Neurocomputing 546 (2023) 126290

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Improved max-value entropy search for multi-objective bayesian optimization with constraints

Daniel Fernández-Sánchez [a,*], Eduardo C. Garrido-Merchán [b], Daniel Hernández-Lobato [a]

[a] Computer Science Department, Universidad Autónoma de Madrid, Francisco Tomás y Valiente 11, 28049 Madrid, Spain
[b] Faculty of Economics and Business Administration, Comillas Pontifical University, Alberto Aguilera 23, 28015 Madrid, Spain

ARTICLE INFO

Article history:
Received 20 April 2022
Revised 23 February 2023
Accepted 30 April 2023
Available online 12 May 2023
Communicated by Zidong Wang

Keywords:
Bayesian optimization
Constrained multi-objective scenario
Information theory

ABSTRACT

We present MESMOC+, an improved version of Max-value Entropy search for Multi-Objective Bayesian optimization with Constraints (MESMOC). MESMOC+ can be used to solve constrained multi-objective problems when the objectives and the constraints are expensive to evaluate. It is based on minimizing the entropy of the solution of the optimization problem in function space (i.e., the Pareto front) to guide the search for the optimum. The cost of MESMOC+ is linear in the number of objectives and constraints. Furthermore, it is often significantly smaller than the cost of alternative methods based on minimizing the entropy of the Pareto set. The reason for this is that it is easier to approximate the required computations in MESMOC+. Moreover, MESMOC+'s acquisition function is expressed as the sum of one acquisition per each black-box (objective or constraint). Therefore, it can be used in a decoupled evaluation setting in which it is chosen not only the next input location to evaluate, but also which black-box to evaluate there. We compare MESMOC+ with related methods in synthetic, benchmark and real optimization problems. These experiments show that MESMOC+ has similar performance to that of state-of-the-art acquisitions based on entropy search, but it is faster to execute and simpler to implement. Moreover, our experiments also show that MESMOC+ is more robust with respect to the number of samples of the Pareto front.

1. Introduction

There are many problems where the goal is to optimize several objectives $f_1(\mathbf{x}), \ldots, f_K(\mathbf{x})$ while fulfilling certain constraints $c_1(\mathbf{x}) \geqslant 0, \ldots, c_C(\mathbf{x}) \geqslant 0$, where K and C are the number of objectives and constraints, respectively. Usually, the input space $\mathscr{X}$ is bounded, *i.e.*, we optimize in $\mathscr{X} \subset \mathbb{R}^d$, where d is the dimensionality of $\mathscr{X}$. For example, one might want to maximize the speed of a robot while simultaneously minimizing its energy consumption [1]. Moreover, one would like to avoid breaking any of its joints. To achieve this, one could change the dimensions of the robot's gears and the materials of its manufacturing process. Another example would be to minimize the classification error of a deep neural network while at the same time minimizing the time needed to predict. A constraint can be not exceeding a certain amount of memory space. In this second example, one could modify the learning rate of the network, the number of hidden layers and the number of neurons in each hidden layer.

In problems where several objectives are optimized, most of the time there is no single optimal point but a set of optimal points: the Pareto set $\mathscr{X}^\star$ [2]. The objective values associated to the points in $\mathscr{X}^\star$ constitute the Pareto front $\mathscr{Y}^\star$. All the points in the Pareto set are optimal because they are not *Pareto dominated* by any other point in $\mathscr{X}$. In a minimization context, a point $\mathbf{x}_1$ *Pareto dominates* $\mathbf{x}_2$ if $f_k(\mathbf{x}_1) \leqslant f_k(\mathbf{x}_2), \forall k = \{1, \ldots, K\}$, with at least one strictly minor inequality. This means that given the Pareto set, it is impossible to improve the value in one objective without deteriorating the values for the other objectives. Pareto points dominate all other points in $\mathscr{X}$ since they achieve better results in at least one objective. If constraints are considered, the points of the Pareto set must be valid, *i.e.*, they must satisfy $c_j(\mathbf{x}) \geqslant 0, \forall j = \{1, \ldots, C\}, \forall \mathbf{x} \in \mathscr{X}^\star$. Then, Pareto points are simply the non-dominated points in the feasible space $\mathscr{F}$, *i.e.*, the subset of points from $\mathscr{X}$ that satisfy all constraints. Importantly, however, as generally the potential size of $\mathscr{X}^\star$ is infinite (and therefore also that of $\mathscr{Y}^\star$), it is necessary to approximate the Pareto set in the feasible space by a finite set of points. Given an approximation, a final user may pick-up a point

* Corresponding author.
E-mail addresses: daniel.fernandezs@uam.es (D. Fernández-Sánchez), ecgarrido@icade.comillas.edu (E.C. Garrido-Merchán), daniel.hernandez@uam.es (D. Hernández-Lobato).

https://doi.org/10.1016/j.neucom.2023.126290
0925-2312/

from this set according to the objective preferences, *e.g.*, prediction error versus computational time in the case of the neural network.

The problems described above have three main characteristics. First, there is no analytical form for the objectives nor the constraints, *i.e.*, they can be considered as black-boxes. For example, in the case of the robot we may have to carry out an experiment with the actual robot to evaluate the objectives or the constraints. In the case of the neural network we will have to train it to measure the prediction error. Second, the evaluations may be contaminated by noise. This means that two evaluations at the same input can produce different results. In the case of the robot, this can be due the impact of the experimental conditions or errors in the robot's building process. In the case of the neural network, this can be due to a different initialization of the weights. Third, new evaluations are quite expensive in some way, *e.g.*, economically or temporally. For example, in the case of the robot, building the actual robot may take several days, or we will have to carry out an expensive computer simulation. In the case of the neural network, training the network can take several hours. To solve the type of problems described efficiently and reduce the number of evaluations of the black-boxes performed, one can use a set of techniques called Bayesian optimization (BO) [3].

BO methods first build a probabilistic model to estimate the potential values of the black-boxes in the unexplored regions of the input space $\mathscr{X}$. Typically, this model is a GP [4]. Then, they use an acquisition function to measure the expected utility of performing an evaluation at each potential point of $\mathscr{X}$. This acquisition function uses the information provided by the probabilistic model. The steps described are repeated iteratively. Namely, at each iteration the BO method evaluates the selected point, updates the probabilistic models, and computes and maximizes the acquisition function to choose the next point at which to evaluate the objectives and the constraints. This process is repeated for a fixed number of iterations, determined by the computational budget. Once the BO method is finished, the probabilistic models are optimized to obtain an approximate solution of the optimization problem. Summing up, BO methods use the probabilistic models to guide the search and carefully choose the next point to evaluate. If the black-boxes are expensive, this can save a lot of expensive evaluations [5]. Specifically, using the probabilistic models will avoid performing evaluations in regions of the input space that are not very useful. The key for BO success is that the acquisition function is very cheap to evaluate compared to the black-boxes. Therefore, it is worth spending a bit of time choosing in an intelligent way the point that should be evaluated next. Nevertheless, in problems where a lot of evaluations are needed, a fast acquisition function is expected to be more useful than a slow acquisition function.

In a constrained multi-objective optimization scenario a decoupled evaluation setting becomes important [6]. Specifically, the evaluation of black-boxes often involves performing different experiments or simulations. Recalling the example of the robot, we may perform a simulation to know if any joint breaks, whereas to measure its speed or energy consumption, we may need to build and test the robot. Similarly, in the neural network example, knowing its classification error requires training and validating the network. By contrast, to determine its prediction time, or if it needs more memory than available, it is only necessary to build the network using, *e.g.*, random weights. Therefore, in the setting described it may be important to choose not only where to evaluate the black-boxes next, but also what black-box to evaluate. This corresponds to a decoupled evaluation setting. This is different from a coupled evaluation setting where one evaluates at each iteration all black-boxes at the candidate location.

In the literature, there are a few BO acquisition functions that can handle constrained multi-objective optimization problems. Some of these acquisition functions are adaptations of *Expected hyper-volume improvement* (EHI) [7] to handle constraints [8–10]. In a minimization context, the hyper-volume is the volume of points in function space above objective values $\hat{\mathscr{Y}}^\star$ associated to an estimate of the Pareto set $\hat{\mathscr{X}}^\star$, with respect to a reference point. The hyper-volume is maximized by the actual Pareto frontier $\mathscr{Y}^\star$ [11]. Therefore, it makes sense to try to choose the next point to evaluate as the one that is expected to maximize this quantity the most. The acquisition functions described have to deal with an intractable integral that arises from computing the expected hyper-volume improvement. The differences between these methods are due to the way they approximate the intractable integral. In [9,10] a Monte Carlo approximation samples from the predictive distribution of the probabilistic models. Although easy to implement, the approximate acquisition can be equal to zero in almost the entire input space after a few evaluations. The reason for this is the difficulty for improving the current hyper-volume just by sampling from the predictive distribution of the GP when the prediction uncertainty is small. The approach in [8] does not have this problem, but is more expensive to evaluate as it needs to run a Metropolis–Hastings algorithm for the approximation. In general, all these strategies, tend to explore too much and perform many evaluations at the boundaries of the input space [12]. Furthermore, none of these methods can handle a decoupled evaluation setting since it is not possible to measure the expected improvement in the hyper-volume by simply evaluating a single objective.

Entropy search is a set of successful acquisition functions [13–17,12,18,19]. Furthermore, some of them can handle multiple objectives and several constraints [12,19]. Here, we extend the unconstrained single-objective acquisition function *max-value entropy search* (MES) [17], which is based on the reduction of the entropy of the solution of the optimization problem in function space, to work with several objectives and constraints. In the constrained multi-objective case, the solution in function space is given by the Pareto front $\mathscr{Y}^\star$ associated to the Pareto set $\mathscr{X}^\star$ in the feasible space. Intuitively, if the entropy of the Pareto front is small, we expect to be closer to solving the optimization problem. Therefore, it makes sense to choose the next evaluation as the one that is expected to reduce the most this entropy. A difficulty is, however, that the resulting acquisition function is intractable. Thus, it needs to be approximated, and the approximation plays a critical role in the performance. While in the literature there are already some attempts to perform an extension of MES to the constrained multi-objective case [19], the existing approximations of the acquisition function are expected to perform poorly. In particular, the approximation considered by the authors of [19] results in a method that simply tries to maximize each objective and constraint independently. This is expected to miss the potential effects of the constraints in the solution of the optimization problem. It will also ignore relevant points of the Pareto front, and will only focus on the extreme points, *i.e.*, when each objective achieves the minimum possible value. Our approximation of the exact acquisition is more accurate, as shown by our experiments, which leads to better optimization results. Other extensions of entropy search for the constrained multi-objective setting consider reducing the entropy of the Pareto set in the feasible space $\mathscr{X}^\star$, instead of the entropy of the Pareto front $\mathscr{Y}^\star$ [12]. The acquisition function in this case is more complicated and more difficult to approximate. Our experiments show that our proposed approach performs similar to this technique, but the resulting acquisition function is cheaper to compute and to optimize.

We call our method *improved max-value entropy search for multi-objective optimization with constraints* (MESMOC+). At each iteration of the optimization process, MESMOC+ chooses to evaluate the point that is expected to reduce the most the entropy of the Pareto front $\mathscr{Y}^\star$. The entropy of $\mathscr{Y}^\star$ is characterized by the predic-

tive distribution of the probabilistic models for the objectives and constraints given the observed data. The reduction of the entropy of $\mathscr{y}^\star$ means that more information about the solution of the problem is available and that we are closer to finding its solution [13,14]. We have carried out several experiments involving synthetic and real optimization problems showing that MESMOC+ outperforms the method proposed in [19], to which we refer to as MESMOC. Moreover, it obtains similar and sometimes even better results than other state-of-the-art acquisition functions for multi-objective optimization with constraints from the literature [8,12]. Nonetheless, the computational cost of MESMOC+ per iteration is significantly smaller and it is much easier to implement. On the other hand, importantly, the expression of MESMOC+'s acquisition function is a sum of one acquisition per black-box function. This means that in a decoupled setting MESMOC+ can choose the next input location to evaluate and also which black-box to evaluate. For this, one only has to optimize the acquisition function of each black-box and choose the one with the largest acquisition value. In our experiments we compare the coupled and decoupled variants of MESMOC+, showing that sometimes a decoupled evaluation setting gives better optimization results over a coupled one.

Summing up, (i) MESMOC+ allows for decoupled evaluation settings, while in general, acquisition functions based on the expected improvement of the hyper-volume do not; (ii) MESMOC+ does not suffer from the high explorative behavior of those acquisition functions, which is translated into better optimization results; (iii) the approximation performed in MESMOC+ is more accurate than that of MESMOC and hence performs better; and (iv) MESMOC+ is cheaper to compute than other strategies that aim at reducing the entropy of the Pareto set instead of the Pareto front.

The rest of the article is organized as follows. In the next section, we review the BO methods available in the literature that can solve multi-objective constrained optimization problems. Next, in Section 3, we describe our proposed approach, MESMOC+, and the approach we use to deal with the intractability inherent to our acquisition function based on expected entropy reduction. In Section 4, we describe the benchmark, synthetic and real-world experiments performed and discuss the results obtained, showing that our proposed approach, MESMOC+, achieves similar performance to the state-of-the-art techniques, but is faster to run and simpler to implement. Finally, in Section 5, we give the conclusions of this work.

2. Related Work

In the literature there are other acquisition functions that can deal with multiple objectives and constraints. However, there is little work on the constrained multi-objective Bayesian optimization problem. In this section, we describe the few acquisition functions which, to our knowledge, can handle this kind of problems.

The acquisition function *Bayesian multi-objective optimization* (BMOO) is based on the *expected hyper-volume improvement* (EHI) [7]. In a minimization context, the hyper-volume of an estimate of the Pareto set $\hat{\mathscr{x}}^\star$ is the size of the space of points in function space above the corresponding estimated Pareto front $\hat{\mathscr{y}}^\star$ with respect to a reference point (usually given by the worst potential value for each objective). It is maximized by the actual Pareto front $\mathscr{y}^\star$ [11]. EHI simply chooses the next point as the one that improves the most the current estimate of the hyper-volume. To take into account the constraints, BMOO extends the Pareto dominance rule to introduce a preference for performing evaluations at points that are more likely to be feasible [8]. This extension consists in applying a transformation to the objective and constraint values to deal with situations where all points observed so far are invalid. This transformation is the following:

$$\Psi(\mathbf{y}^f, \mathbf{y}^c) = \begin{cases} (\mathbf{y}^f, \mathbf{0}) & \text{if } \mathbf{y}^c \geqslant 0 \\ (+\infty, \min(\mathbf{y}^c, \mathbf{0})) & \text{otherwise} \end{cases}, \tag{1}$$

where $\mathbf{y}^f$ and $\mathbf{y}^c$ are the vectors observations of the objectives and constraints, respectively. BMOO tries to maximize the expected increase in the hyper-volume after evaluating all black-boxes at a particular input location. Thus, it uses the acquisition function EHI with the extended Pareto dominance rule described above. Specifically, the acquisition function of BMOO measures the expected hyper-volume improvement in the extended space:

$$\alpha(\mathbf{x}) = \mathbb{E}_{\mathbf{y}^f, \mathbf{y}^c}\left[\int_{\mathscr{g}_N} \mathbb{I}(\Psi(\mathbf{y}^f, \mathbf{y}^c) \prec \Psi(\mathbf{y}))d\mathbf{y}\right], \tag{2}$$

where $\mathbf{a} \prec \mathbf{b}$ means that $\mathbf{a}$ is Pareto dominated by $\mathbf{b}$, $\mathbb{I}(\cdot)$ is the indicator function, $\mathscr{g}_N$ is the set of points in functional space not dominated until iteration N, and the expectation is w.r.t the predictive distribution of the GPs for the candidate point $\mathbf{x}$. Since Eq. (2) cannot be calculated analytically, in [8] the authors decided to swap the expectation and the integral, and to approximate the integral by using Monte Carlo samples from a uniform distribution in $\mathscr{g}_N$. However, generating these samples is complicated and they have to resort to a Metropolis–Hastings algorithm for that. BMOO was initially described for noiseless scenarios, but the method can also be applied when the black-boxes are contaminated with noise [12]. In that setting, it is often outperformed by strategies that can take into account evaluation noise. Additionally, the Metropolis–Hastings algorithm is very costly to carry out, which means that BMOO is an expensive acquisition function. Finally, in practice, BMOO does not allow for decoupled evaluations. The reason is that the improvement in the hyper-volume by evaluating a single objective is always zero.

In the literature there are other techniques that have been proposed to approximate the expected hyper-volume improvement. Namely, qEHVI and qNEHVI [9,10]. Both qEHVI and qNEHVI allow for batch evaluation settings in which one chooses not only one candidate point to evaluate next, but a batch of points, to be evaluated in parallel. Notwithstanding, they can also be used in the single point setting we consider in our work. While qEHVI considers a noiseless evaluation setting, qNEHVI allows to take into account evaluation noise. Both strategies approximate the intractable integral of the expected improvement of the hyper-volume (*i.e.*, the one corresponding to Eq. (2), without the extended Pareto dominance rule) by decomposing in boxes the functional space and by sampling from the predictive distribution of the GPs using Monte Carlo techniques combined with the reparameterization trick [20]. That is, unlike BMOO they do not swap the expectation and the integral in (2), and simply approximate the expectation using Monte Carlo samples. To take noise into account qNEHVI also generates Monte Carlo samples of the black-boxes at the points already evaluated. They incorporate constraints simply by multiplying the acquisition function by the probability of feasibility. The reparameterization trick allows to easily compute the gradients of the Monte Carlo approximation using automatic differentiation tools [21]. Such an approximation can be problematic, however. In particular, after each evaluation it will be more and more difficult to improve the current hyper-volume, since the predictive variance will be reduced. That means that, unless a large number of samples are used in the approximation, the acquisition function will be flat and equal to zero in a big region of the input space. This will make difficult optimizing the approximate acquisition function using gradient based techniques. This, together with their observed tendency of exploring too much and evaluating at the boundaries of the input space may lead to a poor performance. Again, since these strategies are also based on the expected improvement of the hyper-volume, they do not allow for decou-

pled evaluations, unlike the proposed approach MESMOC+. The computational cost of qEHVI and qNEHVI is super-polynomial in the number of objectives [9,10]. Moreover, qNEHVI is more expensive since it has to sample from the GPs modeling the objectives at the candidate location at which to evaluate the acquisition function and also at the already evaluated points.

Predictive entropy search for multi-objective optimization with constraints (PESMOC) is another acquisition function that focuses on the reduction of the entropy of the solution of the optimization problem [12]. However, it targets the entropy of the Pareto set $\mathscr{X}^\star$, instead of the entropy of the Pareto front $\mathscr{Y}^\star$. This acquisition function is hence the expected reduction in the entropy of $\mathscr{X}^\star$. Similar to MESMOC+, the entropy of $\mathscr{X}^\star$ is intractable and requires complicated approximations. PESMOC rewrites the expression of the acquisition function noting that the expected reduction in the entropy of $\mathscr{X}^\star$ is the mutual information between $\mathscr{X}^\star$ and $\mathbf{y}$. Because the mutual information is symmetric, it is equivalent to the mutual information between $\mathbf{y}$ and $\mathscr{X}^\star$. Doing this, the acquisition function of PESMOC is:

$$\alpha(\mathbf{x}) = H(\mathbf{y}|\mathscr{D},\mathbf{x}) - \mathbb{E}_{\mathscr{X}^\star}\left[H(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{X}^\star)\right], \tag{3}$$

where the first term of the r.h.s. of (3) is the same as in MESMOC+, *i.e.* the entropy of the predictive distribution, the expectation is w.r.t. the Pareto set $\mathscr{X}^\star$ instead of $\mathscr{Y}^\star$ and the second term of the r.h.s. is the entropy of the predictive distribution conditioned to $\mathscr{X}^\star$ being the solution to the problem. See Section 3 for further details on MESMOC+. The second term of the r.h.s. of Eq. (3) is intractable and must be approximated. The expectation is approximated also by a Monte Carlo average, as in MESMOC+. Furthermore, the method for generating the samples of $\mathscr{X}^\star$ is equivalent to the one used in MESMOC+ (see Section 3). To approximate the entropy of $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{X}^\star)$ they use expectation propagation (EP) [22].

PESMOC is different from MESMOC+ since the former approximates $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{X}^\star)$ and the latter approximates $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{Y}^\star)$. Importantly, the approximation of PESMOC is more complicated than the one of MESMOC+ because $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{X}^\star)$ has more non-Gaussian factors and some of them depend on two variables, which involves working with bi-variate Gaussian distributions. By contrast, all the factors in $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{Y}^\star)$ are univariate, which means that MESMOC+ only has to use one-dimensional Gaussians. The consequence is that MESMOC+'s acquisition function is significantly less expensive to compute and easier to implement. Our experiments show that MESMOC+ gives similar and sometimes better results to those of PESMOC. We have also observed that MESMOC+ is more robust with respect to the number of samples M of the Pareto front than PESMOC, a result that was also observed in the unconstrained single-objective case [17].

The original formulation of *max-value entropy search* (MES) can only deal with single-objective unconstrained problems [17]. There are some extensions of MES to work with one objective and several constraints [23] and with several objectives and no constraints [18,24]. However, none of these methods can deal with multiple objectives and constraints simultaneously. Moreover, the extension to multiple objectives and several constraints is not trivial because one has to handle the restrictions about the objectives (to avoid solutions that dominate the Pareto front) and the constraints (to ignore solutions that are infeasible). These dependencies do not appear in the single-objective nor in the constrained single-objective scenario. As a matter of fact, in the unconstrained single objective setting, the conditional distributions is simply approximated by a truncated Gaussian [17].

Max-value entropy search for multi-objective optimization with constraints (MESMOC) is an acquisition function developed in an independent work [19], which also minimizes the entropy of $\mathscr{Y}^\star$, as MESMOC+ does. Thus, the expression considered by MESMOC for the acquisition function is also given by Eq. (8). However, the proposed approximation for the entropy of $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{Y}^\star)$ is very different. In [19], they focus on maximizing the objectives, while we are interested in minimization. Consider a maximization scenario. Ignore the constraints initially. Let the sampled Pareto frontier be $\mathscr{Y}^\star = \{\mathbf{z}_1,\ldots,\mathbf{z}_S\}$ with S the size of $\mathscr{Y}^\star$ and each $\mathbf{z}_j$, for $j = 1,\ldots,S$, a vector of size K with the associated objective values. Let the j-th component of $\mathbf{z}_i$ be z_i^j. In [19] they argue that a sufficient condition for some point $\mathbf{y}$ being compatible with $\mathscr{Y}^\star$ as the solution of the problem is that $y^j \leqslant \max\left\{z_1^j,\ldots,z_S^j\right\}\ \forall j \in \{1,\ldots,K\}$ (recall a maximization scenario). That is, the value of $\mathbf{y}$ for the j-th objective cannot be larger than the largest value for that objective, according to $\mathscr{Y}^\star$. However, this condition is not complete because $\mathbf{y}$ can be optimal, (*i.e.*, $\mathbf{y}$ is incompatible with $\mathscr{Y}^\star$) even if none of its values are larger than the maximum value for the corresponding objective. E.g., let $K = 2$ and $\mathscr{Y}^\star = \{(1,0),(0,1)\}$. Consider now the point $\mathbf{y} = (0.7, 0.7)$. The components of $\mathbf{y}$ are smaller than $1 = \max\left\{z_1^j,\ldots,z_S^j\right\}\forall j \in \{1,\ldots,K\}$, but this point is optimal and non-dominated by any of the points in $\mathscr{Y}^\star$. This means that $\mathbf{y}$ is not compatible with $\mathscr{Y}^\star$. However, the condition employed in [19], *i.e.*, that $\mathbf{y}$ must satisfy $y_j \leqslant \max\left\{z_1^j,\ldots,z_S^j\right\}\ \forall j \in \{1,\ldots,K\}$, will consider $(0.7, 0.7)$ as a potential point to be predicted by the conditional predictive distribution given $\mathscr{Y}^\star$, $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{Y}^\star)$, which is incorrect. The constraints are incorporated in [19] in an ad hoc way, simply by enforcing that $c_j(\mathbf{x}) \leqslant \max\left\{\tilde{z}_1^j,\ldots,\tilde{z}_S^j\right\}$ for $j = 1,\ldots,C$, where $\{\tilde{\mathbf{z}}_i\}_{i=1}^S$ are the constraint values associated to the points in $\mathscr{Y}^\star$. That is, the constraint values have to be smaller than the maximum constraint values associated to the Pareto front $\mathscr{Y}^\star$, as it is done with the objectives. There is no justification for this.

Summing up, the conditional predictive distribution is given by a factorizing truncated Gaussian distribution. In particular, for the objectives each $f_j(\mathbf{x}) \leqslant \max\left\{z_1^j,\ldots,z_S^j\right\}\ \forall j \in \{1,\ldots,K\}$, and for the constraints, each $c_j(\mathbf{x}) \leqslant \max\left\{\tilde{z}_1^j,\ldots,\tilde{z}_S^j\right\}$ for $j = 1,\ldots,C$. The entropy of a truncated Gaussian distribution has a closed-form expression [17]. Therefore, the approximate acquisition function of MESMOC is

$$\alpha(\mathbf{x}) \approx \frac{1}{M}\sum_{m=1}^{M}\left[\sum_{k=1}^{K}\left(\frac{\varrho_{k,m}^f\phi\left(\varrho_{k,m}^f\right)}{2\Phi\left(\varrho_{k,m}^f\right)} - \log\Phi\left(\varrho_{k,m}^f\right)\right) + \sum_{j=1}^{C}\left(\frac{\varrho_{j,m}^c\phi\left(\varrho_{j,m}^c\right)}{2\left(\Phi\left(\varrho_{j,m}^c\right)\right)} - \log\left(\Phi\left(\varrho_{j,m}^c\right)\right)\right)\right], \tag{4}$$

where $\varrho_{k,m}^f = \varrho_{k,m}^f(\mathbf{x})$, $\varrho_{j,m}^c = \varrho_{j,m}^c(\mathbf{x})$,

$$\varrho_{k,m}^f(\mathbf{x}) = \frac{\zeta_{k,m}^\star - m_k^f(\mathbf{x})}{\left(v_k^f(\mathbf{x})\right)^{1/2}}, \tag{5a}$$

$$\varrho_{j,m}^c(\mathbf{x}) = \frac{\zeta_{j,m}^\star - m_j^c(\mathbf{x})}{\left(v_j^c(\mathbf{x})\right)^{1/2}}, \tag{5b}$$

and

$$\zeta^{\star}_{k,m} = \min\left\{z_1^{k,m}, \ldots, z_S^{k,m}\right\} \ \forall k \in \{1, \ldots, K\}, \varsigma^{\star}_{j,m} = \max\left\{\tilde{z}_1^{j,m}, \ldots, \tilde{z}_S^{j,m}\right\} \ \forall j \in \{1, \ldots, C\}, \quad (6)$$

and S the size of $\mathscr{Y}^{\star}_{(m)}$, *i.e.*, the m-th sample of $\mathscr{Y}^{\star}$. The samples of $\mathscr{Y}^{\star}$ are obtained as in MESMOC+. Namely, by optimizing samples from the GP posterior distribution using a random feature approximation. Moreover, $z_s^{k,m}$ is the k-th objective value associated to the s-th point in $\mathscr{Y}^{\star}_{(m)}$, and $\tilde{z}_s^{j,m}$ is the j-th constraint value associated to the s-th point in $\mathscr{Y}^{\star}_{(m)}$. Note that (4) also involves a sum of one acquisition function per constraint and objective, which means that MESMOC can be used in a decoupled evaluation setting.

Critically, in the provided code in [19], the implementation considered that $\mathscr{Y}^{\star}_{(m)}$ is given by the Pareto front sample, and all the evaluations performed so far. The consequence is that the acquisition proposed in [19] is simply the sum of the standard MES acquisition function for each objective and constraint [17]. This makes sense, and is equivalent to maximizing all the objectives and constraints independently. Maximizing each objective is expected to give good solutions. Maximizing each constraint is expected to provide feasible solutions. This makes, however, difficult finding interesting points of the Pareto frontier that are not close to the maximum values of each objective. Such a strategy is hence expected to be sub-optimal. Besides this, the optimization of the resulting acquisition function in [19] is restricted to those regions of the input space in which the GP means for the constraints are strictly positive. This becomes problematic in problems in which finding feasible points is difficult. In particular, if all the observations are infeasible, the GP means for the constraints will be negative in all the input space (even though the associated GP variance can be high). This will make the BO method fail. If that is the case, a simple solution is to choose randomly the next point to evaluate.

Our experiments show that the accuracy of MESMOC for approximating the acquisition in Eq. (8) is worse than that our proposed method MESMOC+. Furthermore, in several optimization problems MESMOC+ outperforms MESMOC. We believe this is due to the accuracy of our approximation and the limitations of MESMOC to find feasible solutions.

Besides BO methods, meta-heuristics can also solve constrained multi-objective problems. However, these techniques usually require a large number of evaluations to achieve good results [25]. They are specifically designed to optimize objective functions that are cheap to evaluate. They are not adequate to solve problems where the objectives and the constraints are very expensive to evaluate. By contrast, BO methods exploit the information provided by the predictive distribution of the surrogate model to make intelligent decisions about where to evaluate next these functions with the goal of solving the problem in a few steps. Therefore, they are expected to perform much better in a scenario that includes a limited evaluation budget. Summing up, BO methods and meta-heuristics target different problems. Moreover, there is already have evidence from the literature of the superiority of BO methods in the problems we are interested in when compared to meta-heuristics [26].

3. Improved Max-value Entropy Search for Multi-objective Bayesian Optimization with Constraints

In this section, we give the details of the proposed acquisition function *improved max-value entropy search for multi-objective optimization with constraints* (MESMOC+). In BO methods, the maximum of the acquisition function indicates the next point at which to evaluate the black-boxes. To compute the acquisition function, the information provided by the predictive distribution of the probabilistic models is used, and each time that a new point is evaluated, the probabilistic models are updated. Usually, Gaussian processes (GPs) [4] are the probabilistic models employed. For further details on the use of GPs as the underlying model in BO we refer the reader to [5]. Briefly speaking, a GP provides a predictive distribution for each objective or constraint value at each potential input location given the observed data. In our work we assume that the black-boxes are generated from a GP a priori with i.i.d. Gaussian noise with zero mean. For simplicity, the development of MESMOC+ is carried out considering a coupled evaluation setting, in which all black-boxes are evaluated at the same point, later on we explain how to use MESMOC+ in a decoupled setting.

Let $\mathscr{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N}$ be the dataset with the evaluations of the objectives and constraints performed up to iteration N, where $\mathbf{x}_n$ is the input evaluated in the n-th iteration and $\mathbf{y}_n$ is a vector with the values obtained when evaluating the $K + C$ black-boxes at $\mathbf{x}_n$, *i.e.*, $\mathbf{y}_n = (f_1(\mathbf{x}_n), \ldots, f_K(\mathbf{x}_n), c_1(\mathbf{x}_n), \ldots, c_C(\mathbf{x}_n))$. Since MESMOC+ consists in reducing the entropy of the solution in the functional space after an $\mathbf{x}_{N+1}$ evaluation, the MESMOC+ acquisition function is:

$$\alpha(\mathbf{x}) = H(\mathscr{Y}^{\star}|\mathscr{D}) - \mathbb{E}_{\mathbf{y}}\left[H(\mathscr{Y}^{\star}|\mathscr{D} \cup \{(\mathbf{x}, \mathbf{y})\})\right], \quad (7)$$

where $H(\mathscr{Y}^{\star}|\mathscr{D})$ is the entropy of the Pareto front, $\mathscr{Y}^{\star}$, given by the probabilistic models, after observing the current dataset $\mathscr{D}$; $H(\mathscr{Y}^{\star}|\mathscr{D} \cup \{(\mathbf{x}, \mathbf{y})\})$ is the entropy of $\mathscr{Y}^{\star}$ after including the new data point $(\mathbf{x}, \mathbf{y})$ in the dataset; and the expectation $\mathbb{E}_{\mathbf{y}}[\cdot]$ is calculated over the potential values for $\mathbf{y}$ at $\mathbf{x}$, according to the GPs.

Critically, evaluating the entropy of $\mathscr{Y}^{\star}$ is very challenging. In particular, $\mathscr{Y}^{\star}$ is a set of potentially infinite size. In order to avoid this problem, we follow [16] and rewrite Eq. (7) in an equivalent form, as suggested in [17], by noting that Eq. (7) is exactly the mutual information between $\mathscr{Y}^{\star}$ and $\mathbf{y}, I(\mathscr{Y}^{\star}; \mathbf{y})$ [15,16]. Therefore, since $I(\mathscr{Y}^{\star}; \mathbf{y}) = I(\mathbf{y}; \mathscr{Y}^{\star})$, then, we can swap the roles of $\mathscr{Y}^{\star}$ and $\mathbf{y}$ in Eq. (7) and the MESMOC+ acquisition function becomes:

$$\alpha(\mathbf{x}) = H(\mathbf{y}|\mathscr{D}, \mathbf{x}) - \mathbb{E}_{\mathscr{Y}^{\star}}\left[H(\mathbf{y}|\mathscr{D}, \mathbf{x}, \mathscr{Y}^{\star})\right], \quad (8)$$

where $H(\mathbf{y}|\mathscr{D}, \mathbf{x})$ is the entropy of $p(\mathbf{y}|\mathscr{D}, \mathbf{x})$, *i.e.*, the entropy of the predictive distribution of the GPs at $\mathbf{x}$. This is simply a Gaussian distribution. Note that now the expectation is with respect to potential values of $\mathscr{Y}^{\star}$; and $H(\mathbf{y}|\mathscr{D}, \mathbf{x}, \mathscr{Y}^{\star})$ is the entropy of the predictive distribution conditioned to the Pareto front being the solution of the problem.

The expression given in Eq. (8) is the acquisition function targeted by MESMOC+. Thus, at each iteration, the next point to evaluate is chosen as the maximizer of Eq. (8), *i.e.* $\mathbf{x}_{N+1} = \arg\max_{\mathbf{x} \in \mathscr{X}} \alpha(\mathbf{x})$. Note that it is easier to work with this expression than with Eq. (7) because here we do not have to evaluate the entropy of $\mathscr{Y}^{\star}$, a set of potential infinite size. The first term of the r.h.s of Eq. (8) is simply the entropy of the current predictive distribution, and since we assume that there is no correlation between the black-boxes, its expression is the sum of the entropy of $K + C$ Gaussian distributions. These are the predictive distributions of the GPs for each objective and constraint. Namely:

$$H(\mathbf{y}|\mathscr{D}, \mathbf{x}) = \sum_{k=1}^{K} \frac{\log\left(2\pi e v_k^f\right)}{2} + \sum_{j=1}^{C} \frac{\log\left(2\pi e v_j^c\right)}{2}, \quad (9)$$

where $v_k^f = v_k^f(\mathbf{x})$ and $v_j^c = v_j^c(\mathbf{x})$ are the predictive variance for the k-th and j-th objective and constraint respectively. Nevertheless, the evaluation of the second term in the r.h.s. of Eq. (8) is intractable. The expectation can, however, be approximated by generating Monte Carlo samples of $\mathscr{Y}^{\star}$. Using these samples, one can estimate the entropy of $p(\mathbf{y}|\mathscr{D}, \mathbf{x}, \mathscr{Y}^{\star})$ and average the results. To generate

samples of $\mathscr{y}^\star$, we follow the same approach as in [12]. We first use a random feature approximation of the GPs (see [27] for further details) to generate samples of the objectives and constraints, and then we optimize these samples using a grid of points to obtain the final sample of $\mathscr{y}^\star$, the Pareto frontier associated to the Pareto set in the feasible space. This step is cheap because we can evaluate the GP samples at a small cost, unlike the actual black-boxes. Note that an alternative approach is to use more advanced GP sampling techniques like the one described in [28]. The final step is to evaluate the entropy of $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{y}^\star)$, *i.e.*, the entropy of the predictive distribution of the GPs at $\mathbf{x}$ given that $\mathscr{y}^\star$ is the solution of the optimization problem. Unfortunately, this conditional predictive distribution is intractable and has to be approximated. We explain the approximation employed in the next section.

3.1. Approximating the Conditional Predictive Distribution

In the initial formulation of MESMOC+ we consider that the evaluations are noiseless, just as it is done in [17] for *max-value entropy search* (MES). Namely, we approximate $p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x},\mathscr{y}^\star)$ instead of $p(\mathbf{y}|\mathscr{D},\mathbf{x},\mathscr{y}^\star)$, where $\mathbf{f}=\{f_1(\mathbf{x}),\ \ldots,f_K(\mathbf{x})\}$ and $\mathbf{c}=\{c_1(\mathbf{x}),\ \ldots,c_C(\mathbf{x})\}$ are the predicted values for the objectives and constraints at $\mathbf{x}$. At the end of the next section we modify the acquisition function developed to take additive noise into account. The expression of $p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x},\mathscr{y}^\star)$ is obtained using Bayes' rule:

$$p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x},\mathscr{y}^\star)=Z^{-1}p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x})p(\mathscr{y}^\star|\mathbf{f},\mathbf{c}), \tag{10}$$

where Z^{-1} is a normalization constant, $p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x})$ is the predictive probability for the objectives and the constraints at the candidate point $\mathbf{x}$ given $\mathscr{D}$, and $p(\mathscr{y}^\star|\mathbf{f},\mathbf{c})$ is the probability that $\mathscr{y}^\star$ is a valid Pareto front given $\mathbf{f}$ and $\mathbf{c}$. Note that $p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x})$ is simply a product of Gaussians given by the predictive distribution of each GP.

The factor $p(\mathscr{y}^\star|\mathbf{f},\mathbf{c})$ in Eq. (10) removes all configurations of the objectives and constraints values, $(\mathbf{f},\mathbf{c})$, that are incompatible with $\mathscr{y}^\star$ being the Pareto front of the problem. Therefore, $p(\mathscr{y}^\star|\mathbf{f},\mathbf{c})$ must be 0 when $\mathbf{c}$ does not violate the constraints (*i.e*, $\mathbf{c}$ does satisfy $c_j(\mathbf{x})\geqslant 0,\forall j\in\{1,\ldots,C\}$), and $\mathbf{f}$ is not *Pareto dominated* by any $\mathbf{f}^\star\in\mathscr{y}^\star$. Similarly, $p(\mathscr{y}^\star|\mathbf{f},\mathbf{c})$ is 1 if all points $\mathbf{f}^\star$ in the Pareto front $\mathscr{y}^\star$ dominate $\mathbf{f}$, or if $\mathbf{c}$ violates the constraints (*i.e.* at least one constraint is negative at $\mathbf{x}$). This can be expressed, informally, as follows:

$$\begin{aligned} p(\mathscr{y}^\star|\mathbf{f},\mathbf{c}) &\propto \prod_{\mathbf{f}^\star\in\mathscr{y}^\star}\left(1-\prod_{j=0}^{C}\Theta(c_j)\prod_{k=0}^{K}\Theta\left(f_k^\star-f_k\right)\right)\\ &\propto \prod_{\mathbf{f}^\star\in\mathscr{y}^\star}\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right), \end{aligned} \tag{11}$$

where $\Theta(\cdot)$ is the Heaviside step function, $f_k=f_k(\mathbf{x}), c_j=c_j(\mathbf{x}), f_k^\star$ is the k-th value of the vector of values $\mathbf{f}^\star$ of $\mathscr{y}^\star$ and $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)=1-\prod_{j=0}^{C}\Theta(c_j)\prod_{k=0}^{K}\Theta\left(f_k^\star-f_k\right)$. Note that Eq. (11) will only be 1, if $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$ is 1 for all the $\mathbf{f}^\star$ in $\mathscr{y}^\star$. To make $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$ be 1, either $\prod_{j=0}^{C}\Theta(c_j(\mathbf{x}))$ or $\prod_{k=0}^{K}\Theta\left(f_k^\star-f_k(\mathbf{x})\right)$ must be 0. This happens if all the values of $\mathbf{c}$ are greater or equal to 0 or if all the values of $\mathbf{f}^\star$ are lower or equal to those of $\mathbf{f}$, except one which must be strictly minor. These are precisely the conditions that we previously mentioned above Eq. (11).

The computation of the normalization constant and the entropy of Eq. (10) is intractable. Therefore, we need to approximate this distribution. Importantly, we would like the acquisition function to be cheap compared to the cost of evaluating the black-boxes. For this reason, we use Assumed Density Filtering (ADF) for approximate inference [29,22]. ADF simply approximates each non-Gaussian factor in Eq. (10) using a Gaussian distribution. Since the predictive distribution of a GP is Gaussian, the only non-Gaussian factors are the $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$ factors in Eq. (11). Recall that we assume independence among the objectives and constraints. Since the Gaussian distribution is closed under the product operation and the factors $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$ only involve independent Gaussian random variables, the approximation of Eq. (10) is a factorizing Gaussian distribution. In the following section, we describe the specific updates for the parameters of each of these Gaussians, and in Section 3.3, we discuss the final expression of MESMOC+.

Expectation propagation (EP) is another widely used approximate inference method that can be employed to approximate intractable distributions using Gaussian distributions [22]. This method is an improved version of ADF, but is more expensive since it needs to refine each non-Gaussian factor several times and it is not guaranteed to converge. We tried EP in MESMOC+ to approximate the required computations, but the performance obtained by EP was not better than that of ADF. For this reason, we discarded it.

3.2. ADF Updates for MESMOC+

ADF approximates $p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x},\mathscr{y}^\star)$ using a distribution $q(\mathbf{f},\mathbf{c})$ that is Gaussian. Specifically, ADF minimizes, in an iterative way, the Kullback–Leibler divergence between $\hat{p}(\mathbf{f},\mathbf{c})=Z^{-1}\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)q(\mathbf{f},\mathbf{c})$ and $q(\mathbf{f},\mathbf{c})$, with respect to q, where Z is a normalization constant, for each $\mathbf{f}^\star\in\mathscr{y}^\star$. Minimizing the Kullback–Leibler divergence when the approximate distribution q is Gaussian is equivalent to matching moments between the two distributions, $\hat{p}$ and q [22]. Therefore, we are going to adjust the means and covariances of q to have the same mean and covariances as $\hat{p}(\mathbf{f},\mathbf{c})=Z^{-1}\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)q(\mathbf{f},\mathbf{c})$. This process is repeated iteratively for each factor $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$. Note that there is one different factor for each $\mathbf{f}^\star$ in $\mathscr{y}^\star$. It is possible to understand this process as replacing each $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$ using an approximate Gaussian factor [22]. Of course, the processing order of each $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$ affects the approximation. We simply process the factors in a random order.

Now, we describe how to use ADF to obtain the updates for the means and variances of the resulting Gaussian, after incorporating each $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$ factor. Let $q(\mathbf{f},\mathbf{c})=p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x})$. That is, we initialize q to the predictive distribution given by the GP at $\mathbf{x}$. Let $\mathbf{a}=(\mathbf{f},\mathbf{c})$. We can express $\hat{p}(\mathbf{f},\mathbf{c})$ as a product of an arbitrary function $t(\mathbf{a})$ multiplied by a Gaussian distribution $\mathcal{N}(\mathbf{a}|\boldsymbol{\mu},\boldsymbol{\Sigma})$. Namely,

$$Z=\int t(\mathbf{a})\mathcal{N}(\mathbf{a}|\boldsymbol{\mu},\boldsymbol{\Sigma})d\mathbf{a},\qquad \hat{p}(\mathbf{a})=Z^{-1}t(\mathbf{a})\mathcal{N}(\mathbf{a}|\boldsymbol{\mu},\boldsymbol{\Sigma}), \tag{12}$$

where our arbitrary factor $t(\mathbf{a})$ is $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$, and $\mathcal{N}(\mathbf{a}|\boldsymbol{\mu},\boldsymbol{\Sigma})$ is simply $q(\mathbf{f},\mathbf{c})$, where $\boldsymbol{\mu}$ y $\boldsymbol{\Sigma}$ are the vector of means and the covariance matrix of $(\mathbf{f},\mathbf{c})$ under q, respectively. Because we assume independence among the GPs, $\boldsymbol{\Sigma}$ is diagonal.

Then, we can use the following expressions from [22] to refine the mean and variance of de updated q distribution after incorporating $\Omega\left(\mathbf{f}^\star,\mathbf{f},\mathbf{c}\right)$:

$$\mathbb{E}_{\hat{p}(\mathbf{a})}[\mathbf{a}] = \boldsymbol{\mu} + \boldsymbol{\Sigma}\frac{\partial \log(Z)}{\partial \boldsymbol{\mu}},$$
$$\mathbb{E}_{\hat{p}(\mathbf{a})}\left[\mathbf{a}\mathbf{a}^{\mathrm{T}}\right] - \mathbb{E}_{\hat{p}(\mathbf{a})}[\mathbf{a}]\mathbb{E}_{\hat{p}(\mathbf{a})}[\mathbf{a}]^{\mathrm{T}} = \boldsymbol{\Sigma} - \boldsymbol{\Sigma}\left(\frac{\partial \log(Z)}{\partial \boldsymbol{\mu}}\left(\frac{\partial \log(Z)}{\partial \boldsymbol{\mu}}\right)^{\mathrm{T}} - 2\frac{\partial \log(Z)}{\partial \boldsymbol{\Sigma}}\right)\boldsymbol{\Sigma}. \tag{13}$$

Hence, to use ADF, we need to compute Z and the partial derivatives of $\log(Z)$ respect to the means and variances of the objectives and constraints at $\mathbf{x}$. The computation of Z is as follows:

$$\begin{aligned} Z &= \int p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x})\Omega\left(\mathbf{f}^{\star},\mathbf{f},\mathbf{c}\right)d\mathbf{f}d\mathbf{c} \\ &= \int p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x})\left(1 - \prod_{j=0}^{C}\Theta(c_j(\mathbf{x}))\prod_{k=0}^{K}\Theta\left(f_k^{\star} - f_k(\mathbf{x})\right)\right)d\mathbf{f}d\mathbf{c} \\ &= 1 - \prod_{j=0}^{C}\Phi\left(\gamma_j^c\right)\prod_{k=0}^{K}\Phi\left(\gamma_k^f\right), \end{aligned} \tag{14}$$

where $\Phi(\cdot)$ is the cumulative probability distribution of Gaussian, and:

$$\gamma_k^f = \frac{f_k^{\star} - m_k^f(\mathbf{x})}{\left(v_k^f(\mathbf{x})\right)^{1/2}}, \tag{14a}$$

$$\gamma_j^c = \frac{m_j^c(\mathbf{x})}{\left(v_j^c(\mathbf{x})\right)^{1/2}}, \tag{14b}$$

where $m_k^f(\mathbf{x}), v_k^f(\mathbf{x}), m_k^c(\mathbf{x})$ and $v_k^c(\mathbf{x})$ are the mean and variance values of the k-th and j-th predictive distributions of the objectives and constraints at $\mathbf{x}$. At the same time, these are the values of the derivatives $\frac{\partial \log(Z)}{\partial m_k^f}$, $\frac{\partial \log(Z)}{\partial v_k^f}$, $\frac{\partial \log(Z)}{\partial m_j^c}$ and $\frac{\partial \log(Z)}{\partial v_j^c}$:

$$\begin{aligned} \frac{\partial \log(Z)}{\partial m_k^f} &= \frac{(Z-1)}{Z\Phi\left(\gamma_k^f\right)}\left(\frac{-\mathcal{N}\left(\gamma_k^f|0,1\right)}{\left(v_k^f\right)^{1/2}}\right), \\ \frac{\partial \log(Z)}{\partial v_k^f} &= \frac{(Z-1)}{Z\Phi\left(\gamma_k^f\right)}\left(\mathcal{N}\left(\gamma_k^f|0,1\right)\frac{-\gamma_k^f}{2v_k^f}\right), \end{aligned} \tag{15}$$

$$\begin{aligned} \frac{\partial \log(Z)}{\partial m_j^c} &= \frac{(Z-1)}{Z\Phi\left(\gamma_j^c\right)}\left(\frac{\mathcal{N}\left(\gamma_j^c|0,1\right)}{\left(v_j^c\right)^{1/2}}\right), \qquad \frac{\partial \log(Z)}{\partial v_j^c} \\ &= \frac{(Z-1)}{Z\Phi\left(\gamma_j^c\right)}\left(\mathcal{N}\left(\gamma_j^f|0,1\right)\frac{-\gamma_j^c}{2v_j^c}\right), \end{aligned} \tag{16}$$

where $v_k^f = v_k^f(\mathbf{x})$ and $v_j^c = v_j^c(\mathbf{x})$.

Algorithm 1 shows the ADF steps for computing the mean and variance of the approximate conditional predictive distribution. We can see that in each iteration the values of $\tilde{\mathbf{m}}^f, \tilde{\mathbf{v}}^f$, $\tilde{\mathbf{m}}^c$ and $\tilde{\mathbf{v}}^c$ are updated using the values calculated in the derivatives of Eq. (15) and Eq. (16). We can also notice that the order of processing the $\mathbf{f}^{\star}$ points of the Pareto front sample will influence the final result for the means and variances of the approximate conditional predictive distribution. For this reason, we have established this order as random.

Algorithm 1.:

Algorithm 1: ADF Algorithm

Input: $\mathbf{m}^f$, $\mathbf{v}^f$, $\mathbf{m}^c$ and $\mathbf{v}^c$

Initialize: $\tilde{\mathbf{m}}^f = \mathbf{m}^f$, $\tilde{\mathbf{v}}^f = \mathbf{v}^f$, $\tilde{\mathbf{m}}^c = \mathbf{m}^c$ and $\tilde{\mathbf{v}}^c = \mathbf{v}^c$

for each $\mathbf{f}^{\star}$ **in** $\mathscr{Y}^{\star}$ **do**

- $\boldsymbol{\gamma}^f = (\mathbf{f}^{\star} - \mathbf{m}^f)/\sqrt{\mathbf{v}^f}$
- $\boldsymbol{\gamma}^c = \mathbf{m}^c/\sqrt{\mathbf{v}^c}$
- $Z = 1 - \prod_{j=0}^{C}\Phi(\gamma_j^c)\prod_{k=0}^{K}\Phi(\gamma_k^f)$
- $\tilde{\mathbf{m}}^f = \tilde{\mathbf{m}}^f + \tilde{\mathbf{v}}^f\frac{\partial \log(Z)}{\partial \tilde{\mathbf{m}}^f}$
- $\tilde{\mathbf{v}}^f = \tilde{\mathbf{v}}^f - \tilde{\mathbf{v}}^f\left(\frac{\partial \log(Z)}{\partial \tilde{\mathbf{m}}^f}\left(\frac{\partial \log(Z)}{\partial \tilde{\mathbf{m}}^f}\right)^{\mathrm{T}} - 2\frac{\partial \log(Z)}{\partial \tilde{\mathbf{v}}^f}\right)\tilde{\mathbf{v}}^f$
- $\tilde{\mathbf{m}}^c = \tilde{\mathbf{m}}^c + \tilde{\mathbf{v}}^c\frac{\partial \log(Z)}{\partial \tilde{\mathbf{m}}^f}$
- $\tilde{\mathbf{v}}^c = \tilde{\mathbf{v}}^c - \tilde{\mathbf{v}}^c\left(\frac{\partial \log(Z)}{\partial \tilde{\mathbf{m}}^c}\left(\frac{\partial \log(Z)}{\partial \tilde{\mathbf{m}}^c}\right)^{\mathrm{T}} - 2\frac{\partial \log(Z)}{\partial \tilde{\mathbf{v}}^c}\right)\tilde{\mathbf{v}}^c$

return $\tilde{\mathbf{m}}^f$, $\tilde{\mathbf{v}}^f$, $\tilde{\mathbf{m}}^c$ and $\tilde{\mathbf{v}}^c$;

3.3. The MESMOC+ Acquisition Function

After the execution of ADF, the variances of the objectives and the constraints of the predictive distribution at $\mathbf{x}$, conditioned to the Pareto front $\mathscr{Y}^{\star}$, are available. Thus, we obtain the approximation of Eq. (8) by simply combining Eq. (9) with the result from the calculation of the entropy of $p(\mathbf{f},\mathbf{c}|\mathscr{D},\mathbf{x},\mathscr{Y}^{\star})$. Since this distribution is approximated with a Gaussian distribution using ADF, the approximate entropy has a form similar to that of Eq. (9). As a result, the acquisition function can be approximated as the difference between the entropy of two factorizing multi-variate Gaussians:

$$\alpha(\mathbf{x}) \approx \sum_{k=1}^{K} \log\left(v_k^f\right) + \sum_{j=1}^{C} \log\left(v_j^c\right) - \frac{1}{M}\sum_{m=1}^{M}\left[\sum_{k=1}^{K} \log\left(\tilde{v}_k^f\right) + \sum_{j=1}^{C} \log\left(\tilde{v}_j^c\right)\right], \quad (17)$$

where M is the number of Monte Carlo samples of $\mathscr{y}^*$, $v_k^f = v_k^f(\mathbf{x})$, $v_j^c = v_j^c(\mathbf{x})$, $\tilde{v}_k^f = \tilde{v}_k^f\left(\mathbf{x}|\mathscr{y}_{(m)}^*\right)$, $\tilde{v}_j^c = \tilde{v}_j^c\left(\mathbf{x}|\mathscr{y}_{(m)}^*\right)$ are the approximate variances of the conditional distribution before and after conditioning on the sample $\mathscr{y}_{(m)}^*$ of the Pareto front $\mathscr{y}^*$ and $\left\{\mathscr{y}_{(m)}^*\right\}_{m=1}^{M}$ is the set of Monte Carlo samples of $\mathscr{y}^\star$.

Unfortunately, the behavior of MESMOC+ when using Eq. (17) to approximate the acquisition function is not the expected one, because it is highly influenced by a small decrease in the variance of the conditional predictive distribution. This is particularly the case for points that have a very small associated initial variance, *e.g.*, 10^{-5}. The logarithm tends to amplify these minimal differences (*e.g.*, a variance reduction from 10^{-5} to 10^{-6} will result in a log difference that is approximately equal to 2.32) and in consequence, it has a highly exploitative behavior of the BO method which tends to perform evaluations that are very close to points that have already been evaluated. To solve this, we modified MESMOC+'s acquisition function to take into account the absolute reduction in the variance instead:

$$\alpha(\mathbf{x}) \approx \sum_{k=1}^{K}\left(v_k^f\right) + \sum_{j=1}^{C}\left(v_j^c\right) - \frac{1}{M}\sum_{m=1}^{M}\left[\sum_{k=1}^{K}\left(\tilde{v}_k^f\right) + \sum_{j=1}^{C}\left(\tilde{v}_j^c\right)\right]. \quad (18)$$

Lastly, to take into account the noise we just need to add the additive variance noise of each black-box to the respective variance of the predictive distribution before and after conditioning on the Pareto front. So, the final expression of MESMOC+acquisition function is:

$$\alpha(\mathbf{x}) \approx \sum_{k=1}^{K}\left(v_k^f + \left(\sigma_k^f\right)^2\right) + \sum_{j=1}^{C}\left(v_j^c + \left(\sigma_j^c\right)^2\right) - \frac{1}{M}\sum_{m=1}^{M}\left[\sum_{k=1}^{K}\left(\tilde{v}_k^f + \left(\sigma_k^f\right)^2\right) + \sum_{j=1}^{C}\left(\tilde{v}_j^c + \left(\sigma_j^c\right)^2\right)\right], \quad (19)$$

where $\left(\sigma_k^f\right)^2$ and $\left(\sigma_j^c\right)^2$ are the noise variances of each objective and constraint, respectively.

In Fig. 1 we show a flowchart for solving a multi-objectives optimization problem with constraints using BO and the proposed acquisition function MESMOC+. Steps from 3 to 8 are specific to compute our acquisition function MESMOC+. The other steps are common to any BO method for multi-objective constrained problems. The process will start with a random evaluation of the objectives and constraints, performed at step 1. Step 2 tunes all the GP models to the observed data. Step 3 samples from the GPs the objectives and the constraints, which are then optimized in step 4 to obtain a sample of $\mathscr{y}^\star, \mathscr{y}_{(m)}^\star$. Given this sample, the conditional predictive distribution of the GPs is computed using ADF in step 5. Then, in step 6 the difference between the unconditioned predictive variances and the conditioned predictive variances is computed, following Eq. (19). If the number of Monte Carlo samples of $\mathscr{y}^\star$ is reached, the average acquisition function is returned. This acquisition function is then optimized in step 9 to find the next point to evaluate in step 10. If the maximum number of evaluations have been reached in step 11, the GPs predictive means are optimized in step 12 and the obtained solution is returned as the recommended solution for the optimization problem in step 13.

Note that Eq. (19) is the sum of the acquisition of each black box. Namely:

$$\alpha(\mathbf{x}) \approx \sum_{k=1}^{K}\alpha_k^f(\mathbf{x}) + \sum_{j=1}^{C}\alpha_j^c(\mathbf{x}) \quad (20)$$

where

$$\alpha_k^f(\mathbf{x}) = \sum_{k=1}^{K}\left(v_k^f + \left(\sigma_k^f\right)^2\right) - \frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{K}\left(\tilde{v}_k^f + \left(\sigma_k^f\right)^2\right), \quad (21)$$

$$\alpha_j^c(\mathbf{x}) = \sum_{j=1}^{C}\left(v_j^c + \left(\sigma_j^c\right)^2\right) - \frac{1}{M}\sum_{m=1}^{M}\sum_{j=1}^{C}\left(\tilde{v}_j^c + \left(\sigma_j^c\right)^2\right). \quad (22)$$

Therefore, we can apply Eq. (19) directly in a decoupled evaluation setting, where all black-boxes are competing to be evaluated, and each acquisition function is maximized separately. The black-box with the maximum value associated to the acquisition function is chosen for evaluation.

Fig. 2 shows the execution of MESMOC+ in a toy problem with two objectives and one constraint and one-dimensional input x. This figure illustrates steps 2 to 6 of the flowchart displayed in Fig. 1. We consider a single sample of $\mathscr{y}^\star$. The first column displays the current state of the GPs predictive distribution for the objectives and the constraints, after their fit to the observed data carried out in step 2 of the flowchart. We show the predictive mean alongside with a one standard deviation confidence interval. The second column shows a sample of each black-box function and the corresponding Pareto front. This corresponds to steps 3 and 4 of the flowchart. The points of the Pareto front dominate all other points for which the corresponding values of the constraint are positive. In the third column of this figure, we display the predictive distribution of the black-boxes after conditioning to Pareto front sample (shown in the second column) using ADF. This corresponds to step 5 in the flowchart. The fourth column is the resulting acquisition function of MESMOC+, for each black-box. This acquisition function is the absolute difference of the predictive variances before and after conditioning (shown in the first column and third column, respectively). So, the regions where this difference is greatest are where MESMOC+ expects to gain the most information (in terms of variance reduction) about the solution to the problem. This column corresponds to step 6 in the flowchart.

Fig. 3 shows the acquisition function obtained in the problem of Fig. 2, but in a coupled evaluation scenario, where the individual acquisition functions have been added. Therefore, the acquisition in Fig. 3 is simply the sum of the acquisition of all the black-boxes in the fourth column of Fig. 2. We can observe that the maximizer in Fig. 3 is different from the individual maximizers of the acquisition function of each black-box. We can also observe that the maximum in the coupled setting is larger than the individual maximum of the decoupled setting. However, the sum of the individual maximums is larger than the maximum of the coupled setting. Therefore, we expect that a decoupled evaluation setting will be more useful to gain information about the solution to the optimization problem.

3.4. Computational cost of MESMOC+'s acquisition function

The cost of evaluating Eq. (19) is $\mathcal{O}\left(M(K+C)|\widehat{\mathscr{y}}^\star|\right)$, where M is the number of Monte Carlo samples, and K and C are the number of objectives and constraints, respectively. The factor $(K+C)|\mathscr{y}^\star|$ comes from running the ADF algorithm to approximate the variances of predictive distribution conditioned to $\mathscr{y}^\star$. This approximation is run for each candidate point $\mathbf{x}$ at which the acquisition needs to be evaluated, and for each sample of the objectives and constraints. We approximate $\widehat{\mathscr{y}}^\star$ using 50 points, as in [12]. Recall

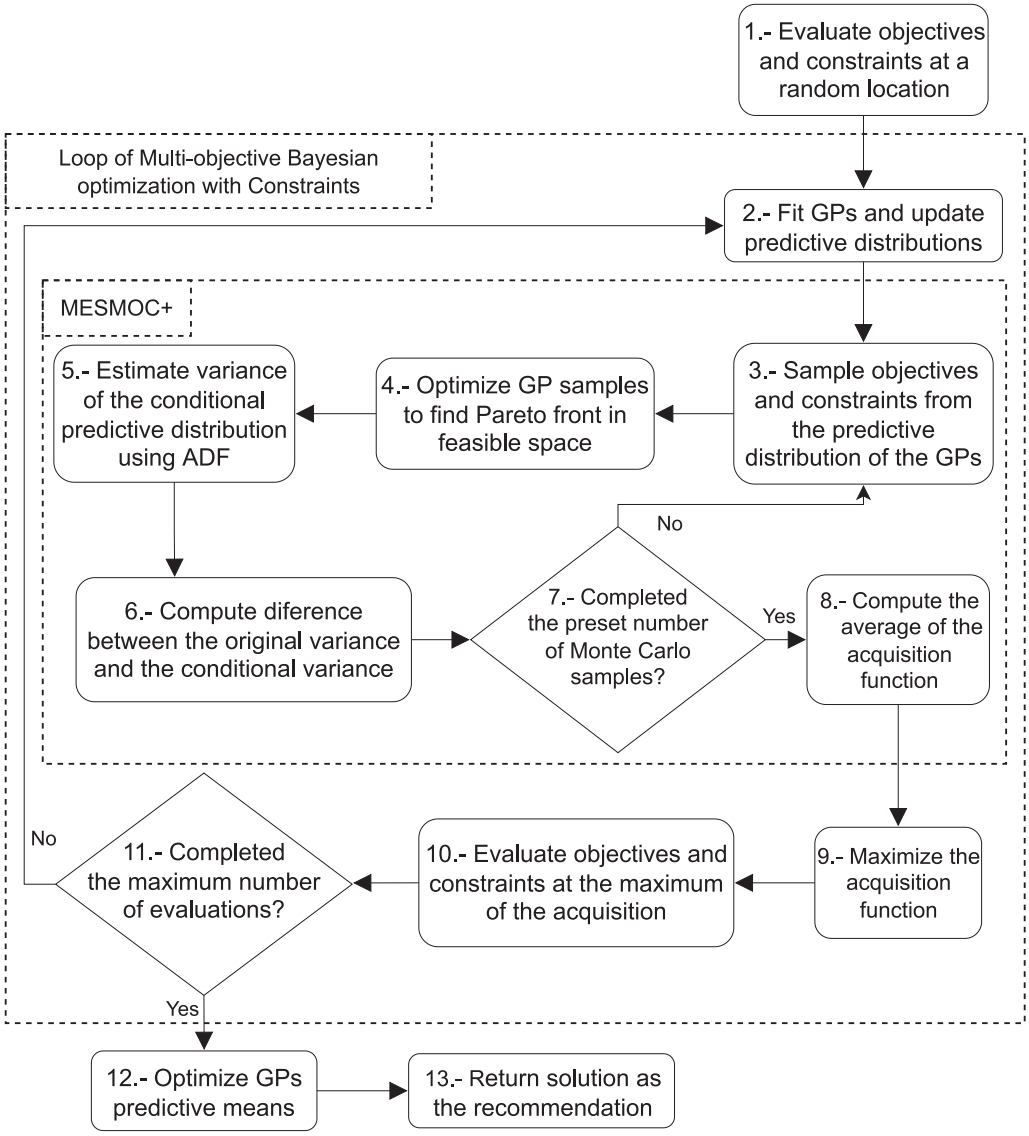

Fig. 1. Flowchart with all the steps required to evaluate the acquisition function MESMOC+ and solving a constrained multi-objective problem using a Bayesian optimization method.

that the maximum size of the Pareto frontier not finite. The acquisition function is optimized using a Quasi-Newton method with the gradient approximated by differences, and to find a good starting value we used a uniform grid of $d \times 1000$ times the dimensionality of the optimization problem. The number of Monte Carlo samples M of $\mathscr{y}^\star$ is 10, as in [12].

4. Experiments

In this section we compare MESMOC+ and its decoupled variant *MESMOC*$+_{\text{dec}}$ with the acquisition functions described in Section 2 (*i.e.*, BMOO, qEHVI, qNEHVI, PESMOC, MESMOC). Moreover, we also include in the comparisons random search (RANDOM) as a simple baseline. The acquisition functions that we compare with are provided in the Bayesian optimization software Spearmint (https://github.com/fernandezdaniel/Spearmint). qEHVI and qNEHVI are implemented in BOTorch [30]. 100 samples are used to approximate the evaluation of the acquisition function and the batch size is set equal to one. We have codified a bridge that allows to call BOTorch acquisition functions from Spearmint. We have also implemented in that software MESMOC+ and MESMOC, closely following the code provided in [19]. We use a Matérn52 with ARD as the kernel of all GPs and to learn their hyper-parameters we use slice sampling with 10 samples, as it is done in [31]. This is also the number of samples considered in MESMOC+, MESMOC and PESMOC for $\mathscr{y}^\star$, $\mathscr{y}^\star$ and $\mathscr{x}^\star$, respectively. To maximize the acquisition function we use L-BFGS using a grid of $d \times 1,000$ points to choose the starting position. The gradients of the acquisition function are approximated by differences. For all experiments we show the results of averaging 100 repetitions alongside the corresponding error bars. The recommendation of each method is obtained by optimizing the means of the GPs at each iteration. To avoid recommending infeasible solutions we follow the approach suggested in [12].

4.1. Quality of the Approximation of the Acquisition Function

We compare in a simple problem the acquisition function of MESMOC+ and MESMOC with the exact acquisition function described in Eq. (8). Since, the problem considered has only two objectives and one constraint, in this setting, quadrature methods

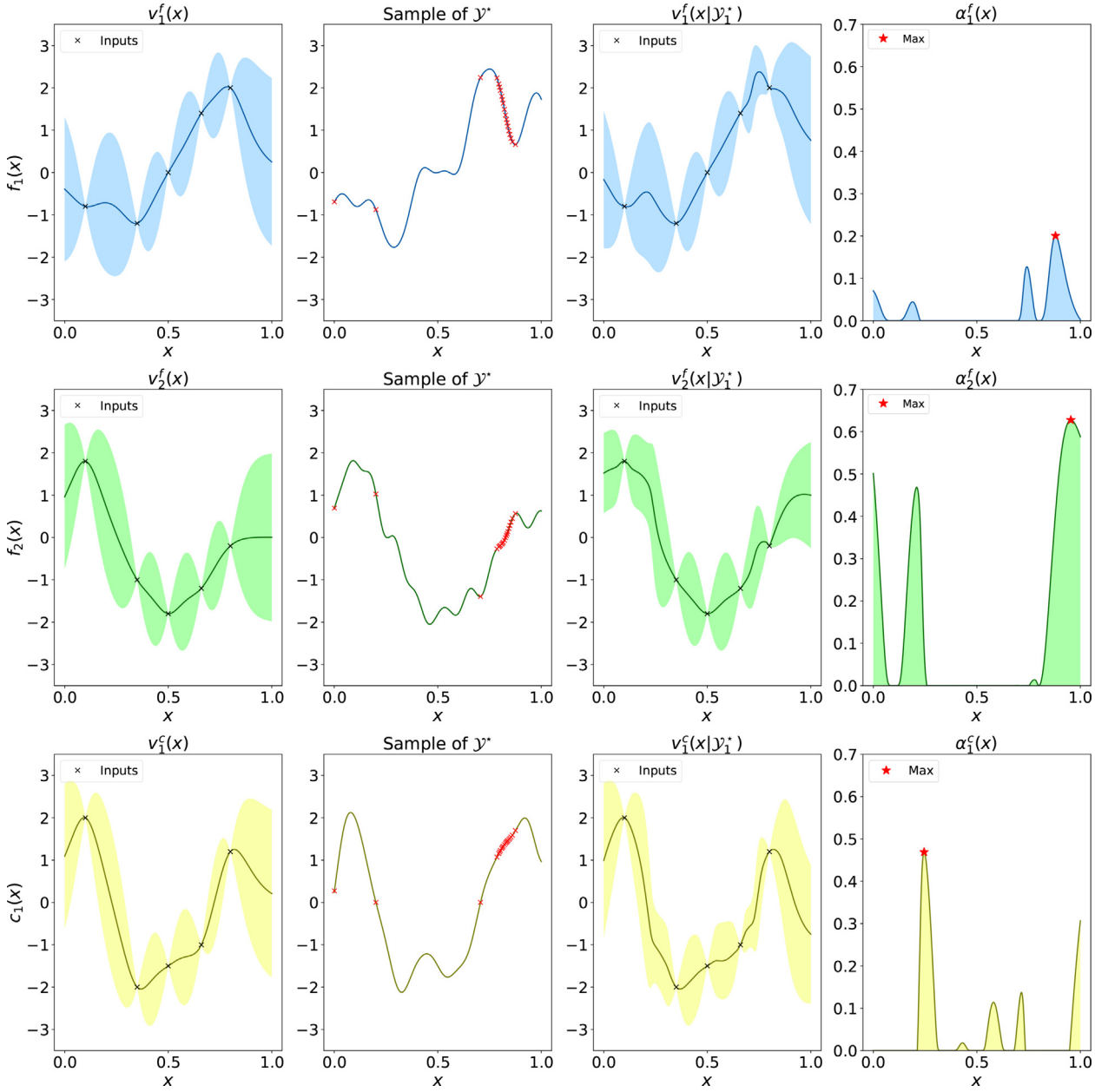

Fig. 2. Different steps needed to compute MESMOC+'s acquisition function in a decoupled setting for two objectives and one constraint. (First column) Predictive distribution for each black-box function conditioned the current inputs, using a GP as probabilistic model. (Second column) Samples of each black-box and corresponding Pareto front $\mathcal{Y}^\star_{(m)}$ in the feasible space displayed using red crosses. (Third column) Predictive distribution of each black-box function conditioned to $\mathcal{Y}^\star_{(m)}$ being the solution to the optimization problem. (Fourth column) Acquisition function MESMOC+'s obtained by the absolute difference in the predictive variances before and after the conditioning.

are feasible to evaluate the entropy of $p(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{Y}^\star)$ (but with a much higher computational cost than MESMOC+ and MESMOC). A quadrature method is expected to provide an approximation that is almost equal to that of the exact acquisition.

The left column of Fig. 4 shows the current state of the predictive distributions for the objectives and constraints and the observed data. The right column shows the acquisition function for MESMOC and MESMOC+ in a coupled evaluation setting. For MESMOC+, we show the results for the proposed method and the method that considers the logarithm of the variance described in Eq. (17). We call this method *MESMOC*$+_{\log}$. Lastly, we also show the results of the quadrature method (Exact) using adaptive quadrature. We can see that the approximation of *MESMOC*$+_{\log}$ is the most accurate, closely followed by MESMOC+. By contrast, the approximation of MESMOC does not look similar to the exact acquisition. Furthermore, MESMOC manually imposes that the acquisition will have a negative value at any point where the mean GP of any of the constraints is negative. This causes it to have a very different value from the exact acquisition in those regions of the input space.

Fig. 5 shows the approximation to the acquisition function by each method in a decoupled scenario. The figure on the top-left of Fig. 5 shows the current observations and predictive distributions for the objectives and constraints. The remaining figures show the acquisition functions of MESMOC, MESMOC+ and *MESMOC*$+_{\log}$, and the results of the quadrature method (Exact), for each objective and constraint. We can observe that MESMOC+ and *MESMOC*$+_{\log}$ are very similar to the exact acquisition, for all the black-box functions. Specifically, where MESMOC+ and *MESMOC*$+_{\log}$ take large values, the exact acquisition takes large

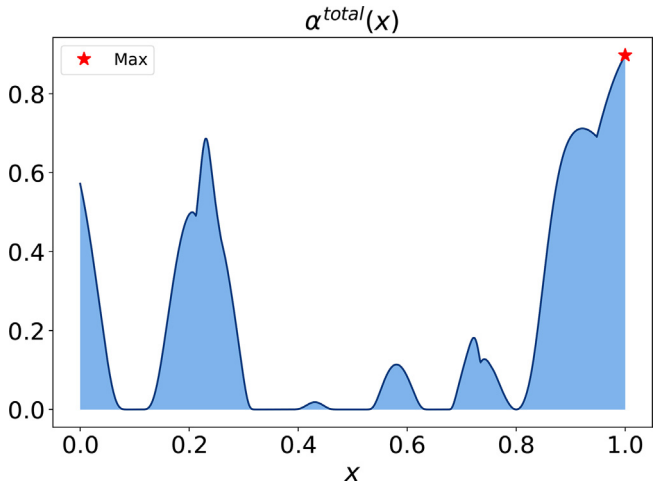

Fig. 3. Acquisition function of MESMOC+ for the coupled setting. In this case $\alpha(\cdot)$ is simply the sum of the acquisition functions of the three black-boxes shown in Fig. 2.

values, and where the exact acquisition decreases, MESMOC+ and *MESMOC*$+_{\log}$ also decrease. By contrast, MESMOC's approximation is dissimilar to the exact acquisition, for all the black-boxes. Furthermore, since MESMOC aims to maximize the constraints, sometimes it gives importance to evaluating the constraints in regions where there may be no expected benefit at all. This behavior can be problematic in the decoupled scenario, where MESMOC can waste useful evaluations simply on trying to maximize a constraint that is already feasible. Fig. 5 shows precisely this behavior.

4.2. Synthetic Experiments

A first batch of experiments evaluate each method in optimization tasks in which the objectives and the constraints are sampled from a GP prior. For each experiment we consider two scenarios: One with noiseless observations and another where the observations are contaminated with standard Gaussian noise with variance 0.1. The first experiment has 4 dimensions, 2 objectives and 2 constraints, and the second has 6 dimensions, 4 objectives and 2 constraints. The performance of each method is measured as the relative difference (in log-scale) of the hyper-volume of the recommendation made and the maximum hyper-volume, with respect to the number of evaluations made. The maximum hyper-volume is obtained using a grid of points and the actual expressions for the black-boxes. Any infeasible recommendation has an associated hyper-volume equal to 0. We report average results over 100 repetitions of the experiments.

The results obtained by each method are shown in Fig. 6. We see that in the 4D experiment, for both scenarios, the best methods are MESMOC+, PESMOC and PESMOC$_{\text{dec}}$. *MESMOC*$+_{\text{dec}}$ also achieves good results when there is no noise. In these experiments MESMOC + is superior to MESMOC, which performs poorly in the noisy settings. MESMOC$_{\text{dec}}$ also performs poorly in general. This is probably as a consequence of the poor approximation of the acquisition function in MESMOC and MESMOC$_{\text{dec}}$. In the 6D experiments we observe similar results, but here *MESMOC*$+_{\text{dec}}$ gets significantly worse results than MESMOC+. This could be related to the removal of the logarithm in the acquisition function of *MESMOC*$+_{\text{dec}}$. Finally, in any of these experiments we observe that in general MESMOC+ and PESMOC give similar results, while MESMOC seems to perform worse. Also we can see that the strategies based on the expected hyper-volume improvement such as BMOO, qEHVI and qNEHVI tend to perform worse than the strategies based on entropy search such as MESMOC+ and PESMOC. The bad performance of qEHVI and qNEHVI is explained because these strategies incorporate the constraints simply by multiplying the unconstrainted acquisition function by the probability of feasibility of the constraints, as estimated by the GPs. This *ad hoc* way of incorporating the constraints makes them evaluate the unfeasible region of the problem many times and prevents them from exploring the feasible space enough to make good evaluations when constraints play an important role in the optimization problem. By contrast, MESMOC+ directly incorporates constraints in the computation of the acquisition function (*e.g.*, when sampling the Pareto front $\mathscr{Y}^\star$). Another limitation of qEHVI and qNEHVI is that they use a Monte Carlo approximation of the acquisition function. As explained in Section 2, after several evaluations, it will be more and more difficult to improve the hyper-volume, which means that the acquisition function of qEHVI and qNEHVI will be flat and equal to zero in almost all the input space. In our experiments we also observed that acquisition functions based on the expected hyper-volume improvement tend to explore too much, often evaluating the black-boxes a the boundaries of the input space.

Fig. 7 shows the average Pareto front obtained by each method on the 4 dimensional synthetic experiments, for the noiseless and the noisy evaluation setting. We do not report results for the 6 dimensional synthetic experiments since there are 4 objectives there, making difficult visualizing the Pareto front. We observe that in general all the methods provide similar Pareto fronts, except for the random search strategy and MESMOC, which provide Pareto fronts that are worse than those of the other methods. In general,

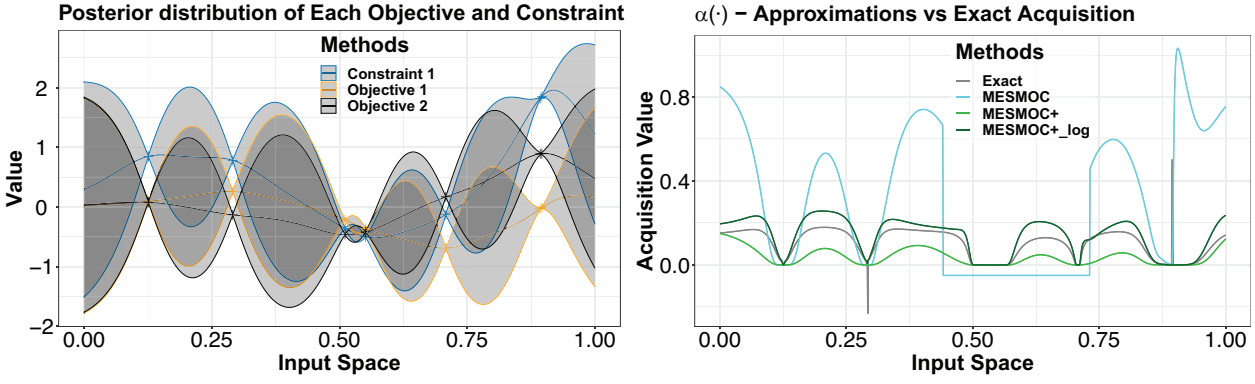

Fig. 4. (left) GP predictive distributions for the objectives and constraints. (right) The corresponding estimated acquisition function of each method, MESMOC+, *MESMOC*$+_{\log}$ and MESMOC, and the exact acquisition (Exact). Best seen in color.

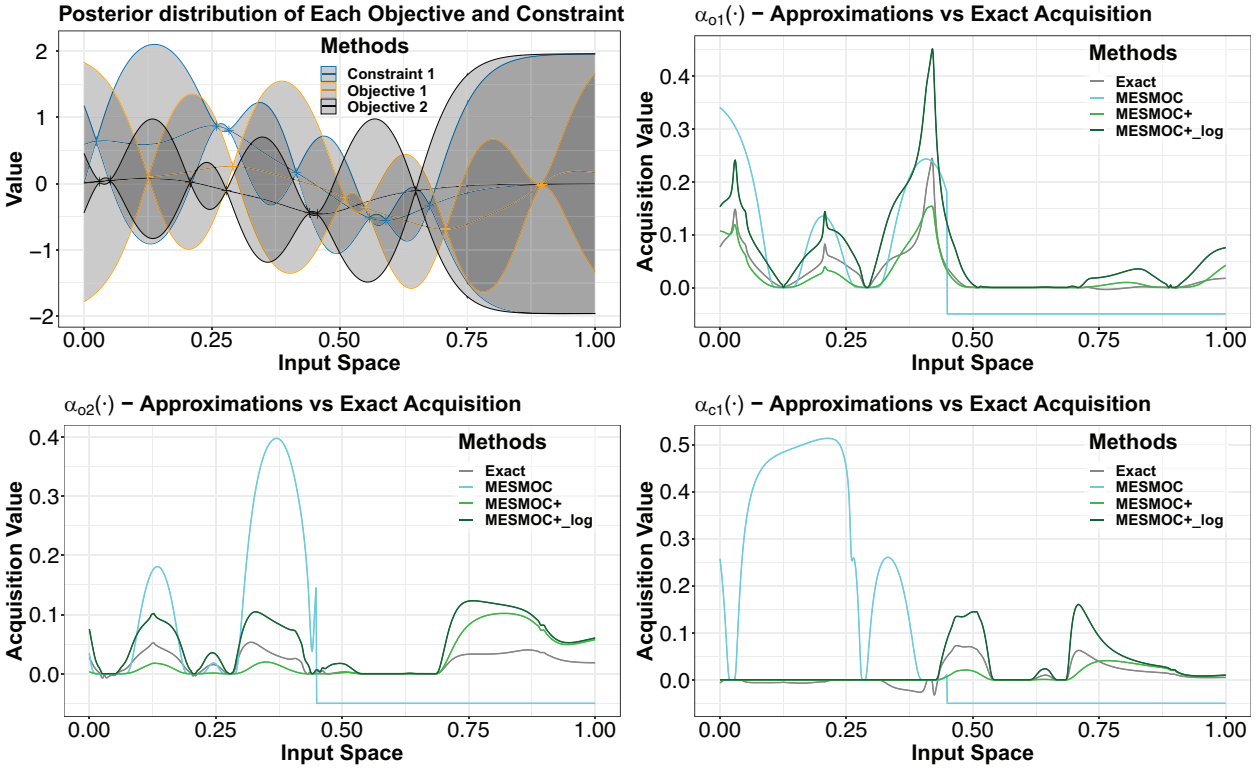

Fig. 5. (top-left) GP predictive distributions for the objectives and constraints. (bottom-left and bottom-right) The corresponding estimated acquisition function of each method, MESMOC+, *MESMOC*$+_{\log}$ and MESMOC, and the exact acquisition (Exact) for each black-box. Best seen in color.

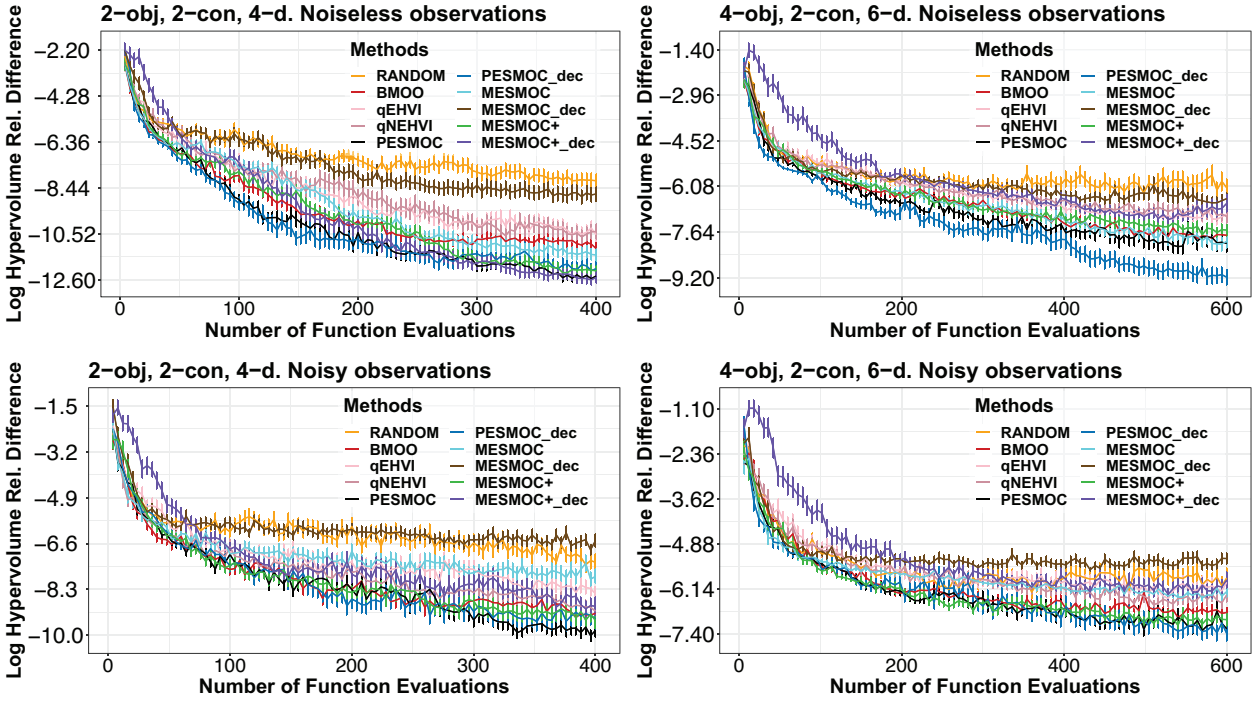

Fig. 6. Average log hyper-volume relative difference between the recommendation of each method and the maximum hyper-volume, with respect to the number of evaluations made. We show the results for the 4-dimensional problem in the left-column and the results for the 6-dimensional problem in the right-column. We consider noiseless (top) and noisy observations (bottom). Best seen in color.

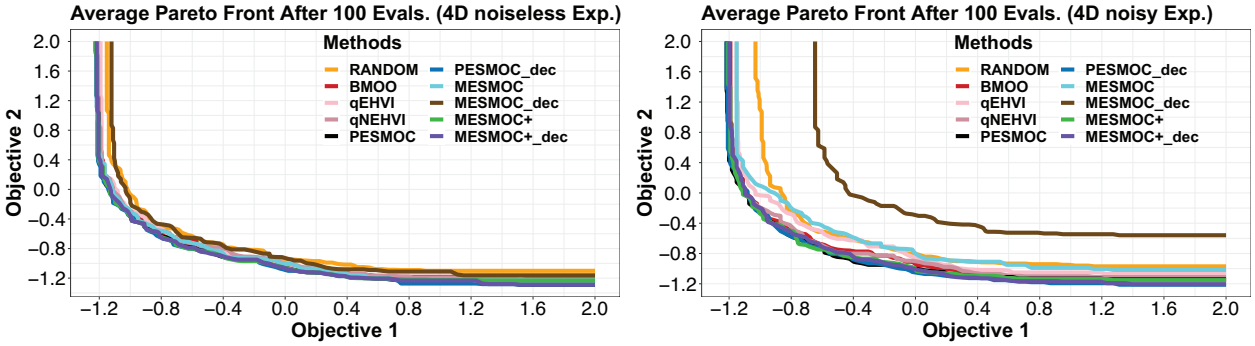

Fig. 7. Average Pareto front of each method when finding an optimal trade-off between the objectives of the 4 dimensional synthetic experiment. Best seen in color.

Table 1
Average execution time of each method per iteration (in sec.) in the synthetic experiments.

	4 Dims	6 Dims
MESMOC+	**13.92 ± 0.200**	44.19 ± 0.455
MESMOC+$_{\text{dec}}$	31.75 ± 1.189	136.75 ± 1.732
MESMOC	15.60 ± 0.123	**27.88 ± 0.242**
MESMOC$_{\text{dec}}$	19.78 ± 0.225	109.83 ± 0.674
PESMOC	26.21 ± 0.276	83.20 ± 0.826
PESMOC$_{\text{dec}}$	45.74 ± 0.543	148.30 ± 0.926
qEHVI	48.20 ± 0.126	72.21 ± 0.350
qNEHVI	50.44 ± 0.148	232.74 ± 5.598
BMOO	22.41 ± 0.219	61.58 ± 0.904

MESMOC+ provides Pareto fronts that are similar to those of the state-of-the-art in these problems.

Table 1 displays the average execution time in seconds per iteration of each method in the synthetic experiment with 4 dimensions (first column) and 6 dimensions (second column), in the noiseless setting. The results for the noisy setting are almost identical and hence omitted. The fastest method is high-lighted in boldface. The second fasted method is underlined. We observe that the times of MESMOC+ and *MESMOC*+$_{\text{dec}}$ are significantly lower than those of PESMOC and PESMOC$_{\text{dec}}$, respectively. This is because MESMOC+'s approximation is cheaper to compute. In particular, it reduces the entropy of the solution of the problem in the function space and uses ADF to approximate the conditional predictive distribution, instead of EP, like PESMOC. On the other hand, the total execution time of MESMOC is similar or just a little lower than the one of MESMOC+. However, although the runtime of MESMOC is smaller that MESMOC+, its performance is much worse. In the decoupled evaluation setting the computational benefits of MESMOC+ compared to PESMOC are smaller than in the coupled evaluation setting. This is probably due to the extra over-head of having to optimize one acquisition function per black-box. The execution times of qEHVI and qNEHVI are also very high. The reason is that these methods have a cost that is super-polynomial in the number of objectives due to the number of box decompositions needed to estimate the hyper-volume [9,10]. Furthermore, qNEHVI is more expensive than qEHVI since it requires sampling from the Gaussian processes not only at the candidate location at which to evaluate the acquisition function, but also at the already evaluated points.

According to [17] max-value entropy search (MES) is more robust than predictive entropy search (PES) with respect to the number of samples of the solution of the optimization problem. A similar behavior could be observed in the constrained multi-objective case. To check this, we also evaluate the results of PESMOC and MESMOC+ with respect to the number of samples M of the Pareto set and the Pareto front considered, respectively. Namely, in the case of MESMOC+, the number of Monte Carlo samples of $\mathscr{y}^\star$ considered to approximate the expectation in Eq. (8), and, in the case of PESMOC, the number of Monte Carlo samples of $\mathscr{X}^\star$ considered to approximate the expectation in Eq. (3). We have evaluated the same the number of samples as in [17]. That is, M equal to $1, 10$ and 100 samples. Therefore, we compare are *MESMOC*+$_M$ and PESMOC$_M$, where $M \in \{1, 10, 100\}$. We have not included MESMOC in the comparison because it has shown bad performance results in previous experiments. In this experiment we do not use slice sampling and simply adjust the GPs by maximizing the marginal likelihood. To avoid over-fitting we use 20 randomly chosen initial evaluations of each black-box in each method.

Fig. 8 shows the results obtained in the 6D experiment considered before. We observe a higher robustness of MESMOC+ with

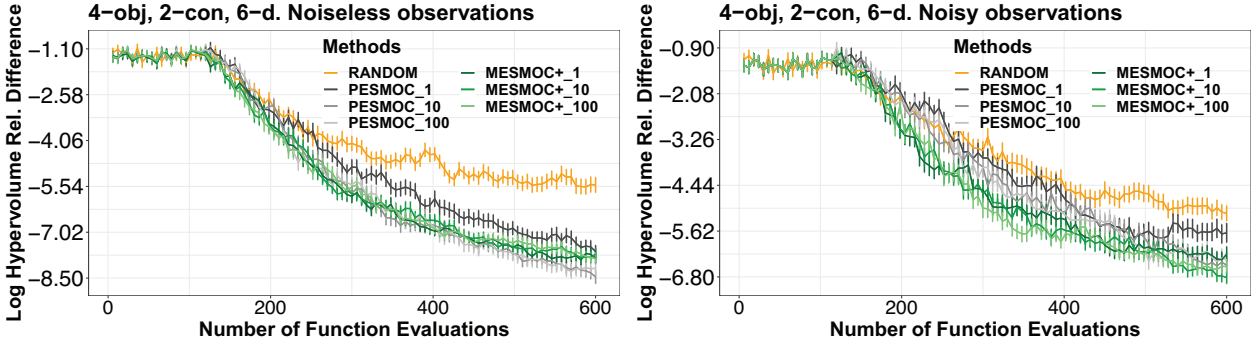

Fig. 8. Average log hyper-volume relative difference between the recommendation of each method and the maximum hyper-volume, with respect to the number of evaluations made, in the 6-dimensional problem. We consider noiseless (left) and noisy observations (right). Best seen in color.

Table 2
Summary of BNH, SRN, TNK and OSY problems used in the benchmark experiments.

Benchmark experiments Problem name	Input space	Objectives $f_k(\mathbf{x})$ and constraints $c_j(\mathbf{x})$
BNH	$x_1 \in [0,\ 5]$	$f_1(\mathbf{x}) = 4x_1^2 + 4x_2^2$
	$x_2 \in [0,\ 3]$	$f_2(\mathbf{x}) = (x_1 - 5)^2 + (x_2 - 5)^2$
		$c_1(\mathbf{x}) \equiv (x_1 - 5)^2 + x_2^2 \leqslant 25$
		$c_2(\mathbf{x}) \equiv (x_1 - 8)^2 + (x_2 + 3)^2 \geqslant 7.7$
SRN	$x_1 \in [-20,\ 20]$	$f_1(\mathbf{x}) = 2 + (x_1 - 2)^2 + (x_2 - 2)^2$
	$x_2 \in [-20,\ 20]$	$f_2(\mathbf{x}) = 9x_1 - (x_2 - 1)^2$
		$c_1(\mathbf{x}) \equiv x_1^2 + x_2^2 \leqslant 225$
		$c_2(\mathbf{x}) \equiv x_1 - 3x_2 + 10 \leqslant 0$
TNK	$x_1 \in [0,\ \pi]$	$f_1(\mathbf{x}) = x_1$
	$x_2 \in [0,\ \pi]$	$f_2(\mathbf{x}) = x_2$
		$c_1(\mathbf{x}) \equiv x_1^2 + x_2^2 - 1 - 0.1\cos\left(\arctan\frac{x_1}{x_2}\right) \geqslant 0$
		$c_2(\mathbf{x}) \equiv (x_1 - 0.5)^2 + (x_2 - 2)^2 \leqslant 0.5$
OSY	$x_1 \in [0,\ 10]$	$f_1(\mathbf{x}) = -\left[25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 + (x_4 - 4)^2 + (x_5 - 1)^2\right]$
	$x_2 \in [0,\ 10]$	
	$x_3 \in [1,\ 5]$	$f_2(\mathbf{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$
	$x_4 \in [0,\ 6]$	$c_1(\mathbf{x}) \equiv x_1 + x_2 - 2 \geqslant 0$
	$x_5 \in [1,\ 5]$	$c_2(\mathbf{x}) \equiv 6 - x_1 - x_2 \geqslant 0$
	$x_6 \in [0,\ 10]$	$c_3(\mathbf{x}) \equiv 2 - x_2 + x_1 \geqslant 0$
		$c_4(\mathbf{x}) \equiv 2 - x_1 + 3x_2 \geqslant 0$
		$c_5(\mathbf{x}) \equiv 4 - (x_3 - 3)^2 - x_4 \geqslant 0$
		$c_6(\mathbf{x}) \equiv (x_5 - 3)^2 + x_6 - 4 \geqslant 0$

Table 3
Summary of CONSTR, Two-bar truss and Welded beam problems used in the benchmark experiments.

Benchmark experiments Problem name	Input space	Objectives $f_k(\mathbf{x})$ and constraints $c_j(\mathbf{x})$
CONSTR	$x_1 \in [0.1,\ 10]$	$f_1(\mathbf{x}) = x_1$
	$x_2 \in [0,\ 5]$	$f_2(\mathbf{x}) = \frac{(1+x_2)}{x_1}$
		$c_1(\mathbf{x}) \equiv x_2 + 9x_1 \geqslant 6$
		$c_2(\mathbf{x}) \equiv -x_2 + 9x_1 \geqslant 1$
Two-bar truss	$x_1 \in [0,\ 0.01]$	$f_1(\mathbf{x}) = x_1\sqrt{16 + x_3^2} + x_2\sqrt{1 + x_3^2}$
	$x_2 \in [0,\ 0.01]$	$f_2(\mathbf{x}) = \max\left(\frac{20\sqrt{16+x_3^2}}{x_1 x_3}, \frac{80\sqrt{1+x_3^2}}{x_2 x_3}\right)$
	$x_3 \in [1,\ 3]$	$c_1(\mathbf{x}) \equiv \max\left(\frac{20\sqrt{16+x_3^2}}{x_1 x_3}, \frac{80\sqrt{1+x_3^2}}{x_2 x_3}\right) \leqslant 10^5$
Welded beam	$h \in [0.125,\ 5]$	$f_1(\mathbf{x}) = 1.10471h^2 l + 0.04811tb(14 + l)$
	$b \in [0.125,\ 5]$	$f_2(\mathbf{x}) = \frac{2.1952}{t^3 b}$
	$l \in [0.1,\ 10]$	$c_1(\mathbf{x}) \equiv 13600 - \tau(\mathbf{x}) \geqslant 0$
	$t \in [0.1,\ 10]$	$c_2(\mathbf{x}) \equiv 30000 - \frac{504000}{t^2 b} \geqslant 0$
		$c_3(\mathbf{x}) \equiv b - h \geqslant 0$
		$c_4(\mathbf{x}) \equiv 64746.022\left(1 - 0.0282346tb^3\right) - 6000 \geqslant 0$
		$\tau(\mathbf{x}) = \sqrt{\gamma(\mathbf{x})^2 + \epsilon(\mathbf{x})^2 + \frac{l\gamma(\mathbf{x})\epsilon(\mathbf{x})}{\sqrt{0.25\left(l^2 + (h+t)^2\right)}}}$
		$\gamma(\mathbf{x}) = \frac{6000}{\sqrt{2}h}$
		$\epsilon(\mathbf{x}) = \frac{6000(14 + 0.5l)\sqrt{0.25\left(l^2 + (h+t)^2\right)}}{2\sqrt{2}hl\left(\frac{l^2}{12 + 0.25(h+t)^2} + \right)}$

respect to M. Specifically, $MESMOC{+}_1$ is always better than PESMOC_1, both in the noiseless and the noisy setting. We can also notice that both methods improve as the number of samples increases. However, the improvements in the case of MESMOC+ are smaller. This method is more invariant with respect to M. For values of $M = 10$ and $M = 100$ the differences between the methods decrease and they become very similar one to another. These results confirm that MESMOC+ gives better results than PESMOC + when M is small and that MESMOC+ is more robust with respect to the number of samples, M.

4.3. Benchmark Experiments

In the previous section we optimized black-box functions generated by sampling from a GP prior. The black-box functions, however, need not be sampled from a GP, and model bias may have an impact on the results. To evaluate all the methods in such a more general setting, we considered a set of optimization problems described in [32,33]. The analytical form of the black-boxes of the different problems considered is displayed in Tables 2 and 3. We test and compare our proposed method, MESMOC+, with the

rest of the methods from the literature, on these benchmark problems. As in the experiments the previous section, we consider two scenarios, one where the observations are noiseless and the other with observations contaminated with additive Gaussian noise. We set the variance of the additive noise to 1% of the range of potential values of the corresponding black-box function. This range of values is found by evaluating each black-box function on a grid. As in Section 4.2, the performance of each method is measured as the relative difference (in log-scale) between the hyper-volume of the recommendation and the maximum hyper-volume. We report average results across 100 repetitions of the experiments.

Fig. 9 and 10 show the results of each method with the corresponding error bars, as a function of the number of evaluations performed. In these experiments we only consider feasible recommendations. If the solution recommended of one method does not fulfill all constraints, that result is ignored. We also recorded the percentage of infeasible solutions suggested by each method. Nevertheless, in practice, we have observed that all the methods compared tend to suggest a similar amount of infeasible solutions, with

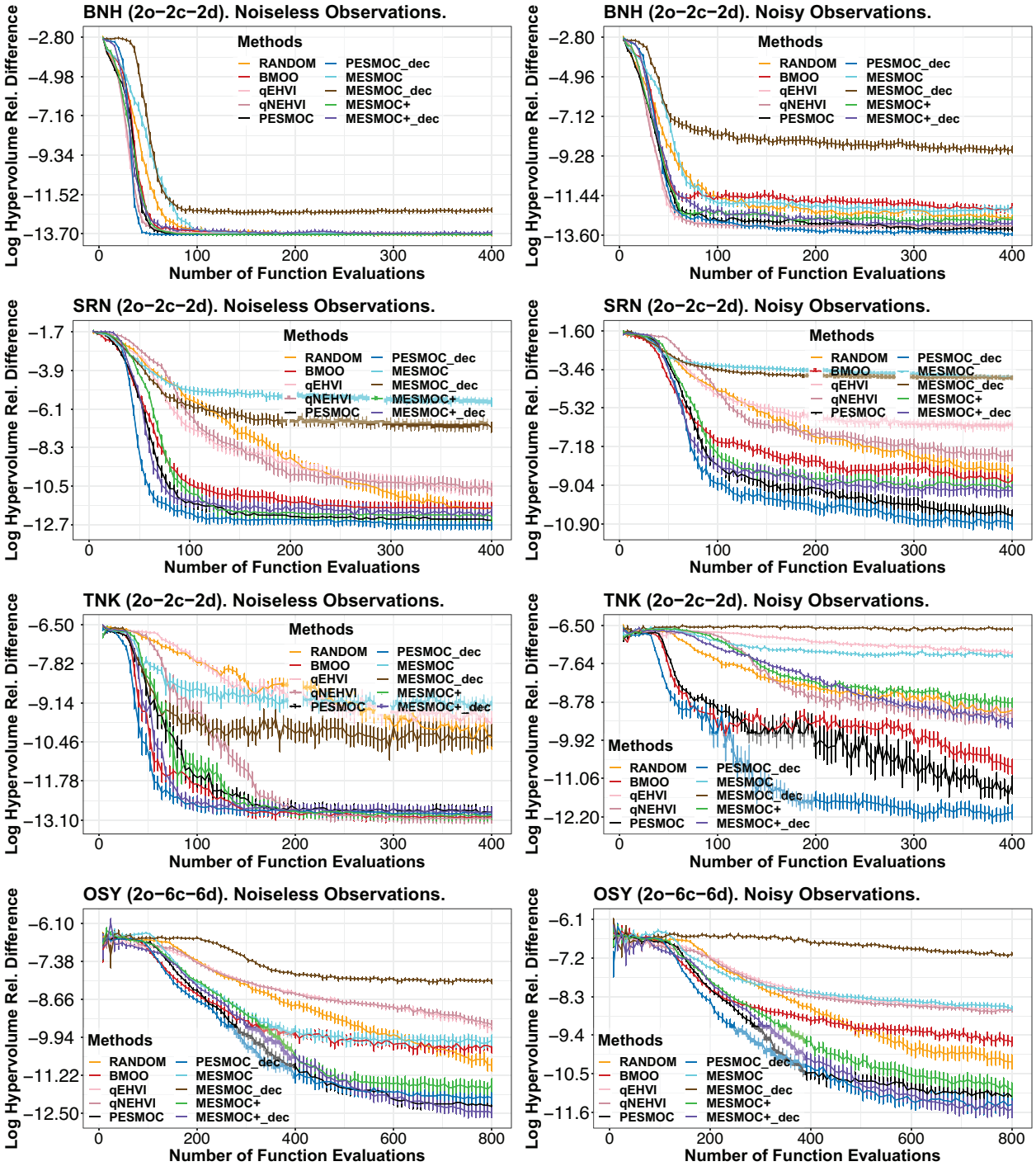

Fig. 9. Average log hyper-volume relative difference between the recommendation of each method at each iteration and the maximum hyper-volume in four benchmark problems in noiseless (left-column) and noisy (right-column) scenarios. Best seen in color.

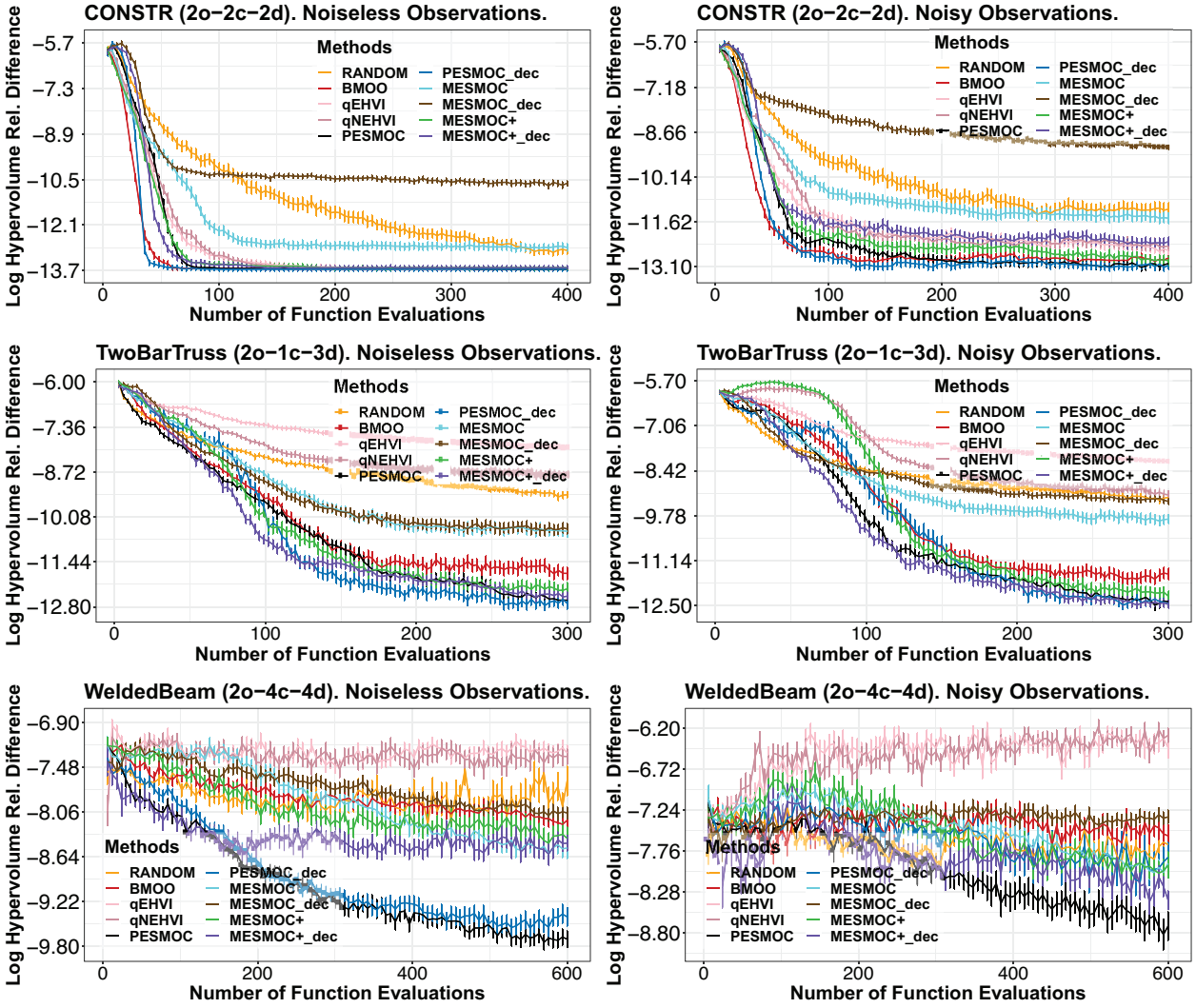

Fig. 10. Average log hyper-volume relative difference between the recommendation of each method at each iteration and the maximum hyper-volume in three benchmark problems in noiseless (left-column) and noisy (right-column) scenarios. Best seen in color.

the exception of RANDOM, which always recommends more invalid solutions as a consequence of not using the probabilistic models to guide the search.

The results displayed in Fig. 9 show that MESMOC+, *MESMOC*+$_{\text{dec}}$, PESMOC and PESMOC$_{\text{dec}}$ obtain the best results, in general. Except in the SRN, TNK and WeldedBeam problems with noise, where MESMOC+ and *MESMOC*+$_{\text{dec}}$ perform not as good as PESMOC and PESMOC$_{\text{dec}}$. We believe that this is probably because the Pareto front is not as informative as the Pareto set about the solution of the optimization problem when: (i) the constraints are highly active in solving the problem, (ii) the observations are noisy and (iii) the proportion of black-boxes is higher than the input dimensions. We also observed that MESMOC and MESMOC$_{\text{dec}}$ perform very poorly and tend to recommend worse solutions than RANDOM, especially MESMOC$_{\text{dec}}$, which is the worst method. Probably this is because of the poor approximation of the conditional predictive distribution, and because of the restriction that limits the evaluations to the input space where the mean of the GPs is positive for the constraints. We can also see that *MESMOC*+$_{\text{dec}}$ usually obtains slightly better results than MESMOC + or finds better results with a smaller number of evaluations. Therefore, using a decoupled acquisition function is advantageous. qEHVI and qNEHVI also perform poorly and often worse than BMOO in some settings such as in the problems OSY, SRN and TNK, in the noisy setting. We believe this is a consequence of their Monte Carlo approximation of the acquisition and the difficulty for finding initial feasible solutions when the feasible space is very small and the constraints are highly active.

Fig. 10 shows the remaining results of the benchmark experiments. We can observe again that MESMOC+, *MESMOC*+$_{\text{dec}}$, PESMOC and PESMOC$_{\text{dec}}$ obtain the best results in the CONSTR and TwoBarTruss problems. In WeldedBeam (noiseless), PESMOC and PESMOC$_{\text{dec}}$ perform the best, but in WeldedBeam (noisy), PESMOC performs better, followed by *MESMOC*+$_{\text{dec}}$. Again, we believe that the poor performance of MESMOC+ and *MESMOC*+$_{\text{dec}}$ in WeldedBeam is because the constraints are highly active in the solution and there is a higher number of black-boxes so that the using Pareto front to guide the search does not give as good results as using the Pareto set. Again, MESMOC and MESMOC$_{\text{dec}}$ perform worse or equal to RANDOM in these problems. qEHVI and qNEHVI also perform poorly in WeldedBeam and TwoBarTruss.

4.4. Finding an Optimal Ensemble

The first experiment with real data consist in tuning the hyper-parameters of an ensemble of classification trees on the German dataset from the UCI repository [34]. This dataset has 1,000 instances, 20 attributes and 2 classes. The hyper-parameters of

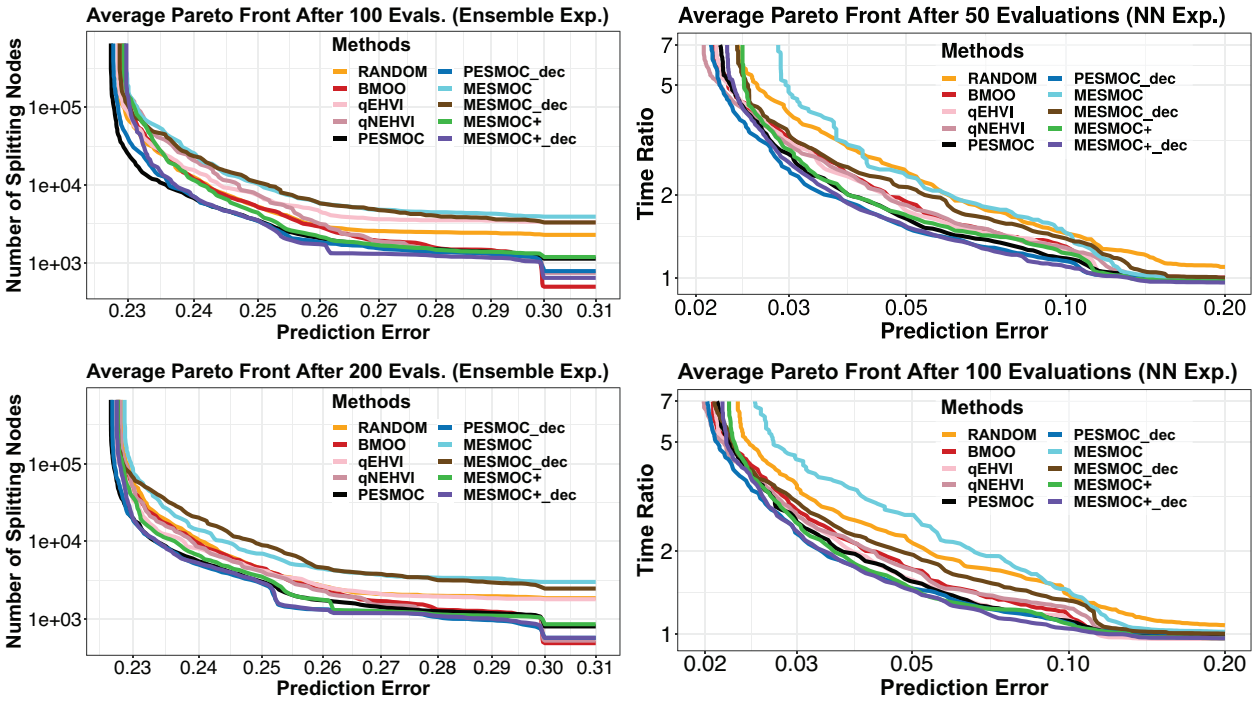

Fig. 11. Average Pareto front of each method when finding an optimal ensemble (left-column) and when finding an optimal neural network (right-column). Best seen in color.

Table 4
Average hyper-volume of each method after 100 and 200 evaluations in the problem of finding an optimal ensemble. Similar results after 50 and 100 evaluations in the problem of finding an optimal neural network. The best result is shown in bold and the second best is underlined.

	Ensemble		Neural Network	
Method	**100 Evals.**	**200 Evals.**	**50 Evals.**	**100 Evals.**
MESMOC+	0.292 ± 0.001	0.322 ± 0.001	47.85 ± 1.186	50.35 ± 0.407
MESMOC+$_{\text{dec}}$	$\mathbf{0.318 \pm 0.001}$	$\underline{0.339} \pm \underline{0.001}$	$\underline{48.71} \pm \underline{0.729}$	$\underline{50.75} \pm \underline{0.489}$
MESMOC	0.220 ± 0.002	0.244 ± 0.002	45.25 ± 3.610	46.27 ± 4.609
MESMOC$_{\text{dec}}$	0.215 ± 0.004	0.234 ± 0.004	45.70 ± 6.343	49.66 ± 1.664
PESMOC	0.310 ± 0.001	0.327 ± 0.001	48.58 ± 0.744	50.38 ± 0.546
PESMOC$_{\text{dec}}$	$\underline{0.312} \pm \underline{0.002}$	$\mathbf{0.340 \pm 0.001}$	$\mathbf{48.95 \pm 0.554}$	$\mathbf{50.83 \pm 0.399}$
qEHVI	0.241 ± 0.003	0.287 ± 0.002	47.69 ± 2.496	50.36 ± 0.801
qNEHVI	0.278 ± 0.002	0.314 ± 0.001	48.32 ± 0.665	50.37 ± 0.371
BMOO	0.294 ± 0.001	0.310 ± 0.001	47.46 ± 2.606	50.06 ± 0.835
RANDOM	0.264 ± 0.001	0.280 ± 0.001	46.23 ± 1.323	48.46 ± 0.932

the ensemble are: the number of trees, the number of attributes to consider to split a node, the minimum number of samples to split a node, the probability of switching the class of each instance [35] and the fraction of training samples on which each tree is trained. We chose two objectives: to minimize the classification error (estimated by a 10-fold-cv method) and to minimize the number of nodes of the trees in the ensemble. These are conflictive objectives since most probably minimizing the prediction error will require larger ensembles. We also choose a constraint: the ensemble has to speed-up its average classification time by at least 25% when using a dynamic pruning technique [36]. We report average results across 100 repetitions of the experiment.

This problem is suitable for decoupled evaluations because the objectives and the constraint can be evaluated separately. The first objective, *i.e.*, the total number of nodes in the ensemble, requires building the ensemble without leaving any data aside for validation. By contrast, the second objective, which involves estimating the generalization performance of the ensemble, requires to train several times the ensemble on different partitions of the data. Similarly, evaluating the constraint involves building a lookup table, which is expensive to build and is different for each ensemble size.

Fig. 11 (left) shows the average hyper-volume Pareto fronts obtained by each method after 100 and 200 evaluations, respectively. Since we are minimizing, the larger the area above the average Pareto front of a method, the better the performance of the method. Fig. 11 (left) shows that *MESMOC*+$_{\text{dec}}$ and PESMOC$_{\text{dec}}$ obtain the best results. Specifically, *MESMOC*+$_{\text{dec}}$ obtains better ensembles of small sizes and PESMOC$_{\text{dec}}$ obtains better ensembles of larger sizes. After 200 evaluations, the average Pareto front obtained by MESMOC+ is similar to that of PESMOC. MESMOC+ finds better ensembles with an error above 25%. By contrast, PESMOC finds better ensembles with a small error. We can also see that MESMOC and MESMOC$_{\text{dec}}$ perform quite poorly. We believe this is a consequence of the difficulty of finding feasible solutions. In that case, these methods will significantly constrain the optimization of the acquisition function, as described in Section 2.

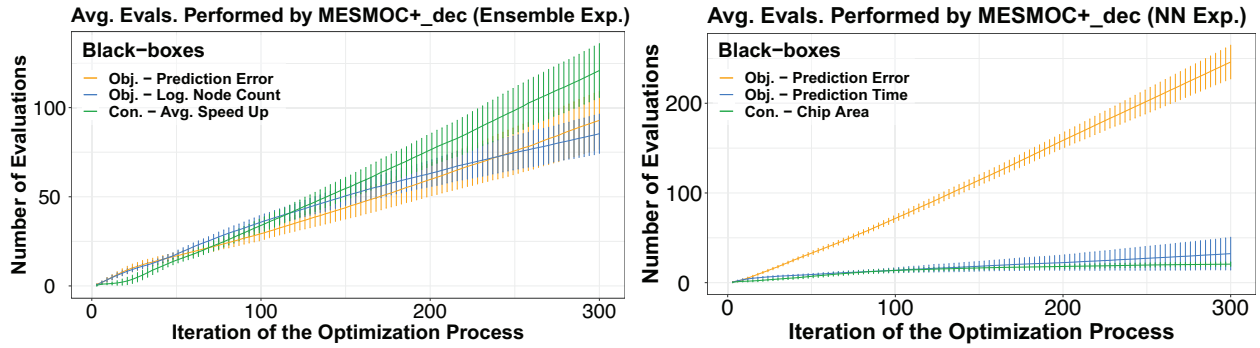

Fig. 12. Number of evaluations performed by *MESMOC*+$_{\text{dec}}$ for each black-box in the problem of finding an optimal ensemble (left) and the problem of finding an optimal neural network (right). Best seen in color.

Table 5
Parameters used for building neural networks. We show the min, max, and step used.

Parameter	Min	Max	Step
Hidden layers	1	3	1
Neurons per layer	5	300	1
Learning rate	e^{-200}	1	ϵ
Dropout	0	0.9	ϵ
ℓ_1 penalty	e^{-200}	1	ϵ
ℓ_2 penalty	e^{-200}	1	ϵ
Memory partition	1	32	2^x
Loop unrolling	1	32	2^x

qEHVI and qNEHVI find worse ensembles than those obtained by MESMOC+ in the decoupled setting after 200 evaluations. Table 4 shows the average hyper-volume of the Pareto front found by each method. We observe that the largest hyper-volume is obtained by *MESMOC*+$_{\text{dec}}$ and PESMOC$_{\text{dec}}$ after 100 and 200 evaluations, respectively, with very similar results. qEHVI and qNEHVI perform in general worse or similar to BMOO, which gives worse hyper-volume results than the proposed method MESMOC+ after 200 evaluations.

Fig. 12 (left) shows the number of evaluations of each black-box performed by *MESMOC*+$_{\text{dec}}$. We observe that such a method evaluates approximately the same number of times each black-box. Therefore, in this case the advantage of the decoupled setting comes from the possibility of choosing different input locations at which to evaluate each black-box, at each iteration. In particular, the coupled version of MESMOC+ has to evaluate all the black-boxes at the same candidate point.

4.5. Finding an Optimal Neural Network

The second experiment with real data consist in tuning the hyper-parameters of a deep neural network on the MNIST dataset [37]. The MNIST dataset contains 60,000 images of 28×28 pixels of hand-written digits. We built the network using Keras. For training the networks we use ADAM with the default parameters [38]. We have divided the dataset into 50,000 instances for training and 10,000 for validation. The hyper-parameters to adjust are shown in Table 5. They are: the number of hidden layers, the number of neurons in each layer, the learning rate, the dropout probability [39], the level of ℓ_1 and ℓ_2 regularization and two parameters related to the codification of the neural network in a chip: the memory partition and the loop unrolling factor. See [12] for more details. We report average results across 100 repetitions of the experiment.

The goal is to minimize the validation error and the prediction time of the network. These are conflictive objectives since most probably minimizing the prediction error will require bigger neural networks with a larger number of hidden units and layers. The constraint chosen invalidates all networks that when codified into a chip result in an area greater than one square millimeter. The calculation of the area needed by each network is obtained using the hardware simulator Aladdin [40]. Again, these objectives and constraints can be evaluated independently. The prediction time is measured as the ratio with respect to the prediction time of the fastest network (*i.e.*, the smallest network). Random weights are used in this case, since we do not need to train the network. The prediction error involves training the network. The constraint, on the other hand, involves running the hardware simulator Aladdin.

We show the average Pareto front obtained by each method, after 50 and 100 evaluations, in the right column of Fig. 11. We observe that PESMOC$_{\text{dec}}$ obtains the Pareto front with the highest hyper-volume, followed by *MESMOC*+$_{\text{dec}}$ and PESMOC. We can also see that after 100 evaluations there is little difference between *MESMOC*+$_{\text{dec}}$, PESMOC$_{\text{dec}}$ and PESMOC. Again, the decoupled variants of MESMOC+ and PESMOC obtain better results than the coupled ones. We can also see that the performance of MESMOC and MESMOC$_{\text{dec}}$ is worse than or similar to that of RANDOM. BMOO and qEHVI give worse results, in general, than the ones of the strategies based on entropy search that can make use of decoupled evaluations. qNEHVI is able to find the best performing neural networks after 50 evaluations. However, it fails to find fast neural networks of moderate error, as shown in Fig. 11. After 100 evaluations the differences with respect to the most accurate network found by qNEHVI become smaller. Table 4 displays the average hyper-volume of the Pareto front of each method after 50 and 100 evaluations. PESMOC$_{\text{dec}}$ obtains the highest hyper-volume closely followed by *MESMOC*+$_{\text{dec}}$.

Finally, the number of evaluations performed by $MESMOC+_{\text{dec}}$ are displayed in the Fig. 12. We observe that most of the evaluations have been carried out on the black-box corresponding to the prediction error. It is expected that this black-box is more difficult to optimize and in consequence the proposed approach, $MESMOC+_{\text{dec}}$, focuses more on it. This agrees with previous results from the literature [12].

5. Conclusions

We have developed MESMOC+, a method for multi-objective Bayesian optimization with constraints. MESMOC+ selects the next point to evaluate as the one that is expected to reduce the most the entropy of the solution of the optimization problem in the function space. Namely, the Pareto front $\mathscr{y}^\star$. Since MESMOC+'s acquisition is expressed as a sum of acquisition functions, one per each different black-box, its computational cost is linear with respect to the number of black-boxes. Moreover, it can be used in a decoupled evaluation setting in which one chooses not only the point at which to evaluate the black-boxes, but also what black-box to evaluate next. MESMOC+ solves several issues of MESMOC, an already existing acquisition function targeting the reduction of the entropy of the Pareto front. Specifically, the approximation of the acquisition function performed by MESMOC+ is more accurate than that of MESMOC. This is translated in better optimization results. In our experiments we have observed that MESMOC+ is competitive with other state-of-the-art methods for Bayesian optimization. However, MESMOC+ is characterized by a significantly smaller cost per iteration. Moreover, MESMOC+ is more robust with respect to the number of Monte Carlo samples of the Pareto front that are needed to approximate the acquisition function. Other strategies based on the reduction of the entropy of the solution of the optimization problem, such as PESMOC, obtain worse results with a smaller number of samples. Importantly, MESMOC+ is much easier to implement than PESMOC. Specifically, MESMOC+ has a smaller number of factors that require approximation in the conditional predictive distribution, and these factors are simpler. Namely, they only involve one random variable. PESMOC's factors, on the other hand, involve two random variables. We have also observed that the decoupled variant of MESMOC+ often obtains significantly better results than those of MESMOC+ in a coupled evaluation setting. Finally, MESMOC+ and its decoupled evaluation variant give in general better results than those of acquisition functions based on the expected improvement of the hyper-volume, such as BMOO qEHVI and qNEHVI. Importantly, these strategies do not allow for a decoupled evaluation setting, since one cannot compute the expected improvement of the hyper-volume simply by evaluating just a single black-box. Therefore, this limitation may hinder their performance in the benefit of MESMOC+.

CRediT authorship contribution statement

Daniel Fernández-Sánchez: Methodology, Software. **Eduardo C. Garrido-Merchán:** Software. **Daniel Hernández-Lobato:** Conceptualization, Validation, Supervision.

Data availability

We have a link in our manuscript to the code used.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors gratefully acknowledge the use of the facilities of Centro de Computación Científica (CCC) at Universidad Autónoma de Madrid. The authors also acknowledge financial support from Spanish Plan Nacional I + D+i, grant PID2019-106827 GB-I00/ AEI/ 10.13039/501100011033.

References

[1] R. Ariizumi, M. Tesch, H. Choset, F. Matsuno, Expensive multiobjective optimization for robotics with consideration of heteroscedastic noise, in: IEEE International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 2230–2235.
[2] Y. Collette, P. Siarry, Multiobjective optimization: principles and case studies, Springer Science & Business Media, 2004.
[3] E. Brochu, V.M. Cora, N. De Freitas, A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning, Technical Report TR-2009-023, University of British Columbia.
[4] C.E. Rasmussen, C.K. Williams, Gaussian Processes for Machine Learning, MIT press, 2006.
[5] B. Shahriari, K. Swersky, Z. Wang, R.P. Adams, N. De Freitas, Taking the human out of the loop: A review of Bayesian optimization, Proceedings of the IEEE 104 (1) (2015) 148–175.
[6] J.M. Hernández-Lobato, M. Gelbart, M. Hoffman, R. Adams, Z. Ghahramani, Predictive entropy search for bayesian optimization with unknown constraints, in: International conference on machine learning PMLR, 2015, pp. 1699–1707.
[7] M. Emmerich, J.-W. Klinkenberg, The computation of the expected improvement in dominated hypervolume of pareto front approximations, Rapport technique, Leiden University 34 (2008) 7–13.
[8] P. Feliot, J. Bect, E. Vazquez, A Bayesian approach to constrained single-and multi-objective optimization, Journal of Global Optimization 67 (1–2) (2017) 97–133.
[9] S. Daulton, M. Balandat, E. Bakshy, Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization, Advances in Neural Information Processing Systems 33 (2020) 9851–9864.
[10] S. Daulton, M. Balandat, E. Bakshy, Parallel bayesian optimization of multiple noisy objectives with expected hypervolume improvement, Advances in Neural Information Processing Systems 34 (2021) 2187–2200.
[11] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE transactions on evolutionary computation 3 (1999) 257–271.
[12] E.C. Garrido-Merchán, D. Hernández-Lobato, Predictive entropy search for multi-objective Bayesian optimization with constraints, Neurocomputing 361 (2019) 50–68.
[13] J. Villemonteix, E. Vazquez, E. Walter, An informational approach to the global optimization of expensive-to-evaluate functions, Journal of Global Optimization 44 (4) (2009) 509–534.
[14] P. Hennig, C.J. Schuler, Entropy search for information-efficient global optimization, Journal of Machine Learning Research 13 (6) (2012) 1809–1837.
[15] J.M. Hernández-Lobato, M.W. Hoffman, Z. Ghahramani, Predictive entropy search for efficient global optimization of black-box functions, Advances in neural information processing systems (2014) 918–926.
[16] D. Hernández-Lobato, J.M. Hernández-Lobato, A. Shah, R. Adams, Predictive entropy search for multi-objective bayesian optimization, in, International Conference on Machine Learning, PMLR (2016) 1492–1501.
[17] Z. Wang, S. Jegelka, Max-value entropy search for efficient bayesian optimization, in: International Conference on Machine Learning, PMLR, 2017, pp. 3627–3635.
[18] S. Belakaria, A. Deshwal, J.R. Doppa, Max-value entropy search for multi-objective Bayesian optimization, in: International Conference on Neural Information Processing Systems (NeurIPS), 2019, pp. 7825–7835.
[19] S. Belakaria, A. Deshwal, J.R. Doppa, Output space entropy search framework for multi-objective bayesian optimization, Journal of Artificial Intelligence Research 72 (2021) 667–715.
[20] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, in: International Conference on Learning Representations, 2014.

[21] J. Wilson, F. Hutter, M. Deisenroth, Maximizing acquisition functions for bayesian optimization, Advances in neural information processing systems 31 (2018) 906–9917.
[22] T.P. Minka, Expectation propagation for approximate bayesian inference, in: Uncertainty in Artificial Intelligence, Vol. 17, 2001, pp. 362–369.
[23] V. Perrone, I. Shcherbatyi, R. Jenatton, C. Archambeau, M. Seeger, Constrained Bayesian optimization with max-value entropy search, NeurIPS Workshop on Meta-Learning.
[24] S. Suzuki, S. Takeno, T. Tamura, K. Shitara, M. Karasuyama, Multi-objective bayesian optimization using pareto-frontier entropy, in: International Conference on Machine Learning, PMLR, 2020, pp. 9279–9288.
[25] F.W. Glover, G.A. Kochenberger, Handbook of metaheuristics, Vol. 57, Springer Science & Business Media, 2006.
[26] J.M. Hernández-Lobato, M.A. Gelbart, B. Reagen, R. Adolf, D. Hernández-Lobato, P.N. Whatmough, D. Brooks, G.-Y. Wei, R.P. Adams, Designing neural network hardware accelerators with decoupled objective evaluations, in: NIPS workshop on Bayesian Optimization, Vol. 10, 2016.
[27] A. Rahimi, B. Recht, et al., Random features for large-scale kernel machines., in: NIPS, Vol. 3, Citeseer, 2007, pp. 1177–1184.
[28] J. Wilson, V. Borovitskiy, A. Terenin, P. Mostowsky, M. Deisenroth, Efficiently sampling functions from gaussian process posteriors, in: International Conference on Machine Learning, PMLR, 2020, pp. 10292–10302.
[29] X. Boyen, D. Koller, Tractable inference for complex stochastic processes, in: International Conference on Uncertainty in Artificial Intelligence, 1998, pp. 33–42.
[30] M. Balandat, B. Karrer, D.R. Jiang, S. Daulton, B. Letham, A.G. Wilson, E. Bakshy, BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization, in: Advances in Neural Information Processing Systems 33, 2020. http://arxiv.org/abs/1910.06403.
[31] J. Snoek, H. Larochelle, R.P. Adams, Practical Bayesian optimization of machine learning algorithms, Advances in neural information processing systems (2012) 2951–2959.
[32] D. Chafekar, J. Xuan, K. Rasheed, Constrained multi-objective optimization using steady state genetic algorithms, Genetic and Evolutionary Computation Conference, Springer (2003) 813–824.
[33] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, IEEE transactions on evolutionary computation 6 (2) (2002) 182–197.
[34] D. Dua, C. Graff, https://archive.ics.uci.edu/ml/datasets/URL+ReputationUCI machine learning repository (2017). https://archive.ics.uci.edu/ml/datasets/URL+Reputation.
[35] G. Martínez-Muñoz, A. Suárez, Switching class labels to generate classification ensembles, Pattern Recognition 38 (10) (2005) 1483–1494.
[36] D. Hernández-Lobato, G. Martínez-Muñoz, A. Suárez, Statistical instance-based pruning in ensembles of independent classifiers, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (2) (2008) 364–369.
[37] Y. LeCun, C. Cortes, C.J.C. Burges, MNIST handwritten digit database, AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist 2.
[38] D.P. Kingma, J. Ba, ADAM: A method for stochastic optimization, International Conference on Learning Representations.
[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, The journal of machine learning research 15 (1) (2014) 1929–1958.
[40] Y.S. Shao, B. Reagen, G.-Y. Wei, D. Brooks, Aladdin: A pre-rtl, power-performance accelerator simulator enabling large design space exploration of customized architectures, in: ACM/IEEE 41st International Symposium on Computer Architecture (ISCA), IEEE, 2014, pp. 97–108.

Daniel Fernández-Sánchez received a B.S. degree in Computer Science from Universidad Autónoma de Madrid (UAM) in 2019 and a M.S. degree in Computational Intelligence and Interactive Systems in 2020 from the same institution. He is currently pursuing a Ph.D. degree at UAM. His thesis topic focuses on developing new methods for Bayesian Optimization based on information theory and on the use of Gaussian processes for prediction tasks. He is also a Teaching Assistant at the department of Computer Science of UAM since January 2021. His current research interests include Bayesian Optimization, Gaussian Processes, and their applications.

Eduardo C. Garrido-Merchán works as an assistant professor of machine learning, statistics and econometrics at the business faculty of Universidad Pontificia Comillas. He studied software engineering, where he was top 1 in its promotion at Universidad Pontificia Comillas, a Msc in artificial intelligence in Universidad Politecnica de Madrid and a Ph.D., Doctor Cum Laude, on Bayesian optimization at Universidad Autonoma de Madrid, where he also taught artificial intelligence and compilers. His research interests are new Bayesian optimization methods, artificial intelligence, Gaussian processes, machine learning and philosophy of artificial intelligence. Daniel Hernández-Lobato obtained a Ph.D. and a M.Phil. in Computer Science from Universidad

Daniel Hernández-Lobato obtained a Ph.D. and a M. Phil. in Computer Science from Universidad Autónoma de Madrid, Spain, in January 2010 and June 2007, respectively. His Ph.D. thesis received the award to the best thesis on Computer Science defended during that academic year in that institution. Between November 2009 and September 2011 he worked as a post-doc researcher at Université catholique de Louvain, Belgium. There he had the opportunity to collaborate with Prof. Pierre Dupont and Prof. Bernard Lauwerys in the identification of biomakers for the early diagnosis of arthritis. In September 2011, he moved back to Universidad Autónoma de Madrid, and since January 2014 he works there as a Lecturer of Computer Science. His research interests are mainly focused on the Bayesian approach to machine learning, including topics such as Bayesian optimization, Gaussian processes, and approximate Bayesian inference.