simple space-filling-based Pareto approximation, and provide a unified framework for developing and evaluating future Pareto-based solution selection strategies.

Keywords: Bayesian optimization; bi-objective; Gaussian process; Pareto solution selection criteria; Pareto front approximation
Mathematics Subject Classification: 90C26, 90C59

1. Introduction

Optimization problems frequently arise in scientific research and engineering applications. Many such problems involve objective functions evaluated through computationally expensive simulations, for example, those based on the finite element method or computational fluid dynamics. When each function evaluation is costly, classical optimization techniques that require a large number of evaluations become impractical. A common approach to alleviate this difficulty is the use of surrogate models (or metamodels), which approximate the expensive objective function using a limited number of sampled observations. Several surrogate modeling techniques have been developed, including support vector regression (SVR) [2,28], radial basis functions (RBFs) [10,66], and Gaussian process regression (GPR) [50, 69]. Since constructing and evaluating these surrogate models is computationally inexpensive compared with the original simulations, they can significantly reduce the overall computational burden.

In this study, we considered the problem of single-objective global optimization of computationally expensive functions. Given an objective function $f : \mathcal{X} \subseteq \mathbb{R}^D \to \mathbb{R}$, the goal is to find an approximate solution under a limited evaluation budget:

$$x^* \approx \arg\min_{x \in \mathcal{X}} f(x), \tag{1.1}$$

where $\mathcal{X}$ is a box-constrained domain.

In many practical applications, evaluating $f(x)$ requires expensive numerical simulations or physical experiments, making direct optimization computationally prohibitive. Bayesian optimization (BO) addresses this challenge by constructing a probabilistic surrogate model of the objective function and sequentially selecting evaluation points that balance exploration and exploitation. Although a variety of surrogate models are available, Gaussian process (GP) surrogate models remain the standard choice because they provide both predictive mean and predictive uncertainty estimates within a unified probabilistic framework [24, 55]. These two quantities form the basis of most Bayesian optimization strategies, either by constructing scalar acquisition functions or by explicitly formulating a bi-objective optimization problem that preserves multiple exploration–exploitation trade-offs.

Bayesian optimization has attracted increasing attention in recent years due to its effectiveness in solving complex optimization problems across a wide range of applications. For example, it has been successfully applied to hyperparameter tuning in machine learning [61], materials and molecular discovery [27], time series forecasting [21], safe parameter tuning in robotics [3], and environmental modeling problems such as contaminant source localization [37, 48]. Recent research has also extended the framework to more challenging settings. Constrained Bayesian optimization has been studied through formulations for unknown constraints, stepwise uncertainty reduction, and broader treatments of constrained problems [25, 47, 59]. Expected improvement and efficient global optimization methods have also been developed for highly constrained or expensive black-box problems [33, 72, 73]. Comparative evaluations of infill sampling criteria for computationally expensive constrained optimization have also been reported [9]. In addition, Bayesian optimization has been extended to high-dimensional problems [22], and fairness-constrained learning [46]. These developments highlight the growing importance of BO as a general framework for efficient global optimization.

The selection of new evaluation points in BO is guided by an acquisition function, often referred to as the infill sampling criterion. This criterion evaluates the potential utility of candidate points based on both the predicted objective value and the uncertainty estimated by the surrogate model. Popular infill sampling criteria include probability of improvement (PI) [11], lower confidence bound (LCB) [12, 60], and expected improvement (EI) [44]. The EI criterion leads to the well-known efficient global optimization (EGO) algorithm [35], which has become a foundational method in surrogate-assisted optimization.

Recent advances in batch Bayesian optimization have introduced a variety of methods for selecting multiple evaluation points simultaneously. Representative approaches include q-expected improvement (qEI), q-upper confidence bound (qUCB), and batch knowledge gradient (qKG) [57, 70, 71]. Asynchronous batch methods, such as PLAyBOOK [1], employ local penalization to encourage diversity among candidate points, while scalable trust-region Bayesian optimization improves performance in high-dimensional settings by restricting the search to adaptive local regions [22]. These approaches have demonstrated the effectiveness of acquisition-function-based batch Bayesian optimization and serve as important baseline methods.

Despite their success, conventional acquisition-function-based Bayesian optimization combines the GP predictive mean and uncertainty into a single scalar acquisition function. Consequently, the exploration–exploitation trade-off is fixed by the chosen acquisition function and its associated parameters. In contrast, Pareto-based Bayesian optimization formulates infill selection as a bi-objective optimization problem using the predictive mean and predictive uncertainty, explicitly preserving multiple exploration–exploitation trade-offs on an approximated Pareto front. A separate Pareto-based solution selection mechanism is then used to select the final evaluation point(s). This decoupling of Pareto front approximation and solution selection provides greater flexibility for algorithm design and motivates the present study, which systematically investigates the individual contributions of these two components.

To avoid fixing the exploration–exploitation trade-off through scalarization, De Ath et al. [16] proposed the ε-Shotgun strategy, which formulates infill selection as a Pareto-based optimization problem using the GP predictive mean and predictive uncertainty. By introducing controlled stochasticity into the Pareto-based solution selection process, it helps prevent premature concentration of the search while maintaining low computational overhead. The effectiveness of the underlying ε-greedy strategy has led to its extension to asynchronous parallel Bayesian optimization [15], multi-objective Bayesian optimization [53], and simulation-based engineering optimization [20]. Related studies have also combined greedy search with diversification strategies and hybrid exploration mechanisms to improve batch construction and robustness [8, 18].

Furthermore, ε-Shotgun has been adopted as a representative greedy batch selection baseline in comparative studies of parallel and asynchronous Bayesian optimization [1, 17]. More broadly, the Bayesian optimization literature includes general methodological overviews [7] and multi-objectivized acquisition strategies that trade off solution quality and uncertainty [30, 31]. Parallel algorithm configuration, adaptive batch selection, and high-dimensional multi-objective extensions have also been investigated [52, 63, 64]. However, existing studies have largely focused on developing new algorithms or extending existing frameworks to different application domains, with limited attention paid to the individual roles of Pareto front approximation and Pareto-based solution selection. This motivates the present study, which systematically investigates these two components within a unified

Bayesian optimization with Pareto-based selection (BOPS) framework.

Main contributions: To address this gap, this paper conducts a comprehensive empirical study of the BOPS framework using multiple Pareto front approximation strategies and representative Pareto-based solution selection methods. The main contributions are summarized as follows:

- We present a unified BOPS framework that separates Pareto front approximation from Pareto-based solution selection, enabling a systematic investigation of their individual and combined effects.
- We study two deterministic Pareto-based solution selection strategies (ClusterHC and CSAW), together with two stochastic variants based on the exploitative ε-Shotgun mechanism, enabling a systematic comparison of deterministic and stochastic Pareto-based solution selection strategies.
- We perform a comprehensive experimental study on the CEC2022 benchmark suite and a real-world optimization benchmark, comparing the BOPS framework with three state-of-the-art acquisition-based batch BO baselines (qEI, qUCB, and qKG). The evaluation includes candidate-set-size sensitivity analysis, convergence analysis, pairwise Wilcoxon tests, critical difference diagrams, and batch diversity analysis.
- We provide practical insights into the design of Pareto-based batch Bayesian optimization, showing that optimization performance is governed primarily by the solution selection mechanism, whereas the choice of the Pareto front approximation strategy has comparatively little influence once a sufficiently accurate Pareto set has been obtained.

The remainder of this paper is organized as follows. Section 2 introduces the background concepts relevant to this work. Section 3 describes the BOPS framework and the methodologies in detail. Section 4 presents the experimental evaluation of Pareto front approximation strategies and Pareto-based solution selection methods. Finally, Section 5 summarizes the main findings and concludes the paper.

2. Background

2.1. Modeling with the Gaussian process

In Gaussian process regression, the unknown function $f(x)$ is assumed to be a realization of a Gaussian process,

$$f(x) \sim \mathcal{GP}(\mu_0(x), k(x, x')). \tag{2.1}$$

Here, $\mu_0(\cdot)$ denotes the prior mean function and $k(\cdot, \cdot)$ is the covariance function, which describes the covariance structure between the function values at two input locations. The covariance function, often referred to as the kernel, encodes prior assumptions about the function such as smoothness and correlation length.

Let

$$X_n = (x_1, \ldots, x_n)^\top, \qquad \mathbf{f}_n = (f(x_1), \ldots, f(x_n))^\top$$

denote the observed input locations and their corresponding function values. Assuming noise-free observations and a constant prior mean, the predictive distribution of $f(x)$ at a new point x is Gaussian [54]:

$$f(x) \mid \mathbf{f}_n \sim \mathcal{N}(\mu_n(x), \sigma_n^2(x)). \tag{2.2}$$

Let $C_n = C(X_n, X_n) \in \mathbb{R}^{n\times n}$ be the covariance matrix with entries

$$(C_n)_{ij} = k(x_i, x_j), \qquad i, j = 1, \ldots, n,$$

and let

$$\mathbf{c}(x) = \big(k(x, x_1), \ldots, k(x, x_n)\big)^\top$$

be the covariance vector between the prediction point x and the observed inputs.

In our implementation, the observed objective values are standardized prior to fitting the Gaussian process surrogate. Consequently, the GP employs a zero-mean prior on the normalized observations, and the predictive distribution follows the standard Gaussian process regression formulation [51]. The predictive mean and variance are then given by

$$\begin{aligned} \mu_n(x) &= \mathbf{c}(x)^\top C_n^{-1} \mathbf{f}_n, \\ \sigma_n^2(x) &= \hat{\sigma}^2 \left[1 - \mathbf{c}(x)^\top C_n^{-1} \mathbf{c}(x)\right], \end{aligned} \tag{2.3}$$

where $\hat{\sigma}^2$ denotes the process variance. The predictive mean $\mu_n(x)$ provides an estimate of the unknown objective function, while the predictive standard deviation $\sigma_n(x)$ quantifies the predictive uncertainty at x. A detailed derivation of Gaussian process regression can be found in [51, 56].

The expressions in (2.3) allow us to construct a probabilistic surrogate model based on the observed data. The predictive mean $\mu_n(x)$ serves as an estimate of the unknown function $f(x)$, while the predictive variance $\sigma_n^2(x)$ quantifies the uncertainty of this estimate. Typically, the uncertainty is small in regions where observations are dense and large in regions where the sampling points are sparse.

The present study focuses on deterministic expensive optimization problems and therefore adopts the standard noise-free Gaussian process formulation. For noisy evaluations, observation noise can be incorporated into the GP model, yielding predictive means and uncertainties that account for stochasticity [24, 51]. Since BOPS operates directly on these quantities, the same Pareto-based selection framework can be applied in principle to noisy settings.

2.2. Hypervolume indicator

Throughout this paper, $\mathcal{P}$ denotes a Pareto set (i.e., a set of non-dominated solutions in the decision space), while $\mathcal{F}$ denotes the corresponding Pareto front (i.e., the set of objective vectors in the objective space).

The hypervolume indicator is a widely used metric for assessing the quality of an approximated Pareto front. It measures the volume of the objective space dominated by the solutions in the set and bounded by a predefined reference point. A larger hypervolume value indicates a better approximation of the Pareto front, as it reflects both convergence toward the true Pareto front and diversity of the solutions.

Let $\mathcal{F} = \{y_1, y_2, \ldots, y_m\}$ denote an approximated Pareto front in the k-dimensional objective space, and let $z_{\text{ref}} \in \mathbb{R}^k$ be a reference point that is dominated by all points in $\mathcal{F}$. The hypervolume indicator of $\mathcal{F}$ with respect to z_{ref} is defined as [32]

$$\text{HV}(\mathcal{F}, z_{\text{ref}}) = \text{Volume}\left(\bigcup_{y \in \mathcal{F}} \{w \in \mathbb{R}^k \mid y \preceq w \preceq z_{\text{ref}}\}\right), \tag{2.4}$$

where $\preceq$ denotes the Pareto dominance relation.

The hypervolume contribution, $\text{HC}(\cdot)$, of a vector $y_i \in \mathcal{F}$ is defined as the decrease in hypervolume when y_i is removed from the set:

$$\text{HC}(y_i) = \text{HV}(\mathcal{F}, z_{\text{ref}}) - \text{HV}(\mathcal{F} \setminus \{y_i\}, z_{\text{ref}}). \tag{2.5}$$

When the correspondence between a Pareto-optimal solution $p_i \in \mathcal{P}$ and its objective vector $y_i \in \mathcal{F}$ is clear from the context, we equivalently write $\text{HC}(p_i)$ to denote the hypervolume contribution of the objective vector associated with p_i.

3. Methodology

3.1. Bayesian optimization with Pareto-based selection (BOPS)

The Bayesian optimization with Pareto-based selection (BOPS) framework consists of two modular components: Pareto front approximation and Pareto-based solution selection. In this study, we systematically investigate the individual and combined effects of these two components within a unified experimental framework.

Figure 1 illustrates the overall workflow of the BOPS framework. Starting from an initial experimental design, a Gaussian process surrogate model is constructed using the available observations. The predictive mean and predictive standard deviation of the GP define the bi-objective optimization problem. An approximated Pareto set is then generated using one of the Pareto front approximation strategies (Factor A), followed by one of the Pareto-based solution selection mechanisms (Factor B) to construct a batch of candidate solutions. The expensive objective function is evaluated at the selected points, the GP surrogate is updated, and the optimization process continues until the stopping criterion is satisfied.

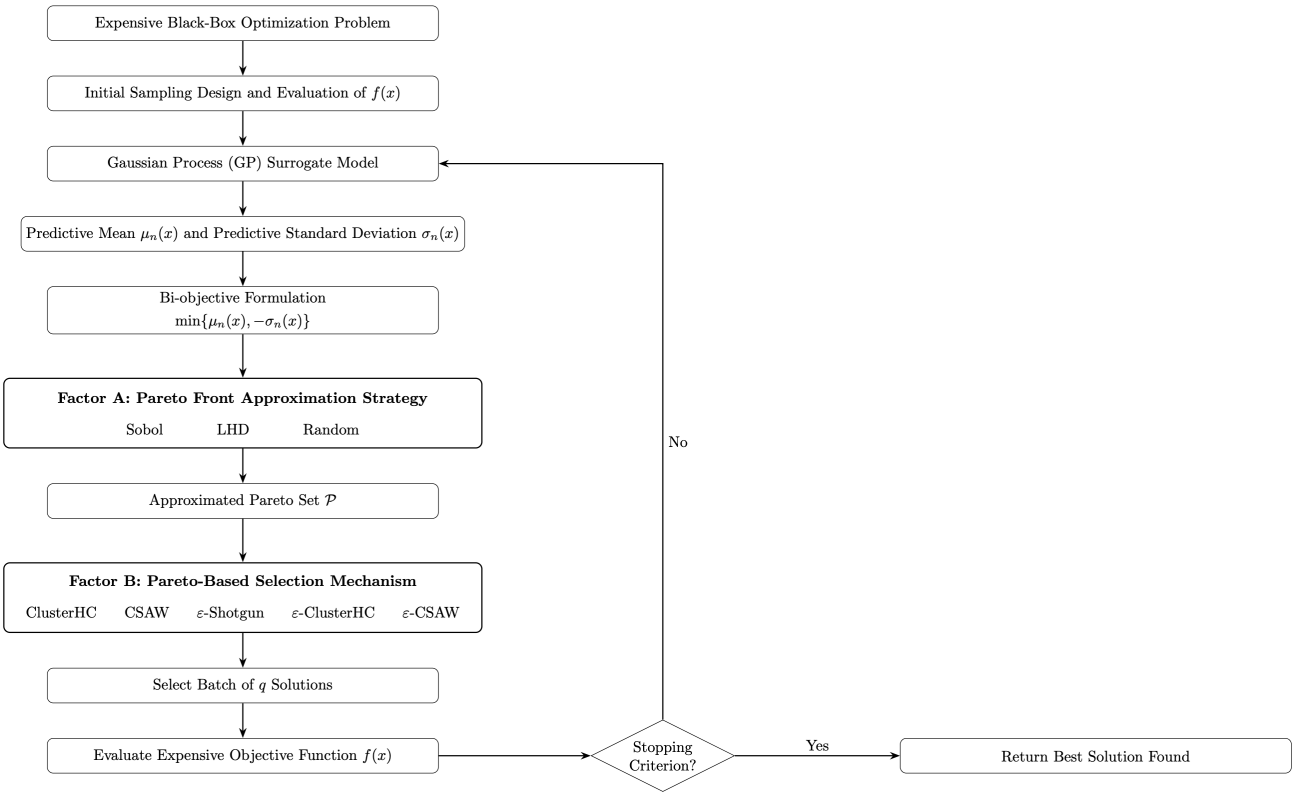


Figure 1. Overall workflow of the Bayesian optimization with Pareto-based selection (BOPS) framework. The workflow consists of two modular components investigated in this study: Factor A (Pareto front approximation) and Factor B (Pareto-based solution selection). Different combinations of these two components define the BOPS variants evaluated in the experimental study.

Algorithm 1 summarizes the corresponding optimization procedure. The following sections describe the Pareto front approximation strategies and Pareto-based solution selection mechanisms in detail.

Algorithm 1 Bayesian optimization with Pareto-based selection (BOPS)

Require: Objective function f; Pareto front approximation strategy $\mathcal{A}$; Pareto-based solution selection criterion ξ
  Generate an initial observation set and fit a GP surrogate model
  while stopping criterion not met **do**
    Approximate the Pareto front of the bi-objective problem (3.1) using strategy $\mathcal{A}$
    Select a batch of candidate solutions from the approximated Pareto set using criterion ξ
    Evaluate the expensive objective function at the selected points
    Update the observation set and retrain the GP surrogate model
  end while
  return Best solution found

As remarked in [38], under the standard minimization setting, the expected improvement (EI)

criterion [35] is a decreasing function of the predictive mean $\mu_n(x)$ and an increasing function of the predictive standard deviation $\sigma_n(x)$, provided that $\sigma_n(x) > 0$. Consequently, any point that maximizes the EI criterion must belong to the Pareto set of the bi-objective problem (3.1). Therefore, the BOPS framework can be interpreted as a generalization of the classical acquisition-based Bayesian optimization.

The subsequent subsections describe the individual components of the BOPS framework in detail. We first formulate the bi-objective optimization problem based on the Gaussian process surrogate, followed by the Pareto front approximation strategies and the Pareto-based solution selection mechanisms.

3.2. Bi-objective formulation

Using the Gaussian process surrogate model described in Section 2.1, the infill selection problem is formulated as a bi-objective optimization problem based on the predictive mean and predictive standard deviation of the surrogate. The predictive mean $\mu_n(x)$ represents the exploitation objective by favoring candidate solutions with low predicted objective values, whereas the predictive standard deviation $\sigma_n(x)$ represents the exploration objective by favoring regions with high predictive uncertainty. Accordingly, the infill selection problem is formulated as

$$\min_{x \in \mathcal{X}} \{\mu_n(x), -\sigma_n(x)\}. \tag{3.1}$$

Here, minimizing $-\sigma_n(x)$ is equivalent to maximizing the predictive uncertainty and therefore encourages exploration.

The decision variables remain the original design variables $x \in \mathcal{X} \subset \mathbb{R}^D$, while Pareto dominance is evaluated in the two-dimensional surrogate objective space defined by $(\mu_n(x), -\sigma_n(x))$. Before constructing the Pareto front, the two surrogate objectives are normalized to ensure a consistent treatment of exploitation and exploration.

Specifically, both objectives are normalized to the interval $[0, 1]$. Since this normalization is a positive affine transformation, it preserves Pareto dominance relationships while providing a consistent scale for subsequent solution selection mechanisms and distance-based calculations.

The approximated Pareto-optimal solutions of problem (3.1) constitute the candidate set from which the next batch of expensive function evaluations is selected. The following subsection describes the space-filling sampling strategies used to approximate this Pareto set.

3.3. Pareto front approximation via space-filling sampling

In this study, Pareto front approximation within the BOPS framework is performed using space-filling sampling strategies rather than population-based evolutionary algorithms. Given a set of candidate solutions generated by a space-filling sampling strategy, the Gaussian process surrogate is evaluated to obtain the predictive mean and predictive standard deviation of each candidate. Non-dominated sorting is then performed in the resulting bi-objective space to construct the approximate Pareto set.

In particular, let $\mathcal{X} = [x^{\text{lb}}, x^{\text{ub}}] \subset \mathbb{R}^D$ denote the bounded search space, where x^{lb} and x^{ub} are the lower and upper bounds, respectively. Given a sampling budget of N, a candidate set $\mathcal{S}_{\text{cand}} = \{x_i\}_{i=1}^N$ is generated using a space-filling sampling strategy. The surrogate model is subsequently evaluated at

each candidate point, and non-dominated sorting is applied in the bi-objective space $\{\mu_n(x), -\sigma_n(x)\}$ to obtain the approximate Pareto set $\mathcal{P}$. The overall approximation procedure is summarized in Algorithm 2.

Algorithm 2 Pareto front approximation via space-filling sampling

Require: GP surrogate model; sampling budget N; search domain $\mathcal{X}$; space-filling sampling strategy
Ensure: Approximate Pareto set $\mathcal{P}$

1: Generate a candidate set $\mathcal{S}_{\text{cand}} \subseteq \mathcal{X}$ of N points using the selected space-filling sampling strategy
2: Evaluate the surrogate objective values $\{\mu_n(x), -\sigma_n(x)\}$ for all candidate points in $\mathcal{S}_{\text{cand}}$
3: Apply non-dominated sorting to the resulting objective vectors
4: Extract the first non-dominated front as the approximate Pareto set $\mathcal{P}$
5: **return** $\mathcal{P}$

Three space-filling sampling strategies are investigated in this study.

1) **Sobol sequence sampling.** Sobol sequences [34] are low-discrepancy, quasi-random sequences designed to provide uniform coverage of high-dimensional search spaces. A scrambled Sobol sequence is used to generate candidate points, typically yielding better space-filling properties than purely random sampling.

2) **Latin hypercube design.** Latin hypercube design (LHD) [14] is a stratified sampling technique in which each dimension is partitioned into N equally sized intervals. One sample is drawn from each interval in every dimension, producing uniform marginal distributions and improved space-filling coverage compared with uniform random sampling.

3) **Uniform random sampling.** Candidate points are sampled independently from a uniform distribution over the unit hypercube and subsequently scaled to the problem bounds. This simple baseline strategy does not explicitly enforce space-filling properties and may produce uneven coverage, particularly in high-dimensional search spaces.

3.4. Pareto-based solution selection

After constructing the approximated Pareto-optimal set, the next step is to select one or more candidate solutions for expensive function evaluation. The choice of solution selection criterion determines how the approximated Pareto set is exploited and therefore plays a central role in balancing exploration and exploitation within the BOPS framework.

The solution selection methods investigated in this study are divided into two categories: deterministic Pareto-based solution selection (ClusterHC and CSAW) and stochastic Pareto-based solution selection (ε-Shotgun, ε-ClusterHC, and ε-CSAW). The latter two are obtained by incorporating the ε-greedy exploration mechanism of ε-Shotgun into the corresponding deterministic Pareto-based solution selection methods.

3.4.1. Deterministic Pareto-based solution selection

Two deterministic Pareto solution selection mechanisms from the optimization literature are adapted within the BOPS framework. ClusterHC originates from the clustering-assisted representative

selection strategy of Jiang et al. [32], originally developed for surrogate-assisted constrained expensive optimization, where representative solutions are selected from a Pareto set generated by simultaneously optimizing the surrogate-based infill criteria Sigma (objective uncertainty) and consLCB (constraint-boundary uncertainty). CSAW originates from the compromise solution selection method of Wang and Rangaiah [67], originally proposed for selecting a preferred solution from a Pareto front in general multi-objective optimization problems. In the present work, both methods are adapted and incorporated into the BOPS framework as deterministic Pareto-based solution selection mechanisms.

I. Clustering-assisted hypervolume contribution selection (ClusterHC)

Clustering-assisted hypervolume contribution selection (ClusterHC) is adapted from the clustering-assisted infill criterion proposed by Jiang et al. [32]. The original method applies the density-based spatial clustering of applications with noise (DBSCAN) algorithm [23] to partition the approximated Pareto set into clusters and selects the solution with the largest hypervolume contribution from each cluster. Since the number of selected solutions depends on the number of clusters identified by DBSCAN, the resulting batch size is not necessarily fixed. The DBSCAN algorithm employs two parameters: the neighborhood radius β and the minimum number of neighboring points, denoted by MinPts. To address cases where the number of detected clusters exceeds the required batch size q, the original method applies k-means clustering [43] to repartition the Pareto set into exactly q clusters.

Algorithm 3 Clustering-assisted hypervolume contribution selection (ClusterHC)

Require: Pareto set $\mathcal{P}$; batch size q; DBSCAN parameters β and MinPts
Ensure: Selected solution set $\mathcal{X}'$

Apply DBSCAN with parameters β and MinPts to the objective vectors of solutions in $\mathcal{P}$, obtaining clusters $\{C_j\}_{j=1}^{J}$
if $J > q$ **then**
  Repartition $\mathcal{P}$ into exactly q clusters using k-means clustering in the objective space
  Replace $\{C_j\}_{j=1}^{J}$ by the resulting q clusters
end if
Compute the hypervolume contribution $\mathrm{HC}(p)$ for each solution $p \in \mathcal{P}$
$\mathcal{X}' \leftarrow \emptyset$
for each cluster C_j **do**
  $p_j^* \leftarrow \arg\max_{p_i \in C_j} \mathrm{HC}(p_i)$
  $\mathcal{X}' \leftarrow \mathcal{X}' \cup \{p_j^*\}$
end for
if $|\mathcal{X}'| < q$ **then**
  $\mathcal{P}_{\text{remaining}} \leftarrow \mathcal{P} \setminus \mathcal{X}'$
  Sort $\mathcal{P}_{\text{remaining}}$ in descending order of $\mathrm{HC}(\cdot)$
  Select the top $(q - |\mathcal{X}'|)$ solutions from $\mathcal{P}_{\text{remaining}}$
  $\mathcal{X}' \leftarrow \mathcal{X}' \cup \{\text{selected solutions}\}$
end if
return $\mathcal{X}'$

However, when DBSCAN identifies fewer than q clusters, the original method may return fewer

than q candidate solutions. Since the BOPS framework requires exactly q candidate solutions at every iteration, we augment the original ClusterHC selection strategy with a hypervolume-based completion step for this case, as summarized in Algorithm 3. Whenever fewer than q cluster representatives are obtained, the remaining solutions are selected from the unchosen Pareto-optimal solutions in descending order of hypervolume contribution. This modification guarantees a fixed batch of q candidate solutions, denoted by $\mathcal{X}'$, while preserving diversity across different trade-off regions and promoting convergence toward high-quality solutions.

II. Clustering-assisted simple additive weighting with entropy-based weights (CSAW)

The clustering-assisted simple additive weighting (CSAW) method is constructed by integrating the simple additive weighting (SAW) method [67] with the DBSCAN-based clustering strategy. While the original SAW method requires manually specified objective weights, we instead employ an entropy-based weighting scheme [75] to automatically determine the importance of each objective. Objectives with lower entropy, indicating greater discriminative power, are assigned larger weights.

Let

$$G = \left(g_{ij}\right) \in \mathbb{R}^{m\times k}$$

denote the objective matrix of the approximated Pareto set $\mathcal{P} = \{x_1, \ldots, x_m\}$, where m is the number of Pareto solutions, k is the number of objectives, and g_{ij} is the value of the jth objective evaluated at the ith Pareto solution. In the present work, $k = 2$, corresponding to the predictive mean $\mu_n(x)$ ($j = 1$) and the negative predictive standard deviation $-\sigma_n(x)$ ($j = 2$).

The resulting entropy-based CSAW criterion operates on the objective matrix G. Each objective is first independently normalized to the unit interval:

$$\tilde{g}_{ij} = \frac{\max_i g_{ij} - g_{ij}}{\max_i g_{ij} - \min_i g_{ij}}. \tag{3.2}$$

Since all objectives are formulated as minimization objectives, this normalization transforms them into benefit attributes, for which larger normalized values indicate better performance.

Entropy is then used to quantify the dispersion of each objective across the approximated Pareto set and thereby determine its discriminatory power [75]. A probability distribution over the m Pareto solutions is first formed as

$$p_{ij} = \frac{\tilde{g}_{ij}}{\sum_{i=1}^{m} \tilde{g}_{ij}}, \tag{3.3}$$

and the normalized entropy of objective j is computed as

$$E_j = -\frac{1}{\ln(m)} \sum_{i=1}^{m} p_{ij} \ln p_{ij}, \tag{3.4}$$

so that $E_j \in [0, 1]$. The corresponding entropy weight is

$$w_j = \frac{1 - E_j}{\sum_{j'=1}^{k}(1 - E_{j'})}. \tag{3.5}$$

Using these entropy-based weights, each Pareto solution is assigned a simple additive weighting (SAW) score [67]:

$$A_i = \sum_{j=1}^{k} w_j \tilde{g}_{ij}, \tag{3.6}$$

which reflects a compromise among the multiple objectives.

The complete CSAW procedure is summarized in Algorithm 4.

Algorithm 4 Clustering-assisted simple additive weighting (CSAW)

Require: Pareto set $\mathcal{P}$; batch size q; DBSCAN parameters β and MinPts
Ensure: Selected solution set $\mathcal{X}'$

Normalize the objective matrix according to (3.2)
Compute the probability distribution, entropy, and entropy weights according to (3.3)–(3.5)
Compute the SAW score for each solution according to (3.6)
Apply DBSCAN with parameters β and MinPts to the normalized objective vectors of solutions in $\mathcal{P}$, obtaining clusters $\{C_j\}_{j=1}^{J}$
if $J > q$ **then**
    Apply k-means clustering to repartition $\mathcal{P}$ into exactly q clusters
    Replace $\{C_j\}_{j=1}^{J}$ by the resulting q clusters
end if
$\mathcal{X}' \leftarrow \emptyset$
for each cluster C_j **do**
    $x_j^* \leftarrow \arg\max_{x_i \in C_j} A_i$
    $\mathcal{X}' \leftarrow \mathcal{X}' \cup \{x_j^*\}$
end for
if $|\mathcal{X}'| < q$ **then**
    $\mathcal{P}_{\text{remaining}} \leftarrow \mathcal{P}_{\text{remaining}} \setminus \mathcal{X}'$
    Sort $\mathcal{P}_{\text{remaining}}$ in descending order of SAW score
    Select the top $(q - |\mathcal{X}'|)$ solutions
    $\mathcal{X}' \leftarrow \mathcal{X}' \cup \{\text{selected solutions}\}$
end if
return $\mathcal{X}'$

Although ClusterHC and CSAW are deterministic Pareto solution selection mechanisms, they can be naturally extended to incorporate controlled stochastic exploration. This extension is inspired by the ε-Shotgun algorithm [16], which combines ε-greedy exploration with Pareto-based batch Bayesian optimization. By replacing the random Pareto solution selection step in ε-Shotgun with deterministic Pareto selection operators, the same exploration mechanism can be incorporated into ClusterHC and CSAW, yielding the stochastic ε-ClusterHC and ε-CSAW variants while preserving the remaining components of the original ε-Shotgun algorithm.

To provide the necessary background, we first briefly review the original ε-Shotgun algorithm before introducing the stochastic Pareto-based solution selection framework.

3.4.2. ε-Shotgun

The ε-Shotgun strategy, proposed by De Ath et al. [16, 17], is a stochastic batch Bayesian optimization strategy that introduces controlled exploration during batch construction. Rather than constructing the entire batch simultaneously, the method first selects a single anchor point using an ε-greedy policy and then generates the remaining batch points by local Gaussian sampling around this anchor. This strategy substantially reduces the computational cost of batch Bayesian optimization while maintaining an effective balance between exploitation and exploration.

The first point of the batch, denoted by x_1', is selected according to the ε-greedy rule

$$x_1' = \begin{cases} \arg\min_{x\in\mathcal{X}} \mu_n(x), & \text{with probability } 1-\varepsilon, \\ \text{randomChoice}(\mathcal{P}), & \text{with probability } \varepsilon, \end{cases} \tag{3.7}$$

where $\mathcal{P}$ denotes the approximated Pareto set obtained from the bi-objective formulation (3.1). With probability $1-\varepsilon$, the algorithm exploits by selecting the point with the smallest predictive mean. With probability ε, the algorithm explores by randomly selecting a solution from $\mathcal{P}$. Since Pareto solutions simultaneously exhibit low predicted objective values and high predictive uncertainty, this exploration strategy encourages sampling in promising yet underexplored regions.

Once the anchor point has been selected, the remaining batch points are generated by sampling from a local Gaussian distribution,

$$x_i' \sim \mathcal{N}(x_1', r^2 I), \qquad i = 2, \ldots, q. \tag{3.8}$$

Since the Gaussian distribution has unbounded support, sampled points falling outside the search domain $\mathcal{X}$ are discarded and resampled until a feasible candidate is obtained. The adaptive sampling radius is defined as

$$r = \frac{|\mu_n(x_1') - f^\star|}{\tilde{L}} + \gamma \frac{\sigma_n(x_1')}{\tilde{L}}, \tag{3.9}$$

with the local search hypercube domain

$$\Omega(x_1') = [x_1' - l,\ x_1' + l]^D \tag{3.10}$$

and the local gradient bound

$$\tilde{L} = \max_{x\in\Omega(x_1')} \|\nabla\mu_n(x)\|. \tag{3.11}$$

The complete ε-Shotgun procedure is summarized in Algorithm 5. Here, $f^\star = \min_{i\le M} f_i$ denotes the best objective value observed so far, where M is the number of function evaluations collected up to the current iteration, $\sigma_n(x_1')$ is the predictive standard deviation at the anchor point, $\gamma \ge 0$ is the exploration parameter, and l is the kernel length scale defining the local search hypercube.

Algorithm 5 ε-Shotgun query point selection for batch Bayesian optimization

Require: Pareto set $\mathcal{P}$; batch size q; exploration probability ε; exploration parameter γ; kernel length scale l; best objective value observed so far $f^\star$

Ensure: Batch of query points $\mathcal{X}'$

1: **if** rand() $< \varepsilon$ **then**
2: $\quad x'_1 \leftarrow$ randomChoice($\mathcal{P}$)
3: **else**
4: $\quad x'_1 \leftarrow \arg\min_{x\in\mathcal{X}} \mu_n(x)$
5: **end if**
6: Compute the local gradient bound $\tilde{L}$ according to (3.11)
7: Compute the sampling radius r according to (3.9)
8: $\mathcal{X}' \leftarrow \{x'_1\}$
9: **while** $|\mathcal{X}'| < q$ **do**
10: $\quad$ Sample $x' \sim \mathcal{N}(x'_1, r^2 I)$ according to (3.8)
11: $\quad$ **if** $x' \in \mathcal{X}$ **then**
12: $\quad\quad \mathcal{X}' \leftarrow \mathcal{X}' \cup \{x'\}$
13: $\quad$ **end if**
14: **end while**
15: **return** $\mathcal{X}'$

The controlled stochastic exploration mechanism of ε-Shotgun can be incorporated into deterministic Pareto-based solution selection methods, giving rise to stochastic Pareto-based selection strategies.

3.4.3. Stochastic Pareto-based solution selection

The deterministic Pareto-based solution selection methods introduced in the previous subsection can be naturally extended to incorporate stochasticity. Inspired by the ε-greedy mechanism of ε-Shotgun, we construct stochastic variants by combining deterministic Pareto-based solution selection with probabilistic switching.

Let $\xi(\cdot)$ denote a deterministic Pareto-based solution selection operator that selects a batch of q candidate solutions from the approximated Pareto set $\mathcal{P}$. Thus,

$$\mathcal{X}' = \xi(\mathcal{P}),$$

where $|\mathcal{X}'| = q$.

The stochastic Pareto-based solution selection framework is summarized in Algorithm 6. With probability ε, the entire batch of q candidate solutions is obtained by applying the deterministic solution selection operator $\xi(\cdot)$ to the approximated Pareto set. Otherwise, with probability $1 - \varepsilon$, the original exploitative mechanism of ε-Shotgun is employed by selecting the anchor point that minimizes the predictive mean and generating the remaining $(q - 1)$ query points according to Steps 6–14 of Algorithm 5. This differs from the original ε-Shotgun strategy, which always generates the remaining batch points by local Gaussian sampling around a single Pareto-selected anchor point. As $\varepsilon \to 1$, the stochastic framework recovers the corresponding deterministic Pareto-based solution selection method. Conversely, as $\varepsilon \to 0$, it reduces to the purely exploitative ε-Shotgun variant.

Algorithm 6 Stochastic Pareto-based solution selection framework

Require: Pareto set $\mathcal{P}$; batch size q; exploration probability ε; deterministic Pareto-based selection operator $\xi(\cdot)$; exploration parameter γ; kernel length scale l; best objective value observed so far $f^\star$

Ensure: Batch of query points $\mathcal{X}'$

1: **if** rand() $< \varepsilon$ **then**
2: $\quad \mathcal{X}' \leftarrow \xi(\mathcal{P})$
3: **else**
4: $\quad x'_1 \leftarrow \arg\min_{x\in\mathcal{X}} \mu_n(x)$
5: $\quad \mathcal{X}' \leftarrow \{x'_1\}$
6: $\quad$ Generate the remaining $(q-1)$ query points according to Steps 6–14 of Algorithm 5
7: **end if**
8: **return** $\mathcal{X}'$

In this work, two deterministic Pareto-based solution selection operators are considered within the stochastic Pareto-based solution selection framework:

- ξ = ClusterHC, defined by Algorithm 3, yielding the ε-ClusterHC strategy;
- ξ = CSAW, defined by Algorithm 4, yielding the ε-CSAW strategy.

Although this study focuses on ClusterHC and CSAW, the framework is readily applicable to any deterministic Pareto-based solution selection criterion, providing a unified approach for incorporating controlled stochasticity into batch Bayesian optimization.

3.5. Theoretical motivation and convergence

The convergence behavior of the BOPS framework is closely related to the exploration–exploitation trade-off induced by the Gaussian process surrogate and the selection mechanism used to construct candidate batches. In the BOPS framework, candidate evaluation points are selected from a Pareto set constructed from the predictive mean $\mu(x)$ and predictive standard deviation $\sigma(x)$ of the surrogate model. These two quantities naturally represent the exploitation and exploration objectives in Bayesian optimization: small values of $\mu(x)$ correspond to promising objective values, while large values of $\sigma(x)$ indicate regions of high predictive uncertainty.

The following lemma summarizes a well-known relationship between the expected improvement acquisition function and the mean–uncertainty Pareto frontier [24, 35].

Lemma 1 (Relation to expected improvement)**.** *Consider a minimization problem and let $f^\star$ denote the best objective value observed so far. Suppose that the Gaussian process predictive distribution at x is*

$$f(x) \mid \mathcal{D}_n \sim \mathcal{N}(\mu(x), \sigma^2(x)), \qquad \sigma(x) > 0.$$

Then, any maximizer of the expected improvement acquisition function lies on the Pareto frontier defined by the bi-objective space $(\mu(x), -\sigma(x))$.

Proof. For minimization, the EI criterion can be written as

$$EI(x) = (f^\star - \mu(x))\Phi(z) + \sigma(x)\phi(z), \qquad z = \frac{f^\star - \mu(x)}{\sigma(x)},$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ denote the standard normal cumulative distribution function and probability density function, respectively.

It is well known that, for fixed $f^\star$, EI is non-increasing in $\mu(x)$ and non-decreasing in $\sigma(x)$ [35]. Suppose that a point x is dominated by another point x' in the objective space $(\mu(x), -\sigma(x))$, meaning that

$$\mu(x') \leq \mu(x), \qquad \sigma(x') \geq \sigma(x),$$

with at least one inequality strict. It follows that

$$EI(x') \geq EI(x),$$

with strict inequality whenever at least one of the dominance relations is strict and $\sigma(x) > 0$. Therefore, a dominated point cannot maximize EI, and any EI maximizer must belong to the Pareto frontier of $(\mu(x), -\sigma(x))$. □

This lemma provides a theoretical motivation for the BOPS framework. Instead of optimizing a single scalar acquisition function, the algorithm explicitly constructs the Pareto front representing the exploration–exploitation trade-off and selects candidate solutions from the corresponding Pareto set using dedicated selection criteria. Consequently, every maximizer of the expected improvement criterion belongs to the Pareto set considered by BOPS. The Pareto set also contains additional candidate solutions corresponding to alternative exploration–exploitation trade-offs that are not necessarily EI maximizers.

The solution selection mechanism further influences the exploration behavior of the algorithm. In particular, the ε-Shotgun strategy introduces controlled stochastic exploration during batch construction, encouraging evaluation points to be selected from diverse regions of the approximated Pareto frontier rather than concentrating prematurely on a single region.

Persistent exploration is a key ingredient in convergence analyses of Gaussian-process-based Bayesian optimization. Under standard assumptions, acquisition strategies that maintain sufficient exploration converge asymptotically to the global optimum [6,57]. Since BOPS explicitly incorporates predictive uncertainty through Pareto construction and stochastic solution selection, its search behavior is consistent with the exploration principles underlying these results. However, this provides theoretical motivation rather than a formal convergence proof. Establishing rigorous convergence guarantees for the complete BOPS framework is left for future work.

4. Numerical experiments

4.1. Experimental setup

The experimental study compares a total of 18 methods. Within the BOPS framework, 15 variants are obtained by combining three Pareto front approximation strategies (Sobol, LHD, and uniform random sampling) with five Pareto-based solution selection methods (ClusterHC, CSAW, ε-Shotgun, ε-ClusterHC, and ε-CSAW). These are further compared with three acquisition-function-based batch Bayesian optimization baselines: q-expected improvement, q-upper confidence bound, and q-knowledge gradient.

To evaluate the performance of the BOPS variants and acquisition-function baselines, we use the 20-dimensional CEC2022 benchmark suite, which comprises shifted and rotated unimodal, multimodal,

and composition functions with known global optimum values. The optimal values reported in Appendix A correspond to the bias constants specified by the benchmark suite, so the global minimum of each function equals the reported value rather than zero.

Each algorithm variant is executed over 30 independent runs to account for stochastic variability. To compare methods under the limited evaluation budgets typical of expensive black-box optimization, all methods are allocated the same budget of 800 function evaluations, with a batch size of $q = 4$ evaluated in parallel at each iteration. This evaluation budget follows the original ε-Shotgun study [16] and is consistent with previous studies using the CEC benchmark suite in Bayesian optimization [29, 74] and surrogate-assisted expensive optimization [13, 42].

An initial observation set of size $5D$, generated using Latin hypercube design (LHD), is used to fit the initial Gaussian process surrogate model, where D denotes the problem dimensionality. For Pareto set construction in Algorithm 2, $N = 1000D$ candidate points are subsequently generated using the sampling-based strategies. This linear scaling balances coverage of the search space and computational cost. The choice of $N = 1000D$ is justified by the candidate-set-size sensitivity analysis presented in Section 4.6. The surrogate is evaluated at these candidate points, after which non-dominated sorting is applied to extract the approximated Pareto-optimal set.

A GP with a Matérn-5/2 kernel is fitted and updated after each batch. The observed objective values are standardized prior to GP fitting, and the kernel hyperparameters are re-estimated at each iteration by maximizing the GP log marginal likelihood using the optimization routines provided by GPy [26].

To facilitate reproducibility, the complete experimental parameter settings are provided in Appendix E (Tables E1 and E2). All experiments are implemented in Python, with details of the external software packages provided in Appendix B. In addition, the implementation of the Pareto-based methods investigated in this study is publicly available at `https://github.com/KittisakChaiyotha/BOPS`.

4.2. Overall performance comparison

In the numerical section, ε-Shotgun, ε-ClusterHC, and ε-CSAW are abbreviated as eShotgun, eClusterHC, and eCSAW, respectively, while ClusterHC and CSAW denote the corresponding deterministic methods.

The complete numerical results are provided in Appendix D. Tables D1–D3 report the performance of the 15 BOPS variants under the three Pareto front approximation strategies, together with the three acquisition-function-based batch Bayesian optimization baselines: qEI, qUCB, and qKG. Results are reported for the twelve CEC2022 benchmark functions and the real-world *push4* benchmark [16, 68]. The following analyses focus on the CEC2022 benchmark suite, while the results for *push4* are discussed separately in Section 4.7.

Overall, the numerical results indicate that both the Pareto front approximation strategy and the Pareto-based solution selection method influence the performance of the BOPS framework. However, the subsequent analyses show that these two components do not contribute equally: the choice of Pareto-based solution selection method has a substantially greater impact on optimization performance than the choice of the Pareto front approximation strategy.

A clear pattern is the consistently strong performance of eShotgun. Across all three Pareto front approximation strategies, eShotgun frequently achieves the best or near-best mean objective value on the CEC2022 benchmark functions, outperforming or matching the three acquisition-function baselines

in most cases (see Tables D1–D3). Among the acquisition-function baselines, qEI and qUCB generally perform competitively, whereas qKG consistently yields the weakest performance on the CEC2022 benchmark suite.

To facilitate a direct comparison of all method–strategy combinations, Figure 2 presents the average convergence curves on the CEC2022 benchmark suite. Since the benchmark functions have different objective scales, the best-so-far objective value of each independent run is normalized to the interval [0, 1] according to

$$\tilde{r}_{m,i}^{(t,p)} = \frac{r_{m,i}^{(t,p)} - \min R_p}{\max R_p - \min R_p}, \tag{4.1}$$

where

$$R_p = \left\{ r_{m,i}^{(t,p)} : m = 1, \ldots, 18,\ i = 1, \ldots, 800,\ t = 1, \ldots, 30 \right\}, \tag{4.2}$$

and $r_{m,i}^{(t,p)}$ denotes the best-so-far objective value obtained by method–strategy combination m at the i-th function evaluation during the t-th independent run on benchmark function p. Thus, R_p contains all best-so-far objective values generated throughout the entire optimization process for benchmark function p. Consequently, $\tilde{r}_{m,i}^{(t,p)} = 0$ corresponds to the best observed performance and $\tilde{r}_{m,i}^{(t,p)} = 1$ to the worst observed performance over all method–strategy combinations, function evaluations, and independent runs for benchmark function p. This normalization is used solely for visualization; all subsequent statistical analyses are performed using the original objective values.

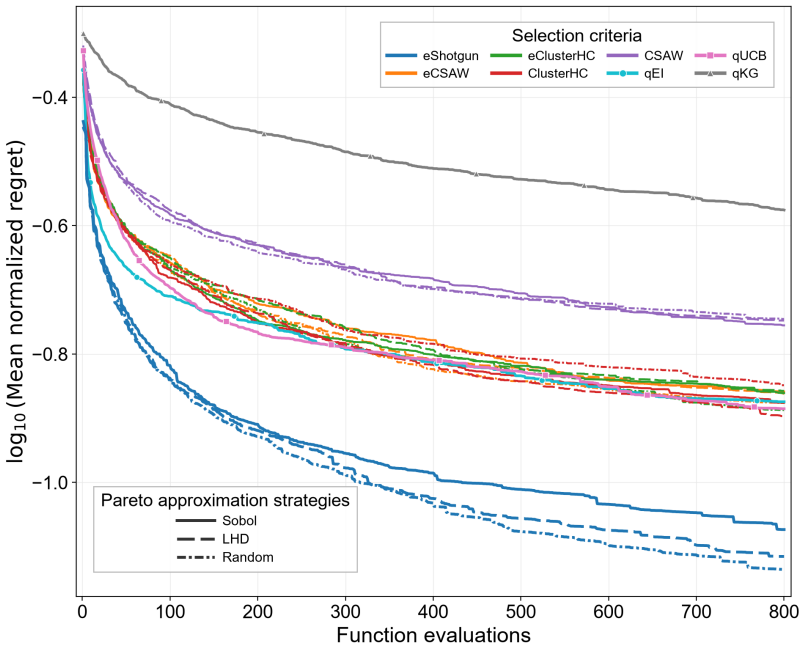


Figure 2. Mean normalized best-so-far convergence curves, averaged over 30 independent runs on the CEC2022 benchmark suite, for all Pareto-based solution selection methods under the three Pareto front approximation strategies. The vertical axis is displayed on a base-10 logarithmic scale. Lower values indicate better optimization performance. Colors denote Pareto-based solution selection methods, line styles denote Pareto front approximation strategies, and the acquisition-function baselines are shown using markers.

Figure 2 is obtained by averaging the normalized values over all $12 \times 30 = 360$ function–run pairs at each function evaluation. To improve the visibility of differences among methods, the vertical axis is displayed on a base-10 logarithmic scale. Several observations can be made. For each Pareto-based solution selection method, the convergence curves corresponding to the three space-filling approximation strategies are nearly indistinguishable throughout the optimization process. This indicates that the choice of space-filling approximation strategy has only a minor influence on the overall convergence behavior.

In contrast, much larger differences are observed among the solution selection methods. Across all three approximation strategies, eShotgun consistently achieves the lowest normalized regret, indicating superior optimization performance throughout the search. ClusterHC, eClusterHC, eCSAW, qEI, and qUCB form a competitive middle group, whereas CSAW converges more slowly and maintains a higher normalized regret. Among all methods, qKG consistently exhibits the weakest convergence behavior.

Overall, the variation associated with the choice of solution selection method is substantially larger than that introduced by the space-filling approximation strategy.

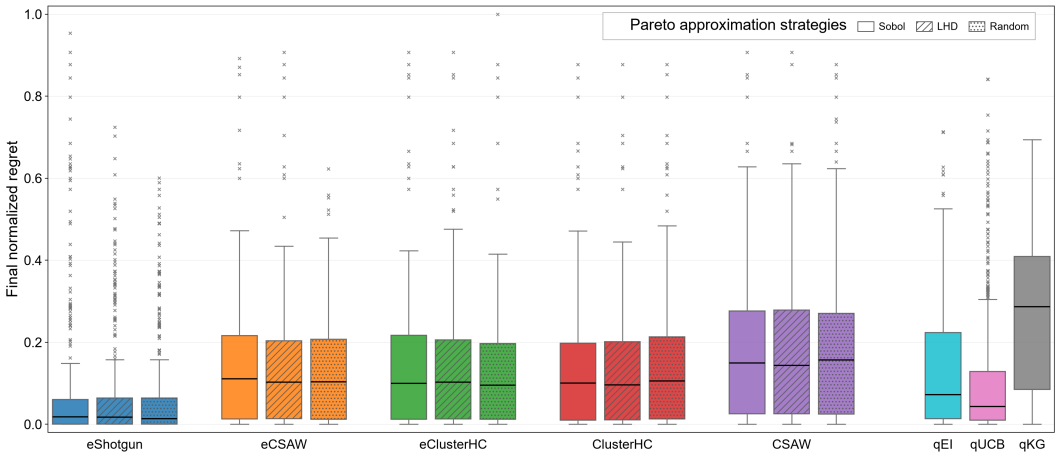


Figure 3. Distribution of the normalized final regret after 800 function evaluations for all method–strategy combinations on the CEC2022 benchmark suite. Each box plot summarizes the pooled normalized final regrets from 30 independent runs over the twelve benchmark functions (360 values). Lower values indicate better optimization performance. Colors denote Pareto-based solution selection methods, hatch patterns denote Pareto front approximation strategies, and the last three box plots correspond to the acquisition-function baselines.

Figure 3 summarizes the distribution of the normalized final regrets for all method–strategy combinations by pooling the final best-so-far objective values over the 30 independent runs and the twelve CEC2022 benchmark functions. Constructed from the same normalized run-level data as Figure 2 at the final function evaluation, the box plots complement the convergence curves by illustrating both the central tendency and the variability of the final optimization performance.

eShotgun consistently achieves the lowest median normalized regret together with the smallest interquartile range, indicating both superior final performance and strong robustness across the benchmark suite. The remaining Pareto-based solution selection methods remain competitive, with

the exception of CSAW, which exhibits a higher median regret and greater variability. Among the acquisition-function baselines, qEI and qUCB remain competitive, although their median performance is generally inferior to that of eShotgun, whereas qKG exhibits substantially higher median regret and considerably greater variability, making it the weakest baseline.

For each Pareto-based solution selection method, the three box plots corresponding to the different space-filling approximation strategies are nearly indistinguishable. The medians and interquartile ranges are highly consistent across Sobol, LHD, and uniform random sampling, indicating that the choice of space-filling approximation strategy has only a minor influence on the final optimization performance.

A comparison between the deterministic methods and their ε-variants reveals different behavior. ClusterHC and eClusterHC exhibit nearly identical distributions, suggesting that randomization provides little additional benefit for this method. In contrast, eCSAW consistently outperforms CSAW, indicating that the ε-enhanced variant provides a clear improvement over the original selection mechanism.

4.3. Pairwise Wilcoxon comparison of the selection methods

To provide a direct statistical comparison among the eight solution selection methods, we perform pairwise Wilcoxon signed-rank tests on their final optimization performance. For each Pareto front approximation strategy, pairwise comparisons are performed over the 12 CEC2022 benchmark functions, and the resulting Win/Tie/Loss counts are aggregated across the three strategies, yielding a total of 36 comparison cases.

Table 1. Pairwise Win/Tie/Loss (W/T/L) comparison of the eight solution selection methods over 36 comparison cases. Each entry compares the row method against the column method using a Wilcoxon signed-rank test ($\alpha = 0.05$). Bold entries indicate an overall winning record ($W > L$).

	eShotgun	eCSAW	eClusterHC	ClusterHC	CSAW	qEI	qUCB	qKG
eShotgun	–	24/7/5	24/6/6	25/4/7	30/5/1	26/7/3	27/9/0	31/4/1
eCSAW	5/7/24	–	0/30/6	0/31/5	32/4/0	11/6/19	10/10/16	32/4/0
eClusterHC	6/6/24	6/30/0	–	2/34/0	33/3/0	10/6/20	12/8/16	32/4/0
ClusterHC	7/4/25	5/31/0	0/34/2	–	34/2/0	11/7/18	10/10/16	34/2/0
CSAW	1/5/30	0/4/32	0/3/33	0/2/34	–	0/7/29	5/4/27	28/5/3
qEI	3/7/26	19/6/11	20/6/10	18/7/11	29/7/0	–	3/24/9	30/6/0
qUCB	0/9/27	16/10/10	16/8/12	16/10/10	27/4/5	9/24/3	–	30/0/6
qKG	1/4/31	0/4/32	0/4/32	0/2/34	3/5/28	0/6/30	6/0/30	–

Table 1 summarizes the pairwise comparisons over the 36 comparison cases. For each comparison case, the final best objective values from the 30 independent runs are compared using a Wilcoxon signed-rank test at the 5% significance level. Each entry reports the numbers of Wins/Ties/Losses (W/T/L), where a win indicates that the row method significantly outperforms the column method, a loss indicates the opposite, and a tie indicates no significant difference.

Several clear patterns emerge. First, eShotgun achieves a positive win–loss record against every competing method. It records 24/7/5, 24/6/6, and 25/4/7 win/tie/loss records against eCSAW,

eClusterHC, and ClusterHC, respectively, demonstrating clear superiority even over its closest competitors. Against qEI, qUCB, CSAW, and qKG, the margins become even larger, achieving 26/7/3, 27/9/0, 30/5/1, and 31/4/1 records, respectively. These results provide strong statistical evidence that eShotgun consistently delivers the best overall optimization performance across the benchmark suite.

The remaining Pareto-based solution selection methods exhibit mixed pairwise results. ClusterHC, eClusterHC, and eCSAW consistently outperform CSAW and qKG, while producing many ties and balanced win–loss records when compared with one another. This indicates that their performance differences are generally modest.

Among the acquisition-function baselines, qEI and qUCB perform more competitively than suggested by the convergence curves alone. Both achieve positive win–loss records against several Pareto-based solution selection methods, although they are consistently outperformed by eShotgun. In contrast, qKG records negative win–loss balances against nearly every competing method, confirming its comparatively weak performance.

4.4. Critical-difference comparison

To complement the pairwise Wilcoxon comparisons, we compare the eight solution selection methods simultaneously using the Friedman ranking procedure followed by the Nemenyi post-hoc test recommended by Demšar [19]. Each method is ranked on every CEC2022 benchmark instance according to its final optimization performance, and the average Friedman ranks are subsequently compared. In the CD diagram, methods connected by a horizontal bar cannot be distinguished statistically at the 5% significance level.

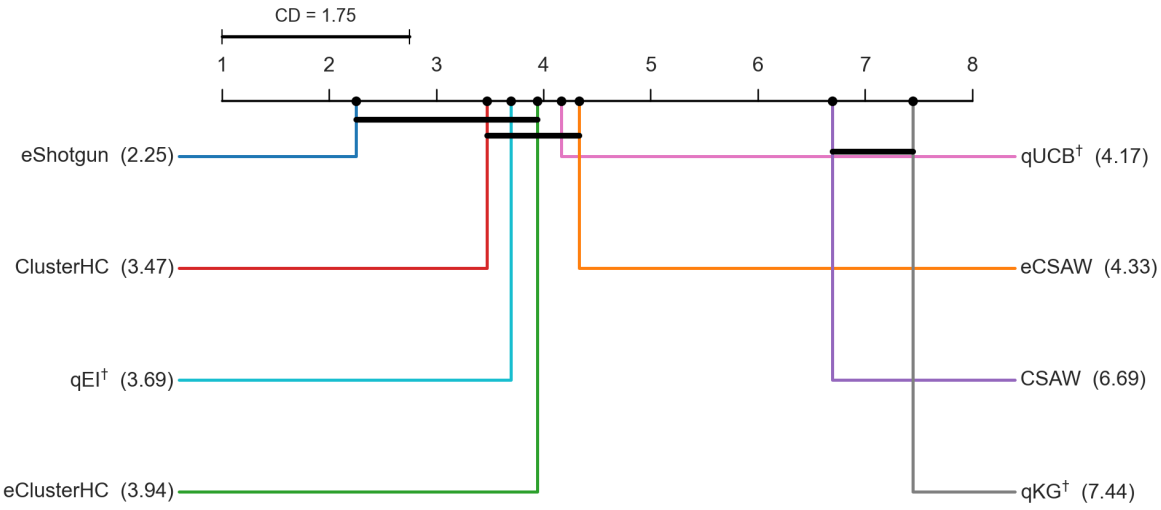


Figure 4. Critical-difference diagram of the average Friedman ranks for the eight solution selection methods, computed over the 36 CEC2022 benchmark instances (12 benchmark functions × 3 Pareto front approximation strategies). Lower average ranks indicate better optimization performance. Methods connected by a horizontal bar are not significantly different according to the Nemenyi post-hoc test at the 5% significance level (CD = 1.75). The dagger (†) denotes acquisition-function baselines.

Figure 4 shows that, unlike the pairwise Wilcoxon tests, which compare one pair of methods at a time, the Nemenyi post-hoc test simultaneously compares all methods and is therefore more conservative, resulting in broader groups of statistically indistinguishable methods. Nevertheless, the overall ranking is consistent with the preceding convergence curves, box plots, and pairwise Wilcoxon analysis: eShotgun achieves the best average rank (2.25), ClusterHC (3.47), qEI (3.69), eClusterHC (3.94), qUCB (4.17), and eCSAW (4.33) form a competitive middle group, whereas CSAW (6.69) and qKG (7.44) consistently rank among the weakest methods.

4.5. Batch diversity and exploration analysis

To better understand the search behavior of the Pareto-based solution selection methods, we examine two complementary diagnostics. The first measures the diversity within each selected batch, whereas the second measures the novelty of the selected batch relative to the previously evaluated archive.

Let $B = \{x_1, \ldots, x_q\}$ denote a selected batch of $q = 4$ points in the normalized search domain $[0, 1]^D$, and let $\mathcal{H}$ denote the set of points evaluated before the current batch.

The within-batch diversity is measured using the log-determinant determinantal point process (DPP) score [36, 39],

$$D_{\mathrm{dpp}}(B) = \log\det K, \qquad K_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\ell^2}\right), \qquad \ell = 0.3\sqrt{D}, \tag{4.3}$$

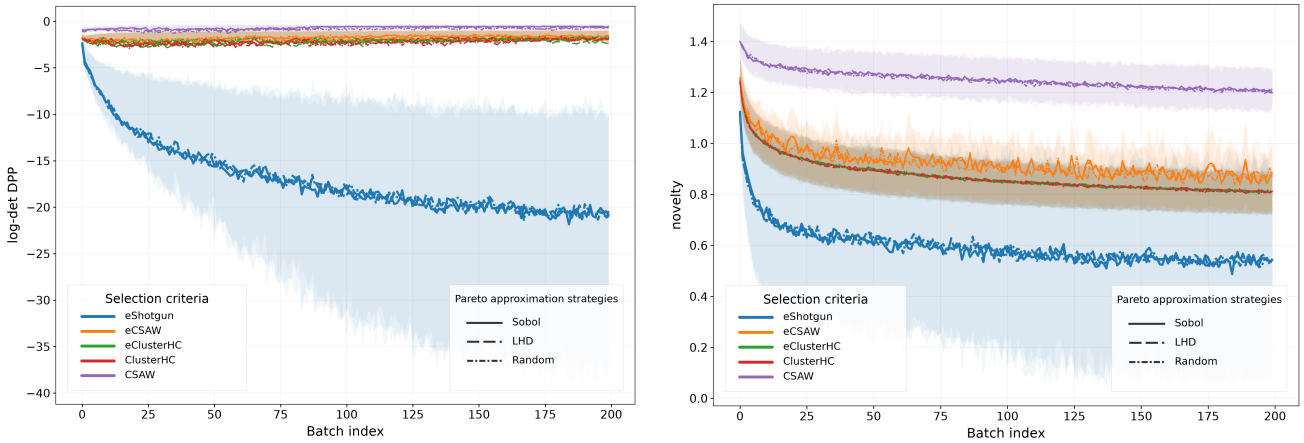
where K is the Gaussian kernel matrix. Larger values of D_{dpp} indicate greater within-batch diversity, whereas smaller values indicate that the selected points are more concentrated.

To quantify exploration relative to the search history, we compute the average novelty score adapted from the novelty search framework [41],

$$D_{\mathrm{nov}}(B) = \frac{1}{q}\sum_{i=1}^{q} \min_{x' \in \mathcal{H}} \|x_i - x'\|. \tag{4.4}$$

Large values of D_{nov} indicate that the selected batch explores regions far from the previously evaluated archive, whereas small values correspond to more local exploration.

Figure 5 presents the average diversity and novelty measures over 30 independent runs and all CEC2022 benchmark functions. The results reveal that the exploration behavior is strongly governed by the solution selection method.



(a) Log-det DPP diversity

(b) Archive-relative novelty

Figure 5. Mechanistic analysis of the Pareto-based solution selection methods. (a) The log-determinant DPP score measures within-batch diversity in kernel space, whereas (b) archive-relative novelty measures the average distance from the selected batch to the previously evaluated archive. Results are averaged over 30 independent runs and all CEC2022 benchmark functions.

The most distinctive behavior is observed for eShotgun, which consistently exhibits the lowest D_{dpp} and D_{nov} throughout the optimization process. This indicates that its selected batches are both internally concentrated and remain close to the previously evaluated archive. This behavior follows directly from its selection mechanism: only the first point is selected using the ε-greedy rule, whereas the remaining batch points are generated by local Gaussian sampling around this anchor point. Consequently, eShotgun deliberately emphasizes local exploitation over global exploration.

Interestingly, eShotgun also exhibits substantially greater variability across independent runs than the other Pareto-based solution selection methods, as reflected by the much wider shaded region in Figure 5. This behavior follows directly from the ε-greedy selection mechanism. With probability $1 - \varepsilon$, the anchor point is selected greedily by minimizing the predictive mean, producing tightly clustered batches with low diversity and novelty. With probability ε, however, the anchor point is selected from the Pareto set, after which the remaining batch points are generated by local Gaussian sampling around this exploratory anchor. These occasional exploratory batches exhibit substantially higher diversity and novelty, resulting in low average diversity together with greater variability across independent runs.

In contrast, the remaining methods exhibit more exploratory behavior, but with different degrees of diversity. In terms of within-batch diversity, CSAW consistently achieves the largest D_{dpp}, whereas ClusterHC, eClusterHC, and eCSAW produce nearly indistinguishable diversity levels. For archive-relative novelty, CSAW again explores farthest from the existing archive, followed by eCSAW, whereas ClusterHC and eClusterHC remain nearly indistinguishable. Together, these complementary diagnostics provide a mechanistic explanation for the different exploration–exploitation behaviors of the Pareto-based solution selection methods and help explain the superior optimization performance of eShotgun observed in the previous sections.

4.6. Sensitivity analysis of the candidate set size

Having examined the optimization performance and search behavior of the selection methods, we now investigate the effect of the candidate set size used for Pareto front approximation.

We compare four candidate set sizes,

$$N = \{100D,\ 1000D,\ 10000D,\ 100000D\},$$

where D denotes the problem dimension. Approximation quality is evaluated using the average Pareto front cardinality ($|P|$) and hypervolume (HV), while computational cost is measured by the Pareto front approximation runtime, from candidate set generation through Pareto front construction, including Gaussian process prediction and non-dominated sorting, but excluding Gaussian process training.

Table 2 summarizes the results averaged over five Gaussian process training set sizes ($n_{\text{obs}} = 100, \ldots, 500$), three Pareto front approximation strategies, twelve CEC2022 benchmark functions, and 30 independent runs.

Table 2. Average Pareto front cardinality, hypervolume, and Pareto front approximation runtimes for different candidate set sizes, averaged over five Gaussian process training set sizes and the three Pareto front approximation strategies.

Candidate size	Avg. $\lvert P\rvert$	Avg. HV	Avg. Runtime (s)
$100D$	11.5	0.816	0.024
$1000D$	17.2	0.944	0.221
$10000D$	25.0	0.983	2.16
$100000D$	40.0	0.998	21.9

As expected, increasing the candidate set size consistently improves the Pareto front approximation. The average Pareto front cardinality increases from 11.5 to 40.0 and the average hypervolume from 0.816 to 0.998 as N increases from $100D$ to $100000D$. However, these gains are accompanied by a substantial increase in computational cost, with the average runtime rising from 0.024 to 21.9 seconds.

The largest improvement is obtained by increasing the candidate set size from $100D$ to $1000D$, where the average hypervolume increases from 0.816 to 0.944 while the runtime remains below 0.25 seconds. Beyond $1000D$, each ten-fold increase in candidate set size yields only modest improvements in hypervolume (0.944 to 0.983 and then 0.998), whereas the runtime increases by approximately an order of magnitude.

For comparison, NSGA-II with a population size of 100 evolved for 100 generations achieves an average hypervolume of 1.0711 under the same experimental setting. Consequently, $N = 1000D$ provides the best trade-off between approximation quality and computational efficiency and is therefore used throughout this paper.

Detailed results for each Gaussian process training set size and Pareto front approximation strategy are provided in Appendix C (Tables C1 and C2), where all three approximation strategies exhibit consistent trends.

4.7. Real-world optimization benchmark

To further evaluate the practical applicability of the BOPS framework, we consider the *push4* robotic manipulation benchmark [16, 68], a widely used expensive black-box optimization problem

for Bayesian optimization, where a robot pushes an object toward a target location. The four decision variables specify the robot's initial position and pushing direction, and the objective is to minimize the Euclidean distance between the final object position and the target.

The same experimental protocol described in Section 4.1 is adopted. Each method is allocated a budget of 800 function evaluations and repeated over 30 independent runs, with the candidate set size fixed at $N = 1000D$. The complete numerical results are reported in Tables D1–D3, where the last rows correspond to the *push4* benchmark.

Figures 6 and 7 compare the convergence behavior and final optimization performance of the five Pareto-based solution selection methods and the three acquisition-function baselines on the *push4* robotic pushing benchmark. The Pareto-based approaches remain competitive on this practical expensive optimization problem, demonstrating that the observations from the synthetic benchmark suite also extend to a real-world application.

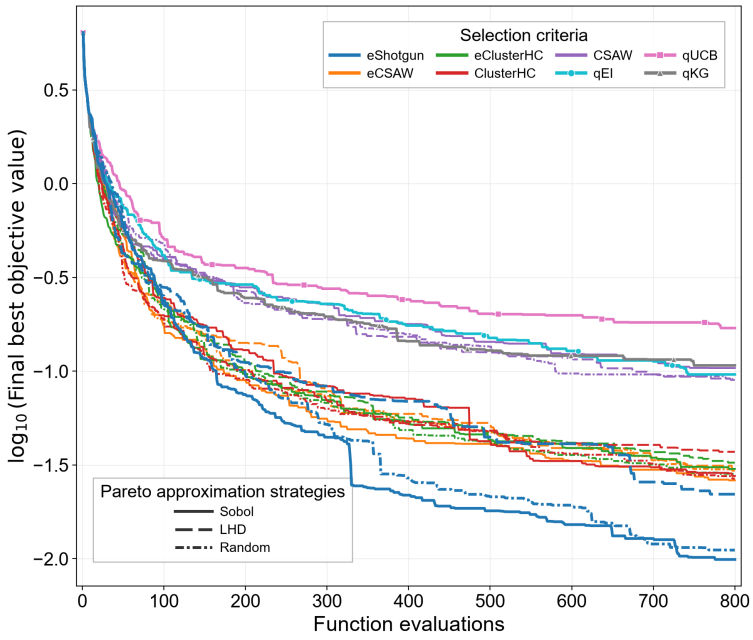


Figure 6. Convergence curves of the five Pareto-based solution selection methods under different Pareto front approximation strategies, together with the three acquisition-function baselines, on the *push4* benchmark. The curves show the mean best objective value over 30 independent runs.

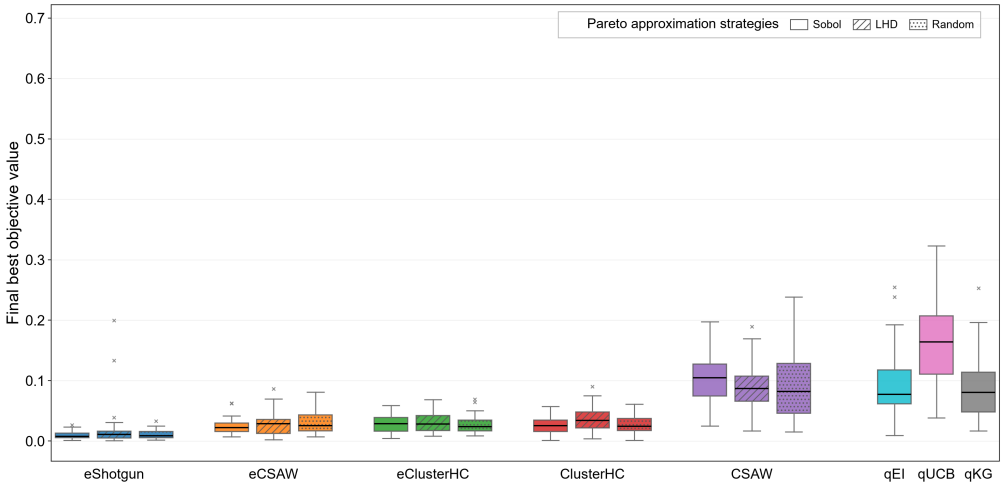


Figure 7. Distribution of the final best objective values over 30 independent runs on the *push4* benchmark.

Again, eShotgun achieves the best performance, exhibiting both the fastest convergence and the lowest final objective values. Among the Pareto-based solution selection methods, ClusterHC, eClusterHC, and eCSAW exhibit similar performance, with all three consistently outperforming CSAW. The boxplots further indicate that these methods exhibit relatively small variability across independent runs, suggesting stable optimization behavior. In contrast, all three acquisition-function baselines converge more slowly and terminate at higher objective values, exhibiting performance comparable to or worse than CSAW.

5. Conclusions

In this paper, we presented a unified framework for studying Pareto-based batch Bayesian optimization by explicitly separating Pareto front approximation from solution selection. This framework enabled a systematic comparison of five Pareto-based solution selection methods under a common experimental setting, together with three sampling strategies for Pareto front approximation.

Extensive experiments on the CEC2022 benchmark suite and the *push4* robotic manipulation benchmark demonstrated that solution selection plays a more important role than Pareto approximation once a sufficiently accurate Pareto set has been obtained. Among the deterministic solution selection methods, ClusterHC consistently achieved better overall performance than CSAW. The results also showed that incorporating a stochastic component into CSAW substantially improves its performance, making it competitive with the top-performing deterministic solution selection methods.

More importantly, the empirical results indicated that solution selection remains a critical yet relatively unexplored component of Pareto-based batch Bayesian optimization. The noticeable performance differences among the evaluated selection strategies suggest that there remains considerable opportunity for developing more effective Pareto-based solution selection methods.

One limitation of the present study is the use of Gaussian process surrogate models, whose cubic computational complexity limits their scalability as the number of observations increases. Extending the BOPS framework to scalable surrogate models [4, 65], such as sparse Gaussian processes,

constitutes an important direction for future research. Furthermore, the present study focused on unconstrained deterministic optimization problems of moderate dimensionality. Future work will investigate the BOPS framework in noisy and constrained Bayesian optimization settings, on higher-dimensional and more computationally demanding real-world optimization problems, and through the development of adaptive Pareto front approximation and solution selection strategies. From a practical perspective, our results suggest that practitioners should place greater emphasis on the design of Pareto-based solution selection strategies than on the particular Pareto approximation method once a sufficiently accurate Pareto set has been obtained. We hope that the unified framework and comprehensive empirical study presented in this work will facilitate future research on Pareto-based batch Bayesian optimization and contribute to the development of more effective solution selection strategies.

Author contributions

K. Chaiyotha: Conceptualization, Methodology, Validation, Formal analysis, Investigation, Data curation, Writing–original draft, Writing–review and editing, Visualization, Software; T. Krityakierne: Conceptualization, Methodology, Validation, Formal analysis, Investigation, Data curation, Writing–original draft, Writing–review and editing, Visualization, Supervision, Resources. All authors have read and agreed to the published version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The first author gratefully acknowledges the Science Achievement Scholarship of Thailand (SAST) for the financial support during his graduate studies.

Conflict of interest

The authors declare no conflict of interest.

References

1. A. S. Alvi, B. B. Ru, J.-P. Calliess, S. J. Roberts, M. A. Osborne, Asynchronous batch Bayesian optimisation with improved local penalisation, *Proceedings of the 36th International Conference on Machine Learning*, PMLR, **97** (2019), 253–262.
2. M. Awad, R. Khanna, Support vector regression, In: *Efficient learning machines*, Berkeley: Apress, 2015, 67–80. http://doi.org/10.1007/978-1-4302-5990-9_4
3. F. Berkenkamp, A. Krause, A. P. Schoellig, Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics, *Mach. Learn.*, **112** (2023), 3713–3747. http://doi.org/10.1007/s10994-021-06019-1

4. M. Binois, N. Wycoff, A survey on high-dimensional gaussian process modeling with application to bayesian optimization, *ACM Transactions on Evolutionary Learning and Optimization*, **2** (2022), 1–26. http://doi.org/10.1145/3545611

5. J. Blank, K. Deb, pymoo: Multi-objective optimization in Python, *IEEE Access*, **8** (2020), 89497–89509. http://doi.org/10.1109/ACCESS.2020.2990567

6. A. D. Bull, Convergence rates of efficient global optimization algorithms, *J. Mach. Learn. Res.*, **12** (2011), 2879–2904.

7. A. Candelieri, A gentle introduction to bayesian optimization, *2021 Winter Simulation Conference (WSC)*, Phoenix, AZ, USA, 2021, 1–16. http://doi.org/10.1109/WSC52266.2021.9715413

8. A. Candelieri, A. Ponti, I. Giordani, F. Archetti, On the use of Wasserstein distance in the distributional analysis of human decision making under uncertainty, *Ann. Math. Artif. Intell.*, **91** (2023), 217–238. http://doi.org/10.1007/s10472-022-09807-0

9. K. Chaiyotha, T. Krityakierne, A comparative study of infill sampling criteria for computationally expensive constrained optimization problems, *Symmetry*, **12** (2020), 1631. http://doi.org/10.3390/sym12101631

10. S. Chen, C. F. N. Cowan, P. M. Grant, Orthogonal least squares learning algorithm for radial basis function networks, *IEEE T. Neural Networ.*, **2** (1991), 302–309. http://doi.org/10.1109/72.80341

11. I. Couckuyt, D. Deschrijver, T. Dhaene, Fast calculation of multiobjective probability of improvement and expected improvement criteria for pareto optimization, *J. Glob. Optim.*, **60** (2014), 575–594. http://doi.org/10.1007/s10898-013-0118-2

12. D. D. Cox, S. John, A statistical method for global optimization, *1992 IEEE International Conference on Systems, Man, and Cybernetics*, Chicago, IL, USA, 1992, 1241–1246. http://doi.org/10.1109/ICSMC.1992.271617

13. R. Dai, J. Jie, Z. Wang, H. Zheng, W. L. Wang, Automated surrogate-assisted particle swarm optimizer with an adaptive parental guidance strategy for expensive engineering optimization problems, *J. Comput. Des. Eng.*, **12** (2025), 145–183. http://doi.org/10.1093/jcde/qwaf023

14. G. Damblin, M. Couplet, B. Iooss, Numerical studies of space-filling designs: Optimization of latin hypercube samples and subprojection properties, *J. Simul.*, **7** (2013), 276–289. http://doi.org/10.1057/jos.2013.16

15. G. De Ath, R. M. Everson, J. E. Fieldsend, Asynchronous ε-greedy Bayesian optimisation, *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, PMLR, **161** (2021), 578–588.

16. G. De Ath, R. M. Everson, J. E. Fieldsend, A. A. M. Rahat, ε-shotgun: ε-greedy batch bayesian optimisation, *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, New York: Association for Computing Machinery, 2020, 787–795. http://doi.org/10.1145/3377930.3390154

17. G. De Ath, R. M. Everson, A. A. M. Rahat, J. E. Fieldsend, Greed is good: Exploration and exploitation trade-offs in bayesian optimisation, *ACM Trans. Evol. Learn. Optim.*, **1** (2021), 1–22. http://doi.org/10.1145/3425501

18. G. De Ath, J. E. Fieldsend, R. M. Everson, What do you mean? The role of the mean function in bayesian optimisation, *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, New York: Association for Computing Machinery, 2020, 1623–1631. http://doi.org/10.1145/3377929.3398118

19. J. Demšar, Statistical comparisons of classifiers over multiple data sets, *J. Mach. Learn. Res.*, **7** (2006), 1–30.

20. B. Do, T. Adebiyi, R. D. Zhang, Epsilon-greedy thompson sampling to Bayesian optimization, *J. Comput. Inf. Sci. Eng.*, **24** (2024), 121006. http://doi.org/10.1115/1.4066858

21. L. Du, R. B. Gao, P. N. Suganthan, D. Z. W. Wang, Bayesian optimization based dynamic ensemble for time series forecasting, *Inform. Sciences*, **591** (2022), 155–175. http://doi.org/10.1016/j.ins.2022.01.010

22. D. Eriksson, M. Poloczek, Scalable constrained bayesian optimization, *International Conference on Artificial Intelligence and Statistics*, PMLR, **130** (2021), 730–738.

23. M. Ester, H.-P. Kriegel, J. Sander, X. W. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, Oregon: AAAI Press, 1996, 226–231.

24. P. I. Frazier, Bayesian optimization, *Recent Advances in Optimization and Modeling of Contemporary Problems*, **2018** 2018, 255–278. http://doi.org/10.1287/educ.2018.0188

25. M. A. Gelbart, J. Snoek, R. P. Adams, Bayesian optimization with unknown constraints, *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, Quebec City, Canada, 2014, 250–259.

26. *GPy: A Gaussian process framework in Python*, 2012–2024. Available from: `http://github.com/SheffieldML/GPy`.

27. F. Häse, M. Aldeghi, R. J. Hickman, L. M. Roch, A. Aspuru-Guzik, Gryffin: An algorithm for bayesian optimization of categorical variables informed by expert knowledge, *Appl. Phys. Rev.*, **8** (2021), 031406. http://doi.org/10.1063/5.0048164

28. C. L. He, Y. Zhang, D. W. Gong, X. F. Ji, A review of surrogate-assisted evolutionary algorithms for expensive optimization problems, *Expert Syst. Appl.*, **217** (2023), 119495. http://doi.org/10.1016/j.eswa.2022.119495

29. J. D. Huang, D. W. Zhan, An adaptive dropout approach for high-dimensional Bayesian optimization, *Optim. Eng.*, **2026** (2026), 1–24. http://doi.org/10.1007/s11081-026-10087-4

30. C. Jiang, M. Q. Li, Multi-objectivising acquisition functions in bayesian optimisation, *ACM Transactions on Evolutionary Learning and Optimization*, **5** (2025), 1–33. http://doi.org/10.1145/3716504

31. C. Jiang, M. Q. Li, Trading off quality and uncertainty through multi-objective optimisation in batch bayesian optimisation, *Proceedings of the AAAI Conference on Artificial Intelligence*, **39** (2025), 27027–27035. http://doi.org/10.1609/aaai.v39i25.34909

32. P. Y. Jiang, Y. S. Cheng, J. X. Yi, J. Liu, An efficient constrained global optimization algorithm with a clustering-assisted multiobjective infill criterion using gaussian process regression for expensive problems, *Inform. Sciences*, **569** (2021), 728–745. http://doi.org/10.1016/j.ins.2021.05.015

33. R. W. Jiao, S. Y. Zeng, C. H. Li, Y. H. Jiang, Y. C. Jin, A complete expected improvement criterion for gaussian process assisted highly constrained expensive optimization, *Inform. Sciences*, **471** (2019), 80–96. http://doi.org/10.1016/j.ins.2018.09.003

34. S. Joe, F. Y. Kuo, Constructing sobol sequences with better two-dimensional projections, *SIAM J. Sci. Comput.*, **30** (2008), 2635–2654. http://doi.org/10.1137/070709359

35. D. R. Jones, M. Schonlau, W. J. Welch, Efficient global optimization of expensive black-box functions, *J. Global Optim.*, **13** (1998), 455–492. http://doi.org/10.1023/A:1008306431147

36. T. Kathuria, A. Deshpande, P. Kohli, Batched Gaussian process bandit optimization via determinantal point processes, *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Red Hook: Curran Associates Inc., 2016, 4213–4221.

37. T. Krityakierne, D. Baowan, Aggregated GP-based optimization for contaminant source localization, *Operations Research Perspectives*, **7** (2020), 100151. http://doi.org/10.1016/j.orp.2020.100151

38. T. Krityakierne, D. Ginsbourger, Global optimization with sparse and local gaussian process models, *International Workshop on Machine Learning, Optimization and Big Data*, Cham: Springer, 2015, 185–196. http://doi.org/10.1007/978-3-319-27926-8_16

39. A. Kulesza, B. Taskar, Determinantal point processes for machine learning, *Found. Trends Mach. Le.*, **5** (2012), 123–286. http://doi.org/10.1561/2200000044

40. A. Kumar, K. V. Price, A. W. Mohamed, A. A. Hadi, P. N. Suganthan, *Problem Definitions and Evaluation Criteria for the 2022 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization*, Technical report, Nanyang Technological University, Singapore, 2021. Available from: `https://github.com/P-N-Suganthan/2022-SO-BO`.

41. J. Lehman, K. O. Stanley, Abandoning objectives: Evolution through the search for novelty alone, *Evol. Comput.*, **19** (2011), 189–223. http://doi.org/10.1162/EVCO_a_00025

42. B. Liu, Q. F. Zhang, G. G. E. Gielen, A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems, *IEEE T. Evolut. Comput.*, **18** (2014), 180–192. http://doi.org/10.1109/TEVC.2013.2248012

43. J. MacQueen, Some methods for classification and analysis of multivariate observations, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, **1** (1967), 281–297.

44. J. Mockus, V. Tiesis, A. Zilinskas, The application of Bayesian methods for seeking the extremum, In: *Towards global optimization*, Amsterdam: North-Holland, 1978, 117–129.

45. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, et al., Scikit-learn: Machine learning in Python, *J. Mach. Learn. Res.*, **12** (2011), 2825–2830.

46. V. Perrone, M. Donini, M. B. Zafar, R. Schmucker, K. Kenthapadi, C. Archambeau, Fair bayesian optimization, *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, New York: Association for Computing Machinery, 2021, 854–863. http://doi.org/10.1145/3461702.3462629

47. V. Picheny, A stepwise uncertainty reduction approach to constrained global optimization, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, PMLR, **33** (2014), 787–795.

48. G. Pirot, T. Krityakierne, D. Ginsbourger, P. Renard, Contaminant source localization via bayesian global optimization, *Hydrol. Earth Syst. Sci.*, **23** (2019), 351–369. http://doi.org/10.5194/hess-23-351-2019

49. pyDOE2 Contributors, *pyDOE2: Design of experiments for Python*, 2018–2024. Available from: `https://github.com/clicumu/pyDOE2`.

50. J. C. Qian, Y. S. Cheng, J. L. Zhang, J. Liu, D. W. Zhan, A parallel constrained efficient global optimization algorithm for expensive constrained optimization problems, *Eng. Optimiz.*, **53** (2021), 300–320. http://doi.org/10.1080/0305215X.2020.1722118

51. C. E. Rasmussen, C. K. I. Williams, *Gaussian processes for machine learning*, Cambridge: MIT Press, 2005. http://doi.org/10.7551/mitpress/3206.001.0001

52. F. Rehbach, M. Zaefferer, A. Fischbach, G. Rudolph, T. Bartz-Beielstein, Benchmark-driven configuration of a parallel model-based optimization algorithm, *IEEE T. Evolut. Comput.*, **26** (2022), 1365–1379. http://doi.org/10.1109/TEVC.2022.3163843

53. F. L. Richardson, G. De Ath, T. Chugh, Is greed still good in multi-objective bayesian optimisation, *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, New York: Association for Computing Machinery, 2024, 2103–2106. http://doi.org/10.1145/3638530.3664189

54. O. Roustant, D. Ginsbourger, Y. Deville, DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization, *J. Stat. Softw.*, **51** (2012), 1–55. http://doi.org/10.18637/jss.v051.i01

55. M. Schonlau, W. J. Welch, D. R. Jones, Global versus local search in constrained optimization of computer models, *IMS Lecture Notes Monogr. Ser.*, **34** (1998), 11–25. http://doi.org/10.1214/lnms/1215456182

56. E. Schulz, M. Speekenbrink, A. Krause, A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions, *J. Math. Psychol.*, **85** (2018), 1–16. http://doi.org/10.1016/j.jmp.2018.03.001

57. N. Srinivas, A. Krause, S. M. Kakade, M. W. Seeger, Gaussian process optimization in the bandit setting: No regret and experimental design, *Proceedings of the 27th International Conference on Machine Learning (ICML)*, Madison: Omnipress, 2010, 1015–1022.

58. N. V. Thieu, Opfunu: An open-source Python library for optimization benchmark functions, *Journal of Open Source Software*, **12** (2024), 8. http://doi.org/10.5334/jors.508

59. J. Ungredda, J. Branke, Bayesian optimisation for constrained problems, *ACM T. Model. Comput. S.*, **34** (2024), 1–26. http://doi.org/10.1145/3641544

60. N. S. Upadhye, R. Chowdhury, Constrained Bayesian optimization with lower confidence bound, *Technometrics*, **66** (2024), 561–574. http://doi.org/10.1080/00401706.2024.2336535

61. A. H. Victoria, G. Maragatham, Automatic tuning of hyperparameters using bayesian optimization, *Evol. Syst.*, **12** (2021), 217–223. http://doi.org/10.1007/s12530-020-09345-2

62. P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, et al., SciPy 1.0: Fundamental algorithms for scientific computing in Python, *Nat. Methods*, **17** (2020), 261–272. http://doi.org/10.1038/s41592-019-0686-2

63. H. Y. Wang, H. Xu, Y. Yuan, Z. Q. Zhang, An adaptive batch bayesian optimization approach for expensive multi-objective problems, *Inform. Sciences*, **611** (2022), 446–463. http://doi.org/10.1016/j.ins.2022.08.021

64. H. Y. Wang, H. Xu, Z. Q. Zhang, High-dimensional multi-objective bayesian optimization with block coordinate updates: Case studies in intelligent transportation system, *IEEE T. Intell. Transp.*, **25** (2024), 884–895. http://doi.org/10.1109/TITS.2023.3241069

65. L. W. Wang, S. Yerramilli, A. Iyer, D. Apley, P. Zhu, W. Chen, Scalable gaussian processes for data-driven design using big data with categorical factors, *J. Mech. Des.*, **144** (2022), 021703. http://doi.org/10.1115/1.4052221

66. W. Z. Wang, H.-L. Liu, K. C. Tan, A surrogate-assisted differential evolution algorithm for high-dimensional expensive optimization problems, *IEEE T. Cybernetics*, **53** (2023), 2685–2697. http://doi.org/10.1109/TCYB.2022.3175533

67. Z. Y. Wang, G. P. Rangaiah, Application and analysis of methods for selecting an optimal solution from the pareto-optimal front obtained by multiobjective optimization, *Ind. Eng. Chem. Res.*, **56** (2017), 560–574. http://doi.org/10.1021/acs.iecr.6b03453

68. Z. Wang, S. Jegelka, Max-value entropy search for efficient Bayesian optimization, *Proceedings of the 34th International Conference on Machine Learning*, PMLR, 2017, 3627–3635.

69. C. K. I. Williams, C. E. Rasmussen, Gaussian processes for regression, *Advances in Neural Information Processing Systems*, **8** (1995), 1–7.

70. J. T. Wilson, F. Hutter, M. P. Deisenroth, Maximizing acquisition functions for Bayesian optimization, *Advances in Neural Information Processing Systems*, **31** (2018), 1–12.

71. J. Wu, P. I. Frazier, The parallel knowledge gradient method for batch Bayesian optimization, *Advances in Neural Information Processing Systems*, **29** (2016), 1–9.

72. W. J. Xu, Y. N. Jiang, B. Svetozarevic, C. N. Jones, Constrained efficient global optimization of expensive black-box functions, *Proceedings of the 40th International Conference on Machine Learning*, Honolulu, Hawaii, USA, **202** (2023), 38485–38498.

73. Y. Zeng, Y. S. Cheng, J. Liu, An efficient global optimization algorithm for expensive constrained black-box problems by reducing candidate infilling region, *Inform. Sciences*, **609** (2022), 1641–1669. http://doi.org/10.1016/j.ins.2022.07.162

74. D. W. Zhan, Expected coordinate improvement for high-dimensional Bayesian optimization, *Swarm Evol. Comput.*, **91** (2024), 101745. http://doi.org/10.1016/j.swevo.2024.101745

75. Y. J. Zhang, Y. Z. Wang, J. P. Wang, Objective attributes weights determining based on shannon information entropy in hesitant fuzzy multiple attribute decision making, *Math. Probl. Eng.*, **2014** (2014), 463930. http://doi.org/10.1155/2014/463930

Appendix

The appendices provide supplementary information to support the reproducibility and completeness of this study. Appendix A summarizes the CEC2022 benchmark functions. Appendix B describes the software implementation. Appendix C presents additional results for the candidate-set-size sensitivity analysis. Appendix D reports the complete numerical results for all experiments. Appendix E lists the experimental parameter settings for all methods evaluated in this study, including the baseline algorithms.

A. CEC2022 test problems

The CEC2022 benchmark suite consists of one unimodal function (F1), four basic multimodal functions (F2–F5), three hybrid functions (F6–F8), and four composition functions (F9–F12). The search domain is $\mathcal{X} = [-100, 100]^D$. The corresponding global optimum value for each test function is reported in the last column.

Table A1. Summary of the CEC2022 test functions [40].

Category	No.	Functions	$\mathbf{f}^*$
Unimodal Function	1	Shifted and full rotated Zakharov function	300
Basic Functions	2	Shifted and full rotated Rosenbrock's function	400
	3	Shifted and full rotated Expanded Schaffer's f_6 function	600
	4	Shifted and full rotated Non-Continuous Rastrigin's function	800
	5	Shifted and full rotated Levy function	900
Hybrid Functions	6	Hybrid Function 1 ($N = 3$)	1800
	7	Hybrid Function 2 ($N = 6$)	2000
	8	Hybrid Function 3 ($N = 5$)	2200
Composition Functions	9	Composition Function 1 ($N = 5$)	2300
	10	Composition Function 2 ($N = 4$)	2400
	11	Composition Function 3 ($N = 5$)	2600
	12	Composition Function 4 ($N = 6$)	2700

B. Software environment

All experiments were implemented in Python using the following software packages.

Batch Bayesian optimization. The implementation builds upon the `eshotgun` package [16], which provides the core batch BO loop, including ε-greedy selection and greedy batch construction. The Pareto-based selection variants are implemented within this framework.

Gaussian process modeling. Surrogate models are constructed using `GPy` [26], with a Matérn 5/2 kernel.

Pareto approximation. Non-dominated sorting is performed using `pymoo` [5].

Sampling. Sobol and Latin hypercube designs are generated via `scipy.stats.qmc` [62], with additional support from `pyDOE2` [49].

Clustering and preprocessing. DBSCAN clustering and min–max normalization are implemented using `scikit-learn` [45].

Benchmark suite. The CEC2022 benchmark functions are implemented using the `opfunu` package [58].

qEI. This jointly optimizes the q-point expected improvement acquisition function using Monte Carlo estimation together with the reparameterization trick [70].

qUCB. This extends the classical GP-UCB acquisition function to the batch setting by maximizing a sample-average approximation of the joint upper confidence bound over the q candidate points [57,70].

qKG. This selects a batch of q points by maximizing the expected increase in the optimal posterior mean after observing the batch, where the expectation is approximated by averaging over hypothetical batch outcomes and the corresponding posterior updates [71].

ε-Shotgun. This selects the first batch point using an ε-greedy rule and then generates the remaining $q-1$ points by Gaussian sampling around it using an adaptive sampling radius determined by the surrogate model [16].

C. Additional results for candidate-set-size sensitivity analysis

D. Complete numerical results

Table C1. Average cardinality and hypervolume per strategy, separated by N_{label} and n_{obs}.

		Cardinality ($\lvert P\rvert$)			Hypervolume (HV)		
N_{label}	n_{obs}	LHD	Random	Sobol	LHD	Random	Sobol
100D	100	11.80	11.86	11.83	0.8130	0.8190	0.8185
	200	11.55	11.49	11.71	0.8205	0.8220	0.8244
	300	11.31	11.44	11.26	0.8156	0.8117	0.8206
	400	11.20	11.43	11.42	0.8068	0.8006	0.8109
	500	11.22	11.35	11.25	0.8218	0.8129	0.8188
1000D	100	19.31	18.79	18.90	0.9413	0.9404	0.9408
	200	17.56	17.80	17.48	0.9460	0.9465	0.9466
	300	17.04	17.27	17.00	0.9444	0.9455	0.9454
	400	17.05	17.14	17.02	0.9346	0.9360	0.9421
	500	16.84	16.55	16.63	0.9425	0.9459	0.9447
10000D	100	30.78	30.47	30.32	0.9818	0.9834	0.9844
	200	26.73	26.60	26.14	0.9864	0.9864	0.9871
	300	25.16	24.81	25.10	0.9811	0.9886	0.9769
	400	24.52	24.31	24.29	0.9761	0.9741	0.9782
	500	24.22	23.87	23.88	0.9831	0.9802	0.9853
100000D	100	51.23	50.73	50.72	0.9975	1.0002	0.9976
	200	41.89	42.03	42.05	0.9988	0.9983	0.9988
	300	38.46	38.41	37.94	1.0029	0.9973	0.9988
	400	36.74	36.62	36.69	0.9917	0.9980	0.9925
	500	35.33	35.32	35.38	0.9998	0.9934	1.0020

Table C2. Average Pareto selection runtime in seconds per strategy.

		Selection time (s)		
N_{label}	n_{obs}	LHD	Random	Sobol
100D	100	0.0090	0.0078	0.0131
	200	0.0149	0.0137	0.0186
	300	0.0224	0.0212	0.0263
	400	0.0295	0.0288	0.0331
	500	0.0392	0.0376	0.0429
1000D	100	0.0840	0.0753	0.0759
	200	0.1450	0.1363	0.1358
	300	0.2216	0.2148	0.2096
	400	0.2935	0.2837	0.2804
	500	0.3971	0.3856	0.3799
10000D	100	0.7461	0.6705	0.7402
	200	1.3456	1.2791	1.3879
	300	2.0735	2.0748	2.1348
	400	2.7977	2.8013	2.9249
	500	3.7292	3.6881	3.8200
100000D	100	7.7919	6.9525	6.8401
	200	14.1748	13.5458	13.3001
	300	21.9962	21.1686	20.9117
	400	28.7789	28.3499	28.3507
	500	38.8540	38.1818	37.6456

Table D1. Results of the BOPS framework on the CEC2022 benchmark suite and the real-world *push4* benchmark using the Sobol Pareto front approximation strategy with a budget of 800 function evaluations. All results are based on 30 independent runs. Smaller values are better, and the best result for each performance measure is shown in bold.

Function	Measures	eShotgun	eCSAW	eClusterHC	ClusterHC	CSAW	qEI	qUCB	qKG
F1	Best	2.7022e+04	3.3741e+04	**2.0686e+04**	2.0747e+04	4.1847e+04	4.0625e+04	3.8067e+04	3.0797e+04
	Mean	**4.2433e+04**	4.6462e+04	4.4489e+04	4.3526e+04	5.5294e+04	5.4671e+04	6.3118e+04	5.0986e+04
	Sd	6.9237e+03	6.5026e+03	6.8573e+03	7.6127e+03	8.0486e+03	9.4912e+03	1.3668e+04	8.7607e+03
	Median	4.2459e+04	4.4852e+04	4.5592e+04	4.5198e+04	5.3358e+04	5.3672e+04	6.3187e+04	5.2342e+04
F2	Best	**4.9587e+02**	9.8459e+02	1.0135e+03	8.2626e+02	1.3908e+03	7.0659e+02	7.0101e+02	1.8893e+03
	Mean	**5.5259e+02**	1.3208e+03	1.2812e+03	1.2946e+03	1.7960e+03	9.7951e+02	1.0163e+03	2.9608e+03
	Sd	7.2188e+01	1.4450e+02	1.2880e+02	1.8018e+02	3.2006e+02	1.7379e+02	1.6607e+02	6.1233e+02
	Median	5.3641e+02	1.3266e+03	1.3049e+03	1.2958e+03	1.7260e+03	9.0698e+02	1.0085e+03	2.9872e+03
F3	Best	**6.0000e+02**	6.0056e+02	6.0032e+02	6.0043e+02	6.0064e+02	6.0021e+02	6.0002e+02	6.0132e+02
	Mean	**6.0000e+02**	6.0073e+02	6.0069e+02	6.0067e+02	6.0093e+02	6.0057e+02	6.0003e+02	6.0172e+02
	Sd	1.8226e-04	7.1748e-02	9.8843e-02	1.0062e-01	1.3179e-01	1.5339e-01	7.5118e-03	2.0335e-01
	Median	6.0000e+02	6.0075e+02	6.0071e+02	6.0069e+02	6.0093e+02	6.0059e+02	6.0003e+02	6.0171e+02
F4	Best	**8.3986e+02**	1.0248e+03	1.0549e+03	1.0621e+03	1.0747e+03	1.0374e+03	9.9687e+02	1.1132e+03
	Mean	**8.9856e+02**	1.1084e+03	1.1064e+03	1.1097e+03	1.1489e+03	1.0963e+03	1.1092e+03	1.1832e+03
	Sd	3.1679e+01	3.4344e+01	2.4696e+01	2.8167e+01	4.1341e+01	2.8906e+01	4.1378e+01	3.2324e+01
	Median	9.0011e+02	1.1168e+03	1.1068e+03	1.1133e+03	1.1535e+03	1.0952e+03	1.1245e+03	1.1838e+03
F5	Best	**9.0027e+02**	9.0555e+02	9.0605e+02	9.0499e+02	9.0516e+02	9.0211e+02	9.0105e+02	9.0547e+02
	Mean	**9.0144e+02**	9.0758e+02	9.0748e+02	9.0704e+02	9.0960e+02	9.0318e+02	9.0193e+02	9.1319e+02
	Sd	1.4760e+00	1.0164e+00	8.7673e-01	9.8425e-01	2.0793e+00	5.9971e-01	5.9758e-01	3.3818e+00
	Median	9.0107e+02	9.0730e+02	9.0736e+02	9.0719e+02	9.0993e+02	9.0319e+02	9.0185e+02	9.1361e+02
F6	Best	**1.4144e+05**	1.6552e+08	1.8998e+08	1.5231e+08	1.9708e+08	9.9413e+06	2.0229e+07	2.5031e+09
	Mean	**6.0429e+05**	4.0114e+08	4.1809e+08	3.9049e+08	5.5409e+08	1.4709e+08	4.9653e+07	4.2345e+09
	Sd	2.4741e+05	1.2675e+08	1.3977e+08	1.1487e+08	1.9402e+08	1.1070e+08	1.5813e+07	8.4562e+08
	Median	5.7908e+05	4.0427e+08	3.8723e+08	3.7781e+08	5.4395e+08	1.1599e+08	4.7070e+07	4.3427e+09
F7	Best	**2.1350e+03**	2.6192e+03	2.8252e+03	2.7219e+03	2.6119e+03	2.5569e+03	2.3539e+03	3.6736e+03
	Mean	**2.3124e+03**	3.2268e+03	3.1752e+03	3.1681e+03	3.7284e+03	2.9706e+03	3.0289e+03	5.5424e+03
	Sd	1.0712e+02	3.3410e+02	2.2560e+02	2.4184e+02	4.8354e+02	2.2157e+02	3.2195e+02	8.9685e+02
	Median	2.2881e+03	3.1935e+03	3.1253e+03	3.1659e+03	3.7128e+03	2.9432e+03	3.1365e+03	5.4306e+03
F8	Best	1.5712e+07	7.5049e+06	1.0256e+05	**8.0025e+03**	4.5883e+07	5.2276e+04	3.0795e+07	1.5952e+07
	Mean	1.3704e+11	6.8828e+10	4.3876e+10	**3.5882e+10**	2.3943e+11	5.8534e+10	7.4086e+10	3.5425e+11
	Sd	1.9856e+11	1.4188e+11	1.0810e+11	4.6612e+10	3.4324e+11	9.9258e+10	1.0155e+11	5.7719e+11
	Median	7.2599e+10	2.4301e+10	6.2156e+09	1.3183e+10	9.2594e+10	1.4498e+10	2.6237e+10	1.3853e+11
F9	Best	**2.4393e+03**	3.2285e+03	3.0139e+03	3.1310e+03	3.5382e+03	3.2721e+03	3.3567e+03	3.8741e+03
	Mean	**2.8973e+03**	3.5375e+03	3.4479e+03	3.4569e+03	3.9828e+03	3.6032e+03	3.5653e+03	5.7409e+03
	Sd	1.8447e+02	1.6646e+02	1.6551e+02	1.8926e+02	2.6072e+02	1.9097e+02	1.3892e+02	8.7618e+02
	Median	2.9319e+03	3.5299e+03	3.4368e+03	3.4834e+03	4.0099e+03	3.6035e+03	3.5687e+03	5.5617e+03
F10	Best	3.1415e+03	2.9477e+03	2.9490e+03	**2.9163e+03**	2.9632e+03	3.3573e+03	3.1676e+03	2.9463e+03
	Mean	5.2520e+03	**3.9464e+03**	4.1701e+03	4.0157e+03	4.1632e+03	4.6573e+03	4.8738e+03	4.2756e+03
	Sd	1.1314e+03	1.4517e+03	1.5164e+03	1.4166e+03	1.4745e+03	6.4570e+02	8.2154e+02	9.1346e+02
	Median	5.5682e+03	3.0113e+03	3.1999e+03	3.0149e+03	3.0888e+03	4.6074e+03	4.9589e+03	4.4192e+03
F11	Best	2.9254e+03	2.9088e+03	2.9370e+03	**2.8959e+03**	3.1147e+03	2.9342e+03	3.1783e+03	4.2223e+03
	Mean	3.7798e+03	3.7026e+03	**3.2742e+03**	3.4545e+03	5.5002e+03	4.2320e+03	4.4168e+03	8.3394e+03
	Sd	6.5872e+02	6.2632e+02	2.1585e+02	3.5663e+02	1.3978e+03	9.2478e+02	8.5546e+02	1.3909e+03
	Median	3.6121e+03	3.5931e+03	3.2504e+03	3.3861e+03	5.0790e+03	4.1255e+03	4.3373e+03	8.3775e+03
F12	Best	**2.9696e+03**	3.1876e+03	3.1441e+03	3.1678e+03	3.1925e+03	3.0151e+03	2.9842e+03	3.4144e+03
	Mean	**3.0476e+03**	3.2462e+03	3.2435e+03	3.2435e+03	3.2934e+03	3.0586e+03	3.0502e+03	3.7493e+03
	Sd	3.1615e+01	3.6985e+01	4.3729e+01	3.9072e+01	4.2830e+01	2.3836e+01	2.9975e+01	1.3950e+02
	Median	3.0417e+03	3.2535e+03	3.2495e+03	3.2442e+03	3.2956e+03	3.0575e+03	3.0553e+03	3.7272e+03
push4	Best	**1.1146e-03**	7.3136e-03	4.6943e-03	1.1628e-03	2.5160e-02	9.5357e-03	3.8704e-02	1.7051e-02
	Mean	**9.9158e-03**	2.5184e-02	3.0141e-02	2.7644e-02	1.0404e-01	9.6135e-02	1.6630e-01	1.0679e-01
	Sd	6.6291e-03	1.3631e-02	1.6189e-02	1.4965e-02	4.7123e-02	5.8469e-02	8.4102e-02	1.2207e-01
	Median	7.9295e-03	2.2115e-02	2.8662e-02	2.5603e-02	1.0480e-01	7.7393e-02	1.6384e-01	8.0304e-02

Table D2. Results of the BOPS framework on the CEC2022 benchmark suite and the real-world *push4* benchmark using the LHD Pareto front approximation strategy with a budget of 800 function evaluations. All results are based on 30 independent runs. Smaller values are better, and the best result for each performance measure is shown in bold.

Function	Measures	eShotgun	eCSAW	eClusterHC	ClusterHC	CSAW	qEI	qUCB	qKG
F1	Best	**2.5232e+04**	3.4578e+04	2.6274e+04	2.6336e+04	4.4881e+04	4.0625e+04	3.8067e+04	3.0797e+04
	Mean	4.3762e+04	4.6959e+04	4.3611e+04	**4.2746e+04**	5.7724e+04	5.4671e+04	6.3118e+04	5.0986e+04
	Sd	8.5438e+03	6.3983e+03	7.6131e+03	7.7229e+03	8.5147e+03	9.4912e+03	1.3668e+04	8.7607e+03
	Median	4.5200e+04	4.7316e+04	4.4082e+04	4.3956e+04	5.6091e+04	5.3672e+04	6.3187e+04	5.2342e+04
F2	Best	**4.8338e+02**	8.9943e+02	9.3827e+02	8.4410e+02	9.3027e+02	7.0659e+02	7.0101e+02	1.8893e+03
	Mean	**5.2798e+02**	1.3176e+03	1.2845e+03	1.2395e+03	1.8437e+03	9.7951e+02	1.0163e+03	2.9608e+03
	Sd	3.8808e+01	2.0520e+02	1.7426e+02	1.5647e+02	4.2034e+02	1.7379e+02	1.6607e+02	6.1233e+02
	Median	5.2795e+02	1.3361e+03	1.2900e+03	1.2736e+03	1.8514e+03	9.0698e+02	1.0085e+03	2.9872e+03
F3	Best	**6.0000e+02**	6.0038e+02	6.0050e+02	6.0049e+02	6.0070e+02	6.0021e+02	6.0002e+02	6.0132e+02
	Mean	**6.0000e+02**	6.0066e+02	6.0069e+02	6.0069e+02	6.0088e+02	6.0057e+02	6.0003e+02	6.0172e+02
	Sd	2.2339e-04	9.3183e-02	8.9262e-02	6.7753e-02	9.2289e-02	1.5339e-01	7.5118e-03	2.0335e-01
	Median	6.0000e+02	6.0066e+02	6.0072e+02	6.0070e+02	6.0090e+02	6.0059e+02	6.0003e+02	6.0171e+02
F4	Best	**8.5324e+02**	1.0587e+03	1.0391e+03	1.0216e+03	1.1008e+03	1.0374e+03	9.9687e+02	1.1132e+03
	Mean	**9.4379e+02**	1.1151e+03	1.1077e+03	1.1009e+03	1.1610e+03	1.0963e+03	1.1092e+03	1.1832e+03
	Sd	9.8144e+01	2.5007e+01	3.1014e+01	3.5971e+01	3.4787e+01	2.8906e+01	4.1378e+01	3.2324e+01
	Median	9.0384e+02	1.1138e+03	1.1099e+03	1.1150e+03	1.1670e+03	1.0952e+03	1.1245e+03	1.1838e+03
F5	Best	**9.0034e+02**	9.0426e+02	9.0571e+02	9.0554e+02	9.0608e+02	9.0211e+02	9.0105e+02	9.0547e+02
	Mean	**9.0131e+02**	9.0742e+02	9.0753e+02	9.0745e+02	9.0952e+02	9.0318e+02	9.0193e+02	9.1319e+02
	Sd	7.6131e-01	1.1852e+00	9.0541e-01	1.0910e+00	1.8624e+00	5.9971e-01	5.9758e-01	3.3818e+00
	Median	9.0102e+02	9.0767e+02	9.0750e+02	9.0759e+02	9.0957e+02	9.0319e+02	9.0185e+02	9.1361e+02
F6	Best	**2.5211e+05**	2.0908e+08	1.1725e+08	1.3995e+08	2.5552e+08	9.9413e+06	2.0229e+07	2.5031e+09
	Mean	**6.3732e+05**	4.3087e+08	4.0497e+08	3.4531e+08	6.2663e+08	1.4709e+08	4.9653e+07	4.2345e+09
	Sd	3.2326e+05	1.4696e+08	1.3342e+08	1.1507e+08	2.3548e+08	1.1070e+08	1.5813e+07	8.4562e+08
	Median	5.3028e+05	4.4255e+08	4.2670e+08	3.5078e+08	6.8418e+08	1.1599e+08	4.7070e+07	4.3427e+09
F7	Best	**1.9826e+03**	2.3457e+03	2.5803e+03	2.3447e+03	2.7508e+03	2.5569e+03	2.3539e+03	3.6736e+03
	Mean	**2.2391e+03**	3.2018e+03	3.1912e+03	3.1619e+03	3.7546e+03	2.9706e+03	3.0289e+03	5.5424e+03
	Sd	1.3247e+02	3.0754e+02	2.3489e+02	3.4216e+02	3.9964e+02	2.2157e+02	3.2195e+02	8.9685e+02
	Median	2.2526e+03	3.2247e+03	3.2300e+03	3.1756e+03	3.7722e+03	2.9432e+03	3.1365e+03	5.4306e+03
F8	Best	4.1086e+06	2.3302e+05	2.9598e+06	2.6561e+08	2.7916e+07	**5.2276e+04**	3.0795e+07	1.5952e+07
	Mean	9.6412e+10	5.1731e+10	**4.5583e+10**	7.1762e+10	1.4163e+11	5.8534e+10	7.4086e+10	3.5425e+11
	Sd	1.5875e+11	6.6684e+10	8.3927e+10	1.5640e+11	1.8929e+11	9.9258e+10	1.0155e+11	5.7719e+11
	Median	2.5463e+10	3.0670e+10	7.8039e+09	1.5507e+10	5.1621e+10	1.4498e+10	2.6237e+10	1.3853e+11
F9	Best	**2.5525e+03**	3.2525e+03	3.2003e+03	3.2533e+03	3.2192e+03	3.2721e+03	3.3567e+03	3.8741e+03
	Mean	**2.9217e+03**	3.4747e+03	3.4892e+03	3.4538e+03	3.9379e+03	3.6032e+03	3.5653e+03	5.7409e+03
	Sd	1.2309e+02	1.0574e+02	1.5130e+02	1.3261e+02	2.2941e+02	1.9097e+02	1.3892e+02	8.7618e+02
	Median	2.9407e+03	3.4838e+03	3.4922e+03	3.4256e+03	3.9894e+03	3.6035e+03	3.5687e+03	5.5617e+03
F10	Best	3.0983e+03	**2.9292e+03**	2.9596e+03	2.9433e+03	2.9955e+03	3.3573e+03	3.1676e+03	2.9463e+03
	Mean	4.5657e+03	3.9799e+03	4.2412e+03	**3.7512e+03**	4.1858e+03	4.6573e+03	4.8738e+03	4.2756e+03
	Sd	8.4355e+02	1.4656e+03	1.4074e+03	1.3008e+03	1.4044e+03	6.4570e+02	8.2154e+02	9.1346e+02
	Median	4.5125e+03	3.0104e+03	3.5915e+03	3.0138e+03	3.0910e+03	4.6074e+03	4.9589e+03	4.4192e+03
F11	Best	3.0108e+03	3.0543e+03	2.7945e+03	**2.7509e+03**	3.4391e+03	2.9342e+03	3.1783e+03	4.2223e+03
	Mean	3.9252e+03	3.5870e+03	3.4720e+03	**3.3990e+03**	5.4326e+03	4.2320e+03	4.4168e+03	8.3394e+03
	Sd	6.7174e+02	3.6062e+02	4.9816e+02	2.8401e+02	1.0852e+03	9.2478e+02	8.5546e+02	1.3909e+03
	Median	3.7071e+03	3.5454e+03	3.3891e+03	3.4270e+03	5.2067e+03	4.1255e+03	4.3373e+03	8.3775e+03
F12	Best	**2.9755e+03**	3.1559e+03	3.1954e+03	3.1506e+03	3.1195e+03	3.0151e+03	2.9842e+03	3.4144e+03
	Mean	**3.0483e+03**	3.2475e+03	3.2410e+03	3.2332e+03	3.3013e+03	3.0586e+03	3.0502e+03	3.7493e+03
	Sd	3.9311e+01	3.7874e+01	2.9730e+01	3.7729e+01	6.9376e+01	2.3836e+01	2.9975e+01	1.3950e+02
	Median	3.0537e+03	3.2471e+03	3.2465e+03	3.2352e+03	3.3245e+03	3.0575e+03	3.0553e+03	3.7272e+03
push4	Best	**9.3108e-04**	2.4662e-03	8.5663e-03	3.8908e-03	1.7192e-02	9.5357e-03	3.8704e-02	1.7051e-02
	Mean	**2.1909e-02**	2.8574e-02	3.2553e-02	3.7196e-02	8.7919e-02	9.6135e-02	1.6630e-01	1.0679e-01
	Sd	4.1159e-02	2.0031e-02	1.7069e-02	2.0412e-02	4.2526e-02	5.8469e-02	8.4102e-02	1.2207e-01
	Median	1.1090e-02	2.8939e-02	2.8312e-02	3.4074e-02	8.7153e-02	7.7393e-02	1.6384e-01	8.0304e-02

Table D3. Results of the BOPS framework on the CEC2022 benchmark suite and the real-world *push4* benchmark using the random Pareto front approximation strategy with a budget of 800 function evaluations. All results are based on 30 independent runs. Smaller values are better, and the best result for each performance measure is shown in bold.

Function	Measures	eShotgun	eCSAW	eClusterHC	ClusterHC	CSAW	qEI	qUCB	qKG
F1	Best	2.0844e+04	**1.9637e+04**	2.3151e+04	2.9190e+04	3.4301e+04	4.0625e+04	3.8067e+04	3.0797e+04
	Mean	**4.0963e+04**	4.5249e+04	4.2074e+04	4.5598e+04	5.5920e+04	5.4671e+04	6.3118e+04	5.0986e+04
	Sd	7.4472e+03	8.9143e+03	7.3610e+03	6.3918e+03	1.0587e+04	9.4912e+03	1.3668e+04	8.7607e+03
	Median	4.0577e+04	4.7100e+04	4.2279e+04	4.5860e+04	5.5034e+04	5.3672e+04	6.3187e+04	5.2342e+04
F2	Best	**4.8079e+02**	1.0533e+03	9.6644e+02	9.8548e+02	1.1816e+03	7.0659e+02	7.0101e+02	1.8893e+03
	Mean	**5.2412e+02**	1.3217e+03	1.2895e+03	1.3268e+03	1.8758e+03	9.7951e+02	1.0163e+03	2.9608e+03
	Sd	3.3921e+01	1.6598e+02	1.6761e+02	1.6793e+02	3.2979e+02	1.7379e+02	1.6607e+02	6.1233e+02
	Median	5.1655e+02	1.3011e+03	1.2806e+03	1.3277e+03	1.8117e+03	9.0698e+02	1.0085e+03	2.9872e+03
F3	Best	**6.0000e+02**	6.0052e+02	6.0052e+02	6.0049e+02	6.0049e+02	6.0021e+02	6.0002e+02	6.0132e+02
	Mean	**6.0000e+02**	6.0071e+02	6.0070e+02	6.0069e+02	6.0085e+02	6.0057e+02	6.0003e+02	6.0172e+02
	Sd	2.1651e-04	9.1475e-02	8.6264e-02	8.8346e-02	1.5298e-01	1.5339e-01	7.5118e-03	2.0335e-01
	Median	6.0000e+02	6.0073e+02	6.0070e+02	6.0071e+02	6.0088e+02	6.0059e+02	6.0003e+02	6.0171e+02
F4	Best	**8.4400e+02**	1.0567e+03	1.0480e+03	1.0598e+03	1.0670e+03	1.0374e+03	9.9687e+02	1.1132e+03
	Mean	**9.3712e+02**	1.1067e+03	1.1105e+03	1.1066e+03	1.1539e+03	1.0963e+03	1.1092e+03	1.1832e+03
	Sd	8.1762e+01	2.7906e+01	3.0685e+01	2.4827e+01	3.9041e+01	2.8906e+01	4.1378e+01	3.2324e+01
	Median	9.1629e+02	1.1080e+03	1.1152e+03	1.1055e+03	1.1536e+03	1.0952e+03	1.1245e+03	1.1838e+03
F5	Best	**9.0015e+02**	9.0516e+02	9.0516e+02	9.0580e+02	9.0588e+02	9.0211e+02	9.0105e+02	9.0547e+02
	Mean	**9.0112e+02**	9.0757e+02	9.0724e+02	9.0758e+02	9.0965e+02	9.0318e+02	9.0193e+02	9.1319e+02
	Sd	8.2672e-01	1.0415e+00	1.0090e+00	1.0240e+00	1.5440e+00	5.9971e-01	5.9758e-01	3.3818e+00
	Median	9.0082e+02	9.0769e+02	9.0742e+02	9.0772e+02	9.0976e+02	9.0319e+02	9.0185e+02	9.1361e+02
F6	Best	**1.8578e+05**	1.5820e+08	1.5820e+08	1.9898e+08	2.4355e+08	9.9413e+06	2.0229e+07	2.5031e+09
	Mean	**6.0921e+05**	4.4241e+08	4.3541e+08	4.0886e+08	6.1880e+08	1.4709e+08	4.9653e+07	4.2345e+09
	Sd	3.2932e+05	1.1241e+08	1.1439e+08	1.0942e+08	1.9556e+08	1.1070e+08	1.5813e+07	8.4562e+08
	Median	4.9570e+05	4.3140e+08	4.3140e+08	4.1267e+08	5.8042e+08	1.1599e+08	4.7070e+07	4.3427e+09
F7	Best	**1.9890e+03**	2.4107e+03	2.4600e+03	2.7443e+03	2.8461e+03	2.5569e+03	2.3539e+03	3.6736e+03
	Mean	**2.1976e+03**	3.1322e+03	3.2188e+03	3.2074e+03	3.7755e+03	2.9706e+03	3.0289e+03	5.5424e+03
	Sd	1.1967e+02	2.8690e+02	3.4253e+02	2.6710e+02	4.7733e+02	2.2157e+02	3.2195e+02	8.9685e+02
	Median	2.1961e+03	3.1376e+03	3.2408e+03	3.2482e+03	3.7311e+03	2.9432e+03	3.1365e+03	5.4306e+03
F8	Best	2.6528e+07	1.8911e+06	2.6325e+07	9.9267e+04	5.0595e+07	**5.2276e+04**	3.0795e+07	1.5952e+07
	Mean	1.6859e+11	6.3774e+10	1.1885e+11	**5.0938e+10**	1.5146e+11	5.8534e+10	7.4086e+10	3.5425e+11
	Sd	2.2673e+11	9.2898e+10	1.5672e+11	9.1951e+10	3.1362e+11	9.9258e+10	1.0155e+11	5.7719e+11
	Median	6.4161e+10	2.9774e+10	4.5266e+10	9.1080e+09	2.9104e+10	1.4498e+10	2.6237e+10	1.3853e+11
F9	Best	**2.4239e+03**	3.0085e+03	3.0085e+03	3.0607e+03	3.4714e+03	3.2721e+03	3.3567e+03	3.8741e+03
	Mean	**2.9209e+03**	3.5011e+03	3.4392e+03	3.4775e+03	3.9611e+03	3.6032e+03	3.5653e+03	5.7409e+03
	Sd	1.7413e+02	1.7387e+02	1.5958e+02	1.7746e+02	2.7301e+02	1.9097e+02	1.3892e+02	8.7618e+02
	Median	2.9402e+03	3.5253e+03	3.4259e+03	3.4852e+03	3.9309e+03	3.6035e+03	3.5687e+03	5.5617e+03
F10	Best	3.1327e+03	**2.9310e+03**	2.9447e+03	2.9312e+03	2.9808e+03	3.3573e+03	3.1676e+03	2.9463e+03
	Mean	4.5719e+03	3.8408e+03	**3.8317e+03**	4.2985e+03	4.3662e+03	4.6573e+03	4.8738e+03	4.2756e+03
	Sd	6.9383e+02	9.6857e+02	1.4336e+03	1.4163e+03	1.4892e+03	6.4570e+02	8.2154e+02	9.1346e+02
	Median	4.5705e+03	3.3027e+03	3.0188e+03	3.6284e+03	3.4783e+03	4.6074e+03	4.9589e+03	4.4192e+03
F11	Best	2.9927e+03	2.8990e+03	2.8983e+03	**2.8247e+03**	3.2660e+03	2.9342e+03	3.1783e+03	4.2223e+03
	Mean	3.7506e+03	3.5578e+03	3.4540e+03	**3.3925e+03**	5.2264e+03	4.2320e+03	4.4168e+03	8.3394e+03
	Sd	6.8893e+02	4.0361e+02	3.7236e+02	4.0205e+02	1.3935e+03	9.2478e+02	8.5546e+02	1.3909e+03
	Median	3.6873e+03	3.5990e+03	3.3543e+03	3.2152e+03	5.0483e+03	4.1255e+03	4.3373e+03	8.3775e+03
F12	Best	2.9995e+03	3.1499e+03	3.1568e+03	3.1701e+03	3.1744e+03	3.0151e+03	**2.9842e+03**	3.4144e+03
	Mean	3.0560e+03	3.2316e+03	3.2415e+03	3.2343e+03	3.3063e+03	3.0586e+03	**3.0502e+03**	3.7493e+03
	Sd	3.2798e+01	4.1415e+01	3.9930e+01	3.5965e+01	5.6250e+01	2.3836e+01	2.9975e+01	1.3950e+02
	Median	3.0516e+03	3.2337e+03	3.2485e+03	3.2375e+03	3.3013e+03	3.0575e+03	3.0553e+03	3.7272e+03
push4	Best	1.7020e-03	6.9684e-03	8.9148e-03	**1.1766e-03**	1.5447e-02	9.5357e-03	3.8704e-02	1.7051e-02
	Mean	**1.1124e-02**	2.9976e-02	2.8385e-02	2.6762e-02	9.0409e-02	9.6135e-02	1.6630e-01	1.0679e-01
	Sd	7.7474e-03	1.7797e-02	1.5571e-02	1.4899e-02	6.0165e-02	5.8469e-02	8.4102e-02	1.2207e-01
	Median	8.7975e-03	2.5633e-02	2.3737e-02	2.4461e-02	8.2211e-02	7.7393e-02	1.6384e-01	8.0304e-02

E. Experimental parameter settings

Table E1. Experimental parameter settings for the Pareto-based solution selection methods. A check mark (✓) indicates that the corresponding parameter is used by the algorithm.

Parameter	ClusterHC	CSAW	ε-Shotgun	ε-ClusterHC	ε-CSAW	Setting
Batch size (q)	✓	✓	✓	✓	✓	4
Initial Latin hypercube design for the GP	✓	✓	✓	✓	✓	$5D$
DBSCAN neighborhood radius (β)	✓	✓		✓	✓	5% of the objective-space range
DBSCAN minimum neighbors (MinPts)	✓	✓		✓	✓	1
ε-greedy probability (ε)			✓	✓	✓	0.1
Exploration parameter (γ)			✓	✓	✓	1
Kernel length scale (l)			✓	✓	✓	Estimated during GP hyperparameter optimization

Table E2. Experimental parameter settings for the acquisition-function baselines. A check mark (✓) indicates that the corresponding parameter is used by the algorithm.

Parameter	qEI	qUCB	qKG	Setting
Batch size (q)	✓	✓	✓	4
Initial Latin hypercube design for the GP	✓	✓	✓	$5D$
Exploration weight (κ)		✓		2.0
Fantasy samples			✓	32
Number of optimization restarts			✓	400
Number of raw initial samples			✓	1000