Bayesian Optimization For Choice Data

Alessio Benavoli
School of Computer Science and Statistics, Trinity College Dublin
Dublin, Ireland
alessio.benavoli@tcd.ie

Dario Azzimonti
Dario Piga
Dalle Molle Institute for Artificial Intelligence (IDSIA), USI/SUPSI
Lugano, Switzerland
{dario.azzimonti,dario.piga}@idsia.ch

ABSTRACT

In this work we introduce a new framework for multi-objective Bayesian optimisation where the multi-objective functions can only be accessed via choice judgements, such as "I pick options $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ among this set of five options $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_5$". The fact that the option $\mathbf{x}_4$ is rejected means that there is at least one option among the selected ones $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ that I strictly prefer over $\mathbf{x}_4$ (but I do not have to specify which one). We assume that there is a latent vector function $\mathbf{u}$ for some dimension d which embeds the options into the real vector space of dimension d, so that the choice set can be represented through a Pareto set of non-dominated options. By placing a Gaussian process prior on $\mathbf{u}$ and by using a novel likelihood model for choice data, we derive a surrogate model for the latent vector function. We then propose two novel acquisition functions to solve the multi-objective Bayesian optimisation from choice data.

CCS CONCEPTS

• **Theory of computation** → **Gaussian processes**; **Active learning**;
• **Mathematics of computing** → *Mathematical optimization*.

KEYWORDS

multi-objective optimization, Bayesian optimization, choice learning

ACM Reference Format:
Alessio Benavoli, Dario Azzimonti, and Dario Piga. 2023. Bayesian Optimization For Choice Data. In *Genetic and Evolutionary Computation Conference Companion (GECCO '23 Companion), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3583133.3596324

1 INTRODUCTION

Real-world optimization problems often involve multiple conflicting objectives that can be technically difficult and costly to quantitatively evaluate. Consider for instance the problem of finding the best recipe for a new cake. To achieve our objective of creating the most delicious, soft, and visually appealing cake possible, we must carefully determine the optimal ingredients and their quantities, such as the precise amount of flour, butter, sugar, and other necessary components. This problem can be formulated as a Bayesian Optimization (BO) task, as the function involved is expensive to evaluate due to the time-consuming process of preparing a cake, which may take hours. Since it is technically difficult and costly to quantitatively measure taste/softness/appearance of a cake, a cheaper and fast alternative is to collect preference data from an Individual (we call her Alice), "cake c_1 is better than cake c_2", and use preference Bayesian Optimisation (BO) [6, 15, 30] to optimise the recipe. However, due to the presence of competing objectives, Alice may not be able to express a preference between cakes, because for instance cake c_1 can be more tasty than c_2, c_2 can be softer than c_1, c_3 can be more visually appealing than c_2. Therefore, Alice may not have a preference between c_1, c_2, c_3 and, if we ask Alice to express a preference, this may lead to inconsistency.

In order to handle situations where objects cannot be compared, it is necessary to use a choice model that enables Alice to make set-based choices. Choice functions offer a mathematical framework for this purpose. For any given set of objects A, they return the corresponding set-valued choice $C(A)$:

$A =$ 

$C(A) =$ 

The statement that an object (a cake) in A is rejected (that is, it is not in $C(A)$) means that there is at least one object in A that Alice strictly prefers over it. Instead, any two objects in $C(A)$ are deemed to be incomparable by Alice.

In this paper, we develop a novel Bayesian optimization algorithm for multi-objective optimisation based on implicit feedback expressed via choice statements.

First, we represent each object (cake in the example) by the feature vector $\mathbf{x} \in \mathbb{R}^{n_x}$ of its characteristics (e.g., amount of butter, sugar, etc.) and assume an implicit feedback expressed via choice data $\{(C(A_s), A_s),\ s = 1, \ldots, m\}$, for example

$$A_s = \{\mathbf{x}_1^{(s)}, \mathbf{x}_2^{(s)}, \mathbf{x}_3^{(s)}, \mathbf{x}_4^{(s)}, \mathbf{x}_5^{(s)}\}$$
$$C(A_s) = \{\mathbf{x}_1^{(s)}, \mathbf{x}_2^{(s)}, \mathbf{x}_4^{(s)}\}.$$

Second, we use a recent model to learn Choice functions from choice data proposed by [8]. This model uses Gaussian Processes (GP) to learn a d-dimensional latent utility vector function, which embeds each feature vector $\mathbf{x}$ into $\mathbb{R}^d$. Through these utilities, each choice set $C(A_s)$ is then represented as a Pareto set of non-dominated options. Third, based on this model, we develop a Choice-based Bayesian multi-objective optimization algorithm (ChoiceBO). In particular, we propose and numerically compare two acquisition functions for ChoiceBO. We assess our framework on a number of multi-objective

GECCO '23 Companion, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0120-7/23/07.
https://doi.org/10.1145/3583133.3596324

benchmark problems assuming the objective functions can only be evaluated through choice-statements. We also compare ChoiceBO against an oracle that knows the true value of the latent functions. We show that, by only working with choice data, we can obtain good performance results in these benchmarks.

2 BACKGROUND

The paper leverages three topics: (1) Bayesian Optimisation (BO); (2) multi-objective BO; (3) choice functions. In this section we briefly review the state of the art of each topic.

2.1 Bayesian Optimisation (BO)

BO [18] aims to find the global maximum of an unknown function which is expensive to evaluate. For a scalar real-valued function g on a domain $\Omega \subset \mathbb{R}^{n_x}$, the goal is to find a global maximiser $\mathbf{x}^o = \arg\max_{\mathbf{x}\in\Omega} g(\mathbf{x})$. BO formulates this as a sequential decision problem [14]: a trade-off between learning about the underlying function g (exploration) and capitalizing on this information in order to find the optimum $\mathbf{x}^o$ (exploitation). BO relies on a probabilistic surrogate model, usually a Gaussian Process (GP) [28], to provide a posterior distribution over g given a dataset $\mathcal{D} = \{(\mathbf{x}_i, g(\mathbf{x}_i)) : i = 1, 2, \ldots, N\}$ of previous evaluations of g. It then employs an *acquisition function* (e.g. Expected Improvement [18, 22], Upper Credible Bound [31]) to select the next candidate option (solution) $\mathbf{x}_{N+1}$. While the true function g is expensive-to-evaluate, the surrogate-based acquisition function is not, and it can thus be efficiently optimized to compute an optimal candidate to be evaluated on g. This process is repeated sequentially until some stopping criterion is achieved.

2.2 Multi-objective (MO) optimization

The goal of MO optimization is to identify the set of *Pareto optimal* options (solutions) such that any improvement in one objective means deteriorating another. Without loss of generality, we assume the goal is to maximize all objectives. Let $\mathbf{g}(\mathbf{x}) : \Omega \to \mathbb{R}^{n_o}$ be a vector-value objective function with $\mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), \ldots, g_{n_o}(\mathbf{x})]^\top$, where n_o is the number of objectives. We recall the notions of Pareto dominated options and non-dominated set.

DEFINITION 1 (PARETO DOMINATE OPTION). *Consider a set of options $\mathcal{X} \subset \Omega$. An option $\mathbf{x}_1 \in \mathcal{X}$ is said to Pareto dominate another option $\mathbf{x}_2 \in \mathcal{X}$, denoted as $\mathbf{x}_1 \succ \mathbf{x}_2$, if both the following conditions are true:*

(1) for all $j \in \{1, 2, \ldots, n_o\}$, $g_j(\mathbf{x}_1) \geq g_j(\mathbf{x}_2)$;
(2) $\exists\, j \in \{1, 2, \ldots, n_o\}$, such that $g_j(\mathbf{x}_1) > g_j(\mathbf{x}_2)$.

DEFINITION 2 (NON-DOMINATED SET). *Among a set of options $A = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$, the non-dominated set of options A' are those that are not dominated by any member of A, i.e.*

$$A' = \{\mathbf{x} \in A : \nexists \mathbf{x}' \in A \text{ such that } \mathbf{x}' \succ \mathbf{x}\}.$$

Given the set of options $\mathcal{X}$, MO aims to find the non-dominated set of options $\mathcal{X}^{nd}$, called the *Pareto set*. The set of evaluations $\mathbf{g}(\mathcal{X}^{nd})$ is called *Pareto front*.

MO BO have been developed for standard cases where multi-objectives can directly be evaluated (they are not implicit). Many approaches rely on scalarisation to transform the MO problem into a single-objective one, like ParEGO [20] and TS-TCH [24] (which randomly scalarize the objectives and use Expected Improvement and, respectively, Thompson Sampling). [19] derived an *expected improvement* criterion with respect to multiple objectives. [27] proposed an *hyper-volume based infill* criterion, where the improvements are measured in terms of hyper-volume (of the Pareto front) increase. Other acquisition functions have been proposed in [4, 13, 16, 26, 33]. The most used acquisition function for MO BO is *expected hyper-volume improvement*. In fact, maximizing the hyper-volume has been shown to produce very accurate estimates [10, 12, 17, 35, 36, 38] of the Pareto front.

2.3 Choice functions

Individuals are often confronted with the situation of choosing between several options (alternatives). These alternatives can be goods that are going to be purchased, candidates in elections, food etc.

We model options, that an agent has to choose, as real-valued vectors $\mathbf{x} \in \mathbb{R}^{n_x}$ and identify the sets of options as finite subsets of $\mathbb{R}^{n_x}$. Let $\mathcal{Q}$ denote the set of all such finite subsets of $\mathbb{R}^{n_x}$.

DEFINITION 3. *A choice function C is a set-valued operator on sets of options. More precisely, it is a map $C : \mathcal{Q} \to \mathcal{Q}$ such that, for any set of options $A \in \mathcal{Q}$, the corresponding value of C is a subset $C(A)$ of A (see for instance [1]).*

The more general interpretation of choice function is as follows. For a given option set $A \in \mathcal{Q}$, the statement that an option $\mathbf{x}_j \in A$ is rejected from A (that is, $\mathbf{x}_j \notin C(A)$) means that there is at least one option $\mathbf{x}_i \in A$ that an agent strictly prefers over $\mathbf{x}_j$. The set of rejected options is denoted by $R(A)$ and is equal to $A \backslash C(A)$. Therefore choice functions represent non-binary choice models, so they are more general than preferences.
It is important to stress again that the statement $\mathbf{x}_j \notin C(A)$ implies there is at least one option $\mathbf{x}_i \in A$ that an agent strictly prefers over $\mathbf{x}_j$. However, the agent is not required to tell us which option(s) in $C(A)$ they strictly prefer to $\mathbf{x}_j$. This makes choice functions a very easy-to-use tool to express choices.

In this paper, we follow an interpretation of choice functions where the set $C(A)$ is seen as the *non-dominated set* in the Pareto sense for some latent function. In other words, let us assume that there is a latent utility vector function $\mathbf{u}(\mathbf{x}_i) = [u_1(\mathbf{x}_i), \ldots, u_d(\mathbf{x}_i)]^\top$, for some dimension d, which embeds the options $\mathbf{x}_i$ into a space $\mathbb{R}^d$. The choice set can then be represented through a Pareto set of non-dominated options. For example, in the cake example, $d = 3$ because we used three criteria to assess quality: taste, softness and appearance. This Pareto-based representation was originally proposed in [25] to learn choice functions using Neural Networks. It was then reformulated in a fully probabilistic way (with a different likelihood) using Gaussian Processes in [8]. This latter approach is more accurate and offers a valuable measure of uncertainty in the estimated utility vector. This uncertainty representation can be used to balance the trade-off between exploration and exploitation in BO. We will discuss in detail this model in the next section.

It is worth noticing that when $d = 1$ and the dimension of A is two, choice-functions reduce to binary-preference modelling (with a single utility function that represent the preferences). In preference-learning, the state-of-the-art surrogate model is based on a method proposed in [9]. This method assumes that there is an unobservable

latent utility function $u(\mathbf{x}_i)$ associated with each training sample $\mathbf{x}_i$, and that the function values $\{u(\mathbf{x}_i) : i = 1, 2, \ldots, N\}$ preserve the preference relations observed in the dataset, that is $u(\mathbf{x}_i) \geq u(\mathbf{x}_j)$ whenever $\mathbf{x}_i$ "better than" $\mathbf{x}_j$. A framework for preference-based BO (PBO) was proposed by [30] and a new acquisition function, inspired by Thomson sampling, was proposed in [15]. More recently, [5, 7] showed that the posterior of GP preference learning is a Skew GP [5]. Based on this exact model, the authors derived a PBO framework based on SkewGPs [6]. To the best of our knowledge, no one has yet developed an implementation for choice-based BO.

3 BAYESIAN LEARNING OF CHOICE FUNCTIONS

In this section, we review the GP-based choice-function learning framework proposed in [8]. We consider objects $\mathbf{x} \in \mathbb{R}^{n_x}$ and, for $\mathbf{x} \in \mathbb{R}^{n_x}$, we model each latent function in the vector $\mathbf{u}(\mathbf{x}) = [u_1(\mathbf{x}), \ldots, u_d(\mathbf{x})]^\top$ as an independent GP [28]:

$$u_j(\mathbf{x}) \sim \mathrm{GP}_j(0, k_j(\mathbf{x}, \mathbf{x}')), \; j = 1, 2, \ldots, d, \tag{1}$$

where the dimension d represent the number of latent objectives that are assumed by the GP prior. Each GP is fully specified by its kernel function $k_j(\cdot, \cdot)$, which specifies the covariance of the latent function between any two points. In all experiments in this paper, the GP kernel is Matérn ($\nu = 3/2$) [28], however the method works for any choice of a valid positive definite kernel function. Having defined the prior on $\mathbf{u}$, we now describe the likelihood.

For each A, $C(A)$ is interpreted as the *undominated set* in the *strong Pareto sense*. $R(A)$ is the set of dominated objects. We assume that there is a latent vector function $\mathbf{u}(\mathbf{x}) = [u_1(\mathbf{x}), \ldots, u_d(\mathbf{x})]^\top$, for some finite dimension d, which embeds the objects $\mathbf{x}$ into a space $\mathbb{R}^d$. The choice set is represented via a Pareto set of strongly undominated objects:

$$\neg \left(\min_{i \in \{1,\ldots,d\}} (u_i(\mathbf{o}) - u_i(\mathbf{v})) < 0, \; \forall \mathbf{o} \in C(A) \right), \forall \mathbf{v} \in R(A), \tag{2}$$

$$\min_{i \in \{1,\ldots,d\}} (u_i(\mathbf{o}) - u_i(\mathbf{v})) < 0, \; \forall \mathbf{o}, \mathbf{v} \in C(A), \; \mathbf{o} \neq \mathbf{v}. \tag{3}$$

Condition (3) means that, for each object in $C(A)$, there is no better object in $C(A)$. Condition (2) means that, for each object $\mathbf{v} \in R(A)$, it is not true ($\neg$ is the logical negation) that all objects in $C(A)$ are worse than $\mathbf{v}$, that is there is at least an object in $C(A)$ which is not worse than $\mathbf{v}$. Therefore, these conditions require that the latent functions values of the objects should be consistent with the choice function implied relations.

To account for errors in Alice's choices, we follow [8] and replace the hard-constraints (2),(3) with soft-constraints. Consider the vectors $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t]^\top$ with $\mathbf{x} \in \mathcal{X}$,

$$\mathbf{u}(\mathbf{x}_i) = [u_1(\mathbf{x}_i), u_2(\mathbf{x}_i), \ldots, u_d(\mathbf{x}_i)],$$
$$\mathbf{u}(X) = [\mathbf{u}(\mathbf{x}_1), \mathbf{u}(\mathbf{x}_2), \ldots, \mathbf{u}(\mathbf{x}_t)]^\top,$$

and the choice dataset

$$\mathcal{D}_m = \{(C(A_s), A_s) : \text{ for } s = 1, \ldots, m\},$$

where $A_s \subset X$ for each $s = 1, \ldots, m$. The likelihood is defined as

$$\begin{aligned} p(\mathcal{D}_m|\mathbf{u}(X)) &= \prod_{k=1}^{m} p(C(A_k), A_k|\mathbf{u}(X)) \\ &= \prod_{k=1}^{m} \prod_{\{\mathbf{o},\mathbf{v}\} \in C_2(A_k)} \left(1 - \prod_{i=1}^{d} \Phi\left(u_i(\mathbf{o}) - u_i(\mathbf{v})\right) \right. \\ &\quad \left. - \prod_{i=1}^{d} \Phi\left(u_i(\mathbf{v}) - u_i(\mathbf{o})\right) \right) \\ &\quad \prod_{\mathbf{v} \in R(A_k)} \left(1 - \prod_{\mathbf{o} \in C(A_k)} \left(1 - \prod_{i=1}^{d} \Phi\left(u_i(\mathbf{o}) - u_i(\mathbf{v})\right) \right) \right) \end{aligned} \tag{4}$$

where $\Phi(\cdot)$ is the Cumulative Distribution Function (CDF) of the standard Normal distribution. The notation $\{\mathbf{o}, \mathbf{v}\} \in C_2(A_k)$ means that the pair $\{\mathbf{o}, \mathbf{v}\}$ is an element of $C_2(A_k)$, which is the set of all possible 2-combination (without repetition) of the elements of the set $C(A_k)$. The product in the last row in (4) is a probabilistic relaxation of (2). The product in the first and second row in (4) is a probabilistic relaxation of (3).

REMARK 4. *For $d = 1$ (that is, the latent dimension is one) and binary options sets, that is $|A_k| = 2$ for each k, the likelihood* (4)*, for a given k, simplifies to*

$$p(C(A_k), A_k|u) = \Phi\left(u(\mathbf{x}_i) - u(\mathbf{x}_j)\right), \tag{5}$$

which has extensively been used in Preferential BO. Therefore, we can consider the likelihood (4) *its extension to the non-binary and multi-utility case.*

Computation of the posterior. The posterior density of $\mathbf{u}(X)$ is

$$p(\mathbf{u}(X)|\mathcal{D}_m) = \frac{p(\mathbf{u}(X))}{p(\mathcal{D}_m)} \prod_{k=1}^{m} p(C(A_k), A_k|\mathbf{u}(X)), \tag{6}$$

where the prior over the component of $\mathbf{u}$ is defined in (1), the likelihood is defined in (4) and the probability of the evidence is $p(\mathcal{D}_m) = \int p(\mathcal{D}_m|\mathbf{u}(X))p(\mathbf{u}(X))d\mathbf{u}(X)$. The posterior $p(\mathbf{u}(X)|\mathcal{D}_m)$ is intractable therefore we follow [8] and use variational inference to learn at the same time the hyper-parameters (lengthscales) $\boldsymbol{\theta}$ of the kernels and a Gaussian approximation of the posterior $p(\mathbf{u}(X)|\mathcal{D}_m)$.[1]

The algorithm of variational inference learns the density $q(\mathbf{u}(X))$, an approximation of the posterior $p(\mathbf{u}(X)|\mathcal{D}_m)$, by minimizing the Kullback–Leibler divergence of q from the posterior. In practice (see [23] for a comparison) this is achieved by maximizing an evidence lower bound

$$ELBO = \underbrace{\int q(\mathbf{u}(X)) \log p(\mathcal{D}_m|\mathbf{u}(X)) d\mathbf{u}}_{\text{likelihood term}} - \underbrace{KL[q(\mathbf{u}(X))||p(\mathbf{u}(X))]}_{\text{KL between priors}}$$

Here we approximate the likelihood term with Monte Carlo integration. Moreover we use an approximation $q(\mathbf{u}(X))$ with a full covariance matrix.

[1] We implemented our model in PyMC3 [29].

Prediction and Inferences. Let $X^* = \{\mathbf{x}_1^*, \dots, \mathbf{x}_p^*\}$ be a set including p test points and $\mathbf{u}(X^*) = [\mathbf{u}(\mathbf{x}_1^*), \dots, \mathbf{u}(\mathbf{x}_p^*)]^\top$. Under the GP prior assumption on u, the conditional predictive distribution $p(\mathbf{u}(X^*)|\mathbf{u}(X))$ is Gaussian and, therefore,

$$p(\mathbf{u}(X^*)|\mathcal{D}_m) = \int p(\mathbf{u}(X^*)|\mathbf{u}(X))p(\mathbf{u}(X)|\mathcal{D}_m)d\mathbf{u}(X) \quad (7)$$

can be easily computed analytically by using the VI Gaussian posterior $p(\mathbf{u}|\mathcal{D}_m)$. In choice-based BO, we are interested in the inference:

$$P(C(A^*), A^*|\mathcal{D}_m) = \int p(C(A^*), A^*|\mathbf{u}(X^*)) \, p(\mathbf{u}(X^*)|\mathcal{D}_m)d\mathbf{u}(X^*), \quad (8)$$

which returns the posterior probability that Alice chooses the objects $C(A^*)$ from the set of objects A^*. This probability can be computed via Monte Carlo sampling from the approximate posterior $p(\mathbf{u}(X^*)|\mathcal{D}_m)$, which is Gaussian.

EXAMPLE 1. *We present a simple toy example to show how this model works. We consider the bi-dimensional vector function* $\mathbf{g}(x) = [e^{-(x+1)^2}, e^{-(x-1)^2}]^\top$ *with* $x \in [-3, 3]$, *see blue and orange function in Figure 1(top). We use* $\mathbf{g}$ *to generate choice data. For instance, consider the set of options* $A_k = \{-0.8, -1.65, -2.39\}$, *given that*

$$\mathbf{g}(-0.8) = [0.039, 0.961]$$
$$\mathbf{g}(-1.65) = [0.001, 0.655]$$
$$\mathbf{g}(-2.39) = [0, 0.145]$$

we have $C(A_k) = \{-0.8\}$ *and* $R(A_k) = A_k \backslash C(A_k) = \{-1.65, -2.39\}$. *In fact, one can notice that* -0.8 *dominates* $-1.65, -2.39$ *on both the objectives. We sample* 50 *inputs* x_i *in a uniform grid in* $[-3, 3]$ *and, using the above approach, we generate* $m = 20$ *random subsets* $\{A_k\}_{k=1}^m$ *of the 50 points each one of size* $|A_k| = 3$. *The location of these points is shown at the bottom of Figure 1(top). To represent the 20 batches of input triplets included in the sets* $\{A_k\}_{k=1}^{20}$, *we utilized different colors. Figure 1(bottom) displays the corresponding values of the two objectives at these points, revealing the Pareto front (we assume we aim to maximise the objectives).*

Fixing the latent dimension $d = 2$, *we will use the previously described model to infer the latent functions from the choice datasets. We compute the posterior means (lines) and* 95% *credible intervals (shaded region) of the latent functions, as shown in Figure 2.*

By examining this plot, it can be noticed that the shape of the learned utility functions (mean) resembles the original utilities, but there are notable differences. This occurs because from choice data, we are only able to deduce the inferred objective functions, with the exception of a monotonic transformation. Only Pareto dominance is maintained during the learning process. Indeed, it can be observed that to the left of the crossing point, the orange function lies above the blue function, while to the right, the opposite holds true. This guarantees that the learned functions represent the Pareto dominance implied by the choice data. As we will see in the next sections, the non-full-identifiability (due to this invariance to monotonic transformations), that is inherent in the utility functions implied by choice data, impacts the efficacy of some of the acquisition functions we utilize for Bayesian optimization.

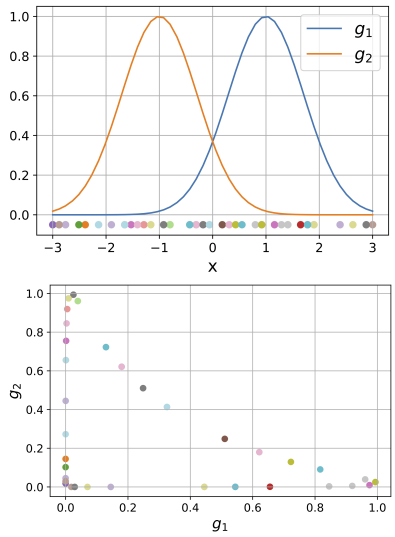

Figure 1: Objectives (top) and Pareto front (below). Colors represent the different batches of input triplets.

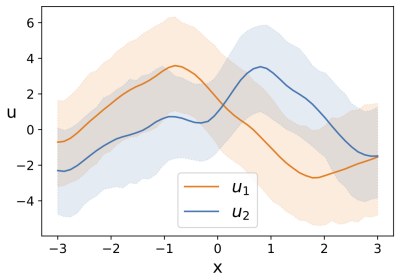

Figure 2: Posterior latent utilities estimated from choice data

4 CHOICE-BASED BAYESIAN OPTIMISATION

In the previous sections, we have introduced a GP-based model to learn latent choice functions from choice data. We will now focus on the acquisition component of Bayesian optimization. In choice-based BO, we never observe the actual values of the functions. The data is $(X, \{C(A_k), A_k\}_{k=1}^m)$, where X is the set of the m training inputs (options), A_k is a subset of X and $C(A_k) \subseteq A_k$ is the choice-set for the given options A_k. We denote the Pareto-set estimated using the GP-based model as $\hat{\mathcal{X}}^{nd}$. In choice-based BO, the objective is to seek a new input point $\mathbf{x}$. Since $\mathbf{g}$ can only be queried via a choice function, this is obtained by optimizing an acquisition function $\alpha(\mathbf{x}, \hat{\mathcal{X}}^{nd})$ w.r.t. $\mathbf{x}$, where $\hat{\mathcal{X}}^{nd}$ is the current (estimated) Pareto-set. We define acquisition functions of the form $\alpha(\mathbf{x}, \hat{\mathcal{X}}^{nd})$, with the aim to find a point that dominates the points in $\hat{\mathcal{X}}^{nd}$. That is, given the set of options $A^* = \mathbf{x} \cup \hat{\mathcal{X}}^{nd}$, we aim to find $\mathbf{x}$ such that $C(A^*) = \{\mathbf{x}\}$.

The acquisition function must also consider the trade-off between exploration and exploitation. Therefore, we propose here two acquisition functions: choiceUCB and choiceThompson.

4.1 choiceUCB

The acquisition function choiceUCB, denoted by $\alpha_{UCB}(\mathbf{x}, \hat{\mathcal{X}}^{nd})$, is equal to the $\gamma\%$ (in the experiments we use $\gamma = 95$) Upper Credible Bound (UCB) of $p(C(A^*), A^*|\mathbf{u}^*)$ with $\mathbf{u}^* \sim p(\mathbf{u}^*|\mathcal{D}_m)$, $A^* = \mathbf{x} \cup \hat{\mathcal{X}}^{nd}$ and $C(A^*) = \{\mathbf{x}\}$.

Note that the requirement for our acquisition function is strong. We could also define $\tilde{\alpha}(\mathbf{x}, \hat{\mathcal{X}}^{nd})$ with different objectives in mind. For example we could seek to find a point $\mathbf{x}$ which allows to reject at least one option in $\hat{\mathcal{X}}^{nd}$. We opted for UCB of $p(C(A^*), A^*|\mathbf{u}^*)$ because it leads to a fast to evaluate acquisition function. In particular we only need to compute one probability for each new function evaluation.

4.2 choiceThompson

We also propose an alternative acquisition function built on the idea of Thompson sampling [32]. We define the acquisition function choiceThompson $\alpha_{Thom}(\mathbf{x}, \mathcal{D}_m)$ as the increase in hyper-volume of the Pareto front for a posterior realization when the Pareto front is calculated by evaluating the posterior utility at $\mathbf{x}$. We can compute this value by (i) generating a posterior realization of the latent utilities at the training inputs X, $\tilde{\mathbf{u}}(X) \sim p(\mathbf{u}(X)|\mathcal{D}_m)$; (ii) computing the Pareto set for this specific realization, denoted by $\tilde{\mathcal{X}}^{nd}$; (iii) evaluate the same realization on $X \cup \{\mathbf{x}\}$, obtaining $\tilde{\mathbf{u}}(X \cup \{\mathbf{x}\})$; (iv) compute the Pareto set implied by $\tilde{\mathbf{u}}(X \cup \{\mathbf{x}\})$, denoted by $\tilde{\mathcal{X}}^{nd}_{\mathbf{x}}$; (v) the value of $\alpha_{Thom}(\mathbf{x}, \mathcal{D}_m)$ is the difference in hyper-volume between the Pareto fronts $\tilde{\mathbf{u}}(\tilde{\mathcal{X}}^{nd}_{\mathbf{x}})$ and $\tilde{\mathbf{u}}(\tilde{\mathcal{X}}^{nd})$.

Note that the posterior realization $\tilde{\mathbf{u}}$ in steps (ii) and (iv) has to be the same otherwise the resulting Pareto sets are not comparable. Usually this is achieved by selecting, a priori, a large set of candidate points, generating a posterior realization on this set and then evaluating $\alpha_{Thom}(\mathbf{x}, \mathcal{D}_m)$ only on those candidate points. Here instead we follow [2] and build a predictive process for each posterior realization. We initially select a large set of points where the posterior GP is initially sampled. We use those samples to train one anchor GP for each realization. The trained anchor GPs provide a fast to evaluate predictive posterior which is a surrogate for the original GP realization. This allows for a continuous optimization of the acquisition function and which is not limited to a fixed number of candidate points. An alternative approach could exploit the efficient posterior samples introduced in [34].

EXAMPLE 2. *We continue the previous example to show the point that maximise the two acquisitions functions proposed previously. The point that maximised the UCB acquisition function is shown in Figure 3-bottom. It is interesting to notice how the point $x_{ucb} = -0.49$ is on the Pareto front and, in particular, in a region where there are no observations. So in this example this acquisition function correctly balances exploitation-exploration. A similar comment holds for the Thompson-based acquisition. Figure 3-top shows two functions sampled from the posterior which are then used as surrogate objectives to maximised the hyper-volume of the current Pareto front. Note again that the shape is very different from the original Gaussian functions, but the crossing and dominance are preserved. This leads to the new point $x_{th} = -0.11$. This point changes when we sample new functions, Figure 3-center, leading to $x_{th} = -0.55$. However, both the selected new points are on the Pareto front Figure 3-bottom.*

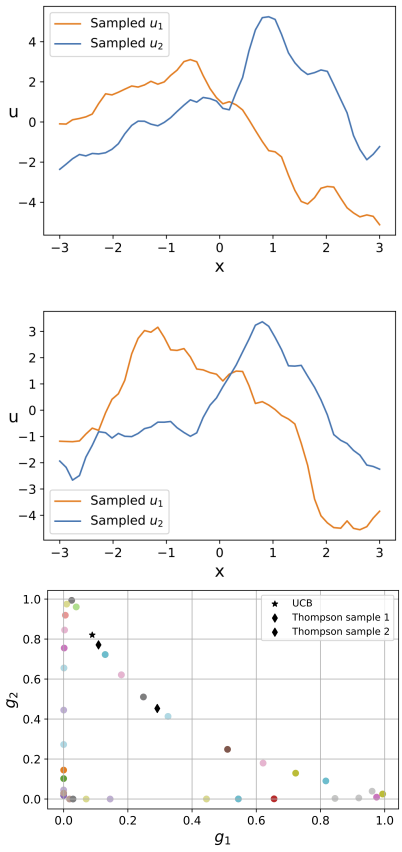

Figure 3: Sampled utilities (top, center) and chosen x_{next} (bottom).

4.3 Query the agent for choices

After computing the maximum of the the acquisition function, denoted with $\mathbf{x}_{new}$, consistently with the definition of the acquisition function, we should query the agent to express their choice among the set of options in $A^* = \mathbf{x}_{new} \cup \hat{\mathcal{X}}^{nd}$. Here we assume that the costly element of our pipeline is the production of an element and not the act of choosing between elements. In the example of section 1, the costly element of comparing three cakes is baking the cakes and not the act of choosing between already made cakes. For this reason, once we have a new set of features $\mathbf{x}_{new}$, we produce the item associated with those features and then we ask the user to produce choice sets $C(A^*_j)$ from option sets A^*_j of fixed size $|A^*_j|$ made with 5 combinations of $\mathbf{x}_{new}$ with $|A^*_j| - 1$ elements from $\hat{\mathcal{X}}^{nd}$. In the experiments we fix $|A^*_j| = 3$.

4.4 Connection with priori and posteriori methods

Multi-objective optimization can be categorized into three types based on the level of engagement of the decision maker (DM): a priori, a posteriori, and interactive approaches (see for instance [21]). A priori approaches require the DM to specify their preferences

before optimization begins. This transforms the problem into a single objective optimization, but it can be challenging for the DM to express their preferences without seeing the alternatives. A posteriori approaches, which are more common, attempt to identify a good approximation of the Pareto frontier, and the DM can then choose their preferred solution from this set. However, finding the entire Pareto front can be computationally expensive. Interactive approaches try to learn the DM's preferences during optimization and focus the search on the most preferred area of the Pareto front. This often involves comparing pairs of solutions and assuming a scalarisation utility function model (e.g., the Tchebychev utility).

Our proposed method is an interactive approach. It aims to efficiently identify a good approximation of the Pareto frontier when the DM's implicit criteria align with the true objectives. However, when the DM's implicit criteria may be different from the true objectives, our method learns the DM's choice function during optimization. In this case, the BO search is focused on the most preferred area of the Pareto front based on the latent utility functions held by the DM.

Consider the example of baking a cake, where the DM (Alice) may use a choice function based on taste, softness, and appearance as the objectives. However, Alice's choice function may also take into account combinations of these criteria reducing to 2D-objectives or even nD-objectives. Indeed, Alice's choice function may be complex and multifaceted, but the ChoiceBO is designed to work without requiring her to explicitly state her decision criteria.

In the next section, we make the simplifying assumption that we know the number of criteria that Alice is considering, but we do not assume that we know the specific form of her utility functions but we learn them based on her choices.

5 NUMERICAL EXPERIMENTS

In this section we assume $d = n_o$ (that is we assume that the latent dimension is known) and evaluate the performance of our algorithm on the use of choice functions in multi-objective BO. See [8] for experimental results on recovering the latent dimensions and on the accuracy of learning choice functions.

We consider an underlying oracle $\mathbf{g}(\mathbf{x})$ from six standard multi-objective benchmark functions: Kursawe ($n_x = 3$, $n_o = 2$), ZDT2 ($n_x = 3$, $n_o = 2$), ZDT1 ($n_x = 4$, $n_o = 2$), DTLZ1 ($n_x = 4$, $n_o = 2$), DTLZ1 ($n_x = 4$, $n_o = 3$), and Vehicle-Safety[2] ($n_x = 5$, $n_o = 3$). These are minimization problems, which we converted into maximizations so that the acquisition function in Section 4 is well-defined. We compare the Choice-GP BO (with $d = n_o$) approach proposed in this paper against ParEGO.[3] For ParEGO, we assume the algorithm can query directly $\mathbf{g}(\mathbf{x})$ and, therefore, we refer to it as Oracle-ParEGO. Note that ParEGO uses a random Tchebyshev scalarisation to scalarise the objectives and then performs standard BO. This approach can only be applied if the values of $\mathbf{g}(\mathbf{x})$ are known (which is necessary in order to define a scalarisation). [4] Conversely, Choice-GP BO can only query $\mathbf{g}(\mathbf{x})$ via choice functions. A parameter that needs to be chosen in advance is $|A_k|$, the size of the choice set. It was observed in [8] that increasing the size of the choice set $|A_k|$ could lead to better estimates, however a human is known to be able to compare well only up to 5 objects. In our experiments we fixed $|A_k| = 3$ as this is a reasonable number for a user. We use both choiceUCB and choiceThompson as acquisition functions and we also consider a quasi-random baseline that selects candidates from a Sobol sequence denoted as "Sobol".

We evaluate optimization performance on the six benchmark problems in terms of log-hyper-volume difference, which is defined as the difference between the hyper-volume of the true Pareto front[5] and the hyper-volume of the approximate Pareto front based on the observed data $\mathcal{X}$. Each experiment starts with 20 initial (randomly selected) input points which are used to initialise Oracle-ParEGO. We generate 10 pairs $\{C(A_k), A_k\}$ of size $|A_k| = 3$ by randomly selecting 10 subsets A_k of these 20 points. These choices $\{C(A_k), A_k\}_{k=1}^{10}$ are used to initialise Choice-GP BO. A total budget of 100 iterations are run for both the algorithms. Further, each experiment is repeated 20 times with different initialization. In these experiments we optimize the kernel hyperparameters by maximising the marginal likelihood for Oracle-ParEGO and its variational approximation for Choice-GP.

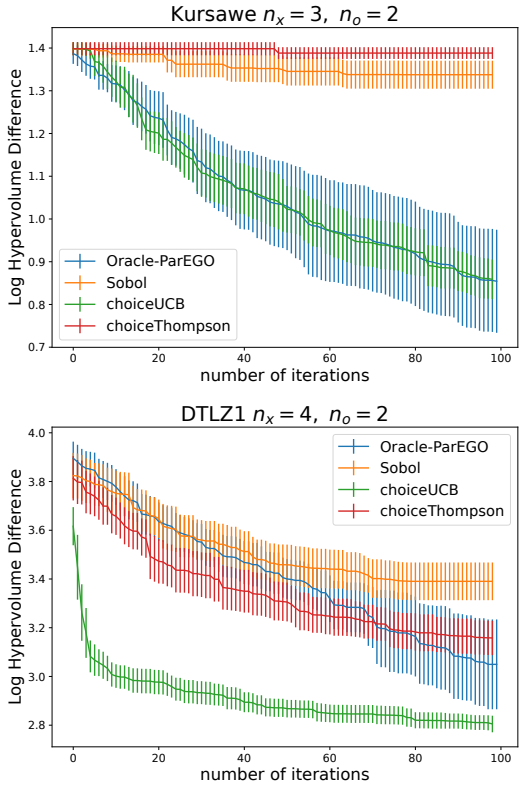

Figure 4: Kursawe and DTLZ1 benchmark functions ($n_o = 2$).

Figures 4, 5, 6 show the performance of the proposed acquisition functions versus the performance of the Oracle and a random method. Note that all benchmarks in figure 4 and 5 have two objectives. We notice that choiceUCB performs much better than choiceThompson

[2]The problem of determining the thickness of five reinforced components of a vehicle's frontal frame [37]. This problem was previously considered as benchmark in [11].
[3]We use the BoTorch implementation [3].
[4]The most recent MO BO approaches mentioned in Section 1 outperform ParEGO. We use ParEGO only as an Oracle reference.
[5]This known for the six benchmarks.

on all benchmarks, both in terms of speed of convergence and smallest log-hyper-volume difference. In most benchmarks choiceThompson tends to get stuck in local optima and does not provide sufficient exploration. On the other hand, choiceUCB is also able to outperform Oracle-ParEGO, in most benchmarks, even if choiceUCB only has access to choice sets. We believe that this performance is motivated by the multiple comparisons we are able to do for each new proposed point. As outlined in section 4.3, for each new proposed point we ask the user to provide five additional choice sets. This does not come at an additional cost for the user, however it provides a much better posterior distribution over the choices. The effect is reflected in exploitation but also in the exploration properties of choiceUCB. In all benchmarks and all repetitions, choiceUCB does not tend to get stuck in local optima.

Focusing on DTLZ1, Kursawe (Figure 4), and DTLZ1 and VehicleSafety (Figure 6), it can be noticed how choiceUCB convergences to the performance of the Oracle-ParEGO at the increase of the number of iterations. The overall performance shows that the proposed approach is very effective. In ZDT1 and ZDT2, Figure 5, choiceUCB outperforms Oracle-ParEGO. The bad performance of Oracle-ParEGO is due to the used acquisition function, which does not correctly balance exploitation-exploration in these two benchmarks. Instead, the UCB acquisition function for choiceUCB works well in all the benchmarks.

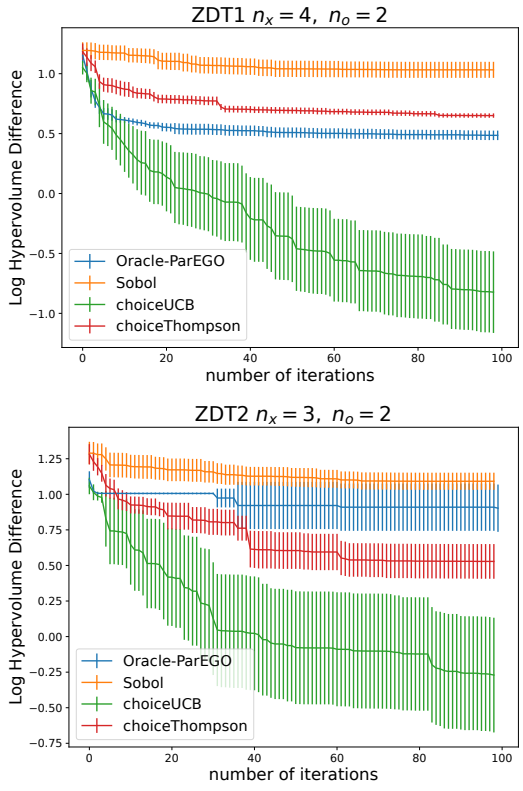

Figure 5: ZDT1 and ZDT2 benchmark functions ($n_o = 2$).

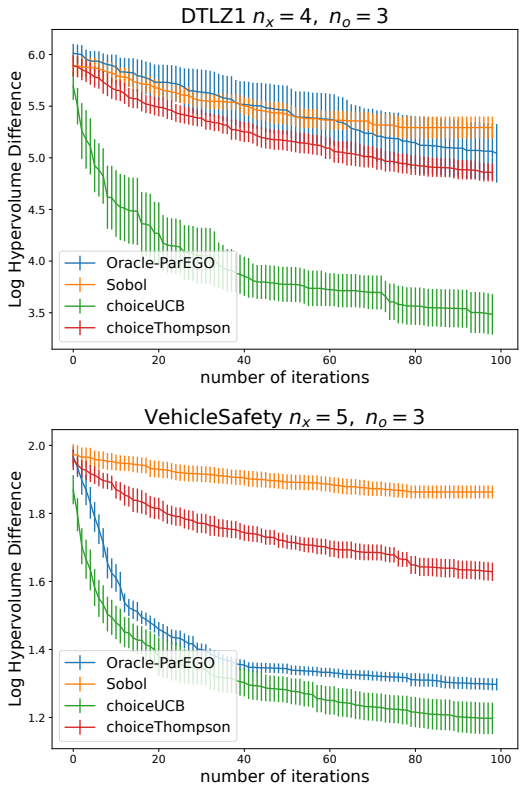

Figure 6: DTLZ1 and Vehicle safety benchmarks ($n_o = 3$).

6 CONCLUSIONS

We have developed a Bayesian method to learn choice functions from data and applied to choice function based Bayesian Optimisation (BO). As future work, we plan to develop strategies to speed up the learning process by exploring more efficient ways to express the likelihood. We also intend to explore different acquisition functions for choice function BO.

ACKNOWLEDGEMENTS

For the first author, this publication has emanated from research conducted with the financial support of the EU Commission Recovery and Resilience Facility under the Science Foundation Ireland Future Digital Challenge Grant Number 22/NCF/FD/10827.

REFERENCES

[1] Fuad Aleskerov, Denis Bouyssou, and Bernard Monjardet. 2007. *Utility maximization, choice and preference*. Vol. 16. Springer Science & Business Media, Berlin.

[2] D. Azzimonti, J. Bect, C. Chevalier, and D. Ginsbourger. 2016. Quantifying Uncertainties on Excursion Sets Under a Gaussian Random Field Prior. *SIAM/ASA Journal on Uncertainty Quantification* 4, 1 (2016), 850–874. https://doi.org/10.1137/141000749

[3] Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. 2020. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 21524–21538. https://proceedings.neurips.cc/paper_files/paper/2020/file/f5b1b89d98b7286673128a5fb112cb9a-Paper.pdf

[4] Syrine Belakaria, Aryan Deshwal, and Janardhan Rao Doppa. 2019. Max-value Entropy Search for Multi-Objective Bayesian Optimization. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2019/file/82edc5c9e21035674d481640448049f3-Paper.pdf

[5] Alessio Benavoli, Dario Azzimonti, and Dario Piga. 2020. Skew Gaussian Processes for Classification. *Machine Learning* 109 (2020), 1877–1902.

[6] Alessio Benavoli, Dario Azzimonti, and Dario Piga. 2021. Preferential Bayesian optimisation with skew Gaussian processes. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, Lille France, 1842–1850. https://doi.org/10.1145/3449726.3463128

[7] Alessio Benavoli, Dario Azzimonti, and Dario Piga. 2021. A unified framework for closed-form nonparametric regression, classification, preference and mixed problems with Skew Gaussian Processes. *Machine Learning* 110, 11–12 (Dec 2021), 3095–3133. https://doi.org/10.1007/s10994-021-06039-x

[8] Alessio Benavoli, Dario Azzimonti, and Dario Piga. 2023. Learning Choice Functions with Gaussian Processes. https://doi.org/10.48550/ARXIV.2302.00406

[9] Wei Chu and Zoubin Ghahramani. 2005. Preference Learning with Gaussian Processes. In *Proceedings of the 22nd International Conference on Machine Learning* (Bonn, Germany) *(ICML '05)*. Association for Computing Machinery, New York, NY, USA, 137–144. https://doi.org/10.1145/1102351.1102369

[10] Ivo Couckuyt, Dirk Deschrijver, and Tom Dhaene. 2014. Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. *Journal of Global Optimization* 60, 3 (2014), 575–594.

[11] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. 2020. Differentiable Expected Hypervolume Improvement for Parallel Multi-Objective Bayesian Optimization. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 9851–9864. https://proceedings.neurips.cc/paper_files/paper/2020/file/6fec24eac8f18ed793f5eaad3dd7977c-Paper.pdf

[12] Michael Emmerich, Kaifeng Yang, André Deutz, Hao Wang, and Carlos M Fonseca. 2016. A multicriteria generalization of bayesian global optimization. In *Advances in Stochastic and Deterministic Global Optimization*. Springer, 229–242.

[13] Michael T. M. Emmerich, Andre H. Deutz, and Jan Willem Klinkenberg. 2011. Hypervolume-based expected improvement: Monotonicity properties and exact computation. In *2011 IEEE Congress of Evolutionary Computation (CEC)*. IEEE, New Orleans, LA, USA, 2147–2154. https://doi.org/10.1109/CEC.2011.5949880

[14] Peter I. Frazier. 2018. A Tutorial on Bayesian Optimization. arXiv:1807.02811 [stat.ML]

[15] Javier González, Zhenwen Dai, Andreas Damianou, and Neil D. Lawrence. 2017. Preferential Bayesian Optimization. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 1282–1291. https://proceedings.mlr.press/v70/gonzalez17a.html

[16] Daniel Hernández-Lobato, José Miguel Hernández-Lobato, Amar Shah, and Ryan P Adams. 2016. Predictive Entropy Search for Multi-objective Bayesian Optimization. In *Proceedings of The 33rd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 48)*. PMLR, New York, New York, USA, 1492–1501.

[17] Iris Hupkens, André Deutz, Kaifeng Yang, and Michael Emmerich. 2015. *Faster Exact Algorithms for Computing Expected Hypervolume Improvement*. Lecture Notes in Computer Science, Vol. 9019. Springer International Publishing, Cham, 65–79. https://doi.org/10.1007/978-3-319-15892-1_5

[18] Donald R Jones, Matthias Schonlau, and William J Welch. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global optimization* 13, 4 (1998), 455–492.

[19] Andy J Keane. 2006. Statistical improvement criteria for use in multiobjective design optimization. *AIAA journal* 44, 4 (2006), 879–891.

[20] Joshua Knowles. 2006. ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation* 10, 1 (2006), 50–66.

[21] Kaisa Miettinen. 1998. *Nonlinear Multiobjective Optimization*. International Series in Operations Research & Management Science, Vol. 12. Springer US, Boston, MA. https://doi.org/10.1007/978-1-4615-5563-6

[22] Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. 1978. The application of Bayesian methods for seeking the extremum. *Towards global optimization* 2, 117-129 (1978), 2.

[23] Hannes Nickisch and Carl Edward Rasmussen. 2008. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research* 9 (2008), 2035–2078. Citation Key: Nickisch2008.

[24] Biswajit Paria, Kirthevasan Kandasamy, and Barnabás Póczos. 2020. A Flexible Framework for Multi-Objective Bayesian Optimization using Random Scalarizations. In *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference (Proceedings of Machine Learning Research, Vol. 115)*, Ryan P. Adams and Vibhav Gogate (Eds.). PMLR, 766–776. https://proceedings.mlr.press/v115/paria20a.html

[25] Karlson Pfannschmidt and Eyke Hüllermeier. 2020. *Learning Choice Functions via Pareto-Embeddings*. Lecture Notes in Computer Science, Vol. 12325. Springer International Publishing, Cham, 327–333. https://doi.org/10.1007/978-3-030-58285-2_30

[26] Victor Picheny. 2015. Multiobjective optimization using Gaussian process emulators via stepwise uncertainty reduction. *Statistics and Computing* 25, 6 (2015), 1265–1280.

[27] Wolfgang Ponweiser, Tobias Wagner, Dirk Biermann, and Markus Vincze. 2008. *Multiobjective Optimization on a Limited Budget of Evaluations Using Model-Assisted $\mathcal{S}$ -Metric Selection*. Lecture Notes in Computer Science, Vol. 5199. Springer Berlin Heidelberg, Berlin, Heidelberg, 784–794. https://doi.org/10.1007/978-3-540-87700-4_78

[28] Carl Edward Rasmussen and Christopher KI Williams. 2006. *Gaussian processes for machine learning*. MIT press Cambridge, MA.

[29] John Salvatier, Thomas V Wiecki, and Christopher Fonnesbeck. 2016. Probabilistic programming in Python using PyMC3. *PeerJ Computer Science* 2 (2016), e55.

[30] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. 2015. Taking the human out of the loop: A review of Bayesian optimization. *Proc. IEEE* 104, 1 (2015), 148–175.

[31] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. 2009. Gaussian process optimization in the bandit setting: No regret and experimental design.

[32] William R. Thompson. 1933. On the Likelihood that One Unknown Probability Exceeds Another in View of the Evidence of Two Samples. *Biometrika* 25, 3/4 (Dec 1933), 285. https://doi.org/10.2307/2332286

[33] Takashi Wada and Hideitsu Hino. 2019. Bayesian Optimization for Multi-objective Optimization and Multi-point Search. arXiv:1905.02370 [stat.ML]

[34] James Wilson, Viacheslav Borovitskiy, Alexander Terenin, Peter Mostowsky, and Marc Deisenroth. 2020. Efficiently sampling functions from Gaussian process posteriors. In *Proceedings of the 37th International Conference on Machine Learning*. PMLR, 10292–10302. https://proceedings.mlr.press/v119/wilson20a.html

[35] Kaifeng Yang, Michael Emmerich, André Deutz, and Thomas Bäck. 2019. Multi-objective Bayesian global optimization using expected hypervolume improvement gradient. *Swarm and evolutionary computation* 44 (2019), 945–956.

[36] Kaifeng Yang, Michael Emmerich, André Deutz, and Carlos M. Fonseca. 2017. *Computing 3-D Expected Hypervolume Improvement and Related Integrals in Asymptotically Optimal Time*. Lecture Notes in Computer Science, Vol. 10173. Springer International Publishing, Cham, 685–700. https://doi.org/10.1007/978-3-319-54157-0_46

[37] R. J. Yang, N. Wang, C. H. Tho, J. P. Bobineau, and B. P. Wang. 2005. Metamodeling Development for Vehicle Frontal Impact Simulation. *Journal of Mechanical Design* 127, 5 (Sep 2005), 1014–1020. https://doi.org/10.1115/1.1906264

[38] Eckart Zitzler, Lothar Thiele, Marco Laumanns, Carlos M Fonseca, and Viviane Grunert Da Fonseca. 2003. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation* 7, 2 (2003), 117–132.