Imperial College London

Department of Computing

Black-Box Constrained Bayesian Optimisation With Tree Ensembles

MSc project report

Author:
Cornelius V. Braun

CID:
02081492

Supervisor:
Dr Ruth Misener

Second marker:
Dr Paul Bilokon

Submitted in partial fulfilment of the requirements for the MSc degree in Advanced Computing of Imperial College London

August 31, 2022

Abstract

Black-box constrained Bayesian Optimisation is a method to optimise black-box objective functions, that are expensive to evaluate, and subject to black-box constraint functions which can only be evaluated point-wise. This approach has been successfully applied to a plethora of real-world problems, such as the tuning of machine learning architectures. In this context, previous work focused exclusively on the usage of Gaussian Processes as surrogates to model the black-box objective and constraint functions. In contrast, tree ensembles are another model type that can be used for this purpose, and have advantages over Gaussian Processes due to their natural scaling to high dimensions and capability of handling continuous and categorical variables alike. Thus, this thesis introduces tree ensembles as surrogate models for black-box constrained optimisation problems. In previous works, the usage of tree ensembles as surrogate models has been discouraged due to a lack of reliable uncertainty metrics and strategies to optimise the response surface. We overcome both of these issues in the present work by proposing a novel uncertainty metric for Mondrian Forest tree ensembles, and the usage of the Nelder-Mead global optimisation strategy. Our experimental results demonstrate the strengths of our approach and showcase that tree ensembles outperform Gaussian Process surrogates on various Bayesian Optimisation benchmark problems.

Acknowledgements

I would like to thank my supervisor Dr Ruth Misener. This Msc thesis would not have been possible without her patience, advice, and support. I am very grateful for her contagious enthusiasm that she inspired me with, and most of all for the opportunities that she gave me. I could not have wished for a better mentoring throughout this project. I also thank my second marker, Dr Paul Bilokon for his feedback on my ideas.

I sincerely thank the members of the Computational Optimisation Group at Imperial College, in particular Shiqiang Zhang and Alexander Thebelt, whom I also thank for their implementation advice.

I would also like to thank all people who taught, supported, and helped me throughout all of my studies. In particular, I thank Dr Rasha Abdel Rahman and Dr Romy Frömer for awakening my enthusiasm for scientific research, and for their mentoring. In similar vein, I thank Dr Ozgur S. Oguz, for investing so much time into teaching me about scientific writing, and having an open ear for questions at all times.

Finally, I would like to thank my friends and family for their support and open ears that brought me through my studies. It would not have been possible without you!

Contents

List of Figures

List of Tables

List of Algorithms

Notation

Acronyms

ARD	automatic relevance determination
BART	Bayesian Additive Regression Trees
BO	Bayesian Optimisation
CMA-ES	Covariance Matrix Adaptation Evolution Strategy
CWEI	Constraint-Weighted Expected Improvement
EI	Expected Improvement
GBRT	Gradient Boosted Regression Tree
GP	Gaussian Process
IECI	Integrated Expected Conditional Improvement
LCB	Lower Confidence Bound
MCMC	Markov Chain Monte Carlo
MF	Mondrian Forest
NG-Boost	Natural Gradient Boosting
NM	Nelder-Mead
NN	Neural Network
PoF	Probability of constraint feasibility
QR	Quantile Regression
RF	Random Forest
RT	Regression Tree
SMAC	Sequential Model-Based Optimisation for General Algorithm Configuration

Symbols

c	Inequality constraint
d	Number of dimensions $d \in \mathbb{N}$.
I	Identity matrix $I \in \mathbb{R}^{d \times d}$
λ	Lifetime parameter of a Mondrian Process
$\mathcal{M}$	Surrogate model (unspecific type)
N	Number of iteration $N \in \mathbb{N}$
n	Tree node n in tree t
m_n	Empirical mean of observations at tree node n
v_n	Empirical var. of observations at tree node n
$\hat{p}$	Cluster centroid
σ^2_{noise}	Noise variance $\sigma^2_{noise} \in \mathbb{R}$
δ_n	Dimension δ that is split by node n
ξ_n	Value $\xi \in \mathbb{R}$ at which node n splits dimension δ_n
t	Tree t in ensemble $\mathcal{T}$
T	Number of trees in ensemble, $T = \lvert\mathcal{T}\rvert$
w_n	Weight of prediction of node n in a tree

η_i	Weight of prediction of tree t_i in ensemble
$\mathbf{x}^*$	Global optimiser of function f. $\mathbf{x}^* \in \mathcal{X}^{d \times d}$
$\mathbf{x}$	Data point $\mathbf{x} \in \mathcal{X}^{d \times d}$
X	Matrix of existing observations
$\mathbf{y}$	Labels of several datapoints X
y	Label of data point $\mathbf{x}$. $y \in \mathbb{R}$.

Sets

$\mathcal{C}$	Set of constraints
$\mathcal{D}$	Data set
$\mathcal{D}[d]$	Features of dataset $\mathcal{D}$ along dim. d
$\mathcal{K}$	Set of clusterings
$\mathbb{N}$	Natural numbers
$\mathbb{R}$	Real numbers
$\mathcal{T}$	Index set for trees of tree ensemble
$\mathcal{X}$	Input domain
$\mathcal{Y}$	Objective function target domain $\mathcal{Y} \subseteq \mathbb{R}$
$\mathcal{Z}$	Constraint function target domain $\mathcal{Z} \subseteq \mathbb{R}$

Functions and probability distributions

$\alpha(\cdot)$	Acquisition function
$\|\cdot\|$	Distance metric
$f(\cdot)$	Black-box function
$\mathbb{1}(\cdot)$	Indicator function
$k_{GP}(\cdot, \cdot)$	GP kernel covariance function
$\mathcal{L}(\cdot, \cdot)$	Loss function
$\textbf{left}(n)$	Left child node of node n
$\mu_{GP}(\cdot)$	GP mean function
$\mathcal{N}$	Gaussian distribution
$\Phi(\cdot)$	Cumulative Density Function (CDF) of $\mathcal{N}$
$relu(x)$	ReLU activtation function
$\textbf{right}(n)$	Right child node of node n
$path(\cdot)$	returns all nodes on the path from root to leaf
$\hat{t}(\cdot)$	Prediction of tree t on x
$\widehat{\mathcal{T}}_n(\cdot)$	The prediction of all trees in a GBRT ensemble after fitting iteration n.
$u(\cdot)$	Uncertainty quantification of surrogate model
$\mathcal{U}(\cdot)$	Utility function

Disclaimer: This work uses the convention of denoting any vector in **bold** and matrices containing multiple data point by capital letters, e.g. $\mathbf{x}$ for a single multidimensional data point and X for multiple points. Furthermore, this thesis uses the convention to minimise any objective function. To be consistent, this means that any objective from other works is converted into a minimisation problem if the original publication describes a maximisation problem. All constraints in this work are in the form of $c(\mathbf{x}) \leq 0$. We want to emphasise at this point that any inequality constraint in the form $c(\mathbf{x}) \leq rhs$ can be transformed into an equivalent constraint in the desired form as $c(\mathbf{x}) \leq rhs \equiv c(\mathbf{x}) - rhs \leq 0 \equiv \hat{c}(\mathbf{x}) \leq 0$.

Chapter 1

Introduction

About half of the global banana production is crucially endangered by the Panama disease, which is caused by a fungus named *Fusarium oxysporum* [1]. Bananas make up 25% of the daily calorie intake in some African countries [2], so protecting banana farms from the disease is a crucial issue. Since *Fusarium oxysporum* is resistant to existing fungicides [1], it is a key issue to develop a fungicide that helps to save existing banana populations.

Consider a company that attempts to design a new fungicide, which is specific to *Fusarium oxysporum*, in the quest to save the banana from the Panama disease. There are many degrees of freedom in the design of such a new chemical. For instance, which compounds shall be mixed, how much of which compound is used, and how long these compounds are mixed together, etc. The goal of the development process is to find a new chemical that reduces the vulnerability of banana plants to Fusarium wilt. Representing each design choice as a mathematical variable, a candidate fungicide can be described by the vector $\mathbf{x}$. Thus, the fungicide development problem can be formalised as a mathematical optimisation problem:

$$\arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \tag{1.1}$$

where $\mathbf{x}$ is a candidate fungicide and f is the objective function that measures the vulnerability of banana plants to the fungus after treatment with fungicide $\mathbf{x}$. Solving such a problem from scratch is generally very difficult due to the number of possible parameter combinations, i.e. the search space $\mathcal{X}$ being high-dimensional. Additionally, to evaluate the objective $f(\mathbf{x})$, a full product must be developed and tested on several plants, which may be time and cost intensive. These problem properties make simple approaches such as random search and grid search inefficient [3], as they suffer from the curse of dimensionality, even for an only 10-dimensional search space. To systematically address optimisation problems like the one described in Eq. 1.1, Mockus *et al.* [4] described an algorithm called *Bayesian Optimisation (BO)*. BO describes a procedure to solve hard optimisation problems, where the objective function is a black-box that can only be evaluated point-wise, and each evaluation of that function is expensive [3]. The principle of this procedure is illustrated in Fig. 1.1. The main advantage of BO over other methods such as grid search lies in the fact that it is an *informed search strategy* – i.e., it learns from each observation that is made and slowly converges towards a global solution. Consequently, BO has been successfully applied in many domains, for instance robotics [5], machine learning [6], or materials design [7], to produce results beyond the previous state-of-the-art. Following Shahriari *et al.* [8], BO is based on two main ingredients: First, a probabilistic *surrogate model* is used to learn the value of the objective function and to measure the uncertainty about the function estimates. Second, a heuristic, called *acquisition function*, is used to capture the benefits of

collecting data at a given point in the domain.

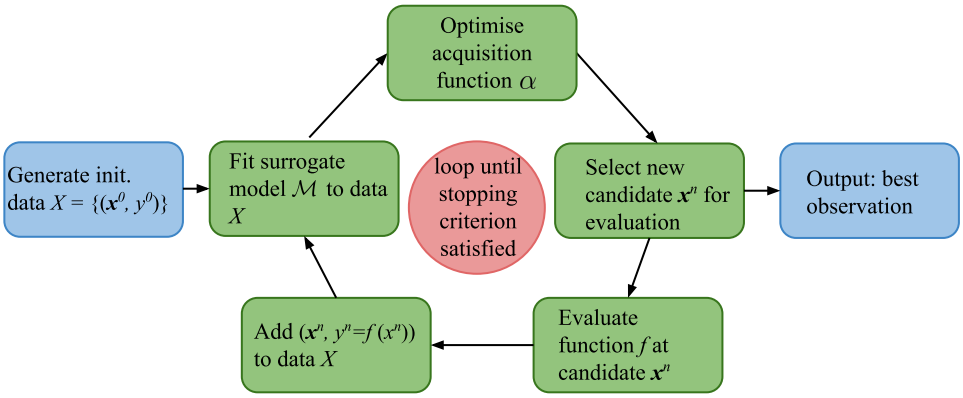


Figure 1.1: The Bayesian Optimisation loop [4]. The procedure is iterated until a stopping criterion is met.

An additional challenge to BO problems may be constraints in the input space, which can be *a-priori* known or unknown. In the context of our fungicide example, known constraints of the fungicide development process may be the sum of costs of all ingredients that are used. Unknown constraints in this process may be the knowledge that there exist combinations of compounds that may result in non-degradable waste, so it is crucial to avoid these combinations to protect the environment, but the specific combinations may be unknown due to the large combinatorial space. Another simple example of a black-box constraint function is the training time of a Neural Network (NN). While the objective function that is minimised during hyperparameter tuning quantifies the model error, model training times must often remain within a reasonable range, but cannot be inferred from the selection of hyperparameters alone, so this quantity is *a-priori* unknown. If BO constraints are unknown, they are also referred to as *black-box* constraints in literature [9]. Consequently, this thesis attempts to solve black-box optimisation problems of the following form:

$$\arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}), \qquad s.t.\ c_k(\mathbf{x}) \leq 0; \forall k \in \{1, \ldots, K\} \tag{1.2}$$

where c_k defines a constraint on the input variables. To address such problems, we perform BO with black-box constraint models, which is characterised by [3]:

1. **Black-box objective function:** The objective function f is not known in closed form and is can only be evaluated point-wise.

2. **Expensive function evaluations:** A single evaluation of the objective function f at point $\mathbf{x}$ is very resource intensive, e.g. it may require very costly studies to evaluate the value of a given candidate solution.

3. **Unknown constraints:** The optimisation problem is constrained by black-box constraints that are not known. Similarly to the black-box objective, the evaluation of these constraints is costly and only observed with noise.

The focus of this work lies on improving current surrogate models to model the functions in Eq. 1.2. In particular, it is our aim to establish tree ensembles as an alternative to Gaussian Process (GP) surrogate models for solving black-box constrained optimisation problems.

1.1 Motivation

Existing research has addressed the problem of BO from many angles, for example by using different acquisition functions and surrogate models [8]. The most common surrogate for BO is the GP [10]. GPs are a natural choice for regression problems in continuous domains due to their abilities to naturally quantify model uncertainty and to extrapolate well to data in unseen regions. In line with this, existing works that perform BO with known [11, 12, 13], or unknown constraints [9, 14, 15, 16, 17], do focus on GP surrogates. The usual approach in works on black-box constrained optimisation is to use a GP [10] surrogate model for each constraint, which models the probability that a data points falls within the feasible region.

Despite their popularity and strong performance on many problems, GPs are known to have weaknesses. Most GP kernel functions assume smooth objective functions [10]. Evidently, such objective functions are not necessarily given in real world applications, in particular if the data is categorical. Consequently, GPs may struggle on domains with categorical features, and their performance depends on the data transformation that is used to encode the features [18, 19, 20]. This is an issue for many fields, such as materials design or drug discovery, where categorical features are common [20]. Other known limitations of GPs include their declining performance on large datasets [21, 22], and in high dimensions [23]. There exist works that address these problems [24, 25, 26, 27], but most of these approaches address only one single of the aforementioned problems at a time and may require intricate implementations. Thus, there is a need for a BO surrogate that (*i*) naturally handles categorical and continuous data alike, (*ii*) scales well to high dimensions, and (*iii*) performs well on large datasets.

Regression Trees (RTs) are machine learning models that can be naturally used for regression on categorical and continuous data alike. Since single trees tend to overfit the training data, tree-based ensembles such as Random Forests (RFs), Gradient Boosted Regression Trees (GBRTs), and Mondrian Forests (MFs) have been proposed [28, 29, 30], which alleviate this problem. Tree ensembles have a multitude of desirable properties. They make accurate predictions [31], scale well to high-dimensional function spaces [32], and can naturally handle mixed function spaces, that consist of categorical and continuous features. This makes tree ensembles to be some of the most popular machine learning models for regression tasks [33]. In particular, tree-based surrogate models may be suited well to approximate the feasible region of a constrained BO problem, since both, optimisation constraints, and simple branches in a tree, are defined by *less-than-or-equal-to* conditions. Furthermore, tree ensembles are able to outperform GPs in simple regression tasks, as we will show in this work (please note that regression tasks are not equivalent to BO problems). We believe that these reasons make tree-based ensembles excellent candidates as surrogate models for BO with unknown constraints, so one of the main aims of this project is to introduce tree-based ensembles as surrogate models for BO with black-box constraints. To the best of our knowledge, the present work is the first report of using tree-based ensembles as surrogates to identify feasible regions in BO.

A major shortcoming of existing tree ensembles is that most models aim to estimate point-wise responses and thus lack a natural uncertainty estimate [29, 34]. Thus, Shahriari *et al.* [8] have identified the lack of reliable uncertainty estimates as one of the main issues that limit the performance of tree-based models as BO surrogates in BO applications. Several works in the past have aimed to address this issue by either proposing uncertainty metrics that are based solely on the underlying data [32], or based on the ensemble predictions [19, 30, 35, 36]. Still, existing uncertainty metrics for trees are known to have limitations. An issue with the uncertainty estimate currently used for MFs is that it can be overly uncertain in previously explored regions, leading to underexploration during BO. In line with this, it has been demonstrated, that the same metric for RFs may be over- or underconfident and estimate uncertainty intervals that may

be too wide or too narrow [37]. Therefore, we introduce distance-based uncertainty estimation as an alternative way to quantify the uncertainty of the predictions by MF ensembles. Similarly, Malinin *et al.* [38] introduced MC Dropout as an uncertainty estimate for GBRTs, but did not test its performance in BO settings. We thus are the first to establish MC Dropout as an uncertainty estimate for GBRTs as BO surrogates.

A second performance-limiting factor of tree ensembles that Shahriari *et al.* [8] identified is their non-differentiable response surface. Due to this issue, gradient-based optimisation strategies cannot be used to find novel candidate points for the black-box function evaluation. We will investigate multiple gradient-free, global, optimisation strategies to address this issue, and demonstrate that the Nelder-Mead algorithm [39] can be used effectively to perform this step in the BO loop.

1.2 Contributions

The aim of this thesis is to improve BO with black-box constraints. To achieve this goal, we make the following contributions:

1. **We introduce tree ensembles as surrogates for black-box constrained BO**. In Chapter 3, we describe how we use tree ensembles for black-box constrained BO, and in Chapter 6, we experimentally prove that tree ensembles can solve a diverse set of constrained black-box optimisation problems. In Section 6.5, we extend our results and show that tree ensembles are able to outperform GPs on multiple problems.

2. **We improve the uncertainty estimation of established tree ensembles** (Chapter 4). To this end, we introduce the distance-based uncertainty estimation for Mondrian Forests, and the MC Dropout uncertainty estimation for GBRTs which has not been used for BO before. In our experiments, we investigate the role of different uncertainty metrics in the optimisation of black-box optimisation problems with unknown constraints. Our results show that the distance-based uncertainty for MFs allows solving problems that cannot be solved using the commonly used uncertainty metric for MFs.

3. **We investigate gradient-free optimisation strategies to minimise the acquisition function** (Chapter 5). Our experiments demonstrate the superiority of the Nelder-Mead (NM) algorithm over sampling-based and Covariance Matrix Adaptation Evolution Strategy (CMA-ES) optimisation strategies on multiple high-dimensional optimisation benchmarks.

1.3 Report structure

First, Chapter 2 provides an extensive overview of the theoretical background that is the most relevant to this work. This includes an overview of prior work on the BO method (Section 2.1), tree ensembles for regression (Section 2.2), their previous usage in BO (Section 2.3) and uncertainty metrics for tree ensembles (Section 2.4).

The core of this project, introducing tree-based surrogate models for BO with black-box constraints, is introduced in Chapter 3. Then, we present two new ways to quantify surrogate uncertainty in BO (Chapter 4). The last contribution we make, which is the investigation of optimisation strategies for the BO acquisition function, is introduced in Chapter 5.

In Chapter 6, we evaluate all contributions that we make and compare our method to Gaussian Processes as state-of-the-art surrogate models. Finally, Chapter 7 highlights ethical aspects of the presented work, discusses the limitations of this work, and outlines potential future fields of research, before concluding this thesis.

Chapter 2

Background and related work

This chapter gives an overview of the theoretical background that is most relevant to the present research project. We begin by outlining the principles of BO in Section 2.1. Then Section 2.2 gives a brief summary of the most important work on tree-based regression models, notably RFs, GBRTs, and Mondrian Forests. Finally, Section 2.3 provides an overview of BO with tree-based surrogate models.

2.1 Bayesian Optimisation

The general problem setting of BO can be described as the global minimisation of a black-box objection function $f : \mathcal{X} \mapsto \mathcal{Y}$, which is expensive to evaluate [14]. Mathematically, this can be expressed as:

$$\arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \tag{2.1}$$

In this setting, the only assumptions about f are, that its domain $\mathcal{X}$ is known, and that f can be evaluated point-wise at any $\mathbf{x} \in \mathcal{X}$. This implies that the objective function f is treated as a black-box in BO, i.e., there is no known closed-form, let alone information on the differentiability of f. Generally, $\mathcal{Y}$ can be any region, but in the scope of the present project, we focus on $\mathcal{Y} \subseteq \mathbb{R}$, while $\mathcal{X}$ can include continuous, and categorical dimensions.

Given this setting, BO is a simple iterative algorithm, which has a structure that is often referred to as *ask-and-tell* [40, 41, 42]. Each iteration n consists of identifying a prospective candidate $\mathbf{x}^n$ for function evaluation (*ask*) and the evaluation of f at this point (*tell*), resulting in a new observation $y^n = f(\mathbf{x}^n)$, as illustrated in Figure 1.1. This makes BO a sequential optimisation process that terminates after $N \in \mathbb{N}$ iterations.

The most challenging step in this procedure is the selection of an appropriate candidate for the next (costly) query of the objective function. From a decision theoretical perspective, the BO aim is to add data to the set of observations $\mathcal{D}$, such that the negative expected utility is minimised during observation [43]. Given a utility function $\mathcal{U}$, this can be expressed as [8]:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \mathbb{E}_{f|\mathcal{D}} \left[-\mathcal{U}\left(\mathcal{D}, f\right)\right] \tag{2.2}$$

Unfortunately, this quantity is generally very difficult to evaluate or intractable, as it contains an expectation over the black-box function f. Thus, BO uses a heuristic approach to estimate how promising different regions in the input domain are. This heuristic is called *acquisition function*, and will be denoted by α in the remainder of this work. Using a tractable acquisition

function allows selecting the next candidate point at which the objective shall be evaluated, by minimising the acquisition function instead of the negative expected utility in Eq. 2.2. Please note that for the remainder of this work we follow the convention of minimising any objective, so we minimise any acquisition function.

BO acquisition functions are typically based on model estimations of the true objective function. In BO literature, a model that is used to estimate the value of the unknown objective is called *surrogate model*. As the true function f can only be evaluated point-wise, the purpose of a surrogate model $\mathcal{M}$ is to estimate the expected value of f at any point without requiring an actual (costly) evaluation of f. Additionally, surrogates typically provide estimates of the uncertainty on the current estimate of the function value at a point $\mathbf{x}$. This way, it becomes possible to choose the best candidate for the next actual function evaluation, based on previously seen data and surrogate confidence.

Algorithm 1: The Bayesian Optimisation algorithm [8, 14].

input : Objective function f, acquisition function α, time budget N

Initialise surrogate model $\mathcal{M}$
Initialise data $\mathcal{D}^0$ with specified number of initial observations
while $n < N$ **do**
  Select next candidate for evaluation: $\mathbf{x}^{n+1} \leftarrow \arg\min_{\mathbf{x}\in\mathcal{X}} \alpha(\mathbf{x} \mid \mathcal{D}^n)$
  Evaluate f at candidate: $y^{n+1} \leftarrow f(\mathbf{x}^{n+1})$
  Add new observation to dataset: $\mathcal{D}^{n+1} \leftarrow \mathcal{D}^n \cup \{(\mathbf{x}^{n+1}, y^{n+1})\}$
  Fit model $\mathcal{M}$ to data $\mathcal{D}^{n+1}$
  $n \leftarrow n + 1$
return $\arg\min_{(\mathbf{x},y)\in\mathcal{D}^N} \; y$

The pseudocode of the complete BO algorithm following Gelbart [3] and Shahriari *et al.* [8] is outlined in Algorithm 1, and is illustrated on a high level in Fig. 1.1. The algorithm illustrates that the BO loop is typically executed for a pre-defined number of iterations $N \in \mathbb{N}$. In each iteration of this loop, the acquisition function α is minimised to select a point at which f is evaluated. Then the observed data tuple $(\mathbf{x}, y)$ is added to the data, and the surrogate model $\mathcal{M}$ is fitted to the new dataset. Finally, the lowest objective value that was observed during the process is returned.

2.1.1 Surrogate models for BO

A surrogate model approximates the unknown function of BO problems. The only requirement for a BO surrogate is that it provides point-wise function estimates and uncertainty intervals for these estimates. Consequently, many regression models may be used as BO surrogates. The most common BO surrogate model is the Gaussian Process (GP) [10]. Other surrogate models that have been used for BO include tree-based regression ensembles, Tree-structured Parzen Estimators (TPE) [44], and multivariate adaptive regression splines (MARS) [45]. Tree-based regression models are at the heart of the current work, so we will outline this model class in detail in an upcoming section. Due to the popularity of GPs these models are commonly used as performance baselines, so we now provide a brief summary of GPs.

A GP defines a distribution over functions, of which any finite subset of function values has a joint Gaussian distribution [46]. A GP prior on a function $f : \mathcal{X} \mapsto \mathcal{Y} \subseteq \mathbb{R}$ is defined by a mean function $\mu_{GP}(\cdot)$, and a covariance kernel function $k_{GP}(\cdot, \cdot)$:

$$f(\mathbf{x}) \sim \mathcal{GP}\left(\mu_{GP}\left(\mathbf{x}\right), k_{GP}\left(\mathbf{x}, \mathbf{x}'\right)\right) \tag{2.3}$$

Given the above GP prior, a likelihood function, observations $(X, \mathbf{y})$, and query points X_*, a posterior distribution over objective function values can be computed in closed form. Let $K(X, X) \in \mathbb{R}^{d \times d}$ be the covariance function evaluated at all input pairs in X, that is, $[K(X, X)]_{ij} = k(\mathbf{x}^i, \mathbf{x}^j)$. Assuming observation noise $\sigma^2_{noise} \in \mathbb{R}$, the GP posterior distribution is given by:

$$\textit{mean} \qquad \overline{f}_* = K(X_*, X)\left[K(X, X) + \sigma^2_{noise} I\right]^{-1} \mathbf{y} \tag{2.4}$$

$$\textit{covariance} \qquad cov(f_*) = K(X_*, X_*) - K(X_*, X)\left[K(X, X) + \sigma^2_{noise} I\right]^{-1} K(X, X_*) \tag{2.5}$$

Thus, GPs allow to tractably compute the posterior distribution over all possible functions from the prior distribution, given the observations $(X, \mathbf{y})$.

2.1.2 Unconstrained acquisition functions

Acquisition functions quantify how promising a location in the input domain is for function evaluation. With this task at hand, acquisition functions need to balance two aspects: *exploration* and *exploitation*. On the one hand, exploration is necessary to identify unexplored regions in the input domain, which may contain minima of the objective function. On the other hand, exploitation is necessary to improve existing candidate solutions in already well explored regions. To effectively balance exploration and exploitation, acquisition functions make use of the predicted mean $\mathcal{M}(\mathbf{x})$ and uncertainty $u(\mathbf{x})$ of the underlying surrogate model $\mathcal{M}$. This information allows inferring an interval in which the true function value at the point $\mathbf{x}$ is the most likely to lie in. Two of the most common unconstrained acquisition functions for BO are the Lower Confidence Bound [47] and the Expected Improvement [4]:

1. **Lower Confidence Bound (LCB):** the LCB is one of the most simple acquisition functions that exist. This acquisition function acts as a pessimistic heuristic in the face of uncertainty. It is conservatively assuming the lowest possible value of the estimated objective function that is within some multiplicity of the estimated model uncertainty. Minimising this quantity consequently identifies points with either high uncertainty or low predictive mean. Assuming the model surrogate prediction at point $\mathbf{x}$ is denoted by $\mathcal{M}(\mathbf{x})$, the mathematical formulation of LCB is:

$$\alpha_{LCB}(\mathbf{x}) = \mathcal{M}(\mathbf{x}) - \kappa u(\mathbf{x}), \qquad \kappa > 0 \tag{2.6}$$

2. **Expected Improvement (EI):** the EI at a point $\mathbf{x}$ is typically defined as the expected improvement of the minimum observed objective value, if an evaluation of f is made at $\mathbf{x}$. Since we follow the convention of minimising the acquisition function, we present the negative Expected Improvement as EI acquisition function, which is:

$$\alpha_{EI}(\mathbf{x}) = -\mathbb{E}_{p(f)}\left[\max\left\{f(\mathbf{x}), y^\star\right\}\right] \tag{2.7}$$

where $y^\star = \min\{y_1, \ldots, y_n\}$ is the best objective value observed so far. Commonly, this expectation is described in its closed form, which exists under the assumption that the predictive distribution is Gaussian:

$$\alpha_{EI}(\mathbf{x}) = -u(\mathbf{x})\big(Z \cdot \Phi(Z) + \mathcal{N}(Z \mid 0, 1)\big), \qquad Z = \frac{\mathcal{M}(\mathbf{x}) - y^\star}{u(\mathbf{x})} \tag{2.8}$$

In recent years, more sophisticated acquisition functions of unconstrained BO have been proposed [48, 49]. For a more thorough overview of acquisition functions, we refer to Shahriari *et al.* [8].

2.1.3 Optimising the acquisition function

GPs are the most commonly used surrogate models in BO. As their predictive distribution is differentiable, the minimum of the acquisition function (line 4 of Algorithm 1) can be found using gradient-based optimisation techniques, L-BFGS [50] being the most popular method. Tree ensembles, on the other hand, do not permit to use gradient-based techniques due to their non-differentiable response surfaces. Instead, existing works have used sampling-based optimisation approaches [51, 52] or linear mixed integer programming [32]. In theory, any method for global optimisation can be used in this step. Two of the most popular global optimisation methods are the NM algorithm [39] with stochastic restarts, and evolutionary strategies such as the CMA-ES algorithm [53]. In Chapter 5 of this thesis, we present both of these methods as alternatives to the sampling-based optimisation of the acquisition function, and describe them in more detail in Chapter 5.

For a more in-depth overview of general BO methods, see Shahriari *et al.* [8] and Brochu *et al.* [54]. The present work focuses on BO in the presence of unknown, i.e. black-box constraints, which will be outlined next.

2.1.4 Bayesian Optimisation with black-box constraints

Traditionally, BO focuses on unconstrained settings [8]. Constrained BO, on the other hand, extends this setting with additional constraints. If the constraints are known, they may be easily added to the design by simply only considering feasible inputs as candidates for function evaluation (line 4 of Alg. 1) [32]. A more challenging problem arises when the constraint functions are unknown. This setting, where the objective and constraints are black-box functions, is known in literature as BO with black-box constraints. Within the realm of BO with black-box constraint, there is a further specification that can be made on the constraint properties. While *decoupled constraints* can be evaluated separately from the objective function, *coupled constraints* can only be evaluated when the objective function is evaluated [3]. The focus of the present work lies on coupled constraints, which implies that during minimisation of the objective function, BO has the goal to identify feasible regions in the input space.

Typically, the two goals of minimisation of the objective and identification of the feasible region are combined into a single objective. This is done using specific acquisition functions that are minimised at promising feasible points [3, 9]. Following Schonlau [55], the most general way of formulating an acquisition function for constrained problems is by taking the product of an unconstrained acquisition function and the *Probability of constraint feasibility (PoF)*:

$$\alpha_{cons}(\mathbf{x}) = \alpha(\mathbf{x}) \cdot PoF(\mathbf{x}) \tag{2.9}$$

where α is any acquisition function that quantifies the utility of evaluating the objective function at $\mathbf{x}$. Following the formulation of constrained BO in Eq. 1.2, we assume that all constraints can be expressed in the form of $c_k \leq 0$. Thus, the PoF quantifies the probability of the latent constraint functions c_k being smaller than or equal to zero at input $\mathbf{x}$:

$$PoF(\mathbf{x}) = \prod_{k=1}^{K} \int_{-\infty}^{0} P(c_k(\mathbf{x}) \mid \mathbf{x}, y)\, dc_k(\mathbf{x}) \tag{2.10}$$

This quantity can be estimated using a separate surrogate model $\mathcal{M}_{c_k}$ for every k-th constraint, which estimates the mean of the latent function c_k, and an uncertainty interval.

Based on the unconstrained acquisition functions mentioned above, generic *constrained* acquisition functions can be formulated. Following the general formulation of a constrained ac-

quisition function in Eq. 2.9, notable acquisition functions that in the presence of unknown constraints include:

1. **Constraint-Weighted Expected Improvement (CWEI)** [14, 16, 56]: This constrained acquisition function is simply the general constrained acquisition from Eq. 2.9, with $\alpha(\mathbf{x}) = \alpha_{EI}(\mathbf{x})$:

$$\alpha_{CWEI} = \alpha_{EI}(\mathbf{x}) \cdot PoF(\mathbf{x}) \tag{2.11}$$

   Consequently, the (negative) Expected Improvement (Eq. 2.8) is simply weighted by the probability that all constraints are satisfied at $\mathbf{x}$.

2. **Integrated Expected Conditional Improvement (IECI)** [9]: This acquisition function moves beyond CWEI, by not only considering the improvement at $\mathbf{x}$, if $\mathbf{x}$ is chosen as next for function evaluation, but instead it integrates the improvement over the entire input domain $\mathcal{X}$, given that $\mathbf{x}$ is added to the design:

$$\alpha_{IECI}(\mathbf{x}) = -\int_{\mathbf{x}' \in \mathcal{X}} \alpha_{EI}(\mathbf{x}' \mid \mathbf{x} \in \mathcal{D}) PoF(\mathbf{x}') d\hat{\mathbf{x}} \tag{2.12}$$

   where $\alpha_{EI}(\mathbf{x}' \mid \mathbf{x} \in \mathcal{D})$ describes the EI at $\mathbf{x}'$, given that $\mathbf{x}$ is added to the set of observations $\mathcal{D}$.

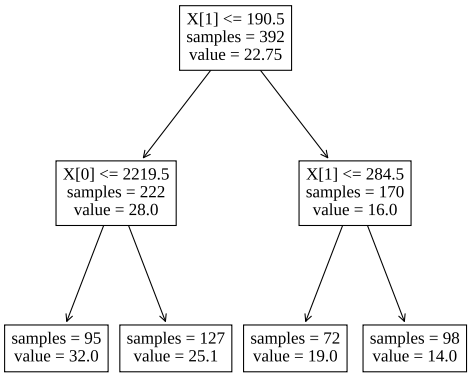
Unknown constraints are a well-studied topic in continuous domains [3, 16]. As mentioned above, the feasible region is typically approximated using acquisition functions that have high values across infeasible regions in the input space. To optimise the acquisition function over a continuous domain, the common choice of surrogate model is the GP [16, 57]. In mixed or categorical domains, non-GP surrogate models are generally a common choice. However, so far little work has been done on BO with unknown constraints in categorical domains. Daxberger *et al.* [18] used a linear basis function model and random Fourier features to perform constrained BO over mixed domains that included categorical dimensions. Still, all the investigated constraints were known, so inference over the feasible region was not necessary. Similarly, Thebelt *et al.* [32] used GBRTs to perform BO over various mixed domains in the presence of known constraints using tree-based surrogates. To our knowledge, we are unaware of any works that use tree-based surrogate models for BO with *unknown* constraints. In particular, learning the feasible region using tree-based surrogates is a novelty that we present in the present work.

2.2 Tree-based regression models

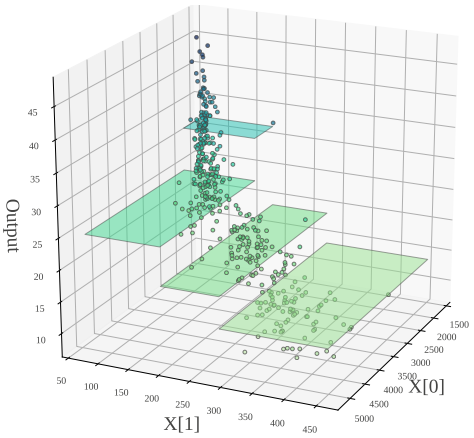
Regression Trees (RTs) are a simple, yet powerful tool for regression. A RT is constructed by recursively subdividing the input domain, where each split divides the given domain into two several parts, and can be represented as the node of a tree. This principle is illustrated in Figure 2.1. Given an input tuple (x_0, x_1), the model checks if $x_1 \leq 190.5$. Depending on the value of x_1, the next decision will either be $x_0 \leq 2219.5$, or $x_1 \leq 284.5$. The resulting regression output is now the mean response of the assigned leaf region. As you can see in Figure 2.1, this procedure results in a piecewise constant and discontinuous response surface.

There are various reasons for the popularity of RTs: They can naturally deal with categorical and continuous inputs at the same time, they are easy to interpret, and they scale well to large datasets and high dimensions [58]. Since single regression trees tend to have high variance [58], i.e., overfit the data, many modern applications have moved away from using single RTs. Instead,

so-called ensemble methods have been conceived as means of controlling the variance of RTs. In short, a tree-ensemble combines several individual RTs to predict a single output. The main difference between existing methods lies in how the individual trees are constructed, to obtain an ensemble. In the context of BO, the most prominent tree-ensemble methods are Gradient Boosted Regression Trees (GBRTs) [29, 59] and Random Forests (RFs) [34]. More recent tree-based ensemble methods that have produced promising results in various works include Bayesian Additive Regression Trees (BART) [36] and Mondrian Forests (MFs) [30]. Next, we will describe these methods in more detail.



(a) Decision tree nodes.



(b) Decision tree response surface. Training data is depicted as spheres.

Figure 2.1: Illustration of a regression tree. 2.1a illustrates the nodes that arise from the splits. 2.1b illustrates the observed data and the piece-wise continuous response surface of the tree.

2.2.1 Random forest models

Random Forests (RFs) [34] are a popular extension of single regression trees. An issue of single regression trees is their high prediction variance [58], as individual trees tend to overfit the training data. RFs address this issue by constructing several regression trees and using their average prediction as the final regression output. Let $\mathcal{RF}$ be a RF model consisting of a set of trees $\mathcal{T}$, where $|\mathcal{T}| = T \in \mathbb{N}$. The prediction of $\mathcal{RF}$ is the average of the prediction of all trees $t \in \mathcal{T}$ [58]:

$$\mathcal{RF}(\mathbf{x}) = \frac{1}{T} \sum_{t \in \mathcal{T}} \hat{t}(\mathbf{x}) \tag{2.13}$$

where $\hat{t}(\mathbf{x})$ is the prediction of tree t on data point $\mathbf{x}$. To construct each of the individual trees in an RF, a random subset of the data is sampled with replacement in a bootstrapping-like procedure called bagging [28]. Then, a regression tree is constructed for each sub-data set. Once T trees are constructed on T bootstrapped data sets, the prediction can be computed as the empirical mean of the tree responses (see Eq. 2.13).

2.2.2 Gradient-boosted tree models

Introduced by Friedmann [29], boosting methods combine several trees and thus incrementally reduce the prediction error. The main difference to the above ensemble methods such as Random

Forests (RFs), boosting constructs a model ensemble sequentially. Crucially, the idea behind GBRTs is to iteratively construct very small RT, so-called *weak-learners* or *stumps*, which reduce the residual with respect to the prediction of the previously constructed weak-learners.

Based on these building blocks, a GBRT ensemble is constructed from multiple stumps following an iterative procedure that minimises prediction error. Given a loss function $\mathcal{L}$ and the convention of denoting the prediction of stump t by $\hat{t}$, a first stump t_1 is defined as:

$$t_1 = \arg\min_t \sum_{i=1}^{|\mathcal{D}|} \mathcal{L}\left(y^i, \hat{t}\left(\mathbf{x}^i\right)\right) \tag{2.14}$$

We use $\widehat{\mathcal{T}}_n(\cdot)$ to denote the prediction of all trees in the ensemble after iteration n. Now, in iteration j of the fitting algorithm, the tree stump t_j is found by solving the following problem [58]:

$$\eta_j, t_j = \arg\min_{\eta,t} \sum_{i=1}^{|\mathcal{D}|} \mathcal{L}\left(y^i,\ \widehat{\mathcal{T}}_{j-1}(\mathbf{x}^i) + \eta\hat{t}\left(\mathbf{x}^i\right)\right) \tag{2.15}$$

An important theoretical result proves that this algorithm is equivalent to performing gradient descent in the function space of the loss function [60], given each step identifies the optimal stump t and weight η. Hence, the name *Gradient*-Boosted Regression Trees. The complete gradient boosting algorithm, following [58], is described in Alg. 2.

Algorithm 2: The GBRT fitting algorithm. Pseudocode adapted from Murphy [58].

input : Objective fct f, iteration budget N

Initialise $t_1 = \arg\min_t \sum_{i=1}^{|\mathcal{D}|} \mathcal{L}(y_i, \hat{t}(\mathbf{x}^i))$

Initialise set of trees $\mathcal{T} \leftarrow \{t_1\}$

for $n = 2 : N$ **do**

  Compute gradient residual: $r_i = -\left[\frac{\partial \mathcal{L}(y^i, \widehat{\mathcal{T}}_{n-1}(\mathbf{x}^i))}{\partial \widehat{\mathcal{T}}_{n-1}(\mathbf{x}^i)}\right]$, $i = 1, \ldots, |\mathcal{D}|$

  Construct next stump: $t_n = \arg\min_{\beta,t} \sum_{i=1}^{|\mathcal{D}|} \left(r_i - \beta\hat{t}(\mathbf{x}^i)\right)$

  Find weight: $\eta_n = \arg\min_\eta \sum_{i=1}^{|\mathcal{D}|} \mathcal{L}\left(r_i,\ \widehat{\mathcal{T}}_{n-1}(x_i) + \eta\hat{t}_n\left(\mathbf{x}^i\right)\right)$

return GBRT model: $\mathcal{GBRT}(\mathbf{x}) = \sum_{t_i \in \mathcal{T}} \eta_i \hat{t}_i(\mathbf{x})$

The result of this fitting procedure algorithm is a sum-of-trees model, which means that the resulting prediction of a GBRT ensemble $\mathcal{GBRT}$ is the weighted sum of all weak-learners:

$$\mathcal{GBRT}(\mathbf{x}) = \sum_{t_i \in \mathcal{T}} \eta_i \hat{t}_i(\mathbf{x}) \tag{2.16}$$

where η_i is the weight assigned to the prediction $\hat{t}_i(\mathbf{x})$ of the tree t_i.

2.2.3 Mondrian Forests

Mondrian Forests (MFs) [19, 30] are an ensemble of so-called Mondrian Tree structures [61]. Each Mondrian Tree can be sampled from a Mondrian Process, which is an algorithm that recursively

splits the input domain with axis-aligned cuts. The main difference between Mondrian Trees and conventional binary regression trees lies in the underlying fitting procedure. Classical regression trees define the splits in the tree by minimising an error criterion. Mondrian Trees, on the other hand, define the splits stochastically, so the split location and split value are determined probabilistically, based on the size of the area that a node currently covers, and not based on the data fit. A Mondrian Tree is constructed using the Mondrian Process algorithm that is outlined in Algorithm 3. Here, the stochasticity of the learning procedure is incorporated in two main steps: (*i*) the feature dimension δ_n at which the next node in the tree will split the input space is drawn with a probability proportional to the feature range along this axis (line 11), and (*ii*) the split value ξ_n for a chosen split dimension in sampled from a uniform distribution. Note that in line 11, u_{nd} and ℓ_{nd} represent the upper and lower bounds of the given feature dimension δ_n.

Algorithm 3: The Mondrian Process Algorithm to construct a single Mondrian Tree. Pseudocode adapted from Lakshminarayanan *et al.* [30].

1 **Function** `SampleMondrianTree(`*data* $\mathcal{D}$*, lifetime* λ`):`
2 Initialise: $\mathcal{MT} \leftarrow \emptyset$, LEAVES($\mathcal{MT}$) $\leftarrow \emptyset$, $\tau_{root} \leftarrow 0$
3 `SampleMondrianBlock(`*root,* $\mathcal{D}$*,* λ`)`
4
5 **Function** `SampleMondrianBlock(`*node* n*, data* $\mathcal{D}$*, lifetime* λ`):`
6 Add n to tree $\mathcal{MT}$
7 $\ell_{nd} \leftarrow \min(\mathcal{D}[d])$; $u_{nd} \leftarrow \max(\mathcal{D}[d])$
8 $E \sim \text{EXP}(\sum_d u_{nd} - \ell_{nd})$
9 $\tau_n \leftarrow \min(\lambda, \tau_{parent(n)} + E)$
10 **if** $\tau_n < \lambda$ **then**
11 $\delta_n \sim p(d)$, where $p \propto (u_{nd} - \ell_{nd})$
12 $\xi_n \sim \text{UNIFORM}\,(\min(\mathcal{D}[\delta_n]),\ \max(\mathcal{D}[\delta_n]))$
13 $\mathcal{D}_{left} \leftarrow \{x \in \mathcal{D} : x_{\delta_n} \leq \xi_n\}$; $\mathcal{D}_{right} \leftarrow \{x \in \mathcal{D} : x_{\delta_n} > \xi_n\}$
14 `SampleMondrianBlock(`LEFT(n)*,* $\mathcal{D}_{left}$*,* λ`)`
15 `SampleMondrianBlock(`RIGHT(n)*,* $\mathcal{D}_{right}$*,* λ`)`
16 **else**
17 Add n to LEAVES($\mathcal{MT}$)

Constructing a Mondrian Forest In Algorithm 3, a Mondrian Tree $\mathcal{MT}$ is constructed by recursively splitting the input space $\mathcal{X}$ recursively over a pre-defined period of time λ. For this, in each non-terminal call of `SampleMondrianBlock`, a new split is defined. To do this, the algorithm samples a number $E \in \mathbb{R}$ from an exponential distribution with a rate proportional to the area covered by the node (line 8). For example, if the root node splits the space $[0, 1]^2$ at 0.5 along each dimension, the resulting rate for sampling of the child nodes will be $1 + 0.5$, that is, the sum of the dimensions that a node covers. If $E \geq \lambda$, the process lifetime has ended, so the procedure stops and node n is assigned to be a leaf node.

Given that the lifetime has not yet ended, the area covered by node n is split into two parts by a split (δ_n, ξ_n), where δ_n defines the dimension along which the split is made and ξ_n the exact value of the split. For this, the dimension δ_n along which the split is made is sampled from the set of dimensions based on the observed feature ranges along the different dimensions. The idea here is, that features which differ a lot across the training data will capture the most information, so they can be used to classify the data well. Similarly, given a dimension, the value at which the dimension is split is sampled from a uniform distribution with equal probability for each

value along the axis. Once a split is sampled, this defines two new subsets of the data domain, $\mathcal{D}_{left}$ and $\mathcal{D}_{right}$, which contain all training data points that fall into the regions specified by the split.

Predictions using Mondrian Trees In difference to the previously discussed RT models, a Mondrian Tree prediction is not simply the mean of all observed training samples at a leaf node. Instead, a Mondrian Tree considers all the nodes on the path from root to leaf where a new instance is classified to. In this step, each node n defines the parameters of a Gaussian, using the mean m_n and empirical variance v_n over all data that is stored at this node n. Following this formulation, the predictive distribution $p(y \mid \mathbf{x}, t)$ of tree t is a mixture of Gaussians defined, by the weighted label distributions at each node on $path\,(t(\mathbf{x}))$. Formally, this distribution is defined as:

$$p(y \mid \mathbf{x}, t) = \sum_{n \in path(t(\mathbf{x}))} w_n(\mathbf{x}) \mathcal{N}(y \mid m_n, v_n) \tag{2.17}$$

where $n \in path\,(t(\mathbf{x}))$ are all nodes on the path from root to leaf, and m_n, v_n are the means and variances of all training samples that lie in the subregion covered by node n, and w_n is the weight that is associated with the prediction of node n. Consequently, the prediction of a single tree for a new data point $\mathbf{x}$ is:

$$\hat{t}(\mathbf{x}) = \sum_{n \in path(t(\mathbf{x}))} w_n(\mathbf{x}) m_n \tag{2.18}$$

Intuitively, this prediction procedure allows every node along the path to contribute to the prediction, specifically, based on how certain the classification of $\mathbf{x}$ at each node is.

Given a tree node n, the weight w_n is defined based on how likely it is that a new data point $\mathbf{x}$ lies within the region that is defined by that node n. Given the probability $p_n(\mathbf{x})$ that $\mathbf{x}$ does not lie within the region defined by n (Lakshminarayanan *et al.* [30] call this "branching-off"), the prediction weight $w_n(\mathbf{x})$ for node n is defined as:

$$w_n(\mathbf{x}) = \begin{cases} p_n(\mathbf{x}) \prod\limits_{k \in anc(n)} (1 - p_k(\mathbf{x})) & \text{if } n \text{ is no leaf} \\ 1 - \sum\limits_{k \in anc(n)} w_k(\mathbf{x}) & \text{else} \end{cases} \tag{2.19}$$

where $p_n(\mathbf{x})$ is estimated the as follows:

$$p_n(\mathbf{x}) = 1 - \exp\left(- \left(\tau_n - \tau_{parent(n)}\right) \left[\sum_{i=1}^{d} \left(\text{relu}\left(\mathbf{x}_i - u_{ni}\right) + \text{relu}\left(\ell_{ni} - \mathbf{x}_i\right) \right) \right] \right), \tag{2.20}$$

$$\tag{2.21}$$

Since $p_n(\mathbf{x}) = 0$ if $\mathbf{x} \in [\ell_{nd}, u_{nd}]$, the prediction of a Mondrian Tree in Eq. 2.18 can be simplified to:

$$\hat{t}(\mathbf{x}) = \sum_{n \in path(t(\mathbf{x}))} p_n(\mathbf{x}) m_n \prod_{k \in anc(n)} (1 - p_k(\mathbf{x})) \tag{2.22}$$

Predictions using Mondrian Forests As a Mondrian Forest is an ensemble of independent Mondrian Trees, the predictive distribution of the forest $\mathcal{MF}$ is defined as a sum-of-trees-model,

similar to RF ensembles. This results in the following predictive distribution:

$$p(y \mid \mathbf{x}) = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} p(y \mid \mathbf{x}, t) \tag{2.23}$$

where $\mathcal{T}$ is the set of individual trees in the ensemble. Consequently, the posterior predictive distribution of a MF is a normal distribution $\mathcal{N}$ of which the first moments is used for point predictions. The moment can be estimated empirically [62], resulting in the following MF prediction:

$$\begin{aligned} \mathcal{MF}(\mathbf{x}) &= \mathbb{E}_{p(y|\mathbf{x})}[y] \\ &= \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \hat{t}(\mathbf{x}) \\ &= \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \sum_{n \in path(t(\mathbf{x}))} w_n(\mathbf{x}) m_n \end{aligned} \tag{2.24}$$

2.2.4 Miscellaneous

2.2.4.1 Bayesian Additive Regression Trees

Bayesian Additive Regression Trees (BART) [36] are a Bayesian formulation of tree ensembles. Similar to RFs and MFs, the idea of BART is to use an ensemble of several trees to predict a single prediction at an input location $\mathbf{x}$. The main difference between BART and other tree ensembles lies in the way that the individual trees are constructed. While a Random Forest is constructed by directly minimising an error function, a BART ensemble induces a prior distribution $p(\theta)$ over the structure of the binary decision trees, and subsequently uses the posterior distribution $p(\theta \mid \mathbf{y})$ given training data, to sample the trees. Given an ensemble of sampled trees, BART inference works similarly to inference with other tree ensembles, such as RFs.

Using the above intuition, we do now outline the BART prior for binary trees. The purpose of the BART prior is to define the structure of the trees in the ensemble. The prior contains three stochastic components: First, $p(t_i)$ is a prior over the structure of the binary tree t_i, i.e. the tree depth, and splits that the tree encompasses. Second, $p(m_{\ell_{ij}} \mid t_i)$ is a prior distribution over the value of leaf node ℓ_{ij} in tree t_i, and $p(\sigma^2_{noise})$ is a prior over the noise variance. Assuming independence between these variables, the full BART prior distribution can be expressed as follows [63]:

$$p(\theta) = p(\sigma^2_{noise}) \left[\prod_{t_i \in \mathcal{T}} \prod_{\ell_{ij} \in t_i} p(m_{\ell_{ij}} \mid t_i) p(t_i) \right] \tag{2.25}$$

The specific formulation of $p(t_i)$, $p(\sigma^2_{noise})$, and $p(m_{\ell_{ij}} \mid t_i)$ can depend on domain knowledge, e.g. in scenarios where deep trees are desirable, $p(t_i)$ should put weight on such trees. For more information on the default choice formulation of $p(t_i)$, $p(\sigma^2_{noise})$, and $p(m \mid t_i)$, we refer to Chipman *et al.* [36], for other formulations of the BART prior components, readers are referred to the work of Hill *et al.* [64].

Given a prior distribution $p(\theta)$ and observations $\mathbf{y}$, tree ensembles can be sampled from the posterior $p(\theta \mid \mathbf{y})$. For this, Markov Chain Monte Carlo (MCMC) methods, such as the Metropolis Hastings algorithm [65, 66] or Gibbs sampling [67] can be used. Once a BART tree ensemble is defined, a point-wise prediction at $\mathbf{x}$ is then simply the average of all sampled tree

predictions, similar to Eq. 2.13. Additionally, it is easily possible to estimate model prediction uncertainty based on the quantiles of the predictions of the sampled trees. Hence, BART offers a natural way to estimate a predictive mean and standard deviation for BO applications.

2.2.4.2 NG-Boost

Recently, a novel boosting-based surrogate for unconstrained BO has been presented by Duan *et al.* [68]. The method, called Natural Gradient Boosting (NG-Boost), uses several GBRT ensembles that together define a mixture of Gaussians. For training, each GBRT ensemble is not used to approximate the predictive mean function directly, but instead, learn the parameters of the Gaussian $\mathcal{N}(\theta_1, \theta_2)$. Intuitively, NG-Boost achieves this by maintaining one GBRT ensemble $\mathcal{M}_{\theta_i}$ per distributional parameter θ_i, for instance $\mathcal{N}(\theta_1, \theta_2) = \mathcal{N}(\mathcal{M}_{\theta_1}(\mathbf{x}), \mathcal{M}_{\theta_2}(\mathbf{x}))$ given the above Gaussian. The result of the fitting procedure is a mixture of Gaussians. To construct each ensemble, the authors present a boosting-like approach, which performs gradient descent on a loss function. In difference to standard GBRTs, the NG-Boost algorithm does not optimise the constructed trees directly using gradient descent, but instead the parameters of the distribution predictive distribution are optimised via maximum likelihood estimation, which indirectly determines the ensemble.

2.3 Tree ensembles as Bayesian Optimisation surrogate models

Tree-based models were first used as BO surrogates by Hutter *et al.* [35], in their Sequential Model-Based Optimisation for General Algorithm Configuration (SMAC) method. Following this work, several other methods adopted RFs as BO surrogates [51, 52]. A major strength of RFs as BO surrogates is their low computational cost, making them one of the fastest BO surrogates available [19, 35]. This makes RFs particularly attractive in the context of large, and high-dimensional datasets. However, Nickson *et al.* [37] and Shahriari *et al.* [8] pointed out that the SMAC [35] surrogate and its empirical uncertainty estimates may be overconfident in previously unexplored regions, and that the RF predictive mean suffers from unrealistic extrapolations in regions with little data. Additionally, Lim *et al.* [69] demonstrated that RFs may also be underconfident, resulting in much larger uncertainty intervals than GPs. These results demonstrate that RFs may be unable to approximate the true underlying function well enough, and do not indicate appropriate information about the models' uncertainty with regard to this approximation. Another weakness of RFs for BO is that while RFs may perform better than GP surrogates in categorical or mixed domains [35, 70, 71], they have shown weaker performance than other surrogates, such as GPs, in continuous domains [32, 70, 72]. Since all of these results have been produced on problems with no, or only *a-priori* known constraints, the capabilities of RFs to learn the feasible region and minimise a black-box objective function simultaneously have not yet been established, but we hypothesise that other tree ensembles might be more suitable for this task.

Another tree-based surrogate for BO are boosting-based ensembles for regression, for example, GBRTs. In fact, several libraries for BO offer GBRTs as surrogate models [32, 73, 74]. It is a well-established fact in general machine learning literature, that GBRTs are among the best performing models for regression [33]. In line with this, Thebelt *et al.* [32] demonstrated that GBRTs can outperform RFs as BO surrogates in several scenarios. Similarly, Lim *et al.* [69] demonstrated that GBRTs may outperform RFs in unconstrained BO. Still, both evaluations suggest that there are scenarios in which GPs are superior surrogate models. The performance differences between these two surrogate models are not yet fully understood. It appears that

it heavily depends on the benchmark function if GBRTs beat GPs or not [19]. Since no work has tested GBRTs as surrogates for BO with unknown constraints, it is unclear which surrogate model is to be preferred.

A recent GBRT-based regression algorithm is NG-Boost [68]. While some works demonstrated that the NG-Boost method can achieve promising results [19, 68], outperforming other tree-based GBRT surrogate models in BO, Kim and Choi [19] noted that it does not scale to high-dimensional settings as well as other tree-based surrogates such as RFs or GBRTs. Furthermore, [75] showed that NG-Boost produces state-of-the-art uncertainty estimates, but its predictive mean tends to perform worse than standard GBRT models in regression settings.

Recent years also saw BART [36] be used as BO surrogates. In contrast to GBRTs and RFs, BART are derived using a Bayesian framework. Due to their Bayesian formulation, BART models tend to extrapolate better than other models, as predictions in regions far from training data will follow the prior distribution, much like GPs. This property makes BART attractive as BO surrogates, since exploration can be carried out more efficiently [20, 76, 77]. In line with this, BART were shown to be able to produce better outcomes than other tree-based surrogates when performing BO [20]. However, BART have some notable drawbacks as BO surrogates: due to their sampling-based training procedure, BART struggle in high dimensions [78], and fitting these models may be computationally more expensive than other tree-based surrogates, such as RFs and GBRTs [19, 20]. In particular, BART models are constructed in batch mode, i.e., they cannot be trained online sequentially, as GBRTs may be, which typically makes them slow surrogates [19].

Other tree-based Bayesian models that have been used as BO surrogates are MFs [19, 30]. Just like BART, MFs are a Bayesian approach that offers a prior. Again, this generally makes MFs extrapolate better than RFs and GBRTs to unseen data, since the model can fall back onto a sensible prior [78]. Hence, MFs have been successfully applied as surrogates in unconstrained BO in recent years [19, 78]. Different to BART, MFs do not require expensive sampling procedures and can be trained in batch mode: The fact that Mondrian trees can be constructed online makes them interesting for BO, since they have been shown to be fitted in $\mathcal{O}(|\mathcal{D}| \log |\mathcal{D}|)$, given the number of data points is denoted by $|\mathcal{D}| \in \mathbb{N}$. This gives MFs an advantage over BART ensembles, which also extrapolate well, but may struggle in high dimensions and take longer to train [19]. Since BO is a setting that heavily depends on the surrogates' ability to extrapolate, we believe that MFs may be effective surrogates for BO with unknown constraints. In light of this, the properties of MFs still require a better understanding in BO scenarios to comprehend their strengths and weaknesses.

The main question that remains unanswered in all previous work is whether tree ensembles can be used to learn feasible regions of BO problems and if so, how effective different tree ensembles are at doing so. Consequently, our aim is to close this gap in literature with the present work.

2.4 Quantifying uncertainty

2.4.1 Notions of uncertainty in ML and optimisation

To make statements about the uncertainty of predictions, it is necessary to understand the sources of uncertainty that exist. In the literature, there are two distinct sources of uncertainty that are commonly distinguished: *aleatoric* and *epistemic* uncertainty [79]:

- **Aleatoric uncertainty** describes the uncertainty due to the stochasticity of the observations [79, 80]. For example, this uncertainty can be caused by measurement noise, and it is the

reason why even if the underlying function were known, the prediction error would not reduce to zero since the data will be unlikely to be perfectly aligned with it, as illustrated in Fig. 2.2.

- **Epistemic uncertainty** on the other hand describes the uncertainty due to lack of knowledge about the underlying model of the data. Intuitively, this refers to not knowing what the underlying model should be in areas without training data, hence it is sometimes called *model uncertainty* [79]. In Figure 2.2, this issue is apparent, because it is impossible to extrapolate the correct function, given the current data, without making any assumptions.

It is important to note that while epistemic uncertainty can typically be reduced by collecting more data, or adjusting the algorithm in use, aleatoric uncertainty is more difficult to be reduced by optimisation methods, as it is related to the data itself [81].

Knowing about the different sources of uncertainty is particularly relevant in the context of BO, because the notion of uncertainty is at the heart of the underlying exploration-exploitation trade-off. If the assumed underlying function is continuous, the epistemic uncertainty is small in regions where data has already been observed. On the other hand, the uncertainty about the predictions will grow, the further away one moves from observed data. If the used uncertainty metric may reflect this, BO will automatically emphasise the exploration of unseen regions in the input space. Consequently, multiple authors have emphasised that it should be the goal of BO to reduce epistemic uncertainty [19, 38, 83].

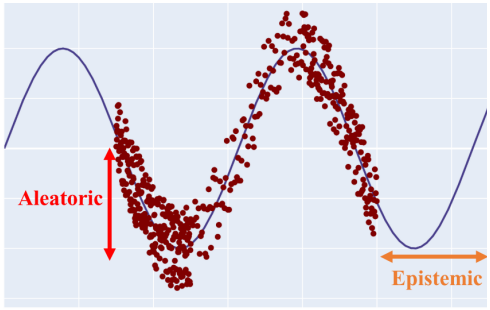


Figure 2.2: Illustration of epistemic and aleatoric uncertainty. Source: Abdar *et al.* [82].

2.4.2 Quantifying uncertainty in regression trees

Uncertainty estimation is a crucial part of BO, as the estimated uncertainty is used to balance exploration and exploitation. In fact, one of the main reasons for why tree-based surrogates are relatively unpopular for BO is due to the lack of reliable uncertainty intervals [8]. In this section, we give an overview of the efforts that have been made so far in constructing uncertainty intervals for regression trees or ensembles. The approaches to quantifying prediction uncertainty can be broadly classified into two categories: (i) the usage of data-based metrics, and (ii) the usage of ensemble-based metrics. Data-based uncertainty defines a metric based on the previously seen data points, independent of the underlying model, which means that they measure how similar a new input is to previously seen inputs. Ensemble-based metrics estimate uncertainty based on the trained model predictions, for instance the empirical variance of model predictions. Note that these categories are *not* synonymous with aleatoric and epistemic uncertainties, as for instance the variance of predictions across trees in an ensemble captures both aleatoric and model uncertainty [19]. For this section, we assume that a model prediction can be expressed as a predictive score, and an associated uncertainty interval around the predictive score. The width of the uncertainty interval will be denoted by $u(\mathbf{x})$.

2.4.2.1 Data-based uncertainty metrics

The most common data-based uncertainty metrics are based on distance metrics. In this scenario, given a distance metric $\|\cdot\|$, the uncertainty $u(\mathbf{x})$ at a point $\mathbf{x}$ is proportional to the distance of $\mathbf{x}$ to a specified point $\mathbf{x}'$ in the dataset $\mathcal{D}$. For example, Thebelt *et al.* [32] use such an uncertainty metric, where the specified point $\mathbf{x}'$ is the closest point to $\mathbf{x}$:

$$u(\mathbf{x}) = \min_{\mathbf{x}' \in \mathcal{D}} \|\mathbf{x} - \mathbf{x}'\| \tag{2.26}$$

where choosing $\|\cdot\|$ as Manhattan distance seems to yield better results than the Euclidean distance in high-dimensional settings [84]. This approach can be easily extended to any transformation of the distance. For instance, kernel-based surrogates, i.e. GPs [10], typically use stationary kernels, which rely on a distance metric to quantify prediction uncertainty. Although there is a plethora of kernel covariance functions for GPs that use other information than just the distance between two data points [46, 85], we are not aware of any data-based uncertainty measures that go beyond simple distance quantifications as in Eq. 2.26. A main strength of distance-based uncertainty estimates is that they only rely on the input features of previous observations. This makes distance-based uncertainty estimates invariant to observation noise, because they ignore the model response surface.

Despite their simple and meaningful nature in continuous domains, a shortcoming of these distance-based metrics is that they struggle to provide meaningful estimates in categorical domains. In particular, if categorical data is encoded by a one-hot encoding, distance-based uncertainty measures become binary quantities. Consequently, the estimated uncertainty intervals become nonsmooth, so only a small difference in inputs may lead to a large difference in uncertainty, which is undesirable for BO applications [86]. A way to address these shortcomings is to perform data transformations in order to obtain meaningful representations of categorical variables in a domain where distance metrics can be used efficiently. For instance, Thebelt *et al.* [32] employ the tree structure distance metric that was presented by Mišić [87] to obtain distance estimates in categorical domains. Additionally, input transformations may help to handle high-dimensional data, as the number of features may be reduced. In this vein, Thebelt *et al.* [32] propose a data-based uncertainty measure that relies on the distance in latent space. For this, the data is clustered using an arbitrary clustering algorithm such as k-Nearest-Neighbour, and subsequently the uncertainty is defined as distance to the closest cluster centroid $\hat{\mathbf{p}}_k$:

$$u(\mathbf{x}) = \min_{\hat{\mathbf{p}}_k \in \mathcal{K}} \|\mathbf{x} - \hat{\mathbf{p}}_k\| \tag{2.27}$$

To the best of our knowledge, no dimensionality reduction algorithm beyond clustering [32] or Principal Component Analysis [88] has been combined with tree-based surrogates yet. In the context of BO with non-tree-based surrogates, more advanced techniques have been applied for the construction of latent spaces. For instance, using the distance in the latent space of an autoencoder [89, 90] or neural networks [91].

2.4.2.2 Ensemble-based uncertainty metrics

Ensemble-based uncertainty metrics capture the differences of the individual model outputs across the ensemble. The idea behind this is that a certain prediction will be supported by large parts of the ensemble. This makes ensemble-based metrics naturally perform equally well on categorical and continuous data, as the individual model responses can be computed effectively irrespective of the domain.

Empirical prediction variance. A straightforward way to estimate uncertainty from an ensemble is to use the variation of the predictions directly. This approach assumes a Gaussian predictive distribution, and estimates the first and second central moment of this distribution empirically. Given we denote the mean and variance of all training labels assigned to leaf ℓ by m_ℓ and v_ℓ respectively, this means that for a given data point $\mathbf{x}$, the resulting uncertainty $u(\mathbf{x})$ is defined as:

$$\mathcal{M}(\mathbf{x}) = \mathbb{E}_{t \in T}[\hat{t}(\mathbf{x})] = \frac{1}{T} \sum_{t \in \mathcal{T}} \hat{t}(\mathbf{x}) \tag{2.28}$$

$$\begin{aligned} u(\mathbf{x}) &= \mathbb{V}_{t \in T}[\hat{t}(\mathbf{x})] \\ &= \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \left(\left(\sum_{\ell \in t} v_\ell \mathbb{1}(\mathbf{x} \in \ell) \right)^2 + \left(\sum_{\ell \in t} m_\ell \mathbb{1}(\mathbf{x} \in \ell) \right)^2 \right) - \mathbb{E}_{t \in T}[\hat{t}(\mathbf{x})]^2 \end{aligned} \tag{2.29}$$

as it was demonstrated by Hutter *et al.* [92] (remember that $\hat{t}$ denotes the prediction of tree t). This illustrates that, intuitively, predictions are uncertain if the trees within the ensemble disagree, as well as if the predictions made by the individual trees are varying a lot, i.e., are uncertain themselves. This uncertainty metric was popularised by Hutter *et al.* [35], who used the empirical variance of the individual tree estimates across the RF to perform BO with SMAC. Today, it is one of the most popular uncertainty estimates for tree ensembles, being proposed for Mondrian Forests [62, 78], NG-Boost [68], and BART [19].

While the empirical variance uncertainty estimate has the desirable property that it will typically grow, the further a data point is from the training data, it has been criticised as well. Crucially, for RF models, the variance is estimated across trees that are all trained on a slightly different data set, so Hüllermeier and Waegeman [93] noted that the uncertainty estimate captures aleatoric rather than epistemic uncertainty. This problem does not exist for stochastically trained tree ensembles such as MFs and NG-Boost, and variance-based uncertainty estimation has shown promising results for these models [19, 78]. Still, we are not aware of any work that compares the performance of these metrics to other possible uncertainty estimators for these surrogates, so the practical properties of this metric remain unclear.

Quantile regression. The idea behind the Quantile Regression (QR) method is to provide an uncertainty interval, much like a confidence interval, by training multiple ensembles on different quantiles of the data to predict the probability that a specified percentage of the data is below the given value. Mathematically, this can be achieved by changing the optimised loss function to the quantile loss [94, 95]. The quantile loss for a quantile of $0 < \tau < 1$ is as follows:

$$\mathcal{L}_\tau(y, y') = \begin{cases} (y' - y)\tau & \text{if } y' \geq y \\ (y - y')(1 - \tau) & \text{else} \end{cases} \tag{2.30}$$

Using this loss, three tree ensembles can be trained, which estimate the expected prediction value, the lower quantile bound, and the upper quantile bound. These tree values can subsequently be used as predictive mean and uncertainty estimate for BO. Due to its straightforward implementation and intuitive results, quantile regression is the underlying uncertainty metric in the Scikit-Optimize [73] implementation of GBRTs. Furthermore, quantile regression has been used for RFs [95, 96], and is the default uncertainty metric for BART ensembles [36].

Tree overlap proximity-based uncertainty. Another ensemble-based uncertainty metric for GBRTs was presented by Thebelt *et al.* [32]. Originally developed in Mišić [87], this metric defines uncertainty based on the concept of proximity. In tree ensembles, the proximity of two observations $\mathbf{x}, \mathbf{x}'$ is defined as the proportion of trees for which $\mathbf{x}$ and $\mathbf{x}'$ are classified to the same leaf:

$$\text{proximity}(\mathbf{x}, \mathbf{x}') = \frac{1}{T} \sum_{t \in \mathcal{T}} \mathbb{1} \left\{ \ell_t(\mathbf{x}) = \ell_t \left(\mathbf{x}' \right) \right\} \tag{2.31}$$

where $\ell_t(\mathbf{x})$ returns the leaf that $\mathbf{x}$ falls to in tree t. Using this metric of proximity, the uncertainty for a prediction made at $\mathbf{x}$ can be defined as:

$$u(\mathbf{x}) = \min_{\mathbf{x}' \in \mathcal{D}} \text{proximity}(\mathbf{x}, \mathbf{x}') \tag{2.32}$$

Despite having different theoretical embeddings, the empirical prediction variance and proximity-based uncertainty are very similar, since they are both low in the case where all trees classify a data point similarly. However, a weakness of the proximity-based uncertainty lies in it being ignorant of the actual prediction at the leaf node. Specifically, on functions that exhibit only gradual changes, this metric will be underconfident compared to a GP or MF, potentially leading to overexploration of the domain during BO.

Jackknifing. Another approach of estimating uncertainty intervals from RFs is the Jackknife [97]. To calculate a Jackknife estimate of the prediction variance in a RF, each sample in the data is iteratively excluded exactly once, and the remaining data is used to regress a prediction, over which finally the variance is used as an estimate of the model uncertainty. Eq. 2.33 gives an illustration of this procedure in mathematical terms. Let $\mathcal{T}$ be the set of trees in a random forest model, that was constructed from a set of bootstrap samples B. Let $\mathcal{T}_{-i}$ be the trees of the random forest ensemble that was trained on all bootstrap samples not containing the i-th data point. Then, the Jackknife uncertainty estimate can be defined as:

$$u(\mathbf{x}) = \frac{N-1}{N} \sum_{i=1}^{M} \left(\mathcal{M}_{\mathcal{T}_{-i}}(\mathbf{x}) - \mathcal{M}_{\mathcal{T}}(\mathbf{x}) \right)^2 \tag{2.33}$$

where $\mathcal{M}_{\mathcal{T}_{-i}}(\mathbf{x})$ defines the prediction of a random forest when all bootstrap samples containing data point i are excluded:

$$\mathcal{M}_{\mathcal{T}_{-i}}(\mathbf{x}) = \frac{\sum\limits_{\substack{t \in \mathcal{RF} \\ t \text{ not trained on } i}} \hat{t}(\mathbf{x})}{\left| \substack{t \in \mathcal{RF} \\ t \text{ not trained on } i} \right|} \tag{2.34}$$

and the mean of all trees is defined as:

$$\mathcal{M}_{\mathcal{T}}(\mathbf{x}) = \frac{1}{T} \sum_{t \in \mathcal{T}} \hat{t}(\mathbf{x}) \tag{2.35}$$

Despite being its theoretical properties, the Jackknife has been criticised for its performance in practise, specifically Scillitoe *et al.* [78] have noted that the Jackknife fails to account for uncertainty arising from training and test data differing in modelling tasks. Additionally, Jackknifing compares the predictions of the same model on slightly different data sets, which results in the estimation of aleatoric uncertainty, much like prediction variance as uncertainty estimate for

RFs [93].

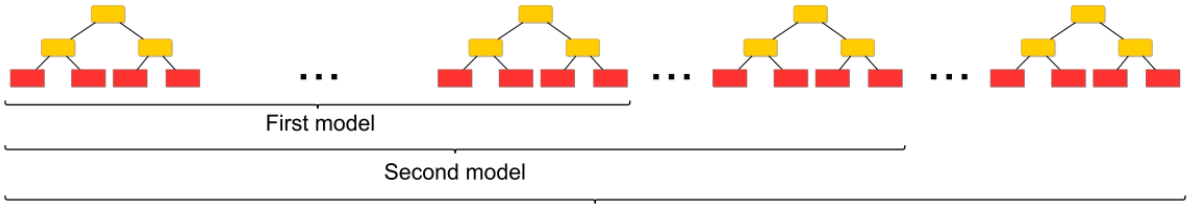


Figure 2.3: The virtual ensemble used by Malinin *et al.* Source [38].

Bayesian Model Averaging. Recently, Malinin *et al.* [38] presented Bayesian Model Averaging to estimate model uncertainty in GBRTs ensembles. This approach follows Gal and Ghahramani [98], who demonstrated that using dropout in a neural network during testing is equivalent to drawing samples from the distribution of model architectures, which can be used to MC approximate a GP in the limit. To apply this idea to GBRTs, Malinin *et al.* [38] train a single GBRT ensemble, and subsample multiple *virtual ensembles*, which are then considered as draws from the distribution of GBRT models. This idea is illustrated in Fig. 2.3. This approach offers important theoretical properties, as the uncertainty measure enables to capture uncertainty about the optimal model configuration, as the number of trees varies across the data set. However, all trees within the virtual GBRT ensemble are highly correlated by construction, which may ultimately lead to bad uncertainty estimates and overconfident surrogate models. Additionally, by construction, different trees in a GBRT ensemble will predict different parts of the input space better than others, so simply leaving out parts of the ensemble may lead to underfitting models. Furthermore, other uncertainty sources in the model, for instance prediction weights and tree depth, are not captured by this approach. Still, Malinin *et al.* [38] demonstrate that MC Dropout is a valid uncertainty metric, which can be successfully used in outlier detection tasks. However, it is unclear how well the Bayesian Model Averaging approach by Malinin *et al.* [38] translates to BO. To investigate this question, this work introduces MC Dropout as an uncertainty estimate in BO tasks.

Method	**Construction procedure**	**Uncertainty estimation strategies**
Random Forests [34]	Bagging [28]	Empirical prediction variance [35, 92] Jackknife [97] Quantile Regression [95]
Gradient-boosted trees [29]	Boosting [29]	Quantile Regression [95] Proxmity-based metric [32, 87] Bayesian Model Averaging [38]
Mondrian Forests [30]	Stochastic splitting of large regions in trees	Empirical prediction variance
NG-Boost [68]	Boosting w. natural gradients Optimising distributional parameters	Empirical prediction variance
BART [36]	MCMC sampling from posterior distribution	Quantiles of posterior samples [36] Empirical prediction variance [19]

Table 2.1: A summary of tree-ensemble-based surrogates for Bayesian Optimisation. The listed uncertainty strategies represent existing options for uncertainty estimation when using a specific tree ensemble. Note that only one uncertainty metric can be used at a time.

Chapter 3

Solving black-box constrained Bayesian Optimisation with tree ensembles

In this chapter, we propose to use tree ensembles as surrogate models for BO with black-box constraints. To this end, we model each function in the problem setting, i.e. the objective and constraint function(s), with a separate tree ensemble, and combine the resulting approximations in an appropriate acquisition function that is used to select the next candidate point at which the black-box objective and constraint functions are evaluated. To analyse the performances of different tree ensembles, we will compare three different tree-based surrogate models in this work:

1. Random Forests [34].
2. Gradient-Boosted Regression Trees [28]
3. Mondrian Forests [30, 62]

To the best of our knowledge, this is the *first* work that solves constrained BO with tree ensembles.

We begin this chapter with the mathematical formalisation of the optimisation problem that we try to solve (Section 3.1). Next, we provide a theoretical motivation for using tree-based ensembles to solve this problem (Section 3.2). We then describe the version of the Bayesian Optimisation algorithm that we use (Section 3.4), and lastly we discuss details of our implementation (Section 3.5).

3.1 Problem Formulation

BO with unknown constraints is a problem that is characterised by a black-box objective function which shall be optimised under the presence of (multiple) black-box constraint functions. We assume that the objective function, and each constraint function, can only be evaluated point-wise, i.e. the underlying function is unknown in its closed form. We furthermore assume that the evaluation of the objective and constraint conditions is expensive, so the number of point-wise evaluations shall be kept at a minimum.

Let $f : \mathcal{X} \mapsto \mathcal{Y} \subseteq \mathbb{R}$ be the objective function that shall be minimised. Note that $\mathcal{X} \subseteq R$ does *not* hold in general, since the feature space of many optimisation problems may contain categorical variables, e.g. the choice of activation function in a NN layer. Let $c : \mathcal{X} \mapsto \mathcal{Z} \subseteq \mathbb{R}$ be a constraint function, such that the corresponding constraint condition is satisfied iff. $c(\mathbf{x}) \leq 0$.

We define $\mathcal{C}(\mathcal{X}) = \{c_1(\cdot), \ldots, c_K(\cdot)\}$ as the set of black-box constraint functions, which all must be satisfied to solve the problem. This results in the following constrained optimisation problem:

$$\begin{aligned} \mathbf{x}^* &\in \arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \\ s.t.\ c_k(\mathbf{x}) &\leq 0, \qquad \forall c_k \in \mathcal{C}(\mathcal{X}) \end{aligned} \tag{3.1}$$

3.2 Motivation

This thesis is *not* the first work to address the problem in Eq. 3.1 via surrogate modelling. As outlined in Section 2.1, notable works include Schonlau *et al.* [56] and Gelbart *et al.* [14]. However, none of these previous works has modelled the unknown objective and constraint functions using any surrogate models other than GPs. This is surprising, as GPs are known to have limitations. For instance, their performance may deteriorate on large datasets [21, 22], and in high dimensions [23]. There do exist works that address these problems [25, 26, 27]. Still, GPs are known to struggle on multimodal functions [32], and a particular issue remains: while most synthetic benchmark problems are smooth, many real-world problems feature categorical inputs and non-smooth objective functions, for example in materials design [20]. GPs however, assume that the underlying function is smooth, so they may struggle on such problems. In contrast, tree ensembles are naturally able to model such non-smooth functions, as outlined Chapter 2. In addition, some tree ensembles are among the strongest machine learning models for regression tasks [31, 32]. This is illustrated in Table 3.1, where we compare the performance of tree ensembles as regression-point estimators to that of a GP with automatic relevance determination (ARD) 5/2-Matérn kernel on well-established benchmarks for regression [98, 99], and tree models outperform GPs on several datasets from literature. Thus, regression tree ensembles have great potential as BO surrogates. Additionally, the scalability of tree ensembles to high dimensions [100] compared to conventional GPs makes them ideal candidate surrogate models for learning feasible regions, since an independent model needs to be trained for each constraint.

Dataset	Gaussian Process	Random Forest	GBRT	Mondrian Forest
Concrete strength [101]	5.33 ± 0.14	5.42 ± 0.33	$\mathbf{4.57 \pm 0.36}$	7.12 ± 0.44
Energy efficiency [102]	$\mathbf{1.33 \pm 0.10}$	1.83 ± 0.11	1.36 ± 0.16	1.71 ± 0.28
Wine Quality Red [103]	0.80 ± 0.07	$\mathbf{0.59 \pm 0.05}$	0.62 ± 0.04	$\mathbf{0.59 \pm 0.05}$
Yacht Hydrodynamics [104, 105]	$\mathbf{0.38 \pm 0.04}$	1.25 ± 0.43	1.10 ± 0.35	6.66 ± 1.05

Table 3.1: Performance of tree ensembles and Gaussian Processes as regression point estimators. Average test performance in RMSE for several tree ensemble methods on popular regression benchmark problems that were used in Hernández-Lobato and Adams [99]. Displayed are mean error and one standard deviation across 5 runs. Best method is highlighted in **bold**.

3.3 Modelling constraints with tree ensembles

Following BO convention, we assume the predictive distribution of a surrogate model to be Gaussian [8, 9, 14, 32], i.e., it is defined by two parameters: the predictive mean and the standard deviation. We estimate these two parameters using (i) a surrogate model that estimates the predictive mean and (ii) an uncertainty metric, which estimates the second central moment of the predictive distribution. These two values are subsequently used as parameters of a multivariate Gaussian predictive distribution $\mathcal{N}(\mathbf{x} \mid \mathcal{M}(\mathbf{x}), u(\mathbf{x}))$, where $\mathcal{M} : \mathcal{X} \mapsto \mathbb{R}$ is a tree ensemble,

which takes an input $\mathbf{x}$ and returns a regression estimate, and $u_{\mathcal{M}}(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}$ is an uncertainty estimator that returns a point-wise uncertainty estimate of $\mathcal{M}$ at $\mathbf{x}$.

To learn the feasible region, we model each constraint c_k with a separate surrogate model $\mathcal{M}_{c_k}$. Consequently, given $K \in \mathbb{N}$ constraints, there will be $K+1$ surrogate models, which are trained: one surrogate model $\mathcal{M}_f$ that learns the objective function f, and one surrogate $\mathcal{M}_{c_k}$ for each constraint function c_k, $k \in \{1, \dots, K\}$. The *type* of model, i.e. GBRT or RF, is chosen to be consistent across all surrogates. For instance, if the objective f is modelled using a GBRT ensemble, so is each respective constraint function.

Given the set of constraints $\mathcal{C}(\mathcal{X})$, we follow the literature on constrained black-box optimisation, and use the CWEI acquisition function (Eq. 2.11) [16, 56], which is the product of the expected improvement and probability of constraint feasibility at a given input point. Following previous works [9, 16, 56], we assume independent constraints, which allows us to compute the PoF as:

$$\begin{aligned} PoF(\mathbf{x}) &= \prod_{k=1}^{K} P(\mathcal{M}_{c_k}(\mathbf{x}) \leq 0) \\ &= \prod_{k=1}^{K} \int_{-\infty}^{0} p(\mathcal{M}_{c_k}(\mathbf{x}') \leq 0)\ d\mathcal{M}_{c_k}(\mathbf{x}') \\ &= \prod_{k=1}^{K} \Phi(0 \mid \mathcal{M}_{c_k}(\mathbf{x}), u_{\mathcal{M}_{c_k}}(\mathbf{x})) \end{aligned} \tag{3.2}$$

where $\Phi(0 \mid \mathcal{M}_{c_k}(\mathbf{x}), u_{\mathcal{M}_{c_k}}(\mathbf{x}))$ denotes the cumulative density function (CDF) of the multivariate Gaussian distribution $\mathcal{N}(\mathbf{x} \mid \mathcal{M}_{c_k}(\mathbf{x}), u_{\mathcal{M}_{c_k}}(\mathbf{x}))$. Since the predictive distribution is assumed to be a Gaussian, we can exploit the closed form expression of EI [56], which we introduced in Eq. 2.8. Consequently, we optimise the following function:

$$\begin{aligned} \alpha_{CWEI}(\mathbf{x}) &= \alpha_{EI}(\mathbf{x}) \cdot PoF(\mathbf{x}) \\ &= u_f(\mathbf{x})\big(Z \cdot \Phi(Z) + \mathcal{N}(Z \mid 0, 1)\big) \left[\prod_{k=1}^{K} \Phi(0 \mid \mathcal{M}_{c_k}(\mathbf{x}), u_{\mathcal{M}_{c_k}}(\mathbf{x}))\right], \qquad Z = \frac{\mathcal{M}_f(\mathbf{x}) - y^\star}{u_f(\mathbf{x})} \end{aligned}$$

where $y^\star$ is the best feasible objective value that has been observed previously.

3.4 Optimising a constrained objective with tree ensembles

In the present work, we investigate constrained BO with tree ensemble surrogate models. To solve Eq. 3.1 while having a limited budget of evaluations $N \in \mathbb{N}$, we use the algorithm for constrained BO [9, 14, 16, 32], which can be seen as an extension of Algorithm 1. The resulting Algorithm is listed in Alg. 4. The main difference between the two aforementioned algorithms lies in the acquisition function and number of surrogate models. Instead of training a single surrogate model and using the EI acquisition function, Alg. 4 trains a separate surrogate model for each function in the problem, i.e. objective or constraint functions. The resulting model predictions are combined in the acquisition function that is minimised in line 4 of Algorithm 4. As stated above, we use the CWEI acquisition function [56] (Eq. 2.11), as it is probably the most widely used acquisition function for constrained BO, yielding strong results on a variety of problems [9, 14, 16]. Consequently, we minimise over $\alpha = \alpha_{CWEI}$ in line 4 of Algorithm 4. In the situation that no feasible point is known, the CWEI is undefined. In this case, we follow

Gardner *et al.* [16] and optimise only the second part of the equation, that is the PoF.

In addition to the specified input parameters in this algorithm, the following hyperparameters need to specified for optimisation: the tree ensemble type that is used for each surrogate model, the acquisition function that is minimised in the BO loop, and the optimisation strategy that is used in line 4 of the Algorithm. While we use the CWEI acquisition function in all experiments, we compare the performances of different tree ensembles in our experimental evaluation. Due to the non-differentiable response surfaces of tree ensembles, higher order optimisation techniques such as gradient descent or *L-BFGS* [50], which are used to optimise GP surrogate models, cannot be used. Instead, we rely on non-gradient based optimisation strategies, which are described in Chapter 5.

To initialise the surrogate models (lines 1 and 3 of the Algorithm), a number of random points is sampled from the domain $\mathcal{X}$ and evaluated with respect to their objective value and constraint feasibility. The results of these evaluations are used as initial training samples of the surrogate models. To sample pseudo-random points, we use Sobol's sequence [106], which guarantees exploration of the space and has been successfully used in BO [3, 19, 26].

Algorithm 4: The Bayesian Optimisation algorithm for problems with unknown constraints [9, 14, 32]. Differences to the unconstrained BO algorithm are highlighted in pink.

input : Objective fct f, constraint fcts $\{c_1, \dots, c_K\}$, acquisition fct α, time budget N

1 Initialise objective surrogate model $\mathcal{M}_f$

2 Initialise data $\mathcal{D}_f^0$ with specified number of initial observations

3 Init constraint surrogate models $\mathcal{M}_{c_1}, \dots, \mathcal{M}_{c_K}$

4 Initialise data $\mathcal{D}_{c_1}^0, \dots, \mathcal{D}_{c_K}^0$ with specified number of initial observations

5 **while** $n < N$ **do**

6 Select next candidate for evaluation: $\mathbf{x}_{n+1} \leftarrow \arg\min_{\mathbf{x} \in \mathcal{X}} \alpha(\mathbf{x} \mid \mathcal{D}_f^n, \mathcal{D}_{c_1}^n, \dots, \mathcal{D}_{c_K}^n)$

7 Evaluate objective function f at candidate: $y_f^{n+1} \leftarrow f(\mathbf{x}^{n+1})$

8 Add new observation to objective dataset: $\mathcal{D}_f^{n+1} \leftarrow \mathcal{D}_f^n \cup \{(\mathbf{x}^{n+1}, y_f^{n+1})\}$

9 Fit objective surrogate model $\mathcal{M}_f$ to data $\mathcal{D}^{n+1}$

10 **for** *constraint* $k \in [1, \dots, K]$ **do**

11 Evaluate constaint function c_k at candidate: $y_{c_k}^{n+1} \leftarrow c_k(\mathbf{x}_{n+1})$

12 Add new observation to constraint datasets: $\mathcal{D}_{c_k}^{n+1} \leftarrow \mathcal{D}_{c_k}^n \cup \{(\mathbf{x}^{n+1}, y_{c_k}^{n+1})\}$

13 Fit constraint surrogate model $\mathcal{M}_{c_k}$ to data $\mathcal{D}_{c_k}^{n+1}$

14 $n \leftarrow n + 1$

15 **return** $\arg\min_{(\mathbf{x},y) \in \mathcal{D}_f^N} y, \quad s.t.\ c_k(\mathbf{x}) \leq 0, \forall k \in [1, \dots, K]$

3.5 Implementation

The source code of our implementation is available on GitHub[1]. All code is implemented using version 3.9 of the Python programming language. We implemented Algorithm 4 based on the `scikit-optimize` implementation of unconstrained BO [73], and `entmoot` implementation of the BO algorithm with known constraints [32] [2]. Both software packages are licenced under the BSD 3-Clause Licence, which allows the usage and modification of the code. Using parts of existing software packages offers the benefit that the resulting software integrates well into the

[1] `https://github.com/cornelius-braun/constrained-bo-trees`

[2] `https://github.com/cog-imperial/entmoot`

`scikit-learn` API, which is used in both aforementioned software packages. For instance, this permits to easily integrate `scikit-learn`'s implementation of input spaces that allows to define continuous, mixed and discrete domains, from which points can easily be sampled. Following the licence requirements, each file that is based on a different software package, carries the copyright of the original author(s) and our copyright, as the contents were altered and extended. At a high level, our contributions encompass the implementation of the distance-based and MC Dropout uncertainties, the surrogate model wrappers, as well as the optimiser, which performs the Bayesian Optimisation. For a detailed overview of which code parts are new contributions, we refer to the listing in our GitHub repository[3].

We implement the tree ensembles using the following open source software packages: RF surrogate models are implemented using the `scikit-optimize` package [73], GBRTs surrogate models are implemented using the `LightGBM` package[4], and MFs using the `scikit-garden` package[5].

Tree ensemble training has many hyperparameters, such as the number of trees in the ensemble, or the minimum number of observations per leaf. For this work, we use the default values of the respective software packages with two exceptions: (*i*) since BO involves less training data than many other regression problems, we decrease the minimum number of observations per leaf from 20 to 2 for GBRTs (for MFs this is already the default), and (*ii*) we increase the number of trees in MF ensembles slightly from 10 to 20 to match the predictive performance of GBRT ensembles (which contain 100 trees per default).

Instructions on how to use the software can be found in our README[6]. For more info on the software usage, we refer to the Appendix.

[3] `https://github.com/cornelius-braun/constrained-bo-trees/blob/main/contributions.txt`
[4] `https://github.com/microsoft/LightGBM`
[5] `https://github.com/scikit-garden/scikit-garden`
[6] `https://github.com/cornelius-braun/constrained-bo-trees/blob/main/README.md`

Chapter 4

Improving the uncertainty estimation of tree ensemble surrogate models

In the previous chapter, we introduced tree ensembles as surrogate models in BO with unknown constraints. Shahriari *et al.* [8] identified the unsatisfactory performance of existing uncertainty metrics for trees as one drawback of using tree ensembles as BO surrogates. The aim of this chapter is to address this issue by establishing novel uncertainty metrics for MF and GBRT ensembles in the context of BO. The uncertainty metrics that we analyse are:

1. Distance-based uncertainty estimation for MFs
2. MC Dropout uncertainty estimation for GBRTs as BO surrogates

To the best of our knowledge, this is the *first* work that uses these uncertainty metrics in the context of BO.

In this chapter, we first present a motivation for why novel ways of estimating model uncertainty are needed (Section 4.1). Then, we present the distance-based uncertainty estimate for MFs (Section 4.2). Subsequently, we outline the MC Dropout uncertainty estimate for GBRTs (Section 4.3). Finally, we give an overview of the most important implementation details of these uncertainty estimates (Section 4.4).

4.1 Motivation

In BO, the predictive distribution of a surrogate model is typically assumed to be a multivariate Gaussian distribution, parametrised by the point prediction of a regression model and the predictive uncertainty of that model. Given this predictive distribution, various acquisition functions, such as the EI (Eq. 2.11) can be computed. In particular, in constrained settings, the uncertainty of the constraint surrogates is used to compute the probability of constraint feasibility, as we describe in Eq. 3.2. Consequently, a well-calibrated uncertainty estimate is essential to the success of BO, as the optimisation objective directly depends on it. For example, if the uncertainty estimate was too low for several regions, the learnt feasible region would be very small and the model will probably not explore in the regions where the uncertainty estimate is overconfident. Likewise, underconfident uncertainty estimates may lead to overexploration of the input space.

While GPs offer a natural quantification of uncertainty, Shahriari *et al.* [8] identified the lack of such uncertainty metrics as one of the main reasons why tree ensembles are less used as BO surrogates. Several works succeeding this claim have acknowledged this issue and made efforts

to overcome it [19, 30, 32, 87]. Still, for many tree ensemble types such as MFs, we are not aware of uncertainty estimates other than the empirical prediction variance being used, despite their known weaknesses that we discuss in Section 2.4.2. Motivated by this gap in literature, it is the aim of this chapter to improve the uncertainty estimation of GBRTs and MFs.

In particular, MFs have been shown to perform well as surrogate models [62, 78], but it appears likely that these results might at least partly be due to the strong performance of MFs as point estimators in regression settings. Specifically, the usage of the empirical prediction variance as standard uncertainty estimate of MFs has been criticised by Kim and Choi [19], since this estimate may be influenced by observation noise, leading to the underexploration of certain areas relative to others during BO due to high uncertainty estimations for already explored regions. To address this criticism, we introduce distance-based uncertainty estimation as a new way to quantify the prediction uncertainty of a MF. Since distance-based uncertainty estimates are invariant to observation noise, we hypothesise that distance-based uncertainty estimation may improve the performance of MF surrogate models. To the best of our knowledge, this work represents the first effort to improve the uncertainty estimation of MFs, and consequently is the first to use the presented uncertainty metric for MFs.

A second uncertainty estimate that is considered in this chapter is the MC Dropout uncertainty for GBRTs. Based on the successful use of MC Dropout as uncertainty estimate for NNs, Malinin *et al.* [38] presented MC Dropout uncertainty for GBRTs for outlier detection, but we are not aware of it being used in the context of BO before. Despite the promising results that Malinin *et al.* [38] report on outlier detection tasks, we hypothesise that this uncertainty estimate may not be suitable for BO, since the ensembles that are averaged in this procedure will be highly correlated, which should result in very low uncertainty estimates even in unexplored regions. To test this hypothesis, we are, to the best of our knowledge, the first to use the MC Dropout uncertainty estimate for GBRTs in the context of BO.

4.2 Distance-based uncertainty estimation for Mondrian Forests

To quantify the uncertainty of a MF prediction at a point $\mathbf{x} \in \mathcal{X}$, we propose to use a distance-based metric. Originally, this distance-based metric was conceived for GBRTs by Thebelt *et al.* [32], which we discussed in Section 2.4.2. The idea behind this uncertainty metric is to use the distance to the closest previously observed point as uncertainty quantification. Given continuous data, we quantify the distance between two data points using the Euclidean distance:

$$dist(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_2^2 = \sum_{d=1}^{D} \left(\mathbf{x}_d - \mathbf{x}'_d\right)^2 \tag{4.1}$$

Following Thebelt *et al.* [100], we replace $\|\cdot\|_2$ with the *Goodall4* distance metric [107] in categorical domains:

$$dist(\mathbf{x}, \mathbf{x}') = Goodall4(\mathbf{x}, \mathbf{x}') = \frac{1}{D} \sum_{d=1}^{D} S_d\left(x_d, x'_d\right) \tag{4.2}$$

where

$$S_d(x_d, x'_d) = \begin{cases} \frac{count(\mathcal{D}[d]=x_d)(count(\mathcal{D}[d]=x'_d)-1)}{|\mathcal{D}|(|\mathcal{D}|-1)}, & \text{if } x_d = x'_d \\ 0, & \text{otherwise} \end{cases}$$

Here, $count(\mathcal{D}[d] = x_d)$ quantifies how often the d-th feature in dataset $\mathcal{D}$ has the same category as point $\mathbf{x}$ along dimension d, and D denotes the number of dimensions of the input space. In mixed spaces, we sum the distance between two data points across all dimensions, choosing the Euclidean distance for continuous and *Goodall4* for categorical domains [100].

While the GP uncertainty estimate naturally scales with the predictive mean, distances do not scale with the magnitude of the observed function values. To circumvent this issue, we scale all uncertainty estimates using Min-Max-scaling to a range of $[0, \max_{\mathbf{x} \in \mathcal{D}} y]$, i.e. at most the largest absolute value observed so far. This makes it likely that the uncertainty intervals are likely to cover all uncertain predictions. Since the uncertainty directly influences the predicted acquisition function values, this scaling is crucial to ensure that the contribution of the local uncertainty estimate is large enough to encourage exploration.

Another challenge we need to address when using distance-based uncertainties, is that Euclidean distances grow to very large values quickly, in particular in higher dimensions. Without setting an upper bound to the distance-based uncertainty, the predictive uncertainty would grow too large too quickly. To circumvent this issue, we follow the approach by Thebelt *et al.* [32] and limit the uncertainty to be at most as high as the variance over all previous observations. This leads to the following scaled and clipped uncertainty estimate:

$$u_{dist}(\mathbf{x}) = \arg\min_{\mathbf{x}' \in X} \left[\min \left[dist(\mathbf{x}, \mathbf{x}') \cdot y_{largest},\ \mathbb{V}(\mathbf{y}) \right] \right] \tag{4.3}$$

where $\mathbb{V}(\mathbf{y})$ refers to the variance over all previous observations, and $y_{largest} = \max_{y \in \mathbf{y}}(|y|)$ is the largest absolute value observed so far.

4.3 MC Dropout uncertainty estimation for Gradient Boosted Regression Trees

The idea underlying MC Dropout uncertainty estimation is to embrace the uncertainty over the parameters of a machine learning model by assuming a distribution over all possible model configurations. In the case of GBRTs, an example of such parameters are the split values of the nodes in the trees. As the true distribution over all models is unknown, MC Dropout uncertainty estimation approximates the parameters of the distribution over all tree ensembles, and uses them as predictive mean and uncertainty estimate [98]. Introduced for outlier detection with GBRTs by Malinin *et al.* [38], this uncertainty estimate has not been introduced in the context of BO before.

Given data $\mathcal{D}$, the posterior distribution over GBRT model configurations is specified by:

$$p(t \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid t) p(t)}{p(\mathcal{D})} \tag{4.4}$$

Assuming a Gaussian prior $p(t)$ over the tree configurations in a GBRT ensemble, and a Gaussian likelihood $p(\mathcal{D} \mid t)$, the parameters of this distribution can be estimated via MC approximation, resulting in the predictive posterior $p(y \mid \mathbf{x}, \mathcal{D})$ [38], which is used as predictive distribution in BO. Specifically, the first moment, that is the mean of the distribution, is used for point estimates, and the second central moment, i.e the variance, is used as uncertainty estimate. However, to compute both moments, an intractable expectation needs to be computed [98]. In particular, the expectation $\mathbb{E}_{p(t|\mathcal{D})}$ must be computed, which is not possible in closed form. To circumvent this issue, Eq. 4.5 uses a sampling-based MC approximation, which averages the predictions of multiple GBRT ensembles that were drawn from the distribution $p(t \mid \mathcal{D})$. Similar

to the prediction variance estimate for MFs, this leads to point-prediction estimates of:

$$\begin{aligned} \mathcal{GBRT}_{MC}(\mathbf{x}) &= \mathbb{E}_{p(t|\mathcal{D})}\left[\hat{t}(\mathbf{x})\right] \\ &= \frac{1}{M}\sum_{m=1}^{M}\hat{t}(\mathbf{x}), \qquad t \sim p(t \mid \mathcal{D}) \end{aligned} \tag{4.5}$$

with the MC Dropout uncertainty quantified by the empirical prediction variance:

$$\begin{aligned} u_{MC}(\mathbf{x}) &= \mathbb{V}_{p(t|\mathcal{D})}[\hat{t}(\mathbf{x})] \\ &= \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\left(\left(\sum_{\ell\in t} v_\ell \mathbb{1}(\mathbf{x}\in\ell)\right)^2 + \left(\sum_{\ell\in t} m_\ell \mathbb{1}(\mathbf{x}\in\ell)\right)^2\right) - \mathbb{E}_{p(t|\mathcal{D})}[\hat{t}(\mathbf{x})]^2 \end{aligned} \tag{4.6}$$

which is equivalent to the expected value and variance across the predictions of all possible ensembles.

The main challenge in the computation of the resulting uncertainty estimate lies in drawing samples $t \sim p(t \mid \mathcal{D})$ from the posterior distribution over GBRT ensembles. To avoid training multiple tree ensembles in parallel, we employ the virtual ensemble approach in Fig. 2.3 by Malinin *et al.* [38]. This approach reduces the computational complexity of the estimation by sampling sub-ensembles from a single large GBRT ensemble. Since GBRT ensembles are constructed sequentially (see Algorithm 2), the sub-ensembles cannot be sampled completely randomly from the entire ensemble, without deteriorating the performance of the sub-ensemble. Instead, we restrict the drawn subsamples to encompass only sequentially constructed ensembles, i.e. tree 1 to tree k, for $k \leq K$, as illustrated in Figure 2.3. Malinin *et al.* [38] called this approach *virtual ensemble* construction and demonstrated its validity in their work.

4.4 Implementation

The implementation of all tree ensembles and the BO algorithm is the same as described in Section 3.5.

The implementation of the distance-based uncertainty is based on the implementation for GBRTs by Thebelt *et al.* [100]. However, we significantly improve the runtime of this implementation by using `numpy` broadcasting [108] to compute the distance to the closest previously seen observation. Additionally, we extend the implementation with the aforementioned Min-Max-scaling of the uncertainty measures.

The MC Dropout uncertainty estimate for GBRTs is implemented as described in [38]. This uncertainty estimate relies on a Monte Carlo approximate, which we approximate by sampling multiple sub-ensembles from a trained tree ensemble and average their outcomes. Important hyperparameters in this procedure are the size of the sub-samples and the number of sampling iterations. Following Malinin *et al.* [38], we only sample sub-ensembles that include at least half the number of estimators in the full ensemble. Since we only consider sub-ensembles that have been constructed sequentially, there are $\frac{ensemble\ size}{2}$ distinct sub-ensembles that can be sampled. Since MC approximations of integrals improve with the number of samples drawn, we use all $\frac{ensemble\ size}{2}$ distinct sub-ensembles in our computation. Following Section 3.5, we use the `LGBM` default parameters for GBRTs, which means that $ensemble\ size = 100$, so we sample 50 sub-ensembles to estimate the MC Dropout uncertainty estimate.

Chapter 5

Selecting the optimisation strategy to minimise the acquisition function

In this thesis, we introduce tree ensembles as surrogate models in BO with unknown constraints. Shahriari *et al.* [8] identified two main reasons for why tree ensembles are unpopular as BO surrogate models: the lack of reliable uncertainty estimates, and the non-applicability of gradient-based optimisation strategies due to the non-differentiable response surfaces of tree ensembles. While we addressed the first issue in the previous chapter, it is the aim of this chapter to address the second issue by improving the optimisation of the acquisition function in Algorithm 4 (see line 4). To this end, we discuss three of the most popular gradient-free optimisation strategies:

1. Sampling-based optimisation
2. The Nelder-Mead (NM) algorithm with stochastic restarts [39]
3. The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [53]

This chapter is outlined as follows, we first present a motivation for improving optimisation strategy of the acquisition function (Section 5.1). Then, we discuss the optimisation techniques that we use in the present work (Section 5.2), and finally we outline the implementation details of these techniques (Section 5.3).

5.1 Motivation

Optimising the acquisition function is crucial for the success of BO. In each iteration of Algorithm 4, the next candidate point at which the objective and constraint functions are evaluated is chosen by minimising the acquisition function (Algorithm 4, line 4). If the optimisation strategy that is used in this step fails to determine the true minimum, expensive evaluations of the oracle will be wasted. In the absence of higher-order information, previous works using tree ensembles as surrogate models in unconstrained optimisation or BO with known constraints, commonly used sampling-based optimisation to minimise the acquisition function [19, 51, 73]. However, this optimisation strategy has major weaknesses. In particular, it offers no theoretical guarantees, as the optimisation is stochastic, and furthermore, its performance deteriorates in higher dimensions. The latter is due to the fact that the number of samples that are needed to cover the input space increases exponentially with the number of dimensions. To circumvent these issues, Thebelt *et al.* [32] presented a mixed integer formulation of BO with trees that can

be solved using mixed integer program solvers such as Gurobi[1]. In their experiments, Thebelt *et al.* [32] prove that this approach of optimising the acquisition function works well in unconstrained settings, and settings with known constraints. However, their approach comes with several weaknesses. Since it only allows for linear or quadratic acquisition functions to be used, Thebelt *et al.* [32] use the LCB acquisition function. While this function may offer theoretical guarantees, its practical outcomes heavily depend on the tuning of the exploration hyperparameters, leaving it vulnerable to over- or underexploration [109, 110]. A second shortcoming of formulating BO as a quadratic mixed integer program lies in the fact that the predictive mean of some tree ensembles, such as MFs, is *not* linear, so these surrogate models cannot be used when following this approach. To alleviate these issues, we propose using other popular algorithms for global optimisation. Specifically, we discuss the NM algorithm [39], and CMA-ES [53] as alternatives to sampling-based optimisation for global gradient-free optimisation.

5.2 Optimisation of the acquisition function when using tree ensemble surrogates

In the present work, we use and compare three methods for minimising the equation in Eq. (2.11): (*i*) sampling-based optimisation, (*ii*) the CMA-ES algorithm [53], and (*iii*) the NM algorithm [39]. This section provides a short overview of the methods that will be evaluated. While we expect that the latter two methods both outperform sampling-based optimisation, we will investigate empirically in a later chapter which of the two latter methods performs better.

Algorithm 5: The CMA-ES algorithm. For a detailed discussion of how the distributional parameters μ, σ, and C are updated in lines 8-10, we refer to Hansen [111] and Dang *et al.* [112]. Algorithm adapted from Dang *et al.* [112].

input : Objective fct f, N points keep k, Init sigma σ, Population size λ, time budget N

1 Initialise $C \leftarrow I,\ p_c \leftarrow 0, p_\sigma \leftarrow 0,\ n \leftarrow 0$
2 **while** *not converged* $\wedge\ n < N$ **do**
3 Sample population $\mathbf{x}^i \sim \mathcal{N}(mean = \mathbf{0}, covariance = C)$, for $i \in [1, \ldots, \lambda]$
4 // update mean
5 $\mathbf{m} \leftarrow \mathbf{m} + \sigma\mathbf{z}$, where $\mathbf{z} = \sum_{i=1}^{k} w_i \mathbf{x}^i$
6 // update step-size
7 $p_\sigma \leftarrow$ `update_p_sigma`$(p_\sigma, \sigma, C, \mathbf{p})$
8 $\sigma \leftarrow$ `update_sigma`(σ, p_σ)
9 // update covariance matrix
10 $p_c \leftarrow$ `update_pc`$(p_c, \mathbf{p})$
11 $C \leftarrow$ `update_covariance`$(C, \sigma, p_c, \mathbf{x}^1, \ldots, \mathbf{x}^\lambda)$
12 $n \leftarrow n + 1$
13 **return** $\arg\min_{\mathbf{x} \in \mathcal{D}} f(\mathbf{x})$

5.2.1 Sampling-based optimisation

Sampling-based optimisation stochastically minimises a function. To this end, the target function is evaluated at a specified number of random points, and the best observation across all of these evaluations is used as the minimum of the function. In our case, the target function is the

[1] `https://www.gurobi.com/`

CWEI acquisition function, which is defined by one or multiple surrogate models, and the best of these evaluations of the acquisition function is used as the next candidate in line 4 of Alg. 4. Typically, this method's weaknesses are exposed in high domains, where a very high number of samples is needed to cover the spaces. Consequently, we expect this optimisation method to perform badly in high dimensions, in particular.

5.2.2 The CMA-ES algorithm

CMA-ES belongs to the class of evolutionary algorithms, where in each step a set of candidate solutions is evolved, while the best solutions are kept. This process resembles natural selection, as the fittest candidate solutions are chosen to "survive" and new candidates are derived from these solutions. Introduced by Hansen and Ostermeier [53], CMA-ES is a particular strategy for evolving the set of candidate solutions, by sampling new candidate solutions from a multivariate Gaussian whose parameters are defined based on the best candidate solutions. The general algorithm is listed in Algorithm 5. For a detailed discussion of how the distributional parameters are evolved, we refer to Hansen [111].

5.2.3 The Nelder-Mead algorithm

The NM algorithm is a local search algorithm, that may be converted into a global optimisation algorithm by combining the results of multiple restarts from different locations [3]. In particular, using restarts aids to avoid local minima that the algorithm may converge to. The result is a robust algorithm for derivative-free optimisation that shows good performances in practise. The algorithm starts with a simplex, whose vertices are iteratively adapted, such that the function values at those points improve in each iteration. The full algorithm is listed in Algorithm 6.

Algorithm 6: The Nelder-Mead algorithm. Adapted from Gao and Han [113].

input : Objective fct f, init simplex Δ, Number of dim. D, time budget N, hyperparams $\alpha, \beta, \gamma, \delta$

1 **while** *not converged* $\wedge\ n < N$ **do**
2 Sort points in Δ: $\mathbf{x}^1, \ldots, \mathbf{x}^{D+1} \leftarrow \arg\operatorname{sort}_{f(\mathbf{x}^i)} \mathbf{x}^i$
3 Compute centroid: $\overline{\mathbf{x}} \leftarrow \texttt{centroid}(\mathbf{x}^1, \ldots, \mathbf{x}^D)$
4 Compute reflection point: $\mathbf{x}^r \leftarrow \overline{\mathbf{x}} + \alpha(\overline{\mathbf{x}} - \mathbf{x}^{D+1})$
5 $\mathbf{x}^{D+1} \leftarrow \mathbf{x}^r$ if $f(\mathbf{x}^1) \leq f(\mathbf{x}^r) < f(\mathbf{x}^D)$
6 // Expansion
7 **if** $f(\mathbf{x}^r) < f(\mathbf{x}^1)$ **then**
8 Compute expansion point: $x_e \leftarrow \overline{x} + \beta(x_r - \overline{x})$
9 $\mathbf{x}^{D+1} \leftarrow \arg\min_{\{\mathbf{x}^e, \mathbf{x}^r\}} f(\mathbf{x})$
10 // Contraction
11 **if** $f(\mathbf{x}^D) \leq f(\mathbf{x}^r) < f(\mathbf{x}^{D+1})$ **then**
12 Compute outside contraction point $\mathbf{x}^o \leftarrow \overline{\mathbf{x}} + \gamma(\mathbf{x}^r - \overline{\mathbf{x}})$
13 $\mathbf{x}^{D+1} \leftarrow \mathbf{x}^o$ if $f(\mathbf{x}^o) \leq f(\mathbf{x}^r)$
14 **else if** $f(x^r) \geq f(x^{D+1})$ **then**
15 Compute inside contraction point $\mathbf{x}^i \leftarrow \overline{\mathbf{x}} - \gamma(\mathbf{x}^r - \overline{\mathbf{x}})$
16 $\mathbf{x}^{D+1} \leftarrow \mathbf{x}^i$ if $f(\mathbf{x}^i) \leq f(\mathbf{x}^r)$
17 Shrink: $\mathbf{x}^i \leftarrow \mathbf{x}^1 + \delta(\mathbf{x}^i - \mathbf{x}^1)$, for $i \in [2, \ldots, D+1]$
18 $n \leftarrow n + 1$
19 **return** $\arg\min_{\mathbf{x} \in \mathcal{D}} f(\mathbf{x})$

5.3 Implementation

We implement the sampling-based optimisation by sampling $20,000$ points from a multivariate uniform distribution that is bounded by the limits of the input space. For example, the G6 benchmark that we use in our evaluation (see Table 6.1) is defined over $\mathcal{X} = [13.5, 14.5] \times [0.5, 1.5]$, so $x_1 \sim U(13.5, 14.5), x_2 \sim U(0.5, 1.5)$.

For NM optimisation, we use the adaptive implementation of `scikit-learn`[2], which is based on Gao and Han [113]. For all hyperparameters, in particular $\alpha, \beta, \gamma, \delta$, we use the default values of the implementation mentioned above. We use five restarts, which are initialised with the five best points among $20,000$ random function evaluations, i.e. the first run uses the best point, the second run uses the second best, etc. Given the initial point of a single run, the initial simplex δ is defined automatically by `scikit-learn`.

For CMA-ES, we use the `pycma` package[3]. Following the BoTorch tutorial on CMA-ES[4], we set $\sigma = 0.2$, and double the population size with respect to the aforementioned tutorial to 100 to guarantee a good performance in high dimensions. All other hyperparameters of the algorithm are chosen to be the default values of the `pycma` implementation. As initial value, we choose the best point among $20,000$ random function evaluations. Unfortunately, the `pycma` implementation does not support the optimisation over categorical dimensions, so the evaluation of this method will be limited to continuous domains.

For both NM and CMA-ES optimisation, the resulting optimum is compared to the best observed function value when performing random sampling optimisation and the better value is chosen. This step helps to escape local minima during the optimisation.

[2] `https://scikit-learn.org/`
[3] `https://github.com/CMA-ES/pycma`
[4] `https://botorch.org/tutorials/optimize_with_cmaes`

Chapter 6

Evaluation

In this chapter, we analyse the performance of our proposed approach. The goal of this section is to incrementally analyse the impact of the different contributions that we make. This means that we:

1. Outline the experimental setup of our experiments (Section 6.1),
2. Experimentally prove that tree ensembles can be used as surrogates for black-box constrained BO, and that GBRTs and MFs outperform RFs as such (Section 6.2),
3. Demonstrate how the distance-based uncertainty metric for Mondrian Forests, which we proposed in Chapter 4, improves the performance of MFs as BO surrogates (Section 6.3),
4. Demonstrate that using the Nelder-Mead optimisation algorithm improves the optimisation outcomes compared to other global optimisation methods(Section 6.4),
5. Show that Mondrian Forest surrogates with our novel uncertainty estimate outperform state-of-the-art GP surrogate models on several benchmark problems (Section 6.5).

6.1 General experimental setup

In each problem setting that we analyse, the constraint functions are unknown to the optimiser, and the feasible region is approximated using tree ensembles. We repeat each experiment with different random seeds for five iterations. As results, we report the mean and standard deviation across these runs[1]. All experiments are computed on a Linux machine with an Intel Core i7-7700K @ 4.20GHz CPU with 16 GB RAM. We do not use GPU support for any experiment. In Section 6.3, we report runtimes. For this, the wall-clock time is measured and averaged across five iterations.

While we vary the type of tree ensemble, uncertainty metric, and optimisation method across our experiments, the choice of acquisition function is not subject of our analysis, so we keep this parameter constant across all experiments, choosing the CWEI acquisition function (see Section 3.5 for a justification of this choice).

We compare the performances of the presented tree ensembles on a variety of constrained black-box optimisation tasks: the Welded Beam Design problem [25, 114, 115], the XGBoost hyperparameter tuning task by Daxberger *et al.* [18], and many synthetic benchmarks on continuous and mixed domains [14, 16, 25, 114, 116, 117, 118, 119]. This includes three own

[1]The seed values were generated randomly and are $[854203, 901350, 320477, 968248, 81922]$.

constrained benchmark problems, consisting of established objective functions with new constraint conditions, which are described in the Appendix. The benchmarks that we test our algorithm on are listed in Table 6.1. We report a total of 13 problems, of which 11 are synthetic, and 2 are real-world problems (Welded Beam and XGBoost tuning). A detailed mathematical description of the problems, as well as a visualisation of the 2D benchmarks, can be found in the Appendix. In Table 6.1, we list the number of initial points that are used in this problem setting. In all experiments, the evaluation of these initial points is included in the total number of iterations. For instance, if the optimisation is run for 50 iterations, and there are 8 initial points, the surrogate models are used to choose the next candidate for evaluation in the last 42 iterations, while the first 8 iterations evaluate the objective at the given initial points.

Problem	d_{cont}	d_{cat}	p	ρ	**Known optimum**	**Studied by**	**N. init. points**
Branin	2	0	1	69.8782%	0.3979	[14, 120, 121]	8
Rosenbrock	2	0	1	48.8489%	0.0	–	8
G6	2	0	2	1.1237%	−6961.8138	[114, 116, 122]	8
Gardner	2	0	1	1.6226%	0.2532	[16, 123]	8
Alpine (modified)	2	0	1	90.6292%	−1.0	–	8
Townsend	2	0	1	50.0139%	-3.2	[117]	8
Sphere	2	0	1	17.4385%	0.0	–	8
Welded Beam	4	0	5	37.4383%	2.3811	[25, 114, 115]	16
Ackley	20	0	2	0.0035%	0.0	[25, 124]	16
Keane Bump	30	0	2	99.9999%	*unknown*	[25, 125, 126]	16
Mixed Branin	2	2	1	3.8110%	−0.8143	[118]	8
Func-3C	2	3	1	2.6029%	−0.2315	[119, 127, 128]	8
XGBoost tuning	7	3	1	*unknown*	~ 0.0	[18, 127]	16

Table 6.1: Constrained optimisation benchmark problems. d_{cont} specifies the number of continuous input dimension, and d_{cat} the number of categorical input dimensions. The horizontal line in the table separates problems on continuous domains from problems on mixed domains. d specifies the number of constraints. To estimate the size of the feasible region(s), we uniformly sampled 10^6 points and evaluated their feasibility. The resulting relative size of the feasible region(s) is listed in the ρ column. The number of initial points refers to the number of points that all datasets are initialised with in the respective experiment.

6.2 Analysis of tree-based surrogate models for black-box constrained Bayesian Optimisation

Chapter 3 proposes using tree ensembles as surrogates for BO with unknown constraints. The purpose of this section is to empirically validate this approach by demonstrating that tree ensembles can be used for black-boxed constrained BO. Additionally, we intend to gain insight into the performance differences between different tree ensemble types. The tree ensembles that we consider in this analysis are:

1. Random Forests (RFs) [34] with prediction variance uncertainty
2. Gradient Boosted Regression Trees (GBRTs) [28] with Quantile Regression (QR) uncertainty
3. Mondrian Forests (MFs) [30, 62] with prediction variance uncertainty

Additionally, we report the performances of a random baseline, called DUMMY, which randomly selects a point from a uniform distribution bounded by the limits of the input space (this replaces the minimisation in line 4 of Algorithm 4 with sampling a point).

Based on the results of tree ensembles on regression tasks (Table 3.1), we expect GBRTs to perform well, since these models should be able to approximate the underlying constraint functions well. Following the literature, we also expect MFs to perform well, since they successfully solve unconstrained BO tasks [62], and their response surface is expected to be closer to that of a Gaussian Process (GP) than that of the other models [19]. In contrast, we hypothesise worse results for RFs, based on the results in previous works [19, 37].

The rest of this section is organised as follows: After a brief description of the setting of all experiments in this section, we first qualitatively assess the performance of the different surrogate types. Then, we analyse the performances quantitatively on continuous as well as categorical benchmarks. Finally, we summarise and discuss the implications of our results.

6.2.1 Additional experimental setup specifications

The purpose of this section is to compare the results of different tree ensembles. Hence, we keep the uncertainty metric for each respective tree ensemble type constant across all experiments in this section. The uncertainty metric that is used with each respective tree ensemble is listed above. Not varying the uncertainty across experiments helps to attribute performance differences to the type of tree ensemble rather than to the uncertainty metric. We selected the given uncertainty metrics, since they are the default uncertainty estimates in the implementations of the respective tree ensembles that we use (see Section 3.5 for more details on our implementation and Section 2.4.1 for a thorough discussion of the uncertainty metrics).

For the same reasons as we do not vary the uncertainty metrics in this section, we also keep the optimisation method constant across all experiments in this section (see Section 6.4 for an analysis of the impact of other optimisation techniques). To keep the runtime of the experiments as low as possible, all experiments in this section use the sampling-based optimisation procedure to minimise the acquisition function (see Chapter 5 for an extensive explanation of this method).

6.2.2 Qualitative performance of tree ensembles as BO surrogates

We first perform a qualitative analysis of the surrogate model performances by examining the approximated feasible regions and exploration behaviour across the models. To this end, we run 50 iterations of BO on the first three constrained optimisation problems in Table 6.1.

The Branin Hoo problem was one of the first problems that was used for BO with unknown constraints [14]. We, therefore, examine the performance of tree ensembles on this benchmark problem to gain insight whether tree-based surrogate models are able to learn the underlying feasible region of this problem as well. The G6 problem has a very small feasible region, so this benchmark tests how well the different tree ensembles are able to detect and model such a small region. Finally, the 2D Rosenbrock function with the given constraints has not been used in literature before. We created this benchmark problem, to analyse the model capacities to learn discontinuous, and non-linear constraint functions. For a mathematical definition of all problems, we refer to the Appendix.

The results in Figure 6.1 confirm our initial hypothesis, that MF surrogate models are able to learn the feasible region of constrained BO problems well. In particular, MF surrogates with prediction variance uncertainty estimates display a sensible uncertainty about the underlying feasible region in the light of prior function evaluations. Furthermore, we observe on the Branin

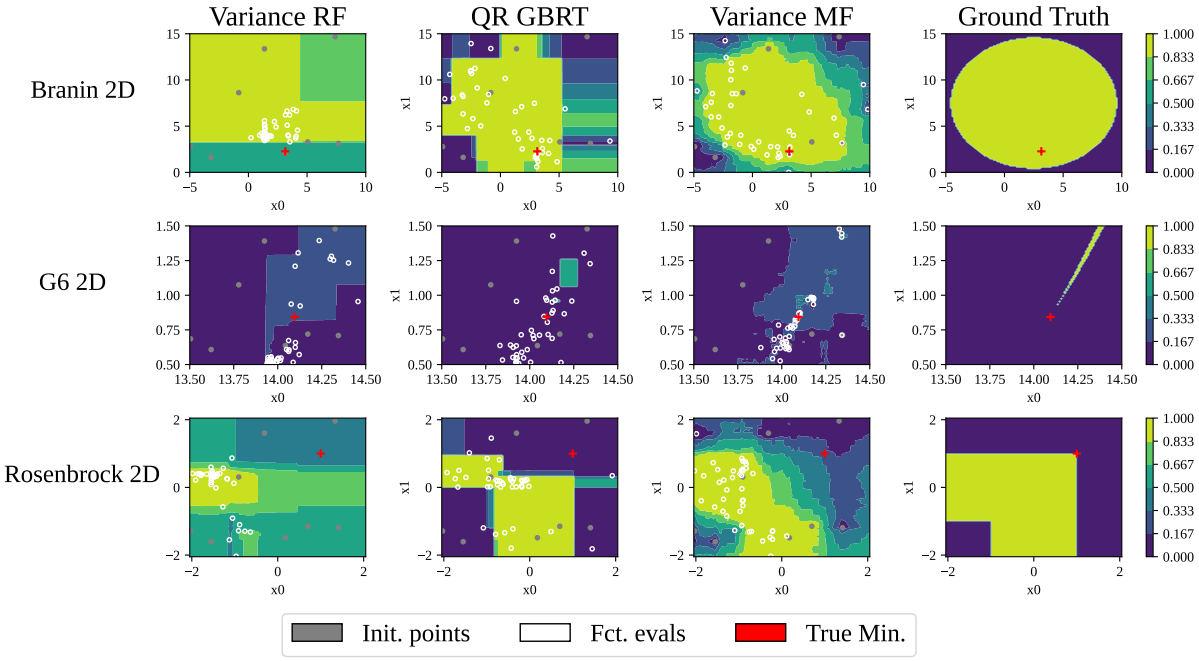


Figure 6.1: Learnt feasible regions of tree-based BO surrogates. We depict the learnt probability of constraint feasibility after constrained BO on the 2D Branin Hoo function [14], 2D G6 function [116], and 2D Rosenbrock [129] benchmarks. Results are after initialisation with 50 function evaluations (8 random initial points and 42 points selected by the optimiser).

Hoo and Rosenbrock problems, that MFs are able to approximate continuous and discontinuous feasible regions alike. This highlights the capacity of MFs as BO surrogates.

Similarly, GBRTs are able to learn feasible regions well enough to evaluate points, which lie mostly in the true feasible region. We can also observe that the approximated regions are compositions of axis-aligned boxes. This is a benefit on problems where the underlying feasible region has this property as well, e.g. the Rosenbrock 2D example. On the other hand, this may lead to worse extrapolation on other problems, as we observe in the case of the Branin Hoo problem, where the learnt feasible region is far from the ground truth for $x_0 \in [5, 10]$. Still, the approximated feasible regions are close enough to the ground truth to encourage exploration near the true function minimum.

As hypothesised, RF surrogate models perform poorly. On each problem, the learnt feasible region is dissimilar to the ground truth. Additionally, the function evaluations made by these surrogate models are confidently exploring regions far from the actual function minimum. While MF and GBRT ensembles mostly explore in a sensible region around the true minimum, RFs explore only small regions that are not including the true optimum.

6.2.3 Quantitative performance of tree ensembles as BO surrogates

Performance in continuous domains Figure 6.2 shows the per iteration best objective found by different tree ensemble types on multiple benchmarks (see Table 6.1 and Appendix for details on the problem settings). The first row of Figure 6.2 contains problems in continuous domains. While the Branin problem is easy to solve, but popular in literature, the G6 problem is challenging due to the small feasible region, and the Keane Bump problem is difficult to optimise due to its high number of input dimensions and multimodality.

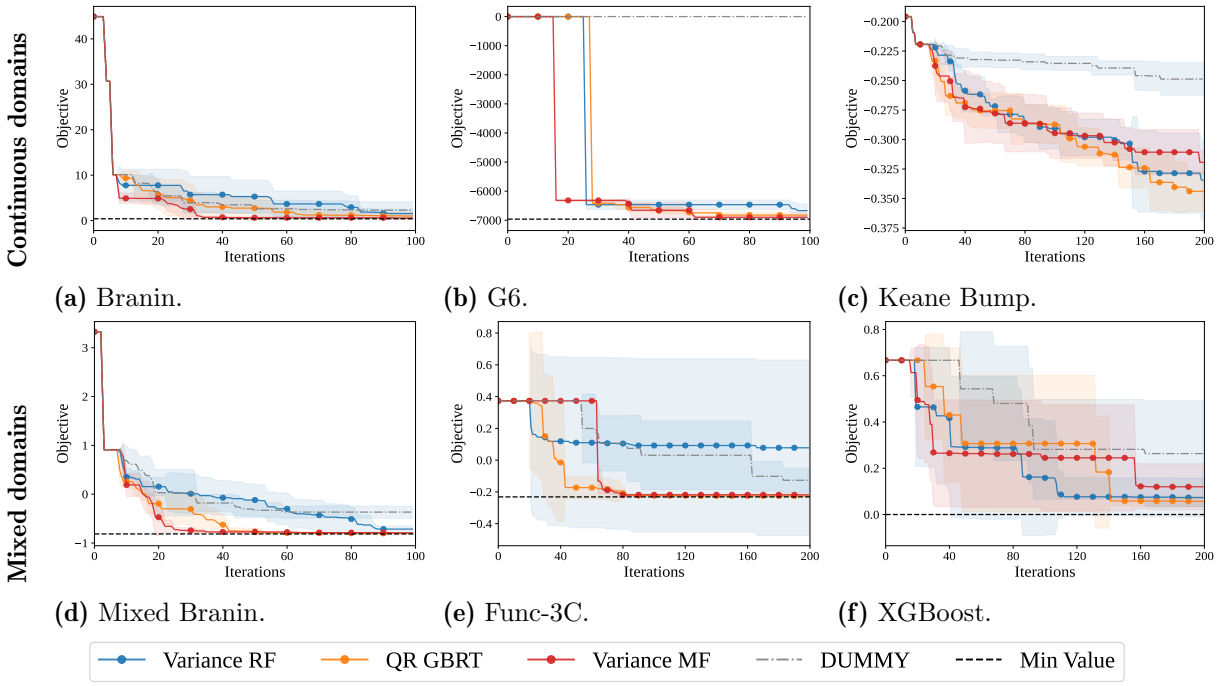
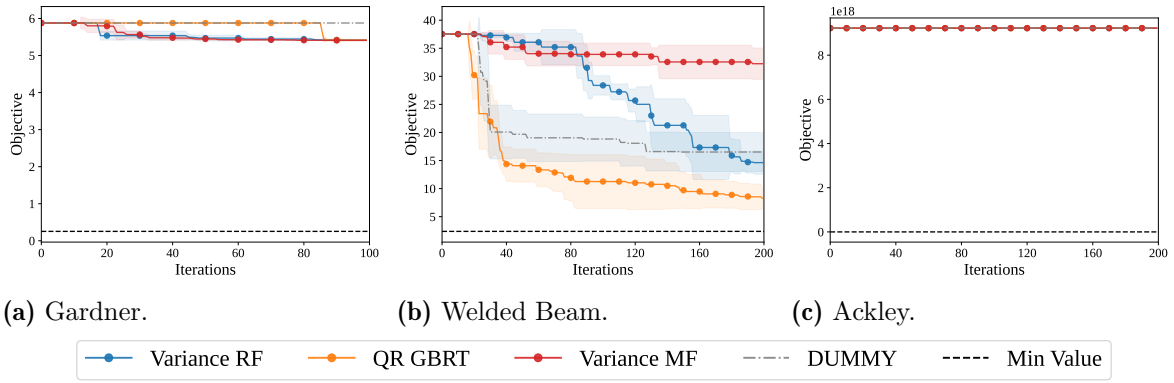


Figure 6.2: Tree surrogate performances on continuous and mixed domains. The first row contains problems on strictly continuous domains, while the second row presents the performance on mixed domains. Depicted are the minimum feasible values obtained per iteration. In case no feasible value was found yet at a given iteration, the maximum feasible value across all model is depicted. In all experiments, the CWEI acquisition function is minimised using sampling-based optimisation. For all models, we indicate the tree ensemble type $\in$ {RF, GBRT, MF} and uncertainty metric $\in$ {QR, Variance}. DUMMY refers to a baseline of randomly sampling the next candidate point.

The results in Figure 6.2 demonstrate that tree ensembles can effectively be used to solve BO problems with unknown constraints in continuous domains. All tree ensembles outperform the random baseline (DUMMY) on the continuous benchmarks. We observe that GBRTs outperform RFs on all continuous benchmarks. Similarly, on all but one of the continuous problems (Keane Bump), MFs outperform RFs as surrogate models, converging faster or to lower function values. As hypothesised, RFs perform the worst out of all tree ensemble types on the low-dimensional benchmarks (Figure 6.2a and Figure 6.2b), being the slowest to minimise the objective function in these settings, and not converging to the true minimum on either function (the lowest observed value on the Branin function is 1.54 while the minimum is at 0.4). Surprisingly, RFs perform better than MFs on the high-dimensional Keane Bump function. We hypothesise that this result may be due in part to the variance uncertainty metric, leading to weak results for MFs.

In Figure 6.5, we present another set of benchmark problems, which we observed to be difficult for tree-based surrogate models. No surrogate model performs better than random search on the Gardner and Ackley problems (Figure 6.5b and Figure 6.5f). In particular, no surrogate model is able to find the feasible region on the Ackley problem. The performance on the Welded Beam problem shows that tree-based surrogate models can perform better than random search, but in particular MFs and RFs struggle to perform well on this problem, while GBRTs perform better than random search.

Performance in mixed domains The second row of Figure 6.2 shows the per iteration best objective found by different types of tree ensembles on the mixed domain benchmarks from



(a) Gardner. **(b)** Welded Beam. **(c)** Ackley.

Figure 6.3: Difficult problems for tree-ensemble surrogates. Depicted are the minimum feasible values obtained per iteration. In case no feasible value was found yet at a given iteration, the maximum feasible value across all model is depicted. In all experiments, the CWEI acquisition function is minimised using sampling-based optimisation. For all models, we indicate the tree ensemble type $\in$ {RF, GBRT, MF} and uncertainty metric $\in$ {QR, Variance}. DUMMY refers to a baseline of randomly sampling the next candidate point.

Table 6.1. Our results demonstrate that tree ensembles can be used to solve BO problems with unknown constraints in mixed input spaces that encompass continuous and categorical dimensions. We observe that GBRTs and MFs perform well across all benchmarks. While GBRTs converge faster than MFs on the Func-3C problem (Figure 6.2e), the converse holds on the two other benchmark problems. Again, our results show weaker performances for RF surrogate models compared to the other two ensemble types on tow of the three problems. In particular, they do perform worse than other tree ensembles on the Mixed Branin problem (Figure 6.2d), and do not converge at all on the Func-3C problem (Figure 6.2e).

6.2.4 Discussion and Summary

In this section, we investigated whether tree ensembles can learn the feasible regions of black-box constrained BO problems. The experimental results of this section demonstrate that:

1. **Tree ensembles can learn the feasible region of black-box constrained problems**: We could solve multiple established benchmark problems from continuous and mixed domains using tree ensemble surrogate models. In particular, our results show that tree ensembles perform better than random search even in high dimensions, and in scenarios with multiple constraints. This experimentally proves that the approach that we present in Chapter 3 is valid.

2. **GBRTs surrogates generally outperform RFs**: The results in Figure 6.5 and Figure 6.6 show that GBRTs perform better than or as well as RFs on all presented benchmark problems. We hereby extend the results of previous work, which reported a worse performance of RFs on *unconstrained* black-box optimisation [8, 19, 37] to *constrained* optimisation problems. Our qualitative results in Figure 6.1 suggest that a reason for the weaker performance of RFs may lie in the fact that these surrogate models tend to overconfidently explore non-ideal regions, where a lot of function evaluations are made despite the true minimum lying elsewhere. This result is in line with Nickson *et al.* [37], who were the first to report on the tendency of RF surrogate models to be overconfident in BO.

3. **MF surrogates perform better than RFs on some problems:** Despite the qualitative results in Figure 6.1 suggesting the superiority of MFs, there is no clear evidence for one ensemble being consistently stronger than the other. While MFs perform better on the Branin, G6, Mixed Branin, and Func-3C problems, RFs perform better on the Welded Beam and Keane Bump problems. Although this shows that MFs perform better than RFs more often than the opposite is the case, our results do not support a general statement suggesting the superiority of MFs. Still, our qualitative results make us believe that MFs may be the better choice overall, and that their weaker performance in the Welded Beam and Keane Bump problems may be overcome if the uncertainty metric is improved. This issue will be addressed in the next section.

4. **No clear winner between MFs and GBRTs:** We observe in Figure 6.1 that the response surface of GBRTs tends to be less close to the ground truth compared to MFs. However, GBRT surrogate models outperform MFs on several of the presented problems (e.g. Figure 6.2c and Figure 6.2e). Consequently, the above results do not present conclusive evidence whether GBRTs or MFs are to be preferred over one-another. Therefore, we will investigate this question more thoroughly in the next section, where we evaluate the performance of these two tree ensemble types using the same uncertainty metric.

5. **There is room for improvement**: Although we have proven that tree ensembles can be used to solve constrained black-box optimisation problems, our results also highlight that there exist problems that are challenging for tree ensemble surrogates (see Figure 6.3). In particular, the Gardner and Ackley problems cannot be solved by any surrogate in the experiments presented (see Figure 6.3). We hypothesise that these problems may be too difficult for the evaluated tree ensembles, due to the sampling-based optimisation that was used for all the above experiments, as well as the properties of the uncertainty metrics used. MFs struggle on the Gardner, Keane Bump and Welded Beam problems (Figure 6.3a, Figure 6.2c and Figure 6.3b), which have large feasible regions and several local optima. We hypothesise that MFs with prediction variance uncertainty may be underexploring the input domain, in particular in the presence of the multiple feasible regions (Gardner problem), or many local optima (Keane Bump problem). Since the distance-based uncertainty metric that we presented in Chapter 4 ignores the estimated function value, we hypothesise that this uncertainty estimate may help address these issues. We investigate this claim in the next section.

6.3 Analysis of novel uncertainty metrics for tree ensembles

In Chapter 4, we present the novel distance-based uncertainty for MF BO surrogate models. We hypothesise that this uncertainty estimate for MFs will improve the performance of these surrogates, as it may reduce effects of under- and overconfidence of the traditional prediction variance uncertainty for MFs. Second, we proposed to use the MC Dropout uncertainty estimate for GBRTs in the context of BO for the first time. We hypothesise that this uncertainty estimate will *not* improve the performance of GBRT surrogates, because the sub-ensembles that are used in the MC approximation are highly correlated. This correlation will lead to small uncertainty intervals that discourage necessary exploration (see Chapter 4 for more details on this). To test our hypotheses, we compare the presented uncertainty metrics, to other, well-established uncertainty estimates, which are listed in Table 6.2. Again, we include the DUMMY random baseline from Section 6.2 in our experiments as a reference. Like in the previous section, all experiments in this section use the sampling-based optimisation strategy to minimise the CWEI acquisition

function. This section is structured as follows: we first analyse the impact of the different

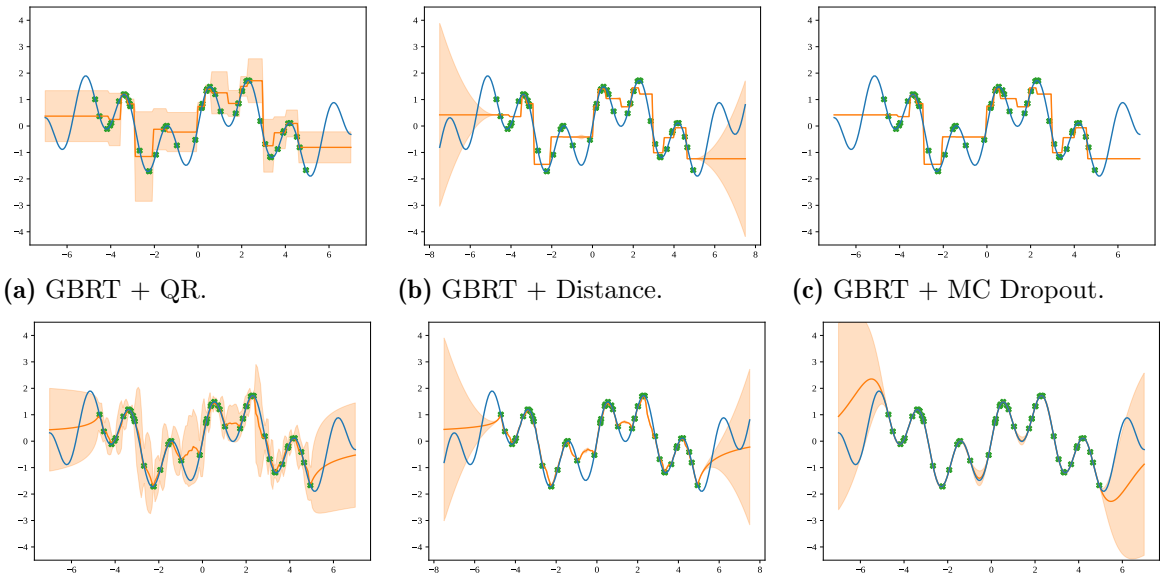
Gradient Boosted Regression Trees (GBRT)	Mondrian Forests (MFs)
MC Dropout [38] (new for BO)	Distance-based (ours)
Prediction variance [30, 62, 78]	Quantile regression (QR) [95]
Distance-based [32]	

Table 6.2: Analysed uncertainty metrics. The distance-based uncertainty for MFs and MC Dropout uncertainty for GBRTs have been introduced in this work in Chapter 4. For background on all other uncertainty estimates, we refer to Section 2.4.2.

uncertainty estimates on the predictive distribution of the surrogate models, and compare it to the predictive distribution of a GP. Subsequently, we perform a quantitative analysis of the different uncertainty metrics and evaluate the outcomes of BO on several benchmarks. Then, we analyse the runtimes of the uncertainty estimation techniques. Finally, we summarise and discuss our findings.

6.3.1 Impact of uncertainty metrics on predictive distributions

Figure 6.4 displays the learnt objective function after fitting GBRTs and MFs models to 50 training data points. As hypothesised, we observe that the MC Dropout uncertainty for the GBRT ensemble leads to very small uncertainty intervals. In particular, we observe that the uncertainty does not increase in regions far from the training data. In contrast, the distance-based uncertainty estimate produces narrow uncertainty intervals in regions where observations have been made and increasingly wide intervals in unexplored regions. Thus, the predictive



(a) GBRT + QR. **(b)** GBRT + Distance. **(c)** GBRT + MC Dropout.

(d) MF + Variance. **(e)** MF + Distance. **(f)** Gaussian Process.

Figure 6.4: Estimated predictive distribution for different tree ensembles. Estimated predictive distribution for different tree ensembles using different uncertainty metrics for the function $f(x) = \sin(x) + x \sin\left(\frac{10}{3}\right)$, which is depicted in blue. Green markers indicate training points. The learnt function and uncertainty intervals are plotted in orange. Displayed intervals are two times the estimated uncertainty interval.

distribution of the MF + Distance surrogate (Figure 6.4e) is closest to that of a GP (Figure 6.4f). In particular, the predicted uncertainty interval is fitting the data better than the variance-based uncertainty estimate for MFs, and the QR uncertainty estimate for GBRTs, which both produce too wide uncertainty intervals in regions that are covered by training data.

The qualitative results are supported in Table 6.3, where the Kullback-Leibler (KL) divergence of a GP to the predictive distribution of the different surrogates is listed. The results show that the MF + Distance ensemble performs the best or second best on all problems, producing the lowest KL divergences. This confirms the qualitative results in Figure 6.4. We observe that for both GBRTs and MFs, the distance-based uncertainty estimate yields the best results. Additionally, we observe that on each benchmark, MFs outperform GBRTs, which follows the trend we observe in Figure 6.1. Although the MC Dropout uncertainty estimate leads to an improvement over the QR uncertainty estimate, it consistently performs worse than the distance-based uncertainty estimate and any MF ensemble.

Method \ $f(x)$	$\frac{x^3}{100} + \epsilon$	$\sin(x) + x \sin\left(\frac{10}{3}\right)$	$\frac{1}{x}\sqrt{\frac{x}{4}} + \sin(x)$
GBRT + Quantile Regression	$\underline{00.01 \pm 00.01}$	432.20 ± 2029.78	77.70 ± 623.98
GBRT + Distance	$\underline{00.01 \pm 00.01}$	52.64 ± 232.02	$\underline{01.50 \pm 05.63}$
GBRT + MC Dropout	00.04 ± 00.02	73.30 ± 330.64	11.44 ± 55.06
MF + Variance	$\mathbf{00.00 \pm 00.00}$	$\underline{6.22 \pm 17.15}$	4.38 ± 15.60
MF + Distance	$\underline{00.01 \pm 00.01}$	$\mathbf{0.68 \pm 01.41}$	$\mathbf{0.09 \pm 00.12}$

Table 6.3: Kullback–Leibler divergences of tree ensembles to Gaussian Process. Kullback–Leibler divergence of a GP to the results by tree ensembles with different uncertainty measures. The horizontal line separates MF- and GBRT-ensembles. The best results per problem are **bold**, the second best are underlined. All runs are repeated 100 times. In the first column, we sample $\epsilon \sim \mathcal{N}(0.2, 1)$.

6.3.2 Comparison of uncertainty metrics on constrained BO problems

Figure 6.5 displays the performances of GBRT and MF surrogates for different uncertainty metrics being used. We observe that on all benchmarks reported problems, that can be solved by MFs, the distance-based uncertainty is superior to the variance-based uncertainty. The only exception is the Ackley problem, which is too difficult to be solved by any surrogate. These results demonstrate that the distance-based uncertainty for MFs that we present improves the performance of MFs as surrogates on high and low-dimensional, synthetic, and real-world benchmarks with unknown constraints. In particular, the distance-based uncertainty enables to solve the previously challenging Gardner problem (Figure 6.5b), and improves the performance of MF surrogates considerably on the Keane Bump and Welded Beam problems.

For GBRTs surrogates, we observe that the distance-based uncertainty is producing good results on all benchmark problems presented as well, except for the Ackley problem. The MC Dropout uncertainty estimate produces better outcomes than the QR uncertainty on the Welded Beam problem, but performs worse on the Keane Bump problem. Given the same uncertainty (distance-based uncertainty), we observe only negligible differences between GBRT and MF ensembles.

As stated above, the Ackley problem cannot be solved by any surrogate model. This shows that there exist problems that are generally difficult for tree ensembles. We hypothesise that this benchmark exposes the weaknesses of the sampling-based optimiser, since it is defined on $\mathbb{R}^{20}$, and has a very small feasible region (0.0035%), which is difficult to detect. We address this issue

in the next section, which investigates the effects of using non-sampling-based optimisations strategies to minimise the acquisition function.

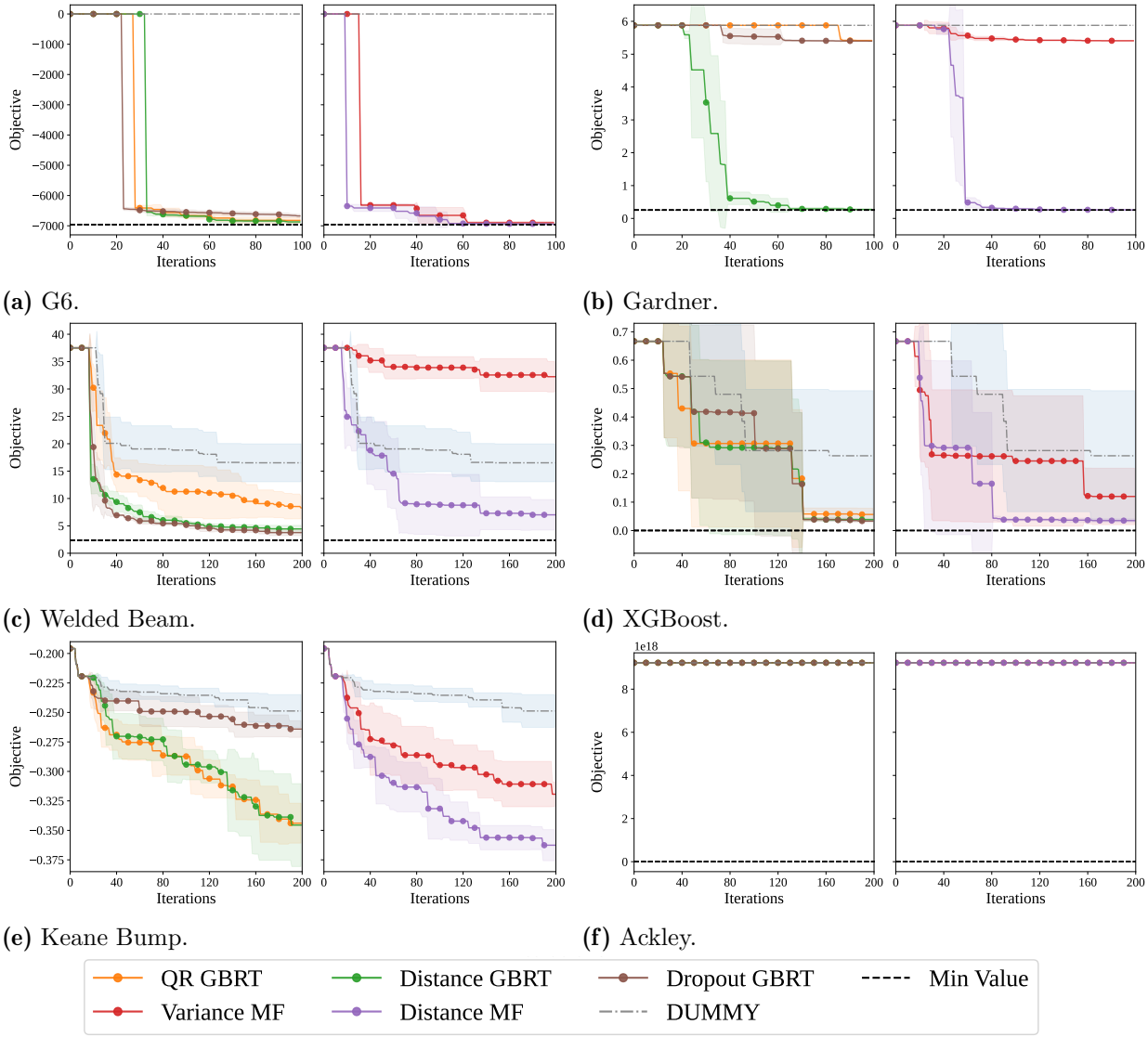


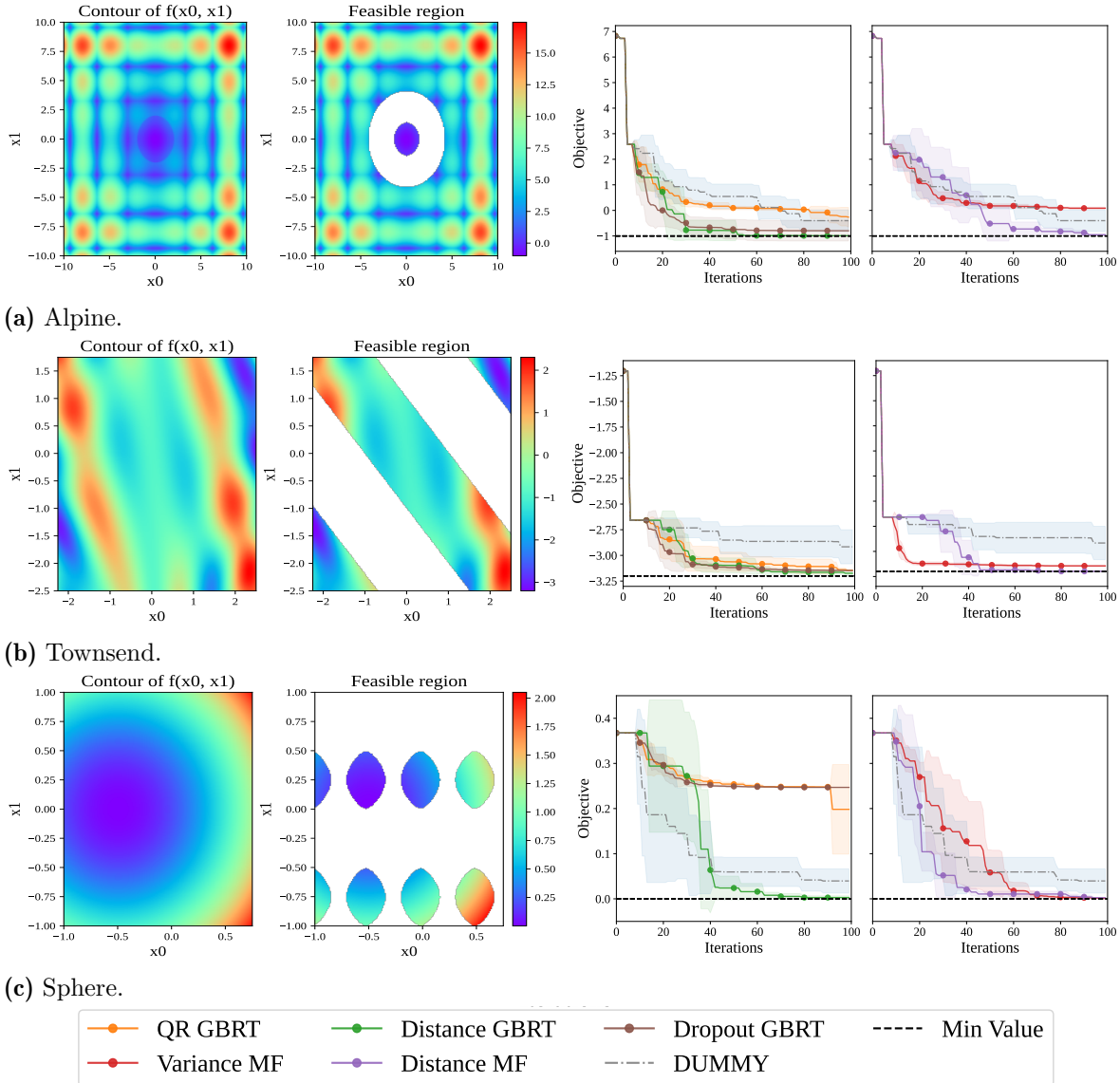
Figure 6.5: Comparison of uncertainty metrics. Comparison of different uncertainty metrics for tree-ensemble-based surrogates for BO with black-box constraints. Depicted are the minimum feasible values obtained per iteration. If no feasible value was found yet at a given iteration, the maximum feasible value across all model is depicted. In all experiments, the CWEI acquisition function is minimised using sampling-based optimisation. For all models, we indicate the tree ensemble type $\in$ {GBRT, MF} and uncertainty metric $\in$ {QR, Distance, Variance, MC Dropout}. DUMMY refers to a baseline of randomly sampling the next candidate point.

6.3.2.1 Investigating why the distance-based uncertainty performs well

The results in Figure 6.5 show that distance-based uncertainty estimation improves the performance of MF surrogates. On problems such as the Gardner function, the difference from other uncertainty estimates is large for both GBRTs, and MFs. This subsection has the purpose to empirically investigate why and when the distance-based uncertainty performs better than other

uncertainties.

The Gardner problem is difficult as it contains multiple disjoint feasible regions, which requires escaping the non-optimal one if it is encountered, much like a local optimum in unconstrained optimisation (see Appendix for a visualisation of the feasible region). Similarly, the Keane Bump objective function and Welded Beam design problem possess multiple local optima that need to be avoided or escaped during optimisation. Thus, we hypothesise that the distance-based uncertainty estimate helps to guarantee sufficient exploration to find global optima in the presence of local optima due to a multimodal objective or specific constraint boundaries.



(a) Alpine.

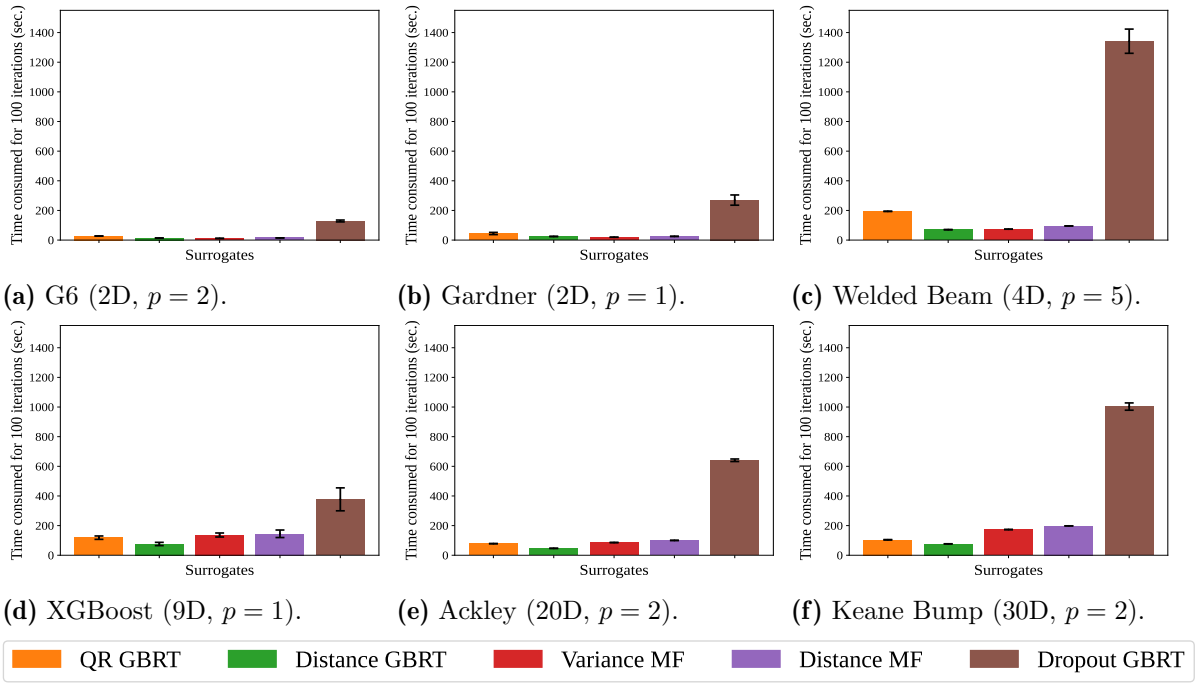
(b) Townsend.

(c) Sphere.

Figure 6.6: Analysis of tree-based surrogate models on functions with multiple local optima. Comparison of different uncertainty metrics for tree-ensemble-based surrogates for BO with black-box constraints. Depicted are the minimum feasible values obtained per iteration. If no feasible value was found at a given iteration, the maximum feasible value across all models is depicted. In all experiments, the CWEI acquisition function is minimised using sampling-based optimisation. For all models, we indicate the tree ensemble type $\in$ {GBRT, MF} and uncertainty metric $\in$ {QR, Distance, Variance, MC Dropout}. DUMMY refers to a baseline of randomly sampling the next candidate point.

To test this hypothesis, we evaluate different uncertainty metrics on problems with the given characteristics: the constrained Townsend function [117], a modified version of the Alpine function (own), and a Sphere problem (own). All these problems were designed to include multiple disjoint feasible regions and vary in the degree of difficulty of the objective function, e.g., Alpine being more difficult than the Sphere function. The resulting problems are displayed in Figure 6.6, and the mathematical problem definitions can be found in the Appendix.

The results in Figure 6.6 confirm our hypothesis. We observe that for all three problems, the distance-based uncertainty surrogates find the true minima, while the usage of other uncertainties leads to slower or no convergence to the true optimum. Although the convergence of the Distance MF surrogate is initially slower than that of the Variance MF surrogate, the lowest observed value on all functions is lower when using the distance-based uncertainty. In particular, the variance-based uncertainty surrogate only finds local optima on all three problems. For example, while the Variance MF surrogate converges to a local optimum of the Townsend function of $f(\mathbf{x}) \approx -3.15$ (at the top right of the plotted domain), the Distance MF surrogate finds the true minimum of $f(\mathbf{x}) = -3.2$ at $\mathbf{x} = (-2.25, -1.2964)$. On the Alpine and Sphere problems, the observed performance differences are even greater, with the Distance MF surrogate outperforming the Variance MF surrogate on both problems. Similarly, for GBRTs, we observe that the distance-based uncertainty produces the best results of all uncertainty metrics compared. Thus, the results show that the distance-based uncertainty improves BO on multimodal objective functions and problems with multiple feasible regions.



(a) G6 (2D, $p = 2$). **(b)** Gardner (2D, $p = 1$). **(c)** Welded Beam (4D, $p = 5$).

(d) XGBoost (9D, $p = 1$). **(e)** Ackley (20D, $p = 2$). **(f)** Keane Bump (30D, $p = 2$).

Figure 6.7: Runtime comparison of tree-based surrogate models. Plotted are the average wall-clock times in seconds for different surrogates and uncertainty metrics. For each benchmark, we report the runtime for 100 iterations of BO with sampling-based minimization of the acquisition function. On display are the mean and standard deviation, averaged across five repetitions. All functions are sorted by number of input dimensions in ascending order. p refers to the number of constraints.

6.3.3 Runtime analysis

Figure 6.7 shows the average wall-clock times of running 100 iterations of BO on several benchmark functions. We observe that the MC Dropout uncertainty estimation consistently takes more than twice the time than all other uncertainty estimates do. Furthermore, we observe that the distance-based uncertainty is the fastest uncertainty for GBRTs. We believe that the QR uncertainty is slower than the distance-based uncertainty for GBRTs since quantile regression requires the training of a separate tree ensemble for each quantile, which represents a considerable overhead compared to the distance-based uncertainty. For MFs, the prediction variance uncertainty is quicker than the distance-based uncertainty for MFs. The variance-based uncertainty of MFs requires no extra computation during prediction, which makes it faster than the distance-based uncertainty. Overall, the speed differences between GBRT and MFs are only small given the same uncertainty (distance-based), with GBRTs being slightly faster, although the gap seems to slowly increase with the number of dimensions.

We furthermore observe that all runtimes increase notably with the number of dimensions and number of constraints of the problems. While the optimisation of the 2D problems (Gardner and G6) is very fast for all uncertainty estimates, the optimisation times are the highest on the Welded Beam, XGBoost, and Keane Bump problems. We believe that solving the XGBoost problem requires more time, since each function evaluation includes training an XGBoost ensemble for each function evaluation. For the Welded Beam problem, we believe that the longer optimisation time can be attributed to the number of constraints in this problem, as each constraint is learnt by a separate tree ensemble. Lastly, we hypothesise that the longer optimisation time on the Keane Bump problem is due to the increased problem dimensionality, as we can see a steady increase in prediction time for problems with two constraints from G6 (2D) over Ackley (20D) to Keane Bump (30D).

6.3.4 Discussion and Summary

In this section, we investigated the performance of the novel uncertainty metrics that were introduced in Chapter 4. The experimental results of this section demonstrate that:

1. **The distance-based uncertainty for MFs improves the predictive performance:** The results in Figure 6.5 demonstrate that the distance-based uncertainty improves the performance of MFs as BO surrogates considerably compared to the classical prediction variance uncertainty. In particular, on problems such as the Gardner and Welded Beam function, we achieve improvements by an order of magnitude compared to the previously used prediction variance uncertainty. Our results also show that the predictive distribution of the surrogates is closer to that of a GP when using the distance-based uncertainty estimate (Figure 6.4). We hypothesise that this may explain the strong performance of the distance-based uncertainty for MFs. In Figure 6.6, we investigated why the novel uncertainty helps to improve the surrogate performance. Here, we demonstrated that our novel uncertainty estimate is superior to the predictive variance uncertainty, since it effectively avoids local minima during the optimisation. Lastly, our runtime comparison reveals that although the prediction variance uncertainty estimate is faster to compute, the difference is very small and, in our eyes, does not justify the usage of a worse performing uncertainty estimate.

2. **MC Dropout uncertainty for GBRT can be used for BO, but it is slow:** The second uncertainty estimate that we outlined in Chapter 4, is MC Dropout for GBRTs by Malinin *et al.* [38]. Since this uncertainty estimate has previously been unused in the context of BO, it was unclear how well it performs. Prior to our experimentation, we hypothesised that this

uncertainty estimate may be unsuitable to balance exploration and exploitation because the sub-ensembles that are used for the MC approximation are correlated. To our surprise, the MC Dropout GBRT surrogates perform better than the random baseline and quantile regression uncertainty on most problems. Nevertheless, the distance-based uncertainty estimate for GBRTs is superior or equal on all benchmark problems. In particular, problems that require exploration, such as the Gardner and Keane Bump benchmarks, could not be solved with a satisfying result when using the MC Dropout uncertainty estimate. These results are in line with the observations that we made in Table 6.3, where the KL divergence of a GP to GBRTs was consistently lower when using the distance-based uncertainty estimate compared to the MC Dropout uncertainty. While the MC Dropout uncertainty does perform similarly well or better than the QR uncertainty on most problems, our runtime analysis unveils that the MC Dropout uncertainty estimate is computationally demanding. On all problems, the MC estimation takes more than twice as long as other uncertainty estimate computations. The reason for this inefficiency lies in the fact that this uncertainty estimate relies on a sampling-based approximation of the predictive distribution, which requires computational effort. Consequently, we do not recommend the usage of the MC Dropout uncertainty for GBRTs surrogates, as other uncertainty estimates perform equally well or better while being much faster to compute.

3. **There are no large differences between MF and GBRT:** The introduction of the distance-based uncertainty for MFs allows one to compare GBRTs and MFs given the same uncertainty estimator. Our runtime analysis in Figure 6.7 unveils no large differences in training times between the two methods. Similarly, the results in Figure 6.5 and Figure 6.6 reveal similar predictive performances of the Distance MF and Distance GBRT surrogates. This result is surprising, considering the qualitative analysis of the predictive distribution in Figure 6.4, where the MF predictive distribution is much closer to the ground truth than that of GBRTs.

4. **There is (still) room for improvement:** At the end of the previous experimental section, we noted that the investigated surrogate models need improvements to be used for BO. This section demonstrates that the usage of novel uncertainty metrics constitutes such an improvement. In particular, the Gardner and Welded Beam problems become solvable for MFs when using the distance-based uncertainty estimate. Still, the Ackley problem remains unsolved by any surrogate. This shows that there remains room for improvement. The next section investigates if this improvement can be achieved by using a non-sampling-based optimisation strategy of the acquisition function.

6.4 Analysis of gradient-free acquisition optimisation strategies

In Chapter 3 we present sampling-based optimisation, the Nelder-Mead (NM) algorithm, and the CMA-ES algorithm as gradient-free optimisation strategies to minimise the acquisition function during BO. The purpose of this section is to compare these methods empirically. We hypothesise that the sampling-based optimisation strategy that was used in previous experimental sections may perform well in lower dimensions but has its limitations in higher dimensions, as the number of points that need to be sampled to cover the space increases exponentially with the number of dimensions. Consequently, we hypothesise that both, the NM optimisation strategy and the CMA-ES algorithm, perform better than the sampling-based optimisation strategy with the largest performance differences in high dimensions, i.e. $d \geq 10$. Additionally, we compare the performances of the NM optimisation strategy and the CMA-ES strategy. Since the

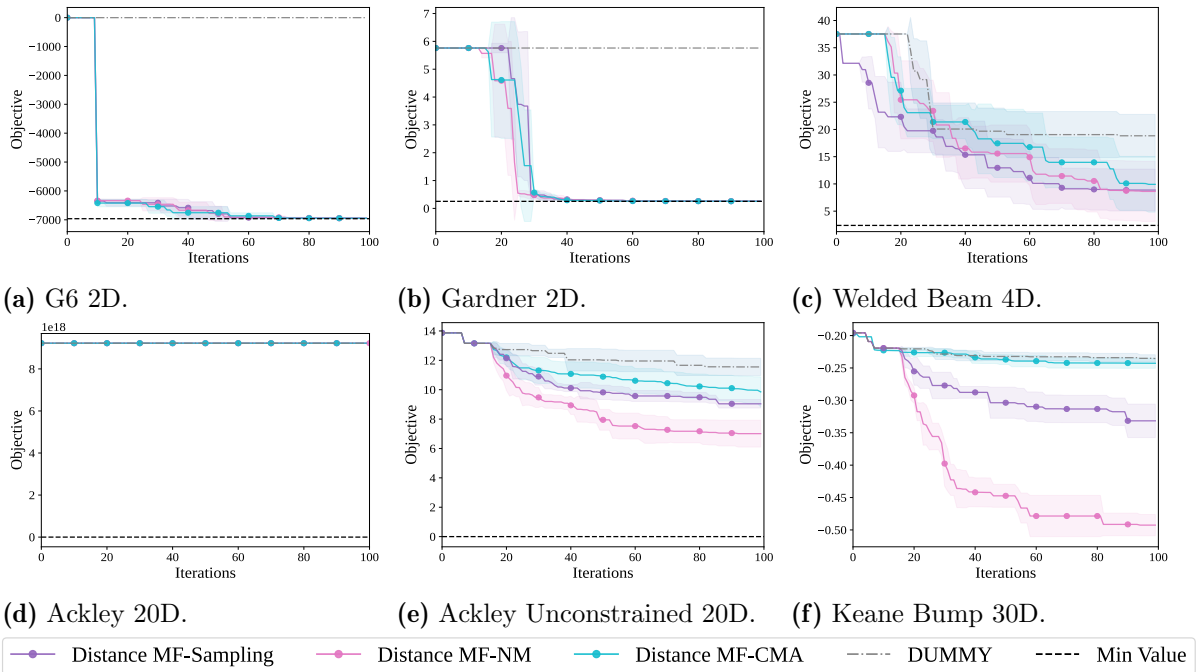
`pycma` implementation only allows continuous inputs, we only evaluate the NM algorithm on the benchmarks in mixed domains. We have no *a-priori* hypothesis which of the two methods will perform better.

Since the purpose of this section is to compare different strategies for acquisition function optimisation, we keep the tree ensemble and uncertainty type constant across all experiments in this section. Due to the strong performance in the evaluations above, we choose to evaluate all optimisation strategies when using MF surrogates with distance-based uncertainty. Again, we compare all runs against a random baseline, DUMMY, that randomly samples the next candidate point from the input space.

The remainder of this section is structured as follows: first, we analyse the performance of different optimisation strategies on problems on continuous domains. Then, we compare the performances on mixed domains. Finally, we summarise and discuss our findings.

6.4.1 Analysis of optimisation strategies on continuous domains

Figure 6.8 shows the performance of the three different optimisation strategies: sampling, Nelder Mead (NM), and evolutionary strategies (CMA-ES). As hypothesised, the NM optimisation strategy outperforms the sampling-based optimisation strategy in high dimensions. Furthermore, the results show that the sampling-based optimisation strategy performs well in lower dimensions, producing similar results to the NM algorithm in less than 20 dimensions. To our surprise, the CMA-ES algorithm performs the worst of all three methods. While it performs



(a) G6 2D. **(b)** Gardner 2D. **(c)** Welded Beam 4D.

(d) Ackley 20D. **(e)** Ackley Unconstrained 20D. **(f)** Keane Bump 30D.

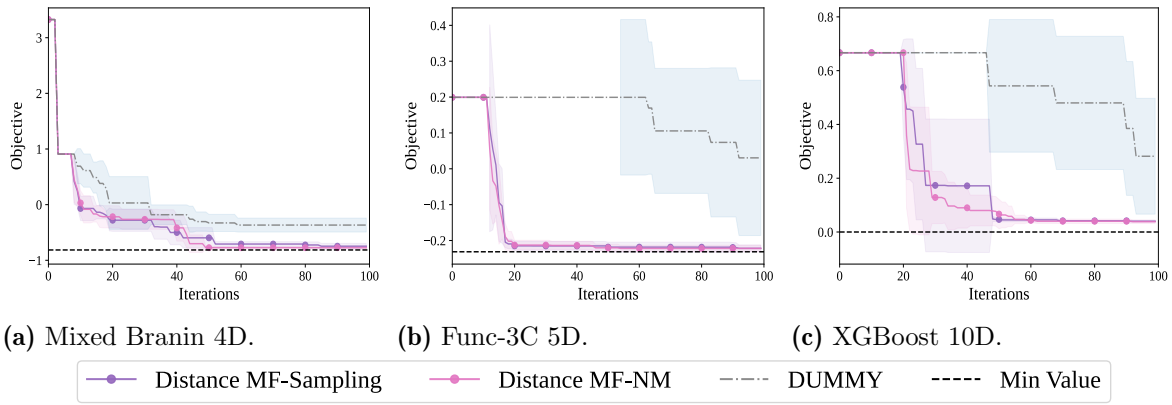
Figure 6.8: Comparison of acquisition function optimisation strategies. Comparison of different optimisation strategies to find the minimum of the acquisition function (see line 4 in Alg. 1). Depicted are the minimum feasible values obtained per iteration. In case no feasible value was found yet at a given iteration, the maximum feasible value across all model is depicted. We compare strategies $\in$ {Sampling, Nelder-Mead (NM), CMA-ES (CMA)}. DUMMY refers to a baseline that ignores the acquisition and randomly sampling the next candidate point from the input space. The Ackley Unconstrained problem is the same as the Ackley problem, but with constraints removed.

as well as sampling-based optimisation on the Keane Bump, G6, and Gardner functions, it is outperformed on the unconstrained Ackley function and Welded Beam problem. We notice that despite using non-sampling-based optimisation strategies, the 20D Ackley problem cannot be solved, as no surrogate is able to locate the feasible region. In order to test the optimisation strategies on a high-dimensional benchmark other than the Keane Bump function, we additionally report the performances on the Ackley function without the specified constraints. The results on the unconstrained Ackley function show that the NM algorithm improves the result of BO considerably in high dimensions, an effect we observe even more so on the 30D Keane Bump function.

6.4.2 Analysis of optimisation strategies on mixed domains

Figure 6.9 shows the performance of the Nelder Mead optimisation strategy and sampling-based optimisation strategy on the problems that are defined on mixed domains, which include continuous and categorical dimensions. We have to exclude the CMA-ES algorithm from this comparison, since the `pycma` implementation of the algorithm only supports continuous inputs.

The results on mixed domains show that the sampling-based optimisation and NM-based optimisation strategies perform equally well on problems in mixed domains. Both optimisation strategies solve all problems. For the XGBoost problem, note that the minimised objective is the cross-validation loss, so the dashed line at 0.0 is hard to reach in reality, which is why the results on this problem of around 0.04 are very good. These results show that tree ensembles generally can solve the selected problem well. All three problems are solved quickly, i.e. the best objective value is found no later than after 50 iterations.



(a) Mixed Branin 4D. **(b)** Func-3C 5D. **(c)** XGBoost 10D.

Figure 6.9: Effect of improved acquisition function optimisation in mixed domains. Comparison of different optimisation strategies to find the minimum of the acquisition function (see line 4 in Alg. 1). Depicted are the minimum feasible values obtained per iteration. In case no feasible value was found yet at a given iteration, the maximum feasible value across all model is depicted. We compare strategies $\in$ {Sampling, Nelder-Mead (NM)} for Mondrian Forest (MF) surrogate models. DUMMY refers to a baseline that ignores the acquisition and randomly samples the next candidate point.

6.4.3 Discussion and Summary

Shahriari *et al.* [8] identified the lack of reliable optimisation strategies for the acquisition function, as one of the main shortcomings when using tree ensemble surrogates for BO. In this section, we investigated the performance of the optimisation strategies that were discussed in Chapter 5. The experimental results of this section demonstrate that:

1. **The NM algorithm improves the performance in continuous domains:** As hypothesised, using a non-sampling-based optimisation strategy improves the result of the BO algorithm using tree ensemble surrogates. Specifically, the NM algorithm produces large improvements over the sampling-based optimisation strategy on high-dimensional continuous domains. This demonstrates that the NM algorithm can be used to successfully overcome the optimisation issue that Shahriari *et al.* [8] had pointed out.

2. **The NM algorithm and sampling perform equally well in mixed domains:** On benchmark problems in mixed domains, we do not observe any notable performance differences between sampling-based and NM-based optimisation. We hypothesise that this is due to the same reason as why there are no large differences between the two strategies on the G6, or Gardner problems: the sampling-based optimisation performs well in low dimensions. Since all benchmarks in mixed domains are defined in spaces with at most ten dimensions, we believe that the space can still be covered relatively well by sampling. Consequently, we hypothesise that a similar effect as for continuous benchmarks in higher dimensions may be observed, where the NM-based optimisation of the acquisition function achieved clearly better results than the sampling-based strategy.

3. **The CMA-ES algorithm fails to perform better than stochastic optimisation:** To our surprise, the third method for global optimisation that we tested, the CMA-ES algorithm, fails to improve the performance of tree ensemble surrogates. In particular, in higher dimensions, the performance of the CMA-ES algorithm is insufficient. We believe that a reason for this is that the CMA-ES algorithm typically requires a lot of function evaluation to perform well [25]. However, this number of evaluations is limited in high dimensions, as its computational complexity scales quadratically with the number of input dimensions [130].

4. **The Ackley problem cannot be solved:** In the previous section, we failed to solve the 20D Ackley problem and hypothesised that a reason for this may be the sampling-based optimisation strategy that we used in these experiments. In this section, we observe that non-sampling-based optimisation strategies fail to solve the problem as well. Since no method makes it possible to find a feasible point, we conclude that this is a weakness of the presented approach. When the feasible region is very small and no feasible point is known, it is very difficult to solve the optimisation problem.

6.5 Comparison of tree ensembles and Gaussian Processes as surrogate models for constrained BO

Chapter 3 of this work introduced tree ensembles as surrogate models for black-box constrained BO. We improved our method in Chapter 4, where we presented novel uncertainty estimates. Our experiments in Section 6.3, demonstrate that the distance-based uncertainty estimate for MFs produces strong results on various benchmark problems. We further improved our method in Chapter 5 by presenting the NM algorithm as a strategy to optimise the acquisition function. In Section 6.4, we demonstrated that the NM algorithm again improves the performance, in particular in high dimensions.

The motivation of our work was to improve existing approaches for black-box constrained BO, as they suffer from several shortcomings that are common to GP surrogate models. Thus, it remains an open question remains as to how well our novel approach compares to GPs as state-of-the-art surrogate models. To this end, this section compares the performance of Distance MF

surrogates with NM optimisation, and GP surrogates with ARD Matérn 5/2 kernel.

One well-known limitation of GPs is the deterioration of their performance in high dimensions [131]. Since MFs showed to perform well on the high-dimensional Keane Bump problem in previous sections, we hypothesise that MFs may be superior to GPs on high-dimensional problems. Additionally, we observed that Mondrian Forests using the distance-based uncertainty estimate perform particularly strong on problems that are multimodal and require a considerable amount of exploration. Consequently, we hypothesise that MFs may outperform GPs on benchmarks such as the Alpine and Unconstrained Ackley problems, as the gradient-based optimisation of GPs may get stuck in local optima. Meanwhile, no tree ensemble was able to solve the constrained 20D Ackley problem. Since a 200D version of this problem was solved with GPs by Eriksson *et al.* [131], we hypothesise that GPs will perform better than tree ensembles on this problem.

6.5.1 Additional experimental setup specifications

In this section, we compare tree ensembles against GP surrogate models. We implement all GPs using GPFlow [132] and use the `trieste` package[2] to implement the BO algorithm (Algorithm 4) for GPs. For each GP, we use the ARD Matérn 5/2 kernel. All hyperparameters are optimised using the L-BFGS-B algorithm [133], which is the default optimiser of `trieste`. More details of our GP implementation are discussed in the Appendix.

6.5.2 Results of experimental comparison between Mondrian Forests and Gaussian Processes

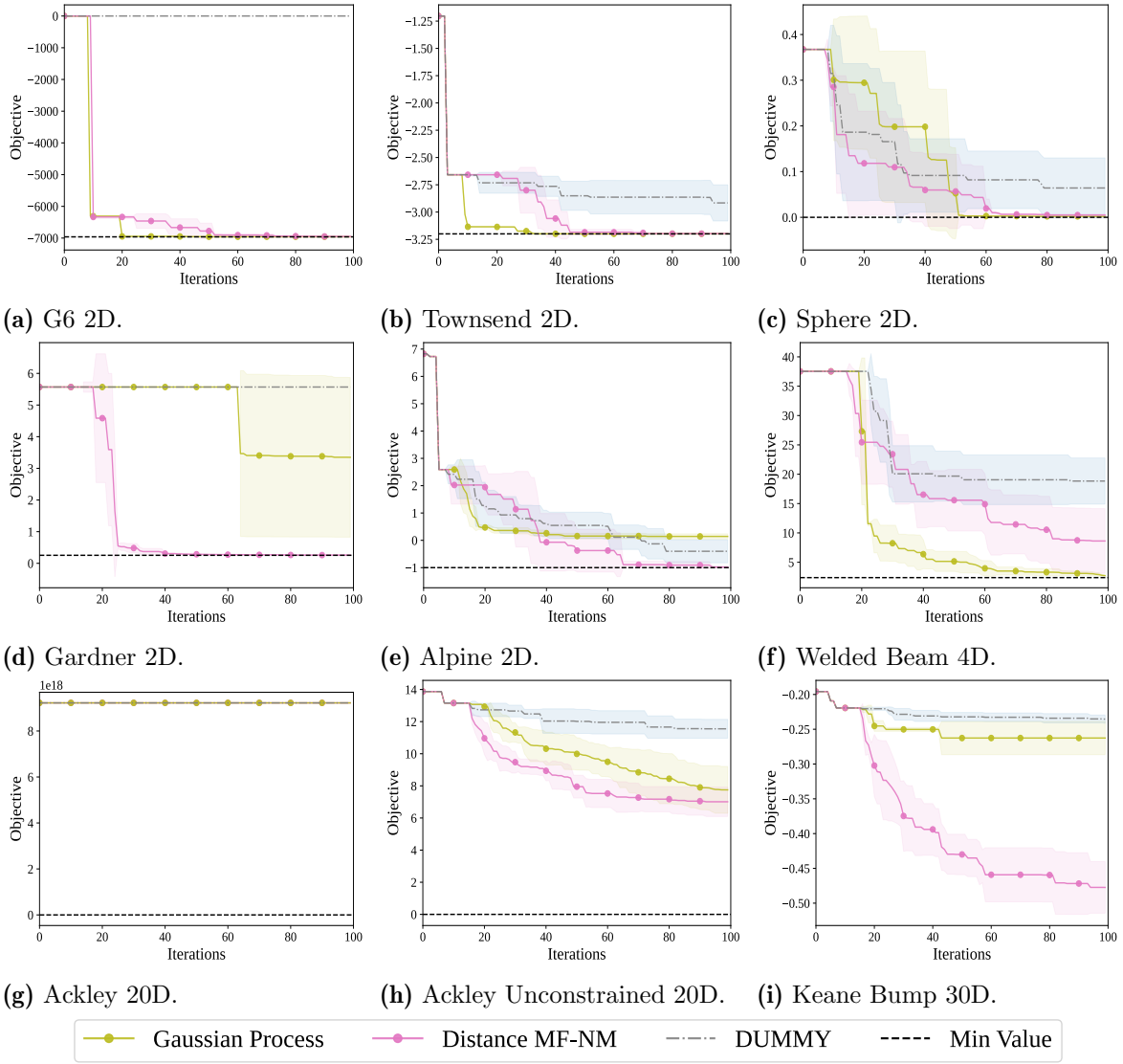
Figure 6.10 displays the performances of GPs and MFs on nine continuous benchmark problems (see Table 6.1 for more information). We observe that both MFs and GPs find the true minimum on the G6, Townsend, and Sphere problems. We also observe that GPs converge faster to the true optimum on all of these tree problems, although the difference is only small on the Sphere problem.

While GPs clearly outperform MFs on the Welded Beam problem, the Mondrian Forest surrogates outperform GPs on the Gardner, Alpine, Unconstrained Ackley, and Keane Bump problems. To our surprise, no surrogate model is capable of solving the constrained Ackley problem. We observe that these results exhibit the trend that MFs perform better than the given GP surrogates on problems in high dimensions. Additionally, these results confirm our initial hypothesis that MFs perform better than GPs on objective functions that are highly multimodal, as this characteristic is shared by the Alpine, Ackley, and Keane functions.

Figure 6.11 shows the performance of GPs and MF on three challenging benchmark problems on mixed domains (see Table 6.1 for more information). To our surprise, GPs converge faster on the XGBoost problem. However, at the same time, the performance of MFs is better on the other two problems in Figure 6.10. Specifically, GPs converge slower to the true optimum on the Func-3C problem, and fail to converge to the true minimum on the Mixed Branin problem. We hypothesise that these performance differences may be due to problem-specific properties of the objective function surfaces. In the case of the Func-3C and Mixed Branin problems, the optimal value of the continuous variables highly depends on the value of the categorical variables, i.e. whether increasing or decreasing the value of the continuous inputs minimises the objective, depends strongly on the value of the categorical variables. When varying values of the categorical dimensions of the XGBoost problem on the other hand, the optimal value of the continuous variables may not be affected as strongly by the value of the categorical variables.

[2] `https://secondmind-labs.github.io/trieste/0.12.0/index.html`

Of course, when changing the booster type from *gbtree* to *glinear*, the regularisation term may need adjustment, but a value of 0.0009 is likely to still be better than one of 109. Since GPs with Matérn kernel assume continuous objective functions, we hypothesise that the Mixed Branin and Func-3C objective functions may be harder to learn for GPs than the XGBoost objective, as we hypothesise that the XGBoost objective function may be smoother in most parts of the input domain.

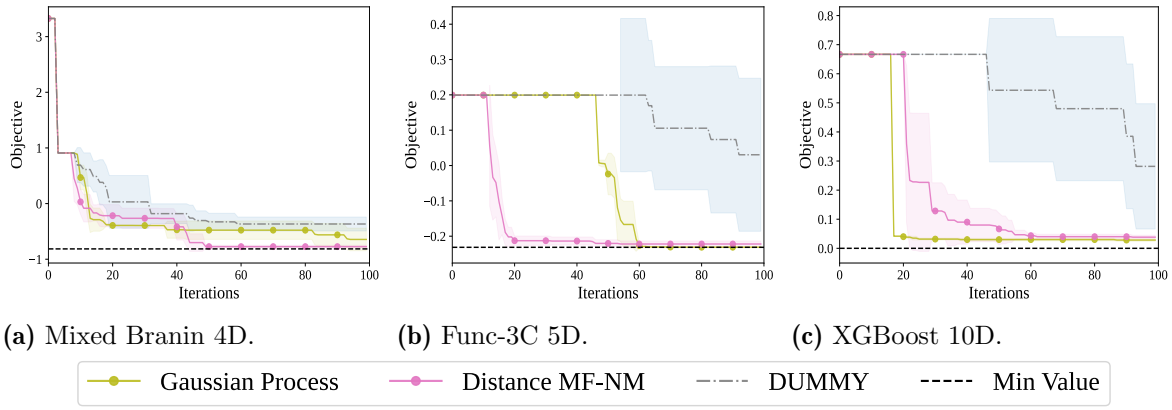


(a) G6 2D. **(b)** Townsend 2D. **(c)** Sphere 2D.

(d) Gardner 2D. **(e)** Alpine 2D. **(f)** Welded Beam 4D.

(g) Ackley 20D. **(h)** Ackley Unconstrained 20D. **(i)** Keane Bump 30D.

Figure 6.10: Comparison of tree ensembles and Gaussian Processes in continuous domains. Comparison of Gaussian Process (GP) and Mondrian Forest (MF) surrogate models. Depicted are the minimum feasible values obtained per iteration. In case no feasible value was found yet at a given iteration, the maximum feasible value across all model is depicted. DUMMY refers to a baseline that ignores the acquisition and randomly sampling the next candidate point from the input space. Note that for the MF surrogate, we use Nelder Mead acquisition function optimisation and distance-based uncertainty.



(a) Mixed Branin 4D. **(b)** Func-3C 5D. **(c)** XGBoost 10D.

Figure 6.11: Comparison of tree ensembles and Gaussian Processes in mixed domains. Comparison of Gaussian Process (GP) and Mondrian Forest (MF) surrogate models. Depicted are the minimum feasible values obtained per iteration. In case no feasible value was found yet at a given iteration, the maximum feasible value across all model is depicted. DUMMY refers to a baseline that ignores the acquisition and randomly sampling the next candidate point from the input space. Note that for the MF surrogate, we use Nelder Mead acquisition function optimisation and distance-based uncertainty.

6.5.3 Discussion and Summary

In this section, we compared the performance of the presented tree ensemble-based optimiser against GPs as state-of-the-art surrogate models for BO. Our results reveal the following:

1. **MFs outperform GPs on multimodal functions in continuous domains:** Our results unveil that the presented BO software outperforms a state-of-the-art GP-based Bayesian Optimiser (`trieste`) on multiple continuous benchmark problems from low and high dimensions. On four out of the nine presented problems, MFs perform better than GPs, while GPs clearly outperform MFs only on a single problem (Welded Beam Design). The benchmarks on which GPs struggle in this comparison mostly feature objective functions that are multimodal, which causes GPs to explore local optima and either converge later or not at all to the true optimum. MFs on the other hand, perform better on these problem instances. Additionally, we observe that the performance differences become larger in higher dimensions, as our results on the 30D Keane Bump and 20D unconstrained Ackley problem show. Thus, our results confirm our initial hypothesis that tree ensembles can help to address the issue of GPs struggling in high dimensions.

2. **MFs perform better than GPs on discontinuous objective functions:** Our results in Figure 6.10 unveil stronger performances of MFs on the Mixed Branin and Func-3C problems, while GPs converge quicker on the XGBoost problem. We believe that the reason for the weak performances of GPs on the two former problems lies in the fact that the objective functions of these problems are highly discontinuous. Since the given GPs with Matérn kernel assume continuous objective functions, this could explain their weaker performances on these problems. While being defined on a mixed domain too, we believe that the effect of varying a categorical variable may not be that strong on the optimal values of the other input variables in the case of the XGBoost problem, which could explain the better performance of GPs on this benchmark.

3. **GPs are more efficient on simple problems:** The relatively easy G6, Sphere, and Townsend problems, could be solved by both surrogate model types that we compared in this section.

However, on all three problems, GPs found the optimal solution faster than MFs. All of these problems have in common that the corresponding objective functions are relatively simple, and easy to learn. We believe that MFs converge slower because the presented scaling of the distance-based uncertainty makes MFs explore more than GPs. While this may be a downside on problems where the solution is easy to find, e.g. the aforementioned problems, this also enables MFs to solve problems with highly multimodal objective functions.

4. **The Ackley problem is too difficult for any surrogate:** Throughout this chapter, we attempted to solve the constrained Ackley optimisation problem and failed to do so. While our hypothesis was that this is due to insufficiently accurate uncertainty estimates, or poor optimisation strategies of the acquisition functions, we observed in this section that state-of-the-art GP surrogate models fail to solve this problem as well. This demonstrates that the inability of tree ensembles to solve this problem cannot be attributed to the shortcomings of tree ensembles as surrogate models, but rather to the inherent difficulty of the problem. Not only is the 20D Ackley problem high-dimensional, its feasible region is also very small relative to the size of the input domain ($\sim 0\%$). This optimisation problem thus exposes a general, unsolved, issue for BO, which is the detection of a feasible region without knowing a single feasible point.

Chapter 7

Conclusion

The present work began with the development of a fungicide that can be used on bananas as a practical scenario in which Bayesian Optimisation can be helpful. Motivated by this example, this project set out to improve the state-of-the-art in BO for black-box optimisation with unknown constraints by learning the objective and constraint functions using tree ensembles. This purpose of this section is to evaluate our achievements, the limitations of our approach, and to finally give an answer if we made a step toward saving the banana from the Panama disease.

This section is structured as follows: Section 7.1 provides a brief discussion of the most relevant ethical, legal, and social issues surrounding this work, Section 7.2 summarises the main achievements that have been presented in this work, Section 7.3 discusses some limitations of our approach, and finally, we provide a brief summary of the entire project.

7.1 Legal, social, ethical, and professional considerations

This section discusses the most relevant social, ethical, and legal issues related to the present project. The main issues that we identify are the potential misuse of the projects' results, ethical issues in datasets, and copyright implications.

This project focusses on leveraging tree-based machine learning methods for BO, which can be used to effectively find the best properties among many candidate solutions. Since almost any product property can be optimised with BO, the presented project results have the potential to be used in military applications, for instance in the development of armour materials [134]. Although this is a theoretical project, we do want to stress that we do see the objective of this project in creating an optimisation system for civilian applications *exclusively*. Still, the presented solver can be subject to misuse. This misuse could be of criminal use, for instance, in the development of illegal chemical substances such as recreational drugs.

In Table 3.1, we report the performance of machine learning models on several regression benchmark datasets from Hernández-Lobato and Adams [99]. In contrast to the latter work, we decided not to analyse model performances on the Boston Housing dataset [135], due to well-known racist bias in this dataset [136]. Similar racist bias has been reported for data from other domains [137, 138]. While the issue of the Boston Housing dataset is well-known and can thus be avoided by using alternative datasets, the optimiser that is presented in this thesis may be used on other data, that includes discriminating bias or features. Any practitioner that applies optimisation algorithms in real-world settings must be aware of this issue and guarantee optimisation solutions that do not systematically discriminate against any minority group. For a more thorough discussion of this issue, we refer to Knight *et al.* [139] and D'ignazio and Klein [140].

In the present project, we present a BO optimisation solver for research usage purposes. With this goal in mind, the resulting code is licenced under the BSD 3-Clause Licence. The project source code is based in parts on the openly available source code from the `scikit-optimize` [73], `scikit-garden`[1], and `entmoot` [32] packages for BO. All three packages are licenced under the BSD 3-Clause Licence[2], which permits redistribution, usage, and modification of the code. To the best of our knowledge, there is no copyright infringement or licencing conflict.

7.2 Achievements

The goal of this project was to improve Bayesian Optimisation surrogate modelling. In particular, we aimed to address issues in high dimensions and categorical inputs of existing surrogate models, i.e., Gaussian Processes in particular. To this end, we made the following contributions:

1. **We established tree ensemble surrogates for black-box constrained Bayesian Optimisation:** Our work is the first that performs Bayesian Optimisation with tree ensemble surrogates to solve black-box optimisation problems with unknown constraints. To this end, Chapter 3 adapted the results of previous work on constrained BO to tree ensemble surrogates. We validated this approach in Section 6.2, where we investigated the performance of different types of tree ensembles, and showed that Gradient Boosted Regression Trees and Mondrian Forests produce better performances compared to Random Forests, due to their superior exploration behaviour.

2. **We presented novel ways to estimate the uncertainty of tree ensembles:** We presented the distance-based uncertainty for MFs, and the MC Dropout uncertainty for GBRTs, which has previously been used for BOs. We proved experimentally that the novel distance-based uncertainty metric for Mondrian Forests helps improves the performance of MF surrogates by an order of magnitude. Furthermore, our work is the first to use the MC Dropout uncertainty estimate for GBRTs in the context of BO. Our results show that the MC Dropout uncertainty estimate for GBRTs can be used to solve black-box constrained optimisation problems, but its computational costs make it an unappealing choice for this task.

3. **We demonstrated that well-performing gradient-free optimisation strategies for BO exist:** Specifically, we showed the Nelder-Mead algorithm can be used to address the problem of the lack of gradients when using tree-based surrogate models. We show that the NM algorithm not only outperforms the sampling-based optimisation strategy, but also the CMA-ES strategy for global optimisation.

4. **We demonstrate that the resulting approach outperforms Gaussian Process-based Bayesian Optimisation on a multitude of benchmark problems:** Our results show that GPs may perform better on simple, small-scale problems, but that MFs performed better on high-dimensional benchmark problems. Additionally, our analysis unveiled that GPs struggle to find minima of discontinuous, multimodal objective functions, while MFs perform better on these problems.

In summary, our empirical results show that the resulting software is able to successfully find solutions to synthetic and real-life optimisation problems that are subject to unknown constraints.

[1] `https://github.com/scikit-garden/scikit-garden`
[2] `https://opensource.org/licenses/BSD-3-Clause`

Being the first work to use tree ensembles as surrogates for black-box constrained optimisation, we have improved on state-of-the-art in this area through presenting an alternative to the commonly used Gaussian Process surrogate models. In particular, we have shown that problems which may not be solved using Gaussian Process surrogates may be solved using Mondrian Forests as surrogate models, and optimising the acquisition function using the Nelder-Mead algorithm. Additionally, we have successfully established a novel uncertainty metric for MFs, which outperformed the existing uncertainty metric by an order of magnitude on many benchmark problems.

7.3 Limitations

Although we have successfully improved the performance of BO on black-box constrained problems, our work has several limitations:

1. **Finding a feasible point remains difficult:** Throughout this report, we cannot solve the constrained 20D Ackley problem. Although we initially believed this issue may be solved by improving the uncertainty metric and optimisation strategy, our results show that this is not the case. Even more so, Figure 6.10 shows that GPs fail to solve this problem as well. The fact that we could not find any surrogate model that solves this problem exposes a weakness of the presented approach: If no feasible point is known *a-priori* and the feasible region is very small, finding any feasible point is very difficult. This issue can be circumvented by initialising the optimiser with at least one feasible point [141, 142], but this approach may be inapplicable in scenarios where such a point is unknown. Other works have focused on improving the acquisition function that is optimised when no feasible point is known [143] with some success, but work in this direction is still necessary.

2. **Tree ensembles may extrapolate worse than Gaussian Processes:** Despite the practical success of the presented approach, the presented tree ensembles may not extrapolate as well as Gaussian Processes to unseen data: While GBRTs predict the same label as the closest training data point, predictions of MFs slowly converge towards the mean of the training data the further away inputs move from the training data. The latter produces a similar extrapolation as the Matérn kernel on many regression problems. However, there do exist special kernel functions for special problem classes, e.g. periodic kernels, which are likely to extrapolate better than any tree ensemble on periodic objective functions. On the other hand, the same kernel is likely to perform worse on other problems. Ultimately, this is a problem of choosing the right hyperparameters. We believe that MFs are strong surrogate models due to their flexibility, which allows them to perform well on many different problems.

3. **The complexity of the Nelder Mead algorithm scales with the dimensionality:** The complexity of a single iteration of the Nelder-Mead algorithm is in $\Theta(d^2)$ given d dimensions [144]. Although some efforts to improve the speed of the method exist [145, 146], we fear that our method's efficiency may deteriorate in high dimensions, i.e. $d \geq 100$. Thus, we strongly encourage future research to address this research by exploring alternative global optimisation methods that were not investigated in this work, e.g. particle swarm methods [147].

7.4 Future work

Being the first work to use tree ensembles as surrogates for black-box constrained optimisation, there are many opportunities for future work to further improve the existing approach:

1. **Extend comparisons:** In the present work, we compare MF surrogates against GPs as state-of-the-art surrogate models. While we use the versatile Matérn kernel in our implementations, there exist multiple other kernel functions that may perform better on some problems. For example, Cornford *et al.* [148] propose a kernel that is designed for discontinuous objective functions, and Thebelt *et al.* [149] propose a kernel that performs well on mixed domains. Similarly, several efforts have been made to improve the performance of GPs in high dimensions, e.g. Eriksson *et al.* [131]. It would be an interesting next step to compare the work we present in this thesis to the aforementioned approaches.

2. **Explore transformations of distance-based uncertainty estimates:** In this work, we present a distance-based uncertainty estimate for Mondrian Forests, which quantifies the distance to the closest point in the training data. We transform this distance using Min-Max-scaling to obtain estimates in a reasonable range. The chosen transformation is not the only sensible choice of transformation, and a thorough exploration of alternatives in the future would be desirable. For example, we observed that the uncertainty estimate of Gaussian Processes seems to grow logarithmically with the distance to the closest training point, while our present estimate grows quadratically. Hence, we propose to explore a log-transformation, such as $\mathbf{x}' := \log(1 + \mathbf{x})$ in future works. This transformation may help, because it will lead to larger uncertainty estimates closer to existing data, which may be appropriate in particular in high dimensions.

3. **Explore other uncertainty estimates:** The present work focused on two new uncertainty estimates: the MC Dropout uncertainty estimate for GBRTs had been previously unused in the context of BO, and the novel distance-based uncertainty estimate for MFs. Our approach was motivated by the strong performances of trees as point-predictors on regression problems, and the lack of reliable uncertainty estimates for these ensembles. Other works have used the predictive power of tree ensembles to construct GP kernels, which work better on categorical data than other kernels [85, 149, 150]. At the same, our comparison of GPs and trees shows that the predictive mean of tree ensembles may be more accurate than that of GPs. Thus, we believe that it may be interesting to investigate a surrogate model that reverses the aforementioned approach of using a tree-based kernel, and instead use the predictive mean of a tree ensemble, and the uncertainty of a GP. We believe that it may be possible to combine the strengths of both model types in this way, and achieve even stronger results.

7.5 Project summary

We began this work on Bayesian Optimisation, motivated by the endangerment of the banana by the Panama disease. To protect existing banana farms from the disease, a novel fungicide must be developed, which can be done much more effectively by formulating the design process as a black-box constrained optimisation problem that is solved using the Bayesian Optimisation algorithm. Due to the shortcomings of Gaussian Processes our work introduced tree ensembles as surrogate models for black-box constrained Bayesian Optimisation. Previously, Shahriari *et al.* [8] identified the lack of reliable uncertainty metrics and acquisition function optimisation

strategies as issues that limit the performance of tree ensembles as BO surrogate. We successfully addressed both issues in our work by introducing a novel uncertainty metric for Mondrian Forests that improves their performance by an order of magnitude, and by proving experimentally that the Nelder-Mead algorithm can be used to optimise the acquisition function during BO. Our experiments unveiled that the resulting approach outperforms Gaussian Processes on a multitude of benchmarks. In particular, problems in high dimension, that are characterised by a highly multimodal, or discontinuous, objective function may be solved better when using Mondrian Forest surrogate models with the presented distance-based uncertainty metric than when using GPs. We thus successfully improved the performance of BO surrogates, and in particular of tree ensembles, for Bayesian Optimisation on black-box constrained optimisation problems. Assuming that the efficacy of fungicide ingredients is a function with the specified characteristics, we thus made a step toward saving the banana from the Panama disease.

Bibliography

1. Ploetz RC. Fusarium wilt of banana. Phytopathology 2015; 105:1512–21
2. Food and Agriculture Organization of the United Nations. Banana facts and figures. 2022. Available from: `https://www.fao.org/economic/est/est-commodities/oilcrops/bananas/bananafacts/en/` [Accessed on: 2022 May 26]
3. Gelbart MA. Constrained Bayesian optimization and applications. Doctoral dissertation. Harvard University, Graduate School of Arts & Sciences, 2015. Available from: `https://dash.harvard.edu/handle/1/17467236`
4. Mockus J, Tiesis V, and Zilinskas A. The application of Bayesian methods for seeking the extremum. Towards global optimization 1978; 2:2
5. Lizotte DJ, Wang T, Bowling MH, Schuurmans D, *et al.* Automatic Gait Optimization With Gaussian Process Regression. *IJCAI*. Vol. 7. 2007 :944–9
6. Malkomes G, Schaff C, and Garnett R. Bayesian optimization for automated model selection. Advances in Neural Information Processing Systems 2016; 29
7. Li C, Rubin de Celis Leal D, Rana S, Gupta S, Sutti A, Greenhill S, Slezak T, Height M, and Venkatesh S. Rapid Bayesian optimisation for synthesis of short polymer fiber materials. Scientific reports 2017; 7:1–10
8. Shahriari B, Swersky K, Wang Z, Adams RP, and De Freitas N. Taking the human out of the loop: A review of Bayesian optimization. Proceedings of the IEEE 2015; 104:148–75
9. Gramacy RB and Lee HK. Optimization Under Unknown Constraints. arXiv preprint arXiv:1004.4027 2010
10. Rasmussen CE. Gaussian processes in machine learning. *Summer school on machine learning*. Springer. 2003 :63–71
11. Audet C, Denni J, Moore D, Booker A, and Frank P. A surrogate-model-based method for constrained optimization. *8th symposium on multidisciplinary analysis and optimization*. 2000 :4891
12. Da Veiga S and Marrel A. Gaussian process modeling with inequality constraints. *Annales de la Faculté des sciences de Toulouse: Mathématiques*. Vol. 21. 3. 2012 :529–55
13. Maier M, Rupenyan A, Bobst C, and Wegener K. Self-optimizing grinding machines using Gaussian process models and constrained Bayesian optimization. The International Journal of Advanced Manufacturing Technology 2020; 108:539–52
14. Gelbart MA, Snoek J, and Adams RP. Bayesian optimization with unknown constraints. arXiv preprint arXiv:1403.5607 2014
15. Snoek JR. Bayesian optimization and semiparametric models with applications to assistive technology. Doctoral dissertation. University of Toronto, 2013. Available from: `https://tspace.library.utoronto.ca/handle/1807/43732`

16. Gardner JR, Kusner MJ, Xu ZE, Weinberger KQ, and Cunningham JP. Bayesian Optimization with Inequality Constraints. *ICML*. Vol. 2014. 2014 :937–45

17. Ludl PO, Heese R, Höller J, Asprion N, and Bortz M. Using machine learning models to explore the solution space of large nonlinear systems underlying flowsheet simulations with constraints. Frontiers of Chemical Science and Engineering 2022; 16:183–97

18. Daxberger E, Makarova A, Turchetta M, and Krause A. Mixed-variable bayesian optimization. arXiv preprint arXiv:1907.01329 2019

19. Kim J and Choi S. On Uncertainty Estimation by Tree-based Surrogate Models in Sequential Model-based Optimization. *International Conference on Artificial Intelligence and Statistics*. PMLR. 2022 :4359–75

20. Lei B, Kirk TQ, Bhattacharya A, Pati D, Qian X, Arroyave R, and Mallick BK. Bayesian optimization with adaptive surrogate models for automated experimental design. npj Computational Materials 2021; 7:1–12

21. Bonilla EV, Chai K, and Williams C. Multi-task Gaussian process prediction. Advances in neural information processing systems 2007; 20

22. Snelson E and Ghahramani Z. Local and global sparse Gaussian process approximations. *Artificial Intelligence and Statistics*. PMLR. 2007 :524–31

23. Garnett R, Osborne M, and Hennig P. Active Learning of Linear Embeddings for Gaussian Processes. *30th Conference on Uncertainty in Artificial Intelligence (UAI 2014)*. AUAI Press. 2014 :230–9

24. Garrido-Merchán EC and Hernández-Lobato D. Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes. Neurocomputing 2020; 380:20–35

25. Eriksson D and Poloczek M. Scalable constrained bayesian optimization. *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021 :730–8

26. Oh C, Gavves E, and Welling M. BOCK: Bayesian optimization with cylindrical kernels. *International Conference on Machine Learning*. PMLR. 2018 :3868–77

27. Wang Z, Gehring C, Kohli P, and Jegelka S. Batched large-scale Bayesian optimization in high-dimensional spaces. *International Conference on Artificial Intelligence and Statistics*. PMLR. 2018 :745–54

28. Breiman L. Bagging predictors. Machine learning 1996; 24:123–40

29. Friedman JH. Greedy function approximation: a gradient boosting machine. Annals of statistics 2001 :1189–232

30. Lakshminarayanan B, Roy DM, and Teh YW. Mondrian forests: Efficient online random forests. Advances in neural information processing systems 2014; 27

31. Feng C, Zhang X, Wei Y, Zhang W, Hou N, Xu J, Jia K, Yao Y, Xie X, Jiang B, *et al.* Estimating surface downward longwave radiation using machine learning methods. Atmosphere 2020; 11:1147

32. Thebelt A, Kronqvist J, Mistry M, Lee RM, Sudermann-Merx N, and Misener R. ENTMOOT: a framework for optimization over ensemble tree models. Computers & Chemical Engineering 2021; 151:107343

33. Nielsen D. Tree boosting with xgboost-why does xgboost win" every" machine learning competition? MA thesis. NTNU, 2016

34. Breiman L. Random forests. Machine learning 2001; 45:5–32

35. Hutter F, Hoos HH, and Leyton-Brown K. Sequential model-based optimization for general algorithm configuration. *International conference on learning and intelligent optimization*. Springer. 2011 :507–23

36. Chipman HA, George EI, and McCulloch RE. BART: Bayesian additive regression trees. The Annals of Applied Statistics 2010; 4:266–98

37. Nickson T, Osborne MA, Reece S, and Roberts SJ. Automated machine learning on big data using stochastic algorithm tuning. arXiv preprint arXiv:1407.7969 2014

38. Malinin A, Prokhorenkova L, and Ustimenko A. Uncertainty in gradient boosting via ensembles. arXiv preprint arXiv:2006.10562 2020

39. Nelder JA and Mead R. A simplex method for function minimization. The computer journal 1965; 7:308–13

40. Clerx M, Robinson M, Lambert B, Lei CL, Ghosh S, Mirams GR, and Gavaghan DJ. Probabilistic inference on noisy time series (PINTS). arXiv preprint arXiv:1812.07388 2018

41. Cisbani E, Del Dotto A, Fanelli C, Williams M, Alfred M, Barbosa F, Barion L, Berdnikov V, Brooks W, Cao T, *et al.* AI-optimized detector design for the future Electron-Ion Collider: the dual-radiator RICH case. Journal of Instrumentation 2020; 15:P05009

42. Rapin J, Bennet P, Centeno E, Haziza D, Moreau A, and Teytaud O. Open source evolutionary structured optimization. *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*. 2020 :1599–607

43. MacKay DJ, Mac Kay DJ, *et al.* Information theory, inference and learning algorithms. Cambridge university press, 2003

44. Bergstra J, Bardenet R, Bengio Y, and Kégl B. Algorithms for hyper-parameter optimization. Advances in neural information processing systems 2011; 24

45. Friedman JH. Multivariate adaptive regression splines. The annals of statistics 1991; 19:1–67

46. Williams CK and Rasmussen CE. Gaussian processes for machine learning. Vol. 2. 3. MIT press Cambridge, MA, 2006

47. Srinivas N, Krause A, Seeger M, and Kakade SM. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. Ed. by Fürnkranz J and Joachims T. Omnipress, 2010 :1015–22

48. Hernández-Lobato JM, Hoffman MW, and Ghahramani Z. Predictive entropy search for efficient global optimization of black-box functions. Advances in neural information processing systems 2014; 27

49. Wang H, Stein B van, Emmerich M, and Back T. A new acquisition function for Bayesian optimization based on the moment-generating function. *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2017 :507–12

50. Liu DC and Nocedal J. On the limited memory BFGS method for large scale optimization. Mathematical programming 1989; 45:503–28

51. Feurer M, Klein A, Eggensperger K, Springenberg J, Blum M, and Hutter F. Efficient and robust automated machine learning. Advances in neural information processing systems 2015; 28

52. Candelieri A, Perego R, and Archetti F. Bayesian optimization of pump operations in water distribution systems. Journal of Global Optimization 2018; 71:213–35
53. Hansen N and Ostermeier A. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. *Proceedings of IEEE international conference on evolutionary computation*. IEEE. 1996 :312–7
54. Brochu E, Cora VM, and De Freitas N. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:1012.2599 2010
55. Schonlau M. Computer experiments and global optimization. Doctoral dissertation. University of Waterloo, 1997. Available from: `https://uwspace.uwaterloo.ca/bitstream/handle/10012/190/nq22234.pdf?sequence=1`
56. Schonlau M, Welch WJ, and Jones DR. Global versus local search in constrained optimization of computer models. Lecture Notes-Monograph Series 1998 :11–25
57. Sui Y, Gotovos A, Burdick J, and Krause A. Safe exploration for optimization with Gaussian processes. *International conference on machine learning*. PMLR. 2015 :997–1005
58. Murphy KP. Machine learning: a probabilistic perspective. MIT press, 2012
59. Friedman JH. Stochastic gradient boosting. Computational statistics & data analysis 2002; 38:367–78
60. Breiman L. Arcing classifier (with discussion and a rejoinder by the author). The annals of statistics 1998; 26:801–49
61. Roy DM, Teh YW, *et al.* The Mondrian Process. *NIPS*. 2008 :1377–84
62. Lakshminarayanan B, Roy DM, and Teh YW. Mondrian forests for large-scale regression when uncertainty matters. *Artificial Intelligence and Statistics*. PMLR. 2016 :1478–87
63. Kapelner A and Bleich J. bartMachine: Machine learning with Bayesian additive regression trees. arXiv preprint arXiv:1312.2171 2013
64. Hill J, Linero A, and Murray J. Bayesian additive regression trees: A review and look forward. Annual Review of Statistics and Its Application 2020; 7:251–78
65. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, and Teller E. Equation of state calculations by fast computing machines. The journal of chemical physics 1953; 21:1087–92
66. Hastings WK. Monte Carlo sampling methods using Markov chains and their applications. 1970
67. Geman S and Geman D. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Transactions on pattern analysis and machine intelligence 1984 :721–41
68. Duan T, Anand A, Ding DY, Thai KK, Basu S, Ng A, and Schuler A. Ngboost: Natural gradient boosting for probabilistic prediction. *International Conference on Machine Learning*. PMLR. 2020 :2690–700
69. Lim YF, Ng CK, Vaitesswar U, and Hippalgaonkar K. Extrapolative Bayesian Optimization with Gaussian Process and Neural Network Ensemble Surrogate Models. Advanced Intelligent Systems 2021; 3:2100101
70. Hutter F, Hoos H, and Leyton-Brown K. An evaluation of sequential model-based optimization for expensive blackbox functions. *Proceedings of the 15th annual conference companion on Genetic and evolutionary computation*. 2013 :1209–16

71. Dewancker I, McCourt M, Clark S, Hayes P, Johnson A, and Ke G. A stratified analysis of Bayesian optimization methods. arXiv preprint arXiv:1603.09441 2016

72. Hutter F, Hoos HH, Leyton-Brown K, and Murphy K. Time-bounded sequential parameter optimization. *International Conference on Learning and Intelligent Optimization*. Springer. 2010 :281–98

73. The scikit-optimize contributors. scikit-optimize/scikit-optimize. Version 0.8.1. 2020 Sep 1. Available from: `https://scikit-optimize.github.io`

74. Jiménez J and Ginebra J. pygpgo: Bayesian optimization for python. Journal of Open Source Software 2017; 2:431

75. Brophy J and Lowd D. Instance-Based Uncertainty Estimation for Gradient-Boosted Regression Trees. arXiv preprint arXiv:2205.11412 2022

76. Horiguchi A, Santner TJ, Sun Y, and Pratola MT. Using BART to Perform Pareto Optimization and Quantify its Uncertainties. Technometrics 2022 :1–11

77. Logan BR, Sparapani R, McCulloch RE, and Laud PW. Decision making and uncertainty quantification for individualized treatments using Bayesian Additive Regression Trees. Statistical methods in medical research 2019; 28:1079–93

78. Scillitoe A, Seshadri P, and Girolami M. Uncertainty quantification for data-driven turbulence modelling with Mondrian forests. Journal of Computational Physics 2021; 430:110116

79. Kendall A and Gal Y. What uncertainties do we need in bayesian deep learning for computer vision? Advances in neural information processing systems 2017; 30

80. Tagasovska N and Lopez-Paz D. Single-model uncertainties for deep learning. Advances in Neural Information Processing Systems 2019; 32

81. Notin P, Hernández-Lobato JM, and Gal Y. Improving black-box optimization in VAE latent space using decoder uncertainty. Advances in Neural Information Processing Systems 2021; 34

82. Abdar M, Pourpanah F, Hussain S, Rezazadegan D, Liu L, Ghavamzadeh M, Fieguth P, Cao X, Khosravi A, Acharya UR, *et al.* A review of uncertainty quantification in deep learning: Techniques, applications and challenges. Information Fusion 2021; 76:243–97

83. Mervin LH, Johansson S, Semenova E, Giblin KA, and Engkvist O. Uncertainty quantification in drug design. Drug discovery today 2021; 26:474–89

84. Aggarwal CC, Hinneburg A, and Keim DA. On the surprising behavior of distance metrics in high dimensional space. *International conference on database theory*. Springer. 2001 :420–34

85. Davies A and Ghahramani Z. The random forest kernel and other kernels for big data from random partitions. arXiv preprint arXiv:1402.4293 2014

86. Gerber P, Jöckel L, and Kläs M. A Study on Mitigating Hard Boundaries of Decision-Tree-based Uncertainty Estimates for AI Models. arXiv preprint arXiv:2201.03263 2022

87. Mišić VV. Optimization of tree ensembles. Operations Research 2020; 68:1605–24

88. Lindauer M, Eggensperger K, Feurer M, Biedenkapp A, Deng D, Benjamins C, Ruhkopf T, Sass R, and Hutter F. SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization. *ArXiv: 2109.09831*. 2021

89. Griffiths RR and Hernández-Lobato JM. Constrained Bayesian optimization for automatic chemical design using variational autoencoders. Chemical science 2020; 11:577–86

90. Siivola E, Paleyes A, González J, and Vehtari A. Good practices for Bayesian optimization of high dimensional structured spaces. Applied AI Letters 2021; 2:e24

91. Janet JP, Duan C, Yang T, Nandy A, and Kulik HJ. A quantitative uncertainty metric controls error in neural network-driven chemical discovery. Chemical science 2019; 10:7913–22

92. Hutter F, Xu L, Hoos HH, and Leyton-Brown K. Algorithm runtime prediction: Methods & evaluation. Artificial Intelligence 2014; 206:79–111

93. Hüllermeier E and Waegeman W. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. Machine Learning 2021; 110:457–506

94. Gneiting T. Quantiles as optimal point forecasts. International Journal of forecasting 2011; 27:197–207

95. Meinshausen N and Ridgeway G. Quantile regression forests. Journal of Machine Learning Research 2006; 7

96. Koenker R and Hallock KF. Quantile regression. Journal of economic perspectives 2001; 15:143–56

97. Wager S, Hastie T, and Efron B. Confidence intervals for random forests: The jackknife and the infinitesimal jackknife. The Journal of Machine Learning Research 2014; 15:1625–51

98. Gal Y and Ghahramani Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *international conference on machine learning*. PMLR. 2016 :1050–9

99. Hernández-Lobato JM and Adams R. Probabilistic backpropagation for scalable learning of bayesian neural networks. *International conference on machine learning*. PMLR. 2015 :1861–9

100. Thebelt A, Tsay C, Lee RM, Sudermann-Merx N, Walz D, Tranter T, and Misener R. Multi-objective constrained optimization for energy applications via tree ensembles. Applied Energy 2022; 306:118061

101. Yeh IC. Modeling of strength of high-performance concrete using artificial neural networks. Cement and Concrete research 1998; 28:1797–808

102. Tsanas A and Xifara A. Accurate quantitative estimation of energy performance of residential buildings using statistical machine learning tools. Energy and buildings 2012; 49:560–7

103. Cortez P, Cerdeira A, Almeida F, Matos T, and Reis J. Modeling wine preferences by data mining from physicochemical properties. Decision support systems 2009; 47:547–53

104. Gerritsma J, Onnink R, and Versluis A. Geometry, resistance and stability of the delft systematic yacht hull series. International shipbuilding progress 1981; 28:276–97

105. Ortigosa I, Lopez R, and Garcia J. A neural networks approach to residuary resistance of sailing yachts prediction. *Proceedings of the international conference on marine engineering MARINE*. Vol. 2007. 2007 :250

106. Sobol' IM. On the distribution of points in a cube and the approximate evaluation of integrals. Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki 1967; 7:784–802

107. Boriah S, Chandola V, and Kumar V. Similarity measures for categorical data: A comparative evaluation. *Proceedings of the 2008 SIAM international conference on data mining*. SIAM. 2008 :243–54

108. Van Der Walt S, Colbert SC, and Varoquaux G. The NumPy array: a structure for efficient numerical computation. Computing in science & engineering 2011; 13:22–30

109. Merrill E, Fern A, Fern XZ, and Dolatnia N. An Empirical Study of Bayesian Optimization: Acquisition Versus Partition. J. Mach. Learn. Res. 2021; 22:4–1

110. Wang Z and Jegelka S. Max-value entropy search for efficient Bayesian optimization. *International Conference on Machine Learning*. PMLR. 2017 :3627–35

111. Hansen N. The CMA evolution strategy: a comparing review. Towards a new evolutionary computation 2006 :75–102

112. Dang VH, Vien NA, and Chung T. A covariance matrix adaptation evolution strategy in reproducing kernel Hilbert space. Genetic Programming and Evolvable Machines 2019; 20:479–501

113. Gao F and Han L. Implementing the Nelder-Mead simplex algorithm with adaptive parameters. Computational Optimization and Applications 2012; 51:259–77

114. Hedar AR and Fukushima M. Derivative-free filter simulated annealing method for constrained continuous global optimization. Journal of Global optimization 2006; 35:521–49

115. Coello CAC and Montes EM. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Advanced Engineering Informatics 2002; 16:193–203

116. Michalewicz Z and Schoenauer M. Evolutionary algorithms for constrained parameter optimization problems. Evolutionary computation 1996; 4:1–32

117. Herten J van der. Bayesian Optimization with black-box constraints. 2017. Available from: `https://gpflowopt.readthedocs.io/en/latest/notebooks/constrained_bo.html` [Accessed on: 2022 Aug 10]

118. Pelamatti J, Brevault L, Balesdent M, Talbi EG, and Guerin Y. Efficient global optimization of constrained mixed variable problems. Journal of Global Optimization 2019; 73:583–613

119. Ru B, Alvi A, Nguyen V, Osborne MA, and Roberts S. Bayesian optimisation over multiple continuous and categorical inputs. *International Conference on Machine Learning*. PMLR. 2020 :8276–85

120. Griffiths RR and Hernández-Lobato JM. Constrained bayesian optimization for automatic chemical design. arXiv preprint arXiv:1709.05501 2017

121. Letham B, Karrer B, Ottoni G, and Bakshy E. Constrained Bayesian optimization with noisy experiments. Bayesian Analysis 2019; 14:495–519

122. Wu Y, Yin Q, Jie H, Wang B, and Zhao J. A RBF-based constrained global optimization algorithm for problems with computationally expensive objective and constraints. Structural and Multidisciplinary Optimization 2018; 58:1633–55

123. Ariafar S, Coll-Font J, Brooks DH, and Dy JG. ADMMBO: Bayesian Optimization with Unknown Constraints using ADMM. J. Mach. Learn. Res. 2019; 20:1–26

124. Ackley D. A connectionist machine for genetic hillclimbing. Vol. 28. Springer Science & Business Media, 2012

125. Keane A. Experiences with optimizers in structural design. *Proceedings of the conference on adaptive computing in engineering design and control*. Vol. 94. 1994 :14–27

126. Mishra SK. Minimization of Keane's bump function by the repulsive particle swarm and the differential evolution methods. Available at SSRN 983836 2007

127. Wan X, Nguyen V, Ha H, Ru B, Lu C, and Osborne MA. Think global and act local: Bayesian optimisation over high-dimensional categorical and mixed search spaces. arXiv preprint arXiv:2102.07188 2021

128. Zuo Y, Dezfouli A, Chades I, Alexander D, and Muir BW. Bayesian Optimisation for Mixed-Variable Inputs using Value Proposals. arXiv preprint arXiv:2202.04832 2022

129. Rosenbrock HH. An automatic method for finding the greatest or least value of a function. The computer journal 1960; 3:175–84

130. Igel C, Suttorp T, and Hansen N. A computational efficient covariance matrix update and a (1+ 1)-CMA for evolution strategies. *Proceedings of the 8th annual conference on Genetic and evolutionary computation.* 2006 :453–60

131. Eriksson D, Pearce M, Gardner J, Turner RD, and Poloczek M. Scalable global optimization via local bayesian optimization. Advances in neural information processing systems 2019; 32

132. Matthews AGdG, Van Der Wilk M, Nickson T, Fujii K, Boukouvalas A, León-Villagrá P, Ghahramani Z, and Hensman J. GPflow: A Gaussian Process Library using TensorFlow. J. Mach. Learn. Res. 2017; 18:1–6

133. Zhu C, Byrd RH, Lu P, and Nocedal J. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. ACM Transactions on mathematical software (TOMS) 1997; 23:550–60

134. Valladares H and Tovar A. Design optimization of sandwich composite armors for blast mitigation using bayesian optimization with single and multi-fidelity data. Tech. rep. 2020

135. Harrison Jr D and Rubinfeld DL. Hedonic housing prices and the demand for clean air. Journal of environmental economics and management 1978; 5:81–102

136. Carlisle M. Racist data destruction? A Boston housing dataset controversy and an experiment in data forensics. 2019. Available from: `https://medium.com/@docintangible/racist-data-destruction-113e3eff54a8` [Accessed on: 2022 Jul 19]

137. Hoffman KM, Trawalter S, Axt JR, and Oliver MN. Racial bias in pain assessment and treatment recommendations, and false beliefs about biological differences between blacks and whites. Proceedings of the National Academy of Sciences 2016; 113:4296–301

138. Ingold D and Soper S. Amazon doesn't consider the race of its customers. Should it? 2016. Available from: `https://www.bloomberg.com/graphics/2016-amazon-same-day/` [Accessed on: 2022 Jul 19]

139. Knight HE, Deeny SR, Dreyer K, Engmann J, Mackintosh M, Raza S, Stafford M, Tesfaye R, and Steventon A. Challenging racism in the use of health data. The Lancet Digital Health 2021; 3:e144–e146

140. D'ignazio C and Klein LF. Data feminism. MIT press, 2020

141. Xu W, Jones CN, Svetozarevic B, Laughman CR, and Chakrabarty A. VABO: Violation-Aware Bayesian Optimization for Closed-Loop Control Performance Optimization with Unmodeled Constraints. arXiv preprint arXiv:2110.07479 2021

142. Zhang Y, Zhang X, and Frazier PI. Two-step Lookahead Bayesian Optimization with Inequality Constraints. arXiv preprint arXiv:2112.02833 2021

143. Zhang S, Yang F, Yan C, Zhou D, and Zeng X. An Efficient Batch-Constrained Bayesian Optimization Approach for Analog Circuit Synthesis via Multiobjective Acquisition Ensemble. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 2021; 41:1–14

144. Singer S and Singer S. Complexity analysis of Nelder-Mead search iterations. *Proceedings of the 1. Conference on Applied Mathematics and Computation, Dubrovnik, Croatia.* PMF–Matematički odjel, Zagreb. 1999 :185–96

145. Pham N. Improved Nelder Mead's simplex method and applications. Doctoral dissertation. Auburn University, Graduate Faculty, 2012. Available from: `http://etd.auburn.edu/handle/10415/2985`

146. Singer S and Singer S. Efficient implementation of the Nelder–Mead search algorithm. Applied Numerical Analysis & Computational Mathematics 2004; 1:524–34

147. Poli R, Kennedy J, and Blackwell T. Particle swarm optimization. Swarm intelligence 2007; 1:33–57

148. Cornford D, Nabney I, and Williams C. Adding constrained discontinuities to gaussian process models of wind fields. Advances in Neural Information Processing Systems 1998; 11

149. Thebelt A, Tsay C, Lee RM, Sudermann-Merx N, Walz D, Shafei B, and Misener R. Tree ensemble kernels for Bayesian optimization with known constraints over mixed-feature spaces. arXiv preprint arXiv:2207.00879 2022

150. Balog M, Lakshminarayanan B, Ghahramani Z, Roy DM, and Teh YW. The mondrian kernel. arXiv preprint arXiv:1606.05241 2016

151. Rahnamayan S, Tizhoosh HR, and Salama MM. A novel population initialization method for accelerating evolutionary algorithms. Computers & Mathematics with Applications 2007; 53:1605–14

152. Townsend A. Constrained optimization in Chebfun. 2014. Available from: `https://www.chebfun.org/examples/opt/ConstrainedOptimization.html` [Accessed on: 2022 Aug 12]

153. Lemonge AC, Barbosa HJ, Borges CC, and Silva FB. Constrained optimization problems in mechanical engineering design using a real-coded steady-state genetic algorithm. Mecánica Computacional 2010; 29:9287–303

154. Molga M and Smutnicki C. Test functions for optimization needs. Test functions for optimization needs 2005; 101:48

155. Beale EML. On an iterative method for finding a local minimum of a function of more than one variable. 25. Statistical Techniques Research Group, Section of Mathematical Statistics ..., 1958

156. Chen T and Guestrin C. Xgboost: A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining.* 2016 :785–94

157. Fisher RA. The use of multiple measurements in taxonomic problems. Annals of eugenics 1936; 7:179–88

158. Balandat M, Karrer B, Jiang D, Daulton S, Letham B, Wilson AG, and Bakshy E. BoTorch: a framework for efficient Monte-Carlo Bayesian optimization. Advances in neural information processing systems 2020; 33:21524–38

Appendix A

Optimisation benchmarks

A.1 Continuous benchmarks

A.1.1 Branin Hoo

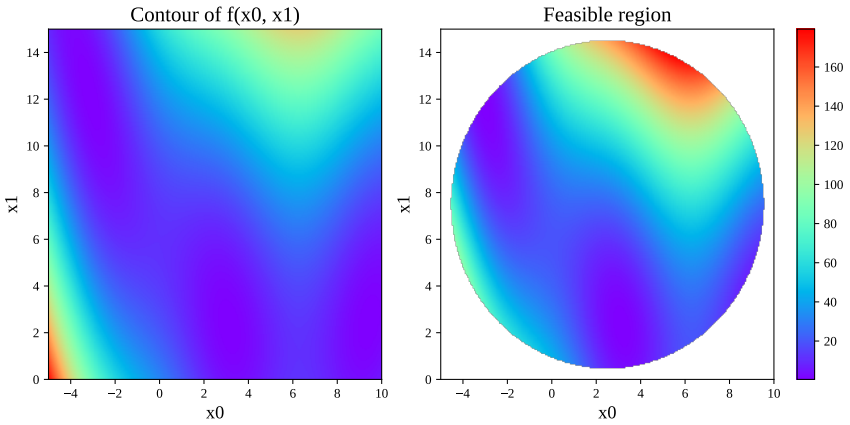


Figure A.1: The 2D Branin Hoo problem with 1 black-box constraint. We depict the objective function and feasible region of the 2D Branin Hoo problem.

The constrained 2D Branin Hoo problem [14] is defined by:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^2} \; & f(\mathbf{x}) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10 \\ s.t. \; & c_1(\mathbf{x}) = (x_1 - 2.5)^2 + (x_2 - 7.5)^2 - 50 \leq 0 \\ & x_1 \in [-5.0, 10.0], x_2 \in [0.0, 15.0] \end{aligned}$$

The function has a constrained minimum $f(\mathbf{x}^*) = 0.397887$ at $(\pi, 2.275)$. Its objective and feasible region are plotted in Figure A.1.

A.1.2 Rosenbrock problem

This is the first work that optimises this version of the constrained 2D Rosenbrock function. We created this benchmark problem with the goal to present a constrained optimisation problem with a discontinuous constraint function. The objective function is the 2D Rosenbrock function

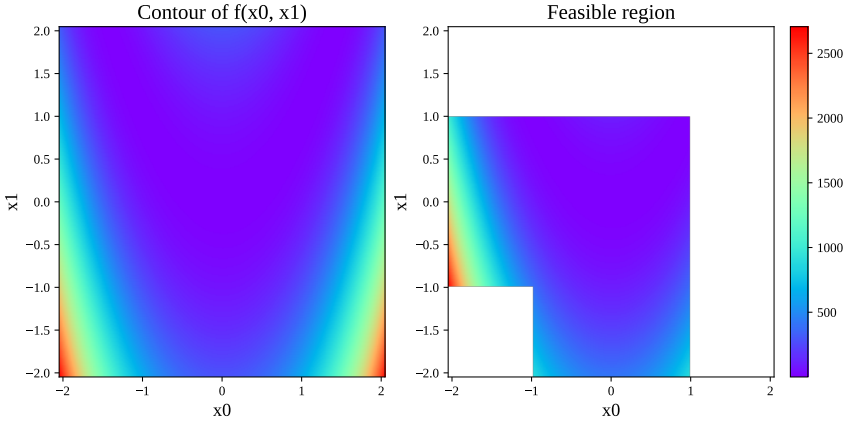


Figure A.2: The 2D Rosenbrock problem with 1 black-box constraint. We depict the objective function and feasible region of the 2D Rosenbrock problem.

[129], and the constraint function is novel. The problem is defined by:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^2} \; & f(\mathbf{x}) = (1-x_1)^2 + 100(x_2 - x_1^2)^2 \\ s.t. \; & c_1(\mathbf{x}) = |\max(x_1, x_2)| - 1 \leq 0 \\ & x_1, x_2 \in [-2.048, 2.048] \end{aligned}$$

The function has a constrained minimum $f(\mathbf{x}^*) = 0.0$ at $(1, 1)$. Its objective and feasible region are plotted in Figure A.2.

A.1.3 G6 problem

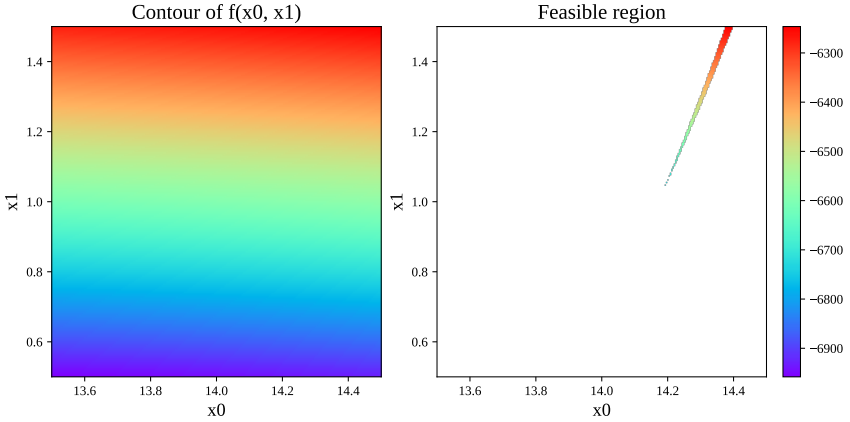


Figure A.3: The 2D G6 problem with 1 black-box constraint. We depict the objective function and feasible region of the 2D G6 problem [116].

The 2D G6 problem [114, 116] is defined by:

$$\begin{aligned} \min_{x\in\mathbb{R}^2} \; & f(\mathbf{x}) = (x_1 - 10)^3 + (x_2 - 20)^3 \\ s.t. \; & c_1(\mathbf{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0 \\ & c_2(\mathbf{x}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0 \end{aligned}$$

$$x_1 \in [13.5, 14.5], x_2 \in [0.5, 1.5]$$

The original input space bounds are defined as $[13, 100] \times [0, 100]$, but we adapted the problem bounds to $[13.5, 14.5] \times [0.5, 1.5]$. The smaller input domain makes this problem easier for surrogates, which have to learn the feasible region. Still, the size of the feasible region remains relatively small.

The function has a constrained minimum $f(\mathbf{x}^*) = -6961.8138$ at $(14.0950, 0.8430)$. Its objective and feasible region are plotted in Figure A.3.

A.1.4 Gardner problem

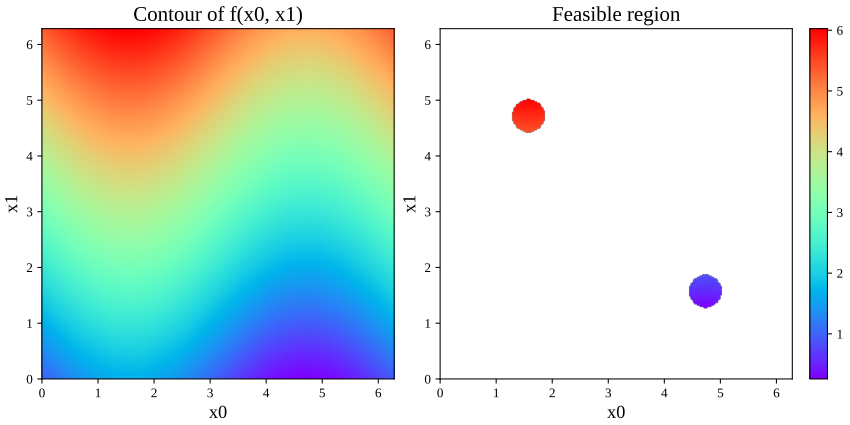


Figure A.4: The 2D Gardner problem with 1 black-box constraint. We depict the objective function and feasible region of the 2D Gardner problem [16].

The constrained 2D Gardner problem [16] is defined by:

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^2} \; & f(\mathbf{x}) = \sin(x_1) + x_2 \\ s.t. \; & c_1(\mathbf{x}) = \sin(x_1)\sin(x_2) + 0.95 \leq 0 \\ & x_1, x_2 \in [0, 2\pi] \end{aligned}$$

The function has a constrained minimum $f(\mathbf{x}^*) = 0.2532$ at $(4.7124, 1.2532)$. Its objective and feasible region are plotted in Figure A.4.

A.1.5 Alpine problem

The 2D Alpine-1 function was introduced by Rahnamayan *et al.* [151]. As the present work focuses on constrained optimisation problems, we added a constraint to the function which is a ring around the region containing the global optimum. We created this benchmark problem with the goal to present a constrained optimisation problem with a highly multimodal objective function. To highlight the consequences of not exploring the full domain on this problem, we modify the objective function, subtracting one in the inner feasible region (see the visualisation Figure A.5). This does not change the exploration behaviour of any surrogate model that we evaluated on the function, while exposing the lack of exploration on this function. The problem

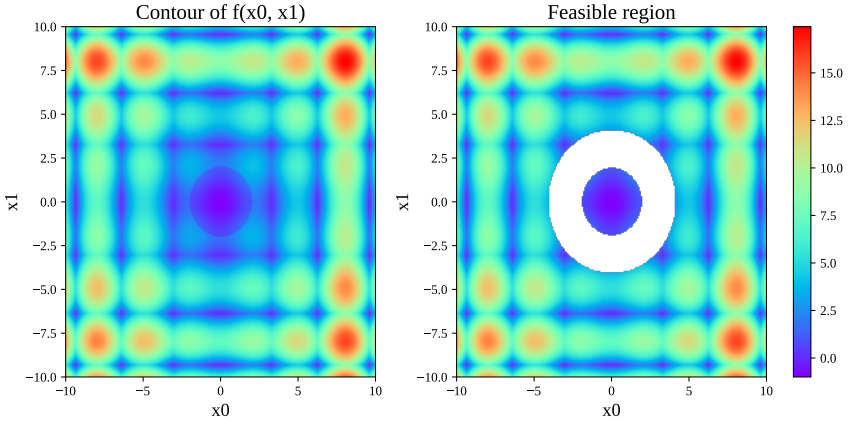


Figure A.5: The 2D Alpine problem with 1 black-box constraint. We depict the objective function and feasible region of the 2D Alpine problem.

is defined by:

$$\min_{\mathbf{x}\in\mathbb{R}^2} f(\mathbf{x}) = \begin{cases} \sum_{i=1}^{2} |x_i \sin(x_i) + 0.1x_i| - 1, & if \ \|\mathbf{x}\|_2 \leq 2 \\ \sum_{i=1}^{2} |x_i \sin(x_i) + 0.1x_i|, & else \end{cases}$$
$$s.t.\ c_1(\mathbf{x}) = (\|\mathbf{x}\|_2 - 2)\,(4 - \|\mathbf{x}\|_2) \leq 0$$
$$x_1, x_2 \in [-10, 10]$$

The function has a constrained minimum $f(\mathbf{x}^*) = -1.0$ at $(0, 0)$. Its objective and feasible region are plotted in Figure A.5.

A.1.6 Townsend problem

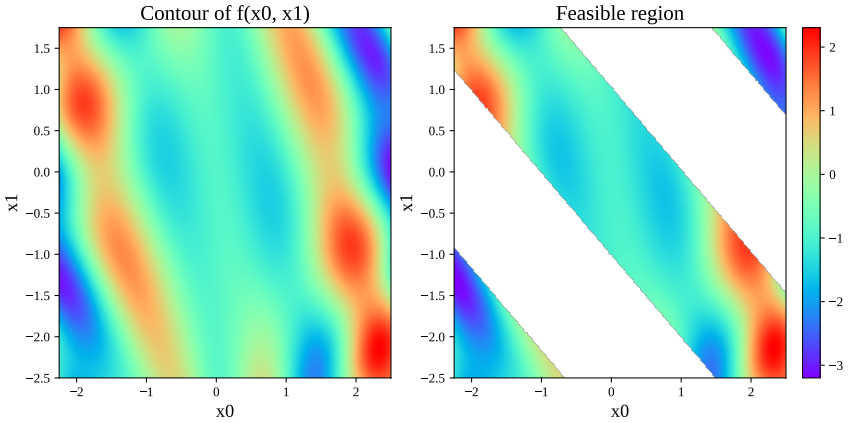


Figure A.6: The 2D Townsend problem with 1 black-box constraint. We depict the objective function and feasible region of the 2D Townsend problem [117].

The Townsend function was introduced by Townsend [152]. In this work, we will use the constrained version that was introduced in by van der Herten [117], which defined multiple

disjoint feasible regions. The problem is defined by:

$$\begin{aligned}
\min_{\mathbf{x}\in\mathbb{R}^2} \; & f(\mathbf{x}) = -\cos\left((x_1 - 0.1)x_2\right)^2 - x_1 \sin\left(3x_1 x_2\right) \\
s.t. \; & c_1(x) = -\cos\left(\frac{3}{2}\pi x_1\right)\cos\left(\frac{3}{2}x_2\right) - \sin\left(\frac{3}{2}\pi x_1\right)\sin\left(\frac{3}{2}x_2\right) \leq 0 \\
& x_1 \in [-2.25, 2.5], x_2 \in [-2.5, 1.75]
\end{aligned}$$

The function has a constrained minimum $f(\mathbf{x}^*) = -3.2$ at $(-2.25, -1.2964)$. Its objective and feasible region are plotted in Figure A.6.

A.1.7 Sphere problem

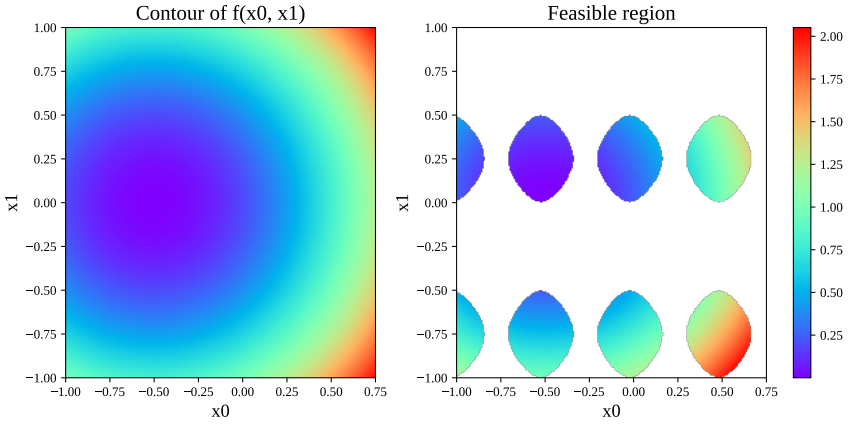


Figure A.7: The 2D Sphere problem with 1 black-box constraint. We depict the objective function and feasible region of the 2D Sphere problem.

The 2D Sphere function is a standard benchmark function in optimisation. As the present work focuses on constrained optimisation problems, we added a constraint to the function which is a ring around the region containing the global optimum. We created this benchmark problem with the goal to present a constrained optimisation problem with a multiple disjoint feasible regions, while the underlying objective is simple to optimise. The problem is defined by:

$$\begin{aligned}
\min_{\mathbf{x}\in\mathbb{R}^2} \; & f(\mathbf{x}) = (x_1 + 0.5)^2 + x_2^2 \\
s.t. \; & c_1(\mathbf{x}) = \sin(4\pi(x_1 - 0.1)) - 2\sin^2(2\pi x_2) + 0.95 \leq 0 \\
& x_1 \in [-1.0, 0.75], x_2 \in [-1., 1.]
\end{aligned}$$

The function has a constrained minimum $f(\mathbf{x}^*) = 0.0$ at $\mathbf{x}^* = (-0.5, 0)$. Its objective and feasible region are plotted in Figure A.7.

A.1.8 Welded Beam Design problem

The 4D welded beam design problem [25, 114, 115, 153] minimises the cost of a welded beam. We use the formulation of [25, 153] who incorporate two of the original constraints into the boundary conditions, leading to the following problem formulation with 5 constraints:

$$\min_{\mathbf{x}\in\mathbb{R}^4} f(\mathbf{x}) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (x_2 + 14)$$

$$
\begin{aligned}
s.t.\ & c_1(\mathbf{x}) = \tau(\mathbf{x}) - 13000 \leq 0 \\
& c_2(\mathbf{x}) = \sigma(\mathbf{x}) - 30000 \leq 0 \\
& c_3(\mathbf{x}) = x_1 - x_4 \leq 0 \\
& c_4(\mathbf{x}) = 6000 - P_c(\mathbf{x}) \leq 0 \\
& c_5(\mathbf{x}) = \delta(\mathbf{x}) - 0.25 \leq 0 \\
& x_1 \in [0.125, 10], x_2, x_3, x_4 \in [0.1, 10]
\end{aligned}
$$

where

$$
\begin{aligned}
\tau(\mathbf{x}) &= \sqrt{\tau_1^2(\mathbf{x}) + \tau_2^2(\mathbf{x}) + \frac{x_2 \tau_1(\mathbf{x}) \tau_2(\mathbf{x})}{\sqrt{0.25(x_2^2 + (x_1 + x_3)^2)}}} \\
\tau_1(\mathbf{x}) &= \frac{6000}{\sqrt{2} x_1 x_2}, \sigma(\mathbf{x}) = \frac{504000}{x_3^2 x_4}, \delta(\mathbf{x}) = \frac{2.1952}{x_3^3 x_4} \\
\tau_2(\mathbf{x}) &= \frac{6000(0.5 x_2 + 14)\sqrt{0.25(x_2^2 + (x_1 + x_3)^2)}}{1.414 x_1 x_2 (x_2^2/12 + 0.25(x_1 + x_3)^2)} \\
P_c(\mathbf{x}) &= 64746.022(1 - 0.0282346 x_3) x_3 x_4^3
\end{aligned}
$$

The true minimum is unknown. The best known result is $f(\mathbf{x}^*) = 2.3811$ at $\mathbf{x}^* = (0.2444, 6.2158, 8.2939, 0.2444)$.

A.1.9 Ackley problem

The 20D Ackley problem [124, 131] is defined by:

$$
\begin{aligned}
\min_{x \in \mathbb{R}^{20}} f(x) = -20 \exp & \left[-0.2 \sqrt{\frac{1}{20} \sum_{i=1}^{20} x_i^2} \right] - \exp \left[\frac{1}{20} \sum_{i=1}^{20} \cos(2\pi x_i) \right] + e + 20 \\
s.t.\ & c_1(\mathbf{x}) = \sum_{i=1}^{20} x_i \leq 0 \\
& c_2(\mathbf{x}) = \|\mathbf{x}\|_2 - 5 \leq 0 \\
& \mathbf{x} \in [-5, 10]^{20}
\end{aligned}
$$

The function has a global minimum $f(\mathbf{x}^*) = 0.0$ at the origin.

A.1.10 Keane Bump problem

The 30D Keane Bump problem [25, 125, 126] is defined by:

$$
\begin{aligned}
\min_{\mathbf{x} \in \mathbb{R}^{30}} f(\mathbf{x}) &= - \left| \frac{\sum_{i=1}^{30} \cos^4(x_i) - 2 \prod_{i=1}^{30} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{30} i x_i^2}} \right| \\
s.t.\ & c_1(\mathbf{x}) = 0.75 - \prod_{i=1}^{30} x_i \leq 0
\end{aligned}
$$

$$c_2(\mathbf{x}) = \sum_{i=1}^{30} -225 \leq 0$$
$$\mathbf{x} \in [0, 10]^{30}$$

The true minimum is unknown. The best known result is $f(\mathbf{x}^*) = -0.818056222$. The location of this value can be found in [126].

A.2 Mixed benchmarks

A.2.1 Mixed Branin problem

The Mixed Branin problem [118] is a mixed domain version of the Branin Hoo problem from Gelbart *et al.* [14]. Additionally to the two continuous dimensions, two categorical dimension are added. The resulting problem is defined by:

$$\min_{x_1,x_2,z_1,z_2 \in \mathcal{X}} f(x_1, x_2, z_1, z_2) = \begin{cases} h(x_1, x_2), & if\ z_1 = A, z_2 = A \\ 0.4h(x_1, x_2), & if\ z_1 = A, z_2 = B \\ -0.75h(x_1, x_2) + 3, & if\ z_1 = B, z_2 = A \\ -0.5h(x_1, x_2) + 1.4, & if\ z_1 = B, z_2 = B \end{cases}$$
$$s.t.\ c_1(x_1, x_2, z_1, z_2) = \begin{cases} x_1x_2 - 0.4, & if\ z_1 = A, z_2 = A \\ 1.5x_1x_2 - 0.4, & if\ z1 = A, z_2 = B \\ 1.5x_1x_2 - 0.2, & if\ z_1 = B, z_2 = A \\ 1.2x_1x_2 - 0.3, & if\ z_1 = B, z_2 = B \end{cases}$$
$$x_1, x_2 \in [0, 1];\ z_1, z_2 \in \{A, B\}$$

where

$$h(x_1, x_2) = \frac{1}{51.9496}\left[\left(\left(15x_2 - \frac{5}{4\pi^2}(15x_1 - 5)^2 + \frac{5}{\pi}(15x_1 - 5) - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(15x_1 - 5) + 10\right) - 54.8104\right]$$

Note that h defines the Branin Hoo function. The function has a constrained minimum $f(\mathbf{x}^*) = -0.814299$ at $\mathbf{x}^* = (1.0, 0.4, A, A)$.

A.2.2 Func-3C

The unconstrained Func-3C problem [119, 127, 128] has enjoyed some popularity as benchmark for BO in mixed domain. We extend this benchmark with a simple disk constraint to transform the problem into a constrained optimisation problem. The resulting problem is defined by:

$$\min_{x_1,x_2,z_1,z_2,z_3 \in \mathcal{X}} f(x_1, x_2, z_1, z_2, z_3) = A(x_1, x_2, z_1) + A(x_1, x_2, z_2) + B(x_1, x_2, z_3)$$
$$s.t.\ c_1(x_1, x_2, z_1, z_2, z_3) = \sum_{i=1}^{5} x_i^2 - 1 \leq 0$$

where

$$
\begin{aligned}
A(x_1, x_2, z) &= \begin{cases} \frac{1}{300} R(x_1, x_2), & if z = 0 \\ \frac{1}{10} S(x_1, x_2), & if z = 1 \\ \frac{1}{50} B(x_1, x_2), & else \end{cases} \\
B(x_1, x_2, z) &= \begin{cases} \frac{1}{2} S(x_1, x_2), & if z = 0 \\ \frac{1}{500} R(x_1, x_2), & else \end{cases} \\
R(x, y) &= (1 - x)^2 + 100(y - x^2)^2 \\
S(x, y) &= \left(4 - 2.1x^2 + \frac{x^4}{3}\right) x^2 + xy + (4y^2 - 4)y^2 \\
B(x, y) &= (1.5 - x + xy)^2 + \left(2.25 - x + xy^2\right)^2 + \left(2.625 - x + xy^3\right)^2 \\
x_1, x_2 &\in [-1, 1]; z_1 \in \{0, 1, 2\}, z_2 \in \{0, 1, 2, 3, 4\}, z_3 \in \{0, 1\}
\end{aligned}
$$

Note that R, S, B define the 2D versions of the Rosenbrock function [129], Six-Hump Camel function [154], and Beale's function [155] respectively. The function has a global optimum of $f(\mathbf{x}^*) = -0.23144967$ at $\mathbf{x}^* = (-0.116834, 0.591213, 0, 0, 0)$.

A.2.3 XGBoost tuning

The XGBoost tuning problem [18, 127] minimises the training error of an XG-Boost GBRT ensemble [156]. The objective function measures the model fit on the Iris dataset [157]. Originally, this problem tests classification accuracy on the Boston Housing data [135], but due to the ethical problems that are associated with this data set (Section 7.1), we test the model on the Iris data. To reliable estimate the model performance, we estimate the model accuracy as mean across all runs of a 5-fold cross-validation. The constraint function restricts the elapsed time of the cross-validation to be less or equal to three seconds. The resulting problem is defined by:

$$
\begin{aligned}
&\min_{\mathbf{params} \in \mathcal{X}} f(\mathbf{params}) = 1 - \text{accuracy}(\text{model}_{\mathbf{params}}) \\
&\quad s.t.\ c_1(\mathbf{params}) = \text{elapsed time in sec}(\mathbf{params}) - 3 \leq 0
\end{aligned}
$$

where

$$
\text{accuracy}(\text{model}_{\mathbf{params}}) = \text{Mean accuracy across 5-fold cross-validation}
$$

We use the Microsoft XGBoost implementation for Python [1]. The parameters that are tuned in this problem are listed in Table A.1.

The original problem in Daxberger *et al.* [18] does not include any constraints. We added the time constraints to emulate a real-world black-box optimisation problem with unknown constraint function. Although the size of the feasible region is difficult to estimate, we observed training times in the range of $0.01-15$ seconds. We measure the wall-clock time of the entire cross validation run. Furthermore, the original problem included only binary categorical variables. To make the search more challenging along these dimensions, we included a third value for the `n_estimators` and `max_depth` parameters. Since this modification increases the search space considerably, we decrease the range of the `alpha` parameter from $[0.000985, 1009.209690]$ to $[0.0, 109.209690]$.

Since the feasible region is defined based on the training time of an XGBoost model, the

actual feasible region may vary from machine to machine on which the experiment is conducted. Consequently, we want to highlight that the results on this particular benchmark problem may be different when repeated. In particular, it may be required to adjust the time constraint on better hardware to prevent the task from becoming too easy.

The function has a theoretical minimum of $f(\mathbf{x}^*) = 0.0$, as the classification accuracy can be zero. However, we do not know where this minimum lies.

Parameter	Description	Range
`alpha`	L1 regularisation term on weights.	$[0.0, 109.209690]$
`lambda`	L2 regularisation term on weights.	$[0.000978, 99.020893]$
`colsample_bylevel`	Subsample ratio of columns for each level.	$[0.046776, 1.0]$
`colsample_bytree`	Subsample ratio of columns when constructing each tree.	$[0.062528, 1.0]$
`learning_rate`	Boosting learning rate (xgb's "eta").	$[0.000979, 0.995686]$
`min_child_weight`	Minimum sum of instance weight(hessian) needed in a child.	$[0.5, 127.042806]$
`subsample`	Subsample ratio of the training instance.	$[0.5, 1.0]$
`booster`	Which booster to use.	$\{gbtree,\ gblinear\}$
`n_estimators`	Number of boosting rounds.	$\{3, 100, 5000\}$
`max_depth`	Maximum tree depth for base learners.	$\{1, 10, 15\}$

Table A.1: XGBoost parameters. The parameters of the XGBoost model that are optimised. The parameter descriptions are taken directly from the documentation of the Microsoft XGBoost documentation [2].

[2] `https://xgboost.readthedocs.io/en/stable/python/python_api.html`

Appendix B

Gaussian Process implementation

Every GP in this work is implemented using GPFlow [132]. While other software for Bayesian Optimisation with GPs exist, e.g. BoTorch [158], we found that the implementation of the CWEI acquisition function in the latter software is ill-defined if no feasible point was found previously.

All GP in this work use the automatic relevance determination (ARD) Matérn 5/2 kernel with constant mean function, a signal variance σ_f^2 and d lengthscale parameters ℓ_i, $1 \leq i \leq d$. The resulting kernel covariance function is:

$$k_{M52}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \left(1 + \sqrt{5r^2(\mathbf{x}, \mathbf{x}') + \frac{5}{3}r^2(\mathbf{x}, \mathbf{x}')}\right) \exp\left(-\sqrt{5r^2(\mathbf{x}, \mathbf{x}')}\right)$$

where

$$r^2(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{d} \frac{(x_i - x'_i)^2}{\ell_i^2}$$

The above kernel function has two types of hyperparameters: signal variance σ_f^2 and d lengthscale parameters ℓ_i. A third type of hyperparameter is the noise variance σ_{noise}^2 of a GP. Following Eriksson *et al.* [131], we use the following bounds for these hyperparameters:

- Signal variance: $\sigma_f^2 \in [0.05, 20.0]$ with initial value 1.0
- Lengthscales: $\ell_i \in [0.005, 2.0]$ with initial value 0.5, $\forall i \in \{1, \ldots, d\}$
- Noise variance: $\sigma_{noise}^2 \in [0.0005, 0.2]$ with initial value 0.005

For numerical stability, we transform the input data X to lie within a hypercube $[0, 1]^d$, and the observed objective function values y are standardised to have mean zero and variance one. Additionally, we transform the observed constraint function values using the bilog transformation that is described by Eriksson *et al.* [131]:

$$bilog(x) \coloneqq sign(x) \log(1 + |x|)$$

We apply the transformation twice to each entry in the label vector y before we fit the GP surrogate models.

Appendix C

Software usage guide

Instructions on how to use the software can be found in our README[1]. Additionally, we created a notebook[2] with an example use case to help users.

[1] https://github.com/cornelius-braun/constrained-bo-trees/blob/main/README.md
[2] https://github.com/cornelius-braun/constrained-bo-trees/blob/main/example_usage.ipynb