Single Interaction Multi-Objective Bayesian Optimization

Juan Ungredda[1], Juergen Branke[1], Mariapia Marchi[2], and Teresa Montrone[2]

[1] University of Warwick, Coventry, UK
J.Ungredda@warwick.ac.uk, juergen.branke@wbs.ac.uk
[2] ESTECO SpA, Trieste, Italy, marchi@esteco.com, montrone@esteco.com

Abstract. When the decision maker (DM) has unknown preferences, the standard approach to a multi-objective problem is to generate an approximation of the Pareto front and let the DM choose from the non-dominated designs. However, if the evaluation budget is very limited, the true best solution according to the DM's preferences is unlikely to be among the small set of non-dominated solutions found. We address this issue with a multi-objective Bayesian optimization algorithm and allowing the DM to select solutions from a *predicted* Pareto front, instead of the final population. This allows the algorithm to understand the DM's preferences and make a final attempt to identify a more preferred solution that will then be returned without further interaction. We show empirically that significantly better solutions can be found in terms of true DM's utility than if the DM would pick a solution at the end.

Keywords: Preference Elicitation · Simulation Optimization · Gaussian Processes · Bayesian Optimization.

1 Introduction

Many real-world optimization problems have multiple, conflicting objectives. A popular way to tackle such problems is to search for a set of Pareto-optimal solutions with different trade-offs, and allow the decision maker (DM) to pick their most preferred solution from this set. This has the advantage that the DM doesn't have to specify their preferences explicitly before the optimization, which is generally considered very difficult.

In case of expensive multi-objective optimization problems, where the number of solutions that can be evaluated during optimization is small, the Pareto front, which may consist of thousands of Pareto-optimal solutions or even be continuous, can only be approximated by a small set of solutions. It is thus unlikely that the solution most preferred by the DM would be among the small set of solutions found by the optimization algorithm, even if these are truly Pareto-optimal solutions.

We suggest tackling this issue by using Bayesian Optimization (BO), a surrogate-based global optimization technique, and letting the DM choose a solution from a *predicted* Pareto front rather than from the identified non-dominated solutions at the end of the run. BO is not only known to be very suitable for expensive optimization as it carefully selects points to evaluate through an acquisition function that explicitly balances exploration and exploitation. It also generates a surrogate model of each objective function. These surrogate models can be optimized by a multi-objective evolutionary algorithm to generate an approximated Pareto front, and as evaluation of the surrogate model is cheap relative to a fitness evaluation, we can generate a fine-granular representation of the approximated Pareto front, consisting of very many solution candidates. This approximated Pareto front with many hypothetical solutions can then be shown to a DM to select from. While we cannot guarantee that the picked solution is actually achievable, the location of the picked solution should still give us a very good idea about the DM's preferences. Essentially, it provides a reference point which we expect to be quite close to what should be achievable. We then continue to run BO for a few more steps, aiming to generate the desired solution or something better.

We believe that the cognitive burden for the DM is not much higher than in standard multi-objective optimization: rather than having to identify the most preferred solution from a discrete approximation of the Pareto front at the end of the run, they now pick the most preferred out of the predicted (larger) set of solutions, but the size of the set presented to the DM should not make a big difference in terms of cognitive effort if the problem has only 2 (perhaps 3) objectives, where the interesting region can be identified easily by inspecting the Pareto front visually. After the final optimization step, the algorithm has to return a single recommended solution based on the elicited preference information. Of course it would be possible to ask the DM again to choose from all non-dominated solutions found, and this would probably further enhance the quality of the identified solution. However, we deliberately limit the preference elicitation in this paper to a *single* interaction.

Compared to the existing literature, we offer the following contributions:

- We are the first to question the common practice of returning the non-dominated solutions *after* optimization.
- Instead, we demonstrate that it is beneficial to let the DM choose from an approximated Pareto front *before* the end of optimization. While we cannot guarantee to be able to find this solution, the found solution still has a better utility than the best solution in the Pareto front obtained in the usual way.
- We examine the influence of the point in time when the DM is asked to pick a solution and show that asking too early may be detrimental, while asking too late may forfeit some of the benefits of the proposed approach.
- We explore the impact of a model mismatch between assumed and true utility function and show that the benefit of our approach, while reduced, remains significant despite model mismatch.

The paper is structured as follows. After a literature review, we formally define the problem considered in Section 3. The proposed algorithm is described in Section 4, followed by empirical results in Section 5. The paper concludes with a summary.

2 Literature Review

Depending on the involvement of the DM in the optimization process, multi-objective optimization can be classified into a priori approaches, a posteriori approaches, and interactive approaches [8, 21]. The field is very large, so we can only mention some of the most relevant papers here. A priori approaches ask the DM to specify their preferences ahead of optimization. This allows to turn the multi-objective optimization problem into a single objective optimization problem, but it is usually very difficult for a DM to specify their preferences before having seen the alternatives. Most multi-objective EAs are a-posteriori approaches, attempting to identify a good approximation of the Pareto frontier, and the DM can then pick the most preferred solution from this set. This is much easier for a DM, but identifying the entire Pareto front may be computationally expensive. Interactive approaches attempt to learn the DM's preferences during optimization and then focus the search on the most preferred region of the Pareto front. While this may yield solutions closer to the DM's true preferences, it also requires additional cognitive effort from the DM.

Our proposed algorithm lies in between a priori and interactive approaches: It generates an initial approximation of the Pareto front, and only requires the DM to pick a solution from this front. It then makes a final attempt to find a more preferred solution based on what the DM has picked, and returns this single final solution to the DM rather than an entire frontier.

BO is a global optimization technique that builds a Gaussian process (GP) surrogate model of the fitness landscape, and then uses the estimated mean and variance at each location to decide which solution to evaluate next. It uses an acquisition function to explicitly make a trade-off between exploitation and exploration (e.g., [17]). A frequently used acquisition function is the expected improvement (EI) [9] which selects the point with the largest expected improvement over the current best known solution as the next solution to evaluate. Recently, BO has been adapted to the multi-objective case, for a survey see [15]. One of the earliest approaches, ParEGO [11] simply uses the Tchebychev scalarization function to turn the multi-objective problem into a single objective problem, but it uses a different scalarization vector in every iteration where the next solution is decided according to EI. Other multi-objective algorithms fit separate models for each individual objective. [5] trains a GP model for each objective, then chooses the next solution to be evaluated according to a hypervolume-based acquisition criterion. Other multi-objective BO approaches include [2, 10, 12].

Recently, a few interactive multi-objective BO approaches have also been proposed [1, 7, 19]. Gaudrie et al. [6] allow the DM to specify a reference point a priori, and use this to subsequently focus the BO search.

3 Problem Definition

The standard multi-objective optimization problem with respect to a particular DM is defined as follows. We assume a D-dimensional real-valued space of possible *solutions*, i.e., $\mathbf{x} \in X \subset \mathbb{R}^D$. The objective function is an arbitrary black box $\mathbf{f} : X \to \mathbb{R}^K$ which returns a deterministic vector *output* $\mathbf{y} \in \mathbb{R}^K$. The (unknown) DM preference over the outputs can be characterized by a utility function $U : \mathbb{R}^K \to \mathbb{R}$. Thus, of all solutions in X, the DM's most preferred solution is $\mathbf{x}^* = \arg\max_{\mathbf{x}\in X} U(f(\mathbf{x}))$. There is a budget of B objective function evaluations, and we denote the n-th evaluated design by $\mathbf{x}^n$ and the n-th output by $\mathbf{y}^n = \mathbf{f}(\mathbf{x}^n)$ where, for convenience, we define the sampled solutions and outputs as $\tilde{X}^n = \{\mathbf{x}^1, \dots, \mathbf{x}^n\}$ and $\tilde{\mathbf{Y}}^n = \{\mathbf{y}^1, \dots, \mathbf{y}^n\}$, respectively.

In a standard EMO procedure, after consuming the budget, the algorithm returns the set of evaluated non-dominated solutions $\Gamma \subset \tilde{X}^n$ and the DM chooses a preferred solution $\mathbf{x}_p$ according to $\mathbf{x}_p = \arg\max_{x\in\Gamma} U(\mathbf{f}(x))$. Figure 1 shows that a solution set may not contain any solution close to the DM's true preferred Pareto-optimal solution, and thus the DM will choose a sub-optimal solution.

Interactive multi-objective optimization algorithms attempt to learn the DM's preferences and then focus the search effort onto the most preferred region of the Pareto front, which allows them to provide a more relevant set of solutions. However, the multiple interactions mean additional cognitive effort for the DM. In this paper, we restrict the interaction to a *single* selection of a most preferred solution from a non-dominated front, as in the standard, non-interactive case. However, we allow to ask the DM for this information *before* the end of optimization, after $B - p$ evaluations. This allows the algorithm to identify potentially more relevant solutions in the final p evaluations. At the end, the algorithm has to return a *single* recommended solution $\mathbf{x}_r$ (rather than asking the DM to choose again), so that the cognitive effort is equivalent to the non-interactive case. The aim is then to minimize the Opportunity Cost (OC) of the chosen sample,

$$OC = U(\mathbf{f}(\mathbf{x}^*)) - U(\mathbf{f}(\mathbf{x}_r)).$$

4 Proposed Approach

This section describes details of the proposed algorithm. Section 4.1 and Section 4.2 show the statistical models used for the objectives and the utility, respectively. Then, Section 4.3 provides background on EI-UU and Section 4.4 considers the case when the DM utility model is different from the model assumed by EI-UU. Finally, Section 4.5 provides a summary of the algorithm.

4.1 Statistical Model of the Objectives

Let us denote the set of evaluated points and their objective function values up to iteration n as $\mathscr{F}^n = \{(\mathbf{x}, \mathbf{y})^1, \dots, (\mathbf{x}, \mathbf{y})^n\}$. To model each objective function $y_j = f_j(\mathbf{x}), \forall j = 1, \dots, K$, we use an independent GP defined by a mean function $\mu_j^0(\mathbf{x}) : X \to \mathbb{R}$ and a covariance function $k_j^0(\mathbf{x}, \mathbf{x}') : X \times X \to \mathbb{R}$. Given $\mathscr{F}^n$, predictions at new locations $\mathbf{x}$ for output y_j are given by the posterior GP mean $\mu_j^n(\mathbf{x})$ and covariance $k_j^n(\mathbf{x}, \mathbf{x}')$. We use the popular squared exponential kernel that assumes $f_j(\mathbf{x})$ is a smooth function, and we estimate the hyper-parameters from $\mathscr{F}^n$ by maximum marginal likelihood. Details can be found in [13].

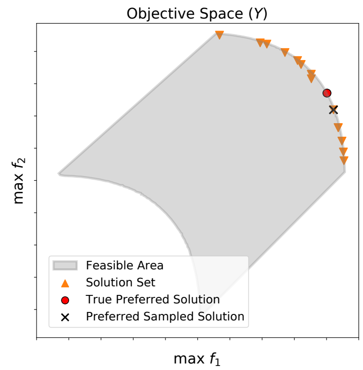


Fig. 1. HOLE test problem (b=0). (orange triangles) Solution set Γ shown to the DM. (red cross) Solution picked by the DM x_p. (red dot) True most preferred solution of the DM.

4.2 Statistical Model over the Utility

After $B-p$ evaluations, the DM selects a point μ^* from an estimated Pareto front $\mathscr{P}$. Therefore, an interaction I is defined by the generated front and the selected solution, $I = (\mathscr{P}, \mu^*)$. The estimated front is generated using an evolutionary algorithm (NSGA-II [3]) on the posterior mean response surfaces of the Gaussian processes.

Let us assume that the DM's utility can be described by a parametric utility function $U(\mathbf{x}, \theta)$ with parameters $\theta \in \Theta$. Similar to [1] and [16], we adopt a Bayesian approach to obtain a distribution over parameters $\theta \in \Theta$. Commonly used likelihood functions include probit and logit [20]. However, for simplicity we assume fully accurate preference responses. Then, if we consider a utility model $U(\theta)$, we would only accept a candidate parameter θ if the best solution from the generated front $\mathscr{P}$ according to $\mu(\theta) = \arg\max_{\mu \in \mathscr{P}} U(\mu, \theta)$ is the solution selected by the DM. The likelihood for θ is then

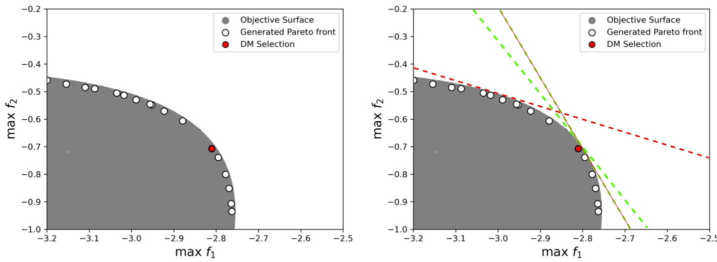
$$\mathscr{L}(\theta) = \mathbb{I}_{\mu(\theta)=\mu^*}.$$

Figure 2.b shows the above process by drawing three randomly generated linear utility functions.

Depending on the utility function model, the set of "compatible" ($\mathscr{L}(\theta) = 1$) parameterizations can be determined easily and in a deterministic way. More generally, if we assume a flat Dirichlet prior on the parameter space Θ, then a posterior distribution over θ is given by Bayes rule as

$$\mathbb{P}[\theta|I] \propto \mathscr{L}(\theta)\mathbb{P}[\theta].$$

We may obtain samples from the posterior distribution $\mathbb{P}[\theta|I]$ by simply generating Dirichlet samples from $\mathbb{P}[\theta]$ and accepting only those that are compatible



(a) (b)

Fig. 2. (a) Estimated Pareto front shown to the DM (white dots) and a single point selected by the DM (red dot). (b) Three scalarizations are generated using a Linear utility function. Some are compatible with the information elicited from the decision maker (dashed green), and thus accepted. Scalarizations that are not compatible would be rejected (dashed red).

with the DM interaction ($\mathscr{L}(\theta) = 1$). This approach may also be immediately extended to multiple and independent interactions where $\mathscr{L}$ may be expressed as the product of all different interactions. Thus, only those parameters compatible with all interactions would be accepted.

4.3 EI-UU with Preference Information

EI-UU [1] is a recently proposed multi-objective BO algorithm that is able to include uncertain preference information. Similar to ParEGO [11] it translates the multi-objective problem into a single-objective problem using an achievement scalarization function. However, ParEGO uses Tchebycheff scalarizations and randomly picks a different scalarization in every iteration to ensure coverage of the entire Pareto front, EI-UU uses linear scalarizations and integrates the expected improvement over all possible scalarizations, so it takes into account different scalarizations simultaneously rather than sequentially over iterations.

Given a parameterization θ, it is possible to determine the utility of the most preferred solution out of the solutions sampled so far, $\mathscr{F}^n$, as $u^*(\theta) = \max_{i=1,\ldots,n} U(\mathbf{x}_i, \theta)$. Then, EI-UU simply computes the expected improvement over all possible realizations for θ and outputs $\mathbf{y} = \mathbf{f}(\mathbf{x})$, i.e.,

$$\text{EI-UU}(\mathbf{x}) = \mathbb{E}_{\theta,\mathbf{y}}[\max\{U(\mathbf{x}, \theta) - u^*(\theta), 0\}].$$

Figure 3 provides an example in two-dimensional solution space, showing two EI landscapes for different θ (part a and b), as well as the corresponding overall improvement over several such realizations (c).

If the utility is linear (as suggested in [1]), the computations are essentially reduced to standard expected improvement, where we integrate over θ using a Monte-Carlo (MC) approximation. Otherwise, the whole expectation must be computed using Monte-Carlo with realisations of each objective at $\mathbf{x}$ generated

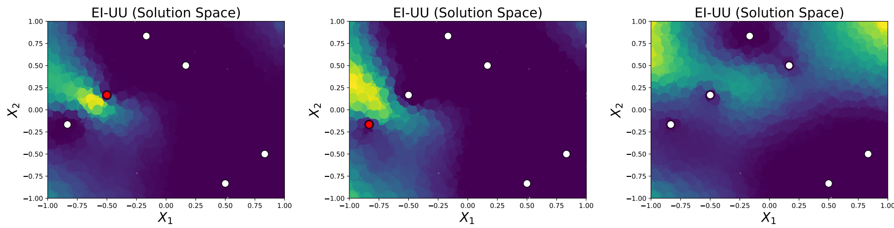


Fig. 3. Expected Improvement over the solution space, with brighter colors indicating higher expected improvement. Circles indicate sampled solutions. (a-b) Expected Improvement according to two different realizations of θ. Most preferred solution according to the specific θ is highlighted in red. (c) shows the MC average over several realizations of θ to obtain EI-UU. The recommended solution by the algorithm is selected according to the maximum value of EI-UU.

as $f_j(\mathbf{x}) = \mu_j^n(\mathbf{x}) + k_j^n(\mathbf{x}, \mathbf{x})Z$, where $Z \sim N(0, 1)$. Then, the overall expectation over θ and Z is computed using a MC average,

$$\text{EI-UU}(\mathbf{x}) \approx \frac{1}{N_\Theta N_Z} \sum_{w=1,t=1} \max\{U(\mathbf{x}, \theta_w, Z_t) - u^*(\theta_w)\}.$$

It is straightforward to accommodate preference information in this approach simply by adapting the distribution of the different scalarizations considered.

4.4 Utility Mismatch

All proposed approaches require a parameterized model of the utility function. In reality, this assumed model may not be able to accurately represent the true utility function of the DM. To mitigate the risk of the DM being misrepresented, we can use more flexible models (e.g., Cobb-Douglas utility function, Choquet Integral, or artificial neural network) or simply allow any of a set of simple models, which is the approach we take below. In general, there is a trade-off: The more restrictive the utility model, the more informative the elicited DM preference information, and the more focused the search in the final p steps may be. On the other hand, if the assumed utility model was wrong, the focus would be put on the wrong area, and the approach could fail. Thus, a more flexible model will provide less benefit, but also smaller risk of getting it wrong.

In this paper, let us consider a given set of L utility model candidates, as $\{U_1(\theta_1), \ldots, U_l, \ldots, U_L(\theta_L)\}$. In the Bayesian framework, one complete utility model U_l is formed by a likelihood function $\mathscr{L}_{U_l}(\theta)$ and a prior probability density function $\mathbb{P}(\theta|U_l)$, where now we explicitly emphasize the dependence of a model U_l on the likelihood and prior distribution. Therefore, to obtain a posterior distribution over models given the interaction I, i.e., $\mathbb{P}(U_l|I)$, we must compute the marginal likelihood, or Bayesian evidence, as

$$\mathbb{P}[I|U_l] = \int_{\theta_l \in \Theta_l} \mathscr{L}_{U_l}(\theta_l)\mathbb{P}(\theta_l|U_l)\mathrm{d}\theta_l.$$

This quantity represents the probability of the data collected, I, given a utility model assumption U_l. However, we rely on approximating each term $\mathbb{P}[I|U_l]$ through MC integration. Hence, a posterior distribution over the different candidate utilities may be computed as

$$\mathbb{P}[U_l|I] = \frac{\mathbb{P}[I|U_l]\mathbb{P}[U_l]}{\sum_{i=1}^{L}\mathbb{P}[I|U_i]\mathbb{P}[U_i]},$$

where $\mathbb{P}[U_l]$ represents a prior distribution for the candidate utility models. If we also consider a uniform distribution over the candidate models as $\mathbb{P}[U_l] = 1/L$, then the posterior distribution only depends on the likelihood distribution terms. This shows that models with higher evidence values tend to have higher weight than other competing candidate models. Finally, EI-UU may be adapted by simply taking the expectation over the posterior $\mathbb{P}[U_l|I]$,

$$\text{EI-UU}(\mathbf{x}) = \sum_{i=1}^{L}\mathbb{E}_{\theta_i,\mathbf{y}}[\max\{U_i(\mathbf{x},\theta_i) - u_i^*(\theta_i), 0\}]\mathbb{P}[U_i|I]$$

4.5 Algorithm

The proposed algorithm follows the standard EI-UU algorithm with a single exception. Instead of letting the DM choose their most preferred solution at the end of optimization (after the budget of B samples has been depleted), we let the DM choose their most preferred solution from an approximated frontier already after $B - p$ samples. This approximated frontier is generated by NSGA-II on the posterior mean function obtained from the GPs for each objective. From the interaction, we derive user preferences as explained above, and evaluate an additional p solutions using this preference information. Finally, at the end of the optimization run, the algorithm returns to the DM the single solution $\mathbf{x}_r$ that it thinks has the best expected performance.

5 Results and Discussion

To assess the performance of the proposed approach, we investigate the Opportunity Cost (OC) dependence on the time when the DM is asked to pick a solution.

5.1 Experimental setup

In all experiments, EI-UU is seeded with an initial stage of evaluations using $2(D+1)$ points allocated by Latin hypercube sampling over X. These evaluations

Algorithm 1: Overall Algorithm.

Input: black-box function, size of Monte-Carlo N_Θ and N_Z for EI-UU, and the number of function evaluations $B - p$ before asking the DM.

0. Collect initial simulation data, $\mathscr{F}^n$, and fit an independent Gaussian process for each black-box function output.
1. **While** $b <$ B **do:**
2. **If** $b = B - p$ **do:**
3. Generate approximated Pareto frontier.
4. DM selects preferred solutions.
5. Compute posterior distribution $\mathbb{P}[\theta|I]$.
7. Compute $\mathbf{x}^{n+1} = \arg\max_{\mathbf{x} \in X}$ EI-UU$(\mathbf{x}; N_\Theta, N_Z)$.
8. Update $\mathscr{F}^n$, with sample $\{(\mathbf{x}, \mathbf{y})^{n+1}\}$
9. Update each Gaussian process with $\mathscr{F}^n$
10. Update budget consumed, $b \leftarrow b + 1$
11. **Return:** Recommend solution, $\mathbf{x}_r = \arg\max_{\mathbf{x} \in X} \sum_{i=1}^{L} \mathbb{E}_{\theta_l}[U_i(\mathbf{x}, \theta)]\mathbb{P}[U_i|I]$

are in addition to the budget B. NSGA-II is run for 300 generations with a population size of 100 to produce a Pareto front approximation.

We use four different test functions:

1. The HOLE function [14] is defined over $X = [-1, 1]^2$ and has $K = 2$ objectives. We use $b > 0$, which produces two unconnected Pareto fronts, and the following function parameters: $q = 0.2$, $p = 2$, $d_0 = 0.02$, and $h = 0.2$.
2. An instance of the DTLZ2 function [4] with $K = 3$ objectives and defined over $X = [0, 1]^3$.
3. The ZDT1 function [4] with $k = 2$ objectives and defined over $X = [0, 1]^3$.
4. The rocket injector design problem [18]. This problem consists of minimizing the maximum temperature of the injector face, the distance from the inlet, and the maximum temperature on the post tip over $X = [0, 1]^4$.

All results are averaged over 20 independent replications and the figures below show the mean and 95% confidence intervals for the OC.

5.2 Preference Elicitation without model mismatch

In this subsection, we look at the benefit that can be gained from letting the DM choose from the approximated Pareto front. We assume the DM has a Tchebychev utility with true underlying (but unknown to the algorithm) parameters θ. The true underlying parameters are generated randomly for every replication of a run using a different random seed. We include an optimistic setting ("Perfect Information"), such that once we interact with the DM, we receive the true underlying parameter θ and use this information in the following optimization steps. This represents the best performance we can hope for.

Figure 4 (**first column**) shows results depending on when the DM selects a point from the Pareto front. The benchmark is the case where the DM picks from the final set of non-dominated solutions after $B - p = 100$ iterations. The figure clearly indicates a trade-off about when the preference information should be elicited. The earlier the DM is involved, the longer it is possible for EI-UU to exploit the preference information gained. On the other hand, the earlier the DM is shown an approximated Pareto front, the less accurate is this Pareto front, and thus the learned preference information may be wrong. For the HOLE function, asking the DM too early (small $B - p$ values) even leads to substantially worse results than letting the DM pick a solution after the end of the optimization ($B - p = 100$). However, for all four functions considered, there is a broad range of settings that yield a significantly better result than when asking the DM after optimization. The intuitive explanation of the above trade-off is confirmed when comparing results with the case when perfect preference information is obtained. In this case, it is best to interact as early as possible, as there is no risk of learning something wrong or less informative due to a poor approximation of the Pareto front.

Figure 4 (**second column**) looks at the dependence of the observed benefit of letting the DM pick a solution early on the budget B, comparing $B = 20, 100, 200$. As expected, the OC decreases with increasing number of function evaluations. If the budget is very small ($B = 20$), it seems not possible to gain much by asking the DM early. Following the intuition above this is not really surprising, as after a very short optimization run the Pareto front is not well approximated and any information gained from the DM may be misleading. For a large budget ($B = 200$), there is still a small benefit to be gained for some of the test functions. However, intuitively, as the available budget tends to infinity, we can expect that the algorithm will return a very dense and accurate Pareto frontier, so the DM will be able to choose their true most preferred solution (or some solution very close to that), and so it is not possible to improve over that. The figure also allows us to appreciate the magnitude of the benefit of letting the DM choose earlier. For example, using a medium budget of $B = 100$ for the HOLE function and letting the DM choose after 80%=80 samples yields an OC almost as small as increasing the budget to $B = 200$ samples and letting the DM choose at the end of the run.

5.3 Preference Elicitation with model mismatch

If the DM's true utility model is different from the parameterized model used by the algorithm there is a mismatch between the DM's true utility model (unknown to the algorithm) and the learned utility model in the algorithm. The consequences of model mismatch are explored next.

So far, we assumed that the true DM utility model is *Tchebychev*, and this model is used when the DM picks a solution from the approximated Pareto front, and again to evaluate the OC at the end. Now, in each replication of the algorithm, a random true *linear* utility model is generated for the DM. However,

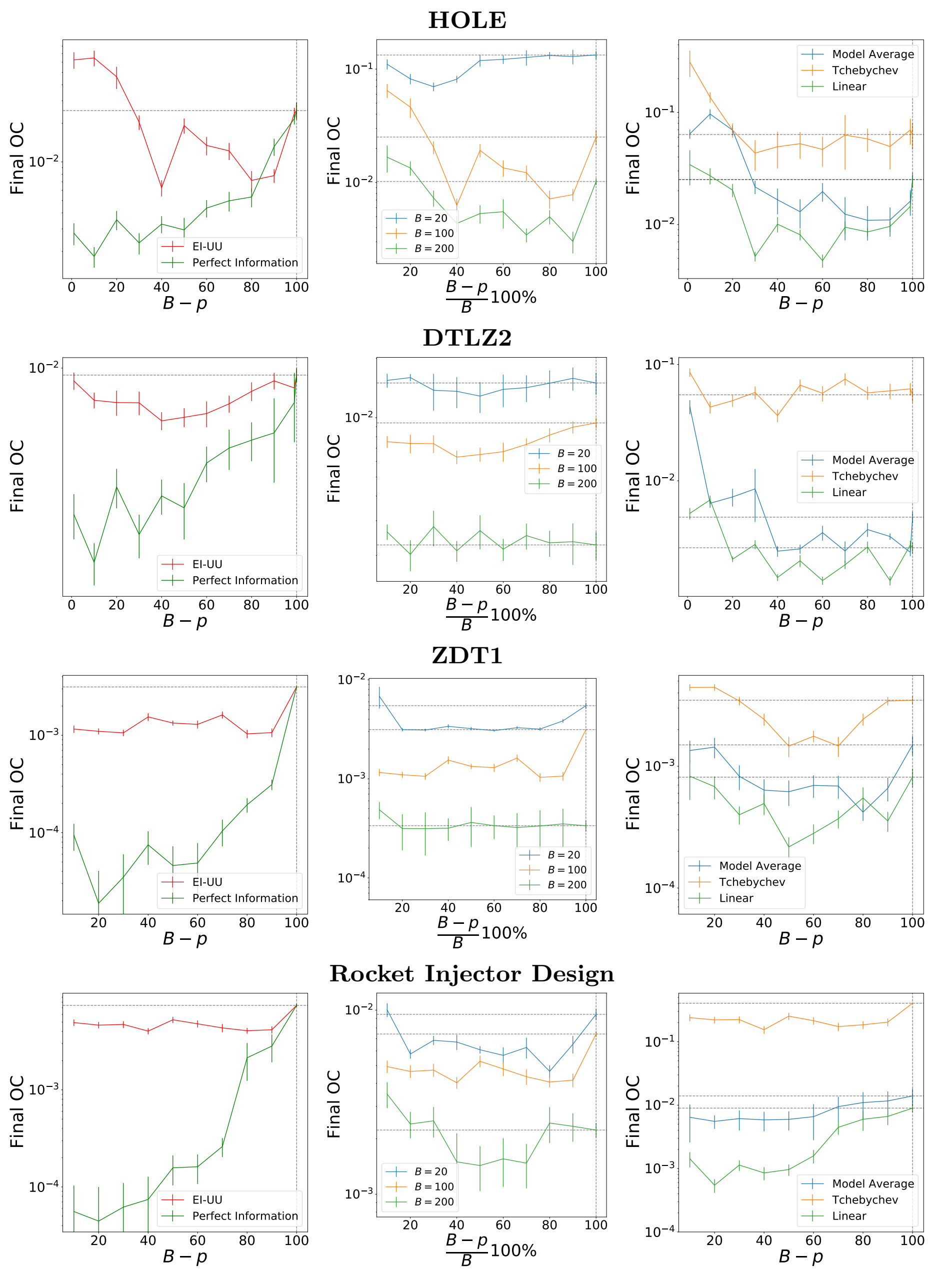


Fig. 4. First column: Final true utility of the generated solution set after $B = 100$ iterations, depending on how many iterations from the start the DM was presented with an approximation of the front $(B - p)$. True and assumed DM utility was *Tchebychev*. **Second column**: Final true utility of the generated solution set after different budgets, depending on the *percentage* of consumed budget before presenting an approximation of the front to the DM. True and assumed DM utility was *Tchebychev*. **Third column**: Final true utility of the generated solution set after $B = 100$ iterations, depending on how many iterations from the start the DM was presented with an approximation of the front $(B-p)$. The DM has a *Linear* utility and we show results for three different utility model assumptions (Linear, Tchebychev, and a model average). The dashed horizontal lines show the OC when the DM selects the best solution after optimization.

all approaches still assume a Tchebychev utility function and use this to focus the search over the last p iterations and also to recommend a final solution.

We consider two candidate utility functions to average EI-UU, Tchebychev and Linear. As Figure 4 (**third column**) shows, model mismatch (orange) leads to substantially worse solutions, even when the DM only picks a solution *after* optimization. This seems counter-intuitive at first, but actually, EI-UU implicitly assumes a distribution of utility functions for scalarization, even if the distribution of parameter values θ is uniform. It appears that even with this relatively weak assumption, a model mismatch leads to significantly worse solutions. It is also clear that if there is a severe model mismatch (the algorithm assumes Tchebychev but the true utility is linear), letting the DM choose a solution before the end of the optimization is not always helpful (for the HOLE and DTLZ2 function, no setting of $B-p$ leads to significantly better results than $B-p = 100$). On the other hand, except for letting the DM choose very early, it is also not significantly worse. For the correct and the more flexible model encompassing Tchebychev and Linear, better results can be obtained for all four test problems by letting the DM choose before the end of optimization. But as expected, the benefit is smaller with the more flexible model, as its flexibility means it can focus less narrowly than if a less flexible but correct model is assumed.

6 Conclusion

For the case of expensive multi-objective optimization, we show how the surrogate models generated by Bayesian optimization can be used not only to speed up optimization, but also to show the DM a *predicted* Pareto front *before* the end of optimization, rather than the sampled non-dominated solutions at the end of the optimization run. Then, the information on the most preferred solution can be used to focus the final iterations of the algorithm to try and find this predicted most preferred solution, or even a better solution, which is then returned to the DM as recommended solution (no more interaction from the DM required - as in the standard case for EMO, the DM only once picks a most preferred solution from a frontier). We demonstrate empirically on four test problems that for various scenarios, the benefit in terms of true utility to the DM is significant.

Future directions of research may include the evaluation on a wider range of test problems, making the time to ask the DM self-adaptive, and to turning the approach into a fully interactive approach with multiple interactions with the DM during the optimization.

Acknowledgements

The first author would like to acknowledge funding from ESTECO SpA and EPSRC through grant EP/L015374/1.

References

1. Astudillo, R., Frazier, P.: Multi-attribute bayesian optimization with interactive preference learning. In: International Conference on Artificial Intelligence and Statistics. pp. 4496–4507. PMLR (2020)
2. Daulton, S., Balandat, M., Bakshy, E.: Parallel bayesian optimization of multiple noisy objectives with expected hypervolume improvement. In: Neural Information Processing Systems (2021)
3. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017
4. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable Test Problems for Evolutionary Multiobjective Optimization, pp. 105–145. Springer London, London (2005)
5. Emmerich, M., Deutz, A., Klinkenberg, J.: Hypervolume-based expected improvement: Monotonicity properties and exact computation. IEEE Congress of Evolutionary Computation (CEC) pp. 2147–2154 (2011)
6. Gaudrie, D., Riche, R.L., Picheny, V., Enaux, B., Herbert, V.: Targeting solutions in bayesian multi-objective optimization: sequential and batch versions. Annals of Mathematics and Artificial Intelligence **88**(1), 187–212 (2020)
7. Hakanen, J., Knowles, J.D.: On using decision maker preferences with parego. In: International Conference on Evolutionary Multi-Criterion Optimization. pp. 282–297. Springer (2017)
8. J.Branke: Mcda and multiobjective evolutionary algorithms. In: Multiple Criteria Decision Analysis, pp. 977–1008. Springer (2016)
9. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. Journal of Global Optimization **13**(4), 455–492 (1998). https://doi.org/10.1023/A:1008306431147, https://doi.org/10.1023/A:1008306431147
10. Keane, A.J.: Statistical improvement criteria for use in multiobjective design optimization. AIAA journal **44**(4), 879–891 (2006)
11. Knowles, J.: Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. IEEE Transactions on Evolutionary Computation **10**(1), 50–66 (2006)
12. Picheny, V.: Multiobjective optimization using Gaussian process emulators via stepwise uncertainy reduction. Statistics and Computing **25**(6), 1265–1280 (2015)
13. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press (2006)
14. Rigoni, E., Poles, S.: NBI and MOGA-II, two complementary algorithms for multi-objective optimizations. In: Dagstuhl seminar proceedings. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2005)
15. Rojas-Gonzalez, S., Van Nieuwenhuyse, I.: A survey on kriging-based infill algorithms for multiobjective simulation optimization. Computers & Operations Research **116**, 104869 (2020). https://doi.org/https://doi.org/10.1016/j.cor.2019.104869
16. Schoenauer, M., Akrour, R., Sebag, M., Souplet, J.C.: Programming by feedback. In: Xing, E.P., Jebara, T. (eds.) Proceedings of the 31st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 32, pp. 1503–1511. PMLR, Bejing, China (22–24 Jun 2014), https://proceedings.mlr.press/v32/schoenauer14.html

17. Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., de Freitas, N.: Taking the human out of the loop: A review of bayesian optimization. Proceedings of the IEEE **104**(1), 148–175 (2016). https://doi.org/10.1109/JPROC.2015.2494218
18. Tanabe, R., Ishibuchi, H.: An easy-to-use real-world multi-objective optimization problem suite. Applied Soft Computing **89**, 106078 (2020). https://doi.org/https://doi.org/10.1016/j.asoc.2020.106078, https://www.sciencedirect.com/science/article/pii/S1568494620300181
19. Taylor, K., Ha, H., Li, M., Chan, J., Li, X.: Bayesian preference learning for interactive multi-objective optimisation. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 466–475 (2021)
20. Wirth, C., Akrour, R., Neumann, G., Fürnkranz, J.: A survey of preference-based reinforcement learning methods. J. Mach. Learn. Res. **18**(1), 4945–4990 (Jan 2017)
21. Xin, B., Chen, L., Chen, J., Ishibuchi, H., Hirota, K., Liu, B.: Interactive multiobjective optimization: A review of the state-of-the-art. IEEE Access **6**, 41256–41279 (2018)