Abstract

This thesis discusses the mechanism of Bayesian Optimisation from the view of Decision Theory. It will mainly focus on showing how one can apply some of the most basic, commonly known criteria from decision theory to some basic optimising problems of Bayesian Optimisation (single point proposal, not noisy, not multi-objective, not mixed-space optimisation). We will do that by suggesting a number of new acquisition functions inspired by a few decision criteria such as: the Maximax criterion, the Maximin criterion, the Minimax-Regret criterion, the Pessimism-Optimism Index criterion of Hurwicz and the Hodges-Lehmann criterion. The infill function from Minimax-Regret-criterion will also be studied and it shows potential for noisy BO problems. These new acquisition functions will then be used for optimising some artificial test functions, their performances will be benchmarked and compared with the common ones, e.g. Expected Improvement (EI) and Lower Confidence Bound (LCB). We rank the results and come to conclusion that while EI is still the most suitable infill function for these theoretical optimisation problems, the infill function from Hurwicz criterion also has great potential.

Key words: bayesian optimisation, expected improvement, global optimisation, decision theory, maximin, minimax, decision under uncertainty

Contents

List of Notations

Symbol	Description
$\mathcal{X} \subset \mathbb{R}^d$	input space of dimensionality d
$\boldsymbol{x} \in \mathcal{X}$	input value, design point
x^*	location of the next test point or the optimal point
x^+	location of the most potential point so far
$\mathcal{H}$	function space
$f_{1:t} \in \mathcal{H}$	realisation of target function from $f(x_1)$ to $f(x_t)$
$\mathcal{D}_{1:t} = \{(x_i, f_i)\}_{i=1}^t$	set of observations (data) up to time t
$\sigma_t^2(x_{t+1})$	predictive variance (uncertainty) of the function at new test point x_{t+1}, given observed data $\mathcal{D}_{1:t}$
n	number of unique design locations
Θ	parameter space or set of states of nature
$\theta \in \Theta$	weights/parameter vector
p	dimensions of parameter vector θ
$\mathcal{I}_p$	unit matrix
$\mathcal{GP}$	Gaussian distribution over function (Gaussian Process)
K	Covariance matrix
$k(.,.)$	kernel function
l	length-scale (hyperparameter in kernel function $k(.,.)$)
$\epsilon \sim \mathcal{N}(0, \sigma^2)$	zero-mean Gaussian noise with variance σ^2
$\hat{f}(\boldsymbol{x}) \in \mathcal{H}$	estimated surrogate model for f
$\hat{s}^2(\boldsymbol{x})$	estimated variance of $\hat{f}(\boldsymbol{x})$
$\phi : \mathbb{R} \to \mathbb{R}^+, \quad \Phi : \mathbb{R} \to [0,1]$	standard normal density and distribution function
$\mathbb{A}$	set of acts
$u(.) : (\mathbb{A} \times \Theta) \to \mathbb{R}, \quad (a, \theta) \mapsto u(a, \theta)$	utility function
$l(.) : (\mathbb{A} \times \Theta) \to \mathbb{R}, \quad (a, \theta) \mapsto l(a, \theta)$	loss function
$r(.) : (\mathbb{A} \times \Theta) \to \mathbb{R}, \quad (a, \theta) \mapsto r(a, \theta)$	risk/regret function
$d(x, x') : \mathbb{R}^2 \to \mathbb{R}, \quad (x, x') \mapsto d(x, x') = \|\|x - x'\|\|$	Euclidean distance function

Note: Without loss of generality, objective functions are minimised throughout this work.

List of Figures

List of Tables

List of Algorithms

1 Introduction

This thesis focuses on showing the connection between Decision Theory and Bayesian Optimisation (BO), in other words, how one could use Decision Theory to design effective BO strategies for optimising mathematical (in the scope of this thesis: theoretic) problems.

A good way to open this Introduction chapter is to briefly explain the idea of BO and the basic characteristics of Decision Theory (DT). BO is widely used as an effective way to find solutions for the so-called design problems. One example of design problems would be in **applied science**, where engineers design different components of machines for more effective execution; or where scientists design experiments to learn more about some particular phenomena. Another example would be **software design problems**: how one develops software so that it would run smoothly in real-time with the least computational power. All of these design problems often involve **a lot of options**, which are often **high dimensional**, and usually also have **interactions** with each other, which would make it challenging to find the best solution for each particular problem.

The success stories of applications of BO are varied: Brochu et al. (2010) presents extensions of BO with experiments with **hierarchical reinforcement learning**; Snoek et al. (2012) and Thornton et al. (2013) show different ways one could apply BO into **tuning the hyperparameters** in the field of machine learning. In robotics: Lizotte et al. (2007) discusses about **Automatic Gait Optimisation** using Gaussian Process Regression while Martinez-Cantin et al. (2007) explains the Active Policy Learning for Robot Planning.

Similar to BO's objective, which is to solve optimisation problems, the field of Decision Theory (DT) aims to study decision problems in order to create **strategies** to optimise the outcome (e.g.: maximise the profit or minimise the loss). We face problems every day when we need to decide for some actions - one typical example would be: whether one should bring umbrella to work today? Bringing umbrella could feel uncomfortable, but if it rained then it would not be too great without one. More sophisticated examples would be: Majidi et al. (2019) reviewing the use of information gap decision theory for modeling uncertainty in **energy and power systems** or Hayes et al. (2022) discussing the application of multi-objective decision-making methods to complex problems, offering insights into practical implementations (**reinforcement learning**).

We will discuss the most important components of a Bayesian Optimisation, then how to choose these components based on some criteria from decision theory. The thesis is hence structured as follows: in Chapter 2, we discuss all the crucial components making a typical **Bayesian Optimisation** strategy. Chapter 3 is where the foundation of **Decision Theory** will be explained in details (to some extent). Based on that, we will come up with some new methods (infill functions) combining knowledge in both fields in Chapter 4. In Chapter 5, we put these methods into use by benchmarking their usages in an experiment. Finally, Chapter 6 is preserved for concluding remarks.

2 Fundamental principles of Sequential Model-Based Optimisation

2.1 Bayesian Optimisation

Bayesian Optimisation (BO) has quite a long history, which dates back to the early 1930s, when Thompson's work pointed out the exploration-exploitation trade-off, a crucial idea when working with BO (see Thompson (1933a), Shahriari et al. (2016a)). The term BO was first used in the 1970s by Mockus et al. (2014).

So what exactly should one know about the method of BO to understand this thesis? Mathematically, BO tries to find a global minimiser (or maximiser) of an (closed form unknown) objective function. In this thesis, we will focus on finding the **global (!) minimum** of a function, hence the general mathematical expression is stated:

$$x^* = \arg\min_{x \in \mathcal{X}} f(x)$$

with

$$\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \mathcal{X}_3 \times ...\mathcal{X}_d$$

a **d**-dimensional design space of interest. If one wants to maximise a function (let's call it g) instead, the problem can now be regarded as the minimisation of the transformed function

$$f(x) = -g(x)$$

.

Here in the expression, one should understand that $\mathcal{X}$ could take values of type numerical, as well as of type categorical, but in this thesis, $\mathcal{X}_i$ is defined as a compact subset of real numbers set $\mathbb{R}$. In this thesis, and typically $\mathcal{X}$ is a hyperrectangle: $x \in \mathbb{R}^d : ai \leq xi \leq bi$. That would mean that the value space for x is bounded, which would make our job of finding the optimum a bit easier. There are also a couple of properties and assumptions that should be mentioned here (also see Frazier (2018)):

- The target function f is assumed to be black-box, which has no closed form (problematic for finding extremum), hence no first- or second-order derivatives - gradient descent, Newtons method, or quasi-Newton methods are not plausible here.
- However, it could be evaluated at any query point x in the domain $\mathcal{X}$, although evaluations are normally cost-sensitive - the number of available evaluations should be limited, since each evaluation normally takes a lot of time/money.
- The dimension d should not be too large. The dimension of d smaller than 20 is suggested by Frazier (2018) in order for BO to work best.

One more thing to mention here is, that in many scenarios, the evaluations' output are stochastic (corrupted with noises), but in this thesis we will only consider the cases without noises.

Since it is expensive to evaluate these functions, the goal is to find the optimum with the least amount of points to be evaluated. A **sequential model-based** approach to solving this stated problem is **BO** (see Hutter et al. (2011)). But first, what should we understand under a **sequential approach**?

Generally, a sequential search algorithm runs N queries, at each query, e.g.: at iteration n it selects a x_{n+1}, evaluate function f to get y_{n+1}, and in the very end it selects the best possible evaluation of the optimiser. Then it comes to the question of: what is **sequential model-based optimisation** (SMBO)? As the name suggests, SMBO methods would fit a regression **model** (response surface model) that later will be use for optimisation (see Jones (2001a)).

A SMBO strategy is a combination of many components: designing each of these components can be done separately and then put together to set up a strategy that works best for a specific optimisation problem. This is considered as an **advantage** of using the SMBO.

More exactly, we will have two main components: a **probalistic surrogate model** and **an acquisition function**. We apply the **Bayesian Theorem** to create SMBO strategy: over the possible candidate objective functions we prescribe a prior belief (here the initial surrogate model) and then sequentially refine this model as data are observed via Bayesian posterior updating. The Bayesian posterior represents our updated beliefs in the candidate objective function we are optimising. Details will be covered in chapter 2.2.

Alongside this probabilistic model, we can sequentially induce **acquisition functions** which leverage the uncertainty in the posterior to manage a trade-off between exploration and exploitation. More on this will be discussed in chapter 2.3.

2.2 Surrogate Model

Surrogate models, as the name might suggest, are used as a cheaper alternative to investigate / learn the characteristics of the sample at hand, since target black-box functions are usually considered cost sensitive. Now, in order to fit a surrogate model that reflects our belief of what the black-box (target) function should look like, we want to apply the principle of Bayesian Linear Regression, where we assume not only that our target function follows a certain distribution, but also the **parameters** (weights) from its regression function will follow a distribution. With that in mind, we fit a surrogate model with the available **initial sample points**, then **update** this model every time we get new sample points (refining the model). More details on this should be discussed later in chapter 2.2.1.

For the sake of completeness, it should be mentioned that there are also **parametric** surrogate models (such as the Beta-Bernoulli bandit model using Thompson sampling or generalised linear models, see Shahriari et al. (2016a) section II). However, the **focus of**

this thesis lies on the **non-parametric** ones, in particular Gaussian Process (GP) Models. **Random Forests** would also be a common choice, especially when sample points contain a mixture of categorical and numerical features; we would not discuss this topic further - for more information, one could read Guo et al. (2023) and Ma et al. (2023). Also in these papers Li et al. (2024), Snoek et al. (2015), Springenberg et al. (2016), the use of **neural networks** is proposed as an alternative to GPs to model distributions over functions.

2.2.1 Gaussian Process Regression

Throughout this thesis we will use Gaussian Process (the Regression one) (GPR) as our only surrogate model, hence a good understanding of this subject will be crucial for the later chapters. The explanation stated under is inspired by this lecture from Bischl (2019).

The idea of GPR starts from Linear Regression, more exactly the Bayesian Linear Regression, where in our infamous base model $y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)}$ (with all its important distributional assumptions), we will also assume that the outcome parameter vector θ (the weights) is stochastic (instead of a constant) and follows a certain distribution. We can put this mathematically:

$$\theta \sim \mathcal{N}(0, \tau^2 \mathcal{I}_p)$$

with $\mathcal{I}_p$ the unit matrix and p the number of dimensions of the vector θ. With the well-known distributional assumption $y|X, \theta \sim \mathcal{N}(X^T\theta, \sigma^2 \mathcal{I}_n)$ (with $\mathcal{I}_p$ the unit matrix and n the number of observations), using the Bayesian Theorem $P(\theta|X, y) = \frac{P(y,X|\theta)P(\theta)}{P(y,X)}$, we can obtain a posterior estimate of $\theta|y, X$.

This approach should be very intuitive, as we will get a better understanding (better estimation) of θ every time we observe new values from our sample space $\mathcal{X}$, which is also called online update (as we update our understanding along the way). Also, from this we will not only get the point estimate of θ, but also an understanding of the uncertainty of this very point estimation. We want to mention very briefly (more details on this could be read from Rasmussen and Williams (2006)) one more special feature of this approach with these very assumptions: the posterior $\theta|y, X$ would also follow a normal distribution. These should come in handy when it comes to inference later on.

From this point of view with the Bayesian Linear Model (which is usually called weight-space view), we move one step forward into the so-called function-space view. Here, instead of searching for the θ (from the parameter space Θ) that fits our target function $f(x|\theta)$ the most, we search for all the possible $f(x|\theta)$ from the function space $\mathcal{H}$ that fits our sample of points the most.

We will need to assume that these candidate functions follow a certain distribution (in our case of GPR a normal distribution, just as the weight-space-view approach above). Formally Bischl (2019) **defines** the Gaussian Process Regresison as:

A Gaussian Process (Regression) is stochastic process (a collection of random variables collected by time or space), and a function f(x) is generated by a Gaussian process $\mathcal{G}$ if for any finite set of inputs $x^{(1)}, ..., x^{(n)}$, the corresponding vector of function values has a multivariate normal distribution (MVN):

$$f = (f^{(1)}, ..., f^{(n)}) \sim \mathcal{N}(\mathbf{m}, K)$$

with

$$\mathbf{m} := (m(x^{(i)}))_i, K := (k(x^{(i)}, x^{(j)}))_{i,j}$$

where m(x) being the mean function and k(x, x') being the covariance function.

Just as a Gaussian distribution is a distribution over a random variable, the Gaussian Process is completely specified by its mean function m(.) and covariance function K(.,.), which is defined as such:

$$m(x) = \mathbb{E}[f(x)]$$

$$k(x, x') = \mathbb{E}[(f(x) - \mathbb{E}[f(x)])(f(x') - \mathbb{E}[f(x')])]$$

so that

$$f(x) \sim \mathcal{GP}(\mu(x), k(x, x'))$$

If one have no better understanding of the to-be-assessed sample, one would normally assume: **prior mean is a zero function**: $\mu(x) = 0$; another assumption for the prior mean could be found here Martinez-Cantin et al. (2009).

The kernel or covariance function is chosen so that points x_i, x_j that are **closer** in the input space have a **larger** positive correlation, encoding the belief that they should have more similar function values than points that are far apart. The kernel should also have the property that the resulting covariance matrix is positive semi-definite (p.s.d.), regardless of the collection of points chosen. For this, there are a couple of common choices, one of which would be the squared exponential function:

$$k(x_i, x_j) = exp(-\frac{1}{2l^2}||x_i - x_j||^2)$$

with $l = 1$ so called **length-scale** (More on this in next section 2.2.2).

This is one very popular and intuitive kernel, as this function converges to 1 when the two values of x_i and x_j get closer together and to zero otherwise. We need this as a Bayesian optimisation method will converge to the optimum if the distance from one point to the nearest observation is zero, according to Mockus (1994) (also understood as conditional variance converges to zero). Other mentioned condition for the convergence would be, that the acquisition function is continuous and approximately minimizes the risk (the expected deviation from the global minimum at a fixed point x). Also, more sophisticated kernels will be discussed later on.

Having the basics of GP, what we are interested now is how to apply this to our main goal of Bayesian Optimisation, more exactly how to fit one. We start constructing our surrogate function by sampling from a prior, as any Bayesian method depends on a prior distribution, by definition. We choose $\{x_{1:t}\}$ and evaluate the mean at each point x_i and covariance function at each pair x_i and x_j, so in the end we sampled the pairs $\{x_{1:t}, f_{1:t}\}$. How we sample these points effectively would be one interesting topic because of its importance: in case too few points are chosen/ if the points do not cover the sample space $\mathcal{X}$ well, the fit of f may be poor and thus points proposed based on f may be suboptimal. Meanwhile, if one sample way too many points for the initial design, there would be not much left for further investigation, as the budget is limited as we mentioned in the beginning. In this thesis we can only briefly mentioned three approaches: we can generate designs either completely at random, coarse grid designs or by using space-filling Latin Hypercube Designs. We will choose the last approach for our experiment later on in this thesis. One could read McKay et al. (1979a) for more information.

With the zero mean function as the prior mean, we assume that the values of the surrogate function are drawn according to a MVN distribution $N(0.\mathbf{K})$, with $\mathbf{K}$ the kernel/ covariance matrix:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_t, \mathbf{x}_1) & \cdots & k(\mathbf{x}_t, \mathbf{x}_t) \end{bmatrix}$$

Now that we got the initial prior surrogate function, we would wish to refine this function every time we sample a new position from the sample space of our target model. Suppose that we sampled this new position $\{x_{t+1}, f_{t+1}\}$, then we would get this distribution that $f_{1:t}$ and f_{t+1} are jointly Gaussian, applying the properties of Gaussian Processes:

$$\begin{bmatrix} f_{1:t} \\ f_{t+1} \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \mathbf{K} & \mathbf{k} \\ \mathbf{k}^\top & k(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}) \end{bmatrix}\right),$$

with k $= \begin{bmatrix} k(\mathbf{x}_{t+1}, \mathbf{x}_1) & k(\mathbf{x}_{t+1}, \mathbf{x}_2) & \cdots & k(\mathbf{x}_{t+1}, \mathbf{x}_t) \end{bmatrix}$

Here we want to calculate the posterior distribution function (an updated surrogate model). We may then compute the conditional distribution of f(x) given these observations using Bayesian rule (more details see Rasmussen and Williams (2005)):

$$P(f_{t+1}|\mathcal{D}_{1:t}, \mathbf{x}_{t+1}) = \mathcal{N}\left(\mu_t(\mathbf{x}_{t+1}), \sigma_t^2(\mathbf{x}_{t+1})\right)$$

where

$$\mu_t(\mathbf{x}_{t+1}) = \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{f}_{1:t}$$
$$\sigma_t^2(\mathbf{x}_{t+1}) = k(\mathbf{x}_{t+1}, \mathbf{x}_{t+1}) - \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{k}.$$

are the sufficient statistics of the predictive posterior distribution: $P(f_{t+1}|\mathcal{D}_{1:t}, \mathbf{x}_{t+1})$.

Let us consider an example (also see Bischl (2019)): assume we observed one single training point x = -0.5 with f(-0.5) = 1 and we want to predict at the test point x* = 0.5. We also assume a zero mean with k(x,x') = $\exp(-\frac{1}{2}||x-x'||)$, then we receive this covariance matrix:

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} 1 & 0.61 \\ 0.61 & 1 \end{bmatrix}\right).$$

Let us assume that we observe the point $f(x) = 1$. We can compute the posterior distribution:

$$\begin{aligned} f_*|x_*, x, f &\sim \mathcal{N}\left(k_*^\top K^{-1} f,\ k_{**} - k_*^\top K^{-1} k_*\right) \\ &\sim \mathcal{N}\left(0.61 \cdot 1^{-1} \cdot 1,\ 1 - (0.61)^2\right) \\ &\sim \mathcal{N}\left(0.61,\ 1 - 0.3721\right) \\ &\sim \mathcal{N}\left(0.61,\ 0.6279\right) \end{aligned}$$

We understand here the result of f(x_*) = 0.61 is the Maximum a Posteriori (MAP) estimate.

From this we can generalise to calculating the posterior giving multiple unobserved test points, where f_* does not only take the value f_{t+1} but should be $f_* = [f(f(x_*^{(1)}), ..., f(x_*^{(m)})]$ and

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \mathbf{K} & \mathbf{K}_* \\ \mathbf{K}_*^\top & \mathbf{K}_{**} \end{bmatrix}\right),$$

where $\mathbf{K}_* = (k(x^{(i)}, x_*^{(j)}))_{i,j}$ and $\mathbf{K}_{**} = (k(x_*^{(i)}, x_*^{(j)}))_{i,j}$

Analogously, we receive our formula for the posterior distribution at the end:

$$f_*|X_*, X, f \sim \mathcal{N}\left(K_*^\top K^{-1} f,\ K_{**} - K_*^\top K^{-1} K_*\right)$$

With this we can keep the loop of sampling new points from the posterior and updating new posterior distribution running for as long as our budget allows, and this very idea should be exactly how we will fit our surrogate model for our Bayesian Optimisation. Gaussian Process Regression (GPR) with the metric of uncertainty is very suitable for SMBO (more on this in chapter 2.3). The predictions of the model are also considered easy to compute, as the number of query points is realtive small (in the sequential decision making setting).

Training a Gaussian Process Regression:
At this point, we have already been through the most important part of utilising a surrogate model. We know that for this task we would need suitable covariance functions, so choosing the proper kernel function and getting all of its hyperparameters (e.g.: length-scale l in our example kernel of squared exponential function) right for the specific task is crucial. According to Bischl (2019), we could learn the numerical hyperparameters of a selected covariance function **during** GP training, which is a very nice property of GPs. This fitting should be quite similar to fitting a Linear Regression: let us go back to our

assumptions that y = f(x) + ϵ with $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and $f(x) \sim \mathcal{GP}(0, k(x, x'|\theta))$. Then we could rewrite so that $y \sim \mathcal{N}(0, \mathbf{K}+\sigma^2\mathbf{I})$, and from here we get the marginal log-likelihood:

$$\log p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}) = \log \left[(2\pi)^{-n/2} |\mathbf{K} + \sigma^2 \mathbf{I}|^{-1/2} \exp\left(-\frac{1}{2} \mathbf{y}^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y} \right) \right]$$

$$= -\frac{1}{2} \mathbf{y}^\top (\mathbf{K} + \sigma^2 \mathbf{I})^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K} + \sigma^2 \mathbf{I}| - \frac{n}{2} \log 2\pi.$$

with $\boldsymbol{\theta}$ denoting the parameters of the covariance function (i.e. the hyperparameters).

From here we would be able to select the hyperparameters by maximising the marginal likelihood, more exactly by calculating the partial derivatives of the likelihood w.r.t. the hyperparameters:

$$\frac{\partial}{\partial \theta_j} \log p(\mathbf{y} \mid \mathbf{X}, \boldsymbol{\theta}) = \frac{\partial}{\partial \theta_j} \left(-\frac{1}{2} \mathbf{y}^\top \mathbf{K}_y^{-1} \mathbf{y} - \frac{1}{2} \log |\mathbf{K}_y| - \frac{n}{2} \log 2\pi \right)$$

with

$$\frac{\partial}{\partial \theta_j} \mathbf{K}^{-1} = -\mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial \theta_j} \mathbf{K}^{-1} \quad \text{and} \quad \frac{\partial}{\partial \theta} \log |\mathbf{K}| = \operatorname{tr}\left(\mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial \theta} \right)$$

we get

$$= \frac{1}{2} \mathbf{y}^\top \mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial \theta_j} \mathbf{K}^{-1} \mathbf{y} - \frac{1}{2} \operatorname{tr}\left(\mathbf{K}^{-1} \frac{\partial \mathbf{K}}{\partial \theta} \right)$$

$$= \frac{1}{2} \operatorname{tr}\left(\left(\mathbf{K}^{-1} \mathbf{y} \mathbf{y}^\top \mathbf{K}^{-1} - \mathbf{K}^{-1} \right) \frac{\partial \mathbf{K}}{\partial \theta_j} \right)$$

We end this subchapter by mentioning the calculation complexity of GPR: from the formula we just mentioned above: the training of a GP scales **cubically** with the number of **observations**, as the computation overhead of inverting the kernel $\mathbf{K}$ (more precisely decomposing it) could take $\mathbf{O}(n^3)$ (where n is the size of the training sample) and thus create a **bottle neck**. That would also be one of the reason why it is **not** suitable for using BO for large datasets. A few ways have been suggested so far to make GP estimation work for large datasets; one example would be **focalised GP** from Wei et al. (2024). Otherwise, the computation of the partial derivatives requires only $O(n^2)$ time per hyperparameter, once the $\mathbf{K}^{-1}$ is known.

2.2.2 More on kernels/ covariance functions

What does a kernel tell us? Basically, a kernel shows the **structure** of the response functions we can fit, especially the smoothness of the drawn samples. While the prior mean provides a possible offset, the kernel function controls the smoothness and amplitude of the GP samples.

In this thesis, we would focus on **Matérn kernels**, which are a very flexible class of stationary kernels, whose formulas depend only on distances between points. Stationary kernels do not change their properties in case of shifting, a.k.a. translationally invariant: k(x,x+d) =k(0,d)=k(d)

Generally, the Matérn covariance between two points is given by:

$$k_{\text{Mátérn}}(\mathbf{x}, \mathbf{x}') == \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \frac{d}{l} \right)^{\nu} K_{\nu} \left(\sqrt{2\nu} \frac{d}{l} \right)$$

where Γ is the gamma function, $\mathbf{K}_\nu$ is the **modified Bessel function of the second kind**(for more information one can read Cuingnet (2023)), $d = ||x - x'||$ the distance between to query points (in our thesis the euclidean distance), l the length-scale, ν are positive parameters of the covariance and σ^2 (a hyperparameter) serves as a scaling factor that determines the magnitude of the kernel function. More on length-scale l (according to Bischl (2019)): the characteristic length-scale l shows how far one need to move in input space $\mathcal{X}$ for the function values to become uncorrelated - higher induces smoother functions, lower induces more wiggly functions. This explanation is intuitive: smaller l means that three abitrary points x and x' and x" that are "close" together (e.g.: w.r.t. Euclidean distance) can have three drastically different values of the corresponding f(x), f(x') and f(x"), which makes the line connecting these three points change its direction drastically from one point to another, and makes it more "wiggly" or "zig-zag".

A Gaussian process with Matérn covariance is $\lceil \nu \rceil - 1$ times differentiable in the mean-square sense.

Here are some of the most common Matérn kernels:

- For $\nu = \frac{1}{2}$:

$$k_{\text{Mátérn1}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{d}{l} \right),$$

- For $\nu = \frac{3}{2}$:

$$k_{\text{Mátérn3}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \left(1 + \frac{\sqrt{3}d}{l} \right) \exp\left(-\frac{\sqrt{3}d}{l} \right),$$

- For $\nu = \frac{5}{2}$:

$$k_{\text{Mátérn5}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \left(1 + \frac{\sqrt{5}d}{l} + \frac{5d^2}{3l^2} \right) \exp\left(-\frac{\sqrt{5}d}{l} \right).$$

- For $\nu \to \infty$, the Matérn covariance converges to the squared exponential covariance function as mentioned above:

$$k_{\text{SQ-EXP}}(\mathbf{x}, \mathbf{x}') = exp(-\frac{1}{2l^2}d^2)$$

In order to understand how one derive the result formula for the Matérn covariance for each ν, one should read Cuingnet (2023). Here we only need the concrete formula for $k_{\text{Mátérn3}}$ as we will apply it later in our experiment. Although the squared exponential covariance function is very popular, it is infinitely differentiable, which makes the corresponding GP too smooth, which is unrealistic for modeling most of the physical processes. The Matérn kernel 5/2 or 3/2 is a good compromise, balances out flexibility and smoothness.

Some more insights of each kernels could be found in this tutorial Brochu et al. (2010). In Figure 1 we see a visualization of the kernel profiles and samples from the corresponding priors and posteriors.

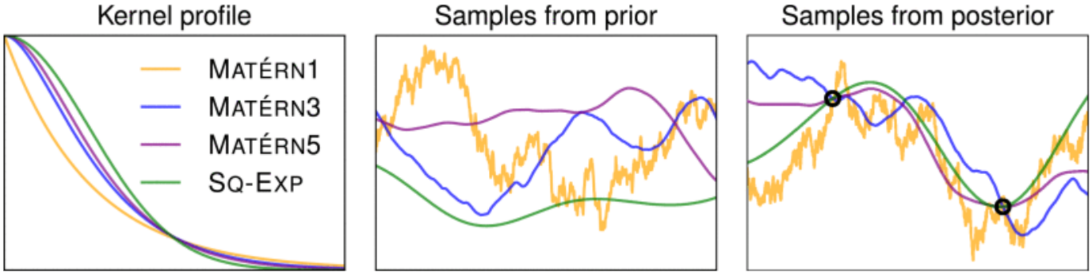

Figure 1: **Left:** common kernel profiles visualized, with the horizontal axis representing the distance $r > 0$. **Middle:** GPR prior functions with different kernels; with the squared exponential kernel, the output function is the smoothest of all four, and with the Matérn1 the roughest. **Right:** Updated posterior functions given two new points (black circles). **Source:** Brochu et al. (2010)

2.3 Acquisition Function

We have shown the statistical model representing our belief about the black-box function f at iteration n - what we need now is a policy for selecting the next query points. We usually do that using an "assessment" function, which normally is called **acquisition function**(AF) or **infill function**(IF). Given that our primary focus is on identifying the optimum with the least time we have to sample (since it is cost sensitive), it is salient that we construct our AF wisely.

The core idea behind most AF is, that the decision coming from infill functions will represent an automatic trade-off between exploration (where the objective function is very uncertain about its predictions (with high posterior variance/ standard deviation)) and exploitation (with the smallest posterior mean, in case of minimisation). The acquisition function is typically designed so that a high acquisition (high output of the AF) corresponds to potentially the best optimal position in the sample space (in our case: lowest value of acquisition function means optimal location to sample from; whether it is the location with the optimum (lowest value in our case), or the location with great uncertainty (where it might reveal the most information about our black-box target function)). We usually want to our AF to have a close form; it helps especially when it is differentiable, which makes AF-optimisation easier later on.

Let's start with the most naive infill function, the **pure mean** optimisation function. It could be expressed as: $\alpha_{mean}(x; \mathcal{D}_n) = \mu_n(x)$ and our goal is to find $x^* = \arg\min_{x \in \mathcal{D}} \alpha_{mean}(x|\mathcal{D})$. It is naive though straightforward, as we will choose the location where the posterior mean has the lowest value, without considering the posterior variance / standard deviation. It should be considered too naive/ not optimal, as we will very likely only find **local optima** instead of **global optimum**. Figure 2 demonstrates exactly the problem.

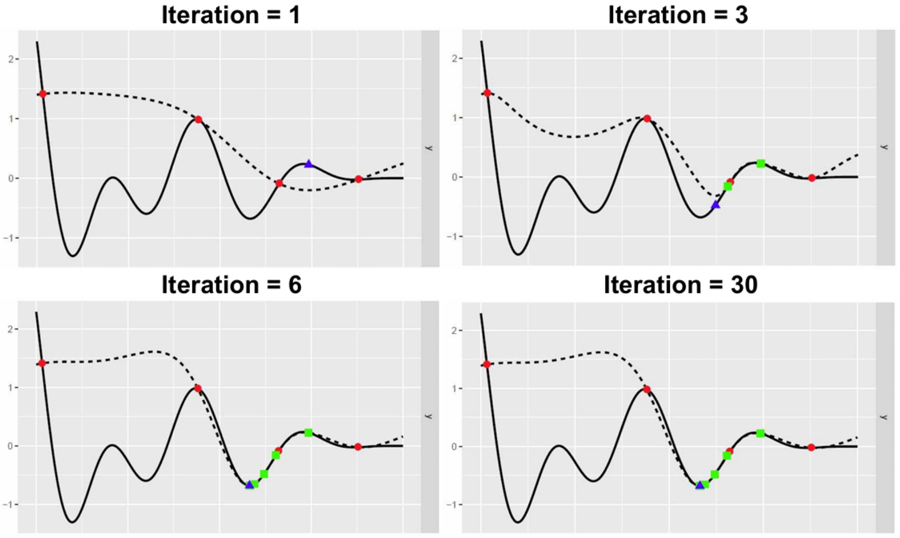

Figure 2: Here the red points are the initials sampled points for the initial surrogate function. The triangular blue point shows the location where to evaluate next. The green points are the newly evaluated location. The dotted line is our surrogate function and the black line is the black-box objective function. The problem of **local optimum** is visualised: although the location of our **global optimum** is on the left side, we are stuck on the right side of the plot for the whole 30 evaluations as the posterior mean always indicate that is where the optimum lies. Source: Mayer (2017)

With that example we want to emphasise the need of considering the posterior acquisition function as well. We will now discuss some of the most common infill functions in BO while categorising them according to their main goal. The work below is inspired by Brochu et al. (2010) and Shahriari et al. (2016b).

2.3.1 Improvement-based Acquisition Functions

These infill functions favour the locations where the value of objective function are likely to improve upon the incumbent optimum. There are two common acquisition functions we want to mention here, which are Probability of Improvement (PI) and Expected Improvement (EI).

The early work of Kushner (1964) on a ”new” method of locating the Maximum Point of an Arbitrary Multipeak Curve measure the probability that a new location of x will lead to an improvement upon the incumbent $f(x^+)$ (with $x^+ = \arg\min_{x_i \in x_{1:t}} f(x_i)$ the best location so far). The mathematical expression would be:

$$\alpha_{PI}(x; \mathcal{D}_n) = \mathcal{P}(f(x^+) > f(x)) = \Phi(\frac{f(x^+) - \mu_n(x)}{\sigma_n(x)})$$

with Φ the standard normal cumulative distribution function (CDF).

For this criterion, all improvements are treated equally (no matter large or small), so the function tends to select points that certainly (with less uncertainty/small $\sigma_n(x)$) lead to (maybe very small) improvement over points leading to (potential but less certain) larger improvement. From Jones (2001b): in general the heuristic used for an unknown target causes PI to exploit quite aggressively (while we actually want to balance between exploitation and exploration).

One trick to overcome this is to add another ξ (> 0, trade-off parameter) into the function (see Jones (2001b)):

$$\alpha_{PI}(x; \mathcal{D}_n) = \mathcal{P}(f(x^+) > f(x) + \xi) = \Phi(\frac{f(x^+) - \mu_n(x) - \xi}{\sigma_n(x)})$$

Figure 3 should visualise the idea above: So we fit a GP as our surrogate function with all the sample points (four black (cross) points and one red triangular point) and get the purple line as result. The reg triangular point is our incumbent at this moment, as the evaluation at this very point is the minimum (of all sample points). Assume we want to pick one of the two red dots to evaluate next, with the hope that we will find a new (global) minimum. We apply the PI for this task, where we will get that only the red dot on the left will bring improvement to our task of optimising, as intuitively only the green area shows a chance of improvement.

As mentioned, the Probability of Improvement would only consider if there is any improvement at any point, but not assess how much of the improvement it gives, which tends to lead to exploitation over exploration, which again leads to local optimum. This is when the alternative acquisition function of **Expected Improvement**(EI) comes into play, one that takes into account not only the probability of improvement, but the magnitude of the improvement a point can potentially yield. The idea could be expressed as such: the expected improvement of one point x or EI(x) = E(I(x)) which is the expected value of the improvement at point x, a.k.a. how much improvement could we expect at

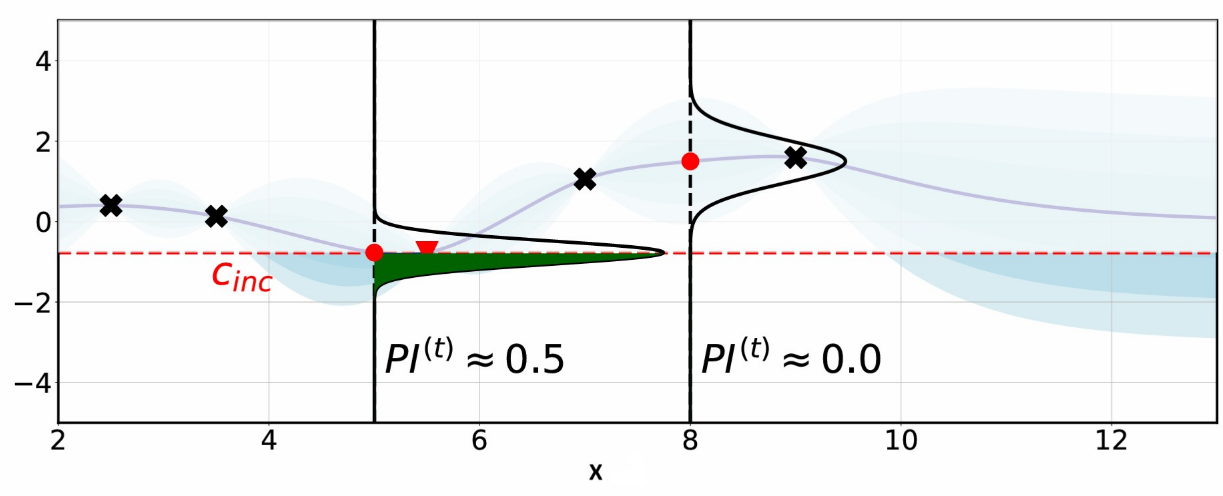

Figure 3: We fit a GP model with five sample points, then we want to evaluate at one of the two red dots: while the left red dot would bring improvement to our optimisation, the right red point does not show any chance of improvement.
Source: Hutter (n.d.)

this one particular point. For this thesis we consider the basic form of EI, which is the one-step positive improvement over the current incumbent, and rewrite this as:

$$a_{\text{EI}}(\mathbf{x}) = \mathbb{E}_y\big(\max\{f(x^+) - y, 0\}\big) = \int_{-\infty}^{\infty} \max\{f(x^+) - y, 0\} p(y)\, dy.$$

with y = f(x) and p(y) := $\mathcal{P}(Y|x, \mathcal{D}_n) = \mathcal{N}(\hat{f}(x), \hat{s}^2(x))$ and $f(x^+)$ our incumbent (minimum point at this very moment).

From here we want to get to a closed form for this formula. As $\max\{f(x^+) - y, 0\}$ only returns non-zero value when we get some improvement at a new point x ($y < f(x^+)$), we rewrite the formula:

$$a_{\text{EI}}(\mathbf{x}) = \int_{-\infty}^{\infty} (f(x^+) - y) p(y) dy$$

$$= \int_{-\infty}^{f(x^+)} (f(x^+) - y) \frac{1}{\sqrt{2\pi}\hat{s}(\mathbf{x})} \exp\left(- \frac{\left(y - \hat{f}(\mathbf{x})\right)^2}{2\hat{s}(\mathbf{x})^2} \right) dy$$

Here if we set:

$$u := \frac{y - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}, \quad \frac{du}{dy} = \frac{1}{\hat{s}(\mathbf{x})}, \quad z := \frac{f(x^+) - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}$$

we would get

$$= \int_{-\infty}^{z} (f(x^+) - \hat{f}(\mathbf{x}))\phi(u) du - \int_{-\infty}^{z} (u\hat{s}(\mathbf{x}))\phi(u) du$$

Now with:

$$\Phi(z) = \int_{-\infty}^{z} \phi(u) du$$

where $\phi()$ and $\Phi()$ denote the probability density function (PDF) and cumulative distribution function (CDF) of the **standard normal** distribution,

and the definition of density function:

$$\int_{-\infty}^{z} u\phi(u)du = -\phi(z).$$

we would end up with:

$$\alpha_{\mathrm{EI}}(\mathbf{x}) = z\hat{s}(\mathbf{x})\Phi(z) + \hat{s}(\mathbf{x})\phi(z)$$

$$= \left(f(x^+) - \hat{f}(\mathbf{x})\right)\Phi\left(\frac{f(x^+) - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\phi\left(\frac{f(x^+) - \hat{f}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right).$$

We can also add the trade-off parameter ξ like before with PI, which helps us to EI(.) in a generalized form which controls the trade-off between global search and local optimisation (exploration/exploitation) (see Lizotte (2008)):

$$\mathrm{EI}(\mathbf{x}) = \begin{cases} (f(x^+) - \hat{f}(\mathbf{x}) - \xi)\Phi(\frac{f(x^+)-\hat{f}(\mathbf{x})-\xi}{\hat{s}(\mathbf{x})}) + \hat{s}(\mathbf{x})\phi(\frac{f(x^+)-\hat{f}(\mathbf{x})-\xi}{\hat{s}(\mathbf{x})}) & \text{if } \hat{s}(\mathbf{x})) > 0 \\ 0 & \text{if } \hat{s}(\mathbf{x})) = 0 \end{cases}$$

We get our next point by maximising this equation. Lizotte (2008)'s experiments suggest that setting $\xi = 0.01$ (scaled by the signal variance if necessary) works well in almost all cases.

2.3.2 Confidence bound criteria

The idea could be tracked back to Dennis and John (1992) and Cox and John (1997), when they introduced an algorithm they call "Sequential Design for Optimization" (SDO). SDO selects points for evaluation based on the **Lower Confidence Bound** of the prediction site:

$$LCB(x) = \mu(x) - \lambda\sigma(x)$$

where $\lambda \geq 0$. By minimising this function we will get the next query point. There are theoretically derived guidelines for selecting and scheduling[1] the hyperparameter λ to minimise regret optimally (see Srinivas et al. (2012)). The main idea of this method should be quite easy to follow (see Figure 4): again we use the example above (by the explanation of PI) with 5 initial points, we get the purple line as the posterior mean of our GP, and the blue area as our Confidence Bound (we get this with $\mu(x) - 3\sigma(x)$, since each point on this line follows a normal distribution). As we want to find the next minimum, we are only interested in the lower part of this confidence bound, hence the name of **Lower Confidence Bound**. Then we could easily see that red cross would be our next most potential candidate point for a global minimum. We can also set the

[1] The confidence parameter α in LCB methods is often scheduled to decrease over time, starting with high exploration (large α) and gradually shifting toward exploitation (smaller α) as more data is gathered. This ensures that the optimisation process starts broadly exploring the space, but eventually focuses on refining promising regions.

confidence parameter α to any other value, in this case it is 3. Later we will come up with **alternative** acquisition functions inspired by this LCB strategy. There is also Upper Confidence Bound with similar idea of finding the maximum: we maximise this to find the point UCB(x) $= \mu(x) + \lambda\sigma(x)$

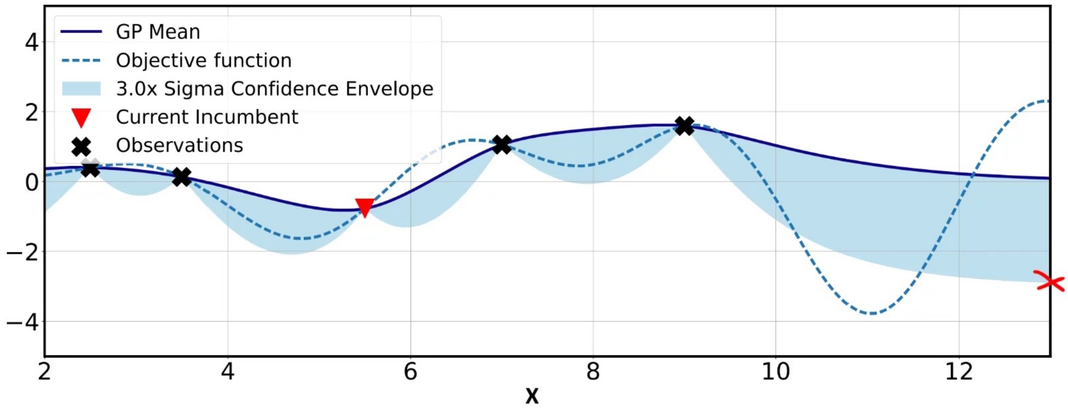

Figure 4: We fit a GP model with five sample points, then apply the LCB strategy with the confidence parameter of 3, where we will decide that the next query point should be the red cross. **Source:** Hutter (n.d.)

There are also many more common acquisition functions, which would not be discussed in this thesis, such as: Thompson Sampling (a randomized strategy sampling a **reward function** from the posterior and selects the arm with the highest simulated reward, see Thompson (1933b)), and Entropy Search (measures the **expected information gain** at point x and selects the point that offers the most information about the unknown x, see Villemonteix et al. (2007), Hennig and Schuler (2011)).

2.4 Infill Optimisation

As mentioned before, in order to get our next query point, we will need to maximise the infill (acquisition) function u() (or minimise it in case of LCB). Compared to the original black-box objective function, u() is usually considered to be inexpensive. One typical method for optimising this is DIRECT (see Jones et al. (1993)). Brochu et al. (2010) explains the idea neatly: DIRECT is a deterministic, derivative-free optimiser, using the existing samples of the black- box target function to decide how to proceed to **DI**vide the feasible space into finer **RECT**angles. ”A particular advantage in active learning applications is that DIRECT can be implemented as an any-time algorithm, so that as long as the user is doing something else, it continues to optimize, and when interrupted, the program can use the best results found to that point in time”.

Grid search and random search could also be used for this task: all in all, we simply want to evaluate a huge number of points in $\mathcal{X}$. However, in later chapter where we put Bayesian optimisation into use, we will use a slightly more advanced named **focus**

search, which basically shrinks the search space $\mathcal{X}$ and then applies random search. The pseudo-code is as follow in 1: in each iteration we apply random search to find one best point $x^*_{u,v}$ (obviously by using our acquisition function) - with this point we will shrink our initial search space in all of its dimensions accordingly, depending on the type of the variable in each dimension (numeric or categorical). Then we repeat this n times to find (one, in our case) next query point.

Algorithm 1 Infill Optimisation: Focus Search

```
Require: infill criterion $c : \mathcal{X} \to \mathbb{R}$, control parameters $n_{\text{restart}}, n_{\text{iters}}, n_{\text{points}}$
for $u \in \{1, \dots, n_{\text{restart}}\}$ do
    Set $\tilde{\mathcal{X}} = \mathcal{X}$
    for $v \in \{1, \dots, n_{\text{iters}}\}$ do
        Generate random design $\mathcal{D} \subset \tilde{\mathcal{X}}$ of size $n_{\text{points}}$
        Compute $x^*_{u,v} = (x^*_1, \dots, x^*_d) = \arg\min_{x \in \mathcal{D}} c(x)$
        Shrink $\tilde{\mathcal{X}}$ by focusing on $x^*$:
        for each search space dimension $\tilde{\mathcal{X}}_i$ in $\tilde{\mathcal{X}}$ do
            if $\tilde{\mathcal{X}}_i$ is numeric: $\tilde{\mathcal{X}}_i = [l_i, u_i]$ then
                $l_i = \max\{l_i, x^*_i - \frac{1}{4}(u_i - l_i)\}$
                $u_i = \min\{u_i, x^*_i + \frac{1}{4}(u_i - l_i)\}$
            end if
            if $\tilde{\mathcal{X}}_i$ is categorical: $\tilde{\mathcal{X}}_i = \{v_{i1}, \dots, v_{is}\}, s > 2$ then
                $\bar{x}_i$ = sample one category uniformly from $\tilde{\mathcal{X}}_i \setminus x^*_i$
                $\tilde{\mathcal{X}}_i = \tilde{\mathcal{X}}_i \setminus \bar{x}_i$
            end if
        end for
    end for
end for
return $x^* = \arg \min_{u \in \{1,\dots,n_{\text{restart}}\}, v \in \{1,\dots,n_{\text{iters}}\}} c(x^*_{u,v})$
```

3 Fundamental principles of Classical Decision Theory

The main goal of this chapter is to discuss all the most common and fundamental criteria of optimality of actions, which is one of the main topics of this thesis - understanding them will help us later on when we will apply their mechanics in order to create new alternative acquisition functions. But first we will need to go through all the basic terminologies of Decision Theory, and we will do that by studying the classical decision problem.

3.1 The classical decision problem

3.1.1 The basic model

Perhaps the most used example for introducing fundamentals of a classical decision problem is the Savage omelette problem (see *The foundations of statistics. By Leonard J. Savage, John Wiley & Sons, Inc., 1954, 294 pp* (1954)), a dilemma for one decision maker: one tries to make an omelette; there are five eggs already broken into a bowl and an unbroken egg number six. One has three options: either break the sixth egg into the same bowl, break open it into another saucer, or throw it away without checking it. For more context, the first five eggs are not rotten, but there is a chance that the sixth egg is and the only way to find out is to break open the egg. Now if he opens the egg into the bowl then there is a chance he might ruin the whole 6 eggs if the sixth one is rotten; putting it in the saucer seems like a better solution, but now one has two things to clean up. It leaves us with throwing the egg away, which is not ideal considering wasting a whole perfectly good egg. The options leading to actions are called **acts**; the two conditions (good or rotten) of the sixth egg are called **states of nature**; and the results from after the decision is made are named **consequences**. Table 1 summarises the decision problem.

Acts	**States of nature**	
	Good	**Rotten**
Break into bowl	six-egg omelette	no omelette / six eggs wasted
Break into saucer	six-egg omelette + one extra thing to clean	five-egg omelette + one extra thing to clean
Throw away	five-egg omelette + one wasted egg	five-egg omelette

Table 1: Savage's Omelette Dilemma

Depends on what the decision maker labels as huge **risk** (cleaning extra thing or wasting egg(s)), he will then come to his own conclusion of what decision turns out best for him. In this thesis, we only consider the basic **classical** decision problems where there are finite number of:

- **acts** to choose from
- **states of nature** to be aware of
- **consequences** (gain/**utility** or **risk**), each corresponds to the act and the state of nature, so e.g.: if there are three acts and three states then there will be $3 \times 3 = 9$ consequences

If the decision maker has no empirical data or observed information about the **likelihood** of various outcomes or **states of nature**, we call this decision problem a **no data** decision problem, which means that we will have to consider the problem of making decisions **under uncertainty** and **under ignorance**. That, once again, would mean that an agent knows about all the states of nature and their utilities, but no other information about these states of nature or prior probabilities about them are known (see Savage (1951); Vajda (1959)). From here we introduce a general definition of a classical decision problem (CDP) (also see Jansen (2015)):

A classical (no-data) decision problem (CDP) (in utility form) is a triplet:

$$U := (\mathbb{A}, \Theta, u(.))$$

with

- $\mathbb{A}$ a (finite) non-empty set of **acts**
- Θ a (finite) non-empty set of **states of nature**
- u(.) a mapping $(\mathbb{A} \times \Theta) \to \mathbb{R}$, $(a, \theta) \mapsto u(a, \theta)$; the output should also be a (finite) non-empty set of utilities.

Remark: these problems could also be defined in **loss** form (instead of **utility form**) by including l (loss) instead of u (utility) in the triplet, depending on whether the values of consequences should be interpreted as loss or utility.

When the problem U is finite (as in the example of Savage's dilemma), we can formulate the triplet as such: $\mathbb{A} = \{a_1, .., a_n\}$ and $\Theta = \{\theta_1, .., \theta_m\}$ and the utilities can be put on a table:

u(a$_i$, θ_j)	θ_1	...	θ_m
a$_1$	u(a$_1$, θ_1)		u(a$_1$, θ_m)
$\vdots$	$\vdots$	...	$\vdots$
a$_n$	u(a$_n$, θ_1)	...	u(a$_n$, θ_m)

Table 2: Table with all acts and states

Here, the entry (i,j) of the table equals the utility of the pair $(a_{i,j})$. The letters n and m always denote the cardinalities of the sets A and Θ respectively.

3.2 Different criteria of Decision Theory

Having stated the essential components defining a CDP, the question now is how one should make their decision about which action should be chosen? We start with a special case of CDP, where it might happen that in a problem we have a clear winning strategy, where choosing to follow an action would lead to the best outcomes (in favour of the decision maker), regardless of states of nature; or the other way around where one action should not be chosen at all under any condition. This leads to the following definition (see Luce and Raiffa (1989), Jansen (2015)):

Let U be any CDP. An action $a \in \mathbb{A}$ is said to be **inadmissible**, if there exists an action $a \in \mathbb{A}$, such that:

1. u(a^*, θ) $\geq u$(a,θ) $\forall\ \theta \in \Theta$
2. u(a^*, θ) $>$ u(a, θ) $\exists\ \theta \in \Theta$

If that happens we call a^* strictly dominates a and write $a^* \gg a$ or $a^* \succeq a$. Table 4 illustrates an example of inadmissible action: a_2 strictly dominates a_1, which makes a_1 inadmissible. a_3 and a_2 on the other hand cannot be compare w.r.t. the relation $\gg$. In this case one will work with a reduced set of actions, consisting of only a_2 and a_3.

In most cases, however, there would be no inadmissible action: we need to come up with criteria that make all actions comparable meanwhile still being admissible. This motivates our next general definition (see Jansen (2015)):

Let U be a CDP, then we call any mapping a **criterion** such that:

$$\Phi : \mathbb{A} \to \mathbb{R}, a \mapsto \Phi(a)$$

An action $a^* \in \mathbb{A}$ is **optimal** for the criterion Φ^U, if $\Phi^U(a^*) \geq \Phi^U(a)$ holds for all a $\in \mathbb{A}$

Acts	**States**		
	θ_1	θ_2	θ_3
a_1	12	13	14
a_2	14	15	16
a_3	21	10	16
a_4	13	17	12

Table 3: Example of inadmissible action. Action a_2 strictly dominates a_1, which makes a_1 inadmissible.

Before we dive into the criteria for optimal decision making, let us mention one more relevant concept of decision theory: the different types of uncertainty. Jansen (2015) classifies into 2 types, depending on the nature of the **process generating** the states of nature:

- Type I: The realised state of nature is **independent** of the action chosen by the actor. The process generating the states of nature can be compared to an ideal lottery, as every state occurs with a fixed and known (classical) probability. The nature cannot influence the generation of the states.

- Type II: For every action the actor chooses, the nature will pick (one of the) state(s) minimising the actor's utility. In this case, the nature can fully influence the process generating the states of nature.

As these two sound very unrealistic and restrictive w.r.t. non-academic situations, Jansen (2015) also states two modifications:

- Type I^*: Exactly like in the Type I situation, but the exact probability mass function on the set θ is assumed to be unknown to the actor (decision-maker). Based on the quality of information about the process generating the states, the decision-maker may be able to provide their best subjective estimate of the true probability distribution.

- Type II^*: Again, the nature acts as an antagonist to the actor. However, now the nature will not know exactly which action the decision maker will choose in order to pick the state minimising the actor's utility. Jansen (2015) provides the example of playing chess against a friend, where the friend will try to pick theleast favourable move possible, but he could not possibly know exactly which move you would go for after that.

Having the two types stated, we want to construct our criteria based on this belief about the type of uncertainty. Remark from Jansen (2015): it would be horrible to construct criteria for the optimality of decisions based on wrong beliefs about the true type of uncertainty. Our following discussion will base on the assumption that actor's belief and the true type of uncertainty coincide. For the sake of completeness it should also be mentioned that this thesis only dicusses aboout some of the most common criteria based on **cardinal utility**: on the assumption that the consequences (utilities) of the acts for the different states of nature (of the underlying utility table) can be ranked in a cardinal order. Beyond that there are also criteria based on ordinal utility (where the cardinality is not available/questionable) and criteria based on Preference Systems, where instead of an utility table the actors' preferences is represented by a preference system.

3.2.1 Optimal criteria under type I/I^* uncertainty: Bernoulli-/Bayes-Actions

As the probability of each state is known (Type I), a straightforward criterion would be: an action is optimal iff it maximises the expected utility under the state of nature. With that in mind let us first define the so-called **expected ultility**. According to Augustin (2024), the **expected ultility** (of an action a w.r.t. π) of a data-free decision problem $(\mathbb{A}, \Theta, u(.))$ (in utility form) for each fixed a $\in \mathbb{A}$ can be expressed as:

$$\mathbb{E}_\pi(u(a)) := \int_\Theta u(a, \theta)\, d_\pi(\theta)$$

Important here are the conditions coming with the expression:

- π a probability measure on $(\Theta, \sigma(\Theta))$ (probability for each state of nature to happen)

- $\sigma(\Theta)$ is an σ-algebra over Θ and $\theta \in \sigma(\boldsymbol{\Theta})$ for all $\theta \in \Theta$
- $u(a, \theta)$: $\Theta \to \mathbb{R}$, $\theta \mapsto u(a, \theta)$ is $\sigma(\Theta) - \mathcal{B} - measurable$) and an π- integrable mapping for all a $\in \mathbb{A}$

From that the **Bernoulli criterion** can be defined as:

$$\Phi(., \pi) = \mathbb{A} \to \mathbb{R}, a \mapsto \Phi(a, \pi) = \mathbb{E}_\pi(u(a))$$

with our goal is to find the (one) action a^* so that $\mathbb{E}_\pi(u(a^*, \theta)) \geq \mathbb{E}_\pi(u(a), \theta)$ for every a $\in \mathbb{A}$.

Under Type I^* uncertainty, the exact probability measure π is unknown, which makes the criterion above inapplicable - instead we will use the Bayes-criterion: using a prior-distribution ξ estimating π. In this thesis we will not dig into more details, if one wants to know more please refer to Augustin (2002) and Jansen (2015).

3.2.2 Optimal criteria under type II/II^* uncertainty: Maximin-Actions

Under Type II the decision maker should always expect the worst state of nature possible (which would minimise their utility (pessimistic)) . Looking this way, one should opt for the action that maximises the utility in the worts-case-scenario. This criterion is widely known as the **Maximin-principle** (or Wald-rule) (see Wald (1951) for further details), and could be mathematically defined as such:

$$\Phi(a) = \mathbb{A} \to \mathbb{R}, a \mapsto \Phi(a) = \inf_{\theta \in \Theta} u(a, \theta)$$

If the decision maker concerns about loss (risk) more than how much they could earn through the action, then the criterion could be defined as:

$$\Phi(a) = \mathbb{A} \to \mathbb{R}, a \mapsto \Phi(a) = \sup_{\theta \in \Theta} l(a, \theta)$$

with the logic that $\sup_{\theta \in \Theta} l(a, \theta)$ equivalent to the case of $\inf_{\theta \in \Theta} u(a, \theta)$. As our goal is to find an action a^* so that $\inf_{\theta \in \Theta} u(a*, \theta) \geq \inf_{\theta \in \Theta} u(a, \theta)$ for every $a \in \mathbb{A}$, one could express the criterion in this way: an action $a^* \in \mathbb{A}$ is optimal iff:

$$\Phi(a^*) = \max_{a \in \mathbb{A}}(\min_{\theta \in \Theta}(u(a, \pi)))$$

hence the name of the criterion. It is worth to mention that even under type II^* uncertainty, this **maximin** strategy will still be the best approach - while more details on this will not be discussed further, one can read Jansen (2015) for more information.

Apart from these pessimistic point of view of the outcome state of nature (that always minimises actor's gain / maximises actor's loss (risk/regret), there exists a criterion where we assume the best outcome in favour of the actor: the **maximax** (Max-Max) criterion. Knowing the two criteria above, it is almost intuitive to come up with the expression for this optimal strategy: an action $a*$ is optimal iff:

$$\Phi(a^*) = \max_{a \in \mathbb{A}}(\max_{\theta \in \Theta}(u(a, \theta)))$$

While this criterion should be considered too optimistic for most of the cases in real life, it is still included here in this thesis as a foundation for our next criterion (a combination of the aforementioned criteria): the **Hurwicz** criterion. First presented in a paper in 1951, it introduces an approach for balancing optimism and pessimism in decision making **under uncertainty** (see Hurwicz (1951) and Hurwicz (2024)):

$$\Phi(a) = \mathbb{A} \to \mathbb{R}, a \mapsto \Phi(a) = \alpha \max_{\theta \in \Theta} u(a, \theta) + (1 - \alpha) \min_{\theta \in \Theta} u(a, \theta) \tag{1}$$

with $\alpha \in [0, 1]$ the **optimism** parameter and $|\Theta| < \infty$.

There is one more criterion from this uncertainty type, which is the **Minimax** regret criterion. The idea is quite similar to the maximin action when we are considering **loss** instead of gain. Again, from a pessimistic point of view (i.e.: we assumed the worst thing (the biggest possible loss would happen)), we would want to **minimise** the **largest possible loss** (that could happen if we choose one particular action to follow).

We could define the criterion mathematically as follow (see Augustin (2024)): Given the data-free decision problem (i.e. Type II) of $(\mathbb{A}, \Theta, l(\cdot))$ in loss form with Θ and $\mathbb{A}$ finite. The decision problem $(\mathbb{A}, \Theta, r(\cdot))$ in loss form with

$$r : \mathbb{A} \times \Theta \to \mathbb{R}, \quad (a_i, \theta_j) \mapsto r(a_i, \theta_j)$$
$$r(a_i, \theta_j) = l(a_i, \theta_j) - \min_{\ell=1,\ldots,n} \big(l(a_\ell, \theta_j)\big)$$

is called **induced regret problem**.

Each $a^* \in \mathbb{A}$ with

$$\max_{j=1,\ldots,m} r(a^*, \theta_j) \leq \max_{j=1,\ldots,m} r(a, \theta_j) \quad \text{for all } a \in \mathbb{A}$$

is called **minimax-regret action**. Another way to express this would be:

$$\Phi(a) = \mathbb{A} \to \mathbb{R}, a \mapsto \Phi(a) = \arg\min_{a \in \mathbb{A}} (\max_{j=1,\ldots,m} r(a, \theta_j)) \tag{2}$$

3.2.3 Combination of Type I and II: The Hodges & Lehmann criterion

Instead of having a clear case of under which uncertainty type we are working with, a lot of cases will be a mixture of those two extremes: a criterion considering both at the same time would be needed, and in fact there are many different proposed criteria in the literature, however, this work will focus only on one of them: The criterion of Hodges Lehmann from 1952 (see Hodges and Lehmann (1952)). The criterion could be expressed as such:

$$\Phi(a) = \mathbb{A} \to \mathbb{R}, a \mapsto \Phi(a) = \alpha \mathbb{E}_\pi(u(a)) + (1 - \alpha) \min_{\theta \in \Theta} u(a, \theta)$$

with $\alpha \in [0, 1]$ the **confidence** parameter or **trade-off** parameter.

The idea is quite intuitive: Any type of uncertainty can be viewed as a trade-off between Type I and Type II uncertainty. The optimal criterion under Type I uncertainty is assumed to be the Bernoulli-criterion (here the first addend in the formula), the optimal criterion under Type II uncertainty is assumed to be the Maximin criterion (the second addend in the formula). If α was closer to 1, the more the optimal criterion would agree with the Bernoulli-criterion; vice versa the optimal criterion would align with the Maximin-criterion when the α got closer to 0. The question now is how one can be certain which value should α take/ which type of uncertainty is underlying the CDP of interest? In reality one can only have vague guesses that it is neither strict Type I nor strict Type II uncertainty - the best solution would be to fix α best possible and act as if the criterion perfectly fits the underlying uncertainty type (Jansen (2015)).

3.2.4 Example Case Study

For better intuition of all the criteria discussed above, let us dive into a simple example, where we will have the chance to apply what we just learnt. We will use the same example from Table 4, but now without considering act a_1 (being inadmissible).

Acts	**States**		
	θ_1	θ_2	θ_3
a_2	14	15	16
a_3	21	10	16
a_4	13	17	12

Table 4: Same table from Table 4, without the inadmissible actions.

First, let us assume that that is all what we know about our situation, and we need to make the best decision out of it (with the most gain). That would be Type II uncertainty, where we could apply the Maximax, Maximin and Hurwicz criteria. We start by getting the maximum (for the Maximax criterion), minimum value (for the Maximin criterion) of ultilities from each act (across all states), put them in a new collumn and from that getting the maximum value, which will be the best decision for Maximax and Maximin criteria (see Table 5). For demonstration of the criterion of Hurwicz we set the optimism parameter $\alpha = 0.6$.

Then we move on to the Minimax criterion: for this we need to create a regret table from the original utility table: in each state of nature we calculate the regret(risk) for each act by:

- first we pick out the largest return in each state (across all acts),
- then we calculate the regret of each act in one state: by subtracting the return (that we get from this act) from this largest return (from first step)

Now that we got our regret table, we carry on with the Minimax criterion: for each state of nature we pick out the maximal regret (the **max** in mini**max**, highlighted in red). Then

Acts	States			Maximax	Maximin	Hurwicz
	θ_1	θ_2	θ_3			
a_2	14	15	16	16	14	$0.6 \times 16 + (1 - 0.6) \times 14 = 15.2$
a_3	21	10	16	21	10	$0.6 \times 21 + (1 - 0.6) \times 10 = 16.6$
a_4	13	17	12	17	12	$0.6 \times 17 + (1 - 0.6) \times 12 = 15$

Table 5: The Maximax, Maximin, and Hurwicz criteria approaches to solving a single decision problem. For each act we pick out the maximal outcome(gain), from there we select the highest value of the three, which should be the optimal solution according to the criterion of Maximax. Maximin analogous. By Hurwicz we need to apply the formula mentioned earlier in subchapter 3.2.2, then pick the maximum. Both Maximax and Hurwicz criterion suggest that we should go for act a_3, while with Maximin the best optimal action would be a_4

we select the minimal regret out of all states (the **min** in **min**imax, again highlighted in red). All in all, the criterion compares ”what I could have done” with ”what I have done”; Table 6 illustrates the idea.

States	Regret (Acts)			Minimax
	a_2	a_3	a_4	
θ_1	$21 - 14 = 7$	$21 - 21 = 0$	$21 - 13 = 8$	8
θ_2	$17 - 15 = 2$	$17 - 10 = 7$	$17 - 17 = 0$	7
θ_3	$16 - 16 = 0$	$16 - 16 = 0$	$16 - 12 = 4$	4

Table 6: The Minimax criterion approach to solving a single decision problem, with calculated regret values. According to this criterion we should go for act a_4

By using Bernoulli criterion, we are assuming that we know for certain the probability that each state of nature would happen. Let us assume our CDP under strict type I uncertainty and let ξ be the probability measure on $(\theta, \mathcal{P}(\theta)$ with $\xi(\theta_1) = 0.3$, $\xi(\theta_2) = 0.4$. Then for each act we calculated the expected utility(gain) with table 7 then pick out the maximal outcome (highlighted in red).

Acts	States of nature			Expected Value
	θ_1	θ_2	θ_3	
a_2	14	15	16	$0.3 \times 14 + 0.4 \times 15 + 0.3 \times 16 = 14.9$
a_3	21	10	16	$0.3 \times 21 + 0.4 \times 10 + 0.3 \times 16 = 16.6$
a_4	13	17	12	$0.3 \times 13 + 0.4 \times 17 + 0.3 \times 12 = 14.3$

Table 7: The Bernoulli criterion approach to solving a single decision problem. According to this criterion we should go for act a_3

With Hodges and Lehmann criterion we assume that underlying uncertainty type is completely known to the actor - this would mean that the exact value for the trade-off parameter α is known. Let us again assume the probability measure on $(\theta, \mathcal{P}(\theta)$ with $\xi(\theta_1) = 0.3$ and $\xi(\theta_2) = 0.4$ and $\alpha = 0.5$. Then we implement the calculation for this criterion:

- first we calculate the expected value for each act (calculation of the Bernoulli criterion's part)
- second for each act we pick out the minimal outcome(gain) across all the states of nature (calculation of the **min** part in Maxi**min**)
- then we apply the formula (1)

Table 8 illustrates the criterion. As we are already done with the calculation, what we need to do is combining Table 5 and Table 7, then carry on the last calculation and pick the optimal action (with maximum outcome (profit/gain)).

Acts	**States**			**Bernoulli part**	**Maximin part**	**Hodges and Lehmann**
	θ_1	θ_2	θ_3			
a_2	14	15	16	$0.3 \times 14 + 0.4 \times 15 +$ $+ 0.3 \times 16 = 14.9$	14	$0.5 \times 14.9 + 0.5 \times 14$ $= 14.45$
a_3	21	10	16	$0.3 \times 21 + 0.4 \times 10$ $+ 0.3 \times 16 = 16.6$	10	$0.5 \times 16.6 + 0.6 \times 10$ $= 14.3$
a_4	13	17	12	$0.3 \times 13 + 0.4 \times 17$ $+ 0.3 \times 12 = 14.3$	12	$0.5 \times 14.3 + 0.5 \times 12$ $= 13.15$

Table 8: The Hodges and Lehmann criterion approach to solving a single decision problem, with $\lambda = 0.5$. This criterion opts for the action a_2

4 Alternative Acquisition Functions applying Decision Theory

As the goal of this thesis is to discuss the mechanism underlying the method of Bayesian Optimisation from the view of a decision theoretician, this chapter aims to introduce some new infill functions applying the optimally criteria discussed in the previous chapter.

Revision: Before we deep dive into these infill functions, we wish to revise our goal of optimisation: in chapter 2.1 it is stated: we need to find x in a sample space of $\mathcal{X}$ of d dimensions, so that our unknown function f(x) get its optimal value (in context of this thesis the minimum value), so we want to find the one x^* so that:

$$x^* = \arg\min_{x \in \mathcal{X}} f(x)$$

with $\mathcal{X}$ a d-dimensional space. Our approach is that we get our initial values and fit a surrogate model (Gaussian Process Regression), from that we receive our parameter: the posterior mean response μ and the posterior standard deviation σ. With these two and using a certain acquisition function we could decide which location in the $\mathcal{X}$ space might contain the optimal value we are looking for. Since we want to find the optimal value with the least "digging" through the search space $\mathcal{X}$ as possible (because of cost-sensitivity), one typical ("cheap") approach would that we make all future decisions optimally a.k.a. the Bellman's principle of optimisation (Bellman (1952)). That would mean that our acquisition function should opt for the best possible candidate it could find. This approach is called **myopic**, as we only look one step ahead at a time into the future. Other approach considering multi-steps ahead at a time into the future are also proven to be useful (though more expensive, see Yue and Kontar (2020), Di Fiore and Mainini (2024), Jiang et al. (2020)), but we won't talk about them here any further.

Now in order to come up with new infill functions applying the optimal criteria from decision theory, we need to determine how the relevant concepts in two fields (Decision Theory and Bayesian Optimisation) relate to each other (within the scope of this thesis). We start with the concept of **act** (or action) from Decision Theory: The actor makes a decision to **choose an act** from a set of infinite possible acts - in the case of black-box function that would be **choosing an x** from a set $\mathcal{X}$ of all possible values for x. Then we have states of natures - this one is a bit trickier - one would have to understand how the Gaussian Process really works. Figure 5 gives a picture of how the nature would affect on its outcome states.

It could be understood that **each state of nature** would equal to **one different regression function** in Figure 5. As the possible functions are finite, our states of nature are also finite. Since we cannot be sure of the exact regression function underlie the our black-box function, there would always be uncertainty, which is indicated by the confidence bound with standard deviation σ (Figure 7).

From the confidence bound (CB) we now have an understanding of how the states of

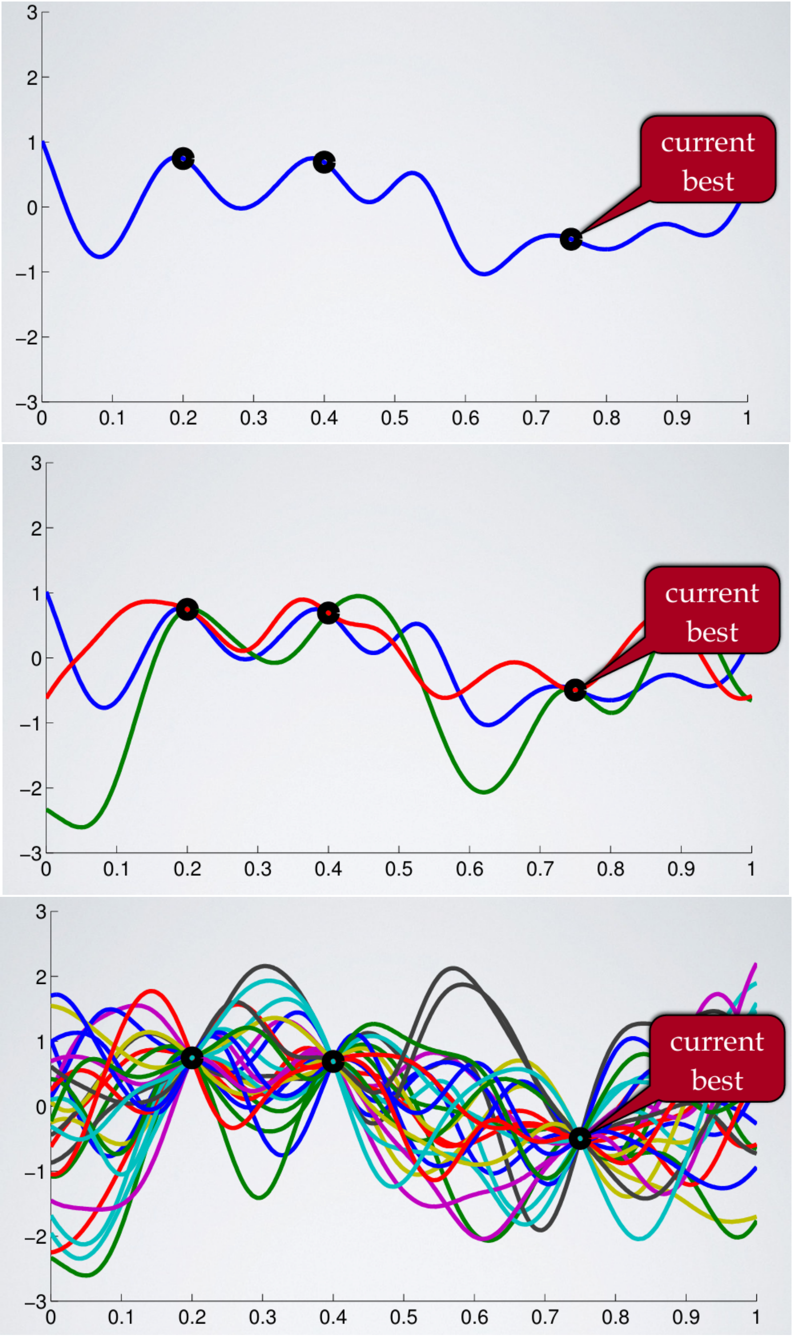

Figure 5: Given the first three initial points from the sample space we could fit a finite amount of regression function. This is exactly why fitting a simple regression function would not work too well - knowing all the possibility using GP is crucial. Source: Adams (2018)

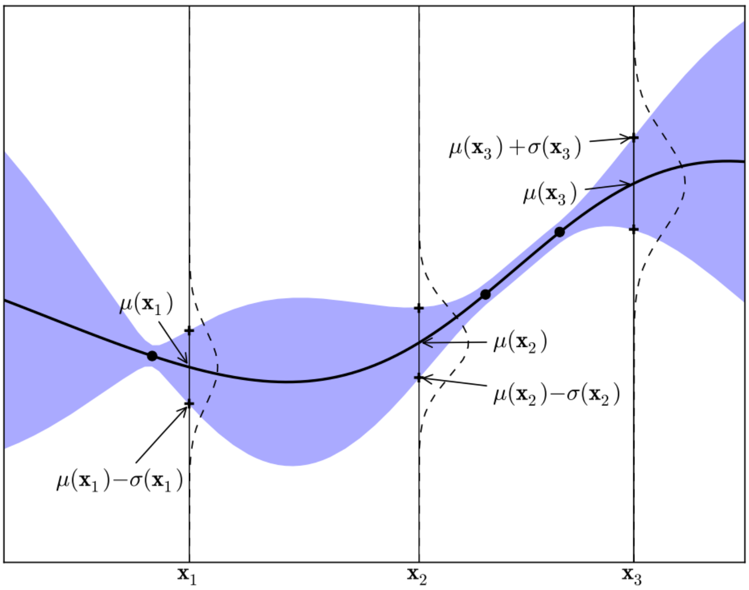

Figure 6: From the first three initial points we fit a Gaussian Process Model, which comes with a confidence bound (in purple). Source: Mayer (2017)

nature look like (where the limits are). For example when one choose x_1 (the action), the states of nature would result in a range of evaluation values $[\mu(x_1)-\sigma(x_1),\ \mu(x_1)+\sigma(x_1)]$, which could be interpreted here as utility (gain). In our case of searching for the minimum, if we choose x_1 as our next "act", we "gain" the most if the state of nature results in the evaluation on the border of the lower bound, and we "gain" least if we "land" on the border of the upper bound during the evaluation.

4.1 Maximax criterion

As a revision, the **Maximax** criterion refers to the case when the decision maker is very optimistic that the best outcome would always happen, regardless of action they choose. With that mindset the actor would neglect the bad cases and only focus on picking the best case scenario option. For example, in the case of choosing one from the three options x_1, x_2 and x_3 in Figure 7, one would make decision based solely on the evaluation on lower border of the lower confidence bound (with the value $\mu(x_i)-\sigma(x_i)$ with i $\in 1, 2, 3$). If these were the only 3 options to pick from, one would go with x_1 as the value of $\mu(x_1) - \sigma(x_1)$ is the lowest out of the three.

From that we define our first alternative infill criterion using the maximax strategy: x^* should be chosen as the next location to evaluate iff:

$$x^* = \arg\min_{x\in\mathcal{X}}(\mu(x) - \lambda\sigma(x))$$

One should recognise now that this is the exact mathematical definition of the acquisition function using the lower confidence bound criterion (LCB), as mentioned in chapter 2.3. We could also controll the trade-off between exploration (higher σ) and exploitation (lower μ) by including $\lambda > 0$ as a parameter. We will howerver set by default $\lambda = 1$.

4.2 Maximin criterion

As opposed to the cases using the maximax criterion, using the Maximin criterion (as an acquisition function in Bayesian Optimisation) shows that the decision maker is pessimistic with the outcome, that they always picture the worst case would happen, and aim to maximise the result in that case (here getting the largest value of y_i in any x_i location, which is when the possible value of y_i lands on the upper confidence bound). Therefore the objective of problem could be expressed as:

$$x^* = \arg\min_{x \in \mathcal{X}} (\mu(x) + \lambda\sigma(x))$$

Here the decision maker consider the worst possible evaluation at each candidate location x_i, then pick out the minimum out of these evaluation $\mu(x_i) + \lambda\sigma(x_i)$.

4.3 Hurwicz criterion

From the idea of Hurwicz presented in chapter 3.2, we are here firstly trying to find the compromise between an optimistic decision (when the y_i value lands on the lower confidence bound) and a pessimistic decision (when the y_i value lands on the upper confidence bound). The α (a.k.a. the **coefficient of realism**) takes value in the bounded interval between 0 and 1 and shows how optimistic we are that the best result of y_i will happen when we decide to evaluate the point x_i. The mathematical expression would hence be:

$$x^* = \arg\min_{x \in \mathcal{X}} (\alpha(\mu(x) - \lambda\sigma(x)) + (1-\alpha)(\mu(x) + \lambda\sigma(x))$$

This could be expressed in a shortened form:

$$x^* = \arg\min_{x \in \mathcal{X}} (\mu(x) + (1-2\alpha)\lambda\sigma(x))$$

4.4 Minimax Regret criterion

In Bayesian Optimisation, a state of nature corresponds to a specific realisation of the unknown function f(x) that the surrogate model is attempting to approximate (see Figure 5, where each line through those 3 points could be considered a possible state of nature). The surrogate model, such as a Gaussian Process (GP), provides a probability distribution over these states. As another revision: for each x, the GPR gives:

- A predictive mean $\mu(x)$, which is the expected value of f(x)
- A predictive standard deviation $\sigma(x)$, which quantifies uncertainty about f(x)

Each state of nature corresponds to one possible realisation of f(x) drawn from the GP posterior distribution.

The risk measures the performance of a specific decision x relative to the unknown states of nature. In the minimax context, we calculate risk as such: Regret for choosing x under a specific state of nature is the difference between:

- The best possible gain (across all decisions and given the state of nature).
- The gain achieved by x.

In terms of the surrogate model: The best possible gain is approximated as: $\min_{x \in \mathcal{X}} f(x)$ (as we try to minimise the target function). For that the true regret (risk) for a decision x in a specific state of nature f(x) should be:

$$R(x, f) = f(x) - f(x^*)$$

with x^* the location where a **specific realisation of** f gets its minimum value in this iteration. From this we want to consider for **each decision x** (in this iteration) the **greatest risk** that is available, then from those potential risks (each respective to one decision x, across all realisations of f) we pick out the one decision that has the smallest risk out of all. With mathematical expression one could write, that the decision x' is optimal for this criterion iff:

$$x' = \arg \min_{x \in \mathcal{X}} (\max_{f \in \mathcal{F}} (f(x) - f(x^*)))$$

with $\mathcal{F}$ set of all possible realisation for the unknown function f. This ensures that even in the worst-case realisation of f, the regret of choosing the optimal candidate x' is as small as possible.

But as we do not know this f(x) for each particular state of nature,, we rely on the GP posterior (mean $\mu(x)$ and variance $\sigma(x)^2$ to approximate regret. First we approximate the true minimum:

$$\min_{x \in \mathcal{X}} f(x) = \min_{x \in \mathcal{X}} \mu(x) = \hat{f}_{min}$$

Then the approximate Regret for a decision x should be:

$$r(x) = \mu(x) - \hat{f}_{min}$$

To account for uncertainty in the GP model, we adjust the regret using the variance:

$$r(x) = \mu(x) + \lambda\sigma(x) - \min_{x \in \mathcal{X}} (\mu(x) + \lambda\sigma(x)) \tag{3}$$

where $\lambda > 0$ is a parameter controlling the tradeoff between exploration (higher σ) and exploitation (lower μ). The criterion should be then written as:

$$x' = \arg \min_{x \in \mathcal{X}} (\max_{f \in \mathcal{F}} r(x))$$

Now let us dive a bit deeper to the function (3) to find out what should be the x that minimises $\mu(x) + \lambda\sigma(x)$. We already established that $\mu(x) + \lambda\sigma(x)$ is the upper confidence bound, given the posterior μ and σ at a certain iteration. Then the minimum location of this upper bound should be at one of the already-sampled points, more exactly our incumbent point/the sampled point with the lowest f(x) value. Knowing this, the next most potential point would hence be the exact same point that we already sampled!

Now it may sound like we are running in circle at this point, as we would not find a better point if the most potential point is one of the initial points. But we should also consider the case that **noises exist** - that would mean evaluating at one same point would bring a slightly different value after every iteration. That would also mean that the posterior σ^2 at this point is not equal to 0 (we call it **nugget**). This is when this **Minimax** criterion comes into play, when evaluating the same sample points actually update our understanding of our black-box function. As mentioned before, we will only deal with the cases where there is no noise; the discussion on this criterion thus ends here.

4.5 Bayes criterion

The main objective is to choose the action with the best expected utility (here is the minimum). As our surrogate model is the Gaussian process, we could directly extract the expected value of y_i at any location of x_i ($\mu(x_i)$). We come to the expression, which is the same as the criterion of reinstating pure mean $\mu(x)$ minimization (pure exploitation):

$$x^* = \arg\min_{x \in \mathcal{X}} \mu(x)$$

4.6 Hodges-Lehmann criterion

This approach is a combination of the aforementioned Maximin Criterion and the Bayes Criterion. The weighting of the trade-off between these two extreme criteria can be controlled by a trade-off parameter $\alpha \in [0, 1]$:

$$x^* = \alpha(\arg\min_{x \in \mathcal{X}} \mu(x)) + (1 - \alpha)(\arg\min_{x \in \mathcal{X}}(\mu(x) + \lambda\sigma(x)))$$

This function could also be expressed in a shortened form as:

$$= \arg\min_{x \in \mathcal{X}}(\mu(x) + (1 - \alpha)\lambda\sigma(x))$$

5 Experiments

5.1 Experiments with synthetic test functions

We are now in the last part of this thesis, where we are going to benchmark the new infill functions with different **synthetic** testing functions (in total of 12 objective functions) from the package **smoof** (see Bossek (2017)) to get a first understanding of how they would perform. Although we know the explicit mathematical form of all these testing functions and where exactly their minima are located (details will be explained later on), we still consider them as black-box functions for the sake of our test (i.e. we are only able to get the output values y when we input some values of x, any more details of the objective function than that should be considered beyond our knowledge). The idea of this test is inspired by Bischl et al. (2018).

Now for each of these target functions, we run the BO algorithm in order to find the **target global minimum**, each time with different infill functions - for each pair of testing and infill function we will repeat the test 10 times, then measure and compare the best result we receive and the running time needed to reach that decision. For the whole benchmark the package for implementing BO strategies will be **mlrMBO** (see Bischl et al. (2018)).

5.1.1 Problem design

With problem design we mean the design of the objective functions. In this thesis, we will only focus on the Model-Based Single-Objective Optimisation, which involves optimising (minimising in our case) a single scalar function, often representing a single goal or criterion.

For this benchmark we pick four 2-dimensional, four 4-dimensional and four 6-dimensional target function, which are all continuous, and single-objective: Ackley, Deflected Corrugated Spring, Rosenbrock and Schwefel. All are defined in the R-package **smoof** (see Bossek (2017)) and have been subject to optimisation benchmarks previously. Table 9 shows the properties of each of the 12 objective functions (4 ”classes” of functions, each comes with 3 functions in 3 different dimensions (2, 4 and 6)) used in this benchmark.

Test functions	Formula	Properties	Domain	Minimum Location
Ackley	f(x) $= -20exp(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i}) - \exp(\frac{1}{d}cos(2\pi x_i))$	differentiable, multimodal	[-32.8,32.8]	$x_i = 0$ and $f_{min} = 0$
Deflected Corrugated Spring	f(x) $= 0.1\sum_{i=1}^{d}(x_i - \alpha)^2 - cos(K\sqrt{\sum_{i=1}^{d}(x_i - \alpha)^2})$ $\alpha = K = 5$ by default	highly multimodal, symmetric global optimum far away from the next best local optimum	[0,2α]	$x_i = \alpha$ and $f_{min} = -1$
Rosenbrock	f(x)= $\sum_{i=1}^{d-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2$	unimodal, non-convex, differentiable	[-30,30]	$x_i = 1$ and $f_{min} = 0$
Schwefel	f(x) $= \sum_{i=1}^{d} -x_i sin(\sqrt{\lvert x_i \rvert})$	highly multimodal, global optimum far away from the next best local optimum	[-500,500]	$x_i = 420.9687$ $d = 2 : f_{min} = -418.9829$ $d = 4 : f_{min} = -1675.9316$ $d = 6 : f_{min} = -2513.8974$

Table 9: Test functions used for benchmarking and their properties

5.1.2 Algorithm design

The algorithms here are the BO strategies, which in our thesis only differ from each other in their acquisition functions. Figure 7 demonstrates a common routine of a typical BO algorithm. We start our algorithm with collecting initial sample points from the black box target function.

Initial sampling set: Sampling **randomly** is a plausible approach for this, but that would be inefficient as we might only collect all the points from a small area in the sample space. Since the evaluations are considered costly, we want to find a way to sample the least amount of points as possible while still capturing the essence of the black box function. For that reason we will generate design for each problem by using a space-filling maximin Latin Hypercube Design (see McKay et al. (1979b)). We then set the size of the design to be 5 times the number of the black-box function's parameters - i.e.: if the problem is 2-dimensional then there will be 10 initial sampled points.

Definition of the surrogate regression model: Till this point it is clear that the our first choice w.r.t. choosing a surrogate model is GPR. It is also a rational choice such that if the sample space $\mathcal{X} \in \mathbb{R}^d$, then Kriging (a.k.a. GPR, with nugget effect of $1e^{-6}$) (see Jones et al. (1998)) would be recommended and provides state-of-the-art performance. In our test we pick a fixed kernel the Matérn 3/2 which is mentioned in chapter 2.2.2 and holds the form:

$$k_{\text{Matérn3}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \left(1 + \frac{\sqrt{3}d}{l}\right) \exp\left(-\frac{\sqrt{3}d}{l}\right),$$

with again $d = ||x - x'||$, l the length-scale and σ^2 a scaling factor.

More details on this kernel could be found here Heuvelink (2000).

Definition of the infill function: as mentioned above we will want to put our new infill functions in chaper 4 into use here. Besides our alternative functions, we will also benchmark the popular **Expected Improvement** infill function to compare it with our new functions.

Infill optimisation: We use **focus search** (mentioned in chapter 1) to optimise the infill functions (getting the next evaluation point), and keep all the parameters in default settings as in the package **mlrMBO**: $n_{restarts} = 3$, $n_{iters} = 5$ and $n_{points} = 1000$.

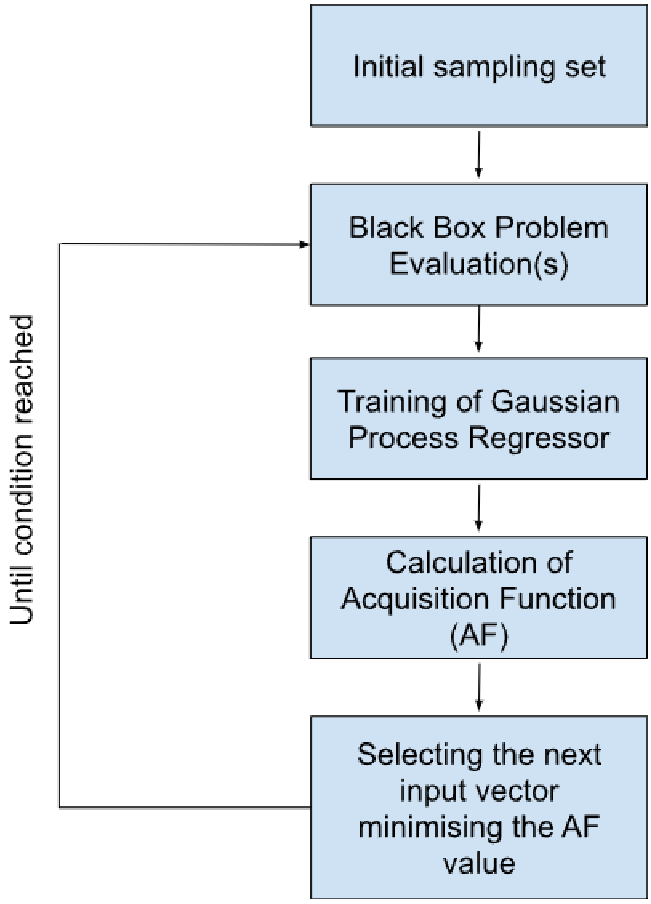
Initial sampling set
Black Box Problem Evaluation(s)
Training of Gaussian Process Regressor
Calculation of Acquisition Function (AF)
Selecting the next input vector minimising the AF value
Until condition reached

Figure 7: Basic routine of an BO algorithm. Source: paretos (2021)

5.1.3 Execution of Experiments

Every experiment terminates if a total number of 100 function evaluations has been reached and returns the optimal configuration, regardless of the problem's dimension. Each experiment is repeated 10 times, as mentioned earlier.

5.1.4 Evaluation of Results

Besides the quality of the solutions, runtime/computational overhead will also be considered. Note that we are basically measuring the overhead of the optimization algorithms, as the synthetic test functions are evaluated in microseconds.

5.2 Results

From this benchmark we get Table 8 (see in the Appendix), where all the possible optimal results of y (from 10 trials) are shown in the y-axis with box plots, each box plot comes from a different BO strategy with a respective infill function.

Now let us interpret the results: the infill function of Expected Improvement (EI) proves its popularity, when its results are the best by **7 out of 12** functions (all the 2D-functions except 2D-Deflected-Corrugated-Spring function, all the 4D-function except for 4D-Rosenbrock function and the 2D-Deflected-Corrugated-Spring function) w.r.t. the best outcome of y achieved. **In second place** is the infill function inspired by the Hurwicz criterion (see chapter 4), having the best y outcome from 3 functions (4D- and 6D-Rosenbrock and 6D-Schwefel function). **In third place** is the infill function inspired by the criterion of Maximax (which is the Lower Confidence Bound (LCB) acquisition function, see chapter 4) with 2 functions (4D-Ackley function and 4D-Deflected-Corrugated-Spring function (tying with EI infill function)).

What is worth to mention is that, by the 6D-Schwefel function, EI came last with quite a significant difference in results compared to other four infill functions. Here Hurwicz came first and the infill function inspired by the Hodges-Lehmann criterion performed quite well. Hodges-Lehmann infill function also performed pretty well by 4D-Rosenbrock and 6D-Ackley functions. The infill function inspired by the Maximin criterion seemed to be quite unstable when performing and hence unreliable — it only achieved good results by 2D-Rosenbrock and performed better than EI by 6D-Schwefel.

We also got the Table 10, where we can see for each optimisation algorithm with each infill function how long **in average** each algorithm took to get to its results, and its average rank. Regarding ranking we want to elaborate further to avoid any confusion: we have 5 algorithms x 12 test functions x 10 repetitions — in the end we get 600 rows of results + their running time. Now we group this 600 rows into 12 groups according to 12 test functions, from here we rank the algorithm w.r.t. its best outcome y (i.e.: with the same test function the algorithm with the best outcome y gets the first rank (first place) etc.) As there are also repetitions — one algorithm will have 10 ranks for just 1 test function — hence we will need to take the mean of these 10 ranks to get the average rank of each algorithm w.r.t. one test function. At this point we should have 12 test functions x 5 average ranks (for each algorithm), so again each algorithm has overall 12 average ranks across all the test functions — this is where we take the mean again for each algorithm, so that in the end, each algorithm should only have **one** average rank. The same idea is applied to calculating the average running time of each algorithm across all test functions and repetitions.

Algorithm	**Average Rank**	**Average runtime in minutes**
ei	21.6	129.72
hodgesLehmann	30.2	130.49
hurwicz	23.0	132.81
maximax	20.4	137.72
maximin	32.3	133.95

Table 10: Average ranks and runtime on artificial test functions.

So how should one interpret this? The average rank should be out of 50 (as for each test function there are 5 algorithms x 10 repetitions). In first place we have the algorithm perform by Maximax infill function (LCB infill function, see Chapter 4), which is closely followed by EI, then Hurwicz. Considering also the average runtime, the EI proved again why it is the most commonly used acquisition function. If one wants an alternative for EI, then Hurwicz and Maximax should be taken into account. Not to forget that this is only a quite simple benchmark, as we only repeated the test 10 times and each test only looped 100 times. Also all the hyper-parameters are pretty much kept "by default" (without tuning), which is actually not a optimal approach, as BO is known for not performing well directly out-of-the-box — in our test we received quite good results, but it could happen that the algorithms could not even converge to an optimum at all (see Frazier (2018), Turner et al. (2021)).

6 Conclusion and Discussion

This thesis has shown how one could apply common **Decision Theory (DT) criteria** to creating new strategies for particular **Bayesian Optimisation (BO) problems**. First we discussed the main idea, some use cases of BO and how one could implement a BO strategy from start to end. Here we broke everything down into components — two most crucial components of which are **Surrogate Function** (with Gaussian Process Regression (GPR)) and **Acquisition Function** (in this thesis continuous and differentiable functions). In the part of GPR, we talked about what the elements making up a GPR model are, and how we could train one 'from scratch'. From here the readers should also have a good understanding on how a GPR predicts and how it updates its hyperparameters during fitting.

Then we moved on to explaining the essence of **Decision Theory**, where we discussed its usage in everydaylife, then mentioned some more important applications of this study, before we talk about a **Classical Decision Problem** and its elements. After that, we moved on to the salient part of this thesis — the ideas of **Decision Criteria**, as we needed this to come up with new acquisition functions for our BO algorithms. We ended this chapter by considering a case study, where we got to apply all the criteria we learned.

In Chapter 4, we explained the transition from the idea of Decision Criteria to implementing to optimising functions using GPR. We came to the realisation that the infill function inspired by the Maximax criterion is similar to the already-well-known Lower Confidence Bound infill function. We also saw the potential of the infill function from Minimax-Regret-criterion, but we would not go any further on this topic — this could be a good topic for further study (e.g. for noisy BO).

Then in Chapter 5, we put these alternative infill functions into use, with a (quite) basic benchmark test, where we tried to optimise 12 artificial test functions using BO and our new acquisition functions. We got the test's results including all the best results from the optimisation and also their running time. We rank these results accordingly and came to conclusion that EI is still the most suitable infill functions for these particular optimisation problems. Here remained potentials for larger tests with more computational power.

A Appendix

(see next page)

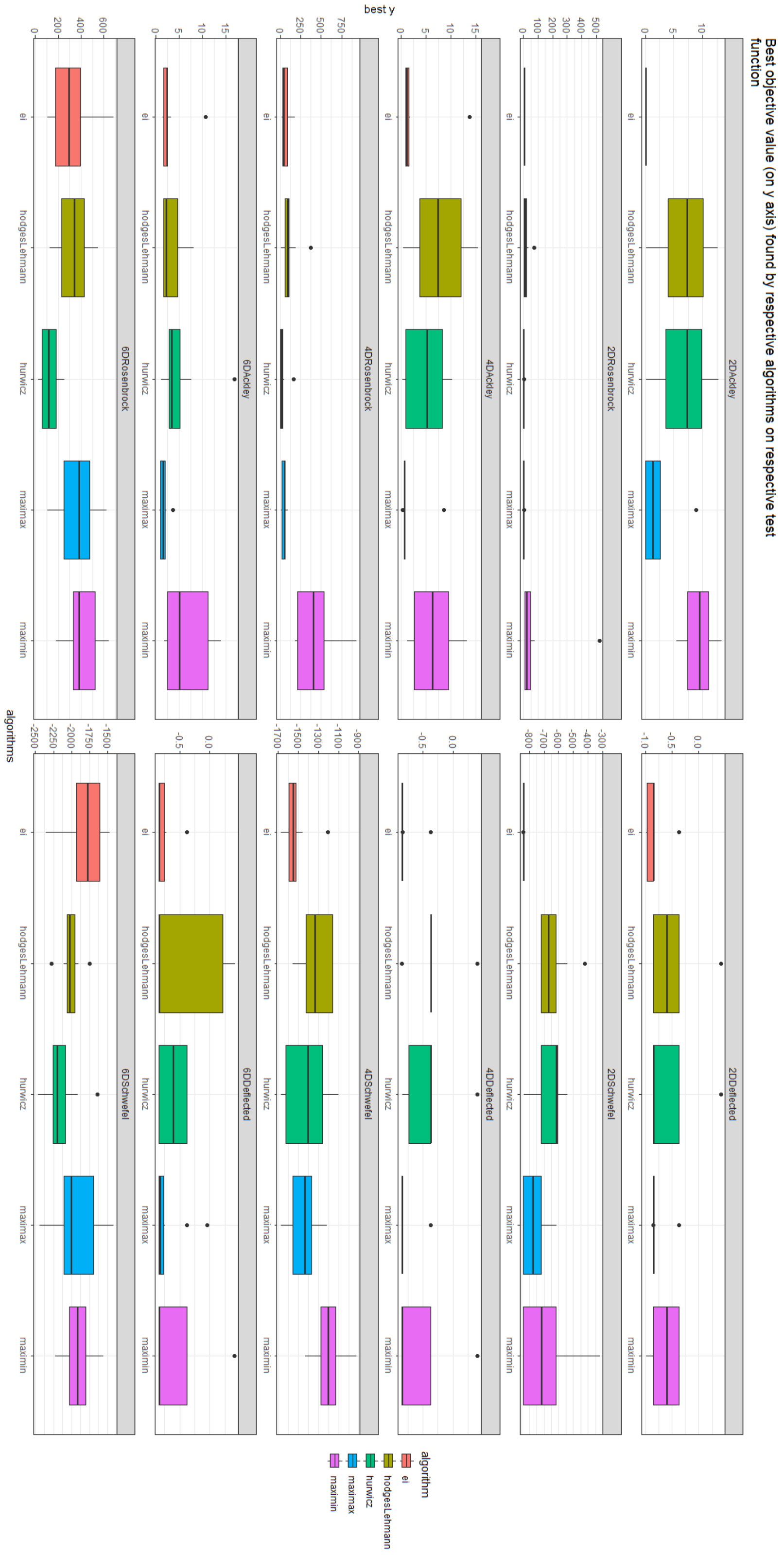

Figure 8: Best objective value of f(x) found by different algorithms on different test (10 repetitions) function.

B Electronic appendix

The code produced in this work can be found on:
https://github.com/DucAnhValentinoNguyen/dtMBO

The benchmark test was run on my personal computer: with AMD Ryzen 9 6900HS with Radeon Graphics, 3.30 GHz and 8 physical cores. It took me almost a whole day (see Table 10 for better understanding).

References

Adams, R. P. (2018). A tutorial on bayesian optimization for machine learning.
URL: *https:www.cs.toronto.edu/ rgrosse/courses/csc411*$_f$*18/tutorials/tut8*$_a$*dams*$_s$*lides.pdf*

Augustin, T. (2002). Expected utility within a generalized concept of probability — a comprehensive framework for decision making under ambiguity, *Statistical Papers* **43**(1): 5–22.
URL: *https://EconPapers.repec.org/RePEc:spr:stpapr:v:43:y:2002:i:1:p:5-22*

Augustin, T. (2024). Lecture desicion theory, chapter bayes decisions in the ‘virtual risk situation.
URL: *https://moodle.lmu.de/mod/resource/view.php?id=1887619*

Bellman, R. (1952). On the theory of dynamic programming, *Proceedings of the National Academy of Sciences* **38**(8): 716–719.
URL: *https://www.pnas.org/doi/abs/10.1073/pnas.38.8.716*

Bischl, B. (2019). Gaussian process, `https://slds-lmu.github.io/i2ml/chapters/19_gaussian_processes/`. Accessed: 2025–02-26.

Bischl, B., Richter, J., Bossek, J., Horn, D., Thomas, J. and Lang, M. (2018). mlrmbo: A modular framework for model-based optimization of expensive black-box functions.
URL: *https://arxiv.org/abs/1703.03373*

Bossek, J. (2017). smoof: Single- and multi-objective optimization test functions, *The R Journal* .
URL: *https://journal.r-project.org/archive/2017/RJ-2017-004/index.html*

Brochu, E., Cora, V. M. and de Freitas, N. (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning.
URL: *https://arxiv.org/abs/1012.2599*

Cox, D. and John, S. (1997). Sdo: A statistical method for global optimization.

Cuingnet, R. (2023). On the computation of the logarithm of the modified bessel function of the second kind.
URL: *https://arxiv.org/abs/2308.11964*

Dennis, D. and John, S. (1992). A statistical method for global optimization, *[Proceedings] 1992 IEEE International Conference on Systems, Man, and Cybernetics* pp. 1241–1246 vol.2.
URL: *https://api.semanticscholar.org/CorpusID:122348801*

Di Fiore, F. and Mainini, L. (2024). Nm2-bo: Non-myopic multifidelity bayesian optimization, *Knowledge-Based Systems* **299**: 111959.
URL: *http://dx.doi.org/10.1016/j.knosys.2024.111959*

Frazier, P. I. (2018). A tutorial on bayesian optimization.
URL: *https://arxiv.org/abs/1807.02811*

Guo, J., Zan, X., Wang, L., Lei, L., Ou, C. and Bai, S. (2023). A random forest regression with bayesian optimization-based method for fatigue strength prediction of ferrous alloys, *Engineering Fracture Mechanics* **293**: 109714.
URL: *https://www.sciencedirect.com/science/article/pii/S0013794423006720*

Hayes, C. F., Rădulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L. M., Dazeley, R., Heintz, F., Howley, E., Irissappane, A. A., Mannion, P., Nowé, A., Ramos, G., Restelli, M., Vamplew, P. and Roijers, D. M. (2022). A practical guide to multi-objective reinforcement learning and planning, *Autonomous Agents and Multi-Agent Systems* **36**(1).
URL: *http://dx.doi.org/10.1007/s10458-022-09552-y*

Hennig, P. and Schuler, C. J. (2011). Entropy search for information-efficient global optimization.
URL: *https://arxiv.org/abs/1112.1217*

Heuvelink, G. (2000). Interpolation of spatial data: Some theory for kriging: M.l. stein, springer, new york, 1999. hardcover, 247 pp., us$49.95, isbn 0-387-98629-4, Geoderma$ **96** : 153–154.

Hodges, J. L. and Lehmann, E. L. (1952). The use of previous experience in reaching statistical decisions, *Annals of Mathematical Statistics* **23**: 47–58.
URL: *https://api.semanticscholar.org/CorpusID:119870153*

Hurwicz, L. (1951). The generalised bayes-minimax principle: A criterion for decision-making under uncertainty.
URL: *https://cowles.yale.edu/sites/default/files/2024-03/s-0355.pdf*

Hurwicz, L. (2024). The hurwicz criterion.
URL: *https://www.leonidhurwicz.org/hurwicz-criterion/*

Hutter, F. (n.d.). Computationally cheap acquisition functions, `https://moodle.ki-campus.org/pluginfile.php/41046/mod_videotime/intro/w06_bo_t02_cheaacq.pdf/`. Accessed: 2025–03-10.

Hutter, F., Hoos, H. H. and Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration, *in* C. A. C. Coello (ed.), *Learning and Intelligent Optimization*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 507–523.

Jansen, C. (2015). *Decision making under partial information using precise and imprecise probabilistic models*, PhD thesis.
URL: *http://nbn-resolving.de/urn/resolver.pl?urn=nbn:de:bvb:19-epub-25591-1*

Jiang, S., Jiang, D., Balandat, M., Karrer, B., Gardner, J. and Garnett, R. (2020). Efficient nonmyopic bayesian optimization via one-shot multi-step trees, *in* H. Larochelle,

M. Ranzato, R. Hadsell, M. Balcan and H. Lin (eds), *Advances in Neural Information Processing Systems*, Vol. 33, Curran Associates, Inc., pp. 18039–18049.
URL: *https://proceedings.neurips.cc/paper$_f$iles/paper/2020/file/d1d5923fc822531bbfd9d87d4760914*
Paper.pdf

Jones, D. (2001a). A taxonomy of global optimization methods based on response surfaces, *J. of Global Optimization* **21**: 345–383.

Jones, D., Perttunen, C. and Stuckman, B. (1993). Lipschitzian optimisation without the lipschitz constant, *Journal of Optimization Theory and Applications* **79**: 157–181.

Jones, D. R. (2001b). A Taxonomy of Global Optimization Methods Based on Response Surfaces, *Journal of Global Optimization* **21**(4): 345–383.
URL: *https://doi.org/10.1023/A:1012771025575*

Jones, D., Schonlau, M. and Welch, W. (1998). Efficient global optimization of expensive black-box functions, *Journal of Global Optimization* **13**: 455–492.

Kushner, H. J. (1964). A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise, *Journal of Basic Engineering* **86**: 97–106.
URL: *https://api.semanticscholar.org/CorpusID:62599010*

Li, Y. L., Rudner, T. G. J. and Wilson, A. G. (2024). A study of bayesian neural network surrogates for bayesian optimization.
URL: *https://arxiv.org/abs/2305.20028*

Lizotte, D. (2008). Practical bayesian optimization.

Lizotte, D., Wang, T., Bowling, M. and Schuurmans, D. (2007). Automatic gait optimization with gaussian process regression, *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 944–949.

Luce, R. and Raiffa, H. (1989). *Games and Decisions: Introduction and Critical Survey*, Dover Books on Mathematics, Dover Publications.
URL: *https://books.google.de/books?id=msC$_h$0wtCO8C*

Ma, Y., Cao, D. and Feng, J. (2023). Application of random forest algorithm with bayesian optimization to reservoir prediction: Application of random forest algorithm with bayesian optimization to reservoir prediction, *2023 International Conference on New Trends in Computational Intelligence (NTCI)*, Vol. 1, pp. 412–418.

Majidi, M., Mohammadi-Ivatloo, B. and Soroudi, A. (2019). Application of information gap decision theory in practical energy problems: A comprehensive review, *Applied Energy* **249**: 157–165.
URL: *https://www.sciencedirect.com/science/article/pii/S0306261919308037*

Martinez-Cantin, R., de Freitas, N., Brochu, E., Castellanos, J. and Doucet, A. (2009). A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot, *Autonomous Robots* **27**(2): 93–103.
URL: *https://doi.org/10.1007/s10514-009-9130-2*

Martinez-Cantin, R., Freitas, N., Doucet, A. and Castellanos, J. (2007). Active policy learning for robot planning and exploration under uncertainty.

Mayer, V. (2017). Alternative infill-kriterien für random forests in sequential model-based optimization.
URL: *http://nbn-resolving.de/urn/resolver.pl?urn=nbn:de:bvb:19-epub-38411-8*

McKay, M. D., Beckman, R. J. and Conover, W. J. (1979a). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, *Technometrics* **21**(2): 239–245.
URL: *http://www.jstor.org/stable/1268522*

McKay, M. D., Beckman, R. J. and Conover, W. J. (1979b). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code, *Technometrics* **21**(2): 239–245.
URL: *http://www.jstor.org/stable/1268522*

Mockus, J. (1994). Application of Bayesian approach to numerical methods of global and stochastic optimization, *Journal of Global Optimization* **4**(4): 347–365.
URL: *http://link.springer.com/10.1007/BF01099263*

Mockus, J., Tiesis, V. and Zilinskas, A. (2014). *The application of Bayesian methods for seeking the extremum*, Vol. 2, pp. 117–129.

paretos (2021). Bayesian optimization (bayes opt): Easy explanation of popular hyperparameter tuning method.
URL: *https://www.youtube.com/watch?v=M-NTkxfd7-8t=353s*

Rasmussen, C. E. and Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning*, The MIT Press.
URL: *https://doi.org/10.7551/mitpress/3206.001.0001*

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*, The MIT Press.

Savage, L. J. (1951). The theory of statistical decision, *Journal of the American Statistical Association* **46**(253): 55–67.
URL: *http://www.jstor.org/stable/2280094*

Shahriari, B., Swersky, K., Wang, Z., Adams, R. and De Freitas, N. (2016a). Taking the human out of the loop: A review of bayesian optimization, *Proceedings of the IEEE* **104**(1): 148–175. Publisher Copyright: © 1963-2012 IEEE.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. and De Freitas, N. (2016b). Taking the human out of the loop: A review of bayesian optimization, *Proceedings of the IEEE* **104**(1): 148–175. Publisher Copyright: © 1963-2012 IEEE.

Snoek, J., Larochelle, H. and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms.
URL: *https://arxiv.org/abs/1206.2944*

Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M. M. A., Prabhat and Adams, R. P. (2015). Scalable bayesian optimization using deep neural networks.
URL: *https://arxiv.org/abs/1502.05700*

Springenberg, J. T., Klein, A., Falkner, S. and Hutter, F. (2016). Bayesian optimization with robust bayesian neural networks, *in* D. Lee, M. Sugiyama, U. Luxburg, I. Guyon and R. Garnett (eds), *Advances in Neural Information Processing Systems*, Vol. 29, Curran Associates, Inc.
URL: *https://proceedings.neurips.cc/paper$_f$iles/paper/2016/file/a96d3afec184766bfeca7a9f989fc7e*
Paper.pdf

Srinivas, N., Krause, A., Kakade, S. M. and Seeger, M. W. (2012). Information-theoretic regret bounds for gaussian process optimization in the bandit setting, *IEEE Transactions on Information Theory* **58**(5): 3250–3265.
URL: *http://dx.doi.org/10.1109/TIT.2011.2182033*

The foundations of statistics. By Leonard J. Savage, John Wiley & Sons, Inc., 1954, 294 pp (1954). *Naval Research Logistics Quarterly* **1**(3): 236–236.
URL: *https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800010316*

Thompson, W. R. (1933a). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples, *Biometrika* **25**(3/4): 285–294.
URL: *http://www.jstor.org/stable/2332286*

Thompson, W. R. (1933b). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples, *Biometrika* **25**(3/4): 285–294.
URL: *http://www.jstor.org/stable/2332286*

Thornton, C., Hutter, F., Hoos, H. H. and Leyton-Brown, K. (2013). Auto-weka: Combined selection and hyperparameter optimization of classification algorithms.
URL: *https://arxiv.org/abs/1208.3719*

Turner, R., Eriksson, D., McCourt, M., Kiili, J., Laaksonen, E., Xu, Z. and Guyon, I. (2021). Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020.
URL: *https://arxiv.org/abs/2104.10201*

Vajda, S. (1959). Games and decisions. by r. duncan luce and howard raiffa. pp. xi, 509. 70s. 1957. (j wiley 38;amp; sons), *The Mathematical Gazette* **43**(344): 152–153.

Villemonteix, J., Vazquez, E. and Walter, E. (2007). An informational approach to the global optimization of expensive-to-evaluate functions.
URL: *https://arxiv.org/abs/cs/0611143*

Wald, A. (1951). Statistical decision functions.
URL: *https://api.semanticscholar.org/CorpusID:4175992*

Wei, Y., Zhuang, V., Soedarmadji, S. and Sui, Y. (2024). Scalable bayesian optimization via focalized sparse gaussian processes.
URL: *https://arxiv.org/abs/2412.20375*

Yue, X. and Kontar, R. A. (2020). Why non-myopic bayesian optimization is promising and how far should we look-ahead? a study via rollout, *in* S. Chiappa and R. Calandra (eds), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, Vol. 108 of *Proceedings of Machine Learning Research*, PMLR, pp. 2808–2818.
URL: *https://proceedings.mlr.press/v108/yue20b.html*