Multi-objective optimisation via the R2 utilities

Ben Tu[†] Nikolas Kantas[†] Robert M. Lee[‡] Behrang Shafei[‡]

Abstract

The goal of multi-objective optimisation is to identify a collection of points which describe the best possible trade-offs among the multiple objectives. In order to solve this vector-valued optimisation problem, practitioners often appeal to the use of scalarisation functions in order to transform the multi-objective problem into a collection of single-objective problems. This set of scalarised problems can then be solved using traditional single-objective optimisation techniques. In this paper, we formalise this convention into a general mathematical framework. We show how this strategy effectively recasts the original multi-objective optimisation problem into a single-objective optimisation problem defined over sets. An appropriate class of objective functions for this new problem is that of the R2 utilities, which are utility functions that are defined as a weighted integral over the scalarised optimisation problems. As part of our work, we show that these utilities are monotone and submodular set functions which can be optimised effectively using greedy optimisation algorithms. We then analyse the performance of these greedy algorithms both theoretically and empirically. Our analysis largely focusses on Bayesian optimisation, which is a popular probabilistic framework for black-box optimisation.

1 Introduction

Decisions that are made in the real-world lead to a variety of consequences that have to be considered beforehand. It is rare that a decision leads only to positive changes that would benefit everyone involved. In many cases, there is a trade-off that has to be struck which tries to balance between many different criteria. Multi-objective optimisation formalises this decision making problem mathematically as a vector-valued optimisation problem. The solution to this problem is a set of decisions, whose objective values lead to the best possible trade-offs among the multiple criteria. Equipped with this solution set, a decision maker can then make a more informed decision which takes into account all the possible trade-offs.

Many efforts over the past few decades have been spent developing the theory and algorithms that are used to solve this multi-objective optimisation problem [58, 28]. The aim of this paper is to consolidate the core ideas of these approaches into a mathematical framework that will support the future development of algorithms and the theory behind them. Notably, we study the family of R2 utility functions, which connects three different fields of optimisation: multi-objective optimisation, single-objective optimisation and set optimisation. We support these connections with general theoretical results and illustrative examples that provide intuition. Whilst these results are not surprising and are already known for a specific instance of R2 utility, namely, the hypervolume indicator [102, 86], to the best of our knowledge they have not been presented in the most general form and are absent from the literature. Furthermore,

[†]Imperial College London, United Kingdom
[‡]BASF SE, Germany

these clear connections can also be exploited in order to design more principled algorithms. To illustrate this, we present a motivating example of how these ideas can be applied for Bayesian optimisation, which is a popular black-box optimisation strategy that has gained a lot of interest and success over the recent decade. We also provide general performance bounds for this case and detailed numerical examples.

1.1 Structure of the paper

The remainder of the paper is organised as follows: In Section 2, we define the multi-objective optimisation problem and introduce the two main philosophies that are often used to solve this problem: the scalarisation perspective and the utility perspective. In Section 3, we unify these two methodologies by introducing the R2 utilities, which form a family of utility functions that are defined using scalarisation functions. We then show that the family of R2 utilities satisfies many desirable properties, which makes them a sensible criterion to optimise. In Section 4, we give a discussion on how we can solve the multi-objective optimisation problem using R2 utilities. We focus our attention on greedy optimisation strategies because they come with theoretical guarantees. In particular, we prove a general performance bound which is satisfied by any greedy Bayesian optimisation algorithm based on R2 utilities. In Section 5, we evaluate these greedy Bayesian optimisation strategies empirically in light of these new theoretical results. Finally, in Section 6, we conclude the paper with a summary and discussion of future work. In Appendix A, we give the proofs of the main results.

2 Preliminaries

Consider a vector-valued function $f : \mathbb{X} \to \mathbb{R}^M$ defined over a D-dimensional space of feasible inputs $\mathbb{X} \subseteq \mathbb{R}^D$. The multi-objective maximisation problem is denoted by the equation

$$\max_{\mathbf{x} \in \mathbb{X}} f(\mathbf{x}), \tag{1}$$

where the maximum is defined via the Pareto partial ordering relation.

Definition 2.1 (Pareto domination) *The weak, strict, and strong Pareto domination are denoted by the binary relations $\succeq, \succ$ and $\succ\!\!\succ$, respectively. We say that a vector $\mathbf{y} \in \mathbb{R}^M$ weakly, strictly, or strongly Pareto dominates another vector $\mathbf{y}' \in \mathbb{R}^M$, if*

$$\begin{aligned} \mathbf{y} \succeq \mathbf{y}' &\iff \mathbf{y} - \mathbf{y}' \in \mathbb{R}^M_{\geq 0}, \\ \mathbf{y} \succ \mathbf{y}' &\iff \mathbf{y} - \mathbf{y}' \in \mathbb{R}^M_{\geq 0} \setminus \{\mathbf{0}_M\}, \\ \mathbf{y} \succ\!\!\succ \mathbf{y}' &\iff \mathbf{y} - \mathbf{y}' \in \mathbb{R}^M_{> 0}, \end{aligned}$$

respectively, where $\mathbf{0}_M \in \mathbb{R}^M$ denotes the M-dimensional vector of zeros.

Definition 2.2 (Pareto optimality) *Consider a function $f : \mathbb{X} \to \mathbb{R}^M$, we say an input $\mathbf{x} \in \mathbb{X}$ is weakly or strictly Pareto optimal if the objective vector $f(\mathbf{x})$ is not strongly or strictly dominated, respectively, by any other objective vector $f(\mathbf{x}')$ with $\mathbf{x}' \in \mathbb{X} \setminus \{\mathbf{x}\}$.*

The convention in multi-objective optimisation is to target the strictly Pareto optimal points. The collection of strictly Pareto optimal inputs is called the Pareto set, $\mathbb{X}^* = \arg\max_{\mathbf{x} \in \mathbb{X}} f(\mathbf{x}) \subseteq \mathbb{X}$, whilst the corresponding image is called the Pareto front, $\mathbb{Y}^* = f(\mathbb{X}^*) = \max_{\mathbf{x} \in \mathbb{X}} f(\mathbf{x}) \subset \mathbb{R}^M$.

Set Pareto domination. Practitioners are often only interested in identifying a discrete approximation of the Pareto front. For this reason, we will consider working in the space of finite sets: $\mathbb{B}(\mathbb{R}^M) = \{Y \subseteq \mathbb{R}^M : |Y| < \infty\}$. In practice, we are interested in making comparisons between different Pareto front approximations. To that end, we will now extend the Pareto partial ordering to be defined over finite sets. Informally, we will say a set $A \in \mathbb{B}(\mathbb{R}^M)$ dominates another set $B \in \mathbb{B}(\mathbb{R}^M)$ if the region dominated by A contains the region dominated by B.

Definition 2.3 (Dominated region) *For a set of vectors $Y \in \mathbb{B}(\mathbb{R}^M)$, the weak dominated region is defined as the collection of vectors which is weakly dominated by at least one vector in this set, that is $\mathbb{D}_{\preceq}(Y) = \bigcup_{\mathbf{y} \in Y} \{\mathbf{a} \in \mathbb{R}^M : \mathbf{a} \preceq \mathbf{y}\}$.*

Definition 2.4 (Set Pareto domination) *We say a set of vectors $A \in \mathbb{B}(\mathbb{R}^M)$ weakly or strictly Pareto dominates another set of vectors $B \in \mathbb{B}(\mathbb{R}^M)$, if*

$$A \succeq B \iff \mathbb{D}_{\preceq}(A) \supseteq \mathbb{D}_{\preceq}(B),$$
$$A \succ B \iff \mathbb{D}_{\preceq}(A) \supset \mathbb{D}_{\preceq}(B),$$

respectively.

Remark 2.1 *The above definition of set domination might differ slightly from alternative definitions given in the literature [41]. Nevertheless, the core idea remains the same: a set dominates another if for any element in the latter set, we can always find an element in the former set which dominates it.*

Reformulating the multi-objective optimisation problem. Designing algorithms to solve the multi-objective optimisation (1) directly is challenging because the Pareto partial ordering means that not all vectors or sets of vectors are comparable with each other. To address this limitation, decision maker's often rely on the use of scalarisation functions, $s : \mathbb{R}^M \to \mathbb{R}$, or utility functions, $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$, in order make comparisons between vectors or sets of vectors, respectively. In what follows, we will review these two classes of functions and show how they effectively reformulate the multi-objective optimisation problem as a different optimisation problem. We will then unify these two perspectives in Section 3, by showing how it is possible to construct a sensible utility function from a collection of scalarisation functions.

2.1 Scalarisation perspective

A well-known approach for solving the multi-objective optimisation problem (1) is to transform it into a collection of single-objective optimisation problems. These problems can then be jointly solved using classical techniques from single-objective optimisation. We refer to this reformulation as the scalarisation perspective of multi-objective optimisation because we typically rely on the use of scalarisation functions in order to establish these single-objective problems. This perspective is also sometimes referred to as the decomposition-based approach to multi-objective optimisation [97, 52, 84].

Scalarised optimisation problems. Formally, in the scalarisation perspective, we are interested in solving a collection of single-objective optimisation problems obtained applying a family of scalarisation functions: $\{s_{\boldsymbol{\theta}} : \mathbb{R}^M \to \mathbb{R} : \boldsymbol{\theta} \in \Theta\}$. In particular, we want to solve the scalarised optimisation problem,

$$\max_{\mathbf{x} \in \mathbb{X}} s_{\boldsymbol{\theta}}(f(\mathbf{x})) \tag{2}$$

jointly for all scalarisation parameters $\boldsymbol{\theta} \in \Theta$. Intuitively, each scalarised problem corresponds to one point that we are interested in targetting. If the number of scalarisation parameters

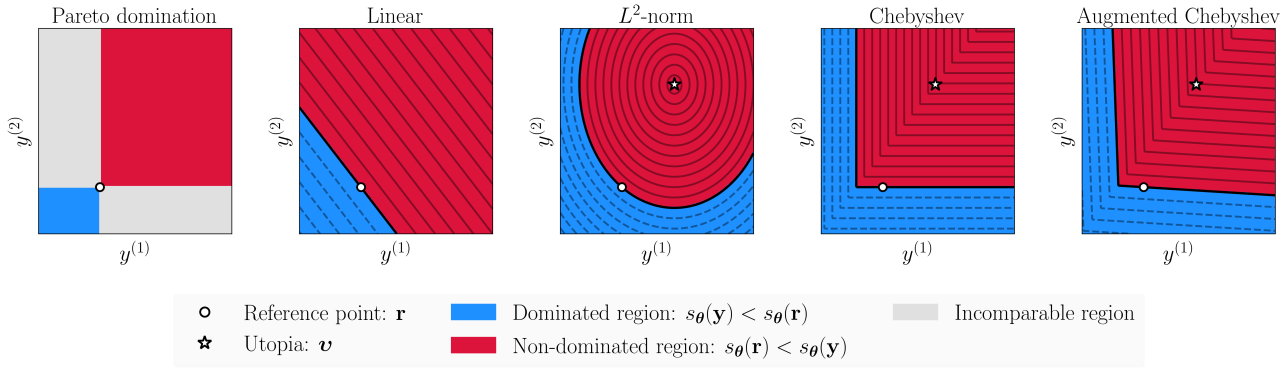

Figure 1: *A comparison of the domination regions based on the standard Pareto partial ordering over vectors and some popular scalarisation functions in the two-objective setting.*

is finite, $|\Theta| < \infty$, then we are only targetting a finite number of points. In contrast, if we had a separate scalarised problem for each strictly Pareto optimal point, then jointly solving this set of problems would be equivalent to solving the original multi-objective optimisation problem.

Ideally, we want the solutions to the scalarised optimisation problems to be Pareto optimal in some sense. This property turns out to be satisfied when the scalarisation function preserves the Pareto partial ordering. Specifically, whenever one vector dominates another, we want the scalarised value to also be greater. This notion of preserving the Pareto partial ordering can be interpreted as a notion of monotonicity.

Definition 2.5 (Monotonicity) *A scalarisation function $s : \mathbb{R}^M \to \mathbb{R}$ is monotonically increasing, strictly monotonically increasing, or strongly monotonically increasing if*

$$\begin{aligned} \mathbf{y} \succeq \mathbf{y}' &\implies s(\mathbf{y}) \geq s(\mathbf{y}'), \\ \mathbf{y} \succ \mathbf{y}' &\implies s(\mathbf{y}) > s(\mathbf{y}'), \\ \mathbf{y} \succcurlyeq \mathbf{y}' &\implies s(\mathbf{y}) > s(\mathbf{y}'), \end{aligned}$$

for any vectors $\mathbf{y}, \mathbf{y}' \in \mathbb{R}^M$, respectively. Similarly, the function is monotonically decreasing, strictly monotonically decreasing, or strongly monotonically decreasing if the reverse inequalities holds.

A well-known result in multi-objective optimisation states that whenever the scalarisation function is strictly or strongly monotonically increasing, then the solution set is Pareto optimal [58, Part 2, Theorem 3.5.4]—this result is summarised in Proposition 2.1 and proved in Appendix A.1.

Proposition 2.1 (Monotonicity implies optimality) *Consider an objective function $f : \mathbb{X} \to \mathbb{R}^M$ and a scalarisation function $s : \mathbb{R}^M \to \mathbb{R}$. If the scalarisation function is strictly or strongly monotonically increasing over the feasible objective space, then the solutions in $X_s^* = \arg\max_{\mathbf{x} \in \mathbb{X}} s(f(\mathbf{x})) \subseteq \mathbb{X}$ are strictly or weakly Pareto optimal, respectively.*

Example 2.1 (Popular scalarisation functions) *There are many possible scalarisation functions that we can use in practice [57]. Below, we give a few examples of some popular scalarisation functions. In particular, we consider the linear, L^p-norm (Lp), Chebyshev (Chb) and*

augmented Chebyshev (AugChb) scalarisation functions:

$$s_{\mathbf{w}}^{Linear}(\mathbf{y}) = \sum_{m=1}^{M} w^{(m)} y^{(m)}, \tag{3}$$

$$s_{(\boldsymbol{\upsilon},\mathbf{w})}^{Lp}(\mathbf{y}) = -\left(\sum_{m=1}^{M} |w^{(m)}(\upsilon^{(m)} - y^{(m)})|^p\right)^{1/p}, \tag{4}$$

$$s_{(\boldsymbol{\upsilon},\mathbf{w})}^{Chb}(\mathbf{y}) = -\max_{m=1,\dots,M} w^{(m)}(\upsilon^{(m)} - y^{(m)}), \tag{5}$$

$$s_{(\boldsymbol{\upsilon},\mathbf{w},\gamma)}^{AugChb}(\mathbf{y}) = s_{(\boldsymbol{\upsilon},\mathbf{w})}^{Chb}(\mathbf{y}) + \gamma s_{\mathbf{w}}^{Linear}(\mathbf{y}), \tag{6}$$

respectively, where $\mathbf{y} \in \mathbb{R}^M$ *is an objective vector,* $\mathbf{w} \in \Delta^{M-1} := \{\mathbf{y} \in \mathbb{R}^M_{\geq 0} : ||\mathbf{y}||_{L^1} = 1\}$ *is a weight vector lying in the non-negative* M*-dimensional simplex,* $\boldsymbol{\upsilon} \in \mathbb{R}^M$ *is an ideal*[1] *reference point,* $p \geq 1$ *is a constant controlling the* L^p*-norm, and* $\gamma \geq 0$ *is a penalty parameter. In Figure 1, we illustrate the contours of these scalarisation functions in the bi-objective setting, for one particular choice of scalarisation parameter.*

Notably, the linear, Chebyshev, and augmented Chebyshev scalarisation functions are strongly monotonically increasing over the whole objective space and therefore they satisfy the result of Proposition 2.1. On the other hand, the Lp scalarisation function is only strongly monotonically increasing in the space dominated by the reference point and therefore it only satisfies this result when the reference point is set accordingly.

Remark 2.2 (Chebyshev scalarisation) *Proposition 2.1 ensures that some Pareto optimal solutions can be targetted when we vary the scalarisation parameter. It does not give any guarantees on targetting all of the Pareto optimal solutions. These type of results are typically only possible for explicit choices of scalarisation functions. For example, it is known that we can target all the weakly Pareto optimal solutions, which are strictly Pareto comparable with the reference vector* $\boldsymbol{\upsilon} \in \mathbb{R}^M$*, by varying the weight parameter* $\mathbf{w} \in \Delta^{M-1}$ *in the Chebyshev scalarisation (5) function [58, Part 2, Theorem 3.4.5].*

Remark 2.3 (Objective transformations) *Scalarisation functions are naturally sensitive to the scales of the objectives. Thus, it is common for practitioners to apply a transformation function,* $\tau : \mathbb{R}^M \to \mathbb{R}^M$*, to the objective vector before applying the scalarisation function* $s(\tau(\mathbf{y})) \in \mathbb{R}$*. For instance, one might consider applying a linear transformation, which normalises the feasible objective values to the unit hypercube,* $[0,1]^M$*, before computing any of the distance-based scalarisation functions in Example 2.1. In this paper, we will implicitly assume that any such transformation has already been included in the definition of the scalarisation function.*

2.2 Utility perspective

One of the primary challenges in multi-objective optimisation concerns the quantification of the quality of an approximate Pareto front. As observed by Zitzler et al. [101], the set Pareto domination is only a partial ordering relation, which means that we are not always able to determine whether one approximation set is better than another. Therefore, the burden often falls onto the decision maker to characterise whether one set is more preferable than another, even when the sets are not Pareto comparable. To that end, decision maker's often rely on utility functions, $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$, in order to determine the quality of an approximate Pareto front.

[1] An ideal or utopia point is a vector whose objective values are desirable and conversely, the disagreement or nadir point is a vector comprised of undesirable objective values. Both of these vectors are typically set according to the decision maker's preferences [90].

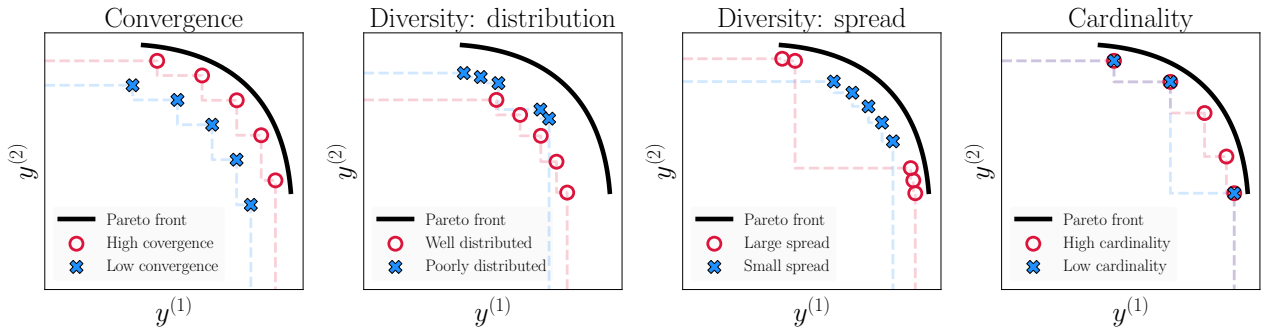

Figure 2: *A comparison of the aspects that determine the quality of a Pareto front approximation.*

This naturally leads to the utility perspective of multi-objective optimisation, which considers the problem of directly optimising this utility function. This perspective is also sometimes referred to as the indicator-based approach to multi-objective optimisation [97, 52, 84].

Utility optimisation problem. Formally, in the utility perspective, we are interested in optimising the utility function

$$\max_{X \subseteq \mathbb{X}, |X| \leq P} U(f(X)) \tag{7}$$

for some number $P > 0$. We have included a cardinality constraint into this set optimisation problem in order to emphasise the fact that we are typically interested in building a discrete approximation of the Pareto front. Naturally, we could lift this constraint if an abundance of computational power was available.

Ideally, the utility function should be designed to preserve the Pareto partial ordering over sets. Similar to the notion of monotonicity for scalarisation functions, we introduce the notion of compliancy for utility functions.

Definition 2.6 (Pareto compliancy) *A utility function $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$ is weakly or strictly Pareto compliant if*

$$A \succeq B \implies U(A) \geq U(B),$$
$$A \succ B \implies U(A) > U(B),$$

for any sets of vectors $A, B \in \mathbb{B}(\mathbb{R}^M)$, respectively.

The Pareto compliancy property implies that the utility optimisation problem (7) can be solved by using some subset of the Pareto set. When the Pareto front is finite[2], the strict Pareto compliancy property leads to a stronger result which states that the smallest set which maximises the utility is the Pareto front; that is, $\max_{X \in \mathbb{X}, |X| \leq P} U(f(X)) = U(f(\mathbb{X}^*)) = U(\mathbb{Y}^*)$ for any $P \geq |\mathbb{Y}^*|$.

Qualitative aspects. In the multi-objective literature, the utility function is often referred to as a performance metric or a performance indicator. Following Riquelme, von Lücken, and Baran [71], there are three core qualitative aspects that a multi-objective performance metric inherently tries to quantify.

1. **Convergence.** This aspect is concerned with the accuracy of the approximation, namely, how "close" the approximate front is to the true Pareto front. This aspect requires the

[2]Extending this result to the infinite setting is non-trivial. Specifically, we would have to deal with some measurability issues involved with extending Definition 2.4.

specification of some notion of distance in the objective space, which can be especially challenging when the objectives describe quantities that are defined on different scales.

2. **Diversity.** This aspect is concerned with the distribution and spread of the points in the approximate front. The distribution is related to the relative distance among the points, whereas the spread considers the range of the objectives that are covered by the points. As illustrated in Figure 2, a set can have a good distribution, but a poor spread and vice versa.

3. **Cardinality.** This aspect refers to the number of points in the approximate front. Naturally, a larger number of points is typically preferred.

In Figure 2, we illustrate these three qualities in the two-objective setting. As observed in this example, an approximation can do well on one of these aspects and poorly on another. In essence, all performance metrics in multi-objective optimisation work by quantifying some or all of these aspects and then striking a suitable balance between them. On the surface it appears that there are many different ways to design a performance metric which strikes a balance among these three qualitative properties. For instance, a recent survey [2] identified over fifty different performance metrics that have been documented in the literature. In this paper, we will focus our attention on the *R2 utilities*, which form a general class of utility functions that are inspired by the R2 metric [41]. This family turns out to possess many desirable properties and contains many of the most often used performance metrics as special cases—more details are given in Section 3.

Other useful properties. We will now define two other desirable properties which we will need later on. The first definition is the monotone property, which states that a utility function is non-decreasing in the sense that adding more points to a set does not decrease its utility. The second definition is the diminishing returns property, which states that the gain in utility is less for larger sets. In this paper, we will say a set function, $U : \mathbb{B}(\Omega) \to \mathbb{R}$, is *submodular* if it satisfies the diminishing returns property. We will consider the setting where the ground set is the space of vectors: $\Omega = \mathbb{R}^M$. Note that this definition of submodularity differs subtly from the conventional setting [3, 50], which additionally assumes that the ground set is finite.

Definition 2.7 (Monotone) *For a ground set Ω, the set function $U : \mathbb{B}(\Omega) \to \mathbb{R}$ is monotone if for any sets $A, B \in \mathbb{B}(\Omega)$ with $A \subseteq B$ we have that $U(A) \leq U(B)$.*

Definition 2.8 (Diminishing returns) *For a ground set Ω, the set function $U : \mathbb{B}(\Omega) \to \mathbb{R}$ satisfies the diminishing property if and only if for any sets $A, B \in \mathbb{B}(\Omega)$ with $A \subseteq B$ and $\mathbf{c} \in \Omega$ we have that $U(A \cup \{\mathbf{c}\}) - U(A) \geq U(B \cup \{\mathbf{c}\}) - U(B)$.*

3 R2 Utilities

The family of R2 utilities offers a natural way to connect between the scalarisation and utility perspectives of multi-objective optimisation, by defining a utility function based on a family of scalarisation functions. In particular, given a family of scalarisation functions $\{s_{\boldsymbol{\theta}} : \mathbb{R}^M \to \mathbb{R} : \boldsymbol{\theta} \in \Theta\}$ and a probability density $p(\boldsymbol{\theta}) \geq 0$, the corresponding R2 utility function is defined as the average maximum scalarised value

$$U(Y) = \mathbb{E}_{p(\boldsymbol{\theta})}[S_{\boldsymbol{\theta}}(Y)] = \mathbb{E}_{p(\boldsymbol{\theta})}\left[\max_{\mathbf{y} \in Y} s_{\boldsymbol{\theta}}(\mathbf{y})\right] \tag{8}$$

for some set of vectors $Y \in \mathbb{B}(\mathbb{R}^M)$, where $\mathbb{E}_{p(\boldsymbol{\theta})}[\cdot]$ denotes the expectation with respect to the density $p(\boldsymbol{\theta})$. Intuitively, the maximum scalarised value, described by the set function,

$S_{\boldsymbol{\theta}} : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$,

$$S_{\boldsymbol{\theta}}(Y) = \max_{\mathbf{y} \in Y} s_{\boldsymbol{\theta}}(\mathbf{y}),$$

evaluates the quality of targetting one point corresponding to the scalarised optimisation problem (2). By taking a weighted average over these problems, the R2 utility quantifies some notion of average quality when we are interested in targetting multiple points. To keep the discussion focussed, we will assume throughout this work that the R2 utilities of interest are well-defined in the sense that integrals over the scalarisation parameters exist and are finite. For convenience, we will denote the maximum feasible scalarised value and utility by $S_{\boldsymbol{\theta}}(f(\mathbb{X})) := \max_{\mathbf{x} \in \mathbb{X}} s_{\boldsymbol{\theta}}(f(\mathbf{x}))$ and $U(f(\mathbb{X})) := \mathbb{E}_{p(\boldsymbol{\theta})}[\max_{\mathbf{x} \in \mathbb{X}} s_{\boldsymbol{\theta}}(f(\mathbf{x}))]$, respectively.

R2 metric. In the original work by Hansen and Jaszkiewicz [41], the R2 metric, $I^{\mathrm{R2}} : \mathbb{B}(\mathbb{R}^M) \times \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$, was defined as the difference in R2 utility between a set of objectives $Y \in \mathbb{B}(\mathbb{R}^M)$ and a user-specified reference set $Y_R \in \mathbb{B}(\mathbb{R}^M)$, that is

$$I^{\mathrm{R2}}(Y, Y_R) = U(Y_R) - U(Y). \tag{9}$$

In other words, we can interpret the R2 metric as a notion of regret for the R2 utility. Consequently, all the results that we develop in what follows for the R2 utility can also be applied for the R2 metric. Specifically, in Section 4, we study the problem of maximising an R2 utility, but equivalently we could have considered the problem of minimising an R2 metric because the initial term $U(Y_R) \in \mathbb{R}$ is simply a constant.

Qualitative aspects. The R2 utilities turn out to be a very interpretable class of multi-objective performance metrics. In particular, we can relate each component of these utility functions with the three main qualitative aspects that we described earlier in Section 2.2.

1. **Convergence.** This aspect is determined by the choice of scalarisation function, which assesses the quality of each scalarised optimisation problem. To see this clearly, we can rewrite the difference between the total possible utility and the attained utility by the following expectation:

   $$U(f(\mathbb{X})) - U(f(X)) = \mathbb{E}_{p(\boldsymbol{\theta})}[S_{\boldsymbol{\theta}}(f(\mathbb{X})) - S_{\boldsymbol{\theta}}(f(X))].$$

   The difference within this expectation denotes the regret for each scalarised problem. Notably, a small regret indicates a high convergence for the scalarised problem. As the maximum utility, $U(f(\mathbb{X})) \in \mathbb{R}$, is simply a constant, we can interpret large utility values as an indication of high convergence.

2. **Diversity.** This aspect is determined by the probability distribution, which controls the relative importance placed on each scalarised optimisation problem. As a result, we can potentially capture and assess different notions of spread and distribution by altering this density appropriately.

3. **Cardinality.** This aspect is determined by the max operation which ensures the utility function is monotone and therefore the addition of more points will never decrease the utility. Note that the utility function would also be monotone if we replaced the max operation with the sum operation. The main shortcoming of the summing approach is that it favours a larger set containing many poor performing points over a smaller set of high performing points.

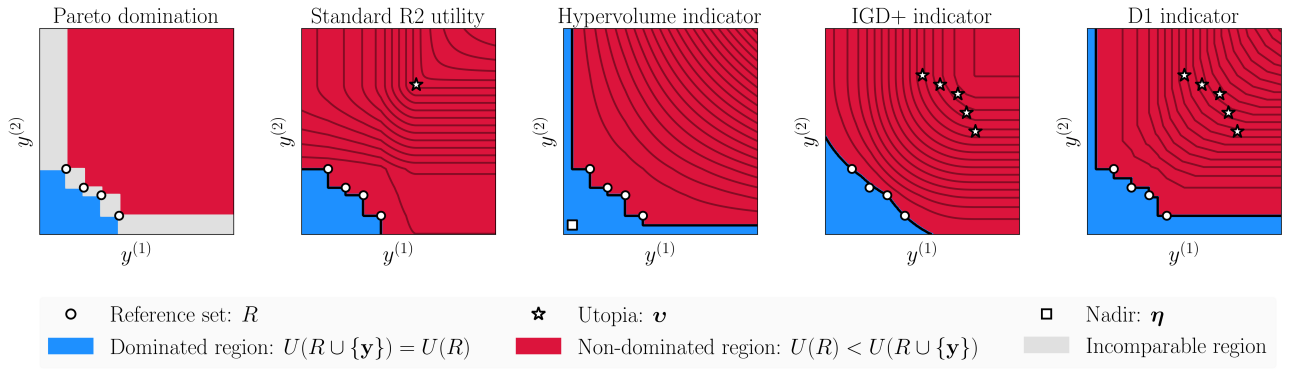

Figure 3: *A comparison of the improvement regions based on the standard Pareto partial ordering over sets and some different R2 utilities in the two-objective setting. For the IGD+ utility, we set $p = 2$ and $q = 1$, whilst for the D1 utility we set $\mathbf{w} = (1/M, \ldots, 1/M)$.*

Useful properties. Another useful result about the R2 utility functions is that they satisfy both the monotone property and the diminishing returns property—we state this result in Proposition 3.1 and prove it in Appendix A.2. Intuitively, both of these properties seem sensible for a performance metric. The monotone property suggests that the quality of a set of vectors should not deteriorate as we add more points, whilst the submodularity property suggests that the improvement in adding more points should diminish with respect to the size of the set.

Another desirable property for an R2 utility function is Pareto compliancy. This property turns out to be dependent only on the choice of scalarisation functions. In Proposition 3.2, we give a sufficient condition for weak Pareto compliancy, which we prove in Appendix A.3. Specifically, if the scalarisation functions preserves the weak Pareto partial ordering, then so does the corresponding R2 utility. Establishing a sufficient condition for strict Pareto compliancy turns out to be non-trivial because we would have to make additional measurability assumptions. More precisely, we would have to show that the set of scalarisation parameters $\{\boldsymbol{\theta} \in \Theta : S_{\boldsymbol{\theta}}(A) > S_{\boldsymbol{\theta}}(B)\}$ has non-zero measure for any sets $A, B \in \mathbb{B}(\mathbb{R}^M)$ such that $A \succ B$.

Proposition 3.1 *All R2 utilities are monotone and submodular.*

Proposition 3.2 *If the scalarisation functions $s_{\boldsymbol{\theta}} : \mathbb{R}^M \to \mathbb{R}$ are monotonically increasing for all $\boldsymbol{\theta} \in \Theta$, then the resulting R2 utility is weakly Pareto compliant for any choice of probability distribution $p(\boldsymbol{\theta}) \geq 0$.*

Remark 3.1 (Single-objective setting) *The classical objective for single-objective optimisation is also an R2 utility, namely, $U[Y] = \max_{y \in Y} y$. As a result, all of the properties that we have shown here also applies to the standard single-objective criterion.*

3.1 Special cases

We now highlight the fact that many of the most widely-used performance metrics in multi-objective optimisation can be viewed as special cases of R2 utilities. In particular, from a recent survey [71] of the most used performance metrics, we see that the standard R2 metric, hypervolume indicator, inverted generational distance and D1 indicator were among the top in the list. All of these utilities turn out to be special cases of R2 utilities. In Figure 3, we illustrate the contours of these R2 utilities for a simple two-dimensional example.

3.1.1 Standard R2 metric

In the original work by Hansen and Jaszkiewicz [41], the R2 metric (9) was defined explicitly using the average weighted Chebyshev distance (5)

$$U^{\mathrm{R2}}(Y) = \mathbb{E}_{\mathbf{w}\sim\mathrm{Uniform}(\Delta^{M-1})}\left[\max_{\mathbf{y}\in Y} s^{\mathrm{Chb}}_{(\boldsymbol{\upsilon},\mathbf{w})}(\mathbf{y})\right] \tag{10}$$

for some user-specified reference point $\boldsymbol{\upsilon} \in \mathbb{R}^M$. The original motivation behind the use of the Chebyshev scalarisation function came from the well-known result that we described in Remark 2.2. The use of a uniform distribution over the simplex weights was mostly done for convenience. As noted by other researchers [90], potentially we could have used some other weight distributions if we were interested in assessing specific parts of the Pareto front.

Augmented variant. A limitation of the Chebyshev scalarisation function is that it also targets weakly Pareto optimal solutions, whereas in many cases we are only interested in the strictly Pareto optimal solutions. One possible strategy to avoid these weakly Pareto optimal solution is to incorporate an L^1-penalty into the Chebyshev scalarisation function [58, Part 2, Section 3.4.5]. For instance, Zitzler et al. [101] considered using the augmented Chebyshev function (6) instead.

Weak Pareto compliancy. As a result of Proposition 3.2, we can conclude that both of these variants of the standard R2 utility are weakly Pareto compliant because their corresponding scalarisation functions are monotonically increasing.

3.1.2 Hypervolume indicator

One of the most popular performance criterion in multi-objective optimisation is the hypervolume indicator [102]. The hypervolume indicator, $I^{\mathrm{HV}} : \mathbb{B}(\mathbb{R}^M) \times \mathbb{R}^M \to \mathbb{R}$, is a function that computes the volume of the dominated region between a set of objectives $Y \in \mathbb{B}(\mathbb{R}^M)$ and a reference point $\boldsymbol{\eta} \in \mathbb{R}^M$, that is

$$I^{\mathrm{HV}}(Y, \boldsymbol{\eta}) = \int_{\mathbb{R}^M} \mathbb{1}[\mathbf{z} \in \cup_{\mathbf{y}\in Y}[\boldsymbol{\eta}, \mathbf{y}]]d\mathbf{z}, \tag{11}$$

where $\mathbb{1}$ is the indicator function.

Strict Pareto compliancy. A larger hypervolume is preferred over a smaller one. The hypervolume indicator is known to be a strictly Pareto compliant utility [103], when we restrict it to sets which lie in the space that strictly Pareto dominates the reference point: $Y \in \mathbb{B}(\mathbb{D}_{\succ}(\{\boldsymbol{\eta}\})) \subset \mathbb{B}(\mathbb{R}^M)$.

Hypervolume scalarisation function. Early work on the R2 indicator [10] identified many similarities between the behaviour of the standard R2 utility and that of the hypervolume indicator. As the hypervolume indicator grew in its popularity, a lot of work has been focussed on identifying faster and more efficient strategies to compute this indicator. After many different efforts over the last decade [45, 54, 78, 24, 96], it was slowly realised that we can write the hypervolume indicator as an R2 utility. This result is summarised in Proposition 3.3 and the proof is presented in the references [78, Section 3.2], [24, Section 2], and [96, Lemma 5]. Intuitively, the main idea of the proof is to rewrite the hypervolume integral using polar coordinates. Under this formulation, the hypervolume scalarisation function (12) works by computing the volume contribution over each angle and the resulting R2 utility computes the total integral of all of these contributions.

Proposition 3.3 (Hypervolume utility) *The hypervolume indicator eq. (11) can be written as an R2 utility*

$$U^{\text{HV}}(Y) := I^{\text{HV}}(Y, \boldsymbol{\eta}) = \mathbb{E}_{\boldsymbol{\lambda} \sim \text{Uniform}(\mathcal{S}_+^{M-1})} \left[\max_{\mathbf{y} \in Y} s^{\text{HV}}_{(\boldsymbol{\eta}, \boldsymbol{\lambda})}(\mathbf{y}) \right]$$

for any $Y \in \mathbb{B}(\mathbb{R}^M)$ and $\boldsymbol{\eta} \in \mathbb{R}^M$, where the hypervolume scalarisation function is defined as the following transformation of the Chebyshev scalarisation function (5),

$$s^{\text{HV}}_{(\boldsymbol{\eta}, \boldsymbol{\lambda})}(\mathbf{y}) = \frac{\pi^{M/2}}{2^M \Gamma(M/2+1)} \min_{m=1,\dots,M} \left(\frac{\max(0, y^{(m)} - \eta^{(m)})}{\lambda^{(m)}} \right)^M, \tag{12}$$

with $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt$ denoting the Gamma function and $\boldsymbol{\lambda} \in \mathcal{S}_+^{M-1} := \{\mathbf{y} \in \mathbb{R}^M_{>0} : ||\mathbf{y}||_{L^2} = 1\}$ denoting a weight parameter that is distributed uniformly over the space of positive unit vectors.

Monotone and submodular. The hypervolume indicator is known to be a monotone and submodular set function [86]. These two properties have been instrumental in the development of many results and algorithms concerning the hypervolume optimisation problem [40]. In Proposition 3.1, we generalised this result and demonstrated that these two properties actually hold for any R2 utility. As a result, many of these existing ideas can now be applied to the R2 utility optimisation problem (7)—more details are given in Section 4.

3.1.3 Inverted generational distance

Another popular performance metric in multi-objective optimisation is the inverted generational distance (IGD) [14]. This performance metric tries to measure some notion of distance between a finite set of objectives $Y \in \mathbb{B}(\mathbb{R}^M)$ and a finite set of ideal points $\Upsilon \in \mathbb{B}(\mathbb{R}^M)$. Using the formulation by Schutze et al. [76], the IGD indicator, $I^{\text{IGD}_{p,q}} : \mathbb{B}(\mathbb{R}^M) \times \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$, is equal to

$$I^{\text{IGD}_{p,q}}(Y, \Upsilon) = \left(\frac{1}{|\Upsilon|} \sum_{\boldsymbol{\upsilon} \in \Upsilon} \left(\min_{\mathbf{y} \in Y} ||\boldsymbol{\upsilon} - \mathbf{y}||_{L^p} \right)^q \right)^{1/q} \tag{13}$$

for some norms $p, q \geq 1$. A smaller IGD is preferred because it indicates that the set of objective vectors is close to the ideal reference set. The IGD is minimised for any set of vectors containing the ideal points: $Y \supseteq \Upsilon$. It is common to set the outer norm to $q = 1$, which means that the resulting IGD indicator can be interpreted as the average L^p-norm to the set of ideal points. On the other hand, the inner norm is commonly set to $p = 2$.

Original formulation. The traditional definition of the IGD indicator assumes that the average is computed outside of the L^q-norm: $|\Upsilon|^{1/q-1} I^{\text{IGD}_{p,q}}(Y, \Upsilon)$. As described by Schutze et al. [76, Section 3], the additional multiplicative factor in the traditional definition can lead to some undesirable properties. Note that there is no difference between these two formulations when $q = 1$.

Average Lp distance. By applying a transformation on the Lp scalarisation function (4), we can rewrite the IGD indicator as a transformation of an R2 utility. We summarise this result in Proposition 3.4 and prove it in Appendix A.5.

Proposition 3.4 (IGD utility) *The IGD indicator (13) can be written as a transformation of an R2 utility*

$$U^{\text{IGD}_{p,q}}(Y) := -(I^{\text{IGD}_{p,q}}(Y, \Upsilon))^q = \frac{1}{|\Upsilon|} \sum_{\boldsymbol{\upsilon} \in \Upsilon} \max_{\mathbf{y} \in Y} s^{\text{IGD}_{p,q}}_{\boldsymbol{\upsilon}}(\mathbf{y})$$

for any $Y, \Upsilon \in \mathbb{B}(\mathbb{R}^M)$ *and* $p, q \geq 1$*, where* $s_{\boldsymbol{\upsilon}}^{\mathrm{IGD}_{p,q}}(\mathbf{y}) = -||\boldsymbol{\upsilon} - \mathbf{y}||_{L^p}^q$ *is the IGD scalarisation function.*

Weak Pareto compliancy. Since IGD scalarisation function is not monotonically increasing over the whole objective space, this means that the IGD indicator (and utility) is not necessarily weakly Pareto compliant. To address this problem, Ishibuchi et al. [44] proposed the IGD+ indicator, which works by truncating the difference within the L^p-norm at zero: $s_{\boldsymbol{\upsilon}}^{\mathrm{IGD}_{p,q}+}(\mathbf{y}) = -||\max(\boldsymbol{\upsilon} - \mathbf{y}, \mathbf{0}_M)||_{L^p}^q$, where the maximum is computed element-wise. This modification ensures that the scalarisation function is monotonically increasing over the entire objective space and therefore the resulting indicator (and utility) is weakly Pareto compliant.

3.1.4 D1 indicator

Another popular performance metric is the D1 indicator [15]. Similar to the IGD, the goal of the D1 indicator is to measure the distance between a finite set of objectives $Y \in \mathbb{B}(\mathbb{R}^M)$ and a finite set of ideal points $\Upsilon \in \mathbb{B}(\mathbb{R}^M)$. The D1 indicator, $I^{\mathrm{D1}} : \mathbb{B}(\mathbb{R}^M) \times \mathbb{B}(\mathbb{R}^M) \times \Delta^{M-1} \to \mathbb{R}$ is defined as the average weighted Chebyshev distance between the set of points

$$I^{\mathrm{D1}}(Y, \Upsilon, \mathbf{w}) = \frac{1}{|\Upsilon|} \sum_{\boldsymbol{\upsilon} \in \Upsilon} \min_{\mathbf{y} \in Y} \max_{m=1,\ldots,M} w^{(m)}(\upsilon^{(m)} - y^{(m)}), \tag{14}$$

where $\mathbf{w} \in \Delta^{M-1}$ denotes some weight vector. A smaller D1 indicator is preferred because it indicates that the set of objectives is close to the set of ideal points. It is common to see uniform weights being used for each objective: $\mathbf{w} = \mathbf{1}_M / M \in \Delta^{M-1}$.

Average Chebyshev distance. The negative D1 indicator can be obtained as an R2 utility using the Chebyshev scalarisation function (5) and a uniform distribution over the set of ideal points. We summarise this result in Proposition 3.5 and prove it in Appendix A.6.

Proposition 3.5 (D1 utility) *The D1 indicator* (14) *can be written as a transformation of an R2 utility*

$$U^{\mathrm{D1}}(Y) := -I^{\mathrm{D1}}(Y, \Upsilon, \mathbf{w}) = \frac{1}{|\Upsilon|} \sum_{\boldsymbol{\upsilon} \in \Upsilon} \max_{\mathbf{y} \in Y} s_{(\boldsymbol{\upsilon}, \mathbf{w})}^{\mathrm{Chb}}(\mathbf{y})$$

for any $Y, \Upsilon \in \mathbb{B}(\mathbb{R}^M)$ *and* $\mathbf{w} \in \Delta^{M-1}$.

Weak Pareto compliancy. The D1 indicator (and utility) is weakly Pareto compliant because the Chebyshev scalarisation function is monotonically increasing.

3.2 Discussion

The overall goal of multi-objective optimisation is to identify an approximation to the Pareto optimal points. The scalarisation and utility perspectives of multi-objective optimisation offers a strategy to obtain this approximation by recasting the original problem as one which is more amenable to standard optimisation strategies. The R2 utilities unify both of these approaches into one in which a decision maker only needs to specify a family of scalarisation functions and a probability distribution to go along with it. There is no clear-cut answer as to what utilities, scalarisation functions and probability distribution we should use in practice. This choice naturally depends on the goals and preferences of the decision maker, which can in general be hard to elicit [92].

In many cases, decision makers often resort to the use of popular off-the-shelf utility functions such as those listed in Section 3.1. These utility functions are useful for a general decision maker, who is simply interested in obtaining some approximation to the Pareto optimal points, but they might be insufficient for a more specialised user who has specific preferences they want to encode in their approximation. For example, suppose a decision maker has a lexicographical preference on the objectives, which states that some objectives are more important to optimise than others. The previously listed performance metrics might be inadequate for this setting because they do not explicitly encode this preference into their utility values. A more suitable approach would be to design an R2 utility which does encode this preference. For instance, we could use a weight-based scalarisation function such as the Chebyshev scalarisation function (5) and devise a weight distribution that satisfies this lexicographical constraint.

In this paper, we have highlighted only a small selection of R2 utilities in order to showcase the usefulness of the methodology. There are numerous possibilities that we have not discussed. For instance, we could consider R2 utilities defined using reference-line based scalarisation functions [16, 95, 74, 60]. Similarly, we could consider R2 utilities based on transformations of existing scalarisation functions such as the linear or Chebyshev scalarisation function [57, 45, 54, 8, 13].

4 Optimisation problem

Suppose that we have identified an R2 utility which adequately reflects the decision maker's preferences. We will now consider solving the corresponding utility optimisation problem

$$\max_{X \subseteq \mathbb{X}, |X| \leq P} \mathbb{E}_{p(\boldsymbol{\theta})} \left[\max_{\mathbf{x} \in X} s_{\boldsymbol{\theta}}(f(\mathbf{x})) \right] \tag{15}$$

for some $P > 0$. As all R2 utilities are monotone and submodular, this utility optimisation problem turns out to be an instance of a cardinality-constrained submodular optimisation problem, which is known to be NP-hard in general [3, 50]. Nevertheless, we can typically solve these problems approximately using greedy algorithms. These greedy approaches satisfy the well-known performance guarantee by Nemhauser, Wolsey and Fisher [61]. In Section 4.1, we will review how these greedy strategies can be used to approximately solve our utility optimisation problem. We will then move on to Section 4.2, where we will extend the analysis of these greedy strategies to the Bayesian optimisation setting.

In order to prove some later results, we will now assume for convenience that the R2 utility function of interest is non-negative and bounded over the feasible space. Note that any R2 utility can be made non-negative over a bounded space by simply adding a large enough constant.

Assumption 4.1 (Non-negative and bounded) *Consider an objective function $f : \mathbb{X} \to \mathbb{R}^M$ and an R2 utility $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$. We assume that there exists a constant $C > 0$ such that $0 \leq s_{\boldsymbol{\theta}}(f(\mathbf{x})) \leq C$ for all $\mathbf{x} \in \mathbb{X}$ and $\boldsymbol{\theta} \in \Theta$.*

4.1 Greedy algorithms

To solve the R2 utility optimisation problem (15) approximately, we will consider the use of greedy algorithms. These algorithms start with an empty set of inputs $X_0 = \{\}$, which is then incrementally augmented according to some greedy heuristic: $X_n = X_{n-1} \cup \{\mathbf{x}_n\}$ for $n = 1, \ldots, N$, where $\mathbf{x}_n \in \mathbb{X}$ is the point that is greedily selected at the n-th round. The final set X_N is then suggested as a solution, where N is the budget of function evaluations.

4.1.1 Standard greedy approach

In the traditional greedy strategy, we pick the input which achieves the largest utility improvement; that is,

$$\mathbf{x}_{n+1} \in \arg\max_{\mathbf{x}\in\mathbb{X}}(U(f(X_n \cup \{\mathbf{x}\})) - U(f(X_n))). \tag{16}$$

This greedy method comes with the well-known performance guarantee by Nemhauser et al. [61], which we recall in Theorem 4.1. The main consequence of this guarantee is that after P rounds of the greedy algorithm, the utility obtained is guaranteed to be at least $(1 - e^{-1})$ times the best possible utility for this problem. Moreover, if we are allowed to perform even more rounds, then this approximation becomes even better. For instance, after γP rounds, the factor is equal to $(1 - e^{-\gamma})$, which becomes ever closer to one as γ grows.

Theorem 4.1 (Greedy guarantee) *(Nemhauser, Wolsey and Fisher [61]) Consider an objective function $f : \mathbb{X} \to \mathbb{R}^M$ and an R2 utility $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$, satisfying Assumption 4.1. Let $\{X_n\}_{n\geq 1}$ be the greedily selected inputs, then for all positive integers P and N,*

$$U(f(X_N)) \geq (1 - e^{-N/P}) \max_{X\subseteq\mathbb{X}:|X|\leq P} U(f(X)).$$

4.1.2 Approximate greedy approach

The R2 utility can only be evaluated exactly in simple settings such as the case where the set of scalarisation parameters is finite. In general, we rely on approximations of the R2 utility such as the one obtained via Monte Carlo estimation, that is

$$\hat{U}_J(Y) = \frac{1}{J}\sum_{j=1}^{J} \max_{\mathbf{y}\in Y} s_{\boldsymbol{\theta}_j}(\mathbf{y}), \tag{17}$$

where $\boldsymbol{\theta}_j \sim p(\boldsymbol{\theta})$ denotes the J independent samples of the scalarisation parameter. Equipped with this estimate, the approximate greedy strategy proceeds by maximising the estimate of the utility improvement

$$\mathbf{x}_{n+1} \in \arg\max_{\mathbf{x}\in\mathbb{X}}(\hat{U}_J(f(X_n \cup \{\mathbf{x}\})) - \hat{U}_J(f(X_n))). \tag{18}$$

Note that the estimate of the R2 utility (17) could be fixed throughout the duration of the optimisation procedure or could be re-estimated at every iteration. In addition, we could also let the number of samples change over time as well—for simplicity we only consider the static case where the number of samples is fixed throughout.

Random scalarisation. When we re-estimate the utility at each time and use only one Monte Carlo sample, $J = 1$, then we recover the random scalarisation algorithm. This is a popular strategy that has been used before in the area of Bayesian optimisation [48, 63, 96, 11] and bares some similarity with Thompson sampling strategies [83, 73].

Approximate greedy guarantee. By a simple application of Hoeffding's inequality, we can devise a similar performance guarantee to the one above. The main difference is that we now have an additional additive penalty term, which arises from the error that we incur in performing Monte Carlo estimation at every round. This error naturally decreases as the number of Monte Carlo samples increases. For completeness, we state this result in Proposition 4.1 and prove it in Appendix A.8.1.

Proposition 4.1 (Approximate greedy guarantee) *Consider an objective function $f : \mathbb{X} \to \mathbb{R}^M$ and an R2 utility $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$, satisfying Assumption 4.1. Let $\{X_n\}_{n\geq 1}$ denote the set of inputs identified by the approximately greedy strategy using $J > 0$ Monte Carlo samples, then for any $\delta \in (0,1)$ and any positive integers P and N, the following inequality holds with probability $1-\delta$:*

$$U(f(X_N)) \geq (1 - e^{-N/P}) \max_{X \subseteq \mathbb{X}, |X| \leq P} U(f(X)) - \epsilon(\delta, P, N, J),$$

where the error term is given by $\epsilon(\delta, P, N, J) = \sqrt{\frac{2}{J}\log(\frac{4N}{\delta})} \sum_{n=1}^{N} C_{n-1}(1-\frac{1}{P})^{N-n}$, with $C_n := \sup_{\boldsymbol{\theta}\in\Theta}(S_{\boldsymbol{\theta}}(f(\mathbb{X})) - S_{\boldsymbol{\theta}}(f(X_n)))$ denoting the maximum scalarised regret at round $n = 1, \ldots, N-1$ and $C_0 := C$.

Number of samples. As expected, the error decreases to zero as we increase the number of Monte Carlo samples: $\epsilon(\delta, P, N, J) \to 0$ as $J \to \infty$. Note that we can bound this error by an input-independent constant $\epsilon(\delta, P, N, J) \leq CP\sqrt{2\log(4N/\delta)/J}$. As a result, if we wanted to achieve a specific error of $\epsilon > 0$ with probability $1-\delta$, then we could set the number of Monte Carlo samples to be $J = \lceil 2C^2P^2\log(4N/\delta)/\epsilon^2 \rceil$. This number is arguably much larger than what is required in practice and could be easily improved by a more careful application of some stronger concentration inequalities. Nevertheless, this result gives a useful indication of what factors affect the performance guarantee. For instance, we see that the number of objectives M only enters into the calculation via the upper bound C, which is determined by both the objective function and the R2 utility.

Noisy submodular optimisation. The idea of solving a submodular optimisation problem when there is uncertainty in the function evaluations has been addressed before in the area of submodular optimisation under noise. For example, Kempe, Kleinbery and Tardos [47] considered an influence optimisation problem in a social network, whilst Singla, Tschiatschek, and Krause [79] considered some applications for crowdsourced image collection. The fact that we incur an additive error in the performance bound is known to be a direct consequence of the uncertainty in the function evaluation. In our case, this error arises from our Monte Carlo estimate of the R2 utility, which can be easily reduced at the expense of more computational power.

We now illustrate a simple example of this approximate greedy strategy on the hypersphere problem in Example 4.1. In particular, we illustrate empirically how the observed performance changes in this problem when we vary the number of Monte Carlo samples and the number of objectives.

Example 4.1 (Hypersphere) *Consider the R2 utility optimisation problem (15), where the feasible objective space $f(\mathbb{X})$ is the M-dimensional hypersphere and the performance metric U is the standard R2 utility (10) with the reference point set to $\boldsymbol{\upsilon} = (1, \ldots, 1) \in \mathbb{R}^M$. For this problem, the Pareto front of interest is just the surface of the hypersphere lying in the non-negative orthant: $\mathbb{Y}^* = \mathcal{S}_+^{M-1}$, which we defined in Section 3.1.2.*

In Figure 4, we illustrate one run of the approximate greedy strategy on this problem for the $(M = 2)$-dimensional setting with different numbers of Monte Carlo samples J. We plot the result in the scalarisation parameter space. To elaborate, at each time n, we picked a solution $\mathbf{x}_n \in \mathbb{X}$ according to the approximate greedy heuristic (18). This solution is associated with a surface in the scalarisation parameter space: $\{s^{Chb}_{(\boldsymbol{\upsilon},(w,1-w))}(f(\mathbf{x}_n)) : w \in [0,1]\}$. In this illustrative example, the scalarisation parameter can be parametrised using a one-dimensional weight, $\boldsymbol{\theta} = w$, which lies in the unit interval $\Theta = [0,1]$. Geometrically, we can interpret the

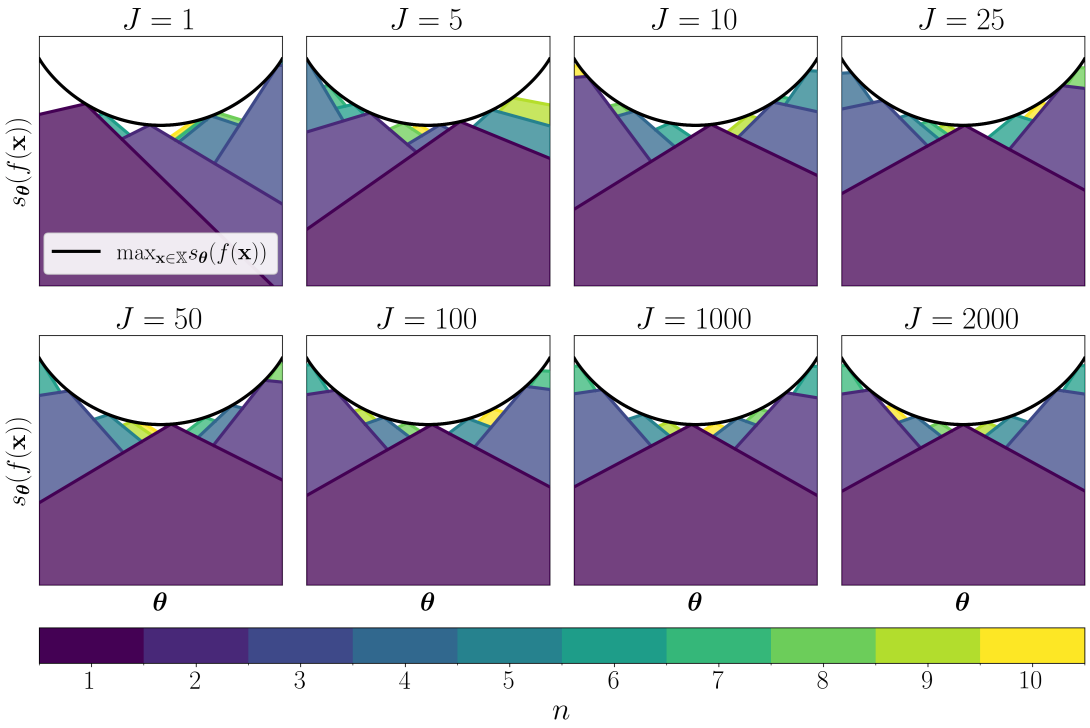

Figure 4: *A visual comparison of the approximate greedy strategy with different number of Monte Carlo samples on the hypersphere problem (Example 4.1).*

R2 utility of this solution, $U(\{f(\mathbf{x}_n)\}) \in \mathbb{R}$, as the area underneath this curve. Moreover, we can interpret the utility of a collection of these solutions as the area underneath the "envelope" of these curves. In the figure, we coloured the gain in utility at each time based on the iteration number n. We see clearly that the gain in utility is much larger at earlier rounds, which is a direct consequence of the diminishing returns property. In addition, we see how the number of Monte Carlo samples effects the points that are selected. Specifically, when the number of samples is small, then there is a large variance in the points that are selected—however, this variance however decreases when the number of samples increases.

In Figure 5, we illustrate the performance of the approximate greedy strategy when we vary the number of objectives as well as the number of Monte Carlo samples. As expected from the diminishing return property, we see that the improvement in utility drops quickly as the number of iterations increases. Moreover, we see a small improvement in performance when we increase the number of Monte Carlo samples.

4.2 Bayesian optimisation

In the previous section, we discussed how greedy strategies can be used to solve the R2 utility optimisation problem (15). All of these strategies rely on solving a sequence of optimisation problems that depend on exact evaluations of the underlying vector-valued objective function. In this section, we extend these ideas to the black-box optimisation setting, where the function evaluations are assumed to be expensive and potentially subject to noise. We focus our study on the popular Bayesian optimisation algorithm, which relies on the use of surrogate models in order to decide which points to evaluate. We summarise the main steps of this optimisation procedure in what follows and provide the pseudo-code in Algorithm 1—for a more in-depth overview, consult the recent review by Garnett [37].

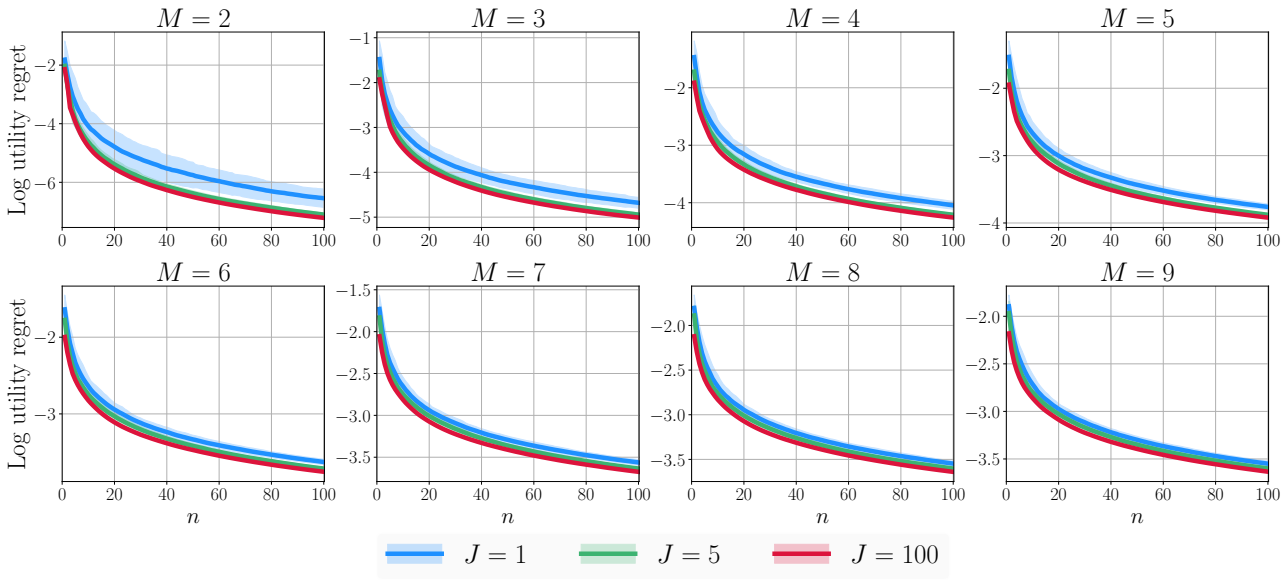

Figure 5: *A performance comparison of the approximate greedy strategy when varying the number of Monte Carlo samples and the number of objectives on the hypersphere problem (Example 4.1). The mean and two standard deviations of the log utility regret,* $\log(U(f(\mathbb{X})) - U(f(X_N)))$*, obtained from 100 independent runs, are plotted. In each experiment, the Pareto front and the standard R2 utility were approximated using* 10^5 *points and samples.*

Modelling. At time n, we have a data set of inputs and outputs, $\mathcal{D}_n = \mathcal{D}_0 \cup \{(\mathbf{x}_t, \mathbf{y}_t)\}_{t=1,\dots,n}$. Given this data set, the Bayesian optimisation routine proceeds by building a probabilistic surrogate model of the objective function. At a high level, this model can be constructed by adopting either a Bayesian perspective or a frequentist perspective.

(i) In the Bayesian paradigm, we assume a prior distribution on the objective function $p(f|\mathcal{D}_0)$ and a likelihood $p(\mathbf{y}|\mathbf{x}, f)$ on the observations. Then by standard conditioning we can obtain a posterior distribution over the function $p(f|\mathcal{D}_n) \propto p(f|\mathcal{D}_0) \prod_{t=1}^{n} p(\mathbf{y}_t|\mathbf{x}_t, f)$.

(i) In the frequentist paradigm, we directly construct an estimate of the function $\hat{f} : \mathbb{X} \to \mathbb{R}^M$. This is often done by minimising some empirical loss function which depends on the observed data points.

To keep the discussion general, we denote the probabilistic surrogate model, at time n, under both settings, by $p(\hat{f}|\mathcal{D}_n)$. In the Bayesian setting, this can be interpreted as a posterior distribution of the model at time n, whilst in the frequentist setting, this can be interpreted as a point mass on our function estimate at time n.

Acquisition. Equipped with a surrogate model, the Bayesian optimisation procedure then proceeds to select a new input to query by maximising some acquisition function $\alpha : \mathbb{X} \to \mathbb{R}$,

$$\mathbf{x}_{n+1} \in \arg\max_{\mathbf{x} \in \mathbb{X}} \alpha(\mathbf{x}|\mathcal{D}_n).$$

This acquisition function is designed to strike a balance between exploring new parts of input space and exploiting the regions of the input space that are known to perform well. In multi-objective Bayesian optimisation, practitioners often rely on the use of scalarisation functions or utility functions in order to navigate this trade-off. In this paper, we show how it is possible to unify these existing approaches by considering a general class of acquisition functions based on the R2 utility.

Algorithm 1

Function BAYESIANOPTIMISATION($\mathcal{D}_0$, N, $p(\hat{f}|\mathcal{D}_0)$, α)**:**

// Initial data set $\mathcal{D}_0$.
// Number of function evaluations N.
// Initial model $p(\hat{f}|\mathcal{D}_0)$.
// The acquisition function $\alpha : \mathbb{X} \to \mathbb{R}$.
for $n = 0, \ldots, N-1$ **do**
  Acquire the new input: $\mathbf{x}_{n+1} \in \arg\max_{\mathbf{x}\in\mathbb{X}} \alpha(\mathbf{x}|\mathcal{D}_n)$.
  Evaluate the input: $\mathbf{y}_{n+1} \approx f(\mathbf{x}_{n+1})$.
  Update the data set $\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{(\mathbf{x}_{n+1}, \mathbf{y}_{n+1})\}$.
  Update the model: $p(\hat{f}|\mathcal{D}_{n+1})$.
end
return The final data set $\mathcal{D}_N$.

4.2.1 Adjusted expected utility improvement

We now introduce a general family of greedy acquisition functions called the adjusted expected utility improvement (AEUI). An acquisition function belongs to this family if it can be written in the form

$$\alpha^{\text{AEUI}}(\mathbf{x}|\mathcal{D}_n) = \alpha^{\text{EUI}}(\mathbf{x}|\mathcal{D}_n) + \mathcal{A}_\delta(\mathbf{x}|\mathcal{D}_n), \tag{19}$$

where $\alpha^{\text{EUI}} : \mathbb{X} \to \mathbb{R}$ denotes the expected utility improvement (EUI) acquisition function,

$$\alpha^{\text{EUI}}(\mathbf{x}|\mathcal{D}_n) = \mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[U(\hat{f}(X_n \cup \{\mathbf{x}\})) - U(\hat{f}(X_n))], \tag{20}$$

and $\mathcal{A}_\delta : \mathbb{X} \to \mathbb{R}_{\geq 0}$ denotes the adjustment function, which is a non-negative function that depends on some parameter $\delta \in (0, 1)$.

Expected utility improvement. The EUI acquisition function, can be interpreted as the natural counterpart to the standard greedy heuristic in (16). Instead of computing the greatest utility improvement with respect to the true objective function, we compute the greatest EUI with respect to the surrogate model. Notably, this family of acquisition functions has appeared numerous times before in the literature—see the recent review by Zhan and Xing [94].

Adjustment function. $\mathcal{A}_\delta$ can be interpreted as a function that accounts for the error from estimating the objective function using the surrogate model. Specifically, we will assume that the adjustment function is an upper bound on the absolute error between the expected utility and the actual utility—we formalise this statement below in Assumption 4.2.

Assumption 4.2 (Utility estimate concentration) *Consider an objective function $f : \mathbb{X} \to \mathbb{R}^M$, an R2 utility $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$, a surrogate model $p(\hat{f}|\mathcal{D}_n)$ and a family of adjustment functions $\{\mathcal{A}_\delta : \mathbb{X} \to \mathbb{R}_{\geq 0} : \delta \in (0, 1)\}$. For any $\delta \in (0, 1)$, we assume that the following inequality holds with probability $1 - \delta$:*

$$\left|U(f(X_n \cup \{\mathbf{x}\})) - \mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[U(\hat{f}(X_n \cup \{\mathbf{x}\}))]\right| \leq \mathcal{A}_\delta(\mathbf{x}|\mathcal{D}_n) \tag{21}$$

for all inputs $\mathbf{x} \in \mathbb{X}$.

Remark 4.1 *Note that the family of EUI acquisition functions is a subset of the family of AEUI acquisition functions. More explicitly, to recover the EUI acquisition policy, we can simply set the adjustment function to be any constant function that satisfies Assumption 4.2.*

Remark 4.2 (Batch optimisation) *All of the results that we develop in this section can easily be extended to the batch optimisation setting, where we pick multiple points at a time rather than just one. More precisely, for the batch setting, we would consider the batch AEUI acquisition function*

$$\alpha^{\text{AEUI}}(X|\mathcal{D}_n) = \mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[U(\hat{f}(X_n \cup X)) - U(\hat{f}(X_n))] + \mathcal{A}_\delta(X|\mathcal{D}_n),$$

where $X \in \mathbb{X}^q$ is a batch of $q > 0$ inputs and $\mathcal{A}_\delta(X|\mathcal{D}_n)$ is the adjustment term that satisfies the batch analogue of the concentration inequality described in Assumption 4.2. For ease of exposition, we have intentionally focussed our discussion on only the sequential setting.

4.2.2 Computing the acquisition function

To compute the AEUI acquisition function, we need to be able to evaluate both the EUI acquisition function and the adjustment function.

Estimating the expected utility improvement. In general, the EUI acquisition function is not a tractable quantity because we might not be able to evaluate the utility exactly or be able to compute the expectation over the surrogate model. Nevertheless, if we can generate samples of the scalarisation parameters and surrogate model, then we can compute a Monte Carlo estimate

$$\hat{\alpha}^{\text{EUI}}(\mathbf{x}|\mathcal{D}_n) = \frac{1}{H}\sum_{h=1}^{H} \hat{U}_J(\hat{f}_h(X_n \cup \{\mathbf{x}\})) - \hat{U}_J(\hat{f}_h(X_n)), \tag{22}$$

where $\hat{f}_h(\cdot) \sim p(\hat{f}|\mathcal{D}_n)$ denotes the independent samples from the model, whilst $\hat{U}_J$ denotes the Monte Carlo estimate of the utility described earlier in (17). In order to develop the performance bound in the next section, we will assume that the estimates for the utility based on the surrogate model are non-negative and bounded by the same constant, $C > 0$, which bounds the actual utility values in Assumption 4.1.

Assumption 4.3 (Non-negative and bounded utility estimates) *Consider an R2 utility and a surrogate model $p(\hat{f}|\mathcal{D}_n)$. We assume that there exists a constant $C > 0$ such that $0 \leq s_{\boldsymbol{\theta}}(\hat{f}(\mathbf{x})) \leq C$ for all $\mathbf{x} \in \mathbb{X}$ and $\boldsymbol{\theta} \in \Theta$, almost surely over all possible model samples $\hat{f}(\cdot) \sim p(\hat{f}|\mathcal{D}_n)$.*

Remark 4.3 (Sampling points instead of paths) *Sampling the whole path $\hat{f}_h$ is expensive and unnecessary in practice. We can simplify the estimate of the EUI acquisition function by marginalising out the dependency on the points that are not considered in the integral. As a result, we only need to sample the function at the considered locations, $\hat{f}_h(X_n \cup \{\mathbf{x}\}) \sim p(\hat{f}(X_n \cup \{\mathbf{x}\})|\mathcal{D}_n)$, which is considerably cheaper.*

Computing the adjustment function. The adjustment term quantifies the error in estimating the utility of the objective function. If we are only interested in employing the EUI acquisition function, then we do not have to compute this quantity because it would only contribute an input-independent constant (Remark 4.1). On the other hand, if we wanted to establish an input-dependent adjustment term, then we would need to make some additional assumptions on both the utility function and the model. For example, if we assumed that the scalarisation functions are Lipschitz in the L^2-norm, then we can set the adjustment function to be proportional to any upper bound on the expected L^2-error. To see this, suppose there

exists a constant $L > 0$ such that $|s_{\boldsymbol{\theta}}(\mathbf{y}) - s_{\boldsymbol{\theta}}(\mathbf{y}')| \leq L||\mathbf{y} - \mathbf{y}'||_{L^2}$ for any $\boldsymbol{\theta} \in \Theta$ and $\mathbf{y}, \mathbf{y}' \in \mathbb{R}^M$, then by the triangle inequality we obtain the inequality

$$\left|U(f(X)) - \mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[U(\hat{f}(X))]\right| \leq L\mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}\left[\max_{\mathbf{x}' \in X} ||f(\mathbf{x}') - \hat{f}(\mathbf{x}')||_{L^2}\right],$$

which holds for any finite set of inputs $X \subseteq \mathbb{X}$. Therefore, if we can bound the expected L^2-error of the function estimate at $X = X_n \cup \{\mathbf{x}\}$, then we have a candidate for the adjustment function. For instance, if we adopt a standard Gaussian process [70] regression set-up, similar to the one described by Chowdhury and Gopalan [11], then we can set the adjustment function to

$$\mathcal{A}_\delta(\mathbf{x}|\mathcal{D}_n) = a_\delta + b_\delta \text{trace}(\boldsymbol{\Sigma}_n(\mathbf{x}, \mathbf{x}))^{1/2}, \tag{23}$$

where $\boldsymbol{\Sigma}_n$ denotes the covariance matrix of the Gaussian process model at time n, whilst the variables a_δ, b_δ are simply some model-dependent constants which are independent of the input.

4.2.3 Theoretical guarantee

We will now extend the greedy performance guarantees described before in Section 4.1 for the Bayesian optimisation setting. Specifically, our main result in Proposition 4.2, gives a performance bound for the general family of approximate AEUI acquisition functions

$$\hat{\alpha}^{\text{AEUI}}(\mathbf{x}|\mathcal{D}_n) = \hat{\alpha}^{\text{EUI}}(\mathbf{x}|\mathcal{D}_n) + \mathcal{A}_\delta(\mathbf{x}|\mathcal{D}_n), \tag{24}$$

where $\hat{\alpha}^{\text{EUI}} : \mathbb{X} \to \mathbb{R}$ is the approximate EUI acquisition function in (22). The proof of this result is presented in Appendix A.8.2.

Proposition 4.2 (Approximate AEUI guarantee) *Consider an objective function $f : \mathbb{X} \to \mathbb{R}^M$ and an R2 utility $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}$ such that Assumptions 4.1 to 4.3 holds. Let $\{X_n\}_{n \geq 1}$ be the inputs selected using the corresponding approximate AEUI acquisition function, then for any $\delta \in (0, 1)$ and any positive integers P and N, the following inequality holds with probability $1 - \delta$:*

$$U(f(X_N)) \geq (1 - e^{-N/P}) \max_{X \subseteq \mathbb{X}, |X| \leq P} U(f(X)) - \sum_{i=1}^{2} \epsilon_i(\delta, P, N, J, H), \tag{25}$$

where the error is a sum of the Monte Carlo error and the model error,

$$\epsilon_1(\delta, P, N, J, H) = \sqrt{\frac{2}{\min(J, H)} \log\left(\frac{12N}{\delta}\right)} \sum_{n=1}^{N} C_{n-1}\left(1 - \frac{1}{P}\right)^{N-n},$$

$$\epsilon_2(\delta, P, N, J, H) = 2\sum_{n=1}^{N} \mathcal{A}_{\delta/(6N)}(\mathbf{x}_n|\mathcal{D}_{n-1})e^{-(N-n)/P},$$

respectively, where $C_n := \sup_{\boldsymbol{\theta} \in \Theta} \mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[S_{\boldsymbol{\theta}}(\hat{f}(\mathbb{X})) - S_{\boldsymbol{\theta}}(\hat{f}(X_n))]$ denotes the expected maximal scalarised regret for $n = 1, \ldots, N-1$ and $C_0 := C$.

Monte Carlo error. As reflected in the performance bound, we incur an additive error from estimating the EUI acquisition function using Monte Carlo. This error naturally decreases when we increase the number of samples. In particular, if we can compute the utility exactly, then this is the equivalent to setting $J \to \infty$. Similarly, if we can compute the expectation exactly, then this is the same as setting $H \to \infty$. Notably, if we can compute both the utility and the expectation exactly, then this error term becomes zero: $\epsilon_1(\delta, P, N, J, H) = 0$. In this case, the resulting performance bound would reflect the one obtained by employing the exact AEUI acquisition function (19).

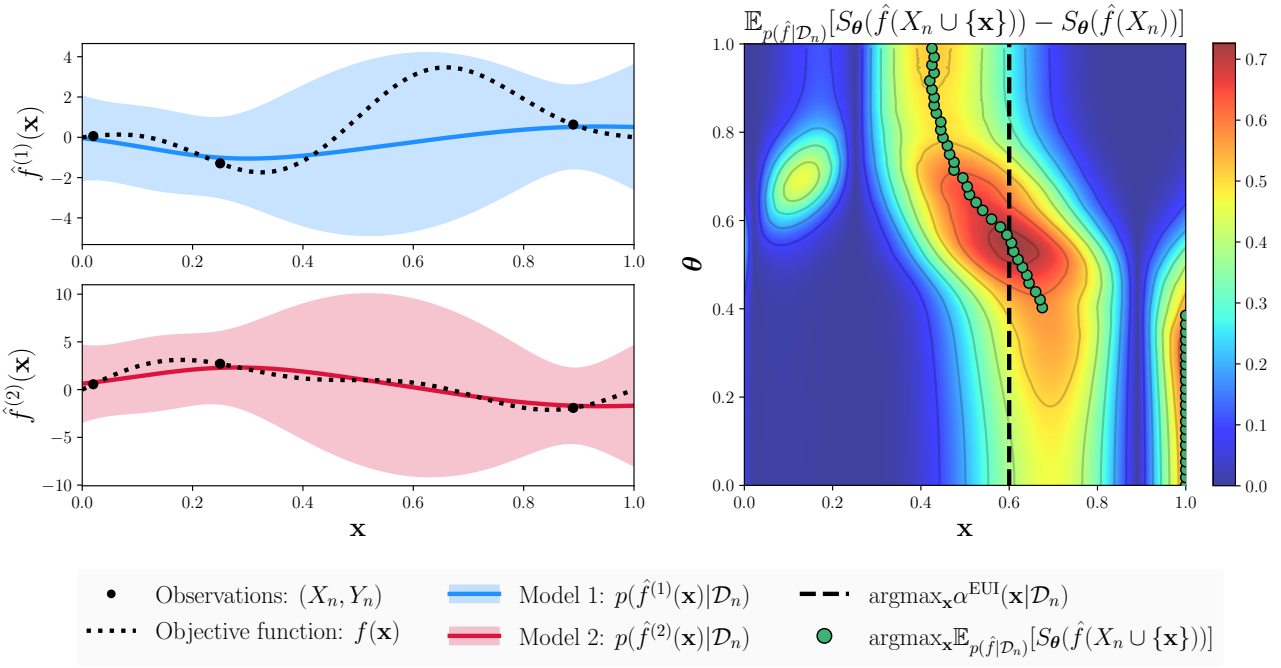

Figure 6: *A visualisation of the EUI acquisition function on a toy problem.*

Model error. The second error term in the performance bound reflects the error incurred in approximating the objective function using the surrogate model at every round. Note that the errors from the earlier rounds n are down-weighted exponentially. Therefore, if the model becomes more representative of the true objective function, as we acquire more data, then this error tends towards zero: $\epsilon_2(\delta, P, N, J, H) \to 0$ as $N \to \infty$. This result is expected because there should be no downside in using the surrogate model when it adequately describes the underlying objective function.

Example 4.2 (Toy problem) *In Figure 6, we illustrate the approximate EUI acquisition function (22) on a simple toy problem with two objectives defined on the unit interval* $\mathbb{X} = [0, 1]$. *We consider the hypervolume indicator (11) as our R2 utility and assume a Gaussian process prior on the function with a Gaussian observation likelihood. On the left, we present the mean and 99% credible interval of the Gaussian process posteriors. On the right, we present the contours of the expected scalarised improvement over both the input space and scalarisation parameter space. The black dotted line highlights the best solution according to the EUI acquisition function, whilst the green dots highlights the maximisers for the expected scalarised improvement for each scalarisation parameter.*

In Figure 7, we visualise the approximate EUI acquisition function over many different samples. We see that the maximiser of the approximation slowly converges to the maximiser of the exact acquisition function when the number of samples increases.

4.2.4 Special cases

The family of AEUI acquisition functions (19) contains many existing multi-objective acquisition functions as special cases. In what follows, we will briefly review these different variants. Loosely speaking, the main differences among these approaches come from the choice of R2 utility, the number of Monte Carlo samples and the surrogate model.

Expected improvement. The EUI acquisition function (20) can be interpreted as the multi-objective generalisation of the traditional expected improvement criterion [59, 46]. Many re-

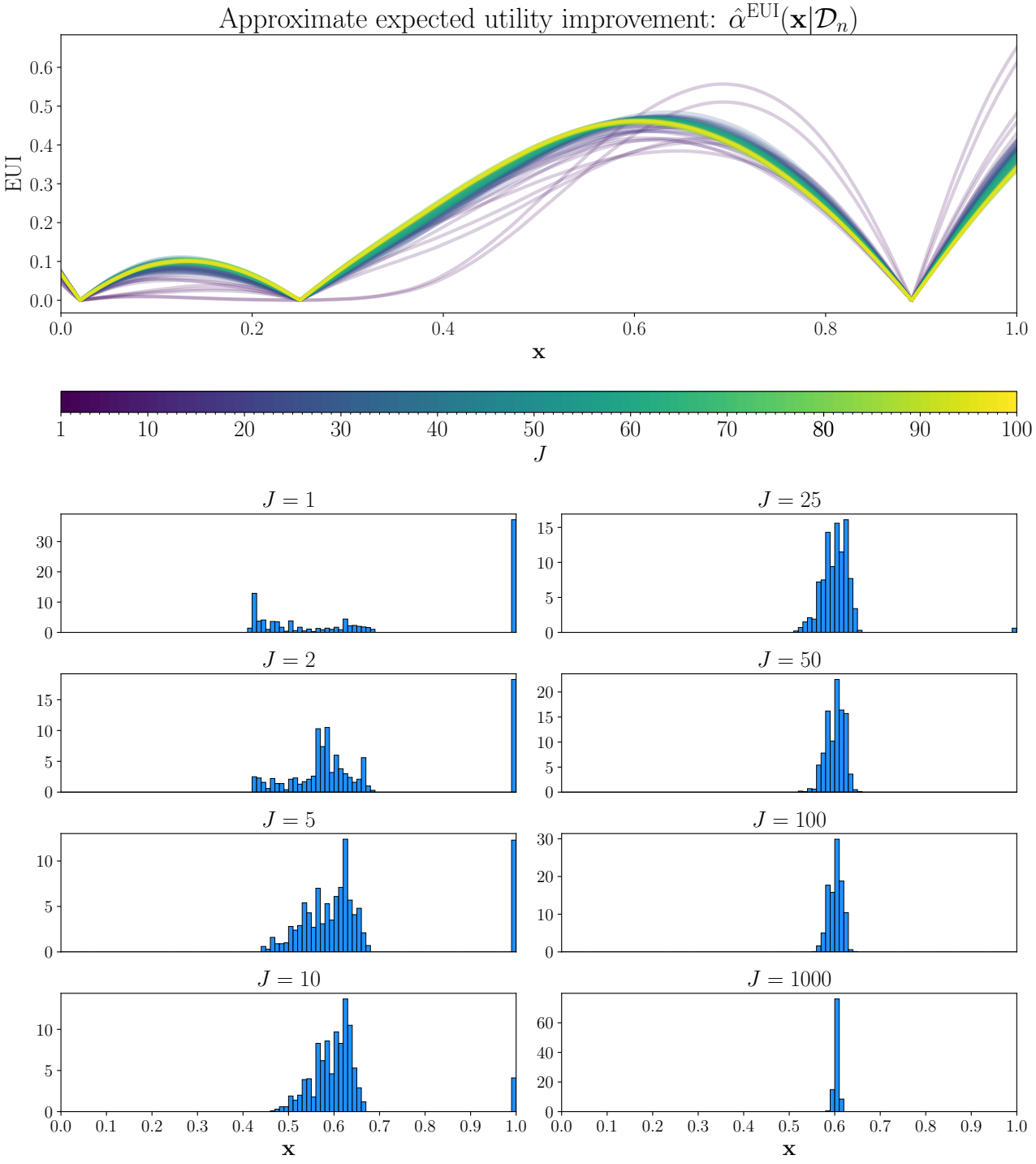

Figure 7: *A visualisation of the approximate EUI acquisition function with different numbers of Monte Carlo samples J on the example described in Figure 6. On the top, we present estimates of the EUI acquisition function obtained using different numbers of Monte Carlo samples—the initial random seed is the same for each estimate. On the bottom, we present the density plots of the maximisers obtained from 1000 independent runs.*

searchers have previously considered using this acquisition function before for some specific choices of utility functions [94, Section 4]. For example, researchers have considered using the standard R2 utility [26], the hypervolume indicator [32], and the IGD indicator [1]—all of which are special cases of the R2 utilities (Section 3.1). Following the example of Deutz, Emmerich, and Yang [26] and Astudillo and Frazier [1], we focussed our attention on the general setting, in which the utility is taken to be any arbitrary R2 utility. As illustrated in both of these papers, the EUI acquisition function[3] can be efficiently estimated using Monte Carlo samples. Complementary to these works, we proved a general performance bound for this strategy in Proposition 4.2, which is to the best of our knowledge, is a new result. This finding is however unsurprising but it gives some justification for why these methods have been so successful in practice.

Thompson sampling. The multi-objective generalisation of the Thompson sampling [83] strategy can be recovered by using only a single model sample, $H = 1$, in the approximate EUI acquisition function (22). This strategy has been proposed before for some specific utility functions. For instance, the TSEMO algorithm [9] considered the case where the hypervolume indicator is used as the utility.

Upper confidence bound. We obtain a multi-objective generalisation of the upper confidence bound (UCB) [81] strategy when we use a single function estimate, $\hat{f} : \mathbb{X} \to \mathbb{R}$, as our surrogate model. Ideally, this function estimate should be an upper bound for the true objective function. For instance, in standard Gaussian process regression, the function estimate could take the form $\hat{f}^{(m)}(\mathbf{x}) = \mu_n^{(m)}(\mathbf{x}) + \beta(\Sigma_n^{(m)}(\mathbf{x}, \mathbf{x}))^{1/2}$, where $\mu_n^{(m)} : \mathbb{X} \to \mathbb{R}$ is the mean function and $\Sigma_n^{(m)} : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$ is the covariance function at time n for objectives $m = 1, \ldots, M$, whilst $\beta \geq 0$ is a trade-off parameter that controls the width of the bound. One example of this variant being used in practice is the SMS-EGO algorithm [69], which considers the case where the hypervolume indicator is used as the utility function.

Random scalarisation. We recover the random scalarisation strategy when we use only a single sample, $J = 1$, to the estimate the utility in the approximate EUI acquisition function (22). One of the earliest examples of this approach being used is the popular ParEGO algorithm [48], which considered the specific setting where the utility was the augmented variant of the standard R2 utility (10). Since then, many other researchers have proposed different variations of this strategy. For example, Paria, Kandasamy, and Póczos [63] proposed the Thompson sampling and UCB variants of this strategy when the surrogate model is a Gaussian process. They showed that both of these approaches satisfy some Bayesian regret bounds when we additionally assume that the family of scalarisations functions are both monotonic and Lipschitz. Subsequently, Zhang and Golovin [96], specialised these results for the hypervolume indicator. Complementary to these works, Chowdhury and Gopalan [11] proved some frequentist regret bounds for the UCB variant of the random scalarisation strategy based on the approximate AEUI acquisition function (24).

Other acquisition functions. There exists several other types of acquisition functions for multi-objective Bayesian optimisations, such as information-theoretic acquisition functions [38, 5, 85], game-theoretic acquisition functions [68, 8], uncertainty-based acquisition functions [67, 104], and many others [49, 56]. These acquisition functions are not necessarily an instances of the AEUI acquisition function, but they have been shown to be an effective strategy to solve

[3] In these works, the EUI acquisition function with the R2 utility was referred to as the expected R2 improvement (ER2I) [26] and the expected improvement under utility uncertainty (EI-UU) [1], respectively.

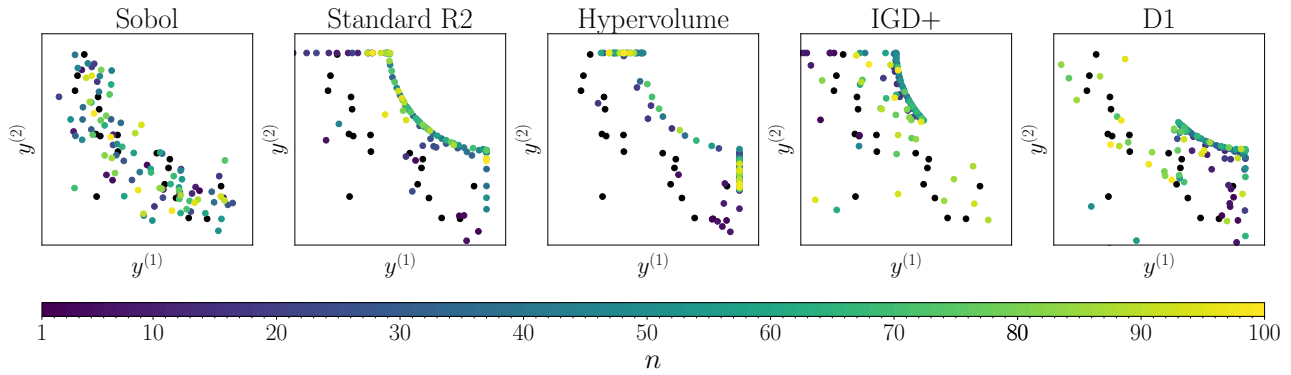

Figure 8: *A visualisation of the vectors obtained after one run of the approximate EUI approach with different choices of R2 utilities on the DTLZ2 benchmark with $D = 8$ and $M = 2$. We set $J = 256$ and $H = 128$ for the EUI estimate in (22).*

the R2 utility optimisation problem (15), in some cases, as illustrated in the aforementioned references. Having said that, in this paper we present a general approach in which the same utility function is used for both the optimisation procedure and the performance assessment because this leads to more interpretable results and clearer conclusions. Nevertheless, there are merits to using the other acquisition functions instead.

5 Numerical experiments

We will now present some case studies to assess the effectiveness of the Bayesian optimisation strategies described in Section 4.2 and the corresponding performance bound in Proposition 4.2. For convenience, we will consider a standard Bayesian optimisation set-up, where our surrogate model is a Gaussian process which is obtained by placing a Gaussian process prior on the objective function and a Gaussian likelihood on the observations. We have implemented all of the algorithms using the Python library BoTorch [4]. For all of the experiments, we initialised the data set by evaluating the function on $2(D+1)$ inputs which were obtained from uniformly sampling over the input space. For all of the performance plots, we have displayed the mean and two standard deviation of the log utility regret obtained from 100 independent runs. The code that was used to run the experiments is available in our Github repository: `https://github.com/benmltu/scalarize`.

5.1 Different utilities

The performance of the EUI acquisition function is heavily dependent on the choice of performance metric. To illustrate this, we consider optimising the bi-objective DTLZ2 benchmark [22] using the four different R2 utilities outlined in Section 3.1. Specifically, we use the standard R2 utility and hypervolume indicator in order to assess the whole Pareto front, but the IGD+ and D1 utilities are used to assess the upper and lower part of the Pareto front, respectively. We compare all of these greedy approaches with the random search baseline: Sobol.

In Figure 8, we present a visualisation of the objective vectors obtained after one run of Bayesian optimisation using the approximate EUI acquisition functions on the DTLZ2 benchmark. We see clearly that the choice of utility function influences the points that are selected. If we use a performance metric which assesses the whole Pareto front, then we end up querying points that try to cover the whole Pareto front. In contrast, if we use a performance metric which targets a particular region of the Pareto front, then we end up with more evaluations in this region.

In Figure 9, we plot the performance of these acquisition functions over the different performance metrics. On the whole, we see that the performance of the EUI acquisition function is either the best or close to it on the performance metric that it was designed on. This pattern continues even after we increase the dimensionality of the input space. Interestingly, there are some cases in which we achieve better performance on one utility by using a different utility in the acquisition function. For example, we noticed that the EUI-R2 approach performed slightly better than the EUI-HV in terms of the hypervolume. Visually, the EUI-R2 approach manages to identify many more central points than the EUI-HV approach. This result is likely a consequence of the geometry of the Pareto front and the choice of reference points. In particular, after normalising the objectives to lie in the unit hypercube, the standard R2 utility was defined as an average weighted distance to the reference point $\boldsymbol{\upsilon} = (1.1, 1.1)$, whereas the hypervolume indicator was defined as an average weighted distance away from the reference point $\boldsymbol{\eta} = (-0.1, -0.1)$. As the Pareto front in this problem is concave, this resulted in superior performance for the former approach because it placed more emphasis on the central points. In contrast, the latter approach placed more emphasis on the corners and therefore had less coverage.

For some additional intuition on the effect of changing the utility, we have included some additional examples in Appendix B which studies the case where the Pareto front is three-dimensional.

5.2 Different configurations

The performance bound in (25) indicates that the quality of the approximate EUI acquisition function is dependent on the number of Monte Carlo samples and the quality of the model. To test this out, we analysed the empirical performance of this method on four different benchmark problems, which have input dimensions $D \in \{4, 5, 6, 7\}$ and output dimensions $M \in \{3, 4\}$. We used the implementation of these problems provided in the following references [82, 17].

- **Rocket injector.** The rocket injector problem [87] considers optimising the performance and efficiency of a component used in a rocket. The objective function is defined over a $(D = 4)$-dimensional space and consists of $M = 3$ objectives.
- **Vehicle safety.** The vehicle design problem [93] considers optimising some variables relating to the safety of a vehicle. This objective function is defined over a $(D = 5)$-dimensional space and consists of $M = 3$ objectives.
- **Bulk carrier.** The bulk carrier design problem [66] considers optimising some variables relating to the design specifications of a bulk carrier. This objective function is defined over a $(D = 6)$-dimensional space and consists of $M = 4$ objectives.
- **Car side impact.** The car side-impact problem [20] considers optimising the performance of a car subject to some safety regulations. This objective function is defined over a $(D = 7)$-dimensional space and consists of $M = 4$ objectives.

In Figure 10, we plot the performance of the approximate EUI acquisition function when we vary the number of scalarisation parameter samples. The label "$J = \infty$", denotes the setting where we compute the hypervolume indicator exactly [17]. On the whole, we see that the performance does indeed improve when we increase the number of samples. This improvement is, however, quite small, which indicates that the variance of the Monte Carlo error is rather minor for these problems.

In Figure 11, we plot the performance of the approximate EUI acquisition function when we vary the number of model samples. We see that the difference in performance is very minor in

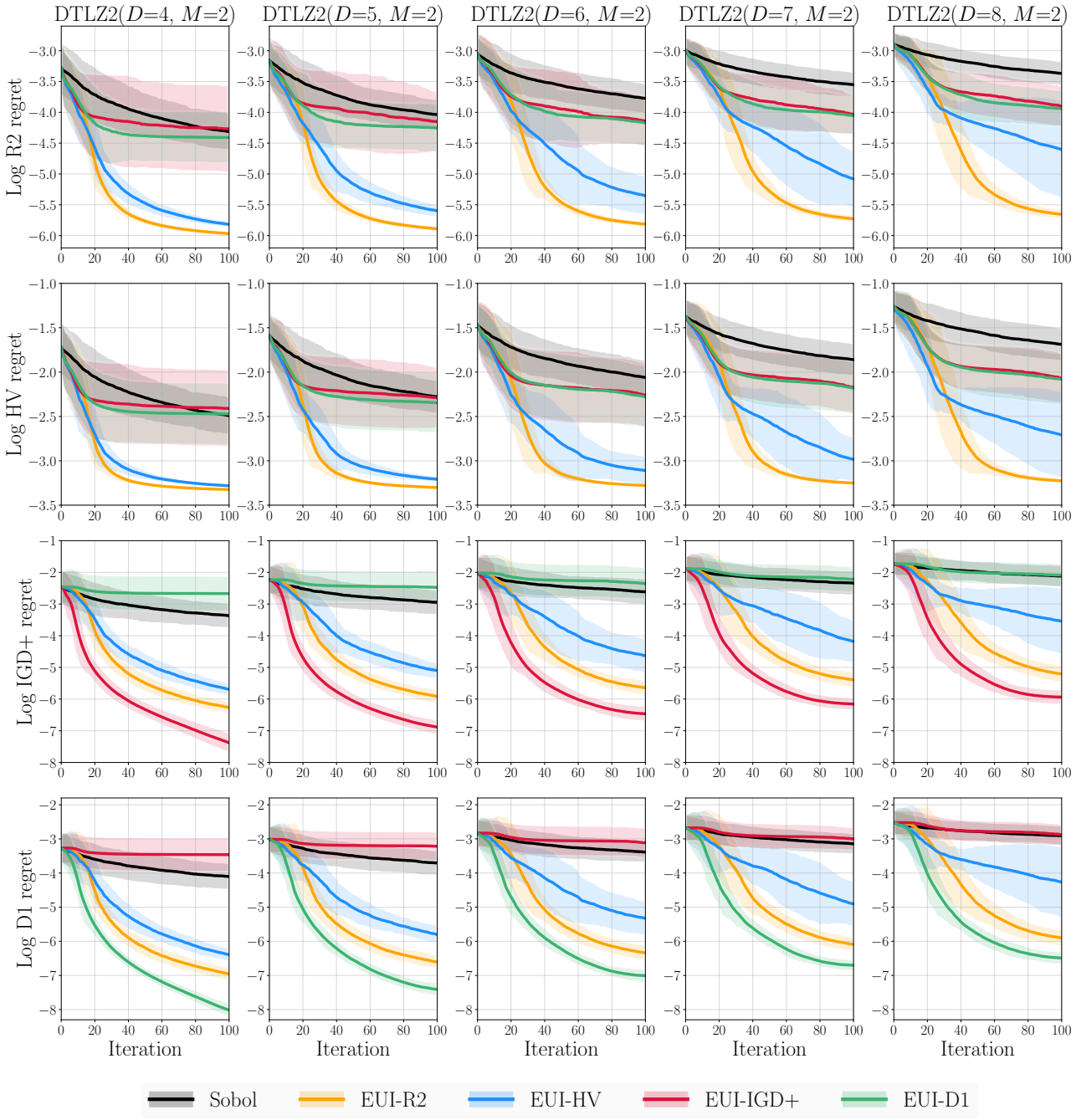

Figure 9: *A performance comparison of the approximate EUI approach with different choices of R2 utilities when applied on the DTLZ2 problem. In each row, we consider a different R2 utility, whereas in each column we consider a different number of inputs. For the EUI estimate* (22)*, we set* $J = 256$ *and* $H = 128$.

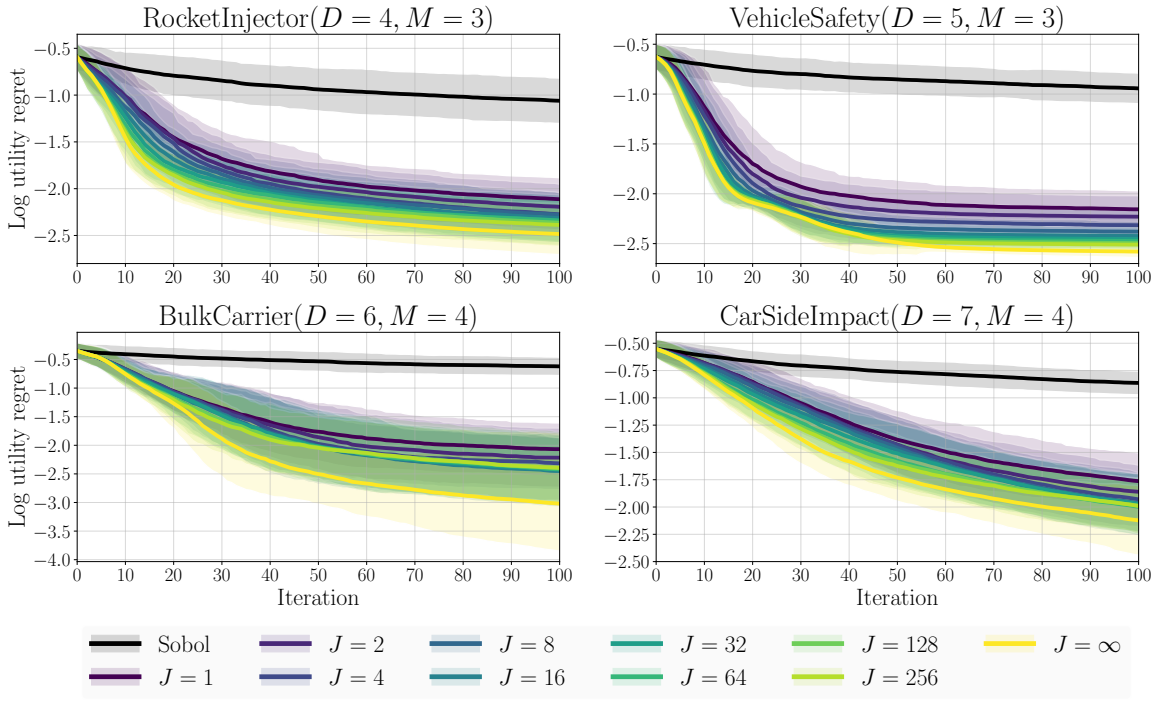

Figure 10: *A performance comparison of the approximate EUI approach when varying the number of samples used to estimate the utility. The utility function is set to the hypervolume indicator, whilst the number of model samples is set to* $H = 128$.

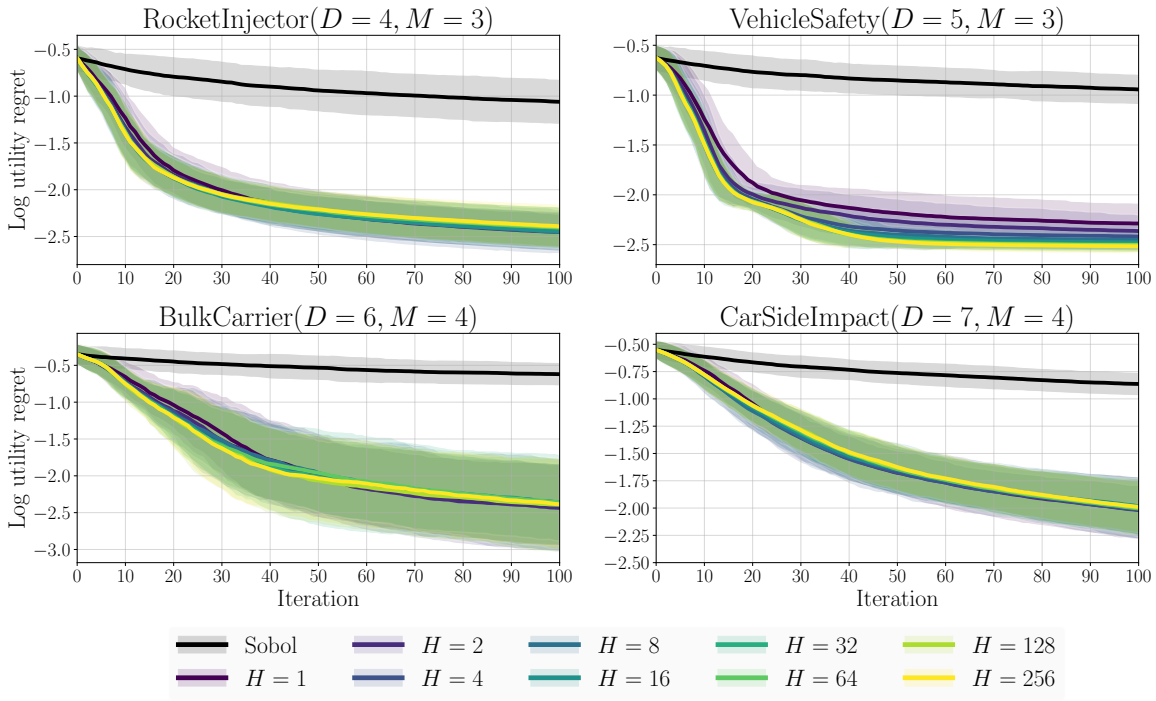

Figure 11: *A performance comparison of the approximate EUI approach when varying number of samples used to estimate the expectation over the model. The utility function is set to the hypervolume indicator, whilst the number of scalarisation parameter samples is set to* $J = 256$.

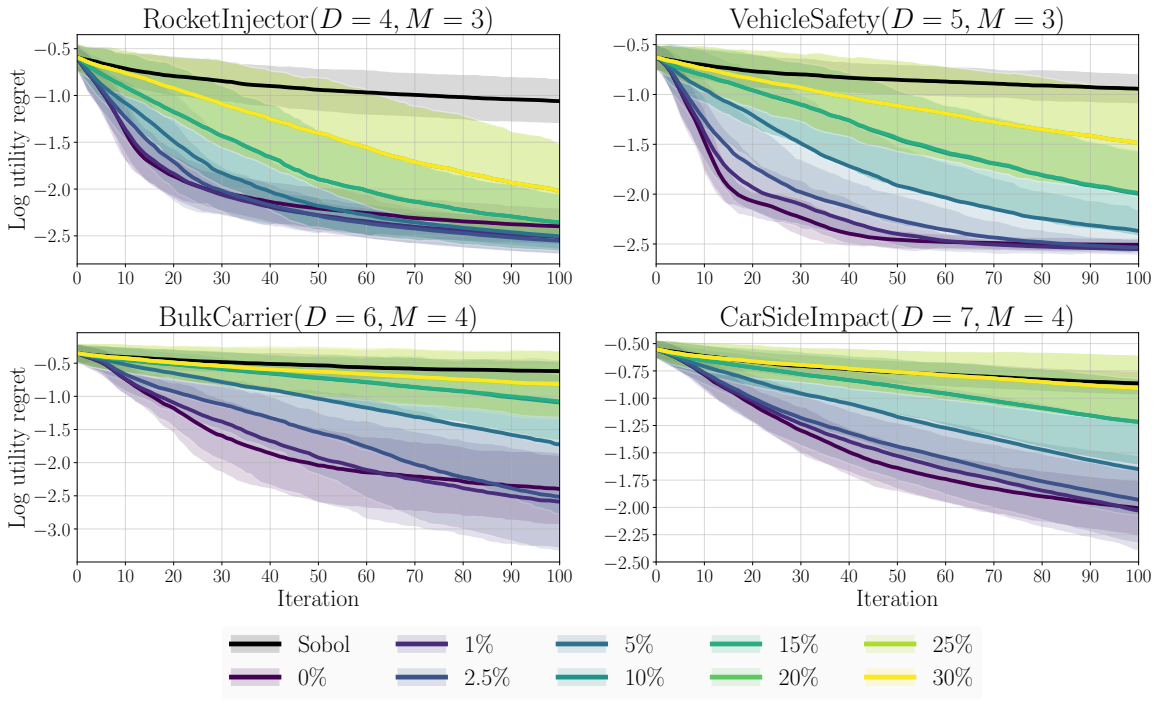

Figure 12: *A performance comparison of the approximate EUI approach when varying the standard deviation of the additive Gaussian noise. The standard deviation is set at a percentage of the objective ranges. The utility function is set to hypervolume indicator, whilst the number of Monte Carlo samples is set to* $J = 256$ *and* $H = 128$.

most problems. There does not appear to be much computational benefit from increasing the number of model samples. In many cases, the Thompson sampling approach, where we just use one sample, performs just as well as the more expensive approaches.

In Figure 12, we plot the performance of the approximate EUI acquisition function when we vary the standard deviation of the additive Gaussian noise which we add to the function evaluations. As expected, the performance degrades when we incorporate more noise into the data. Surprisingly, there are some instances in which adding a small bit of noise actually improves the performance very slightly over the zero noise setting. This is perhaps a consequence of some additional exploration that takes place when the observations are a bit noisy.

5.3 Mixed approaches

The EUI acquisition function can work well when the surrogate model is a good representation of the underlying objective. On other hand, there are some cases in which the EUI acquisition function can be overly greedy in its evaluations and can end up getting stuck in a suboptimal region. To illustrate this, consider the Gaussian mixture model (GMM) problem [18], where each objective is a mixture of Gaussian probability densities. This problem is tricky because the objective function has multiple modes which could potentially mislead the EUI acquisition function. We can see an instance of this happening in the second column of Figure 13; specifically, we see that the EUI strategy manages to quickly locate one of the suboptimal modes and then only queries points near this region for the remaining iterations.

One strategy to overcome the over-exploitative behaviour of the EUI approach is to include a non-constant adjustment term into the acquisition function in order to encourage additional exploration. In the third column of Figure 13, we illustrate the performance of this AEUI strategy when we use the adjustment term in (23) with $b_\delta = 1/M$. Unlike the EUI approach, we do not get stuck in one of the modes. On the contrary, we end up mimicking a space-filling

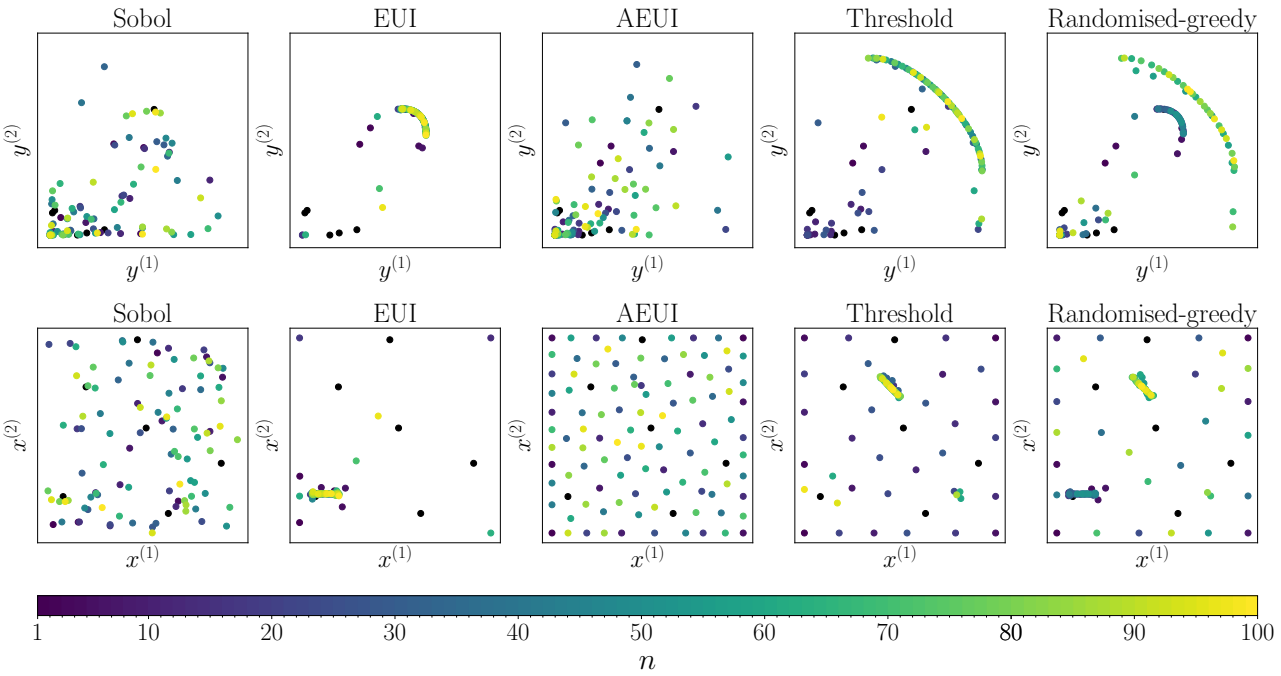

Figure 13: *A visualisation of the points obtained after one run of the different algorithms applied on the bi-objective GMM problem with $D = 2$. For the mixed approaches, we set $p = 0.3$.*

design, where most of the function evaluations are spent in exploration in order to improve the model; only a few evaluations are actually spent in exploiting the model. In order to strike a better balance between exploration and exploitation, we propose using a novel mixed approach where we alternate between an explorative evaluation and an exploitative one. More precisely, we consider the mixed acquisition function

$$\alpha^{\text{EUI}-\lambda_n}(\mathbf{x}|\mathcal{D}_n) = \alpha^{\text{EUI}}(\mathbf{x}|\mathcal{D}_n) + \lambda_n \mathcal{A}_\delta(\mathbf{x}|\mathcal{D}_n),$$

where $\lambda_n \geq 0$ is a time-dependent variable. For example, we could use the *threshold approach*, where we set the schedule deterministically: $\lambda_n = \mathbb{1}[n \leq pN]$ for some $p \in [0, 1]$. Alternatively, we could use a *randomised-greedy*[4] strategy, where we set the schedule randomly by sampling from a Bernoulli distribution: $\lambda_n \sim \text{Bernoulli}(p)$ for some $p \in [0, 1]$. As we can see in the final two columns of Figure 13, the mixed approach manages to find the best of both worlds by spending some evaluations in improving the model and the other evaluations in exploiting what has been learnt.

Remark 5.1 (Mixed performance guarantee) *Note that the performance bound for the mixed approach is justa convex combination of the original bounds in (25). In particular, the Monte Carlo error remains the same, whereas the model error is a convex combination of the EUI model error and the AEUI model error; that is, $\epsilon_2 = (1 - p)\epsilon_2^{\text{EUI}} + p\epsilon_2^{\text{AEUI}}$.*

In Figure 14, we present the performance plots of the mixed approaches on the GMM problem when we vary the number of objectives. We plot the logarithm of the hypervolume regret over many possible choices of $p \in [0, 1]$. Overall, we see that the performance of the mixed approach improves with the parameter p up to a specific point before it begins to deteriorate again. This result perhaps gives some indication that there is an 'optimal' choice of p, which likely depends on the utility, the surrogate model and the unknown objective function. In practice,

[4]The randomised-greedy approach bares some similarity to the ϵ-greedy approach which was previously proposed for the single-objective expected improvement [19]. In the ϵ-greedy approach, we sample uniformly from the input space with probability $\epsilon \in [0, 1]$ as opposed to optimising a different acquisition function.

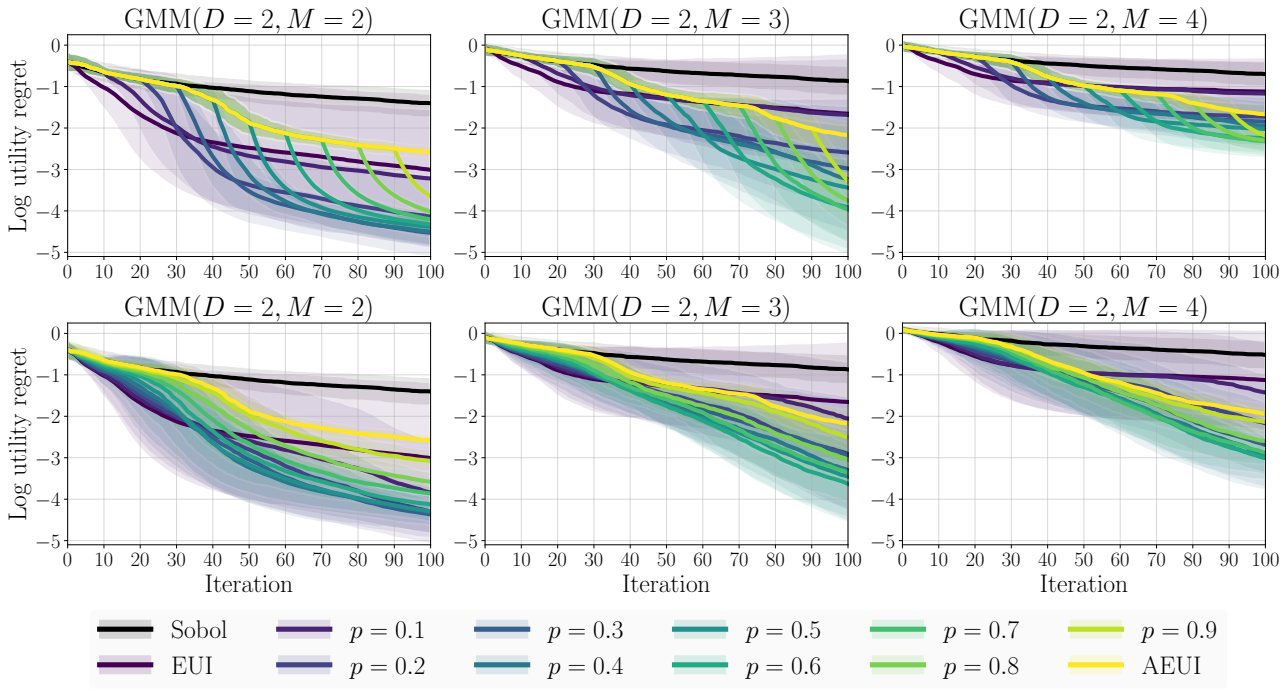

Figure 14: *A performance comparison of the threshold approach (top row) and randomised-greedy approach (bottom row) for different values of $p \in [0, 1]$ on the GMM problem with different number of objectives. The utility function was set to the hypervolume indicator.*

we believe that this parameter should be set by the decision maker based on their confidence in the surrogate model. Specifically, a practitioner can use this parameter to modulate the number of evaluations which should be spent in exploration in order to improve the model and the number that should be spent in exploitation in order to improve the utility.

6 Summary and future work

In this paper, we presented a unified view of multi-objective optimisation. We described the two main philosophies that are used to solve this problem and then showed how they can be unified via the R2 utility functions. We then discussed the practical problem of actually solving this set optimisation problem (15) once we have identified a suitable R2 utility. We focussed our discussion on the use of greedy algorithms because they come with theoretical performance guarantees. Among other things, we analysed the effectiveness of these greedy strategies in the popular Bayesian optimisation setting, where the objective function is modelled using a surrogate model. Notably, the analysis we have provided supports many of the Bayesian optimisation strategies that have been proposed over the past decade. To support these theoretical results, we also conducted several numerical experiments which demonstrate the effectiveness of our general methodology in practice. During our exposition, we also showcased some new variations of the greedy algorithms which arise naturally from our general framework. Overall, this paper has tried to consolidate many different ideas in multi-objective optimisation into one common framework. In what follows we discuss how some of these ideas could be used as a basis for future work.

6.1 Preference elicitation

Practitioners often resort to the use of off-the-shelf performance metrics in order to assess the quality of a Pareto front approximation. As described in Section 3.2, this is counter-productive

in the setting where a decision maker has preferences which are not adequately encoded into these performance metrics. The R2 utilities offers a potential solution to this problem. By carefully choosing the family of scalarisation function and parameter distributions, we can describe very complex preferences in the objective space. A useful direction for future work would focus on giving some practical guidance for setting the R2 utility in many commonly occurring situations. Alternatively, one might consider designing new adaptive strategies which try to learn the utility directly from user-interactions—this line of inquiry is related to the topic of preference learning. For example, the papers by Zintgraf et al. [100] and Lin et al. [53] considered the special case where the decision maker's preferences are described completely by a single scalarisation function. They then proceeded to learn this scalarisation function using different types of preference information. Notably, both of these papers were largely inspired by the Bayesian preference learning strategy of Chu and Ghahramani [12]. In this approach, a Gaussian process prior is placed on the scalarisation function and a likelihood is placed on the preference data. The scalarisation function is then updated using a Bayesian posterior update whenever more preference information is presented. For instance, Lin et al. [53] studied the setting in which the preference data came in the form of pairwise comparisons which were modelled using a probit likelihood, whereas Zintgraf et al. [100] also considered other preferences as well such as ranking and clustering preferences, which were modelled accordingly.

6.2 Constrained optimisation

Real-world problems typically contain constraints which must also be taken into account. In this paper, we have largely overlooked this element of the problem and have implicitly absorbed any notion of a constraint into the definition of the input space $\mathbb{X} \subseteq \mathbb{R}^D$. Naturally, many of the general ideas and results that we have discussed here remains the same when we include constraints into the different scalar-valued optimisation problems such as in (2) and (7). Nevertheless, there are also many practical considerations that have to be considered when we actually want to solve these constrained problems in practice. For instance, if we wanted to employ one of the greedy strategies described in Section 4, then we would need to be able to optimise over this constrained space. This could perhaps be accomplished using a penalty-based optimisation strategy, although there are clearly many other possibilities as well [62]. These extensions are in general non-trivial and requires some additional thought. For example, with a penalisation strategy one has to decide what penalisation to use and where to incorporate it into the problem. Should we include it at the level of the objective function or at the level of the scalarisation or utility function? All of these choices lead to different algorithms and issues which would be interesting to study in their own right.

6.3 Robustness

In real-world problems, the objective function is subject to many sources of uncertainty. A natural framework in which to study these types of problems is robust optimisation [6]. Some effort has been spent over the last few decades in developing these robustness concepts for the multi-objective setting. Many of these extensions rely on the use of scalarisation functions [29, 42, 35, 18]. An interesting concept for future work would be to somehow unify many of these ideas into a common framework such as the one we presented in Section 3 for the standard setting.

6.4 Partially ordered optimisation

The work in this paper has focussed on the multi-objective optimisation problem defined using the Pareto partial ordering. Nonetheless, many of the results that we presented here can also be extended to a more general setting in which the objective function $f : \mathbb{X} \to \mathbb{S}$, takes values in any partially ordered space $\mathbb{S}$. Specifically, we can follow the same recipe and reformulate the optimisation problem by appealing to the use of scalarisation functions $s : \mathbb{S} \to \mathbb{R}$, or utility functions $U : \mathbb{B}(\mathbb{S}) \to \mathbb{R}$. The R2 utilities can also be formulated and optimised in the same way as before. An interesting direction for future work could be focussed on investigating specific instances of this generalisation. For example, some researchers have previously attempted to incorporate user preferences (Section 6.1) by directly defining some preference-based partial orderings [27, 65]. Similarly, we can also treat the constrained (Section 6.2) and robust (Section 6.3) optimisation problem as instances of this generalisation, where we appeal to the constrained [36, 21] and robust [42, 43] Pareto domination rules instead.

6.5 Decoupled evaluations

We assumed throughout this paper that we always make full evaluations of the objective function. In some settings, we can choose to evaluate the objective function on only a subset of the objectives [39]. This might be advantageous when certain objectives are more expensive to evaluate than others. On the whole, it is not immediately clear how we can extend the formulation of the R2 utilities in order to handle this *decoupled* set-up. The main difficulty lies in the fact that we have to assess the quality of a collection of partially observed objectives, which are not immediately comparable under the Pareto partial ordering. This decoupled framework is also related to optimisation problems for missing data [7]. An interesting line of inquiry would be to focus on formalising this problem more concretely and then developing suitable algorithms to solve it.

6.6 Multi-objective decision making

Many of the ideas that we described here can also be exploited in other areas of research concerned with multi-objective decision making problems. Following Trivedi et al. [84], we can broadly categorise multi-objective algorithms into three categories.

1. **Domination-based.** These algorithms adopt the standard multi-objective optimisation perspective, where we solve the problem using only the sorting routines based on the Pareto partial ordering.

2. **Decomposition-based.** These algorithms adopt the scalarisation perspective of multi-objective optimisation (Section 2.1), which considers jointly solving a collection of single-objective problems using scalarisation functions.

3. **Indicator-based.** These algorithms adopt the utility perspective of multi-objective optimisation (Section 2.2), which rely on the use of utility functions in order to determine which sets of vectors are the most desirable.

In Section 3, we demonstrated how all three categories are linked via the R2 utilities (8). A consequence of this connection is that many indicator-based algorithms can potentially be converted into a decomposition-based algorithm via a suitably chosen distribution of scalarisation functions and vice versa. For example, we can exploit this connection in the development of new multi-objective evolutionary algorithms [97, 52, 98], multi-objective reinforcement learning algorithms [72, 89, 88, 99], and multi-objective gradient-based optimisation algorithms (Section 6.7).

6.7 Gradient-based optimisation

The traditional goal of multi-objective gradient-based optimisation is to target a finite collection of Pareto stationary points [51]. This is typically accomplished by appealing to the scalarisation methodology in which scalarised optimisation problems (2) are defined and consequently solved using standard gradient-based methods. There also exist many adaptive variants of these algorithms in which the scalarisation parameters do not have to be specified a priori, but can be updated in an online manner [34, 75, 33, 30, 25, 77, 55]. Notably much less work has been focussed on solving the utility optimisation problem (7) with gradient-based methods [31, 80, 91, 23]. The key challenge with this latter problem is that the (sub-)gradients[5] of a Pareto compliant utility function are zero at any input which is Pareto dominated. In regards to an R2 utility (8), this can happen when an input does not lead to a maximal scalarised value for any scalarisation parameter. This issue is clearly problematic in a gradient-based algorithm because whenever an input becomes sub-optimal, with respect to the current set, then it is no longer updated. Many authors have suggested including some additional dynamics to tackle this issue [91, 23]. However, these methods have not been studied theoretically and their relationship to existing scalarisation-based algorithms is unclear. In particular, we anticipate that there are many useful ideas from the existing scalarisation-based algorithms that could be exploited in order to solve the utility optimisation problem in a more efficient and theoretically justifiable way.

Acknowledgements

Ben Tu was supported by the EPSRC StatML CDT programme EP/S023151/1 and BASF SE, Ludwigshafen am Rhein. Nikolas Kantas was partially funded by JPMorgan Chase & Co. under J.P. Morgan A.I. Faculty Research Awards 2021. We would also like to acknowledge the associate editor and reviewers for their useful suggestions and insights that have helped to improve this paper.

[5] The gradient of a utility function is not necessarily well-defined in practice. For example, in an R2 utility (8) we would need to differentiate a maximum of some scalarised functions. To address this problem it is common to appeal to a more relaxed notion of a gradient [64].

A Proof of results

A.1 Proof of Proposition 2.1

These two results follow from a simple proof by contradiction. Let $s : \mathbb{R}^M \to \mathbb{R}$ denote a scalarisation function which is strictly monotonically increasing over the feasible objective space. Suppose that there exists a solution $\mathbf{x}^* \in X_s^*$ which is not strictly Pareto optimal. Then there must exist an input $\mathbf{x} \in \mathbb{X}$ such that $f(\mathbf{x}) \succ f(\mathbf{x}^*)$. By strict monotonicity, we have $s(f(\mathbf{x})) > s(f(\mathbf{x}^*))$, which contradicts the optimality of $\mathbf{x}^*$.

Similarly, for the other result, let $s : \mathbb{R}^M \to \mathbb{R}$ denote a scalarisation function which is strongly monotonically increasing over the feasible objective space. Suppose that there exists a solution $\mathbf{x}^* \in X_s^*$ which is not weakly Pareto optimal. Then there must exist an input $\mathbf{x} \in \mathbb{X}$ such that $f(\mathbf{x}) \succ\!\!\succ f(\mathbf{x}^*)$. By strong monotonicity, we have $s(f(\mathbf{x})) > s(f(\mathbf{x}^*))$, which contradicts the optimality of $\mathbf{x}^*$.

■

A.2 Proof of Proposition 3.1

In the following paragraph, we will prove that the set function $S_{\boldsymbol{\theta}}$ satisfies the monotone and diminishing returns property for all scalarisation parameters $\boldsymbol{\theta} \in \Theta$. Once we have this result, we can conclude that any R2 utility, which is defined using these set functions, also satisfies these properties because inequalities are preserved when we compute the expectation with respect to a non-negative density $p(\boldsymbol{\theta}) \geq 0$.

Consider any finite set of vectors $A, B \in \mathbb{B}(\mathbb{R}^M)$ with $A \subseteq B$ and any vector $\mathbf{c} \in \mathbb{R}^M$. Then clearly the set function $S_{\boldsymbol{\theta}}$ is monotonic because $S_{\boldsymbol{\theta}}(A) \leq S_{\boldsymbol{\theta}}(B)$ for all $\boldsymbol{\theta} \in \Theta$. We will now show that $S_{\boldsymbol{\theta}}$ also satisfies the diminishing returns property. If the maximum values are the same, $S_{\boldsymbol{\theta}}(A) = S_{\boldsymbol{\theta}}(B)$, then the diminishing returns property is trivially satisfied. If the maximum values are different, $S_{\boldsymbol{\theta}}(A) < S_{\boldsymbol{\theta}}(B)$, then there are two settings to consider. In the first case, we have $S_{\boldsymbol{\theta}}(B) \geq S_{\boldsymbol{\theta}}(\{\mathbf{c}\})$, which implies $S_{\boldsymbol{\theta}}(A \cup \{\mathbf{c}\}) - S_{\boldsymbol{\theta}}(A) \geq 0 = S_{\boldsymbol{\theta}}(B \cup \{\mathbf{c}\}) - S_{\boldsymbol{\theta}}(B)$. Whilst in the second case, we have $S_{\boldsymbol{\theta}}(B) < S_{\boldsymbol{\theta}}(\{\mathbf{c}\})$, which implies

$$\begin{aligned} S_{\boldsymbol{\theta}}(A \cup \{\mathbf{c}\}) - S_{\boldsymbol{\theta}}(A) &= S_{\boldsymbol{\theta}}(\{\mathbf{c}\}) - S_{\boldsymbol{\theta}}(A) \\ &\geq S_{\boldsymbol{\theta}}(\{\mathbf{c}\}) - S_{\boldsymbol{\theta}}(B) = S_{\boldsymbol{\theta}}(B \cup \{\mathbf{c}\}) - S_{\boldsymbol{\theta}}(B). \end{aligned}$$

■

A.3 Proof of Proposition 3.2

Consider any finite set of vectors $A, B \in \mathbb{B}(\mathbb{R}^M)$ with $A \succeq B$. As the scalarisation functions $s_{\boldsymbol{\theta}}$ are monotonically increasing for all scalarisation parameters, this implies $S_{\boldsymbol{\theta}}(A) \geq S_{\boldsymbol{\theta}}(B)$ for all $\boldsymbol{\theta} \in \Theta$. By computing the expectation with respect to the distribution $p(\boldsymbol{\theta}) \geq 0$, we obtain the final result $U(A) \geq U(B)$.

■

A.4 Proof of Proposition 3.3

The proof of this result is presented in the following references: [78, Section 3.2], [24, Section 2] and [96, Lemma 5].

■

A.5 Proof of Proposition 3.4

This result can be derived by simply expanding the definition of the IGD indicator (13):

$$\begin{aligned}(I^{\mathrm{IGD}_{p,q}}(Y,\Upsilon))^q &= \frac{1}{|\Upsilon|}\sum_{\boldsymbol{\upsilon}\in\Upsilon}\Big(\min_{\mathbf{y}\in Y}||\boldsymbol{\upsilon}-\mathbf{y}||_{L^p}\Big)^q \\ &= \frac{1}{|\Upsilon|}\sum_{\boldsymbol{\upsilon}\in\Upsilon}\min_{\mathbf{y}\in Y}||\boldsymbol{\upsilon}-\mathbf{y}||_{L^p}^q \\ &= -\frac{1}{|\Upsilon|}\sum_{\boldsymbol{\upsilon}\in\Upsilon}\max_{\mathbf{y}\in Y}(-||\boldsymbol{\upsilon}-\mathbf{y}||_{L^p}^q) \\ &= -\frac{1}{|\Upsilon|}\sum_{\boldsymbol{\upsilon}\in\Upsilon}\max_{\mathbf{y}\in Y}s_{\boldsymbol{\upsilon}}^{\mathrm{IGD}_{p,q}}(\mathbf{y})\end{aligned}$$

for any $Y,\Upsilon\in\mathbb{B}(\mathbb{R}^M)$ and $p,q\geq 1$.

■

A.6 Proof of Proposition 3.5

This result can be derived by simply expanding the definition of the D1 indicator (14):

$$\begin{aligned}I^{\mathrm{D1}}(Y,\Upsilon,\mathbf{w}) &= \frac{1}{|\Upsilon|}\sum_{\boldsymbol{\upsilon}\in\Upsilon}\min_{\mathbf{y}\in Y}\max_{m=1,\dots,M}w^{(m)}(\upsilon^{(m)}-y^{(m)}) \\ &= -\frac{1}{|\Upsilon|}\sum_{\boldsymbol{\upsilon}\in\Upsilon}\max_{\mathbf{y}\in Y}\Big(-\max_{m=1,\dots,M}w^{(m)}(\upsilon^{(m)}-y^{(m)})\Big) \\ &= -\frac{1}{|\Upsilon|}\sum_{\boldsymbol{\upsilon}\in\Upsilon}\max_{\mathbf{y}\in Y}s_{(\boldsymbol{\upsilon},\mathbf{w})}^{\mathrm{Chb}}(\mathbf{y})\end{aligned}$$

for any $Y,\Upsilon\in\mathbb{B}(\mathbb{R}^M)$ and $\mathbf{w}\in\Delta^{M-1}$.

■

A.7 Proof of Theorem 4.1

The proof of this result is identical to the one described by Krause and Golovin [50, Theorem 1.5].

■

A.8 Performance bounds

The proof of our performance bounds follows a similar pattern to the proof of the original greedy guarantee described by Krause and Golovin [50, Theorem 1.5]. To establish these probabilistic bounds, we will first prove a useful result (Lemma A.1). For convenience of notation, in what follows we denote an optimal set of inputs by

$$X_P^*\in\operatorname*{arg\,max}_{X\subseteq\mathbb{X},|X|\leq P}U(f(X))$$

for any $P>0$.

Lemma A.1 *Consider an objective function $f : \mathbb{X} \to \mathbb{R}^M$, a non-negative R2 utility $U : \mathbb{B}(\mathbb{R}^M) \to \mathbb{R}_{\geq 0}$, and a collection of inputs and scalars $\{X_n\}_{n\geq 1}$ and $\{\rho_n\}_{n\geq 1}$, where $X_n \subseteq \mathbb{X}$ and $\rho_n \in \mathbb{R}$. Suppose for some positive integer P, that there exist a $\delta \in (0, 1)$ such that the following inequality holds with probability $1 - \delta$ for $n \leq N$:*

$$U(f(X_P^*)) \leq U(f(X_n)) + P(U(f(X_{n+1})) - U(f(X_n))) + P\rho_{n+1}.$$

Then the following inequality holds with probability $1 - N\delta$,

$$U(f(X_N)) \geq (1 - e^{-N/P})U(f(X_P^*)) - \sum_{n=1}^{N} \rho_n \Big(1 - \frac{1}{P}\Big)^{N-n}.$$

Proof of Lemma A.1. Let $\zeta_n = U(f(X_P^*)) - U(f(X_n))$; then rearranging the inequality, we obtain

$$\zeta_{n+1} \leq \Big(1 - \frac{1}{P}\Big)\zeta_n + \rho_{n+1}.$$

By repeated use of this inequality, we find that

$$\zeta_N \leq \Big(1 - \frac{1}{P}\Big)^N \zeta_0 + \sum_{n=1}^{N} \rho_n \Big(1 - \frac{1}{P}\Big)^{N-n},$$

which is an inequality that holds with probability $1 - N\delta$ by the union bound (Boole's inequality). As the R2 utility is non-negative, we have that $\zeta_0 = U(f(X_P^*)) - U(f(\emptyset)) \leq U(f(X_P^*))$. Using this result and the fact that $1 - x \leq e^{-x}$ for all $x \in \mathbb{R}$, we obtain the final performance bound

$$U(f(X_N)) \geq (1 - e^{-N/P})U(f(X_P^*)) - \sum_{n=1}^{N} \rho_n \Big(1 - \frac{1}{P}\Big)^{N-n}.$$

■

A.8.1 Proof of Proposition 4.1

We begin the proof by first establishing an inequality between the series of utility values:

$$\begin{aligned}
U(f(X_P^*)) &\leq U(f(X_P^* \cup X_n)) && (26)\\
&= U(f(X_n)) + \sum_{p=1}^{P} (U(f(X_p^* \cup X_n)) - U(f(X_{p-1}^* \cup X_n))) && (27)\\
&\leq U(f(X_n)) + \sum_{\mathbf{x}\in X_P^*} (U(f(X_n \cup \{\mathbf{x}\})) - U(f(X_n))) && (28)\\
&\leq U(f(X_n)) + P(U(f(X_n \cup \{\mathbf{x}^*\})) - U(f(X_n))). && (29)
\end{aligned}$$

The first line (26) follows from the monotone property. The second line (27) by from rewriting the utility as a telescopic sum. The third line (28) by from applying the diminishing returns property and the last line (29) by from setting the input $\mathbf{x} \in \mathbb{X}$ to be a maximiser, $\mathbf{x}^* \in \arg\max_{\mathbf{x}\in X_P^*}(U(f(X_n \cup \{\mathbf{x}\})) - U(f(X_n)))$, and then using the fact that $|X_P^*| \leq P$, where $|\cdot|$ denotes the cardinality of the set.

As the scalarisation parameters are sampled independently, we have the following result by Hoeffding's inequality:

$$\mathbb{P}\Big[\big|(\hat{U}_J(f(X_n \cup \{\mathbf{x}\})) - \hat{U}_J(f(X_n))) - (U(f(X_n \cup \{\mathbf{x}\})) - U(f(X_n)))\big| \geq \xi_n\Big] \leq 2\exp\left(-\frac{2J\xi_n^2}{C_n^2}\right)$$

for any $\mathbf{x} \in \mathbb{X}$, where $C_n = \sup_{\boldsymbol{\theta}\in\Theta}(S_{\boldsymbol{\theta}}(f(\mathbb{X})) - S_{\boldsymbol{\theta}}(f(X_n)))$ for $n \geq 1$ and $C_0 := C$. By manipulating the expression, we find that this inequality will hold with probability $1-\delta$ if we set $\xi_n = C_n\sqrt{\log(2/\delta)/(2J)}$. Using this result, we obtain the following probability bound, which holds with probability $1-2\delta$ by the union bound:

$$\begin{aligned} U(f(X_P^*)) &\leq U(f(X_n)) + P(\hat{U}_J(f(X_n \cup \{\mathbf{x}^*\})) - \hat{U}_J(f(X_n)) + \xi_n) && (30)\\ &\leq U(f(X_n)) + P(\hat{U}_J(f(X_{n+1})) - \hat{U}_J(f(X_n)) + \xi_n) && (31)\\ &\leq U(f(X_n)) + P(U(f(X_{n+1})) - U(f(X_n))) + 2P\xi_n. && (32)\end{aligned}$$

In the first equation (30), we used Hoeffding's inequality for $\mathbf{x}^* \in X_P^*$. In the second equation (31), we used the fact the input $\mathbf{x}_{n+1} \in \mathbb{X}$ is picked according to the approximate greedy strategy and in the last equation (32), we again used Hoeffding's inequality for the approximately greedily selected input $\mathbf{x}_{n+1} \in \mathbb{X}$. Applying Lemma A.1, we find that the following inequality holds with probability $1-2N\delta$:

$$U(f(X_N)) \geq (1-e^{-N/P})U(f(X_P^*)) - \sum_{n=1}^{N} 2\xi_{n-1}\left(1-\frac{1}{P}\right)^{N-n}.$$

■

A.8.2 Proof of Proposition 4.2

Firstly, by Hoeffding's inequality, we obtain the following concentration bound for the expectation over the surrogate model:

$$\mathbb{P}\left[\left|\frac{1}{H}\sum_{h=1}^{H}\hat{U}_J(\hat{f}_h(X)) - \mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[\hat{U}_J(\hat{f}(X))]\right| \geq \xi_{n,1}\right] \leq 2\exp\left(-\frac{2H\xi_{n,1}^2}{C_n^2}\right)$$

where $X = X_n \cup \{\mathbf{x}\}$ for any $\mathbf{x} \in \mathbb{X}$ and $C_n = \sup_{\boldsymbol{\theta}\in\Theta}\mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[S_{\boldsymbol{\theta}}(\hat{f}(\mathbb{X})) - S_{\boldsymbol{\theta}}(\hat{f}(X_n))]$. By manipulating this expression, we find that the inequality holds with probability $1-\delta$ if we set $\xi_{n,1} = C_n\sqrt{\log(2/\delta)/(2H)}$. Similarly, by applying Hoeffding's inequality again, we obtain the following concentration bound for the utility estimate:

$$\mathbb{P}\left[\left|\mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[\hat{U}_J(\hat{f}(X))] - \mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[U(\hat{f}(X))]\right| \geq \xi_{n,2}\right] \leq 2\exp\left(-\frac{2J\xi_{n,2}^2}{C_n^2}\right).$$

By manipulating this expression, we find that the inequality holds with probability $1-\delta$ if we set $\xi_{n,2} = C_n\sqrt{\log(2/\delta)/(2J)}$. We now upper bound the sum of these two parameters by

$$\xi_n := C_n\sqrt{\frac{2}{\min(J,H)}\log\left(\frac{2}{\delta}\right)} \geq \xi_{n,1} + \xi_{n,2}.$$

By repeating the same arguments as before, we obtain the following bound, which holds with probability $1 - 6\delta$:

$$
\begin{aligned}
U(f(X_P^*)) &\leq U(f(X_n)) + P(U(f(X_n \cup \{\mathbf{x}^*\})) - U(f(X_n))) && (33)\\
&\leq U(f(X_n)) + P\Big(\mathbb{E}_{p(\hat{f}|\mathcal{D}_n)}[U(\hat{f}(X_n \cup \{\mathbf{x}^*\}))] + \mathcal{A}_\delta(\mathbf{x}^*|\mathcal{D}_n) - U(f(X_n))\Big) && (34)\\
&\leq U(f(X_n)) + P\Big(\frac{1}{H}\sum_{h=1}^{H} \hat{U}_J(\hat{f}_h(X_n \cup \{\mathbf{x}^*\})) + \mathcal{A}_\delta(\mathbf{x}^*|\mathcal{D}_n) - U(f(X_n)) + \xi_n\Big) && (35)\\
&\leq U(f(X_n)) + P\Big(\frac{1}{H}\sum_{h=1}^{H} \hat{U}_J(\hat{f}_h(X_{n+1})) + \mathcal{A}_\delta(\mathbf{x}_{n+1}|\mathcal{D}_n) - U(f(X_n)) + \xi_n\Big) && (36)\\
&\leq U(f(X_n)) + P(U(f(X_{n+1})) - U(f(X_n))) + 2P(\mathcal{A}_\delta(\mathbf{x}_{n+1}|\mathcal{D}_n) + \xi_n). && (37)
\end{aligned}
$$

The first line (33) follows from equation (29). The second line (34) follows from applying the concentration bound (21) in Assumption 4.2. The third line (35) follows from applying Hoeffding's inequalities above. The fourth line (36) is obtained by using the fact that the point $\mathbf{x}_{n+1} \in \mathbb{X}$ is picked according to the approximate AEUI acquisition function. The final line (37) follows by applying the concentration inequalities again. As before, we can apply Lemma A.1 in order to obtain the final performance bound

$$
U(f(X_N)) \geq (1 - e^{-N/P})U(f(X_P^*)) - 2\sum_{n=1}^{N}(\mathcal{A}_\delta(\mathbf{x}_n|\mathcal{D}_{n-1}) + \xi_{n-1})\Big(1 - \frac{1}{P}\Big)^{N-n},
$$

which holds with probability $1 - 6N\delta$. By using the inequality, $1 - x \leq e^{-x}$ for all $x \in \mathbb{R}$, we obtain the final performance bound.

■

B Additional figures

For some more intuition, we include two extra figures, Figures 15 and 16, which illustrates the impact of the choice of utility function when we use the Bayesian optimisation algorithm with the EUI acquisition function (20). In both of these examples, the number of objectives is $M = 3$ and the utilities that we consider are the four special cases outlined in Section 3.1. For both of these problems, the standard R2 utility and hypervolume indicator are set to target the whole Pareto front, whilst the IGD+ and D1 utilities are used to target two complementary regions of the Pareto front. To elaborate, for the rocket injector problem (Figure 15), we used the IGD+ utility to target the part of the Pareto front where the second objective is larger than the first objective in the normalised space, whilst the D1 utility is used to target the other part of the front. However, for the vehicle safety problem (Figure 16), we used the IGD+ utility to target the part of the Pareto front where the third objective is larger than the first objective in the normalised space, whilst the D1 utility is used to target the other part of the front. Overall, we find that the results here support the findings that we made earlier in Section 5.1: namely, it is generally beneficial to optimise the same utility that is used to assess the performance, although there do exist some cases where using a different utility function leads to better performance. This latter result is likely attributed to problem-dependent features such as the geometry of the objective function and its Pareto front.

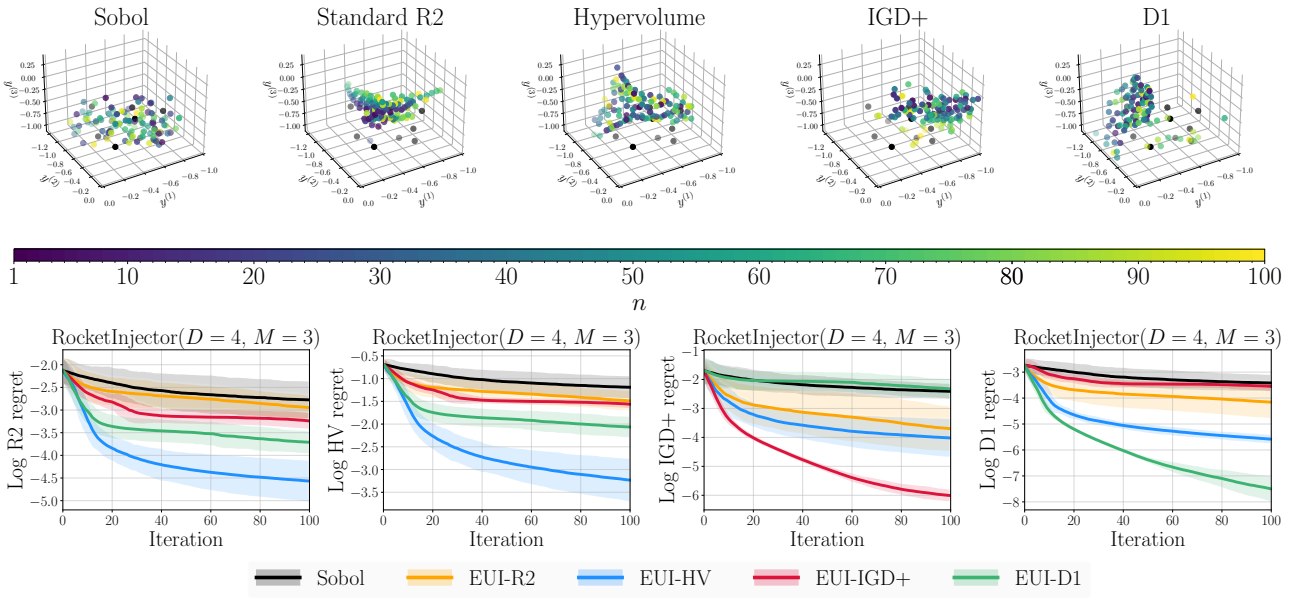

Figure 15: *An illustration of the EUI acquisition functions on the rocket injector problem. On the top row, we present a snapshot of the points obtained after one run of the Bayesian optimisation algorithm. On the bottom, we present the resulting performance plots.*

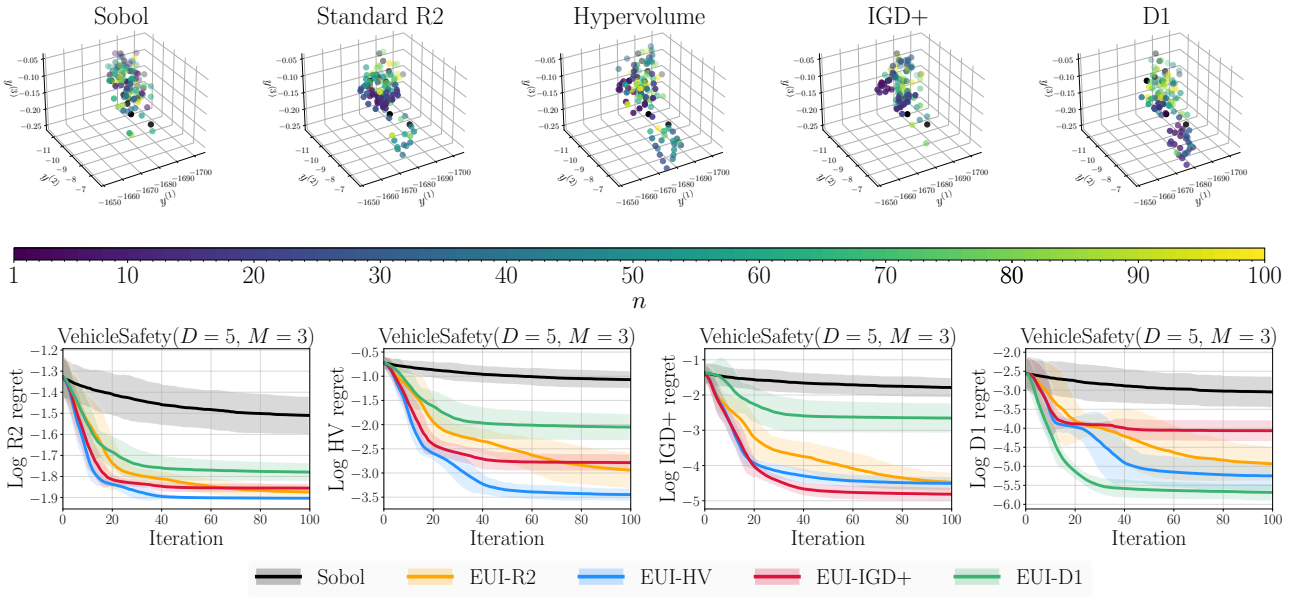

Figure 16: *An illustration of the EUI acquisition functions on the vehicle safety problem. On the top row, we present a snapshot of the points obtained after one run of the Bayesian optimisation algorithm. On the bottom, we present the resulting performance plots.*

C References

[1] R. Astudillo and P. Frazier. Multi-attribute Bayesian optimization with interactive preference learning. In *International Conference on Artificial Intelligence and Statistics*, pages 4496–4507. PMLR, June 2020. Cited on page 23.

[2] C. Audet, J. Bigeon, D. Cartier, S. Le Digabel, and L. Salomon. Performance indicators in multiobjective optimization. *European Journal of Operational Research*, 292(2):397–422, July 2021. Cited on page 7.

[3] F. Bach. Learning with submodular functions: A convex optimization perspective. *Foun-*

dations and Trends® in Machine Learning, 6(2-3):145–373, 2013. Cited on pages 7 and 13.

[4] M. Balandat, B. Karrer, D. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. In *Advances in Neural Information Processing Systems*, volume 33, pages 21524–21538. Curran Associates, Inc., 2020. Cited on page 24.

[5] S. Belakaria, A. Deshwal, and J. R. Doppa. Max-value entropy search for multi-objective Bayesian optimization. In *Advances in Neural Information Processing Systems*, volume 32, pages 7825–7835. Curran Associates, Inc., 2019. Cited on page 23.

[6] A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Press, Aug. 2009. Cited on page 31.

[7] D. Bertsimas, C. Pawlowski, and Y. D. Zhuo. From predictive methods to missing data imputation: An optimization approach. *Journal of Machine Learning Research*, 18(196):1–39, 2018. Cited on page 32.

[8] M. Binois, V. Picheny, P. Taillandier, and A. Habbal. The Kalai-Smorodinsky solution for many-objective Bayesian optimization. *Journal of Machine Learning Research*, 21(150):1–42, 2020. Cited on pages 13 and 23.

[9] E. Bradford, A. M. Schweidtmann, and A. Lapkin. Efficient multiobjective optimization employing Gaussian processes, spectral sampling and a genetic algorithm. *Journal of Global Optimization*, 71(2):407–438, June 2018. Cited on page 23.

[10] D. Brockhoff, T. Wagner, and H. Trautmann. On the properties of the R2 indicator. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, pages 465–472. Association for Computing Machinery, July 2012. Cited on page 10.

[11] S. R. Chowdhury and A. Gopalan. No-regret algorithms for multi-task Bayesian optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 1873–1881. PMLR, Mar. 2021. Cited on pages 14, 20, and 23.

[12] W. Chu and Z. Ghahramani. Preference learning with Gaussian processes. In *International Conference on Machine Learning*, pages 137–144. Association for Computing Machinery, Aug. 2005. Cited on page 31.

[13] T. Chugh. Scalarizing functions in Bayesian multiobjective optimization. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, July 2020. Cited on page 13.

[14] C. A. C. Coello and M. Reyes Sierra. A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm. In *MICAI 2004: Advances in Artificial Intelligence*, Lecture Notes in Computer Science, pages 688–697. Springer, 2004. Cited on page 11.

[15] P. Czyżżak and A. Jaszkiewicz. Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7(1):34–47, 1998. Cited on page 12.

[16] I. Das and J. E. Dennis. Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8(3):631–657, Aug. 1998. Cited on page 13.

[17] S. Daulton, M. Balandat, and E. Bakshy. Parallel Bayesian optimization of multiple noisy objectives with expected hypervolume improvement. In *Advances in Neural Information*

Processing Systems, volume 34, pages 2187–2200. Curran Associates, Inc., 2021. Cited on page 25.

[18] S. Daulton, S. Cakmak, M. Balandat, M. A. Osborne, E. Zhou, and E. Bakshy. Robust multi-objective Bayesian optimization under input noise. In *International Conference on Machine Learning*, pages 4831–4866. PMLR, June 2022. Cited on pages 28 and 31.

[19] G. De Ath, R. M. Everson, A. A. M. Rahat, and J. E. Fieldsend. Greed is good: Exploration and exploitation trade-offs in Bayesian optimisation. *ACM Transactions on Evolutionary Learning and Optimization*, 1(1):1:1–1:22, May 2021. Cited on page 29.

[20] K. Deb, S. Gupta, D. Daum, Jü. Branke, A. K. Mall, and D. Padmanabhan. Reliability-based optimization using evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 13(5):1054–1074, Oct. 2009. Cited on page 25.

[21] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, Apr. 2002. Cited on page 32.

[22] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600)*, volume 1, pages 825–830, May 2002. Cited on page 24.

[23] T. M. Deist, M. Grewal, F. J. W. M. Dankers, T. Alderliesten, and P. A. N. Bosman. Multi-objective learning to predict Pareto fronts using hypervolume maximization. In *arXiv.2102.04523*, Oct. 2021. Cited on page 33.

[24] J. Deng and Q. Zhang. Approximating hypervolume and hypervolume contributions using polar coordinate. *IEEE Transactions on Evolutionary Computation*, 23(5):913–918, Oct. 2019. Cited on pages 10 and 34.

[25] J.-A. Désidéri. Multiple-gradient descent algorithm (MGDA) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5):313–318, Mar. 2012. Cited on page 33.

[26] A. Deutz, M. Emmerich, and K. Yang. The expected R2-Indicator improvement for multi-objective Bayesian optimization. In *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 359–370. Springer International Publishing, 2019. Cited on page 23.

[27] N. Drechsler, R. Drechsler, and B. Becker. Multi-objective optimization in evolutionary algorithms using satisfiability classes. In B. Reusch, editor, *Computational Intelligence*, Lecture Notes in Computer Science, pages 108–117. Springer, 1999. Cited on page 32.

[28] M. Ehrgott. *Multicriteria Optimization*. Springer-Verlag, 2005. Cited on page 1.

[29] M. Ehrgott, J. Ide, and A. Schöbel. Minmax robustness for multi-objective optimization problems. *European Journal of Operational Research*, 239(1):17–31, Nov. 2014. Cited on page 31.

[30] G. Eichfelder. An adaptive scalarization method in multiobjective optimization. *SIAM Journal on Optimization*, 19(4):1694–1718, Jan. 2009. Cited on page 33.

[31] M. Emmerich, A. Deutz, and N. Beume. Gradient-based/evolutionary relay hybrid for computing Pareto front approximations maximizing the S-metric. In *Hybrid Metaheuristics*, Lecture Notes in Computer Science, pages 140–156. Springer, 2007. Cited on page 33.

[32] M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks. Single- and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10(4):421–439, Aug. 2006. Cited on page 23.

[33] J. Fliege, L. M. G. Drummond, and B. F. Svaiter. Newton's method for multiobjective optimization. *SIAM Journal on Optimization*, 20(2):602–626, Jan. 2009. Cited on page 33.

[34] J. Fliege and B. F. Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, Aug. 2000. Cited on page 33.

[35] J. Fliege and R. Werner. Robust multiobjective optimization & applications in portfolio optimization. *European Journal of Operational Research*, 234(2):422–433, Apr. 2014. Cited on page 31.

[36] C. M. Fonseca and P. J. Fleming. Multiobjective optimization and multiple constraint handling with evolutionary algorithms-Part I: A unified formulation. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 28(1):26–37, Jan. 1998. Cited on page 32.

[37] R. Garnett. *Bayesian Optimization*. Cambridge University Press, Jan. 2023. Cited on page 16.

[38] E. C. Garrido-Merchán and D. Hernández-Lobato. Predictive entropy search for multi-objective Bayesian optimization with constraints. *Neurocomputing*, 361:50–68, Oct. 2019. Cited on page 23.

[39] M. A. Gelbart, J. Snoek, and R. P. Adams. Bayesian optimization with unknown constraints. In *Uncertainty in Artificial Intelligence*, pages 250–259. AUAI Press, July 2014. Cited on page 32.

[40] A. P. Guerreiro, C. M. Fonseca, and L. Paquete. The hypervolume indicator: Computational problems and algorithms. *ACM Computing Surveys*, 54(6):119:1–119:42, July 2021. Cited on page 11.

[41] M. P. Hansen and A. Jaszkiewicz. Evaluating the quality of approximations to the non-dominated set. *Technical Report, Institute of Mathematical Modeling*, 1998. Cited on pages 3, 7, 8, and 10.

[42] J. Ide and E. Köbis. Concepts of efficiency for uncertain multi-objective optimization problems based on set order relations. *Mathematical Methods of Operations Research*, 80(1):99–127, Aug. 2014. Cited on pages 31 and 32.

[43] J. Ide and A. Schöbel. Robustness for uncertain multi-objective optimization: A survey and analysis of different concepts. *OR Spectrum*, 38(1):235–271, Jan. 2016. Cited on page 32.

[44] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima. Modified distance calculation in generational distance and inverted generational distance. In *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 110–125. Springer International Publishing, 2015. Cited on page 12.

[45] H. Ishibuchi, N. Tsukamoto, Y. Sakane, and Y. Nojima. Hypervolume approximation using achievement scalarizing functions for evolutionary many-objective optimization. In *2009 IEEE Congress on Evolutionary Computation*, pages 530–537, May 2009. Cited on pages 10 and 13.

[46] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, Dec. 1998. Cited on page 21.

[47] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 137–146. Association for Computing Machinery, Aug. 2003. Cited on page 15.

[48] J. Knowles. ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, Feb. 2006. Cited on pages 14 and 23.

[49] M. Konakovic Lukovic, Y. Tian, and W. Matusik. Diversity-guided multi-objective Bayesian optimization with batch evaluations. In *Advances in Neural Information Processing Systems*, volume 33, pages 17708–17720. Curran Associates, Inc., 2020. Cited on page 23.

[50] A. Krause and D. Golovin. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*. Cambridge University Press, 2014. Cited on pages 7, 13, and 35.

[51] H. W. Kuhn and A. W. Tucker. Nonlinear Programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, volume 2, pages 481–493. University of California Press, Jan. 1951. Cited on page 33.

[52] B. Li, J. Li, K. Tang, and X. Yao. Many-objective evolutionary algorithms: A survey. *ACM Computing Surveys*, 48(1):13:1–13:35, Sept. 2015. Cited on pages 3, 6, and 32.

[53] Z. J. Lin, R. Astudillo, P. Frazier, and E. Bakshy. Preference exploration for efficient Bayesian optimization with multiple outcomes. In *International Conference on Artificial Intelligence and Statistics*, pages 4235–4258. PMLR, May 2022. Cited on page 31.

[54] X. Ma, Q. Zhang, G. Tian, J. Yang, and Z. Zhu. On Tchebycheff decomposition approaches for multiobjective evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 22(2):226–244, Apr. 2018. Cited on pages 10 and 13.

[55] D. Mahapatra and V. Rajan. Multi-task learning with user preferences: Gradient descent with controlled ascent in Pareto optimization. In *International Conference on Machine Learning*, pages 6597–6607. PMLR, Nov. 2020. Cited on page 33.

[56] G. Malkomes, B. Cheng, E. H. Lee, and M. Mccourt. Beyond the Pareto efficient frontier: Constraint active search for multiobjective experimental design. In *International Conference on Machine Learning*, pages 7423–7434. PMLR, July 2021. Cited on page 23.

[57] T. Marler and J. S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, Apr. 2004. Cited on pages 4 and 13.

[58] K. Miettinen. *Nonlinear Multiobjective Optimization*. International Series in Operations Research & Management Science. Springer US, 1998. Cited on pages 1, 4, 5, and 10.

[59] J. Močkus. On Bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference Novosibirsk*, Lecture Notes in Computer Science, pages 400–404. Springer, 1975. Cited on page 21.

[60] N. Namura, K. Shimoyama, and S. Obayashi. Expected improvement of penalty-based boundary intersection for expensive multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 21(6):898–913, Dec. 2017. Cited on page 13.

[61] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming*, 14(1):265–294, Dec. 1978. Cited on pages 13 and 14.

[62] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer-Verlag, 2006. Cited on page 31.

[63] B. Paria, K. Kandasamy, and B. Póczos. A flexible framework for multi-objective Bayesian optimization using random scalarizations. In *Uncertainty in Artificial Intelligence*, pages 766–776. PMLR, Aug. 2020. Cited on pages 14 and 23.

[64] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, Jan. 2014. Cited on page 33.

[65] I. C. Parmee, D. Cvetković, A. H. Watson, and C. R. Bonham. Multiobjective satisfaction within an interactive evolutionary design environment. *Evolutionary Computation*, 8(2):197–222, June 2000. Cited on page 32.

[66] M. G. Parsons and R. L. Scott. Formulation of multicriterion design optimization problems for solution with scalar numerical optimization methods. *Journal of Ship Research*, 48(01):61–76, Mar. 2004. Cited on page 25.

[67] V. Picheny. Multiobjective optimization using Gaussian process emulators via stepwise uncertainty reduction. *Statistics and Computing*, 25(6):1265–1280, Nov. 2015. Cited on page 23.

[68] V. Picheny, M. Binois, and A. Habbal. A Bayesian optimization approach to find Nash equilibria. *Journal of Global Optimization*, 73(1):171–192, Jan. 2019. Cited on page 23.

[69] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze. Multiobjective optimization on a limited budget of evaluations using model-assisted S-metric selection. In *Parallel Problem Solving from Nature – PPSN X*, Lecture Notes in Computer Science, pages 784–794. Springer, 2008. Cited on page 23.

[70] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, 2006. Cited on page 20.

[71] N. Riquelme, C. Von Lücken, and B. Baran. Performance metrics in multi-objective optimization. In *2015 Latin American Computing Conference (CLEI)*, pages 1–11, Oct. 2015. Cited on pages 6 and 9.

[72] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, Oct. 2013. Cited on page 32.

[73] D. Russo and B. Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, Apr. 2014. Cited on page 14.

[74] H. Sato. Inverted PBI in MOEA/D and its impact on the search performance on multi and many-objective optimization. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 645–652. Association for Computing Machinery, July 2014. Cited on page 13.

[75] S. Schäffler, R. Schultz, and K. Weinzierl. Stochastic method for the solution of unconstrained vector optimization problems. *Journal of Optimization Theory and Applications*, 114(1):209–222, July 2002. Cited on page 33.

[76] O. Schutze, X. Esquivel, A. Lara, and C. A. C. Coello. Using the averaged Hausdorff distance as a performance measure in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 16(4):504–522, Aug. 2012. Cited on page 11.

[77] O. Sener and V. Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. Cited on page 33.

[78] K. Shang, H. Ishibuchi, M.-L. Zhang, and Y. Liu. A new R2 indicator for better hypervolume approximation. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, pages 745–752. Association for Computing Machinery, July 2018. Cited on pages 10 and 34.

[79] A. Singla, S. Tschiatschek, and A. Krause. Noisy submodular maximization via adaptive sampling with applications to crowdsourced image collection summarization. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2037–2043. AAAI Press, Feb. 2016. Cited on page 15.

[80] V. A. Sosa Hernández, O. Schütze, and M. Emmerich. Hypervolume maximization via set based Newton's method. In *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V*, volume 288, pages 15–28. Springer International Publishing, 2014. Cited on page 33.

[81] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning*, volume 27, 2010. Cited on page 23.

[82] R. Tanabe and H. Ishibuchi. An easy-to-use real-world multi-objective optimization problem suite. *Applied Soft Computing*, 89, Apr. 2020. Cited on page 25.

[83] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, Dec. 1933. Cited on pages 14 and 23.

[84] A. Trivedi, D. Srinivasan, K. Sanyal, and A. Ghosh. A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE Transactions on Evolutionary Computation*, 21(3):440–462, June 2017. Cited on pages 3, 6, and 32.

[85] B. Tu, A. Gandy, N. Kantas, and B. Shafei. Joint entropy search for multi-objective Bayesian optimization. In *Advances in Neural Information Processing Systems*, volume 35, pages 9922–9938. Curran Associates, Inc., 2022. Cited on page 23.

[86] T. Ulrich and L. Thiele. Bounding the efectiveness of hypervolume-based $(\mu + \lambda)$-archiving algorithms. In *Learning and Intelligent Optimization*, Lecture Notes in Computer Science, pages 235–249. Springer, 2012. Cited on pages 1 and 11.

[87] R. Vaidyanathan, K. Tucker, N. Papila, and W. Shyy. CFD-based design optimization for single element rocket injector. In *41st Aerospace Sciences Meeting and Exhibit*, Aerospace Sciences Meetings, pages 1–21. American Institute of Aeronautics and Astronautics, Jan. 2003. Cited on page 25.

[88] K. Van Moffaert, M. M. Drugan, and A. Nowé. Hypervolume-based multi-objective reinforcement learning. In *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 352–366. Springer, 2013. Cited on page 32.

[89] K. Van Moffaert, M. M. Drugan, and A. Nowé. Scalarized multi-objective reinforcement learning: Novel design techniques. In *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 191–199, Apr. 2013. Cited on page 32.

[90] T. Wagner, H. Trautmann, and D. Brockhoff. Preference articulation by means of the R2 indicator. In *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 81–95. Springer, 2013. Cited on pages 5 and 10.

[91] H. Wang, A. Deutz, T. Bäck, and M. Emmerich. Hypervolume indicator gradient ascent multi-objective optimization. In *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 654–669. Springer International Publishing, 2017. Cited on page 33.

[92] H. Wang, M. Olhofer, and Y. Jin. A mini-review on preference modeling and articulation in multi-objective optimization: Current status and challenges. *Complex & Intelligent Systems*, 3(4):233–245, Dec. 2017. Cited on page 12.

[93] B. D. Youn, K. K. Choi, R.-J. Yang, and L. Gu. Reliability-based design optimization for crashworthiness of vehicle side impact. *Structural and Multidisciplinary Optimization*, 26(3):272–283, Feb. 2004. Cited on page 25.

[94] D. Zhan and H. Xing. Expected improvement for expensive optimization: A review. *Journal of Global Optimization*, 78(3):507–544, Nov. 2020. Cited on pages 18 and 23.

[95] Q. Zhang and H. Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, Dec. 2007. Cited on page 13.

[96] R. Zhang and D. Golovin. Random hypervolume scalarizations for provable multi-objective black box optimization. In *International Conference on Machine Learning*, volume 37, pages 11096–11105. PMLR, Nov. 2020. Cited on pages 10, 14, 23, and 34.

[97] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49, Mar. 2011. Cited on pages 3, 6, and 32.

[98] Z.-H. Zhou, Y. Yu, and C. Qian. *Evolutionary Learning: Advances in Theories and Algorithms*. Springer, May 2019. Cited on page 32.

[99] L. M. Zintgraf, T. V. Kanters, D. M. Roijers, F. A. Oliehoek, and P. Beau. Quality assessment of MORL algorithms: A utility-based approach. In *Annual Machine Learning Conference of Belgium and the Netherlands*, June 2015. Cited on page 32.

[100] L. M. Zintgraf, D. M. Roijers, S. Linders, C. M. Jonker, and A. Nowé. Ordered preference elicitation strategies for supporting multi-objective decision making. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, pages 1477–1485, July 2018. Cited on page 31.

[101] E. Zitzler, J. Knowles, and L. Thiele. Quality assessment of Pareto set approximations. In *Multiobjective Optimization: Interactive and Evolutionary Approaches*, Lecture Notes in Computer Science. Springer, 2008. Cited on pages 5 and 10.

[102] E. Zitzler and L. Thiele. Multiobjective optimization using evolutionary algorithms - A comparative case study. In *Parallel Problem Solving from Nature*, Lecture Notes in Computer Science, pages 292–301. Springer, 1998. Cited on pages 1 and 10.

[103] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, Apr. 2003. Cited on page 10.

[104] M. Zuluaga, A. Krause, and M. Püschel. E-PAL: An active learning approach to the multi-objective optimization problem. *Journal of Machine Learning Research*, 17(104):1–32, 2016. Cited on page 23.