Computational Optimization and Applications (2025) 92:81–123
https://doi.org/10.1007/s10589-025-00696-7

A Bi-Objective Optimization Based Acquisition Strategy for Batch Bayesian Global Optimization

Francesco Carciaghi[1] · Simone Magistri[1] · Pierluigi Mansueto[1] · Fabio Schoen[1]

Received: 1 February 2024 / Accepted: 5 May 2025 / Published online: 23 May 2025

Abstract

In this paper we deal with batch Bayesian Optimization (Bayes-Opt) problems over a box. Bayes-Opt approaches find their main applications when the objective function is very expensive to evaluate. Sometimes, given the availability of multi-processor computing architectures, function evaluation might be performed in parallel in order to lower the clock-time of the overall computation. This paper fits this situation and is devoted to the development of a novel bi-objective optimization (BOO) acquisition strategy to sample batches of points where to evaluate the objective function. The BOO problem involves the Gaussian Process posterior mean and variance functions, which, in most of the acquisition strategies from the literature, are generally used in combination, frequently through scalarization. However, such scalarization could compromise the Bayes-Opt process performance, as getting the desired trade-off between exploration and exploitation is not trivial in most cases. We instead aim to reconstruct the Pareto front of the BOO problem exploiting first order information of the posterior mean and variance, thus generating multiple trade-offs of the two functions without any a priori knowledge. The algorithm used for the reconstruction is the Non-dominated Sorting Memetic Algorithm (NSMA), recently proposed in the literature and proved to be effective in solving hard MOO problems. Finally, we present two clustering approaches, each of them operating on a different space, to select potentially optimal points from the Pareto front. We compare our methodology with well-known acquisition strategies from the literature, showing its effectiveness on a wide set of experiments.

✉ Pierluigi Mansueto
pierluigi.mansueto@unifi.it

Francesco Carciaghi
francesco.carciaghi@unifi.it

Simone Magistri
simone.magistri@unifi.it

Fabio Schoen
fabio.schoen@unifi.it

[1] Global Optimization Laboratory, Department of Information Engineering, University of Florence, Via di S. Marta, 3, 50139 Florence, Italy

Keywords Global Optimization · Batch Bayesian Methods · Bi-Objective Optimization · Acquisition Function · First-order information

Mathematics Subject Classification 62C10 · 90C29 · 90C30

1 Introduction

Bayesian Optimization is a set of highly relevant heuristic optimization approaches whose aim is to approximately find the global optimum of the following problem:

$$\min_{x \in \Omega} f(x), \tag{1}$$

where Ω is typically a simple feasible set, i.e., a box $\{x \in \mathbb{R}^n \mid x \in [l, u],\ l, u \in \mathbb{R}^n \text{ s.t. } l^i \leq u^i\}$, and $f : \mathbb{R}^n \to \mathbb{R}$ is a continuous, expensive-to-evaluate objective function. We assume that $f(\cdot)$ is evaluated without noise. The expensiveness of function evaluations may be caused by different factors: a high consumption of resources (CPU/GPU or clock time), monetary costs, investments, etc.. Application of such context can be found in various fields: hyper-parameters tuning for machine learning algorithms [1], engineering systems design [2, 3], environmental models calibration [4], reinforcement learning [5, 6] and materials/drug design [7, 8] are just a few examples. Thus, the development of strategies to generate a good approximation of a global optimum, within a very limited budget of function evaluations, is of great practical interest and Bayes-Opt stands out as one of the most frequently adopted approaches.

Early papers on the subject can be traced back to [9], whose author can be considered as one of the fathers of Bayes-Opt, or to [10], who greatly contributed to the diffusion of the Bayesian Optimization ideas. Years later, a quite specific Bayes-Opt approach was presented in [3] in an extremely clean and elegant way, attracting much interest from the communities. In particular, the attention of the machine learning community explains the increased importance towards black-box optimization of expensive functions. In recent years, Bayes-Opt was employed both in problems like (1) and in more difficult ones (see, e.g., [11], where instances with both expensive-to-evaluate functions and constraints are taken into account, and [12–14], where Bayes-Opt has been considered in high-dimensional scenarios). A comprehensive overview of recent approaches is provided in [15], while [16] offers a detailed introduction to the core aspects of the theory, along with a historical perspective on Bayesian optimization. Furthermore, many software tools have been recently developed, e.g., `BoTorch` [17], which is widely used in the literature for modern Bayesian Optimization, `DiceOptim` [18], `SMT 2.0` [19], `Trieste` [20], `SMAC3` [21] and `Dragonfly` [22].

We recall that Bayes-Opt mainly consists in two phases: first, a stochastic model of the objective function, i.e., a Gaussian Process, is defined *a priori*, and, then, it is updated after observing $f(\cdot)$ at specific feasible points. The resulting *a posteriori* model contains the updated knowledge we can use in order to predict the objective function value at new not-yet-visited feasible points. We refer the interested reader to [23] for a deep and exhaustive analysis on how to model a surrogate function

through Gaussian Processes. Given the stochastic model, the problem now becomes that of exploiting information contained in the model to guide the search towards likely locations of a global optimum. Assume that, after the objective function $f(\cdot)$ has been sampled at k feasible points $x_1, \ldots, x_k \in \mathbb{R}^n$, a Gaussian model has been fitted; let us denote by $\mu_k(\cdot)$ and $\sigma_k(\cdot)$ the posterior mean and standard deviation of $f(\cdot)$. One of the most frequently employed criteria consists on optimizing a suitably defined *acquisition* function, such as the *Expected Improvement* (`EI`) [10] or the *Lower Confidence Bound* (`LCB`) [24], seeking a compromise between points close to already observed low valued feasible solutions (low $\mu_k(\cdot)$ values) and points in high uncertainty zones (high $\sigma_k(\cdot)$ values) typically located as far as possible from past observations. Here, we deal with Batch Bayesian Optimization, a sub-field of Bayes-Opt where, at each iteration, $q > 1$ feasible solutions are chosen through an *acquisition* criterion to be evaluated on $f(\cdot)$. The idea of using Bayes-Opt for multiple simultaneous function evaluations is not new in the literature, e.g., in [25], based on the exploitation of the `q-EI`, or in [26–28], where the `q-LCB` formulation can be found. Other related works are: approaches based on entropy-based criteria, whose one of the representatives is `GIBBON` [29], providing computationally light-weight yet high performing optimization for a wide range of (even non-Euclidean) BO problems; global optimization approaches (e.g., `GLASSES` [30]) employing (expected value of) "non-myopic" losses that take into account dozens of (simulated) future evaluations before making the decision of where to sample the objective function; extensions of other well-known acquisition functions from the Bayesian literature, such as the *Knowledge Gradient* (`KG`) [31], to the batch case; methodologies, like `TURBO` [32], which, instead of fitting a global surrogate model and balancing exploration and exploitation on the whole search space, maintain a collection of local probabilistic models able to quickly discover excellent objective values. It might seem contradictory, in such an expensive setting, to evaluate the objective function several times before updating the Gaussian Process. However, in many situations, parallel computing machines are available; moreover, the idea of simultaneously and independently computing the objective function is worth considering when we try to reduce the wall-clock time of the overall procedure.

In this paper, we propose a novel gradient-based *acquisition* strategy, aimed to reconstruct the Pareto front of a Bi-Objective Optimization problem involving $\mu_k(\cdot)$ and $\sigma_k(\cdot)$. Unlike the mentioned approaches, we do not rely on combinations of the two posterior functions; indeed, such a way of proceeding might compromise the Bayes-Opt process performance, being difficult in some cases to get the desired trade-off between exploitation and exploration. Reconstructing the Pareto front instead leads to the generation of multiple compromises without any a priori setting, thus allowing to choose, based on some criteria, the most suitable points among many. In general, Multi-Objective Optimization (MOO) received much attention in the last decades; many approaches, mainly extensions of classical scalar optimization techniques, have been developed: [33–42] just represent a non-exhaustive list of this type of methodologies. In particular, some of the most recent ones are capable of managing multiple solutions at each iteration, finally providing an approximation, as accurate as possible, of the Pareto front.

The idea of an acquisition strategy based on MOO has been considered in some recent works from the literature. In [43], $\mu_k(\cdot)$ and $\sigma_k(\cdot)$ are used as objective functions in a BOO problem; the latter is then solved with the *Non-dominated Sorting Genetic Algorithm II* (`NSGA-II`) [41], which is one of the most representative evolutionary algorithms for solving hard MOO problems; after reconstructing the Pareto front of the considered BOO problem, the new points where to evaluate $f(\cdot)$ are randomly selected. The same algorithmic scheme characterizes the work in [44], where a variant of `NSGA-II` [45] is employed. In [46], `NSGA-II` and some of its variants are used to solve a BOO problem where, at each iteration of the Bayes-Opt procedure, the objectives are dynamically chosen based on an ad-hoc defined metric from a pool of well-known acquisition functions, such as `q-EI` and `q-LCB`; then, the candidate solutions selection involves a clustering approach employed in the domain space of the Pareto front points; the resulting acquisition strategy is called `BBO-ABAFMo`. In [47], the BOO problem is composed by the two terms of the `EI`; the considered optimization method is the *Multi-objective Evolutionary Algorithm based on Decomposition* (`MOEA/D`) [48], based on a decomposition strategy to turn the BOO problem into N scalar ones, each of them characterized by a specific weighted sum of the objectives; the selection of points is performed through clustering of the weights vectors defining the N scalar problems. In [49], the authors propose an approach relying on multi-objective optimization problems with multiple objective functions, including $\mu_k(\cdot)$, $\sigma_k(\cdot)$ and `EI`; the resulting multi-objective optimization problems are solved by means of an evolutionary (genetic) optimization algorithm. Finally, in [50, 51] the authors propose `MACE`, an acquisition strategy based on solving an MOO problem composed by three well-known acquisition functions, i.e., `EI`, `LCB` and *Probability of Improvement* (`PI`) [9]; the employed algorithm is the *Differential Evolution for Multiobjective Optimization* (`DEMO`) [52]; candidate points are again randomly chosen from the resulting Pareto front approximation.

Our novel acquisition strategy differs from the mentioned ones in the following aspects:

- At each iteration of the batch Bayes-Opt procedure, we aim to reconstruct, as accurately as possible, the Pareto front of the BOO problem

$$\min_{x \in \Omega} \left(\mu_k(x), -\sigma_k^2(x)\right)^\top \tag{2}$$

  exploiting the gradient information of $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$. To the best of our knowledge, this is the first time in the batch Bayesian optimization literature that first-order information of mean and variance functions has been used to approximate, as accurately as possible, the Pareto front for problem (2). These two functions are the basis for some of the most employed acquisition functions (e.g., `EI`, `LCB`), which scalarize problem (2) without building the Pareto frontier: optimizing $\mu_k(\cdot)$ leads to exploit the current information on the objective function $f(\cdot)$, while $\sigma_k^2(\cdot)$ let us highlight regions where we know little or nothing about the objective function. This duality leads to a resulting Pareto front with a clear meaning, being composed by different tradeoffs between exploitation and exploration of the current information about $f(\cdot)$.

- We employ the *Non-dominated Sorting Memetic Algorithm* [42] to solve problem (2). NSMA combines the (random-based) genetic operators of NSGA-II with tools typical of gradient-based methodologies; as it is shown in [42], through the latter NSMA obtains more refined Pareto front reconstructions than NSGA-II, which is widely used in Batch Bayesian literature; on the other hand, the good exploration capabilities inherited from NSGA-II are used to escape from stationary non-optimal points, thus allowing NSMA to perform better than gradient-based [39] and derivative-free [40] algorithms with stationarity convergence guarantees. With respect to other proposals in the multi-objective literature, NSMA thus combines the typical advantages of evolutionary algorithms (exploration) and gradient-based ones (exploitation), being crucial in determining high quality Pareto front reconstructions. The comparisons, reported in this paper, of NSMA, NSGA-II, a gradient-based [36] and a derivative-free [40] algorithm from the multi-objective literature as solvers for problem (2) support this fact.
- We propose two clustering methodologies, each of them operating on a different space, to choose from the Pareto front the new points where to evaluate $f(\cdot)$; in particular, we analyze the potential effectiveness of such clustering approaches on high-dimensional scenarios ($n \geq 20$), where the curse of dimensionality usually prevents the employed acquisition functions to find good candidate solutions where to evaluate the objective function $f(\cdot)$.

The remainder of the manuscript is organized as follows. In Sect. 2, we recall the main batch Bayes-Opt concepts, focusing on two main aspects: the Gaussian Process regression and the acquisition phase. In this section, we also review the main MOO optimality concepts from Pareto's theory; in Appendix A we also recall the theoretical relations that provably exist between these concepts and Pareto necessary optimality conditions. In Sect. 3, we describe our novel BOO based acquisition strategy, reporting a brief description of NSMA, adapted to the Bayes-Opt context, and the algorithmic scheme of the proposed clustering approaches. Further information on the memetic approach can be also found in Appendix B. In Sect. 4, we analyze the results of thorough computational experiments, mainly highlighting the good performance of our acquisition methodology w.r.t. well-known ones from the batch Bayes-Opt literature. Finally, in Sect. 5 we provide some concluding remarks.

2 Preliminaries

In this section, we first recall the principal batch Bayes-Opt concepts, including the Gaussian Process regression and the acquisition phase. Then, we review the basic MOO optimality concepts from the Pareto's theory.

2.1 Batch Bayesian Optimization Overview

In Algorithm 1, we report the main steps of a batch Bayes-Opt general scheme.

The procedure starts from N_i initial points in the feasible set Ω [Line 1, Algorithm 1]. For these points, we assume to know the value of $f(\cdot)$ [Line 2, Algorithm 1]. Moreover, as anticipated in Sect. 1, we suppose that each $f(\cdot)$ evaluation is performed in a noise-free setting. As in [15], we make use of the following compact notations:

Algorithm 1 Batch Bayes-Opt

Require: $f : \mathbb{R}^n \to \mathbb{R}$, Ω feasible set, $N_i \in \mathbb{N}^+$ number of initial points, $N_M > N_i \in \mathbb{N}$ number of evaluated points (excluding the first N_i evaluations), $q > 1 \in \mathbb{N}$ batch size (number of evaluated points at each iteration), $\mu_0(\cdot)$ mean function, $\Sigma_0(\cdot, \cdot)$ kernel.

1: Let $x_{1:N_i}$ be N_i initial points in Ω
2: Let $f(x_{1:N_i})$ be the values of $f(\cdot)$ at $x_{1:N_i}$
3: Let $k = N_i$
4: **while** $k \leq N_i + N_M$ **do**
5: Let the *prior distribution* on $f(x_{1:k})$ be

$$f(x_{1:k}) \sim \mathcal{N}(\mu_0(x_{1:k}), \Sigma_0(x_{1:k}, x_{1:k})) \tag{3}$$

6: Let the *posterior distribution* on $f(x)$ be

$$\begin{aligned} f(x) \,|\, f(x_{1:k}) &\sim \mathcal{N}\left(\mu_k(x), \sigma_k^2(x)\right) \text{ with} \\ \mu_k(x) &= \Sigma_0(x, x_{1:k})\Sigma_0(x_{1:k}, x_{1:k})^{-1}(f(x_{1:k}) - \mu_0(x_{1:k})) + \mu_0(x) \\ \sigma_k^2(x) &= \Sigma_0(x, x) - \Sigma_0(x, x_{1:k})\Sigma_0(x_{1:k}, x_{1:k})^{-1}\Sigma_0(x_{1:k}, x) \end{aligned} \tag{4}$$

7: Let $x_{k+1}, \ldots, x_{k+q}$ be the new points found by minimizing an acquisition function based on the posterior distribution
8: Evaluate $f(\cdot)$ at $x_{k+1}, \ldots, x_{k+q}$
9: Let $k = k + q$
10: **end while**
11: **return** $x^\star$ s.t. $f(x^\star) \leq f(x),\ \forall x \in x_{1:N_i+N_M}$

given $k \geq 1$,

$$\begin{aligned} x_{1:k} &= [x_1, \ldots, x_k]^\top \in \mathbb{R}^{k \times n}, \\ f(x_{1:k}) &= [f(x_1), \ldots, f(x_k)]^\top \in \mathbb{R}^k. \end{aligned} \tag{5}$$

Now, in order to model the objective function $f(\cdot)$, the Gaussian Process regression [Lines 5-6, Algorithm 1] is carried out. Let us suppose that we have already evaluated $f(\cdot)$ at k points. Given these evaluations, we can define the *prior distribution* on $f(x_{1:k})$ as in (3), where $\mathcal{N}(\mu_0(x_{1:k}), \Sigma_0(x_{1:k}, x_{1:k}))$ denotes the multivariate normal distribution with $\mu_0(\cdot)$ and $\Sigma_0(\cdot, \cdot)$ being the *mean* and *covariance* (*kernel*) functions. The kernel is chosen such that close points have a large positive correlation, based on the idea that similar function values take place in them more likely than in points distant from each other. Similarly to (5), we indicate with $\mu_0(x_{1:k})$ and $\Sigma_0(x_{1:k}, x_{1:k})$ the vector and matrix, respectively, derived from evaluating these two functions at $x_1, \ldots, x_k$, i.e., $\mu_0(x_{1:k}) = [\mu_0(x_1), \ldots, \mu_0(x_k)]^\top$ and

$$\Sigma_0(x_{1:k}, x_{1:k}) = \begin{bmatrix} \Sigma_0(x_1, x_1) & \cdots & \Sigma_0(x_1, x_k) \\ \vdots & \ddots & \vdots \\ \Sigma_0(x_k, x_1) & \cdots & \Sigma_0(x_k, x_k) \end{bmatrix}.$$

The matrix $\Sigma_0(x_{1:k}, x_{1:k})$ should be positive semi-definite, regardless the considered points.

The most commonly used mean function is a constant value: $\mu_0(\cdot) = \mu$. As for the kernel, a popular choice is the *Matérn* kernel [23]; here, we report the formulation that can be found in [15]: given $x, y \in \mathbb{R}^n$,

$$\Sigma_0(x, y) = \alpha_0 \frac{2^{1-\nu}}{\Gamma(\nu)} (\sqrt{2\nu}\|x - y\|_{\alpha_{1:n}})^{\nu} K_{\nu}(\sqrt{2\nu}\|x - y\|_{\alpha_{1:n}}),$$

with $\Gamma(\cdot)$ being the *gamma* function, $\|x - y\|_{\alpha_{1:n}} = \sqrt{\sum_{i=1}^{n} \alpha_i (x^i - y^i)^2}$ and $K_{\nu}(\cdot)$ being the *Modified Bessel* function. In most papers, the parameter ν, which controls the smoothness of the function, is fixed: $\nu = 1/2$, $\nu = 3/2$ and $\nu = 5/2$ are the usual choices. In general, the hyper-parameters contained in these formulas, i.e., $\mu, \alpha_{0:n}, \nu$, can be found by *Maximum Likelihood Estimation* (`MLE`) given the observations $f(x_{1:k})$. For more information on this procedure or on other alternative formulations for $\mu_0(\cdot)$ and $\Sigma_0(\cdot, \cdot)$, the reader is referred to [15].

We can then compute the *posterior* probability distribution, which describes potential values of the objective function $f(\cdot)$ at a new candidate point [Line 6, Algorithm 1]. Using Bayes' rule, the conditional distribution of $f(\cdot)$ given $f(x_{1:k})$ can be computed as a normal distribution $\mathcal{N}(\mu_k(\cdot), \sigma_k^2(\cdot))$ (4), where $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$ are the *posterior* mean and variance functions. The mean function $\mu_k(\cdot)$ works as a point estimate of $f(\cdot)$ and typically assumes small values in solutions close to already-observed low-valued points. On the other hand, $\sigma_k^2(\cdot)$ allows to quantify the uncertainty we have on each feasible set region: the farther we move from the already-observed points, the more the uncertainty on the values assumed by $f(\cdot)$ grows. The values of $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$ can be directly computed using (4) and matrix inversion; still, in order to make the computation faster and more numerically stable, some alternative methodologies are proposed in the literature (e.g., [23]). As for their gradients, although analytical forms can be easily derived ($\Sigma_0(x, y)$ is $\lceil \nu \rceil - 1$ times differentiable w.r.t. $\|x - y\|_{\alpha_{1:n}}$ [23]), *automatic differentiation* packages proposed in libraries like `BoTorch` [17] are usually convenient.

Given the posterior distribution, we then minimize in [Line 7, Algorithm 1] an *acquisition function* $a_k : \mathbb{R}^{q \times n} \to \mathbb{R}$ in order to find $q > 1$ new candidate solutions $x_{k+1}, \ldots, x_{k+q}$ where to evaluate the objective function $f(\cdot)$ [Line 8, Algorithm 1]:

$$\min_{\tilde{x}_{k+1}, \ldots, \tilde{x}_{k+q} \in \Omega} a_k(\tilde{x}_{k+1:k+q}). \tag{6}$$

Unlike $f(\cdot)$, the acquisition functions are usually inexpensive to evaluate; moreover, it is feasible to compute their first and second derivatives, which is not possible with $f(\cdot)$. This fact allows the acquisition functions to be optimized using first-order or second-order optimization methods. A solver that has proved to perform well [15] is the *Limited-Memory Broyden-Fletcher-Goldfarb-Shanno* variant designed for box-constrained problems, i.e., `L-BFGS-B` [53].

When the number of maximum objective function evaluations $N_i + N_M$ is reached, the procedure is stopped and the best found solution $x^\star$ is returned [Line 11, Algorithm 1].

In the next section, we review some of the most employed acquisition functions in the Bayes-Opt literature, reporting their formulations and a brief description of each of them.

2.1.1 Expected Improvement and Lower Confidence Bound

In this section, we review two of the most commonly used acquisition functions: the *Expected Improvement* and the *Lower Confidence Bound*.

Using the notation introduced in Sect. 2.1, the *Expected Improvement* (EI) [3, 10] indicates the expected value of the improvement w.r.t. $f_k^\star = \min_{i\in\{1,\dots,k\}} f(x_i)$, that is, the best found objective function value after the evaluation of k points. Following the steps proposed in [3, 54], we can evaluate the EI as:

$$\begin{aligned} EI_k(x) &= \mathbb{E}_y\left[\max\{f_k^\star - y, 0\} \mid x_{1:k}, f(x_{1:k})\right] \\ &= (f_k^\star - \mu_k(x))\Phi\left(\frac{f_k^\star - \mu_k(x)}{\sigma_k(x)}\right) + \sigma_k(x)\phi\left(\frac{f_k^\star - \mu_k(x)}{\sigma_k(x)}\right), \\ y &\sim \mathcal{N}\left(\mu_k(x), \sigma_k^2(x)\right), \end{aligned} \tag{7}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are, respectively, the cumulative distribution function (CDF) and the probability density function (PDF) of a standard normal random variable. Since the expected improvement should be reasonably maximized as much as possible, according to (6), the problem to solve would be $\min_{x\in\Omega} -EI_k(x)$.

The *Lower Confidence Bound* (LCB) [24] prefers both points with low $\mu_k(\cdot)$, where we expect to have high reward, that is, good values of $f(\cdot)$, and points with large $\sigma_k(\cdot)$, where $f(\cdot)$ is uncertain. LCB can be evaluated as:

$$\begin{aligned} LCB_k(x) &= \mathbb{E}_y\left[\mu_k(x) - |y - \mu_k(x)| \mid x_{1:k}, f(x_{1:k})\right] = \mu_k(x) - \sqrt{\beta}\sigma_k(x), \\ y &\sim \mathcal{N}\left(\mu_k(x), \beta\frac{\pi}{2}\sigma_k^2(x)\right), \end{aligned} \tag{8}$$

where $\beta \in \mathbb{R}^n$ is a parameter.

Both EI and LCB seek a trade-off between points where an high reward is expected and points where we have an high uncertainty. In the LCB formula, the desired trade-off can be explicitly controlled by the constant parameter β: if $\beta \approx 0$, the solver used to optimize LCB will be driven towards points associated to low values of $\mu_k(\cdot)$; on the other hand, if β gets larger, the points with high values of $\sigma_k(\cdot)$ will be preferred. We recall that an optimal EI/LCB solution belongs to the Pareto front of problem (2) [3, 43].

The two formulas (7)-(8) are extended to be used for batch Bayes-Opt, where q new feasible, not-yet-evaluated, solutions are required at each iteration. The formulas of the extensions, called q-EI [55] and q-LCB [26] respectively, are reported in (9)-(10), with $\Sigma_k(\cdot,\cdot)$ indicating the *posterior covariance* function given k observations

of $f(\cdot)$.

$$q\text{-}EI_k(x_{k+1:k+q}) = \mathbb{E}_y\left[\max\left\{f_k^\star - \max_{i\in\{1,\dots,q\}} y_i, 0\right\} \mid x_{1:k}, f(x_{1:k})\right]$$
$$y \sim \mathcal{N}\left(\mu_k(x_{k+1:k+q}), \Sigma_k(x_{k+1:k+q}, x_{k+1:k+q})\right) \tag{9}$$

$$q\text{-}LCB_k(x_{k+1:k+q}) = \mathbb{E}_y\left[\max_{i\in\{1,\dots,q\}} (\mu_k(x_{k+i}) - |y_i - \mu_k(x_{k+i})|) \mid x_{1:k}, f(x_{1:k})\right]$$
$$y \sim \mathcal{N}\left(\mu_k(x_{k+1:k+q}), \beta\frac{\pi}{2}\Sigma_k(x_{k+1:k+q}, x_{k+1:k+q})\right) \tag{10}$$

Both `q-EI` and `q-LCB` are more challenging than their original versions since the integrals arising from the expected value formulation are more intractable; as a consequence, although exact formulations for `q-EI` and its gradient are provided in the literature [56, 57], the optimization process per se is not trivial. For this reason, a variety of alternative methods have been proposed in the literature for the evaluation of these acquisition functions. In this paper, when performing numerical experiments with `q-EI` and `q-LCB`, we rely on one of the most employed techniques: Monte Carlo sampling based integration. The latter, along with *automatic differentiation* packages used in libraries like `BoTorch` [17], allows us to get an estimation of the two acquisition functions and their gradients and, thus, to employ solvers such as `L-BFGS-B` to optimize them. For the sake of brevity, we do not go into the details of either this methodology or other possible strategies, referring the reader to [17, 26].

2.2 Optimality Concepts in Multi-Objective Optimization

Since our acquisition strategy involves the Pareto front reconstruction of a BOO problem (2), in this section we recall the main MOO optimality concepts from Pareto's theory. Let us take as reference the following MOO problem:

$$\min_{x\in\Omega} \mathcal{F}(x) = (f_1(x), \dots, f_m(x))^\top, \tag{11}$$

where $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^m$ is a vector-valued, component-wise continuously differentiable function.

In order to introduce the MOO optimality notions, we define a partial ordering of the points in the objectives space $\mathbb{R}^m$: considering two points $u, v \in \mathbb{R}^m$, we have that

$$\begin{aligned} u < v &\iff u_i < v_i \qquad \forall i \in \{1,\dots,m\},\\ u \le v &\iff u_i \le v_i \qquad \forall i \in \{1,\dots,m\},\\ u \lneqq v &\iff u \le v, u \ne v. \end{aligned}$$

We say that a point $x \in \Omega$ *dominates* $y \in \Omega$ w.r.t. $\mathcal{F}(\cdot)$ if $\mathcal{F}(x) \lneqq \mathcal{F}(y)$.

In MOO, we would like to obtain a point which minimizes all the objectives $f_1(\cdot), \dots, f_m(\cdot)$ at once. However, such a solution is unlikely to exist. For this reason,

we rely on optimality concepts different from the ones for single-objective optimization.

Definition 1 A point $\bar{x} \in \Omega$ is *Pareto optimal* for problem (11) if a point $y \in \Omega$ such that $\mathcal{F}(y) \lneqq \mathcal{F}(\bar{x})$ does not exist. If the property is satisfied only in $\Omega \cap \mathcal{N}(\bar{x})$, with $\mathcal{N}(\bar{x})$ being a neighborhood of $\bar{x}$, then $\bar{x}$ is *locally Pareto optimal*.

In practice, it is difficult to get solutions characterized by the Pareto optimality property. Then, a slightly more affordable condition is introduced.

Definition 2 A point $\bar{x} \in \Omega$ is *weakly Pareto optimal* for problem (11) if a point $y \in \Omega$ such that $\mathcal{F}(y) < \mathcal{F}(\bar{x})$ does not exist. If the property is satisfied only in $\Omega \cap \mathcal{N}(\bar{x})$, with $\mathcal{N}(\bar{x})$ being a neighborhood of $\bar{x}$, then $\bar{x}$ is *locally weakly Pareto optimal*.

We refer to the set of Pareto optimal solutions as the *Pareto set*, while by *Pareto front* we indicate the image of the Pareto set through $\mathcal{F}(\cdot)$.

3 A Novel Bi-Objective Optimization Based Acquisition Strategy

In Sect. 2.1.1, we have reported a brief description of some of the most employed acquisition functions from the Bayes-Opt literature. In each of them, a desired trade-off between exploitation (low $\mu_k(\cdot)$ values) and exploration (high $\sigma_k(\cdot)$ values) is defined, either in an implicit way (`EI`, `q-EI`) or in an explicit one through a parameter (`LCB`, `q-LCB`). However, deciding *a priori* such exploitation-exploration compromise is not trivial in some cases and the Bayes-Opt procedure performance may be affected by this additional difficulty.

Based on this reasoning, in this section, we present a novel acquisition methodology for batch Bayes-Opt. As anticipated in Sect. 1, our proposal is based on the gradient-based optimization of the BOO problem (2) and, in particular, on the reconstruction of its Pareto front. Unlike the acquisition strategies of Sect. 2.1.1, the exploitation-exploration trade-off is not decided *a priori*; on the contrary, the Pareto front provides multiple solutions representing different trade-offs between $\mu_k(\cdot)$ and $\sigma_k(\cdot)$; having such a more complete, *a posteriori* overview, we can then employ specific criteria to choose the solutions we expect to be optimal for problem (1).

The key elements of the acquisition strategy are the following:

- in order to reconstruct, as accurately as possible, the Pareto front of the BOO problem (2), we employ the *Non-dominated Sorting Memetic Algorithm* [42], which combines the exploration capabilities of evolutionary approaches with the effectiveness of gradient-based tools; as shown in [42] and anticipated in Sect. 1, this combination enabled `NSMA` to be superior than state-of-the-art gradient-based/derivative-free MOO algorithms and `NSGA-II`, which is widely employed for this scope in the Bayesian literature;
- we propose two clustering approaches, operating on different spaces, for the selection of q Pareto front points that will be the next candidate solutions where to evaluate $f(\cdot)$; after the proposal, we also analyze the potential effectiveness of such approaches on high dimensional scenarios ($n \geq 20$), which are known to

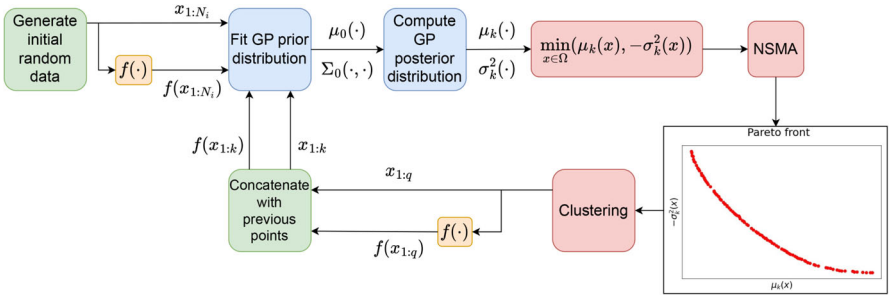


Fig. 1 Flowchart of the batch Bayes-Opt procedure equipped with the BOO based acquisition strategy proposed in Sect. 3

be more difficult in the Bayesian literature as the curse of dimensionality usually leads the acquisition functions failing in finding good candidate solutions.

Remark 1 The paper focuses on batch Bayesian optimization, particularly in scenarios where the evaluation of a (very expensive) objective function can be carried out at a limited number of selected feasible points *in parallel* (e.g., leveraging multiple CPU's). Although the objective function is costly to evaluate, parallelizing such evaluations can significantly reduce the overall wall-clock time. It is clear that a sequential algorithm (i.e., with batch size $q = 1$) may outperform any batch-based approach in terms of solution quality, as it can leverage all previously gathered information when selecting each new evaluation point. In contrast, batch strategies must choose multiple points simultaneously, without the benefit of intermediate feedback. Nevertheless, despite this limitation, batch optimization is often essential in real-world applications where computational resources can be exploited in parallel. We therefore excluded the $q = 1$ case from our experiments, as such a setting lies outside the intended scope of our approach, specifically designed to explore the Pareto front of a bi-objective acquisition function within a batch framework. As we demonstrate in the numerical experiments (Section 4), the advantages of our method become increasingly evident as the batch size grows.

In Fig. 1, we report an overview of the entire batch Bayes-Opt procedure equipped with our BOO based acquisition strategy. The two phases related to the Pareto front reconstruction and clustering are separately discussed in the following two sections.

3.1 Non-dominated Sorting Memetic Algorithm for Bayes-Opt

In Algorithm 2, we report an algorithmic scheme of the *Non-dominated Sorting Memetic Algorithm* [42], adapted to the Bayes-Opt context. Note that here we only report the main steps of the algorithm; for a deeper understanding of its mechanisms, which is beyond the scope of this manuscript, we refer the interested reader to Appendix B and [42]. In the remainder of the section, we indicate with $\mathcal{F}(\cdot) : \mathbb{R}^n \to \mathbb{R}^2$ the vector-valued objective function of problem (2), that is, $\mathcal{F}(\cdot) = (f_1(\cdot), f_2(\cdot))^\top = (\mu_k(\cdot), -\sigma_k^2(\cdot))^\top$, with $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$ being the posterior mean and variance functions (see Sect. 2.1, [Line 6, Algorithm 1]).

Algorithm 2 Non-dominated Sorting Memetic Algorithm for Bayes-Opt

Require: Ω feasible set, $X^i \subset \Omega$, N population size, $n_{opt} \in \mathbb{N}^+$.
1: Let $\mu_k(\cdot)$, $\sigma_k^2(\cdot)$ be the posterior mean and variance functions [Line 6, Algorithm 1]
2: Let $\mathcal{F}(\cdot) = (\mu_k(\cdot), -\sigma_k^2(\cdot))^\top$
3: $X^0 = \texttt{selection}(\mathcal{F}(\cdot), \Omega, X^i, N)$
4: $t = 0$
5: **while** *a stopping criterion is not satisfied* **do**
6: $P^t = \texttt{getParents}(\mathcal{F}(\cdot), \Omega, X^t)$
7: $O^t = \texttt{crossover}(\Omega, P^t)$
8: $\tilde{O}^t = \texttt{mutation}(\Omega, O^t)$
9: $\hat{X}^{t+1} = X^t \cup \tilde{O}^t$
10: $X^{t+1} = \texttt{selection}(\mathcal{F}(\cdot), \Omega, \hat{X}^{t+1}, N)$
11: **if** $t \mod n_{opt} = 0$ **then**
12: $X^{t+1} = \texttt{optimizePopulation}(\Omega, X^{t+1}, \mathcal{F}(\cdot), N)$
13: **end if**
14: $t = t + 1$
15: **end while**
16: $X^\star = X^t$
17: **return** $X^\star$

NSMA is a memetic approach that combines the evolutionary (random-based) operators of NSGA-II [41], which represents the de facto popularity-wise standard for hard box-constrained MOO problems, and the tools typical of MOO gradient-descent methodologies, more efficient and effective than other classes of MOO algorithms on problems with reasonable regularity assumptions.

Like NSGA-II, NSMA works with a fixed size population of N solutions, starting from an initial one denoted by $X^i \subset \Omega$; the population is evolved at each iteration through the application of the NSGA-II operators, i.e., `getParents`, `crossover`, `mutation` and `selection`. `crossover` [Line 7, Algorithm 2] aims to generate new solutions, called *offsprings*, combining the features of selected ones from the current population, called *parents*. The selection of the latter is made by *binary tournament* in the `getParents` function [Line 6, Algorithm 2]: each *parent* is the best, based on specific metrics, of two randomly selected solutions of the population; a brief description of the metrics is reported in Appendix B. `mutation` [Line 8, Algorithm 2] partially modifies the *offsprings* generated by the `crossover` function: the mutated *offsprings* will have some features that are not inherited from the *parents*. After inserting the new, hopefully better, solutions into the population [Line 9, Algorithm 2], the latter is reduced through the `selection` operator [Line 10, Algorithm 2] in order to have exactly N members; the selection of the best N solutions is based on the same metrics used in the `getParents` function. More information about the NSGA-II operators can be found in [41, 42].

The main difference between NSMA and NSGA-II is represented by the `optimizePopulation` function [Line 12, Algorithm 2], which is called every $n_{opt} \in \mathbb{N}^+$ iterations. This function allows to refine selected solutions of the current population X^{t+1} through a gradient-based local solver called *Front Multi-Objective Projected Gradient* (FMOPG), exploiting common [33] and partial [39] descent direc-

tions based on first-order information of the functions $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$. Further details on the `optimizePopulation` function can be found in Appendix B and [42].

By design, the `optimizePopulation` function drives the selected points towards Pareto stationarity, a necessary condition for all types of Pareto Optimality whose definition is reported in Appendix A. However, not being a sufficient condition in general and having no guarantee about the convexity of $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$, we can easily deduce that the new, refined solutions generated by the `optimizePopulation` function could be far from the actual Pareto front of the problem. In such cases, `NSMA` differs from gradient-based and derivative-free algorithms with stationarity convergence guarantees, since it can exploit the `NSGA-II` operators to escape from these stationary, but non-optimal, solutions; on the other hand, `NSMA` is also different from `NSGA-II` as it can refine the new generated points through `FMOPG`, exploiting the gradient information of the functions $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$. This combination of characteristics confirms once again how much the employment of `NSMA` to solve problem (2) in a batch Bayesian optimization context might be appealing.

At the end of the execution, `NSMA` returns the population obtained during the last iteration [Lines 16-17, Algorithm 2]. From this population, we will select q new candidate solutions where to evaluate the objective function $f(\cdot)$ of the original problem (1). In the next section, we report two possible selection criteria.

3.2 Clustering Approaches for Pareto Front Points Selection

After reconstructing, as accurately as possible, the Pareto front of the BOO problem (2) through `NSMA` (Sect. 3.1), we have to select from it q candidate points, which will be then evaluated on the original problem (1). In this section, we propose two clustering-based selection strategies, whose schemes are reported in Algorithm 3. Both strategies rely on the employment of `K-MEANS`, which is arguably the most popular local search algorithm for clustering problems. The `K-MEANS` objective consists in the minimization of the sum-of-squares of the Euclidean distances of the samples to their cluster centers [58]. Besides requiring a few parameters to set (number of clusters to find included), it proves to be particularly efficient and robust in many scenarios.

Clustering in the objectives space The first clustering approach [Lines 2-8, Algorithm 3] involves a `K-MEANS` execution in the objectives space. For each cluster center c_i, with $i \in \{1, \ldots, q\}$, we then find the closest Pareto front point $f_{z_i}^\star$ [Line 6, Algorithm 3]; the corresponding $x_{z_i}^\star$ will be one of the q candidate points to be evaluated on (1) [Line 7, Algorithm 3].

Anticipating the numerical results exposed at the end of Sect. 4.4, performing a clustering operation to choose the q new points from the Pareto front solves an issue typical of approaches such as `LCB`/`q-LCB`: the choice of the weights (e.g., β for `LCB`; see (8)) is not trivial [35], especially when the dimensionality of the problem increases. A bad choice of them could indeed lead to an improper use of the information about the past function observations collected in the Gaussian process and, as a consequence, to compromise the entire optimization procedure. Clustering

Algorithm 3 Clustering Approaches for Pareto Front Selection

Require: $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^2$, Ω feasible set, $X^\star = \left\{x_1^\star, \ldots, x_{|X^\star|}^\star\right\} \subset \Omega$ set of solutions generated by NSMA (Algorithm 2), $q > 1 \in \mathbb{N}$ batch size.

1: **if** *Clustering in objectives space* **then**
2: Let $F^\star = \left\{f_1^\star, \ldots, f_{|X^\star|}^\star\right\} \subset \mathbb{R}^2$ be the image of $X^\star$ through $\mathcal{F}(\cdot)$
3: Perform K-MEANS on $F^\star$ to find q clusters
4: Let $c_{1:q}$ be the q resulting cluster centers
5: **for all** $i \in \{1, \ldots, q\}$ **do**
6: Let $z_i = \underset{z \in \{1,\ldots,|X^\star|\}}{\arg\min} \|f_z^\star - c_i\|$
7: Let $x_i = x_{z_i}^\star$
8: **end for**
9: **else** *(Clustering in variables space)*
10: Perform K-MEANS on $X^\star$ to find q clusters
11: Let $c_{1:q}$ be the q resulting cluster centers
12: Let $x_{1:q} = c_{1:q}$
13: **end if**
14: **return** $x_{1:q}$

on the objectives space allows a more uniform choice from the Pareto front, regardless the problem dimensionality.

Clustering in the variables space In the second clustering strategy [Lines 10-12, Algorithm 3], we perform the K-MEANS algorithm [59, 60] to find q clusters in the variables space [Line 10, Algorithm 3]. The q resulting cluster centers are taken as the new candidates for the original problem (1) [Lines 11-12, Algorithm 3].

Although clustering in the objectives space seems to be more intuitive than clustering in the variables space, the latter could be more effective in high-dimensional scenarios ($n > 20$), where a blind exploration of the feasible set boundaries is usually carried out employing the standard acquisition functions (Sect. 2.1.1). This behavior on high-dimensional problems is well-known in the literature and is called the *boundary issue* [13, 14]. In fact, in high dimensional spaces, it happens very frequently that along the boundaries of the feasible region many points are characterized by high $\sigma_k^2(\cdot)$ values (uncertainty). Thus, most methods based on a compromise between the two objectives, such as q-EI and q-LCB, tend to generate candidate points on the border of the feasible region. This fact can be considered as one of the main reasons why Bayes-Opt methods have been applied in the past only to very low–dimensional problems, where this effect is less relevant. Clustering in the variables spaces could be an *heuristic* solution to the boundary issue: clustering points in the boundaries, far from each other, could end up with setting some cluster centers in the interior of the feasible set, thus allowing to collect information about $f(\cdot)$ in more internal regions; at the next iteration, this information would lead to more informative posterior mean and variance functions and, then, to a better exploitation of the Pareto front reconstruction. In Sect. 4.4, we report some numerical results supporting our arguments.

4 Computational Experiments

In this section, we report the results of thorough computational experiments, mainly comparing the proposed acquisition strategy with state-of-the-art ones from the batch Bayes-Opt literature. The experimental code was written in `Python3`[1] and was run on a computer with the following characteristics: Ubuntu 22.04 OS, Intel(R) Core(TM) i5-10600KF 6 cores 4.10GHz, 32 GB RAM.

4.1 Experimental Settings

In this section, we report the tested acquisition strategies, the functions used for the comparisons and, finally, the employed metrics.

4.1.1 Bayes-Opt Procedure and Acquisition Strategies

For the implementation of the Bayes-Opt procedure (Algorithm 1), including the Gaussian Process regression and the optimization of the standard acquisition functions, we relied on the `BoTorch`[2] library [17]. In our experimentation, we used the Matérn kernel (Sect. 2.1) with $\nu = 5/2$. At each iteration of the Bayes-Opt procedure, the acquisition strategies were run with 100 initial points, uniformly chosen in the feasible set Ω. Regarding the Gaussian Process, we used all the default settings and the recommendations proposed in the `BoTorch` library. Both in the Gaussian Process regression and in the evaluations of the posterior mean and variance functions, the solutions were first scaled in $[0, 1]$ considering the problem bounds; moreover, for numerical stability, the observed $f(\cdot)$ values were min-max scaled before the Gaussian Process regression.

Throughout the section, we indicate with `NSMA` the BOO based acquisition strategy proposed in Sect. 3 of this manuscript. For the `NSMA` implementation we used the code[3] distributed by its authors; moreover, we employed the Gurobi Optimizer (Version 10) [61] in order to solve instances of the search direction problem (B.1).

We decided also to test our methodology with different solvers: `NSGA-II` [41], `DMS` [40] and `MOSQP` [36]. As mentioned in Sect. 1, `NSGA-II` has been already tested to solve a BOO problem similar to (2). Moreover, `NSGA-II` is arguably one of the most representative evolutionary algorithms for hard MOO problems. Thus, in this context, it represents the most direct competitor. As for `DMS` and `MOSQP`, they are among the state-of-the-art derivative-free and gradient-based solvers, respectively; thus, a comparison with these approaches is needed to test the goodness of `NSMA` as solver in our acquisition strategy.

For `NSMA` and `NSGA-II`, we set the population size $N = 100$ and a maximum number of iterations equal to 20; for `NSMA` the parameter $n_{opt} = 5$ and $Q = 0.9$ (see Appendix B); the other parameters values were set as recommended in [41, 42]. As for

[1] The implementation code can be found at github.com/FranciC19/biobj_acquistion_function_for_BO.

[2] The `BoTorch` code can be found at github.com/pytorch/botorch.

[3] The `NSMA` code can be found at github.com/pierlumanzu/nsma.

DMS the hyper-parameters were set according to [40] and the released source code,[4] while for MOSQP they were based on [36] and the related distributed code;[5] in order to solve the SQP problems within the approach, the Ipopt software package [62] was employed. Note that for the SQP problems MOSQP employs the hessian of the functions $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$, which are defined since the Matérn kernel with $\nu = 5/2$ is two-times continuously differentiable (see Sect. 2.1).

All the algorithms were tested with the two clustering approaches proposed in Sect. 3.2 for the Pareto front points selection. The acronyms (X) and (F) indicate if K-MEANS was performed in the variables space or in the objectives one, while (R) indicate a random selection of points from the generated Pareto front.

Our methodology was further compared with three other recently proposed acquisition strategies: BBO-ABAFMo [46], MACE [50, 51] and TURBO [32, 63]. A short introduction of these approaches can be found in Sect. 1. For all of them, a source code.[6,7,8] is available online; the hyper-parameters were chosen according to the latter. For BBO-ABAFMo and MACE, the total number of iterations available for the employed MOO algorithm at each Bayes-Opt procedure iteration is fairly set to 20; as for TURBO, we tested both the variants with one local model, known as TURBO-1 in the referenced manuscript, and with multiple models, TURBO-M; both the variants were run with the default parameters indicated in [32] and in the published code.

Finally, we considered as competitors the acquisition strategies involving the optimization of q-EI, q-LCB and q-LOG-EI [64], a recently proposed variant of q-EI whose members are easier to optimize numerically and have either identical or approximately equal optima as the original. A brief description of q-EI and q-LCB can be found in Sect. 2.1.1. For the integrals estimation, we employed the Sobol Quasi Monte-Carlo Normal Sampler with 512 samples. As suggested in [26], for q-LCB we set the parameter $\beta = \sqrt{3}$. For all the acquisition strategies, we set a maximum number of iterations for the solver L-BFGS-B [53] equal to 100.

In most of the comparisons, we also employed as baseline the results obtained by *Random Search* (RS), whose mechanism consists on randomly selecting q points from the feasible set Ω at each iteration of the Bayes-Opt procedure.

4.1.2 Benchmark Functions

The functions[9] employed for the comparisons are listed in Table 1. As mentioned in Sects. 1-2.1, all the evaluations were supposed to be noise-free. Note that some of the tested functions are scalable w.r.t. the dimensionality n; this feature was crucial in our experimentation to compare the acquisition strategies in (hard) high-dimensional scenarios, where additional difficulties arise (e.g., the boundary issue; for more details on the topic, we refer the reader to Sect. 3.2).

[4] The DMS code can be found at mat.uc.pt/dms.

[5] The MOSQP code can be found at norg.uminho.pt/aivaz/MOSQP

[6] The BBO-ABAFMo code can be found at github.com/JixiangChen-Jimmy/BBO-DMoAF

[7] The MACE code can be found at github.com/Alaya-in-Matrix/pyMACE.

[8] The TURBO code can be found at github.com/uber-research/TuRBO.

[9] The formulations, as well as other information, of some of the tested functions can be found at https://www.sfu.ca/~ssurjano/index.html.

Table 1 Benchmark functions used for the experiments. For each of them, we indicate: the source paper; the name; the tested values for the dimensionality n; the optimal value; if it has/has not the zero solution as a global minimum; the lower and upper bounds

Source	Name	n	$f^\star$	$x^\star = 0_n$	Bounds
[65]	Rastrigin	2, 4, 10, 20, 50, 100	0.0	✓	$[-5.12, 5.12]^n$
[66]	Hartmann	3	-3.86278	✗	$[0.0, 1.0]^n$
		6	-3.32237	✗	$[0.0, 1.0]^n$
[67]	Rosenbrock	2, 4, 10, 20, 50, 100	0.0	✗	$[-5, 10]^n$
[68]	Ackley_1	2, 4, 10, 20, 50, 100	0.0	✓	$[-32.768, 32.768]^n$
[69]	Alpine_1	2, 4, 10, 20, 50, 100	0.0	✓	$[-10, 10]^n$
[70]	Branin	2	0.397887	✗	$x_1 \in [-5, 10]\ x_2 \in [0, 15]$
[71]	Schwefel	2, 4, 10, 20, 50, 100	0.0	✗	$[-500, 500]^n$
[72]	Levy_8	2, 4, 10, 20, 50, 100	0.0	✗	$[-10, 10]^n$
[73]	Michalewicz	2	-1.80130341	✗	$[0, \pi]^n$
		5	-4.687658	✗	$[0, \pi]^n$
		10	-9.66015	✗	$[0, \pi]^n$
[70]	SixHumpCamel	2	-1.0316	✗	$x_1 \in [-3, 3]\ x_2 \in [-2, 2]$
[74]	Bukin_6	2	0.0	✗	$x_1 \in [-15, -5]\ x_2 \in [-3, 3]$
[75]	StyblinskiTang	2, 4, 10, 20, 50, 100	$-39.166166n$	✗	$[-5, 5]^n$
[76]	HolderTable_2	2	-19.2085	✗	$[-10, 10]^n$
	EggHolder	2	-959.6407	✗	$[-512, 512]^n$
[77]	Shekel ($m = 10$)	4	-10.5363	✗	$[0, 10]^n$

For each function, we started the Bayes-Opt procedure with $N_i = 10$ points randomly chosen in Ω; the same initial solutions were considered for all the acquisition strategies. We mainly tested the methodologies with the batch size parameter $q = 3$: ideally, at each iteration of the Bayes-Opt procedure, we would like to obtain as candidates for the original problem (1) two points with good values for $\mu_k(\cdot)$ (exploitation) and for $\sigma_k^2(\cdot)$ (exploration), respectively, and one middle trade-off solution. For the sake of completeness, we also experimented other values for q; the results are commented at the end of the experimental section. In the Bayes-Opt procedure, we set the number of function evaluations (excluding the first N_i ones) $N_M = 60$; thus, considering that $q = 3$, for each function we generated 20 point batches.

For each acquisition strategy and function, we run the execution 20 times, each of which characterized by a different seed for the pseudo-random number generator, so as to reduce the experiments sensibility to the random operations. The results were then obtained by calculating the averages of the metrics values achieved in the twenty tests.

4.1.3 Metrics

We denote by f_k^{best} the best observed objective function value after $k+N_i$ evaluations; moreover, we indicate with f_L^{best} the best $f(\cdot)$ value found at the end of the Bayes-Opt procedure. According to Algorithm 1, the two metrics can be formalized in the following way:

$$f_k^{best} = \begin{cases} \min\limits_{i \in \{1,\dots,N_i\}} f(x_i), & \text{if } k = 0 \\ \min\left\{ f_{k-1}^{best}, f(x_k) \right\}, & \text{otherwise} \end{cases} \tag{12}$$

and $f_L^{best} = f(x^\star)$.

In order to observe the performance trend of the methodologies between one Bayes-Opt procedure iteration and another, the values obtained in terms of f_k^{best} are plotted w.r.t. the number of objective function evaluations k. In the plots, vertical bars are also shown, representing the 95% confidence interval [17] over the f_k^{best} values obtained in the twenty runs.

Another metric we employed is a normalized version of the *regret* [16]:

$$NR_k = \frac{f_k^{best} - f^\star}{f_0^{best} - f^\star}.$$

We trivially deduce that: for all k, $NR_k \geq 0$; lower values for this metric mean better performance. In our experimentation, the NR_k values obtained for all k were summarized by calculating the *AUC* (Area Under Curve); the resulting metric is called *NR-AUC*; like NR_k, small values of *NR-AUC* are associated with good performance. In brief, we would like to minimize the regret at the end of the optimization, but at the same time we would also like to converge in a few iterations to the final solution: this is particularly relevant when dealing with expensive-to-evaluate objective functions.

Lastly, we employed the variant introduced in [78] of the performance profiles [79]. Given a solver and a certain metric, the new tool allows to plot the (cumulative)

distribution function of the relative gap between the score achieved by that solver and the best score among those obtained by all the considered solvers. Formally, let us consider a benchmark of problems $\mathcal{P}$, a set of solvers $\mathcal{S}$ and a metric m such that, for each $\pi \in \mathcal{P}$ and $\sigma \in \mathcal{S}$, $m_{\pi,\sigma}$ indicates the metric value for solver σ on problem π. The relative gap cumulative distribution for a solver σ is given by

$$p_\sigma(\tau) = \frac{1}{|\mathcal{P}|} \left| \left\{ \pi \in \mathcal{P} \;\middle|\; \frac{\left| m_{\pi,\sigma} - \min_{\tilde{\sigma} \in \mathcal{S}} m_{\pi,\tilde{\sigma}} \right|}{\left| \min_{\tilde{\sigma} \in \mathcal{S}} m_{\pi,\tilde{\sigma}} \right|} \leq \tau \right\} \right|,$$

which represents the estimated probability that the relative gap for a solver σ on an arbitrary problem π is within a factor $\tau \in \mathbb{R}$. Note that $p_\sigma(1)$ is the fraction of problems where solver σ had the best performance. For a more technical explanation, we refer the reader to [78]. Similar to the performance profiles, the relative gap cumulative distributions thus allow to appreciate the relative performance and robustness of the considered algorithms; moreover, they result to be particularly useful when evaluating metrics involving objective function values, such as f_k^{best} and *NR-AUC*.

4.2 Preliminary Comparison of the Bi-Objective Optimization Based Acquisition Strategy with `q-LCB` and with different solvers

In this section of computational experiments, we first compare the performance of our BOO based acquisition strategy and `q-LCB` on some selected functions. Note that `q-LCB` (10) is an extension for batch Bayesian optimization of `LCB`, which can be seen as a scalarization of the following BOO problem: $\min_{x \in \Omega}(\mu_k(x), -\sigma_k(x))^\top$, similar to the one we use in our strategy; thus, making such a comparison can be useful to appreciate the strengths of reconstructing the Pareto front to choose the next candidate solutions for problem (1). In Fig. 2, we report the f_k^{best} metric values achieved by `NSMA(F)`, `NSGA-II(F)`, `q-LCB` and `RS` w.r.t. the number of function evaluations k.

On low-dimensional scenarios ($n \leq 6$), the BOO based acquisition strategy turned out to be effective in 2 out of 3 problems; from the very first iterations `NSGA-II(F)` managed to find better solutions than its competitors; moreover, both `NSMA(F)` and `NSGA-II(F)` allowed to find better solutions than `q-LCB` at the end of the Bayes-Opt procedure. In the third problem, i.e., Hartmann, `NSMA(F)` and `NSGA-II(F)` were outperformed by `q-LCB`, but their performance still remained competitive. Unsurprisingly, `RS` overall turned out to be the worst methodology; the same also holds on the high-dimensional problems.

When $n \geq 20$, `NSMA(F)` is the clear winner. In particular, two results of the memetic algorithm have to be highlighted. The first one is the increasing performance gap between `NSMA(F)` and `q-LCB` as the value for n increases, highlighting the difficulty in `q-LCB` of setting a value for the weight β appropriate for every possible dimensionality. Such characteristic has been already anticipated in Sect. 3.2. The second result is that `NSMA(F)` clearly outperformed `NSGA-II(F)`. As men-

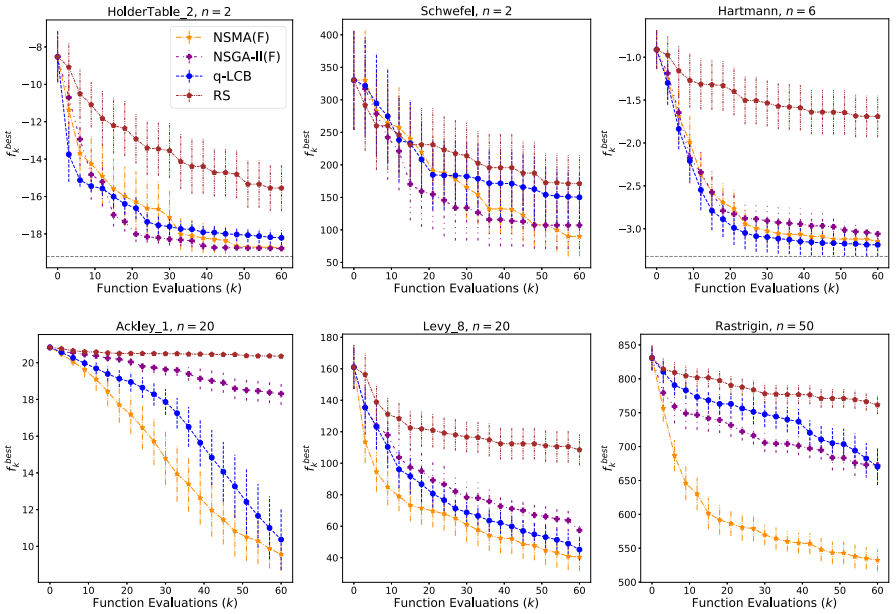


Fig. 2 Plots of the f_k^{best} metric w.r.t. the number of function evaluations k for `NSMA(F)`, `NSGA-II(F)`, `q-LCB` and `RS` on a set of selected functions (see Table 1). When $f^\star \neq 0$, it is represented by a gray dashed line

tioned in [42, Section 4], for large values for n, `NSMA` exploited the combined use of the `NSGA-II` exploration capabilities and the gradient-descent tools to get better and wider Pareto front approximations than the genetic approach. This behavior also occurred in the considered Bayes-Opt context, allowing `NSMA(F)` to propose better solutions for the objective function $f(\cdot)$ evaluation.

Next, we compare `NSMA`, `NSGA-II`, `DMS` and `MOSQP` as solvers for our acquisition strategy. We decided, only for this experimentation, to use for the execution of the solvers at each iteration of the Bayes-Opt procedure a stopping criterion based on a time limit of 15 seconds rather than on a maximum number of iterations. We indeed think that a time-based stopping condition is the fairest way, among the "conventional" ones, to compare algorithms presenting radically different structures: they perform a different number of search steps during the iterations; moreover, some of them require to calculate the gradient, an operation that could have a different cost and, thus, difficult to quantify if we rely on the number of evaluations of the mean ($\mu_k(\cdot)$) and variance ($\sigma_k^2(\cdot)$) functions. We are then aware that the choice of the stopping criterion may influence the outcomes of the approaches. A future study of such influence might thus be worth of future investigations. In Fig. 3, we report the f_k^{best} metric values achieved by `NSMA(F)`, `NSGA-II(F)`, `DMS(F)` and `MOSQP(F)` w.r.t. the number of function evaluations k on Alpine_1 and Schwefel functions.

When $n = 2$, all the four algorithms performed well, being competitive from the very first iterations. On the other hand, for $n > 2$ we have to examine the results on Alpine_1 and the ones on Schwefel separately. On Alpine_1, `MOSQP` was the

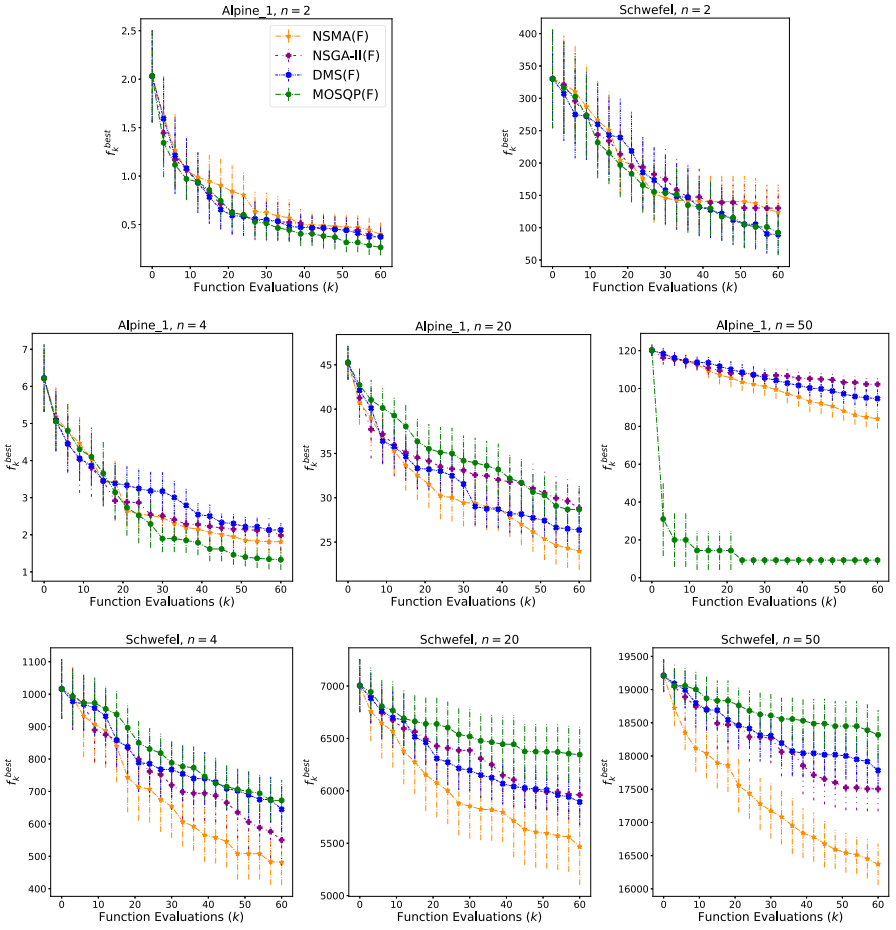


Fig. 3 Plots of the f_k^{best} metric w.r.t. the number of function evaluations k for `NSMA(F)`, `NSGA-II(F)`, `DMS(F)` and `MOSQP(F)` on the Alpine_1 and Schwefel functions (see Table 1)

best algorithm overall, reaching good performance especially for $n = 50$; in this scenario, `NSMA` turned out to be the second algorithm, winning only when $n = 20$; as for `DMS` and `NSGA-II`, as the value for n increased the derivative-free approach outperformed the genetic approach. The results on Alpine_1 are justified by the `MOSQP` operations performed at the beginning of each execution [36], whose aim is to find the extremes of the Pareto front of the BOO problem (2) starting the search from the center of the bounding box; this mechanism obviously favors `MOSQP` on functions, like Alpine_1, where the box center coincides with the global minimum. This argument is also supported by the results of the algorithms on the Schwefel function for $n > 2$, where the optimum is nearer to the problem boundaries: in this scenario, the situation is the opposite, with `NSMA` outperforming all the other solvers and `MOSQP` being the worst algorithm. Regarding `DMS` and `NSGA-II`, the performance on Schwefel are similar, with `NSGA-II` being slightly better for $n \in \{4, 50\}$. We conclude that: if we

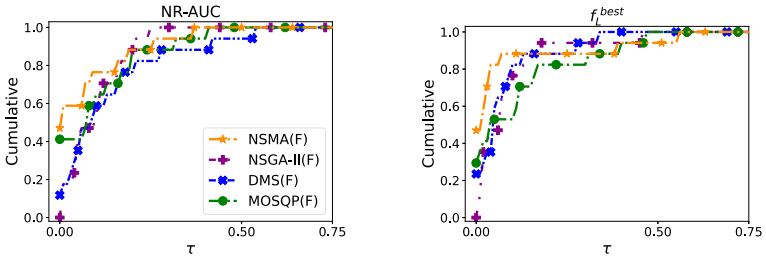


Fig. 4 Plots of the relative gap cumulative distribution for `NSMA(F)`, `NSGA-II(F)`, `DMS(F)` and `MOSQP(F)` on Hartmann, Branin, Schwefel, EggHolder, StyblinskiTang and HolderTable functions (see Table 1)

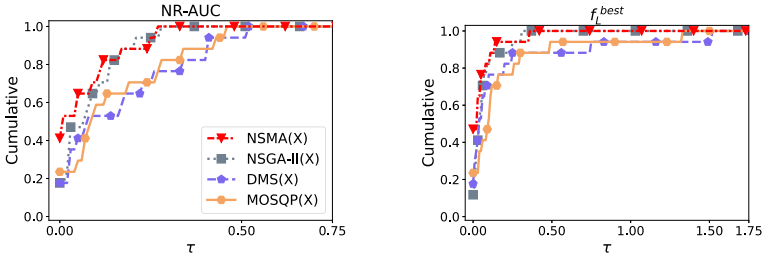


Fig. 5 Plots of the relative gap cumulative distribution for `NSMA(X)`, `NSGA-II(X)`, `DMS(X)` and `MOSQP(X)` on Hartmann, Branin, Schwefel, EggHolder, StyblinskiTang and HolderTable functions (see Table 1). Note that the intervals of the x-axis are set for a better visualization of the numerical results

have prior knowledge that the global optimum is near the center of the bounds, `MOSQP` can be a good solution; however, in practice, this information is typically not available and, in these cases, `NSMA` seems to offer the best trade-off when the optimum is either close to or far from the bounding box center.

We conclude the section showing in Figs. 4-5 the relative gap cumulative distributions for `NSMA(F)`, `NSGA-II(F)`, `DMS(F)`, `MOSQP(F)` and `NSMA(X)`, `NSGA-II(X)`, `DMS(X)`, `MOSQP(X)`, respectively. In order not to advantage any algorithm w.r.t. an other one, we only considered the functions where the optimum is not at the center of the bounding box, i.e., Hartmann, Branin, Schwefel, EggHolder, StyblinskiTang and HolderTable.

In Fig. 4, we observe that all the four solvers performed well in terms of robustness, while they did not have the same effectiveness as `NSMA(F)`. Overall, we can see that the tested gradient-based multi-objective optimization methods `NSMA` and `MOSQP` turned out to be more effective than `DMS` and `NSGA-II`, confirming our thesis on the benefits of exploiting first-order information of $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$ to build the Pareto front of the BOO problem (2). Finally, we can also conclude that, although gradient computation makes `NSMA` more computationally expensive than `NSGA-II`, in practice it allowed `NSMA` to get better performance w.r.t. the genetic approach, a result also confirmed in [42].

Conclusions similar to the ones related to Fig. 4 can be deduced from Fig. 5, where the clustering approach operates on the variables space.

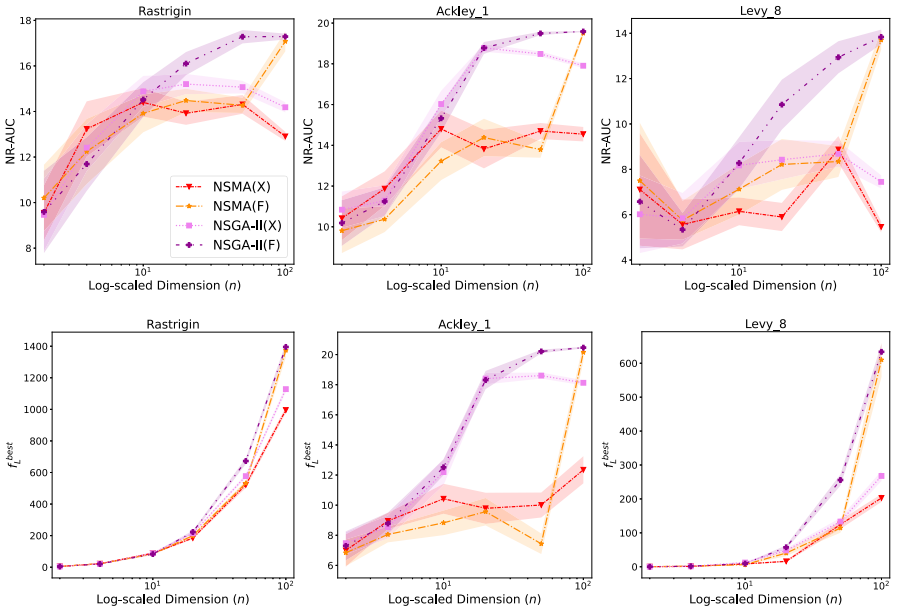


Fig. 6 Plots of the *NR-AUC* and f_L^{best} metrics values achieved by `NSMA(X)`, `NSMA(F)`, `NSGA-II(X)` and `NSGA-II(F)` on Rastrigin, Ackley_1 and Levy_8 (see Table 1) for values of $n \in \{2, 4, 10, 20, 50, 100\}$. The shaded areas indicate the 95% confidence intervals over the metric values obtained in the twenty runs. For a better visualization, the x-axis is log-scaled

4.3 Evaluation of the BOO Based Acquisition Strategy Variants

Attested the goodness of both our BOO based acquisition strategy and `NSMA` as solver, we now focus on the scalability of `NSMA(X)`, `NSMA(F)`, `NSGA-II(X)` and `NSGA-II(F)`, that is, how well they perform as the value for the dimensionality n increases. In Fig. 6, the *NR-AUC* and f_L^{best} metrics values achieved by the four methodologies on a subset of functions tested with $n \in \{2, 4, 10, 20, 50, 100\}$ are shown.

While for low values of n the performance of the four approaches was similar, for high dimensionalities `NSMA(X)` and `NSGA-II(X)` got better overall results than `NSMA(F)` and `NSGA-II(F)`, respectively. The situation is particularly stressed when $n = 100$, where the gap between the first two methodologies and the second ones is clear. As mentioned in Sect. 3.2, clustering in the variables space could be more effective against the *boundary issue*, allowing to better escape from evaluating points always near to the problem boundaries. The results seem to confirm this hypothesis: exploring feasible space regions different from the ones containing the problem boundaries, `NSMA(X)` and `NSGA-II(X)` managed to find better solutions for the original problem (1). We again observe the overall superiority of `NSMA` w.r.t. `NSGA-II`, regardless the employed clustering approach. Thus, after analyzing the results of this section and the previous one, we decided to use only `NSMA(X)` and `NSMA(F)` for the rest of the experiments.

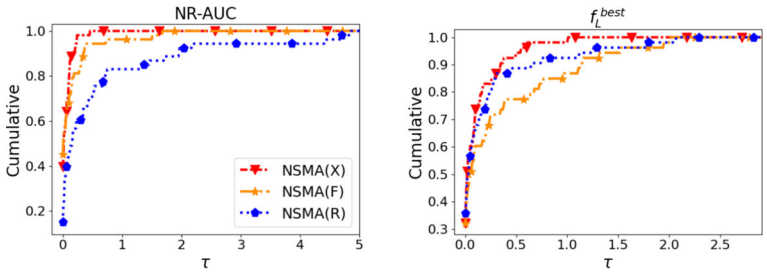


Fig. 7 Plots of the relative gap cumulative distribution for `NSMA(X)`, `NSMA(F)` and `NSMA(R)` considering all the benchmark functions listed in Table 1. Note that the intervals of the x-axis are set for a better visualization of the numerical results

We conclude the section evaluating the effectiveness of our clustering approaches w.r.t. the random selection of points from the Pareto front generated by `NSMA`; in according to the experimental settings, we call this last methodology `NSMA(R)`. In Fig. 7, we report the relative gap cumulative distributions obtained in terms of *NR-AUC* and f_L^{best} by the three methodologies (`NSMA(X)`, `NSMA(F)` and `NSMA(R)`) on all the benchmark functions. We observe that: `NSMA(X)` was the best algorithm w.r.t. both the metrics; `NSMA(F)` outperformed `NSMA(R)` in terms of *NR-AUC*, while it is the opposite w.r.t. f_L^{best}. This last result is particularly interesting, as it suggests that `NSMA(F)` managed to obtain better solutions than `NSMA(R)` from the very first iterations, while the latter obtained solutions of higher quality only at the end of the Bayesian procedure execution. A possible justification for this fact may be that `NSMA(F)` is more prone to getting trapped in a local minimum of the objective function. In such cases, the random selection of new candidates by `NSMA(R)` may facilitate escaping these points and allow the optimization process to proceed more effectively. An integration of the two selection mechanisms might thus be worth of future investigations.

4.4 Comparisons with the State-of-the-art Acquisition Strategies

In this section, we investigate the performance of `NSMA(X)`, `NSMA(F)`, `q-EI`, `q-LOG-EI`, `q-LCB` and `RS` more specifically. In Fig. 8, we report the plots of the f_k^{best} metric values obtained by the six acquisition methodologies on a set of selected functions, some of which were already addressed in Sect. 4.2.

In the low-dimensional instances ($n \leq 6$), the proposed acquisition strategies were competitive: `NSMA(X)` and `NSMA(F)` had a similar trend as `q-LCB`, especially performing well in terms of f_L^{best}. Unlike the three mentioned approaches, in these functions `q-EI` and `q-LOG-EI` seem to struggle, having competitive performance but never outperforming its competitors. Also in this comparison, `RS` turned out to be the worst acquisition strategy overall, regardless the value for n.

As the value for n increased, `NSMA(X)` and `NSMA(F)` turned out to be the best algorithms overall. The standard acquisition strategies `q-EI` and `q-LCB` still had good results on some scenarios: both approaches had similar behavior as our proposals on

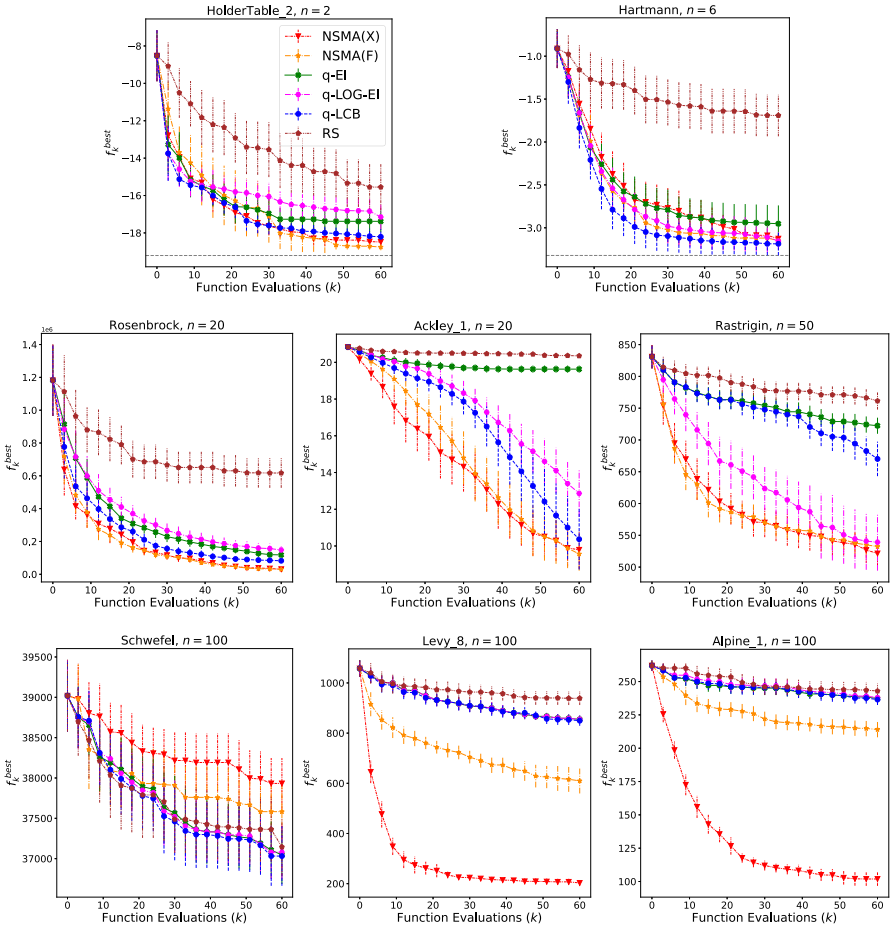


Fig. 8 Plots of the f_k^{best} metric w.r.t. the number of function evaluations k for `NSMA(X)`, `NSMA(F)`, `q-EI`, `q-LOG-EI`, `q-LCB` and `RS` on a set of selected functions (see Table 1). When $f^\star \neq 0$, it is represented by a gray dashed line

the Rosenbrock function with $n = 20$, and a competitive result was also obtained by `q-LCB` on the Ackley_1 function at the same dimensionality. However, for higher values of n, the general superiority of `NSMA(X)` and `NSMA(F)` become clear. As for `q-LOG-EI`, it improved the performance of `q-EI` on Ackley_1 with $n = 20$ and on Rastrigin with $n = 50$, confirming that it is more numerically stable as acquisition function to optimize. However, it did not reach the same performance as `NSMA(X)` and `NSMA(F)`; moreover, for $n = 100$ it generally struggled as `q-EI` in finding good candidates where to evaluate the real objective function $f(\cdot)$.

Fig. 8 is supported by Tables 2-3, where the values obtained by the six methodologies in terms of *NR-AUC* and f_L^{best}, along with the standard deviations, are reported. In low-dimensional cases, `NSMA` turned out to be competitive, especially when clustering was performed on the objectives space (`NSMA(F)`). In the Schwefel function with

Table 2 NR-AUC and f_L^{best} average values with standard deviations achieved by NSMA(X), NSMA(F), q-EI, q-LOG-EI, q-LCB and RS over 20 runs on the first set of selected functions shown in Figure 8. The marked-in-bold values and the underlined ones are, respectively, the best and second best for a given metric on a considered function

Function	Method	NR-AUC	f_L^{best}
HolderTable_2 ($n = 2$)	NSMA(X)	4.47(±1.725)	−18.484(±0.836)
	NSMA(F)	4.891 (±3.622)	**−18.758 (±0.387)**
	q-EI	5.473 (±2.44)	−17.379 (±1.341)
	q-LOG-EI	6.316 (±2.513)	−17.15 (±1.517)
	q-LCB	**4.461 (±1.319)**	−18.208 (±0.925)
	RS	11.074 (±4.833)	−15.548 (±2.814)
Hartmann ($n = 6$)	NSMA(X)	6.292 (±3.095)	−3.127 (±0.103)
	NSMA(F)	5.092(±2.352)	−3.149 (±0.073)
	q-EI	6.282 (±3.726)	−2.953 (±0.487)
	q-LOG-EI	5.319 (±3.274)	−3.154(±0.235)
	q-LCB	**4.229 (±2.894)**	**−3.187 (±0.306)**
	RS	15.41 (±4.599)	−1.691 (±0.554)
Rosenbrock ($n = 20$)	NSMA(X)	3.385(±1.487)	30865.769(±14710.851)
	NSMA(F)	**3.346 (±1.196)**	**30617.591 (±19354.843)**
	q-EI	5.611 (±1.769)	118150.206 (±44968.612)
	q-LOG-EI	6.121 (±2.379)	148337.216 (±63702.058)
	q-LCB	4.388 (±1.908)	82420.471 (±28912.746)
	RS	12.515 (±3.198)	617075.926 (±206017.299)
Ackley_1 ($n = 20$)	NSMA(X)	**13.817 (±2.164)**	9.79(±2.359)
	NSMA(F)	14.384(±2.099)	**9.566 (±2.08)**
	q-EI	19.078 (±0.389)	19.629 (±0.448)
	q-LOG-EI	16.994 (±1.307)	12.865 (±2.902)
	q-LCB	16.081 (±1.927)	10.37 (±3.774)
	RS	19.685 (±0.213)	20.351 (±0.22)

$n = 100$, q-LCB, q-EI and q-LOG-EI were the best algorithms in terms of both metrics; in all the other high-dimensional instances, the memetic method outperformed the competitors, with NSMA(X) being by far the most effective methodology.

The results on the instances with $n = 100$ are worth mentioning separately. On the Schwefel function, our approaches struggled to reach the performance of the competitors. A possible justification of this fact is the Schwefel function global optimum position within the feasible set, which is quite near to the boundary. As mentioned in Sect. 3.2, NSMA(X) represents an heuristic solution to the *boundary issue*: it could allow to escape from a massive exploration of the boundaries, which is a typical behavior on high-dimensional instances of acquisition strategies based on the posterior mean ($\mu_k(\cdot)$) and standard deviation ($\sigma_k(\cdot)$) functions. By design, NSMA(X) will be thus driven to find good solutions in the interior. NSMA(F) has a similar, albeit less pronounced, behavior; indeed, it obtained better results than NSMA(X) in the

Table 3 NR-AUC and f_L^{best} average values with standard deviations achieved by `NSMA(X)`, `NSMA(F)`, `q-EI`, `q-LOG-EI`, `q-LCB` and `RS` over 20 runs on the second set of selected functions shown in Fig. 8. The marked-in-bold values and the underlined ones are, respectively, the best and second best for a given metric on a considered function

Function	Method	NR-AUC	f_L^{best}
Rastrigin ($n = 50$)	`NSMA(X)`	14.311(±0.934)	**521.437 (±43.237)**
	`NSMA(F)`	**14.269 (±0.762)**	532.328(±36.479)
	`q-EI`	18.206 (±0.592)	722.386 (±28.876)
	`q-LOG-EI`	15.409 (±1.689)	539.156 (±102.2)
	`q-LCB`	17.926 (±0.738)	670.171 (±61.911)
	`RS`	18.921 (±0.647)	761.45 (±31.155)
Schwefel ($n = 100$)	`NSMA(X)`	19.65 (±0.355)	37932.468 (±718.743)
	`NSMA(F)`	19.455 (±0.423)	37582.778 (±724.76)
	`q-EI`	19.337 (±0.371)	37044.992(±819.658)
	`q-LOG-EI`	19.333(±0.345)	37080.634 (±760.139)
	`q-LCB`	**19.309 (±0.391)**	**37032.126 (±841.883)**
	`RS`	19.341 (±0.474)	37144.518 (±811.363)
Levy_8 ($n = 100$)	`NSMA(X)`	**5.46 (±0.556)**	**202.875 (±26.346)**
	`NSMA(F)`	13.699(±0.965)	609.754(±114.544)
	`q-EI`	17.456 (±0.648)	854.655 (±33.536)
	`q-LOG-EI`	17.457 (±0.649)	855.295 (±35.182)
	`q-LCB`	17.4 (±0.765)	849.608 (±46.816)
	`RS`	18.346 (±1.224)	938.743 (±61.101)
Alpine_1 ($n = 100$)	`NSMA(X)`	**10.057 (±0.45)**	**101.89 (±11.758)**
	`NSMA(F)`	17.301(±0.759)	214.062(±13.418)
	`q-EI`	18.735 (±0.666)	237.291 (±8.674)
	`q-LOG-EI`	18.863 (±0.748)	238.003 (±8.688)
	`q-LCB`	18.753 (±0.758)	236.497 (±9.304)
	`RS`	19.069 (±0.778)	242.893 (±12.699)

Schwefel function, but, at the same time, it did not perform well enough to reach the performance of `q-EI`, `q-LOG-EI` and `q-LCB`.

As a proof of the `NSMA(X)` behavior, in Fig. 9 we plot the component-wise minimum distance from the problem boundaries obtained by the six methodologies after each observed point x_k in the Bayes-Opt procedure (Algorithm 1); note that this distance was always compared with the one of the previous point x_{k-1} and the maximum was taken. The resulting metric is formalized in the following equation:

$$d_k^{\Omega} = \begin{cases} \max\limits_{\bar{k}\in\{1,\ldots,N_i\}} \min\limits_{i\in\{1,\ldots,n\}} \min\left\{x_{\bar{k}}^i - l^i, u^i - x_{\bar{k}}^i\right\}, & \text{if } k = 0 \\ \max\left\{d_{k-1}^{\Omega}, \min\limits_{i\in\{1,\ldots,n\}} \min\left\{x_k^i - l^i, u^i - x_k^i\right\}\right\}, & \text{otherwise} \end{cases}. \quad (13)$$

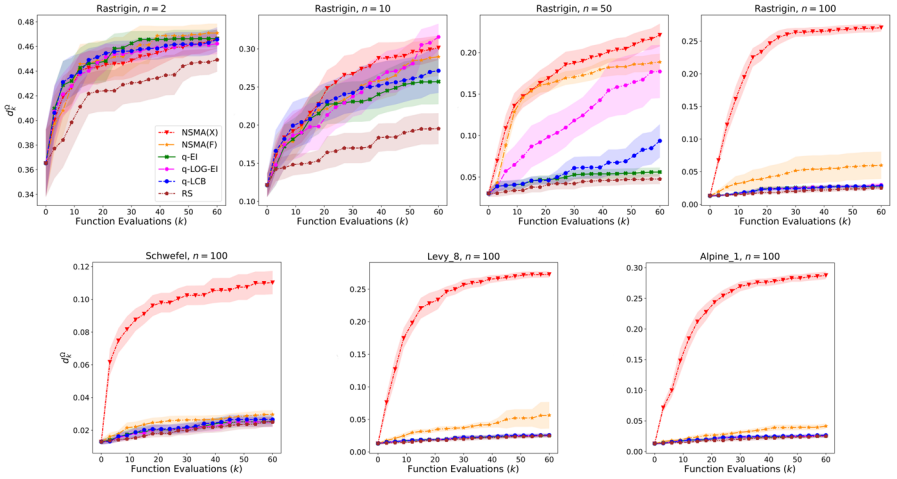


Fig. 9 Plot of d_k^Ω (13) w.r.t. the number of function evaluations k for `NSMA(X)`, `NSMA(F)`, `q-EI`, `q-LOG-EI`, `q-LCB` and `RS` on a set of selected functions (see Table 1). The shaded areas indicate the 95% confidence intervals over the metric values obtained in the twenty runs

First, we can observe that, in the low-dimensional Rastrigin instances ($n \leq 10$), all the acquisition strategies allowed to explore points distant from the boundaries from the very first iterations. The situation is different in the high-dimensional scenarios, where the effects of the boundary issue can be noticed on the performance of `q-EI` and `q-LCB`. Unlike the two mentioned approaches, `NSMA(F)` and `q-LOG-EI` managed to escape from the boundaries in the $n = 50$ case; however, in the hardest scenarios ($n = 100$), even these methods struggled. The only methodology capable of finding points far from the boundaries regardless the problem dimensionality was `NSMA(X)`, with remarkable performance w.r.t. the competitors on the $n = 100$ case. In the Schwefel function, `q-EI`, `q-LOG-EI` and `q-LCB` took advantage of the blind exploration of the problem boundaries, reaching solutions that, however, are extremely far from the global optimum of the problem. Moreover, the boundary exploration cannot be beneficial with the benchmark functions which present a global optimum in a more internal position of the feasible set. In these last scenarios, `NSMA(X)` and `NSMA(F)` proved to be highly effective: on the Levy_8 and Alpine_1 functions with $n = 100$ (Fig. 8), the gap between the two `NSMA`-based acquisition strategies and the standard ones is clear; in particular, we can observe the remarkable results obtained by `NSMA(X)` w.r.t. all the other competitors.

In order to support what has been stated above in Sects. 3.2-4.2, we conclude the section with an analysis on `NSMA(F)`, which, without a proper heuristic for the *boundary issue*, still present superior performance w.r.t. the state-of-the-art competitors as the problem dimension grows. In Fig. 10, we show the Pareto front obtained by `NSMA` at iteration 1, 10 and 20 of the Bayesian optimization procedure, along with the candidate points chosen by `NSMA(F)`, `q-LCB`, `q-EI` and `q-LOG-EI`, on Hartmann ($n = 6$) and Ackley ($n = 20$) problems.

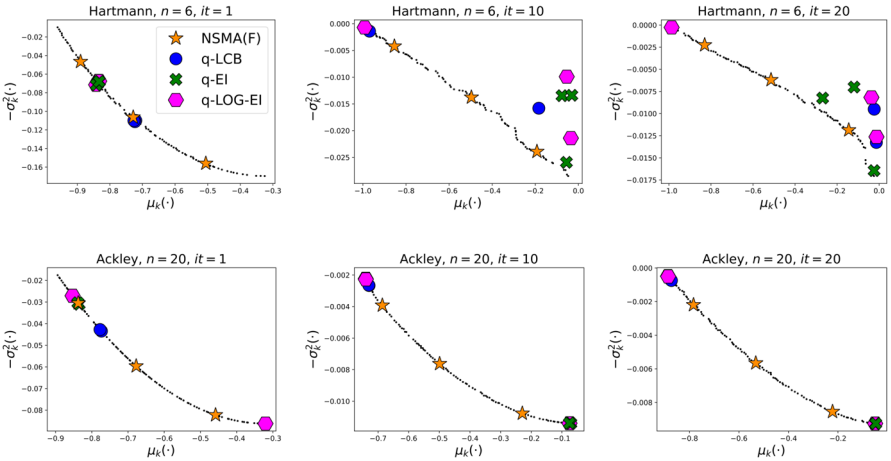
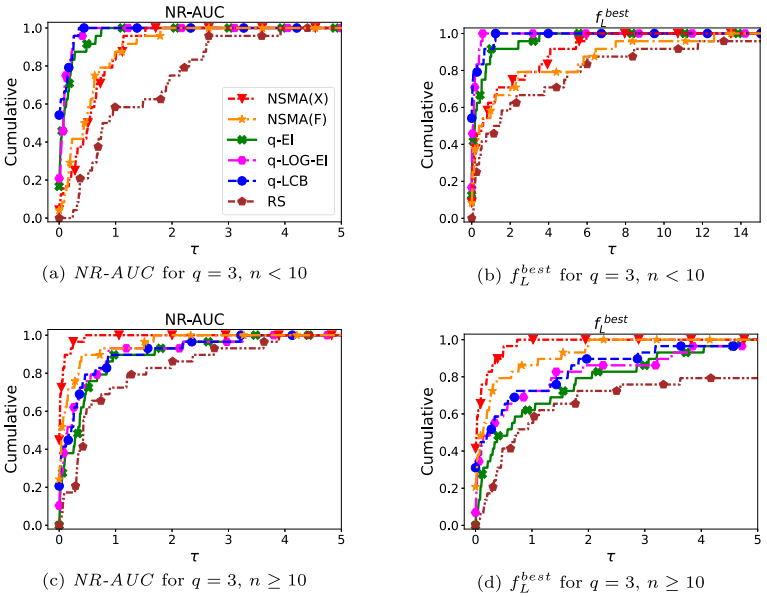


Fig. 10 Candidate points selection by `NSMA(F)`, `q-LCB`, `q-EI` and `q-LOG-EI` on Hartmann ($n = 6$) and Ackley ($n = 20$) problems at iteration 1, 10 and 20 of the Bayesian optimization procedure. The small black dots represent the Pareto front obtained by the `NSMA` execution

We immediately observe the uniform points selection performed by `NSMA(F)` w.r.t. `q-LCB`, `q-EI` and `q-LOG-EI`, which allowed to balance between exploitation of the collected points by the Gaussian Process and exploration of the feasible set. From a multi-objective optimization view, such points were in generally mutually non-dominated w.r.t. the ones chosen by the other approaches, with some (excellent) cases (Hartmann, it $\in \{10, 20\}$) where the solutions returned by the standard approaches were even dominated. In these last scenarios, the solver `L-BFGS-B` used within `q-LCB`, `q-EI` and `q-LOG-EI` did not manage to escape from Pareto stationary points that are far from the Pareto front. However, from a Bayesian optimization point of view all the algorithms were capable of selecting points from different parts/neighbors of the Pareto frontier on the Hartmann problem: such choices allowed the algorithms to find better and better solutions, as can be directly seen in Fig. 8, with `q-LCB` being the most effective strategy.

On the high dimensional scenario of Ackley ($n = 20$), we can observe that after the first iteration `q-EI` struggled to select points other than the extreme point of the Pareto front, i.e., the one which maximizes the variance; such selection was reasonably caused by the standard deviation ($\sigma_k(\cdot)$) term in the `q-EI` function (9) which started to dominate the other one – the mean ($\mu_k(\cdot)$) term. Thus, after the first iteration `q-EI` was only able to explore the feasible set, failing to exploit the information provided by the Gaussian process throughout the optimization process. As a result, `q-EI` struggled to optimize the Ackley problem, as also illustrated in Fig. 8. In contrast, `q-LCB` and `q-LOG-EI` were also able to identify the opposite extreme of the Pareto front – the point minimizing the mean, and this ability allowed them to outperform `q-EI` on the task at hand. However, their tendency to focus solely on the extremes of the Pareto frontier prevented them from matching the performance of `NSMA(F)`. The latter, by uniformly sampling across the entire Pareto front, consistently discovered increasingly better solutions compared to the other methods, as shown in the results for



(a) *NR-AUC* for $q = 3$, $n < 10$ (b) f_L^{best} for $q = 3$, $n < 10$

(c) *NR-AUC* for $q = 3$, $n \geq 10$ (d) f_L^{best} for $q = 3$, $n \geq 10$

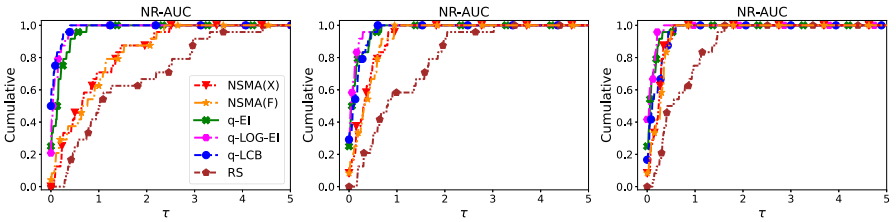
Fig. 11 Plots of the relative gap cumulative distribution for `NSMA(X)`, `NSMA(F)`, `q-EI`, `q-LOG-EI`, `q-LCB` and `RS` considering the low-dimensional ($n < 10$) and high-dimensional ($n \geq 10$) functions listed in Table 1. Note that the intervals of the x-axis are set for a better visualization of the numerical results

the Ackley function ($n = 20$) in Fig. 8. This behavior becomes even more pronounced for problems with $n \geq 20$.

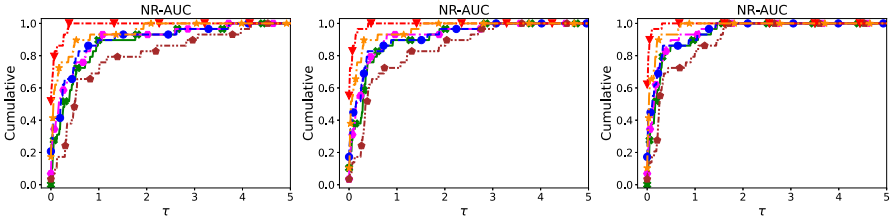
4.5 Overall Performance Evaluation using Relative Gap Cumulative Distributions

In the last section of computational experiments, we first show the relative gap cumulative distributions for `NSMA(X)`, `NSMA(F)`, `q-EI`, `q-LOG-EI`, `q-LCB` and `RS` on the low-dimensional and high-dimensional functions separately. The distributions are plotted in Fig. 11.

The plots summarize some highlights of the previous sections. In the low-dimensional problems, `q-EI`, `q-LOG-EI` and `q-LCB` were the best methodologies in terms of both *NR-AUC* and f_L^{best}; the performance obtained by the three acquisition strategies was similar. In this scenario, `NSMA(X)` and `NSMA(F)` were competitive in terms of robustness, reaching the competitors for values of $\tau \geq 2$ on the *NR-AUC* metric and of $\tau \geq 6$ on the f_L^{best} one. We then conclude that the selection points mechanisms of the state-of-the-art method had a greater impact on the low-dimension scenario. In the high-dimensional cases, the situation is the opposite, with `NSMA(X)` and `NSMA(F)` outperforming `q-EI`, `q-LOG-EI` and `q-LCB` w.r.t. both metrics. In this context, `NSMA(X)` was by far the best algorithm. On the other hand, `NSMA(F)` had the second best performance, allowing us once again to attest the goodness of selecting Pareto front points by clustering on high-dimensional instances (see Sects. 3.2-4.4



(a) *NR-AUC* for $q = 2$, $n < 10$ (b) *NR-AUC* for $q = 5$, $n < 10$ (c) *NR-AUC* for $q = 10$, $n < 10$



(d) *NR-AUC* for $q = 2$, $n \geq 10$ (e) *NR-AUC* for $q = 5$, $n \geq 10$ (f) *NR-AUC* for $q = 10$, $n \geq 10$

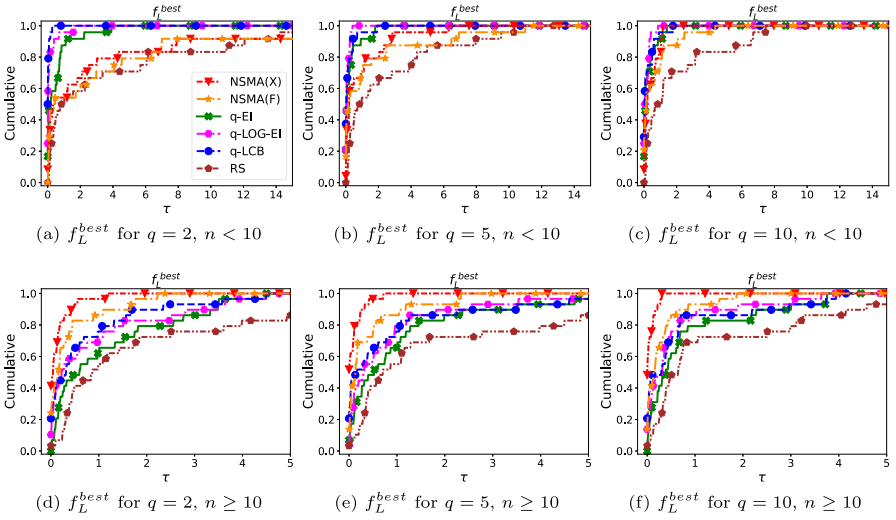
Fig. 12 Plots of the relative gap cumulative distribution for `NSMA(X)`, `NSMA(F)`, `q-EI`, `q-LOG-EI`, `q-LCB` and `RS` in terms of *NR-AUC* considering the low-dimensional ($n < 10$) and high-dimensional ($n \geq 10$) functions listed in Table 1 and different values for the batch size parameter q

for further details). Regarding `RS`, it reached again the worst performance in any considered setting.

The performance of the tested acquisition strategies was further investigated as the value for the batch size parameter q varied. In particular, we considered $q \in \{2, 5, 10\}$. Note that in this experimentation the setting related to the number of function evaluations (excluding the first $N_i = 10$ ones) N_M (see Algorithm 1) remained the same, i.e., $N_M = 60$. Thus, we can easily observe that when $q = 2$ the Gaussian Process regression was performed more times than when $q = 5$. For more details on the impact of q on the Gaussian Process regression, the reader is referred to Sect. 2.1. For the sake of brevity, we again compare the methodologies by plotting in Figs. 12-13 their relative gap cumulative distributions in terms of *NR-AUC* and f_L^{best}.

The general behavior of the acquisition strategies is confirmed with these new plots: `q-EI`, `q-LOG-EI` and `q-LCB` performed better in the low-dimensional cases; `NSMA(X)` and `NSMA(F)` were the best algorithms for high values for n. Comparing the performance for different q, we observe that our proposals got better results as the value of q increased. When $q = 5$ or $q = 10$, the performance gap between the BOO based acquisition strategies and the standard ones in terms of the metrics f_L^{best} and *NR-AUC* is less marked in the low dimensional scenario than when $q = 2$ or $q = 3$; for any value of q, the gap on the high dimensional instances is sharp, with our proposals being by far the best choices. It seems that our proposals become more effective as the number of points we can evaluate in parallel increases. In the Bayes-Opt context, it is difficult to have the resources to evaluate many points in parallel; however, already with values of $q \in \{2, 5, 10\}$, `NSMA(X)` and `NSMA(F)` obtained interesting results.

We conclude the section comparing `NSMA(X)` and `NSMA(F)` with the recently proposed `BBO-ABAFMo`, `MACE`, `TURBO-1` and `TURBO-M` methodologies; again, in



(a) f_L^{best} for $q = 2$, $n < 10$ (b) f_L^{best} for $q = 5$, $n < 10$ (c) f_L^{best} for $q = 10$, $n < 10$

(d) f_L^{best} for $q = 2$, $n \geq 10$ (e) f_L^{best} for $q = 5$, $n \geq 10$ (f) f_L^{best} for $q = 10$, $n \geq 10$

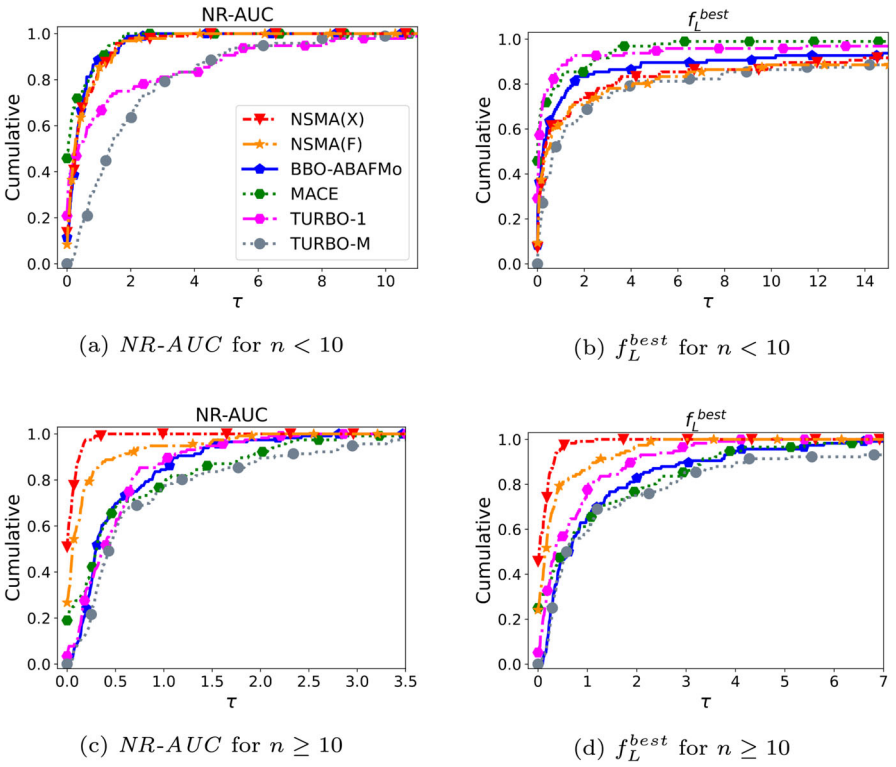
Fig. 13 Plots of the relative gap cumulative distribution for `NSMA(X)`, `NSMA(F)`, `q-EI`, `q-LOG-EI`, `q-LCB` and `RS` in terms of f_L^{best} considering the low-dimensional ($n < 10$) and high-dimensional ($n \geq 10$) functions listed in Table 1 and different values for the batch size parameter q. Note that the intervals of the x-axis are set for a better visualization of the numerical results

order to be concise, in Fig. 14 we show the relative gap cumulative distribution obtained by the six acquisition strategies considering all the problem benchmark and any value of $q \in \{2, 3, 5, 10\}$.

On low-dimensional instances, `MACE` was better than all the other competitors in terms of both metrics; our approaches, along with `BBO-ABAFMo`, had good performance in terms of *NR-AUC* robustness; `TURBO-1` and `TURBO-M` seem to struggle on *NR-AUC* w.r.t. all the competitors, while `TURBO-1` had similar performance as `MACE` in terms of f_L^{best}. In the high-dimensional scenarios, the proposed acquisition strategy again proved to be superior w.r.t. all the tested state-of-the-art approaches, with `NSMA(X)` and `NSMA(F)` being the first and second best algorithms, respectively.

5 Concluding Remarks

In this paper, we dealt with Batch Bayesian Optimization and we proposed a gradient-based novel acquisition strategy for selecting new points where to evaluate the objective function $f(\cdot)$ of the original problem (1). In Bayes-Opt, the choice of the points to evaluate is one of the most crucial operations: the evaluations are usually expensive, causing an high consumption of resources (e.g., CPU/GPU time, financial costs, etc.). Ideally, we would like to obtain a global optimum of the original problem in the minimum number of evaluations. The most employed acquisition strategies consist on selecting new points by minimizing an acquisition function: `q-EI` (9) and `q-LCB` (10) are certainly among the most famous ones.



(a) $NR\text{-}AUC$ for $n < 10$ (b) f_L^{best} for $n < 10$

(c) $NR\text{-}AUC$ for $n \geq 10$ (d) f_L^{best} for $n \geq 10$

Fig. 14 Plots of the relative gap cumulative distribution for `NSMA(X)`, `NSMA(F)`, `BBO-ABAFMo`, `MACE`, `TURBO-1` and `TURBO-M` considering the low-dimensional ($n < 10$) and high-dimensional ($n \geq 10$) functions listed in Table 1 and all the values for the batch size parameter $q \in \{2, 3, 5, 10\}$. Note that the intervals of the x-axis are set for a better visualization of the numerical results

In the proposed acquisition strategy, we consider a Bi-Objective Optimization problem (2) based on the posterior mean ($\mu_k(\cdot)$) and variance ($\sigma_k^2(\cdot)$) functions of the Gaussian Process used in the Bayes-Opt procedure. By means of the *Non-dominated Sorting Memetic Algorithm* [42], we aim to reconstruct, as accurately as possible, the Pareto front of (2), exploiting, for the first time in the batch Bayesian optimization literature, first-order information of the functions $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$, in order to select then potentially optimal solutions for problem (1). The Pareto front reconstruction leads to the generation of multiple solutions representing different compromises between exploitation of the current information about $f(\cdot)$ (low $\mu_k(\cdot)$ values) and exploration of new regions of the problem feasible set (high $\sigma_k^2(\cdot)$ values). All these exploitation-exploration trade-offs would be difficult to reach with the employment of the standard acquisition strategies, usually based on combinations of the two posterior distribution functions. The selection of points from the Pareto front can then be performed through the two proposed clustering approaches. These latter ones work on different spaces: the first approach employs `K-MEANS` in the variables space, while the second one operates in the objectives space.

The proposed strategy was tested with thorough computational experiments, mainly comparing its performance with the one of well-known acquisition methodologies from the Bayes-Opt literature. In the low-dimensional case our proposal turned out to be competitive, while in the high-dimensional scenario it achieved remarkable results, outperforming by far the competitors. In particular, the use of the functions gradients allowed NSMA to have superior performance than NSGA-II, confirming our thesis on the benefits of using first-order information for the Pareto front reconstruction of the BOO problem (2). The clustering approach in the objectives space outperformed the popular state-of-the-art acquisition strategies on high-dimensional scenarios, where setting the objective function weights is a non-trivial task, as is consequently performing uniform choices for the next candidate solutions from the Pareto front. On the other hand, the clustering method in the domain space was undoubtedly the best approach when the dimensionality $n \geq 50$. Its potential effectiveness w.r.t. the boundary issue (typical in the Bayesian literature in such cases), which is analyzed in the theoretical part of the paper, is then confirmed by the computational experiments.

As possible future research directions, the following ones certainly stand out: a variant of the proposed methodology where the batch size parameter q is chosen dynamically at each iteration; the integration of random mechanisms into the presented approaches to escape from poor local minima; an effectiveness study of the presented acquisition strategy in Bayes-Opt procedures different from the standard one (e.g., see [80]); the study of alternative (compositional) approaches (e.g., see [81]); some test with dimensionality reduction techniques to improve the performance of clustering in the variables space, or with other clustering strategies to handle higher-dimensional data better than K-MEANS.

Appendix A Pareto Stationarity

In this appendix, we introduce the concept of *Pareto stationarity*.

Definition A.1 A point $\bar{x} \in \Omega$ is *Pareto-stationary* for problem (11) if we have that

$$\max_{j \in \{1,\dots,m\}} \nabla f_j(\bar{x})^\top d \geq 0, \qquad \forall d \in \mathcal{D}(\bar{x}),$$

where $\mathcal{D}(\bar{x}) = \{v \in \mathbb{R}^n \mid \exists \bar{t} > 0 : \bar{x} + tv \in \Omega,\ \forall t \in [0, \bar{t}\,]\}$.

The property can be also compactly re-written as

$$\min_{d \in \mathcal{D}(\bar{x})} \max_{j \in \{1,\dots,m\}} \nabla f_j(\bar{x})^\top d = 0.$$

If a point $\bar{x}$ is not Pareto stationary for problem (11), then there exists a direction $\hat{d} \in \mathcal{D}(\bar{x})$ such that $\max_{j \in \{1,\dots,m\}} \nabla f_j(\bar{x})^\top \hat{d} < 0$. Thus, supposing the continuously differentiability of $\mathcal{F}(\cdot)$, we have that

$$\lim_{t \to 0} \frac{f_j(\bar{x} + t\hat{d}) - f_j(\bar{x})}{t} = \nabla f_j(\bar{x})^\top \hat{d} < 0, \qquad \forall j \in \{1, \dots, m\}.$$

Direction $\hat{d}$ is thus a *common descent direction* for $\mathcal{F}(\cdot)$ at $\bar{x}$: there exists $\bar{t} > 0$ such that

$$\mathcal{F}(\bar{x} + t\hat{d}) < \mathcal{F}(\bar{x}), \qquad \forall t \in (0, \bar{t}\,].$$

Under differentiability assumptions, Pareto stationarity is a necessary condition for all types of Pareto optimality (see Sect. 2.2); further assuming the convexity of $\mathcal{F}(\cdot)$ (component-wise) and Ω, the condition is sufficient for weak Pareto optimality.

Lemma A.1 ([35, Theorem 3.1]) *The following statements hold:*

- *if $\bar{x}$ is locally weakly Pareto optimal, then $\bar{x}$ is Pareto stationary for problem (11);*
- *if Ω is convex, $\mathcal{F}(\cdot)$ is component-wise convex and $\bar{x}$ is Pareto stationary for problem (11), then $\bar{x}$ is weakly Pareto optimal;*
- *if Ω is convex, $\mathcal{F}(\cdot)$ is twice continuously differentiable and component-wise strictly convex and $\bar{x}$ is Pareto stationary for problem (11), then $\bar{x}$ is Pareto optimal.*

Note that in the considered Bayes-Opt context (1) Ω is assumed to be a bounding box and, thus, it is a convex set.

Unlike (weak) Pareto optimality, Pareto stationarity represents a concrete goal for many MOO algorithms based on first-order information of $\mathcal{F}(\cdot)$ [33].

Appendix B Further Information on Non-dominated Sorting Memetic Algorithm for Bayes-Opt

In this appendix, we provide additional details on some mechanisms of the *Non-dominated Sorting Memetic Algorithm* [42]. For the sake of simplicity, we report the scheme of the approach here below too (Algorithm B.1). A brief description of the latter can be found in Sect. 3.1. The scheme, as well as all the other concepts in this appendix, were adapted to the considered Bayes-Opt context. For a more detailed description of `NSMA`, we refer the reader to [42].

B.1 Selection Metrics

The selection of both the *parents* in the `getParents` function and the best N population solutions in the `selection` operator is driven by two metrics, i.e., the *Ranking* and the *Crowding Distance*.

Given a generic population X, the first metric assigns to its solutions $x_1, \ldots, x_{|X|}$ different domination levels $R = [r_1, \ldots, r_{|X|}]^\top$: a point x_p, with $p \in \{1, \ldots, |X|\}$, with rank $r_p = 0$ means that there does not exist any point $y \in X$ such that $\mathcal{F}(y) \lneqq \mathcal{F}(x_p)$; if $r_p = 1$, it means that there exists $x_{\hat{p}} \in X$, with $\hat{p} \in \{1, \ldots, |X|\}$ and $r_{\hat{p}} = 0$, such that $\mathcal{F}(x_{\hat{p}}) \lneqq \mathcal{F}(x_p)$, but there does not exist $x_{\hat{p}} \in X$ with $r_{\hat{p}} \geq 1$ that dominates x_p; similar reasoning can be applied for the points with ranking greater than 1. In order to obtain the ranking values, the same fast sorting approach employed in `NSGA-II` [41] is also used in `NSMA`.

The *Crowding Distance* $C = [c_1, \ldots, c_{|X|}]^\top$ indicates if the points are in poorly populated areas of the objectives space: points with an high crowding distance are

Algorithm B.1 Non-dominated Sorting Memetic Algorithm for Bayes-Opt

Require: Ω feasible set, $X^i \subset \Omega$, N population size, $n_{opt} \in \mathbb{N}^+$.
1: Let $\mu_k(\cdot)$, $\sigma_k^2(\cdot)$ be the posterior mean and variance functions [Line 6, Algorithm 1]
2: Let $\mathcal{F}(\cdot) = (\mu_k(\cdot), -\sigma_k^2(\cdot))^\top$
3: $X^0 = \texttt{selection}(\mathcal{F}(\cdot), \Omega, X^i, N)$
4: $t = 0$
5: **while** *a stopping criterion is not satisfied* **do**
6: $P^t = \texttt{getParents}(\mathcal{F}(\cdot), \Omega, X^t)$
7: $O^t = \texttt{crossover}(\Omega, P^t)$
8: $\tilde{O}^t = \texttt{mutation}(\Omega, O^t)$
9: $\hat{X}^{t+1} = X^t \cup \tilde{O}^t$
10: $X^{t+1} = \texttt{selection}(\mathcal{F}(\cdot), \Omega, \hat{X}^{t+1}, N)$
11: **if** $t \mod n_{opt} = 0$ **then**
12: $X^{t+1} = \texttt{optimizePopulation}(\Omega, X^{t+1}, \mathcal{F}(\cdot), N)$
13: **end if**
14: $t = t + 1$
15: **end while**
16: $X^\star = X^t$
17: **return** $X^\star$

extremes of the current Pareto front approximation or they are located far from the other solutions of the population; usually, these points are preferred for starting new search steps to get wider Pareto front approximations. Additional information on the *Crowding Distance*, as well as on the *Ranking*, can be found in [41].

The metrics vectors R and C are also used in the `optimizePopulation` function to select the points to refine through the gradient-based local solver; further details on this topic can be found in the next subsection.

B.2 `optimizePopulation` function

In Algorithm B.2, we report the scheme of the `optimizePopulation` function, which we recall to be the main algorithmic difference between `NSMA` and `NSGA-II`. The function allows to refine selected points of the current population X^{t+1} through the gradient-based local solver called *Front Multi-Objective Projected Gradient*, which exploits first-order information of $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$ computable through *automatic differentiation* packages of well-known libraries like `BoTorch` [17] (see Sect. 2.1). The refinement can be performed w.r.t. both the objectives $\mu_k(\cdot)$ and $\sigma_k^2(\cdot)$, only $\mu_k(\cdot)$ or only $\sigma_k^2(\cdot)$, thus contributing to explore the objectives space and to obtain accurate, wide, uniform Pareto front approximations; the subset of indices of the considered objectives is denoted by $\mathcal{I} \subseteq \{1, 2\}$.

Given the *Ranking* and *Crowding Distance* vectors R^{t+1}, C^{t+1} related to the population X^{t+1} [Line 1, Algorithm B.2], the Q-quantile of the finite crowding distances associated to the rank-0 points $\bar{c}_{t+1}$ ([Line 2, Algorithm B.2]; Q is an `NSMA` parameter) and $\hat{X}^{t+1}$ initialized equal to X^{t+1} [Line 3, Algorithm B.2], a point $x_p \in X^{t+1}$ [Line 4, Algorithm B.2] must satisfy the following conditions in order to be optimized through `FMOPG` w.r.t. a subset $\mathcal{I} \subseteq \{1, 2\}$ [Line 5, Algorithm B.2]:

Algorithm B.2 optimizePopulation

Require: Ω feasible set, $X^{t+1} \subset \Omega$, $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^2$, N population size.
1: Let $R^{t+1}, C^{t+1} \in \mathbb{R}^{|X^{t+1}|}$ be the Ranking and Crowding Distance vectors
2: Let $\bar{c}_{t+1}$ be the Q-quantile of the finite Crowding Distances associated to the rank-0 points
3: $\hat{X}^{t+1} = X^{t+1}$
4: **for** $p = 1, \ldots, |X^{t+1}|$ **do**
5: **for** $\mathcal{I} \in 2^{\{1,2\}}$ **do**
6: **if** x_p is s.t. **a)** $r_p = 0, c_p \geq \bar{c}_{t+1}$ **and b)** $\nexists y \in \hat{X}^{t+1} : \mathcal{F}_{\mathcal{I}}(y) \lneqq \mathcal{F}_{\mathcal{I}}(x_p)$ **then**
7: $\tilde{X}^p = \texttt{FMOPG}(\Omega, \hat{X}^{t+1}, x_p, \mathcal{F}(\cdot), \mathcal{I})$
8: $\hat{X}^{t+1} = \hat{X}^{t+1} \cup \tilde{X}^p$
9: **end if**
10: **end for**
11: **end for**
12: $X^{t+1} = \texttt{selection}(\mathcal{F}(\cdot), \Omega, \hat{X}^{t+1}, N)$
13: **return** X^{t+1}

- it must be a rank-0 points with an assigned crowding distance greater than $\bar{c}_{t+1}$ [Line 6, Algorithm B.2, point a)];
- there must not exist a point in $\hat{X}^{t+1}$ that dominates it w.r.t. the subset of objectives $\mathcal{F}_{\mathcal{I}}(\cdot)$ [Line 6, Algorithm B.2, point b)].

If the requirements are satisfied, then x_p is used as starting point for FMOPG; the set of solutions produced by the latter are collected in $\tilde{X}^p$ [Line 7, Algorithm B.2] and then inserted in $\hat{X}^{t+1}$ [Line 8, Algorithm B.2].

Note that at the end of the optimizePopulation function [Line 12, Algorithm B.2], the selection operator is called to reduce the new population $\hat{X}^{t+1}$ to exactly N survivors.

The scheme of FMOPG is reported in Algorithm B.3. Starting from a solution $x_0 \in \hat{X}^0$, the approach iterates until the current solution x_l is Pareto stationary (Appendix A) w.r.t. the considered subset of objectives $\mathcal{F}_{\mathcal{I}}(\cdot)$ [Line 2, Algorithm B.3], with $\mathcal{I} \subseteq \{1, 2\}$.

The search steps are carried out along directions found by solving the optimization problem (B.1); the latter are called *steepest common descent directions* [33] if $\mathcal{I} = \{1, 2\}$, *steepest partial descent directions* [39] if $\mathcal{I} \subset \{1, 2\}$. We can easily deduce that if the problem minimum $\theta^{\mathcal{I}}(x_l) = 0$ then x_l is Pareto stationary w.r.t. $\mathcal{F}_{\mathcal{I}}(\cdot)$, and vice versa; the stopping condition in [Line 2, Algorithm B.3] can be thus verified checking this value. Otherwise, if x_l is not Pareto stationary $\theta^{\mathcal{I}}(x_l) < 0$; we denote as $v_l^{\mathcal{I}}$ the (not necessarily unique) direction associated with $\theta^{\mathcal{I}}(x_l)$, that is, the argument of the problem minimum at x_l [Line 4, Algorithm B.3]. In this last case, we can employ an *Armijo-Type Line Search* to find a step along $v_l^{\mathcal{I}}$ [Lines 5-9, Algorithm B.3], leading to a new feasible point that is not *sufficiently dominated* by any other one in the population $\hat{X}^l$; the resulting solution is added to the population [Line 11, Algorithm B.3]. At the end of the execution, FMOPG returns the sequence of produced point $\{x_l\}$ [Line 14, Algorithm B.3]: based on the line-search criteria [Line 6, Algorithm B.3], we indeed have no guarantee that, for all l, $\mathcal{F}_{\mathcal{I}}(x_{l+1}) \lneqq \mathcal{F}_{\mathcal{I}}(x_l)$; thus, every point could be crucial to obtain accurate and uniform Pareto front approximations.

Algorithm B.3 Front Multi-Objective Projected Gradient

Require: Ω feasible set, $\hat{X}^0 \subset \Omega$, $x_0 \in \hat{X}^0$, $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^2$, $\mathcal{I} \subseteq \{1, 2\}$, $\alpha_0 > 0$, $\beta \in (0, 1)$, $\delta \in (0, 1)$.
1: $l = 0$
2: **while** x_l is not Pareto-stationary w.r.t. $\mathcal{F}_\mathcal{I}$ **do**
3: Compute

$$\theta^{\mathcal{I}}(x_l) = \min_{\substack{d \in \mathcal{D}(x_l) \\ \|d\|_\infty \leq 1}} \max_{j \in \mathcal{I}} \nabla f_j(x_l)^\top d \tag{B.1}$$

4: Let $v_l^{\mathcal{I}}$ be the (not necessarily unique) direction associated with $\theta^{\mathcal{I}}(x_l)$
5: $\alpha = \alpha_0$
6: **while** $x_l + \alpha v_l^{\mathcal{I}} \notin \Omega$ **or** $\exists y \in \hat{X}^l : \mathcal{F}_\mathcal{I}(y) + \mathbf{1}\beta\alpha\theta^{\mathcal{I}}(x_l) < \mathcal{F}_\mathcal{I}(x_k + \alpha v_l^{\mathcal{I}})$ **do**
7: $\alpha = \delta\alpha$
8: **end while**
9: $\alpha_l = \alpha$
10: $x_{l+1} = x_l + \alpha_l v_l^{\mathcal{I}}$
11: $\hat{X}^{l+1} = \hat{X}^l \cup \{x_{l+1}\}$
12: $l = l + 1$
13: **end while**
14: **return** sequence $\{x_l\}$

We conclude the section reporting the theoretical and convergence properties of FMOPG, adapted to the Bayes-Opt context.

Lemma B.1 ([42, Corollary 1]) *Assume that $\mathcal{I} \subseteq \{1, 2\}$, $\hat{X}^0 \subset \Omega$ is a set of feasible points, $x_0 \in \hat{X}^0$ and there does not exist a point $y \in \hat{X}^0$ that dominates x_0 w.r.t. $\mathcal{F}_\mathcal{I}(\cdot)$. Then, the sequence of sets $\{\hat{X}_l\}$ and the sequence of points $\{x_l\}$ generated by FMOPG are such that for all $l = 0, 1, \ldots$, x_l is feasible and not dominated by any point in $\hat{X}^l$ w.r.t. $\mathcal{F}_\mathcal{I}(\cdot)$.*

Lemma B.2 ([42, Proposition 3]) *Let the assumptions of Lemma B.1 hold and x_0 be such that:*

- *x_0 is not Pareto stationary w.r.t. $\mathcal{F}_\mathcal{I}(\cdot)$;*
- *the set $\mathcal{L}(x_0) = \bigcup_{j=1}^2 \{x \in \Omega : f_j(x) \leq f_j(x_0)\}$ is compact.*

Let $\{x_l\}$ be the sequence of points generated by FMOPG. Then $\{x_l\}$ admits limit points and every limit point is Pareto stationary considering the objectives f_j, with $j \in \mathcal{I}$.

Note that, in practice, FMOPG is executed until x_l is approximately Pareto stationary, with $\varepsilon_t \in \mathbb{R}_0^+$ indicating the approximation degree [42]. In particular, in NSMA (Algorithm B.1) a decreasing sequence $\{\varepsilon_t\} \subset \mathbb{R}_0^+$ is considered, so that, during the iterations, we get closer and closer to the Pareto stationarity. Moreover, using this stopping criterion, FMOPG is well-defined, terminating in a finite number of iterations.

Lemma B.3 ([42, Proposition 4]) *Let the assumptions of Lemma B.2 hold and $\varepsilon > 0$. Then, the FMOPG algorithm finds in a finite number of steps a point x_l such that $\theta^{\mathcal{I}}(x_l) \geq -\varepsilon$.*

Acknowledgements We would like to thank the three anonymous reviewers for their precious comments, that allowed us to significantly improve the quality of the manuscript.

Funding Open access funding provided by Università degli Studi di Firenze within the CRUI-CARE Agreement. No funding was received for conducting this study.

Data availability Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Code availability The full code of the experiments can be found at https://github.com/FranciC19/biobj_acquistion_function_for_BO.

Declarations

Competing interests The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Snoek, J., Larochelle, H., Adams, R.P.: Practical Bayesian optimization of machine learning algorithms. In: Pereira, F., Burges, C.J., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems, vol. 25 (2012). https://proceedings.neurips.cc/paper_files/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf
2. Forrester, A., Sobester, A., Keane, A.: Engineering Design Via Surrogate Modelling: a Practical Guide. John Wiley & Sons, Hoboken, New Jersey, United States (2008). https://doi.org/10.1002/9780470770801
3. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient Global Optimization of Expensive Black-Box Functions. Journal of Global Optimization **13**(4), 455–492 (1998). https://doi.org/10.1023/A:1008306431147
4. Shoemaker, Christine A., R.G.R., Fleming, R.C.: Watershed calibration using multistart local optimization and evolutionary optimization with radial basis function approximation. Hydrological Sciences Journal **52**(3), 450–465 (2007). https://doi.org/10.1623/hysj.52.3.450
5. Brochu, E., Cora, V.M., Freitas, N.: A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning (2010). https://doi.org/10.48550/arXiv.1012.2599
6. Lizotte, D.J., Wang, T., Bowling, M.H., Schuurmans, D. et al.: Automatic gait optimization with Gaussian process regression. In: IJCAI, vol. 7, pp. 944–949 (2007). https://dl.acm.org/doi/abs/10.5555/1625275.1625428#d14294314e1
7. Frazier, P.I., Wang, J.: In: Lookman, T., Alexander, F.J., Rajan, K. (eds.) Bayesian Optimization for Materials Design, pp. 45–75. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-23871-5_3
8. Negoescu, D.M., Frazier, P.I., Powell, W.B.: The knowledge-gradient algorithm for sequencing experiments in drug discovery. INFORMS Journal on Computing **23**(3), 346–363 (2011). https://doi.org/10.1287/ijoc.1100.0417
9. Kushner, H.J.: A New Method of Locating the Maximum Point of an Arbitrary Multipeak Curve in the Presence of Noise. Journal of Basic Engineering **86**(1), 97–106 (1964). https://doi.org/10.1115/1.3653121

10. Močkus, J.: On Bayesian methods for seeking the extremum. In: Marchuk, G.I. (ed.) Optimization Techniques IFIP Technical Conference: Novosibirsk, July 1–7, 1974, pp. 400–404. Springer, Berlin, Heidelberg (1975). https://doi.org/10.1007/978-3-662-38527-2_55
11. Audet, C., Le Digabel, S., Saltet, R.: Quantifying uncertainty with ensembles of surrogates for blackbox optimization. Computational Optimization and Applications **83**(1), 29–66 (2022). https://doi.org/10.1007/s10589-022-00381-z
12. Binois, M., Wycoff, N.: A survey on high-dimensional Gaussian process modeling with application to Bayesian optimization. ACM Trans. Evol. Learn. Optim. **2**(2) (2022) https://doi.org/10.1145/3545611
13. Swersky, K.: Improving Bayesian Optimization for Machine Learning Using Expert Priors. University of Toronto (Canada), Toronto, Canada (2017). https://tspace.library.utoronto.ca/bitstream/1807/79505/3/Swersky_Kevin_J_201706_PhD_thesis.pdf
14. Oh, C., Gavves, E., Welling, M.: BOCK : Bayesian optimization with cylindrical kernels. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 3868–3877 (2018). https://proceedings.mlr.press/v80/oh18a.html
15. Frazier, P.I.: Bayesian Optimization, pp. 255–278. Chap. 11. https://doi.org/10.1287/educ.2018.0188
16. Garnett, R.: Bayesian Optimization. Cambridge University Press, Cambridge, United Kingdom (2023). https://doi.org/10.1017/9781108348973
17. Balandat, M., Karrer, B., Jiang, D., Daulton, S., Letham, B., Wilson, A.G., Bakshy, E.: Botorch: A framework for efficient monte-carlo Bayesian optimization. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 21524–21538 (2020). https://proceedings.neurips.cc/paper_files/paper/2020/file/f5b1b89d98b7286673128a5fb112cb9a-Paper.pdf
18. Roustant, O., Ginsbourger, D., Deville, Y.: DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization. Journal of Statistical Software 51(1), 1–55 (2012) https://doi.org/10.18637/jss.v051.i01
19. Saves, P., Lafage, R., Bartoli, N., Diouane, Y., Bussemaker, J., Lefebvre, T., Hwang, J.T., Morlier, J., Martins, J.R.R.A.: SMT 2.0: A surrogate modeling toolbox with a focus on hierarchical and mixed variables gaussian processes. Advances in Engineering Software 188, 103571 (2024) https://doi.org/10.1016/j.advengsoft.2023.103571
20. Picheny, V., Berkeley, J., Moss, H.B., Stojic, H., Granta, U., Ober, S.W., Artemev, A., Ghani, K., Goodall, A., Paleyes, A., Vakili, S., Pascual-Diaz, S., Markou, S., Qing, J., Loka, N.R.B.S., Couckuyt, I.: Trieste: Efficiently Exploring The Depths of Black-box Functions with TensorFlow (2023). https://doi.org/10.48550/arXiv.2302.08436
21. Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., Hutter, F.: SMAC3: A versatile Bayesian optimization package for hyperparameter optimization. Journal of Machine Learning Research **23**(54), 1–9 (2022)
22. Kandasamy, K., Vysyaraju, K.R., Neiswanger, W., Paria, B., Collins, C.R., Schneider, J., Poczos, B., Xing, E.P.: Tuning hyperparameters without grad students: Scalable and robust Bayesian optimisation with Dragonfly. Journal of Machine Learning Research **21**(81), 1–27 (2020)
23. Rasmussen, C.E., Williams, C.: Gaussian Processes for Machine Learning. the MIT Press, Cambridge, Massachusetts, United States (2006). https://doi.org/10.7551/mitpress/3206.001.0001
24. Srinivas, N., Krause, A., Kakade, S.M., Seeger, M.W.: Information-theoretic regret bounds for Gaussian process optimization in the bandit setting. IEEE Transactions on Information Theory **58**(5), 3250–3265 (2012). https://doi.org/10.1109/tit.2011.2182033
25. Wang, J., Clark, S.C., Liu, E., Frazier, P.I.: Parallel Bayesian Global Optimization of Expensive Functions. Operations Research **68**(6), 1850–1865 (2020). https://doi.org/10.1287/opre.2019.1966
26. Wilson, J.T., Moriconi, R., Hutter, F., Deisenroth, M.P.: The reparameterization trick for acquisition functions. arXiv preprint arXiv:1712.00424 (2017) https://doi.org/10.48550/arXiv.1712.00424
27. Desautels, T., Krause, A., Burdick, J.W.: Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. Journal of Machine Learning Research **15**(1), 3873–3923 (2014)
28. Contal, E., Buffoni, D., Robicquet, A., Vayatis, N.: Parallel gaussian process optimization with upper confidence bound and pure exploration. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 225–240 (2013). https://doi.org/10.1007/978-3-642-40988-2_15. Springer
29. Moss, H.B., Leslie, D.S., Gonzalez, J., Rayson, P.: GIBBON: General-purpose information-based Bayesian optimisation. Journal of Machine Learning Research **22**(235), 1–49 (2021)

30. Gonzalez, J., Osborne, M., Lawrence, N.: GLASSES: Relieving the myopia of Bayesian optimisation. In: Gretton, A., Robert, C.C. (eds.) Proceedings of the 19th International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 51, pp. 790–799. PMLR, Cadiz, Spain (2016). https://proceedings.mlr.press/v51/gonzalez16b.html
31. Wu, J., Frazier, P.: The parallel Knowledge Gradient method for batch Bayesian optimization. In: Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 29 (2016). https://proceedings.neurips.cc/paper_files/paper/2016/file/18d10dc6e666eab6de9215ae5b3d54df-Paper.pdf
32. Eriksson, D., Pearce, M., Gardner, J., Turner, R.D., Poloczek, M.: Scalable global optimization via local Bayesian optimization. In: Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32 (2019). https://proceedings.neurips.cc/paper_files/paper/2019/file/6c990b7aca7bc7058f5e98ea909e924b-Paper.pdf
33. Fliege, J., Svaiter, B.F.: Steepest descent methods for multicriteria optimization. Mathematical Methods of Operations Research **51**(3), 479–494 (2000). https://doi.org/10.1007/s001860000043
34. Drummond, L.G., Iusem, A.N.: A projected gradient method for vector optimization problems. Computational Optimization and applications **28**(1), 5–29 (2004). https://doi.org/10.1023/B:COAP.0000018877.86161.8b
35. Fliege, J., Drummond, L.G., Svaiter, B.F.: Newton's method for multiobjective optimization. SIAM Journal on Optimization **20**(2), 602–626 (2009). https://doi.org/10.1137/08071692X
36. Fliege, J., Vaz, A.I.F.: A method for constrained multiobjective optimization based on SQP techniques. SIAM Journal on Optimization **26**(4), 2091–2119 (2016). https://doi.org/10.1137/15M1016424
37. Prudente, L.F., Souza, D.R.: A Quasi-Newton Method with Wolfe line searches for multiobjective optimization. Journal of Optimization Theory and Applications **194**(3), 1107–1140 (2022). https://doi.org/10.1007/s10957-022-02072-5
38. Lapucci, M., Mansueto, P.: A limited memory quasi-Newton approach for multi-objective optimization. Computational Optimization and Applications **85**(1), 33–73 (2023). https://doi.org/10.1007/s10589-023-00454-7
39. Cocchi, G., Liuzzi, G., Lucidi, S., Sciandrone, M.: On the convergence of steepest descent methods for multiobjective optimization. Computational Optimization and Applications, 1–27 (2020) https://doi.org/10.1007/s10589-020-00192-0
40. Custódio, A.L., Madeira, J.A., Vaz, A.I.F., Vicente, L.N.: Direct multisearch for multiobjective optimization. SIAM Journal on Optimization **21**(3), 1109–1140 (2011). https://doi.org/10.1137/10079731X
41. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017
42. Lapucci, M., Mansueto, P., Schoen, F.: A memetic procedure for global multi-objective optimization. Math. Prog. Comp. 15, 227–267 (2023) https://doi.org/10.1007/s12532-022-00231-3
43. De Ath, G., Everson, R.M., Rahat, A.A.M., Fieldsend, J.E.: Greed is good: Exploration and exploitation trade-offs in Bayesian optimisation. ACM Trans. Evol. Learn. Optim. **1**(1) (2021) https://doi.org/10.1145/3425501
44. Gupta, S., Shilton, A., Rana, S., Venkatesh, S.: Exploiting Strategy-Space Diversity for Batch Bayesian Optimisation. In: Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS), vol. 84. Lanzarote (Spain), pp. 1–10 (2018). https://proceedings.mlr.press/v84/gupta18a.html
45. Deb, K.: In: Wang, L., Ng, A.H.C., Deb, K. (eds.) Multi-objective Optimisation Using Evolutionary Algorithms: An Introduction, pp. 3–34. Springer, London (2011). https://doi.org/10.1007/978-0-85729-652-8_1
46. Chen, J., Luo, F., Li, G., Wang, Z.: Batch Bayesian optimization with adaptive batch acquisition functions via multi-objective optimization. Swarm and Evolutionary Computation 79, 101293 (2023) https://doi.org/10.1016/j.swevo.2023.101293
47. Feng, Z., Zhang, Q., Zhang, Q., Tang, Q., Yang, T., Ma, Y.: A multiobjective optimization based framework to balance the global exploration and local exploitation in expensive optimization. Journal of Global Optimization **61**(4), 677–694 (2015). https://doi.org/10.1007/s10898-014-0210-2
48. Zhang, Q., Li, H.: Moea/d: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation **11**(6), 712–731 (2007). https://doi.org/10.1109/TEVC.2007.892759

49. Bischl, B., Wessing, S., Bauer, N., Friedrichs, K., Weihs, C.: MOI-MBO: Multiobjective infill for parallel model-based optimization. In: Pardalos, P.M., Resende, M.G.C., Vogiatzis, C., Walteros, J.L. (eds.) Learning and Intelligent Optimization, pp. 173–186. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-09584-4_17
50. Lyu, W., Yang, F., Yan, C., Zhou, D., Zeng, X.: Batch Bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 3306–3314. PMLR, Stockholmsmässan, Stockholm (Sweden) (2018). https://proceedings.mlr.press/v80/lyu18a.html
51. Zhang, S., Yang, F., Yan, C., Zhou, D., Zeng, X.: An efficient Batch-constrained Bayesian optimization approach for analog circuit synthesis via multiobjective acquisition ensemble. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **41**(1), 1–14 (2022). https://doi.org/10.1109/TCAD.2021.3054811
52. Robič, T., Filipič, B.: Demo: Differential evolution for multiobjective optimization. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) Evolutionary Multi-Criterion Optimization, pp. 520–533. Springer, Berlin, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31880-4_36
53. Zhu, C., Byrd, R.H., Lu, P., Nocedal, J.: Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. ACM Trans. Math. Softw. **23**(4), 550–560 (1997). https://doi.org/10.1145/279232.279236
54. Clark, C.E.: The greatest of a finite set of random variables. Operations Research **9**(2), 145–162 (1961). https://doi.org/10.1287/opre.9.2.145
55. Ginsbourger, D., Le Riche, R., Carraro, L.: A Multi-points Criterion for Deterministic Parallel Global Optimization based on Gaussian Processes. Technical report, Ecole Nationale Supérieure des Mines (March 2008). https://hal.science/hal-00260579
56. Chevalier, C., Ginsbourger, D.: Fast computation of the multi-points Expected Improvement with applications in Batch selection. In: Nicosia, G., Pardalos, P. (eds.) Learning and Intelligent Optimization, pp. 59–69. Springer, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-44973-4_7
57. Marmin, S., Chevalier, C., Ginsbourger, D.: Differentiating the multipoint Expected Improvement for optimal Batch design. In: Pardalos, P., Pavone, M., Farinella, G.M., Cutello, V. (eds.) Machine Learning, Optimization, and Big Data, pp. 37–48. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27926-8_4
58. Hansen, P., Jaumard, B.: Cluster analysis and mathematical programming. Mathematical Programming **79**(1), 191–215 (1997). https://doi.org/10.1007/BF02614317
59. Hartigan, J.A., Wong, M.A.: Algorithm as 136: A k-means clustering algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics) 28(1), 100–108 (1979) https://doi.org/10.2307/2346830
60. Lloyd, S.: Least squares quantization in pcm. IEEE Transactions on Information Theory **28**(2), 129–137 (1982). https://doi.org/10.1109/TIT.1982.1056489
61. Gurobi Optimization, LLC: Gurobi Optimizer Reference Manual (2023). https://www.gurobi.com
62. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming **106**(1), 25–57 (2006). https://doi.org/10.1007/s10107-004-0559-y
63. Santoni, M.L., Raponi, E., Leone, R.D., Doerr, C.: Comparison of high-dimensional Bayesian optimization algorithms on BBOB. ACM Trans. Evol. Learn. Optim. **4**(3), (2024) https://doi.org/10.1145/3670683
64. Ament, S., Daulton, S., Eriksson, D., Balandat, M., Bakshy, E.: Unexpected improvements to Expected Improvement for Bayesian optimization. In: Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., Levine, S. (eds.) Advances in Neural Information Processing Systems, vol. 36, pp. 20577–20612 (2023). https://proceedings.neurips.cc/paper_files/paper/2023/file/419f72cbd568ad62183f8132a3605a2a-Paper-Conference.pdf
65. Törn, A., Zilinskas, A.: Global Optimization vol. 350. Springer, Springer Berlin, Heidelberg (1989). https://doi.org/10.1007/3-540-50871-6
66. Hartman, J.K.: Some experiments in global optimization. Naval Research Logistics Quarterly **20**(3), 569–576 (1973). https://doi.org/10.1002/nav.3800200316
67. Rosenbrock, H.: An automatic method for finding the greatest or least value of a function. The computer journal **3**(3), 175–184 (1960). https://doi.org/10.1093/comjnl/3.3.175

68. Bäck, T., Schwefel, H.-P.: An overview of evolutionary algorithms for parameter optimization. Evolutionary computation **1**(1), 1–23 (1993). https://doi.org/10.1162/evco.1993.1.1.1
69. Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.A.: A novel population initialization method for accelerating evolutionary algorithms. Computers & Mathematics with Applications **53**(10), 1605–1614 (2007). https://doi.org/10.1016/j.camwa.2006.07.013
70. Branin, F.H.: Widely convergent method for finding multiple solutions of simultaneous nonlinear equations. IBM Journal of Research and Development **16**(5), 504–522 (1972). https://doi.org/10.1147/rd.165.0504
71. Schwefel, H.-P.: Numerical Optimization of Computer Models. John Wiley & Sons, Inc., Hoboken, New Jersey, United States (1981). https://dl.acm.org/doi/abs/10.5555/539468
72. Levy, A.V., Montalvo, A., Gomez, S., Calderon, A.: Topics in global optimization. In: Hennart, J.P. (ed.) Numerical Analysis, pp. 18–33. Springer, Berlin, Heidelberg (1982). https://doi.org/10.1007/BFb0092957
73. Molga, M., Smutnicki, C.: Test functions for optimization needs (2005). https://marksmannet.com/RobertMarks/Classes/ENGR5358/Papers/functions.pdf
74. Bukin, A.: New Minimization Strategy For Non-Smooth Functions. Budker Institute of Nuclear Physics preprint BUDKER-INP-1997-79, Novosibirsk (1997)
75. Styblinski, M., Tang, T.-S.: Experiments in nonconvex optimization: stochastic approximation with function smoothing and simulated annealing. Neural Networks **3**(4), 467–483 (1990). https://doi.org/10.1016/0893-6080(90)90029-K
76. Jamil, M., Yang, X.-S.: A literature survey of benchmark functions for global optimisation problems. International Journal of Mathematical Modelling and Numerical Optimisation **4**(2), 150–194 (2013). https://doi.org/10.1504/IJMMNO.2013.055204
77. Opacic, J.: A heuristic method for finding most extrema of a nonlinear functional. IEEE Transactions on Systems, Man, and Cybernetics **1**, 102–107 (1973). https://doi.org/10.1109/TSMC.1973.5408584
78. Cappanera, P., Di Gangi, L., Lapucci, M., Pellegrini, G., Roma, M., Schoen, F., Sortino, A.: Integrated task scheduling and personnel rostering of airports ground staff: A case study. Expert Systems with Applications 238, 121953 (2024) https://doi.org/10.1016/j.eswa.2023.121953
79. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Mathematical Programming **91**(2), 201–213 (2002). https://doi.org/10.1007/s101070100263
80. Hvarfner, C., Hellsten, E.O., Nardi, L.: Vanilla Bayesian Optimization Performs Great in High Dimensions (2024). https://doi.org/10.48550/arXiv.2402.02229
81. Grosnit, A., Cowen-Rivers, A.I., Tutunov, R., Griffiths, R.-R., Wang, J., Bou-Ammar, H.: Are we forgetting about compositional optimisers in Bayesian optimisation? Journal of Machine Learning Research **22**(160), 1–78 (2021)