Engineering with Computers (2024) 40:2143–2159
https://doi.org/10.1007/s00366-023-01900-0

ORIGINAL ARTICLE

Cheap-expensive multi-objective Bayesian optimization for permanent magnet synchronous motor design

Nasrulloh Loka[1] · Mohamed Ibrahim[2,4,5] · Ivo Couckuyt[1] · Inneke Van Nieuwenhuyse[3] · Tom Dhaene[1]

Received: 4 March 2023 / Accepted: 11 September 2023 / Published online: 15 October 2023

Abstract

Bayesian optimization (BO) is a popular optimization technique for expensive-to-evaluate black-box functions. We propose a cheap-expensive multi-objective BO strategy for optimizing a permanent magnet synchronous motor (PMSM). The design of an electric motor is a complex, time-consuming process that contains various heterogeneous objectives and constraints; in particular, we have a mix of cheap and expensive objective and constraint functions. The expensive objectives and constraints are usually quantified by a time-consuming finite element method, while the cheap ones are available as closed-form equations. We propose a BO policy that can accommodate cheap-expensive objectives and constraints, using a hypervolume-based acquisition function that combines expensive function approximation from a surrogate with direct cheap evaluations. The proposed method is benchmarked on multiple test functions with promising results, reaching competitive solutions much faster than traditional BO methods. To address the aforementioned design challenges for PMSM, we apply our proposed method, which aims to maximize motor efficiency while minimizing torque ripple and active mass, and considers six other performance indicators as constraints.

Keywords Bayesian optimization · Multi-objectives optimization · Constrained optimization · Permanent magnet synchronous motor

✉ Nasrulloh Loka
nasrulloh.loka@ugent.be

Mohamed Ibrahim
mohamed.ibrahim@ugent.be

Ivo Couckuyt
ivo.couckuyt@ugent.be

Inneke Van Nieuwenhuyse
inneke.vannieuwenhuyse@uhasselt.be

Tom Dhaene
tom.dhaene@ugent.be

1 Department of Information Technology (INTEC), IDLab, Ghent University-imec, iGent, Technologiepark-Zwijnaarde 126, 9052 Ghent, Belgium

2 Department of Electromechanical, Systems and Metal Engineering, Ghent University, Technologiepark-Zwijnaarde 131, 9052 Ghent, Belgium

3 FlandersMake@UHasselt and Data Science Institute, Hasselt University, Martelarenlaan 42, 3500 Hasselt, Belgium

4 FlandersMake@UGent—Corelab MIRO, 3001 Leuven, Belgium

5 Electrical Engineering Department, Kafrelshiekh University, Kafrelshiekh 33511, Egypt

1 Introduction

Bayesian optimization (BO) [1–4] is a popular surrogate-based data-efficient technique for optimizing complex and time-consuming optimization problems [5, 6]. It is particularly useful when data is limited or expensive to acquire. This paper presents a case study in which both cheap and expensive objective and constraint functions are considered in the design of electric motors.

In the case of electric motor design, many geometric and electromagnetic parameters can affect the motor's performance. BO can help to efficiently identify the optimal combination of these parameters to achieve the desired performance indicators (such as high efficiency and high torque density, as required for use in electric vehicles).

Electric motor design optimization is highly relevant in practice, as electric motors consume about 40% of the generated energy worldwide [7]. Usually, the optimization is done using a Genetic Algorithm (GA) evaluated on Finite Element Methods (FEM), requiring large numbers of evaluations [8]. Such FEM evaluations can take hours to days, depending on the geometries of the motor under study;

consequently, this design optimization problem could benefit substantially from an efficient optimization method [7, 9, 10]. Our approach uses a surrogate model to approximate the expensive FEM evaluations [8, 11, 12]. Objectives and constraints that can be calculated cheaply (i.e., without the need for FEM, such as the total mass of the material) do not require a surrogate model, though; they can be quantified or approximated using deterministic closed-form formulas. Our approach distinguishes between the cheap and expensive functions in the optimization procedure; we show that this yields substantial improvements in data efficiency compared to traditional BO methods.

BO has two core components: a surrogate model and an acquisition function. The surrogate model is used to approximate the expensive output functions (either objectives or constraints) cheaply. The choice for the surrogate model is commonly a Gaussian Process (GP) [13, 14] or any other statistical model with uncertainty quantification capability such as Polynomial Chaos Expansion [15, 16], Neural Networks [17, 18], or Tree Parzen Estimators [19]. Based on the model prediction and the uncertainty quantification, an acquisition function is defined to sequentially search for the optimum design by balancing exploration and exploitation. A lot of BO research is available, accounting for different complexities in the optimization setting, such as batch optimization [20, 21], multi-fidelity [22], constrained optimization [23, 24], and multiple objectives [25, 26]. A review paper discussing problem settings in BO is presented in [27].

In Multi-Objective Bayesian Global Optimization (MOBGO), cheap and expensive objectives are commonly treated in the same manner, i.e., modeled using surrogate models. Some attempts to exploit cheap-expensive properties are presented in recent literature: Allmendinger et al. [28] extend the genetic algorithm approach to deal with cheap objectives by using a fast-first and interleaving method. Wang et al. [29–31] study the relationship between cheap-expensive objectives and search bias in evolutionary algorithm settings. Loka et al. [32] propose a hypervolume-based BO approach considering a mix of cheap and expensive objectives, but only for an unconstrained bi-objective setting.

In this study, a two-stage constrained MOBGO algorithm is presented to optimize a Permanent Magnet Synchronous Motor (PMSM) design. The algorithm explicitly distinguishes between the cheap and expensive output functions. The *first stage* is a constrained active learning (AL) step used to improve the accuracy of the expensive constraint predictions. This is especially useful when these constraint functions are highly irregular (showing many local optima) and/or when the feasible region of the solution space is very small (implying that the initial design may contain only a few or even no feasible designs). The *second stage* is the optimization stage, which uses the proposed hypervolume-based cheap-expensive constrained acquisition function (CEHVI-C). This function extends the work of Yang et al. [33] by incorporating the cheap *objectives* directly in the hypervolume calculation. The *cheap constraints* are accounted for in the optimizer of the BO acquisition function. We show that the resulting algorithm can attain competitive solutions faster than the traditional BO method.

The key contributions of this paper are the following:

- The proposed approach builds on a flexible way to quantify hypervolume, exploiting the distinction between cheap and expensive objectives. This improves the computational effort for calculating this metric. Moreover, contrary to the work in [33], it is applicable to any arbitrary box decomposition approach. Additionally, the algorithm handles expensive and cheap constraints in a clearly distinct way (accounting for the former in the probability of feasibility and for the latter in the optimization of the acquisition function). As shown, this results in an algorithm that is data-efficient and yields high-quality solutions.
- Using the proposed approach, we show that the PMSM design problem can be solved in a data-efficient manner, which outperforms the common approaches used to solve this problem in the literature. This is in itself an insightful result, as the FEM calculations are very expensive.

The rest of this paper is organized as follows: Sect. 2 describes the PMSM under study. Section 3 explains the basics of multi-objective optimization in general, along with the corresponding notation and terminology. Section 4 presents the proposed algorithm for constrained multi-objective problems with cheap and expensive outcomes. In Sect.5, the experimental setup and results are discussed. Finally, Sect. 6 presents the conclusions of this paper.

2 Permanent magnet synchronous motor optimization

A permanent magnet synchronous motor (PMSM) has several advantages compared to other types of electric motors, such as higher-power density and higher efficiency. Consequently, this type of motor is preferred in settings where power density and efficiency are critical, such as in automotive applications. The downside is that this motor uses rare-earth magnets, which are not only very expensive but also unfriendly to the environment because of the recycling problems [34, 35] and the impact of the mining activities [36].

Figure 1 shows a schematic drawing of the motor with the relevant geometrical design parameters, which are further detailed in Table 1. The magnets (referred to as rotor poles,

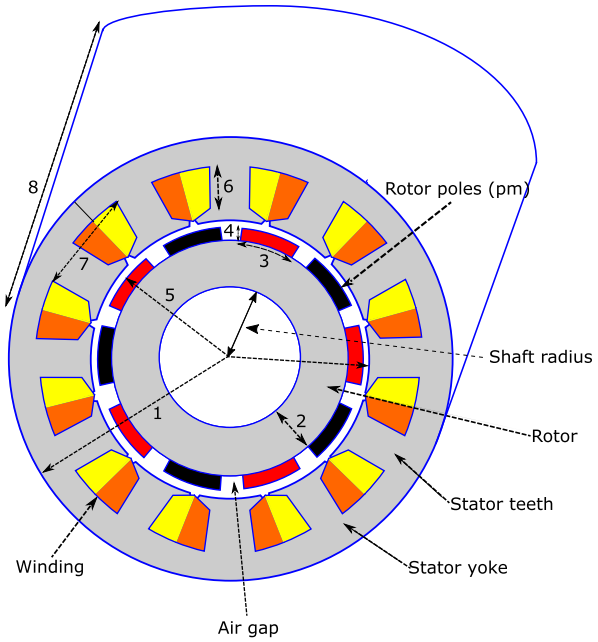


Fig. 1 Motor geometry with geometrical annotations

Table 1 Geometrical design parameters of the PMSM design optimization problem

Design variable	Type	Unit	Value
(1) Stator outer radius (SOR)	Constant	mm	96
(2) Rotor yoke thickness (RYt)	Variable	mm	5.0–20.0
(3) Width of pm (Wpm)	Variable		0.7–0.9
(4) Thickness of pm (Thpm)	Variable	mm	3.0–5.0
(5) Rotor outer radius (ROR)	Variable	mm	35.0–65.0
(6) Slot height (sh)	Variable	mm	10.0–25.0
(7) Slot teeth ratio (STR)	Variable		0.4–0.7
(8) Axial length (Lfe)	Variable	mm	40.0–60.0
(9) Number of pm (p)	Constant		10

pm) are located on the rotor in red and black (to reflect different polarities).

The PMSM design problem has three objectives (see Table 2) and six constraints (see Table 3).

The optimization of these parameters is nontrivial, though, as they tend to have conflicting impacts on the objectives: increasing the axial length of the motor (parameter (8) in Fig. 1), for instance, increases the average torque but simultaneously increases the total mass (and hence, cost). Consequently, this design optimization problem is a constrained multi-objective optimization problem. Some of the performance indicators in Tables 2 and 3 are cheap, meaning that they can be calculated by means of closed-form formulas. The expensive indicators are evaluated using

Table 2 Objectives for the optimization of a PMSM

Objective name	Type	Unit	Optimization type
Efficiency	Expensive	%	Maximization
Torque ripple	Expensive	N.m	Minimization
Total mass	Cheap	kg	Minimization

Table 3 Constraints for the optimization of a PMSM

Constraint name	Type	Constraint
Magnitude of flux density in stator yoke (BmagSY)	Expensive	$g_1 \leq 1.5\,\mathrm{T}$
Magnitude of flux density in stator teeth (BmagST)	Expensive	$g_2 \leq 1.5\,\mathrm{T}$
Thermal loading	Expensive	$g_3 \leq 8\,\mathrm{kW}/m^2$
Average torque (Tavg)	Expensive	$60Nm \leq g_4 \leq 65Nm$
Shaft radius	Cheap	$g_5 \geq 15mm$
Stator yoke thickness	Cheap	$g_6 \geq 5mm$

Finite Element Methods (FEM) [37]. More details on the performance indicators can be found in Appendix 1.

The PMSM problem is similar to the one proposed in [38, 39]. Some work has been done to perform multi-objective optimization on a similar motor design, but it relies on many FEM evaluations [39–41] and handcrafted optimization steps [39, 42].

This study focuses on developing a data-driven approach that also minimizes the number of expensive evaluations, and so that it can be applied to different problems and settings.

3 Constrained MOO: problem formulation

The goal of a constrained Multi-Objective Optimization (MOO) method is to optimize a set of objective functions $\boldsymbol{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_M(\mathbf{x})] \in \mathbb{R}^M$, while satisfying a set of constraints $\boldsymbol{g}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots, g_V(\mathbf{x})] \leq 0 \in \mathbb{R}^V$

$$\begin{aligned} &\min_{\mathbf{x}} \left(f_1(\mathbf{x}), \ldots, f_M(\mathbf{x})\right) \\ &\text{s.t. } g_v(\mathbf{x}) \leq 0, \quad v = 1, \ldots, V \end{aligned} \tag{1}$$

where $M \geq 2$ is the number of objectives, $V \geq 1$ is the number of constraints, and $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$. The set $\mathcal{X}$ is d-dimensional and bounded. Without loss of generality, this paper assumes that the objectives need to be minimized (except when explicitly stated otherwise). In MOO, there typically is no *single* optimal solution $\mathbf{x}^*$ that minimizes all objectives simultaneously while satisfying all constraints; instead, there is a *set of optimal solutions*, referred to as the Pareto set.

Mathematically, the Pareto set for an *unconstrained* optimization problem is defined as:

$$\boldsymbol{P} = \left\{\mathbf{x} \in \mathcal{X} \mid \nexists \mathbf{x}' \in \mathcal{X} : \mathbf{x}' \prec \mathbf{x}\right\} \tag{2}$$

where the notation $\mathbf{x}_b \prec \mathbf{x}_a$ means that $\mathbf{x}_b$ *dominates* $\mathbf{x}_a$. In a minimization problem with M objectives, $\mathbf{x}_b \prec \mathbf{x}_a$ if and only if $f_m(\mathbf{x}_b) \leq f_m(\mathbf{x}_a), \forall m \in \{1,..,M\}$ and $\exists m \in \{1,..,M\}$ such that $f_m(\mathbf{x}_b) < f_m(\mathbf{x}_a)$. Informally, we can say that $\mathbf{x}_b$ dominates $\mathbf{x}_a$ if and only if it is better in at least one objective, while not being worse in any of the other objectives. As evident from Eq. (2), $\boldsymbol{P}$ is defined in the input space; the image of the Pareto set in the objective space is referred to as the Pareto front: $\mathcal{P} = \left\{\boldsymbol{f}(\mathbf{x}) \in \mathbb{R}^M \mid \nexists \mathbf{x}' \in \mathcal{X} : \mathbf{x}' \prec \mathbf{x}\right\}$. In *constrained* problems, only feasible points $\mathbf{x}$ can be part of the Pareto set. We thus define Pareto feasible set as:

$$\boldsymbol{P}_{feas} = \left\{\mathbf{x} \in \mathcal{X} \mid \nexists \mathbf{x}' \in \mathcal{X} : \mathbf{x}' \prec \mathbf{x},\ \boldsymbol{g}(\mathbf{x}) \leq 0,\ \boldsymbol{g}(\mathbf{x}') \leq 0\right\} \tag{3}$$

For ease of notation, we denote $\boldsymbol{P} := \boldsymbol{P}_{feas}$ and $\mathcal{P} := \mathcal{P}_{feas}$ for every constrained problem in this paper.

In this work, Bayesian Optimization (BO) is used to find the Pareto set in a data-efficient manner (i.e., using the smallest possible number of function evaluations). Bayesian optimization has two main components: (1) the surrogate model, which approximates the expensive output functions, and (2) the acquisition function, which guides the BO procedure by sequentially selecting additional input points to evaluate. BO automatically balances exploration and exploitation [13, 14, 43]. The Gaussian Process (GP) is the most popular type of surrogate model used in BO; the technical details of the model can be found in Appendix 2. The proposed acquisition function is a key component and is discussed in the following section.

4 MOBGO algorithm for cheap-expensive objectives and constraints

Previous research on MOBGO algorithms commonly uses an acquisition function based on the hypervolume metric to search for the Pareto optimal points [25, 26, 33, 44]. Very recently, this type of acquisition function has been applied in mixed-variable settings [45], parallel evaluation settings [46], and for high-dimensional problems [47]. Yet, none of these previous works exploit potential differences in the latencies (i.e., the evaluation times) of the different objective functions. In real-life problems, it often occurs that the output functions (objectives and/or constraints) are a mix of cheap and expensive functions.

To the best of our knowledge, the only papers available so far on this topic are [28, 48] (which focus on exploiting latency differences in evolutionary algorithms), and [32] (which presents a BO algorithm limited to bi-objective unconstrained MOO problems). Recently, Buckingham et al. [49] proposed a scalarization-based multi-objective BO approach for a similar problem. However, their method assumes that the cheap objective does not have a closed-form formula.

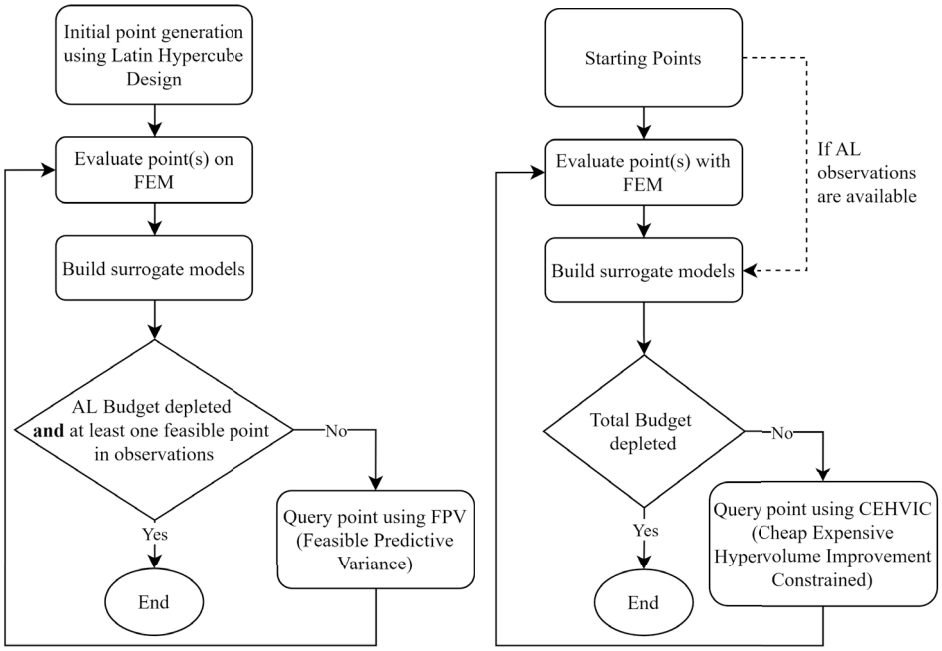
We propose a two-stage optimization approach (as in [50]), which is depicted in Fig. 2. The first stage is optional and is referred to as the Active Learning (AL) stage. It aims to improve the accuracy of the GPs for hard-to-model constraints (if any), using the Feasible Predictive Variance acquisition function discussed in Sect. 4.1. In the AL phase, the initial surrogates for these constraints are estimated based on a set of initial design points, which are evaluated using the expensive FEM models. The most common choice in the BO literature is a maximin Latin Hypercube design, [51] to ensure that the resulting set is space-filling. As the aim of the AL phase is to improve the accuracy of these constraint models, additional points are queried using the Feasible Predictive Variance acquisition function, which is discussed in detail in Sect. 4.1. The AL stage ends when the specified AL budget is depleted *and* there is at least one feasible point present in the observations. In some cases, the feasible area of the problem is very small, which may force the analyst to keep querying points until *both* conditions are fulfilled.

The second stage is the Bayesian optimization stage. If it was preceded by the AL stage, the resulting observations are used as starting points, on which the surrogate models are again estimated. If the AL phase was skipped, the starting points are generated through a space-filling design (usually a maximin Latin Hypercube design, for the reasons stated above), and they are first evaluated with the expensive FEM models to estimate the initial surrogate models. The additional points to evaluate are then selected using the proposed cheap-expensive hypervolume-based expected improvement acquisition function, which is discussed in Sect. 4.2. The algorithm ends when the total budget has been depleted, and the points that have been evaluated as feasible and Pareto-optimal are put forward as the points on the front.

As explained below, the acquisition functions of both stages account for the estimated Probability of Feasibility (PF) of the point with respect to the *expensive* constraints, to avoid spending the budget for evaluating points that are likely infeasible. The feasibility with respect to the *cheap* constraints is handled inside the optimization procedure that maximizes the acquisition functions, as discussed in Sect. 4.2.2.

4.1 Stage 1: active learning (AL)

This stage aims to improve the accuracy of the GP models of the expensive constraints by focusing solely on exploring



(a) Active learning (b) Bayesian Optimization

Fig. 2 Two-stage optimization scheme. The first stage (AL) is optional. We use it in our experiments for the PMSM case. The data set resulting from the active learning steps in **a** are then used as the starting points in **b**. If the AL phase is skipped, the starting points in **b** are generated by a Latin Hypercube design

the region where the model exhibits high uncertainty. This is especially useful when the expensive constraint functions are non-smooth (i.e., highly irregular, showing many local optima, which makes the function hard to model) and/or when the feasible region of the solution space is very small (implying that the initial design may contain only a few or even no feasible designs). In such cases, the information gained during the AL stage results in significant efficiency gains in the optimization stage.

The acquisition function used is the Feasible Predictive Variance (FPV), which is defined as:

$$FPV = \prod_{v=1}^{V} \sigma_v^2 \cdot PF_v(\mathbf{x}_*), \tag{4}$$

where σ_v^2 is the predictive variance of the *hard-to-model* constraint v at x_*, and PF_v refers to the Probability of Feasibility (see e.g. [23]) of x_* for constraint v:

$$PF_v(\mathbf{x}_*) := \Pr[\tilde{g_v}(\mathbf{x}_*) \leq 0] \tag{5}$$

$$= \int_{-\infty}^{0} p\big(\tilde{g_v}(\mathbf{x}_*) \mid \mathbf{x}_*, \mathcal{T}_N\big) d\tilde{g_v}(\mathbf{x}_*), \tag{6}$$

where $\tilde{g_v}(\mathbf{x}_*)$ refers to the Gaussian process outcome for constraint v at $\mathbf{x}_*$, and $\mathcal{T}_N$ denotes the data set already available for constraint v.

In the PMSM case study, we use this stage specifically for the average torque constraint (see Table 3) as this constraint is hard to model, and, moreover, it restricts the number of feasible solutions more severely than the other constraints.

4.2 Stage 2: Bayesian optimization

After finding enough feasible solutions, new points are selected by maximizing the Cheap-Expensive Expected Hypervolume Improvement with Constraints (CEHVI-C) acquisition function:

$$\text{CEHVI-C} = \text{CEHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) \cdot \prod_{v=1}^{V} PF_v(\mathbf{x}_*) \tag{7}$$

CEHVI-C multiplies the proposed CEHVI acquisition function (see Sect. 4.2.1) with the probability of feasibility of *all* expensive constraints g_v ($v = 1 \cdots V$). Assuming that all these constraints are independent, this reduces to a multiplication of the individual $PF_v(\mathbf{x}_*)$. Note that equation 7 also implicitly assumes conditional independence between the objective and constraint functions [23]. Both assumptions are standard in constrained Bayesian optimization.

The *cheap constraints* are not directly incorporated in Eq. 7, since this would likely introduce severe non-smooth behavior in the response surface of the acquisition function. Instead, the cheap constraints are accounted for in the optimization procedure implemented to maximize the acquisition function, which is further detailed in Sect. 4.2.2.

4.2.1 Cheap-expensive expected hypervolume improvement

We extend the EHVI formulation presented in [33] such that (1) it can efficiently handle a mix of cheap and expensive objectives, and (2) it can be used independent of the hyperbox decomposition method chosen to implement the calculations.

In hypervolume-based MOBGO, the notion of improvement by the Lebesgue measure is used. Let us first define the hypervolume indicator (HVI) $\mathcal{H}$ [52, 53]. Given a Pareto front $\mathcal{P}$, the hypervolume indicator $\mathcal{H}$ of this front $\mathcal{P}$ w.r.t. a reference point $\mathbf{r}$ is defined as follows [52, 53]:

$$\mathcal{H}(\mathcal{P}, \mathbf{r}) = \lambda_M\left(\cup_{\mathbf{y}\in\mathcal{P}}[\mathbf{r}, \mathbf{y}]\right), \tag{8}$$

where λ_M is the Lebesgue measure of the region that dominates $\mathbf{r}$ and that is dominated by $\mathcal{P}$ (in $\mathbb{R}^M$, i.e., for $\mathbb{R}^2$, λ_2 is the area of the dominated region, while on $\mathbb{R}^3$, λ_3 is the volume).

Using this definition, we can define the *hypervolume improvement* (also referred to as exclusive hypervolume) generated by a new point $\mathbf{y}_*$ as:

$$\mathcal{H}_{exc}(\mathbf{y}_*, \mathcal{P}, \mathbf{r}) = \mathcal{H}(\mathcal{P} \cup \{\mathbf{y}_*\}, \mathbf{r}) - \mathcal{H}(\mathcal{P}, \mathbf{r}). \tag{9}$$

Based on the definition of hypervolume improvement in Eq. 9, we can define the Expected HyperVolume Improvement (EHVI) at an arbitrary new design point $\mathbf{x}_\star$ as:

$$\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \int_{\mathbb{R}^M} \mathcal{H}_{exc}(\mathbf{y}, \mathcal{P}, \mathbf{r}) \cdot \boldsymbol{\xi}_{\sigma,\mu}(\mathbf{y}) d\mathbf{y}, \tag{10}$$

where $\mathbf{y}$ corresponds to a (random) M-variate objective vector, while $\boldsymbol{\xi}_{\sigma,\mu}(\mathbf{y})$ denotes the value of the M-variate independent normal density function in this vector (given the predictive mean vector $\boldsymbol{\mu} \in \mathbb{R}^M$ at $\mathbf{x}_\star$, and the predictive variance vector $\boldsymbol{\sigma^2} \in \mathbb{R}^M_+$ at $\mathbf{x}_\star$). For ease of notation, let $\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) := \text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma})$.

Let us define a set $\Delta(\mathbf{y}, \mathcal{P}, \mathbf{r})$ which contains (given a Pareto front $\mathcal{P}$, the reference point $\mathbf{r}$ and the output vector $\mathbf{y}$) all the output vectors that currently do *not* belong to the dominated set, but that would be added to it when the vector $\mathbf{y}$ were added to the front [33, 54]:

$$\Delta(\mathbf{y}, \mathcal{P}, \mathbf{r}) = \left\{\mathbf{z} \in \mathbb{R}^M \mid \mathbf{y} \prec \mathbf{z}, \mathbf{z} \prec \mathbf{r} \text{ and } \nexists \mathbf{q} \in \mathcal{P} : \mathbf{q} \prec \mathbf{z}\right\}, \tag{11}$$

For notational simplicity, let $\Delta(\mathbf{y}, \mathcal{P}, \mathbf{r}) := \Delta(\mathbf{y})$. The EHVI in Eq. 10 can then be rewritten as:

$$\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}) = \sum_{i=1}^{N_M} \left(\int_{-\infty}^{y_1=u_1^{(i)}} \cdots \int_{-\infty}^{y_M=u_M^{(i)}} \right) \lambda_M\left[S_M^{(i)} \cap \Delta(\mathbf{y})\right] \cdot \xi_{\mu,\sigma}(\mathbf{y}) d\mathbf{y}, \tag{12}$$

where λ_M refers to the M-dimensional Lebesgue measure, $S_M^{(i)}$ refers to hyperbox i (see Eq. B20 in Appendix 2), and N_M denotes the total number of hyperboxes in the decomposition. Note that Eq. 12 allows for piece-wise integration, given the summation over the different hyperboxes.

Dividing each integration slice $\int_{-\infty}^{y_m=u_m^{(i)}}$ into $(\int_{-\infty}^{y_m=l_m^{(i)}} + \int_{l_m^{(i)}}^{y_m=u_m^{(i)}})$, we obtain:

$$\begin{aligned}&\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}) \\ &= \sum_{i=1}^{N_M} \left(\left(\int_{-\infty}^{y_1=l_1^{(i)}} + \int_{y_1=l_1^{(i)}}^{y_1=u_1^{(i)}} \right) \cdots \left(\int_{-\infty}^{y_M=l_M^{(i)}} + \int_{y_M=l_M^{(i)}}^{y_M=u_M^{(i)}} \right) \right) \\ &\quad \lambda_M\left[S_M^{(i)} \cap \Delta(\mathbf{y})\right] \cdot \xi_{\mu,\sigma}(\mathbf{y}) d\mathbf{y},\end{aligned} \tag{13}$$

as evident, each of the individual terms of this sum consist of the multiplication of M factors, each of which contains the sum of 2 integrals. Since integration is a linear mapping, we can expand each individual term in Eq. 13, resulting in a summation of 2^M terms, each consisting of an M-dimensional integral.

Let us finally define $\mathbf{C}^{(j)_2}$ as a binary representation of such an M-dimensional integral. $\mathbf{C}^{(j)_2}$'s length is thus equal to M. The kth element, $C_k^{(j)_2}$, equals 0 if the kth integral has finite bounds, and 1 if the lower bound is $-\infty$.

Using the results from [26, 33], the EHVI can then be calculated exactly as follows:

$$\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}) = \sum_{i=1}^{N_M} \left(\sum_{j=0}^{2^M} \left(\prod_{k=1}^{M} \omega_e(i, k, C_k^{(j)_2}) \right) \right), \tag{14}$$

with

$$\omega_e\left(i, k, C_k^{(j)_2}\right) := \begin{cases} \Psi\left(l_k^{(i)}, u_k^{(i)}, \mu_k, \sigma_k\right) & \text{if } C_k^{(j)_2} = 0 \\ \vartheta\left(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k\right) & \text{if } C_k^{(j)_2} = 1, \end{cases} \tag{15}$$

where i refers to the index number of the hyperbox, k to the index number of the objective function, and $C_k^{(j)_2}$ is the binary representation of the kth objective. Note that Eq. 14 implicitly uses the independence assumption between the different objectives, to replace the M-dimensional integrals by multiplication of M single-dimensional integrals. As evident from Eq. 15, the calculation of these single integrals can be done exactly, but depends on whether the integral has finite bounds ($C_k^{(j)_2} = 0$) or an infinite lower bound ($C_k^{(j)_2} = 1$). More specifically, for $C_k^{(j)_2} = 1$, we have:

$$\vartheta\left(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k\right) = \left(u_k^{(i)} - l_k^{(i)}\right) \cdot \left(\Phi\left(\frac{l_k^{(i)} - \mu_k}{\sigma_k} \right) \right), \tag{16}$$

where Φ denotes the Cumulative Distribution Function (CDF) of the standard normal distribution. Equation 16

also occurs in [33], but we adjust it here for a minimization context.

For $C_k^{(j)_2} = 0$, we have:

$$\Psi\left(l_k^{(i)}, u_k^{(i)}, \mu_k, \sigma_k\right) = \Psi_{-\infty}\left(u_k^{(i)}, u_k^{(i)}, \mu_k, \sigma_k\right) - \Psi_{-\infty}\left(u_k^{(i)}, l_k^{(i)}, \mu_k, \sigma_k\right), \tag{17}$$

with

$$\Psi_{-\infty}(a, b, \mu, \sigma) := \int_{-\infty}^{b} (a - z)\frac{1}{\sigma}\phi\left(\frac{z-\mu}{\sigma}\right)dz \tag{18}$$

$$= \sigma\phi\left(\frac{b-\mu}{\sigma}\right) + (a-\mu)\left[\Phi\left(\frac{b-\mu}{\sigma}\right)\right], \tag{19}$$

where ϕ and Φ denote the Probability Density Function (PDF) and Cumulative Distribution Function (CDF) of the standard normal distribution, respectively.

We can further refine Eq. 14 to deal efficiently with cheap and expensive objectives. To that end, we introduce an M-dimensional binary vector $\mathbf{t}^f$: the kth element of this vector, t_k^f, equals 0 if the kth objective is cheap, and 1 otherwise.We can then efficiently calculate the resulting Cheap-Expensive Hypervolume Improvement (CEHVI) as follows:

$$\text{CEHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}) = \sum_{i=1}^{N_M}\left(\sum_{j=0}^{2^M}\left(\prod_{k=1}^{M}\omega(i, k, C_k^{(j)_2}, t_k^f)\right)\right), \tag{20}$$

with

$$\omega(i, k, C_k^{(j)_2}, t_k^f) := \begin{cases} \omega_c(i, k, C_k^{(j)_2}) & \text{if } t_k^f = 0 \\ \omega_e(i, k, C_k^{(j)_2}) & \text{if } t_k^f = 1. \end{cases} \tag{21}$$

For the expensive objectives, this expression reduces to ω_e as given by Eq. 15. For the cheap objectives, the calculation of $\omega_c(i, k, C_k^{(j)_2})$ depends on the relative location of y_k w.r.t. $l_k^{(i)}$ and $u_k^{(i)}$, as shown in Table 4. As evident, the resulting values are *deterministic*.

4.2.2 Acquisition function maximization

In every iteration, the query points are obtained by maximizing the acquisition function (usually referred to as *inner optimization*). We use a multi-start optimization method [55, 56] that incorporates the cheap constraints. First, the Monte Carlo method is used to sample 5000 points, and the points that violate cheap constraints are removed. We then select the 10 points with the highest CEHVI-C value and apply Sequential Least Square Programming (SLSQP) with the cheap constraints [57] to each of these in parallel (using the SLSQP implementation of the Scipy [58] library).

Table 4 Calculation of $\omega_c(i, k, C_k^{(j)_2})$ for the cheap objectives

Condition	Value
$y_k < l_k^{(i)} < u_k^{(i)}$	$\omega_c\left(i, k, C_k^{(j)_2}\right) = \begin{cases} 0 & \text{if } C_k^{(j)_2} = 0 \\ (u_k^{(i)} - l_k^{(i)}) & \text{if } C_k^{(j)_2} = 1 \end{cases}$
$l_k^{(i)} < y_k < u_k^{(i)}$	$\omega_c\left(i, k, C_k^{(j)_2}\right) = \begin{cases} (u_k^{(i)} - y_k) & \text{if } C_k^{(j)_2} = 0 \\ 0 & \text{if } C_k^{(j)_2} = 1 \end{cases}$
$l_k^{(i)} < u_k^{(i)} < y_k$	$\omega_c\left(i, k, C_k^{(j)_2}\right) = 0$

5 Result and discussion

5.1 Experiment settings

The proposed hypervolume-based MOBGO algorithm was implemented using Trieste [59] in Python. Before applying our method in the Motor Optimization case, we consider five benchmark functions to test the performance of the proposed algorithm, by testing it on three unconstrained optimization problems (DTLZ1, DTLZ2, and DTLZ3 [60]) and two constrained optimization problems (BNH [61] and SRN [62]). The characteristics of these benchmark functions are presented in Table 5. The reference point indicated in the table is used for the hypervolume computations.

For these benchmark functions, $(d \times 11 + 1)$ initial design points were generated using quasi-random Halton Sampling [63]. The proposed method is compared with EHVI(-Constrained), Random sampling, and NSGA-II (see [64]; we used the version present in PyMOO [65], which accounts for constraints). The total budget for the FEM simulator is set to 100 input evaluations, except for the NSGA-II algorithm: as this method is less data efficient, we allow it to spend 250 input evaluations. The AL budget is set to zero, as the expensive constraints (if any) are not hard to model. Evidently, for the unconstrained problems, the CEHVI-C acquisition function reduces to the CEHVI acquisition function (see Eq. 7).

For the PMSM optimization problem, 35 initial points are generated using Latin Hypercube sampling [51]. The AL budget for the BO methods is set to 10 iterations; we include the AL phase in this problem as the initial design is small, so we expect it to be beneficial, especially for learning the hard-to-model average torque (Tavg) constraint. Both the initial design and the number of AL iterations are deliberately kept small as the FEM model is relatively slow to run. We compare the performance of our algorithm against the same competitors as in the benchmark functions. The total budget equals 100 input evaluations, except for the NSGA-II algorithm (250 evaluations).

Table 5 Full specification of the benchmark functions

Name	Input d	Objectives		Constraints		Reference point
		Expensive	Cheap	Expensive	Cheap	
DTLZ1	6	2	1	0	0	[425, 425, 425]
DTLZ2	6	2	1	0	0	[2.5, 2.5, 2.5]
DTLZ3	6	2	1	0	0	[825, 825, 825]
BNH	2	1	1	1	1	[150, 100]
SRN	2	1	1	1	1	[800, 200]

5.2 Results for benchmark functions

We carried out 10 repetitions for each of the benchmark experiments, to check the robustness of the results against the randomness involved in the algorithms (which is evident in the NSGA-II and Random algorithms; in the BO algorithms, it impacts the multistart design of the inner optimization).

Figure 3 shows the evolution of the mean hypervolume on the different benchmark functions, for the competing algorithms. As shown, the BO approaches (CEHVI and EHVI) clearly outperform the competing algorithms in the unconstrained problem settings (top row). Moreover, the CEHVI algorithm has significantly better performance than the EHVI algorithm in the DTLZ1 and DTLZ3 experiments, which have a disjoint Pareto front [60] and are thus hard to optimize (the DTLZ2, by contrast, has a smooth Pareto front).

In the constrained benchmark problems (bottom row), CEHVI-C again has a clearly higher hypervolume indicator value than the other methods. While both problems have a smooth Pareto front, CEHVI-C outperforms EHVI-C in the SRN problem (in the BNH problem, the performance of both algorithms is similar, as the cheap function is smoother and thus relatively easy to model with GPs).

Table 6 gives an overview of the final hypervolume obtained at the end of the different algorithms, along with the difference (in %) from the true optimal hypervolume. As evident from this table, CEHVI-C is the winner in all test problems except in BNH. Note, though, that NSGA-II only succeeds in outperforming both BO approaches here because we gave it a significantly higher total budget; for limited

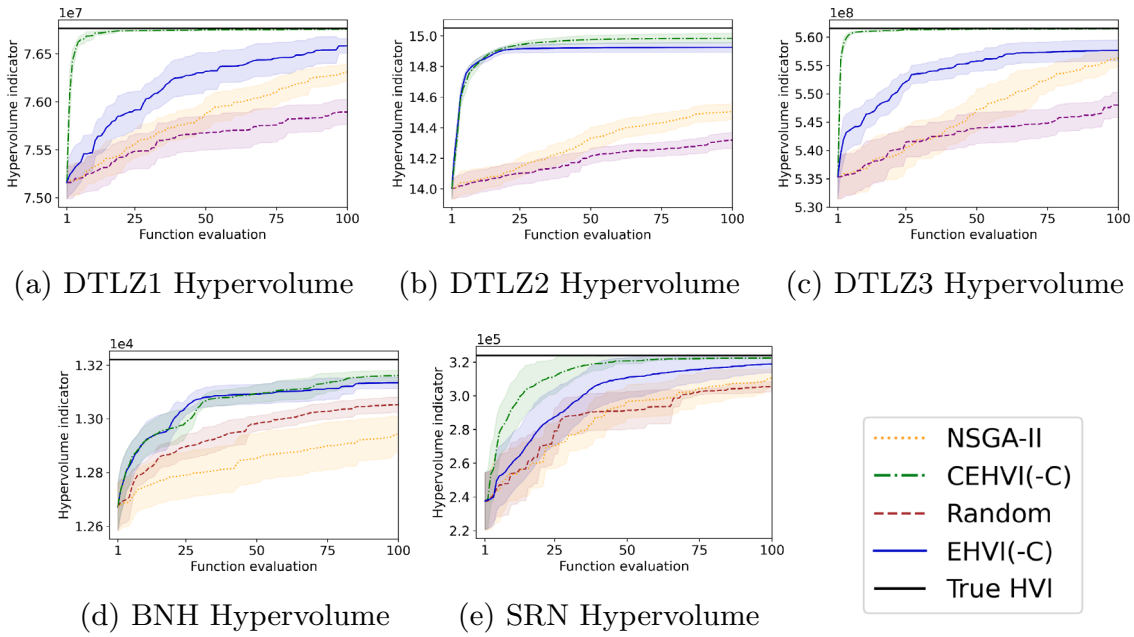


Fig. 3 Evolution of the mean hypervolume on the benchmark problems, with a total budget of 100 evaluations. The shaded areas reflect the 95% confidence intervals based on 10 repetitions

Table 6 Overview of the mean hypervolume and the % difference from the true optimal hypervolume (with 95% confidence interval), obtained at the end of the algorithms, for the benchmark problems

Test Problem	BO Budget	Method	HVI	Δ to Ground truth (%)
DTLZ1	100	Random	7.5892e7 ± 1.3126e5	1.13771 ± 0.27586
	100	EHVI	7.6585e7 ± 7.5134e4	0.23531 ± 0.15791
	100	CEHVI	**7.6759e7**± 4.6615e3	**0.00822 ± 0.00979**
	250	NSGA2	7.6571e7 ± 6.8405e4	0.25402 ± 0.14376
DTLZ2	100	Random	1.4318e1± 0.05256	4.86801 ± 0.56346
	100	EHVI	1.4925e1 ± 0.03135	0.83512 ± 0.33607
	100	CEHVI	**1.4985e1**± 0.03113	**0.43829 ± 0.33369**
	250	NSGA2	1.4713e1± 0.03997	2.24603 ± 0.42848
DTLZ3	100	Random	5.4802e8 ± 2.2293e6	2.40430 ± 0.64053
	100	EHVI	5.5766e8 ± 1.8455e6	0.68636 ± 0.53025
	100	CEHVI	**5.6148e8**± 2.5549e4	**0.00621 ± 0.00734**
	250	NSGA2	5.5988e8± 5.3153e5	0.29195 ± 0.15272
BNH	100	Random	1.3127e4± 1.6295e1	0.69974 ± 0.19886
	100	EHVI-C	1.3134e4 ± 2.1248e1	0.65131 ± 0.25931
	100	CEHVI-C	1.3161e4± 1.9731e1	0.44741 ±0.24080
	250	NSGA2	**1.3174e4**± 3.2091e1	**0.35001 ± 0.39163**
SRN	100	Random	3.1101e5 ± 2.9123e3	3.95808 ± 1.45103
	100	EHVI-C	3.1885e5± 5.2265e3	1.53636 ± 2.60401
	100	CEHVI-C	**3.2240e5**± 4.2819e2	**0.43908 ± 0.21333**
	250	NSGA2	3.1885e5 ± 1.9090e3	1.53684 ± 0.95111

The best result for each problem is highlighted in bold

budgets (<= 100), NSGA-II is clearly inferior (as evident from Fig. 3). To assess the performance on varying input and output dimensions, extra experiments are conducted and presented in Appendix 2.

5.3 Results for PMSM design problem

To check the robustness of the algorithms against randomness, we ran 10 repetitions. The hypervolume values are calculated with reference point [Efficiency, Torque ripple, Total mass] = [0.80, 8.1, 23.].

Figure 4 shows the evolution of the mean hypervolume for the different algorithms. For the BO algorithms, we used the first ten iterations to implement an AL stage: here, points were queried by implementing the FPV acquisition function on the average torque constraint, to improve the corresponding GP model. As evident from the figure, the AL phase already succeeds in improving the hypervolume. In the optimization stage, we see that CEHVI-C performs better than EHVI-C; NSGA-II is clearly inferior to both BO approaches. Actually, it even fails to query feasible points at certain iterations (even the later ones) due to the many constraints in the PMSM design problem. As a result, its hypervolume only improves marginally in the first 100 iterations. As evident from Table 7, which shows the expected hypervolume obtained at the end of the algorithms, the improvements obtained remain marginal even at a higher budget.

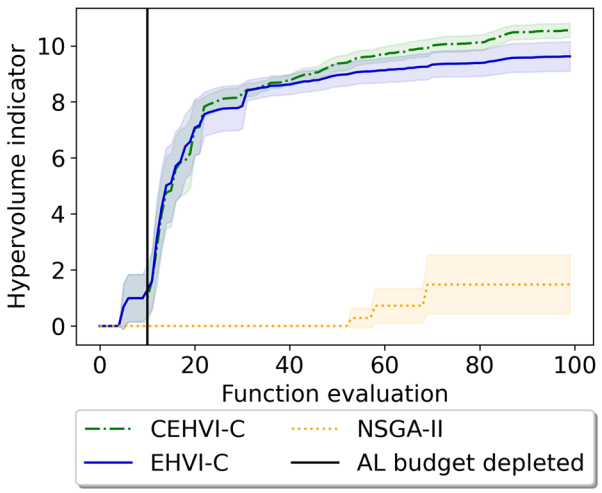


Fig. 4 Evolution of the mean hypervolume for the PMSM design problem, for a total budget of 100 iterations. The shaded areas reflect the 95% confidence interval based on 10 repetitions

Figure 5 illustrates the quality of the final Pareto front obtained by the CEHVI-C and the EHVI-C methods for a single-arbitrary run. Clearly, CEHVI-C succeeds in achieving solutions with a lower Torque ripple and a lower Total mass than EHVI-C without compromising motor efficiency. The CEHVI-C runs also take less computation time, as it avoids any estimations for the cheap objective and constraints.

Table 7 Overview of the mean hypervolume (with 95% confidence interval halfwidth), obtained at the end of the algorithms, for the PMSM design problem

Method	AL Budget	Total Budget	HVI
EHVI-C	10	100	9.6239 ± 0.2607
CEHVI-C	10	100	**10.5554** ± 0.1258
NSGA-II	0	250	1.4818 ± 0.5237

The best result is highlighted in bold

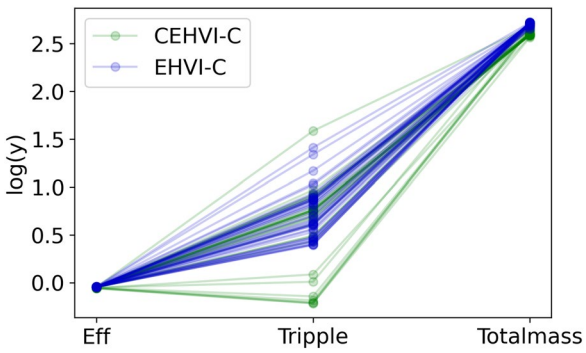


Fig. 5 Log of Pareto front of a single experiment run

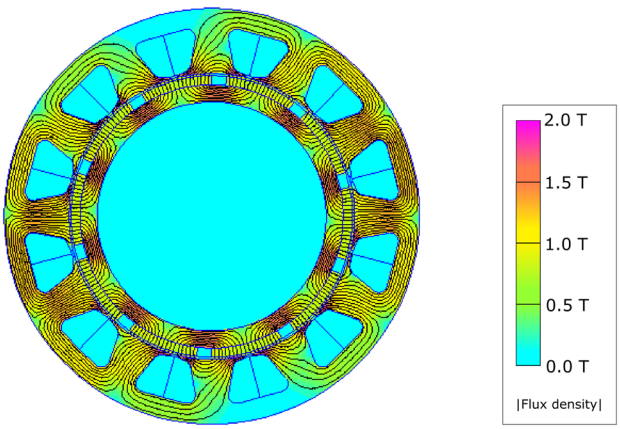


Fig. 6 One of the Pareto-optimal designs found by CEHVI-C

The three objectives of the optimization are the torque ripple, the motor efficiency, and the total mass. The magnet mass is also added because it is the most expensive part of the machine. Generally speaking, the cost of 1 kg of rare-earth magnets equals more than ten times that of 1 kg of copper or 1 kg of iron [66, 67].

Figure 6 shows the optimal geometry of an (arbitrary) Pareto-optimal PMSM design obtained by CEHVI-C, along with the flux lines and flux densities. While there is saturation in some parts of the core, it does not impact the performance metrics in any negative way.

6 Conclusion

In this paper, a hypervolume-based MOBGO approach has been presented and applied in view of optimizing a Permanent Magnet Synchronous Motor design. This design problem consists of a mix of expensive performance metrics (which require FEM evaluations) and cheap performance metrics (which can be evaluated using closed-form expressions). The key strength of the proposed approach is that it distinguishes between these cheap and expensive functions, by only estimating Gaussian Process models for the expensive outcomes. It includes an active learning stage (which uses the FPV acquisition function to improve the accuracy for hard-to-model constraints) and an optimization phase (which uses the proposed CEHVI-C acquisition function, which is a constrained and cheap–expensive version of the well-known EHVI criterion). The performance of the CEHVI-C function was first evaluated on a number of benchmark functions; as shown, it leads to superior performance over the standard EHVI-based approaches, especially when the cheap objective(s) are hard to model with GP. This superiority was further confirmed in the PMSM design results.

The proposed approach is likely beneficial for other engineering design problems that include cheap and expensive outcomes. In future research, we plan to extend the approach further such that it can handle noisy function evaluations. Another interesting topic is to extend the method to be cost-aware. Indeed, the cost of an expensive function evaluation may not be the same over the entire search space; cost-aware BO may then try to find the optimal solutions while minimizing both the number of function evaluations and the resulting evaluation cost.

Appendix 1 details of the PMSM problem

Expensive objectives

The expensive objectives are evaluated using Finite Element Methods (FEM) [37]. The motor torque objective is given by:

$$T_m = \frac{p}{2}\frac{\pi \mu_o D_g L_{fe}}{2L_g} F_s F_r \sin\left(\delta_{sr}\right), \tag{22}$$

where p refers to the number of rotor poles, μ_o is the permeability of the air, D_g is the airgap diameter, L_{fe} is the axial length, and L_g is the length of the airgap. The notations F_s, F_r, and δ_{sr} refer to the magnetomotive force of the stator and the rotor, and the angle between them, respectively. F_s depends mainly on the winding geometry (area, number of

turns, and phases) and on the permeable current density. F_r depends mainly on the properties and geometry of the magnet, as well as on g and L_{fe}. The calculation of F_s and F_r relies on the expensive FEM evaluation.

The efficiency of the motor is given by:

$$\eta = \frac{T_m \omega_r}{T_m \omega_r + P_l} 100, \tag{23}$$

where ω_r is the rotor speed, and P_l refers to the motor losses: copper loss, magnet loss, and iron (stator and rotor core losses). P_l depends on the flux density, geometry, material properties, current density, and the speed of the motor. P_l is evaluated by FEM.

Cheap objectives and constraints

Total mass calculation

The cheap objective function for the motor design problem is the total mass of the following motor parts: part_1 = stator core (silicon steel), part_2 = rotor core (silicon steel), part_3 = winding (copper), and part_4 = rotor poles (magnets). The mass of part_n can be calculated as follows:

$$\text{Volume part}_n = \text{Area part}_n \times \text{Length part}_n \tag{24}$$

$$\text{Mass part}_n = \text{Volume part}_n \times \text{Density part}_n, \tag{25}$$

where steel density = 7267.5 kg/m^2, copper density = 8933 kg/m^2, magnet density = 7400 kg/m^2. Then, the total mass of the motor can be obtained as follows:

$$\text{Total mass} = \sum_{n=1}^{4} \text{Mass part}_n. \tag{26}$$

Cheap constraints

For the calculation of the cheap constraint functions, we first need to define the following constants: airgap length (L_g) = 1, slot opening height (soh) = 1, slot wedge width (swx) = 1, slot width yoke side ratio (swyR) = 1.5, slots = 12. The stator Y thickness (SYt) can then be calculated using following formulas:

$$\text{Rslotmiddle} = \text{ROR} + L_g + \text{soh} + \text{swx} + (0.5 \times \text{sh}) \tag{27}$$

$$\text{LMslot} = \text{STR} \times 2\pi \times \frac{\text{Rslotmiddle}}{\text{slots}} \tag{28}$$

$$\text{sw} = \text{swyR} \times \text{LMslot} \tag{29}$$

$$\text{Rteeth} = \sqrt{(\text{ROR} + L_g + \text{soh} + \text{swx} + \text{sh})^2 + (0.5 \times \text{sw})^2} \tag{30}$$

$$\text{SYt} = \text{SOR} - \text{Rteeth} \tag{31}$$

The second cheap constraint, shaft diameter (ShaftD) is defined by:

$$\text{ShatfD} = 2 \times \text{ROR} - 2 \times \text{RYt} - 2 \times \text{Thpm} \tag{32}$$

Appendix 2 Background on Bayesian optimization

GP model details

The Gaussian Process (GP) model [13, 14, 43] is the most popular type of surrogate model used in BO, especially if the input domain is continuous. Informally, a GP defines a distribution over real-valued functions: $f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$, and is fully specified by its mean function $m(\mathbf{x})$ and its (positive semi-definite) covariance function $k(\mathbf{x}, \mathbf{x}')$.

A GP provides a predictive distribution for the output function under study at unobserved input locations in the search space, given a (limited) set of available input/output data. Suppose we want to model an output function f_m, for which we have evaluated the set of data points $X = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, yielding function evaluations $Y_m = [f_m(\mathbf{x}_1), \ldots, f_m(\mathbf{x}_N)]$. Then, $\mathcal{D}_N = \{X, Y_m\}$ is defined as the observed data so far in our optimization process, and the GP model is trained on these data (usually by means of maximum likelihood estimation, as discussed below).

The means and variances of the predictive distribution $\boldsymbol{f}_\star$, at any set of unobserved data points $X_\star = [\mathbf{x}_{\star 1}, \ldots, \mathbf{x}_{\star L}]$, can then be estimated as follows:

$$\mu_m(X_\star) = \mathbb{E}(\boldsymbol{f}_\star \mid X_\star, \mathcal{D}_N) = K_{\star \text{x}} K_{\text{xx}}^{-1} Y_m \tag{33}$$

$$\sigma_m^2(X_\star) = \text{Var}(\boldsymbol{f}_\star \mid X_\star, \mathcal{D}_N) = K_{\star\star} - K_{\star \text{x}} K_{\text{xx}}^{-1} K_{\star \text{x}}^T \tag{34}$$

where $\mu_m(X_\star)$ is the $L \times 1$ vector with the predictive means, and $\sigma_m^2(X_\star)$ is the $L \times 1$ vector with the predictive variances. The notation K_{xx} refers to the $N \times N$ matrix containing the estimated covariances between the available data, i.e., $k(\mathbf{x}_i, \mathbf{x}_j)$ for $i, j = 1 \ldots N$. The notation $K_{\star \text{x}}$ is the $L \times N$ matrix containing the covariance estimates between the new points $X_\star$ and the N available points, i.e., $k(\mathbf{x}_{\star i}, \mathbf{x}_j)$, for $i = 1 \ldots L, j = 1 \ldots N$. The notation $K_{\star\star}$ refers to the $L \times L$ matrix containing the estimated covariances between the new points, i.e., $k(\mathbf{x}_{*i}, \mathbf{x}_{*j})$ for $i, j = 1 \ldots L$. To model the covariance function, we choose the Matérn 5/2 kernel [68]. This kernel is a common choice when the smoothness of the function is unknown [6], since it does not make any overly

smooth assumptions with respect to the output function under study. It is defined as follows:

$$k(\mathbf{x},\mathbf{x}') = \alpha\left(1+\sqrt{5}r+\frac{5}{3}r^2\right)\exp(-\sqrt{5}r), \tag{35}$$

$$r = \sqrt{\sum_{i=1}^{d}\frac{\left(x_i - x_i'\right)^2}{l_i^2}}, \tag{36}$$

where α is the kernel variance, and l_i is the kernel length scale for the ith dimension.

When training the GP, Maximum Likelihood Estimation (MLE) [69] is commonly used to estimate the hyperparameters $\theta := \{\alpha, l_1, \ldots, l_d\}$:

$$\hat{\theta} = \arg\max_{\theta} \log p(\boldsymbol{f} \mid X, \theta) \tag{37}$$

$$= \arg\max_{\theta} -\frac{1}{2}\left(\log|2\pi K_{\mathrm{x\,x}}| + \boldsymbol{f}^T K_{\mathrm{x\,x}}^{-1}\boldsymbol{f}\right). \tag{38}$$

In the MOBGO case, each expensive output function (objectives as well as constraints) is modeled using a distinct, *single-output* GP.

Expected improvement

In unconstrained single-objective optimization problems, one of the most popular acquisition functions is the Expected Improvement (EI) [1]. As evident from its name, it measures the *improvement* in the objective outcome that the analyst may expect when querying a new point $\mathbf{x}_*$, given the current best objective outcome obtained so far ($\hat{y}$) and the current GP model for the objective function (estimated on the currently available data $\mathcal{D}_n$). Given the GP model assumptions, the predictive outcome $f(\mathbf{x}_*)$ at such a new point is normally distributed ($N(\mu,\sigma^2)$, with μ the predictive mean at $\mathbf{x}_*$ and σ^2 the predictive variance). The improvement function at any arbitrary new point $\mathbf{x}_*$ is then given by the following random variable (without loss of generality, we assume here that we aim to *minimize* the objective function):

$$I(\mathbf{x}_*) := (\hat{y} - f(\mathbf{x}_*))\mathbb{I}(\hat{y} > f(\mathbf{x}_*)), \tag{39}$$

where $\mathbb{I}$ is the indicator function. The EI at $\mathbf{x}_*$ is given by the following closed-form expression:

$$\begin{aligned}\mathrm{EI}\left(\mathbf{x}_*;\mathcal{D}_n\right) &:= \mathbb{E}[I(\mathbf{x}_*)] \\ &= \sigma\phi\left(\frac{\hat{y}-\mu}{\sigma}\right) + (\hat{y}-\mu)\Phi\left(\frac{\hat{y}-\mu}{\sigma}\right).\end{aligned} \tag{40}$$

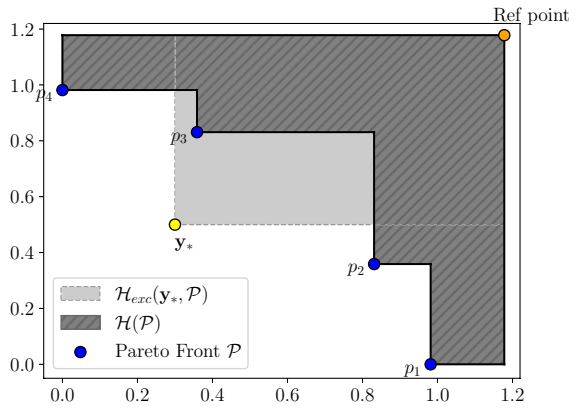


Fig. 7 Illustration of the hypervolume improvement (light grey area) of a new point $\mathbf{y}_*$ given the Pareto front $\mathcal{P}$ (colour figure online)

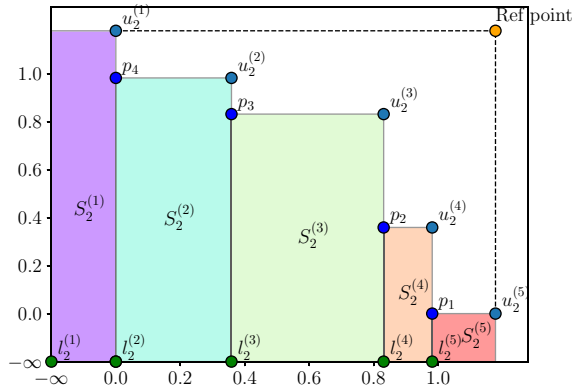


Fig. 8 Hyperbox decomposition on $\mathbb{R}^2$; each hyperbox is shown in a different color (colour figure online)

Hyperbox decomposition

The concept of hypervolume improvement ($\mathcal{H}_{exc}$) in $\mathbb{R}^2$ is illustrated in Fig. 7. To calculate $\mathcal{H}_{exc}$ efficiently (using piece-wise integration), the non-dominated space is partitioned into a set of hyper-boxes or hyper-cells (as few boxes/cells as possible).

Figure 8 illustrates this hyper-box decomposition in $\mathbb{R}^2$ for a Pareto front $\mathcal{P}$ consisting of 4 points. Note that each hyperbox $S_2^{(i)}$ ($i = 1, \cdots, 5$) in this figure can be represented by its lower-bound vector $\mathbf{l}_2^{(i)}$ and its upper bound vector $\mathbf{u}_2^{(i)}$ (both vectors are 2-dimensional in this case). In $\mathbb{R}^M$, the hyper-boxes are thus represented by:

$$S_M^{(i)} = \left(\mathbf{l}_M^{(i)}, \mathbf{u}_M^{(i)}\right) = \left(\left(l_1^{(i)}, \ldots, l_M^{(i)}\right)^\top, \left(u_1^{(i)}, \ldots, u_M^{(i)}\right)^\top\right) \quad \text{for} \quad i = 1, \ldots, N_M, \tag{41}$$

where N_M is the number of hyper-boxes.

Table 8 Hypervolume improvement with 95% confidence interval for the DTLZ1 function

M	d	Method	Budget	HVI	Δ to Ground truth (%)
2	4	Random	100	3.5801e5 ± 3.8568e2	0.55050 ± 0.17285
		EHVI	100	3.5941e5 ± 1.7502e2	0.16443 ± 0.07844
		CEHVI	100	**3.5995e5** ± 4.0028e1	**0.01462** ± 0.01794
		NSGA2	200	3.5973e5 ± 8.0121e1	0.07492 ± 0.03591
	6	Random	100	3.5127e5 ± 1.4763e3	2.42410 ± 0.66164
		EHVI	100	3.5694e5 ± 1.0928e3	0.85069 ± 0.48978
		CEHVI	100	**3.5976e5** ± 8.4896e1	**0.06634** ± 0.03804
		NSGA2	200	3.5780e5 ± 5.3252e2	0.61180 ± 0.23866
	8	Random	100	3.3646e5 ± 2.4519e3	6.53840 ± 1.09890
		EHVI	100	3.5194e5 ± 1.5345e3	2.23770 ± 0.68771
		CEHVI	100	**3.5931e5** ± 1.4983e2	**0.19246** ± 0.06715
		NSGA2	200	3.5162e5 ± 1.5136e3	2.32700 ± 0.67834
3	4	Random	100	2.1574e8 ± 6.3676e4	0.12257 ± 0.04756
		EHVI	100	2.1598e8 ± 1.1133e4	0.01097 ± 0.00832
		CEHVI	100	**2.16008** ± 8.4567e2	**0.00036** ± 0.00063
		NSGA2	200	2.1598e8 ± 8.8661e3	0.00863 ± 0.00662
	8	Random	100	2.1597e8 ± 3.4617e4	0.01509 ± 0.02586
		EHVI	100	2.1550e8 ± 2.3715e5	0.22961 ± 0.17714
		CEHVI	100	**2.1597e8** ± 3.4617e4	**0.01509** ± 0.02586
		NSGA2	200	2.1475e8 ± 2.0456e5	0.58044 ± 0.15279

Reference points of [600, 600] and [600, 600, 600] are used for $M = 2$ and $M = 3$ respectively

Higher HVI indicates better performance, lower the Δ to ground truth indicates better performance (in bold)

Table 9 Hypervolume improvement with 95% confidence interval of the DTLZ2

M	d	Method	Budget	HVI	Δ to Ground truth (%)
2	4	Random	100	0.8046e1 ± 0.0015e1	1.99680 ± 0.29509
		EHVI	100	0.8183e1 ± 0.0001e1	0.31922 ± 0.02189
		CEHVI	100	**0.8192e1** ± 0.0001e1	**0.21739** ± 0.02481
		NSGA2	200	0.8152e1 ± 0.0011e1	0.70169 ± 0.22646
	6	Random	100	0.7895e1 ± 0.0021e1	3.83750 ± 0.42409
		EHVI	100	0.8171e1 ± 0.0001e1	0.46986 ± 0.02671
		CEHVI	100	**0.8191e1** ± 0.0001e1	**0.22889** ± 0.02007
		NSGA2	200	0.8083e1 ± 0.0011e1	1.55390 ± 0.21220
	8	Random	100	0.7709e1 ± 0.0025e1	6.11000 ± 0.50078
		EHVI	100	0.8160e1 ± 0.0003e1	0.60907 ± 0.07758
		CEHVI	100	**0.8180e1** ± 0.0002e1	**0.36247** ± 0.04917
		NSGA2	200	0.7950e1 ± 0.0021e1	3.17340 ± 0.41099
3	4	Random	100	2.5836e1 ± 4.2026e2	2.23270 ± 0.25659
		EHVI	100	2.6318e1 ± 2.5326e2	0.40931 ± 0.15460
		CEHVI	100	**2.6345e1** ± 3.0863e2	**0.30718** ± 0.18843
		NSGA2	200	2.6188e1 ± 4.5699e2	0.89802 ± 0.27901
	8	Random	100	2.5252e1 ± 0.0072e1	4.43990 ± 0.43750
		EHVI	100	2.6251e1 ± 0.0063e1	0.66186 ± 0.38174
		CEHVI	100	**2.6260e1** ± 0.0114e1	**0.62720** ± 0.69741
		NSGA2	200	2.5836e1 ± 0.0081e1	2.23050 ± 0.49500

Reference points of [3, 3] and [3, 3, 3] are used for $M = 2$ and $M = 3$ respectively

Higher HVI indicates better performance, lower the Δ to ground truth indicates better performance (in bold)

Table 10 Hypervolume improvement with 95% confidence interval of the DTLZ3

M	d	Method	Budget	HVI	Δ to Ground truth (%)
2	4	Random	100	1.1989e6 ± 2.1987e3	0.91826 ± 0.29317
		EHVI	100	1.2068e6 ± 1.1462e3	0.26629 ± 0.15283
		CEHVI	100	**1.2098e6** ± 2.3727e2	**0.01397**± 0.03163
		NSGA2	200	1.2082e6 ± 4.9900e2	0.15086 ± 0.06654
	6	Random	100	1.1611e6 ± 9.0820e3	4.04120 ± 1.21100
		EHVI	100	1.1952e6 ± 4.9118e3	1.22010 ± 0.65493
		CEHVI	100	**1.2090e6** ± 3.1147e2	**0.08470**± 0.04153
		NSGA2	200	1.2005e6 ± 3.2024e3	0.78801 ± 0.42700
	8	Random	100	1.0779e6 ± 1.4485e4	10.91600 ± 1.93140
		EHVI	100	1.1688e6 ± 1.2016e4	3.40890 ± 1.60220
		CEHVI	100	**1.2061e6** ± 1.0296e3	**0.32177**± 0.13728
		NSGA2	200	1.1701e6 ± 7.6555e3	3.30100 ± 1.02080
3	4	Random	100	1.3275e9 ± 8.7369e5	0.26579 ± 0.10591
		EHVI	100	1.3306e9 ± 5.7116e5	0.03340 ± 0.06923
		CEHVI	100	**1.3310e9** ± 3.6666e3	**0.00018**± 0.00044
		NSGA2	200	1.3308e9 ± 1.0310e5	0.01452 ± 0.01250
	8	Random	100	1.2705e9 ± 7.1915e6	4.54870 ± 0.87174
		EHVI	100	1.3254e9 ± 1.8606e6	0.42361 ± 0.22554
		CEHVI	100	**1.3306e9** ± 2.3480e5	**0.02847**± 0.02846
		NSGA2	200	1.3090e9 ± 4.2413e6	1.65450 ± 0.51412

Reference points of [1100, 1100] and [1100, 1100, 1100] are used for $M = 2$ and $M = 3$ respectively

Higher HVI indicates better performance, lower the Δ to ground truth indicates better performance (in bold)

Different box-partition algorithms have been presented in the literature, see for instance [25, 26, 33, 70, 71]. In this paper, we use the box-partition algorithm from [25]. This is without loss of generality since the proposed algorithm in this paper is compatible with *any* box-partition algorithm.

Appendix 3 DTLZ results with varying input and output dimensions

To assess the performance of our proposed method under varying input and output dimensions, we conducted an evaluation of the DTLZ functions with combinations of input dimensions of [4, 6, 8] and output dimensions of [2, 3]. All of the scenarios assume 1 cheap objective. The obtained results for DTLZ1, DTLZ2, and DTLZ3 are presented in Tables 8, 9, and 10, respectively. The hypervolume values presented in the tables demonstrate the superiority of the proposed method in comparison to other competing methods.

Acknowledgements This work has been supported by the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" and the "Fonds Wetenschappelijk Onderzoek (FWO)" programmes. We also thank our colleague Jixiang Qing for providing detailed comments that helped improve the paper.

Data availability Not applicable.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. čkus, J (1974) On bayesian methods for seeking the extremum. In: Marchuk GI (eds) Optimization Techniques IFIP Technical Conference Novosibirsk, 1–7, pp. 400–404. Springer, Berlin, Heidelberg (1975)
2. Mockus JB, Mockus LJ (1991) Bayesian approach to global optimization and application to multiobjective and constrained problems. J Optim Theory Appl 70(1):157–172
3. Garnett R (2022) Bayesian Optimization. Cambridge University Press, Cambridge (**in preparation**)
4. Shahriari B, Swersky K, Wang Z, Adams RP, De Freitas N (2016) Taking the human out of the loop: a review of Bayesian optimization. Proc IEEE. https://doi.org/10.1109/JPROC.2015.2494218
5. Jim TM, Faza GA, Palar PS, Shimoyama K (2021) Bayesian optimization of a low-boom supersonic wing planform. AIAA J 59(11):4514–4529

6. Snoek J, Larochelle H, Adams RP (2012) Practical Bayesian optimization of machine learning algorithms. Adv Neural Inf Process Syst 4:2951–2959 arXiv:1206.2944
7. De Almeida AT, Ferreira FJTE, Duarte AQ (2014) Technical and economical considerations on super high-efficiency three-phase motors. IEEE Trans Ind Appl 50(2):1274–1285. https://doi.org/10.1109/TIA.2013.2272548
8. Lei G, Zhu J, Guo Y, Liu C, Ma B (2017) A Review of Design Optimization Methods for Electrical Machines. Energies 10(12):1962. https://doi.org/10.3390/en10121962
9. Makni Z, Besbes M, Marchand C (2007) Multiphysics design methodology of permanent-magnet synchronous motors. IEEE Trans Veh Technol 56(4):1524–1530. https://doi.org/10.1109/TVT.2007.896981
10. Ibrahim MN, Sergeant P (2022) Design and analysis of electric motor with integrated magnetic spring for cyclic loads. IEEE Transact Industrial Electron. https://doi.org/10.1109/TIE.2022.3210513
11. Liu X, Hu C, Li X, Gao J, Huang S (2021) An Online Data-Driven Multi-Objective Optimization of a Permanent Magnet Linear Synchronous Motor. IEEE Transact Magnetics 57(7):1–4. https://doi.org/10.1109/TMAG.2021.3059513
12. Diao K, Sun X, Lei G, Guo Y, Zhu J (2020) Multiobjective System Level Optimization Method for Switched Reluctance Motor Drive Systems Using Finite-Element Model. IEEE Transact Industrial Electron 67(12):10055–10064. https://doi.org/10.1109/TIE.2019.2962483
13. Rasmussen CE, Williams CKI (2018) Gaussian processes for machine learning. The MIT Press, Cambridge. https://doi.org/10.7551/mitpress/3206.001.0001
14. Williams CKI, Rasmussen CE (2006) Gaussian processes for machine learning, vol 2, issue no. 3. MIT Press, Cambridge, MA
15. Wiener N (1938) The homogeneous chaos. Am J Math 60(4):897–936
16. Kaintura A, Dhaene T, Spina D (2018) Review of Polynomial Chaos-Based Methods for Uncertainty Quantification in Modern Integrated Circuits. Electronics 7(3):30. https://doi.org/10.3390/electronics7030030
17. Sun G, Wang S (2019) A review of the artificial neural network surrogate modeling in aerodynamic design. Proc Institut Mech Eng Part G J Aerospace Eng 233(16):5863–5872. https://doi.org/10.1177/0954410019864485
18. Snoek J, Rippel O, Swersky K, Kiros R, Satish N, Sundaram N, Patwary MMA, Prabhat P, Adams RP (2015) Scalable Bayesian optimization using deep neural networks. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37. ICML'15, pp. 2171–2180. JMLR.org
19. Bergstra J, Bardenet R, Bengio Y, Kégl B (2011) Algorithms for hyper-parameter optimization. Adva Neural Inform Proces Syst 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011, 1–9
20. Ginsbourger D, Le Riche R, Carraro L (2008) A Multi-points criterion for deterministic parallel global optimization based on Gaussian processes. Département Méthodes et Modèles Mathématiques pour l'Industrie, 3MIENSMSE, Saint-Étienne, France, Tech. Rep. hal-00260579
21. Contal E, Buffoni D, Robicquet A, Vayatis N (2013) Parallel gaussian process optimization with upper confidence bound and pure exploration. In: Blockeel H, Kersting K, Nijssen S, Zelezný F (eds) ECML/PKDD (1). Lecture Notes in Computer Science, vol. 8188, pp. 225–240. Springer, New York. https://doi.org/10.1007/978-3-642-40988-2_15. https://www.dblp.uni-trier.de/db/conf/pkdd/pkdd2013-1.html. Accessed 13 Oct 2023
22. Kandasamy K, Dasarathy G, Oliva JB, Schneider J, Póczos B (2016) Gaussian process bandit optimisation with multi-fidelity evaluations. Adv Neural Inform Proces Syst **29**
23. Gardner JR, Kusner MJ, Xu ZE, Weinberger KQ, Cunningham JP (2014) Bayesian optimization with inequality constraints. In ICML 2014:937–945
24. Letham B, Karrer B, Ottoni G, Bakshy E (2019) Constrained Bayesian Optimization with Noisy Experiments. Bayesian Anal 14(2):495–519. https://doi.org/10.1214/18-BA1110
25. Couckuyt I, Deschrijver D, Dhaene T (2014) Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. J Global Optim 60(3):575–594. https://doi.org/10.1007/s10898-013-0118-2
26. Emmerich MTM, Deutz AH, Klinkenberg JW (2011) Hypervolume-based expected improvement: Monotonicity properties and exact computation. In: 2011 IEEE Congress of Evolutionary Computation (CEC), pp. 2147–2154. IEEE, New Orleans, LA, USA . https://doi.org/10.1109/CEC.2011.5949880. http://www.ieeexplore.ieee.org/document/5949880/ Accessed 26 Sep 2022
27. Wang X, Jin Y, Schmitt S, Olhofer M (2022) Recent Advances in Bayesian Optimization. arXiv . https://doi.org/10.48550/ARXIV.2206.03301. arXiv:2206.03301
28. Allmendinger R, Handl J, Knowles J (2015) Multiobjective optimization: When objectives exhibit non-uniform latencies. Eur J Oper Res 243(2):497–513. https://doi.org/10.1016/j.ejor.2014.09.033
29. Wang X, Jin Y, Schmitt S, Olhofer M (2021) Transfer Learning Based Co-surrogate Assisted Evolutionary Bi-objective Optimization for Objectives with Non-uniform Evaluation Times. arXiv. arXiv:2108.13339 [cs].arXiv:2108.13339 Accessed 23 Sep 2022
30. Wang X, Jin Y, Schmitt S, Olhofer M, Allmendinger R (2021) Transfer learning based surrogate assisted evolutionary bi-objective optimization for objectives with different evaluation times. Knowl-Based Syst 227:107190. https://doi.org/10.1016/j.knosys.2021.107190
31. Wang X, Jin Y, Schmitt S, Olhofer M (2022) Alleviating Search Bias in Bayesian Evolutionary Optimization with Many Heterogeneous Objectives. arXiv. arXiv:2208.12217 [cs].arXiv:2208.12217 Accessed 23 Sep 2022
32. Loka N, Couckuyt I, Garbuglia F, Spina D, Van Nieuwenhuyse I, Dhaene T (2022) Bi-objective Bayesian optimization of engineering problems with cheap and expensive cost functions. Engineering with Computers. https://doi.org/10.1007/s00366-021-01573-7
33. Yang K, Emmerich M, Deutz A, Bäck T (2019) Efficient computation of expected hypervolume improvement using box decomposition algorithms. J Global Optim 75(1):3–34. https://doi.org/10.1007/s10898-019-00798-7
34. Diehl O, Schönfeldt M, Brouwer E, Dirks A, Rachut K, Gassmann J, Güth K, Buckow A, Gauß R, Stauber R, Gutfleisch O (2018) Towards an Alloy Recycling of Nd-Fe-B Permanent Magnets in a Circular Economy. Journal of Sustainable Metallurgy 4(2):163–175. https://doi.org/10.1007/s40831-018-0171-7
35. Elwert T, Goldmann D, Roemer F, Schwarz S (2017) Recycling of NdFeB Magnets from Electric Drive Motors of (Hybrid) Electric Vehicles. Journal of Sustainable Metallurgy 3(1):108–121. https://doi.org/10.1007/s40831-016-0085-1
36. Luckeneder S, Giljum S, Schaffartzik A, Maus V, Tost M (2021) Surge in global metal mining threatens vulnerable ecosystems. Glob Environ Chang 69:102303. https://doi.org/10.1016/j.gloenvcha.2021.102303
37. Meeker D (2010) Finite element method magnetics. FEMM 4.32:162

38. Wang J, Yuan X, Atallah K (2013) Design optimization of a surface-mounted permanent-magnet motor with concentrated windings for electric vehicle applications. IEEE Trans Veh Technol 62(3):1053–1064. https://doi.org/10.1109/TVT.2012.2227867
39. Metwly MY, Hemeida A, Abdel-Khalik AS, Hamad MS, Ahmed S (2021) Design and multi-objective optimization of a 12-slot/10-pole integrated obc using magnetic equivalent circuit approach. Machines. https://doi.org/10.3390/machines9120329
40. Edhah SO, Alsawalhi JY, Al-Durra AA (2019) Multi-objective optimization design of fractional slot concentrated winding permanent magnet synchronous machines. IEEE Access 7:162874–162882
41. Silva RCP, Rahman T, Mohammadi MH, Lowther DA (2018) Multiple operating points based optimization: Application to fractional slot concentrated winding electric motors. IEEE Trans Industr Electron 65(2):1719–1727. https://doi.org/10.1109/TIE.2017.2756586
42. Sarigiannidis AG, Beniakar ME, Kladas AG (2016) Fast adaptive evolutionary pm traction motor optimization based on electric vehicle drive cycle. IEEE Trans Veh Technol 66(7):5762–5774
43. Görtler J, Kehlbeck R, Deussen O (2019) A visual exploration of gaussian processes. Distill. https://doi.org/10.23915/distill.00017.https://www.distill.pub/2019/visual-exploration-gaussian-processes
44. Fonseca CM, Paquete L, Lopez-Ibanez M (2006) An improved dimension-sweep algorithm for the hypervolume indicator. In: 2006 IEEE International Conference on Evolutionary Computation, pp. 1157–1163. https://doi.org/10.1109/CEC.2006.1688440
45. Sheikh HM, Marcus PS (2022) Bayesian optimization for multi-objective mixed-variable problems. arXiv preprint arXiv:2201.12767
46. Daulton S, Balandat M, Bakshy E (2020) Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization. Adv Neural Inf Process Syst 33:9851–9864
47. Daulton S, Eriksson D, Balandat M, Bakshy E (2022) Multi-objective bayesian optimization over high-dimensional search spaces. In: Uncertainty in Artificial Intelligence, pp. 507–517. PMLR
48. Allmendinger R, Knowles J (2021) Heterogeneous Objectives: State-of-the-Art and Future Research. arXiv. https://doi.org/10.48550/ARXIV.2103.15546 arXiv:2103.15546
49. Buckingham JM, Gonzalez SR, Branke J (2023) Bayesian optimization of multiple objectives with different latencies. arXiv preprint. arXiv:2302.01310
50. Martínez-Frutos J, Herrero-Pérez D (2016) Kriging-based infill sampling criterion for constraint handling in multi-objective optimization. J Global Optim 64(1):97–115. https://doi.org/10.1007/s10898-015-0370-8
51. Viana FAC, Venter G, Balabanov V (2010) An algorithm for fast optimal latin hypercube design of experiments. Int J Numer Meth Eng. https://doi.org/10.1002/nme.2750
52. Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms—a comparative case study. In: Eiben AE, Bäck T, Schoenauer M, Schwefel H-P (eds) Parallel Problem Solving from Nature – PPSN V. Springer, Berlin, Heidelberg, pp 292–301
53. Guerreiro AP, Fonseca CM, Paquete L (2021) The Hypervolume Indicator: Problems and Algorithms. ACM Comput Surveys 54(6): 1–42. https://doi.org/10.1145/3453474 arXiv:2005.00515 [cs]. Accessed 27 Sep 2022
54. Emmerich M, Yang K, Deutz A, Wang H, Fonseca C (2015) A multicriteria generalization of bayesian global optimization. Adv Stoch Determ Global Optim. https://doi.org/10.1007/978-3-319-29975-4_12
55. Muth JF, Thompson GL (1963) Industrial Scheduling. Prentice-Hall, Michigan
56. Martí R, Resende MGC, Ribeiro CC (2013) Multi-start methods for combinatorial optimization. Eur J Oper Res 226(1):1–8. https://doi.org/10.1016/j.ejor.2012.10.012
57. Kraft D (1988) A Software Package for Sequential Quadratic Programming. Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt Köln: Forschungsbericht. Wiss. Berichtswesen d. DFVLR, Germany. https://www.books.google.be/books?id=4rKaGwAACAAJ
58. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, van der Walt SJ, Brett M, Wilson J, Millman KJ, Mayorov N, Nelson ARJ, Jones E, Kern R, Larson E, Carey CJ, Polat İ, Feng Y, Moore EW, VanderPlas J, Laxalde D, Perktold J, Cimrman R, Henriksen I, Quintero EA, Harris CR, Archibald AM, Ribeiro AH, Pedregosa F, van Mulbregt P (2020) SciPy 1.0 Contributors: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nat Methods 17:261–272. https://doi.org/10.1038/s41592-019-0686-2
59. Picheny V, Berkeley J, Moss HB, Stojic H, Granta U, Ober SW, Artemev A, Ghani K, Goodall A, Paleyes A, Vakili S, Pascual-Diaz S, Markou S, Qing J, Loka NRBS, Couckuyt I (2023) Trieste: Efficiently Exploring The Depths of Black-box Functions with TensorFlow. https://doi.org/10.48550/ARXIV.2302.08436 arXiv:2302.08436
60. Deb K, Thiele L, Laumanns M, Zitzler E (2005) Scalable Test Problems for Evolutionary Multiobjective Optimization. In: Abraham A, Jain L, Goldberg R (eds) Evolutionary Multiobjective Optimization, pp. 105–145. Springer, London. https://doi.org/10.1007/1-84628-137-7_6 (Series Title: Advanced Information and Knowledge Processing)
61. Binh TT, Korn U (1997) Mobes: A multiobjective evolution strategy for constrained optimization problems. In: Proceedings of the Third International Conference on Genetic Algorithms, MENDEL97, pp. 176–182
62. Chankong V, Haimes YY (2008) Multiobjective Decision Making: Theory and Methodology. Dover Books on Engineering. Dover Publications, New York. https://www.books.google.be/books?id=o371DAAAQBAJ
63. Owen AB (2017) A randomized halton algorithm in R. arXiv preprint. arXiv:1706.02808
64. Durillo JJ, Nebro AJ, Luna F, Alba E (2010) On the effect of the steady-state selection scheme in multi-objective genetic algorithms. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 5467 LNCS, 183–197. https://doi.org/10.1007/978-3-642-01020-0_18
65. Blank J, Deb K (2020) pymoo: Multi-objective optimization in python. IEEE Access 8:89497–89509
66. Ibrahim MN, Rezk H, Al-Dhaifallah M, Sergeant P (2020) Modelling and design methodology of an improved performance photovoltaic pumping system employing ferrite magnet synchronous reluctance motors. Mathematics. https://doi.org/10.3390/MATH8091429
67. Ma Q, El-Refaie A, Lequesne B (2020) Low-cost interior permanent magnet machine with multiple magnet types. IEEE Trans Ind Appl 56(2):1452–1463. https://doi.org/10.1109/TIA.2020.2966458
68. Minasny B, McBratney AB (2005) The Matérn function as a general model for soil variograms. Geoderma 128(3–4):192–207. https://doi.org/10.1016/j.geoderma.2005.04.003
69. Norden RH (1972) A survey of maximum likelihood estimation. International Statistical Review / Revue Internationale de Statistique 40(3):329–354
70. Emmerich M, Yang K, Deutz A, Wang H, Fonseca CM (2016) A multicriteria generalization of bayesian global optimization. In:

Pardalos PM, Zhigljavsky A, Žilinskas J (eds) Advances in Stochastic and Deterministic Global Optimization. Springer Optimization and Its Applications, pp. 229–242. Springer, Switzerland. https://doi.org/10.1007/978-3-319-29975-4

71. Hupkens I, Deutz A, Yang K, Emmerich M (2015) Faster Exact Algorithms for Computing Expected Hypervolume Improvement. In: Gaspar-Cunha A, Henggeler Antunes C, Coello CC (eds) Evolutionary Multi-Criterion Optimization vol. 9019, pp. 65–79. Springer, Cham. https://doi.org/10.1007/978-3-319-15892-1_5.Series Title: Lecture Notes in Computer Science