TUHH
Hamburg
University of
Technology

ICS
Institute
of Control
Systems

Master's thesis

Efficient and Safe Exploration Methods for Gaussian Processes

Author:
Emir Bajrović

Supervisor: Jannis Lübsen, M.Sc.

Examiner: 1. Prof. Dr.-Ing. Annika Eichler
2. Prof. Dr.-Ing. Timm Faulwasser

March 17, 2025

Master's thesis **M — M-3 — 4**
for Mr. Emir Bajrović Student ID: [redacted] Study Course: ETMS

Title:
Efficient and Safe Exploration Methods for Gaussian Processes

Project Description:
In recent years, Bayesian optimization [1] has emerged as a prominent area of interest due to its effectiveness in optimizing unknown functions with minimal assumptions. The basic concept is to estimate the unknown function at unobserved inputs using a Gaussian process model [2]. This model provides a mean value and variance for the function value at each point in a compact set, based on a finite number of function evaluations. In most cases, constrained optimization is considered [3], [4], either as a stand-alone objective or as a function of the unknown function. The region in which the constraints are satisfied is referred to as a safe region and depends on assumptions about the underlying function. First, the frequentist view [3], [5], [6], which assumes that the function is deterministic and a member of a reproducing kernel Hilbert space (RKHS). Second, the Bayesian view [7], which assumes that the function is stochastic and a sample from a Gaussian process. To determine the subsequent input at which the unknown function should be evaluated, a constrained optimization problem must be solved to ensure that this point lies in the safe region. The problem consists of three elements: the objective, the nonlinear acquisition function, and the constraints. The constraints are typically expressed as $c(x) \leq 0$, where $c(x)$ is also a nonlinear function. The goal of this thesis is to investigate the influence of the frequentist/Bayesian assumption on the function in terms of conservatism and constraint satisfaction. To this end, a literature review will be conducted on safe Bayesian optimization and a safe exploration is implemented in Python. Subsequently, methods for generating test functions that satisfy both assumptions are investigated and the performance and constraint satisfaction of Bayesian optimization will be tested.

Tasks:

1. Literature review on safe Bayesian optimization and recent methods for safe exploration
2. Implementation of a safe exploration in Python
3. Generation of test functions fulfilling the frequentist and Bayesian assumptions
4. Performance and constraint satisfaction of Bayesian optimization under both assumptions
5. Evaluation and discussion

References:

[1] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions", *Journal of Global Optimization*, vol. 13, no. 4, pp. 455–492, Dec. 1998, ISSN: 1573-2916.

[2] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005, ISBN: 026218253X.

[3] Y. Sui, A. Gotovos, J. Burdick, and A. Krause, "Safe exploration for optimization with Gaussian processes", in *32nd Int. Conf. Mach. Learn. (ICML)*, ser. Proceedings of Machine Learning Research, vol. 37, Lille, France: PMLR, Jul. 2015, pp. 997–1005.

[4] J. Kirschner, M. Mutny, N. Hiller, R. Ischebeck, and A. Krause, "Adaptive and safe Bayesian optimization in high dimensions via one-dimensional subspaces", in *36th Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 3429–3438.

[5] S. R. Chowdhury and A. Gopalan, "On kernelized multi-armed bandits", in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, Jun. 2017, pp. 844–853. [Online]. Available: https://proceedings.mlr.press/v70/chowdhury17a.html.

[6] C. Fiedler, C. Scherer, and S. Trimpe, "Practical and rigorous uncertainty bounds for gaussian process regression", May 2021.

[7] A. Lederer, J. Umlauft, and S. Hirche, "Uniform error bounds for Gaussian process regression with application to safe control", in *Advances in Neural Information Pro-cessing Systems*, Jun. 2019, pp. 659–669.

Supervisor: Jannis Lübsen, M.Sc.
Examiner: 1. Prof. Dr.-Ing. Annika Eichler
2. Prof. Dr.-Ing. Timm Faulwasser

Deutscher Titel: Methoden zur effizienten Erkundung von sicheren Regionen mit Gaußprozessen

Start Date: 01.09.2024
Due Date: 01.03.2024

March 17, 2025, Prof. Dr.-Ing A. Eichler

Hereby I declare that I produced the present work myself only with the help of the indicated aids and sources.

Hamburg, March 17, 2025 Emir Bajrović

Abstract

Safe Bayesian Optimization is a powerful method for optimizing black-box functions that are expensive to evaluate and subject to strict safety constraints. To model such functions, Gaussian Processes rose to the state of the art surrogate model. One of the key challenges in safe Bayesian Optimization is the design of uncertainty bounds as overly optimistic bounds jeopardize the containment of the true function which is strictly required for the satisfaction of safety constraint. However, overly conservative bounds generally reduce the optimization performance. Therefore, this thesis systematically reviews the most prominent uncertainty construction methods, resulting from *Frequentist* and *Bayesian* perspectives on the underlying true function. While the former leverages the concept of Reproducing Kernel Hilbert Spaces to construct quantifiable bounds on the model error, the latter exploits Lipschitz continuities. This works aims at quantifying the introduced conservatism of each perspective by providing a practical implementation of the `SafeOpt` and `SafeUCB` algorithms. Both enable the incorporation of strict safety constraints while exhibiting reasonable yet differing optimization performance. Within an experimental setup, the conservatism resulting from the frequentist and bayesian setting is assessed by comparing the convergence rates `SafeOpt` and `SafeUCB` under varying kernel and test functions. The results indicate increased optimization performance due to limited conservatism in the frequentist setting compared to its bayesian counterpart in the absence of safety constraint violations throughout the entire optimization process for both.

[redacted]

Contents

1 Introduction

Optimization problems are present among numerous scientific and engineering disciplines, especially in contexts where function evaluations of some real process are expensive and additionally subject to strict safety constraints. Traditional optimization methods like gradient-based techniques or heuristic search algorithms reveal severe drawbacks when dealing with black-box functions. Bayesian Optimization (BO) has emerged as a powerful strategy to tackle those challenges by incorporating probabilistic surrogate models. Gaussian Processes (GPs) are a non-parametric, supervised machine learning method [1] which gained attention as a prominent choice for such surrogate models due to their capability of providing function output predictions equipped with a mean and variance. In combination, GP-based BO has shown to efficiently guide the search for optimal solutions while minimizing the amount of function evaluations [2], [3].

A pivotal property of BO is balancing the trade-off between exploration and exploitation. The former emphasizes the sampling of function inputs whose output is expected to increase information about the underlying process whereas the latter focusses on the attempt of further optimizing the current best solution. However, real world applications like robotics [4], medical treatment [5] planning or autonomous systems [6] require strict safety constraint satisfaction throughout the entire optimization process since a violation of such constraints can lead to severe consequences such as system failure or hazardous outcomes. Therefore, safe BO has gained prominence as its framework allows the systematic incorporation of predefined safety constraints while simultaneously achieving remarkable optimization behavior. Examples of proposed algorithms that handle the described trade-off in different ways are the `SafeOpt` and `SafeUCB` algorithms which will be considered in the remainder of this work [3], [7].

The construction of uncertainty bounds within the frame of safe BO is thereby essential. On the one hand, properly chosen uncertainty bounds must reliably assure the containment of the underlying real process. On the other hand, the size of modeled uncertainty has a significant influence on the optimization performance as overly conservative bounds hinder convergence rates. The conservatism induced by the choice of uncertainty bound construction is hence an elaborate and active field of research as a trade-off between safety constraint satisfaction and performance must be handled.

The most prominent approaches for constructing uncertainty bounds result from varying assumptions about the modeled process. The *frequentist* view assumes the process to be represented by an unknown but fixed, deterministic function that is a member of the Reproducing Kernel Hilbert Space (RKHS), determined by the kernel function implemented in the GP modeling the unknown function. Thereby, it leverages RKHS norm bounds to derive uncertainty quantification [8], [9]. In contrast, the *bayesian* view assumes the unknown function to be probabilistic as a stochastic sample of the GP in place. Here, uncertainty quantification is achieved by assuming Lipschitz continuity of the unknown function and kernel [10], [11]. Despite the fact that both approaches base on sound theoretical derivations, they introduce conservatism to a varying extent. Understanding and more importantly quantifying the associated conservatism is therefore crucial for improving the efficiency and widening the applicability of safe BO in real-world scenarios.

The aim of this work is thus to contribute to the field of safe BO by providing an experimental evaluation of conservatism resulting from uncertainty quantification strategies and their impact on optimization efficiency in the presence of strict safety constraints. The key contributions include:

1. A theoretical, comprehensive comparison of frequentist an bayesian uncertainty scaling strategies, highlighting their strengths and limitations
2. A software implementation of modified `SafeOpt` and `SafeUCB` algorithms
3. An analysis of convergence rates depending on the uncertainty scaling method and algorithm choice along varying kernel and test functions
4. A discussion on limitations of the practical applicability of the frequentist and bayesian view

Firstly, the required theoretical fundamentals are presented in chapter 2, providing an overview of GP, BO and safe exploration along with a discussion on kernel functions and RKHS theory. Chapter 3 introduces prominent safe BO algorithms, namely the `SafeOpt` and `SafeUCB` algorithm. Moreover, the frequentist and bayesian error bound construction is demonstrated. Chapter 4 presents the experimental setup, including modified version of the considered algorithms, kernel parameterization and the choice of test functions. The empirical findings are presented in chapter 5 along the analysis of the impact of uncertainty quantification on optimization efficiency and safety constraint satisfaction. Finally, chapter 6 provides a summary of key findings from which implications for safe BO are derived and an suggestions for further research directions.

By addressing the challenges of uncertainty quantification in safe BO, this work aims at providing empirical result to pave the way for safer and more effective decision-making in complex real-world scenarios.

2 Fundamentals

One of the core concepts of this work are GPs. State of the art applications of BO oftentimes use GPs as a probabilistic surrogate to model some real process. Generally speaking, GPs are a generalization of Gaussian Distributions (GDs) thus a statistical method to provide information about the probability of some random variable. Therefore, the principles of GPs is introduced by a brief overview of univariate and multivariate GDs. Afterwards, the concept of BO is presented as a sophisticated method to solve optimization problems by modeling the underlying function with a GP particularly for the case of black box optimization, hence when the function to be optimized is unknown and expensive to evaluate. Lastly, GP-based BO is considered for the constrained optimization problem giving rise to the concept of safe BO.

2.1 Gaussian Distributions

Many real-world processes exhibit a probability distribution that is centered around a mean value with some quantifiable spread around it which characterizes as a Gaussian - also called normal - distribution. Examples are human weight, length or blood pressure but also technical observations like noise in electronic circuits. Hence, a random variable $\mathbb{X}$ is a continuous GD if its probability distribution function (PDF) p can be composed by

$$p(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\left(-\frac{x^2}{2\sigma^2}\right)} \tag{2.1}$$

where $\sigma^2 > 0$ is the variance. The mean value m can be determined by the expectation of random variable $\mathbb{X}$, i.e.

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} xp(x)\, dx = m. \tag{2.2}$$

A GD with random variable X can thus be written as $X \sim \mathcal{N}(m, \sigma^2)$. That has been the simple case of a one-dimensional or univariate random variable X. One significant advantage of a GD is the simple extension to a *multivariate* GD, where the random variable is no longer one-dimensional but n-dimensional. In that case, the random variable becomes a random vector $\boldsymbol{X} = (X_1, X_2, ..., X_n) \in \mathbb{R}^n$. The simplicity of the extension to the multivariate case lays in the fact that each variable $X_i (i = 0, ..., n)$ is itself a univariate GD. Hence, one can think of a vector $\boldsymbol{m} = (m_1, m_2, ..., m_n)^T$ carrying the mean values m_i of each univariate GD, allowing to say

$$\boldsymbol{m} = \mathbb{E}[\mathbb{X}] = (\mathbb{E}[X_1], \mathbb{E}[X_2], ..., \mathbb{E}[X_n]). \tag{2.3}$$

The variance from the univariate case translates to covariance matrix $\boldsymbol{\Sigma}$ in the multivarite case. Assuming a symmetric, positive definite matrix $\boldsymbol{\Sigma} = (\Sigma_{ij})_{nxn} \in \boldsymbol{R}^{nxn}$, the probability distribution function of multivariate normal distribution of the random vector $\boldsymbol{X}$ is given by

$$p(\boldsymbol{x} \mid \boldsymbol{m}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{2\pi|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{m})\right), \quad \boldsymbol{x} \in \mathbb{R}^n. \tag{2.4}$$

Note that $\boldsymbol{\Sigma}^{-1}$ denotes the inverse of $\boldsymbol{\Sigma}$ whereas $|\boldsymbol{\Sigma}|$ denotes the determinant of the matrix $\boldsymbol{\Sigma}$. If a random variable $\boldsymbol{X}$ has a probability distribution function as in 2.4, it classifies as a multivariate normal distribution denoted by $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{\Sigma})$. Then,

$$\mathbb{E}[\boldsymbol{X}] = \boldsymbol{m}, Cov[\boldsymbol{X}] = \boldsymbol{\Sigma}. \tag{2.5}$$

While the variance σ^2 in the univariate case embodied the spread around the mean, the covariance matrix Σ contains the variance of each random variable from $\mathbb{X}$ but also quantifies the covariance between pairs of random variables. Hence, the diagonal entries Σ_{ii} carry the variance for the GD of X_i,

$$\mathbb{V}[X_i] = \Sigma_{ii} \tag{2.6}$$

and the off-diagonal entries measure the covariance between different random variables

$$Cov[X_i, X_j] = \Sigma_{ij} = \mathbb{E}[(X_i - m_i)(X_j - m_j)]. \tag{2.7}$$

Assuming a zero off-diagonal covariance matrix, i.e. $\boldsymbol{\Sigma} = diag(\Sigma_1^2, \Sigma_2^2, ..., \Sigma_n^2)$ equation 2.4 simplifies to

$$p(x \mid \boldsymbol{m}, \boldsymbol{\Sigma}) = \prod_{i=1}^{n} \frac{1}{\Sigma_i\sqrt{2\pi}} \exp\left(-\frac{1}{2\Sigma_i^2}(x_i - \theta_i)^2\right). \tag{2.8}$$

It is obvious that for $n = 1$ this results to the PDF of the univariate case as given in 2.1.

2.2 Gaussian Processes

A Gaussian process however, does not deal with finite dimensional random variables but with functions. Therefore, it can be seen as a collection of random variables where any finite subset is a joint GD [1]. Considering some real process f, its function values $f(x)$ for some inputs x are represented by random variables. Comparably to GDs, GPs are fully parameterized by their mean and covariance. However, those are not scalar or vector values but functions. The mean function $\mu(x)$ provides a predicted function value for each input x whereas the covariance function $k(x, x')$ - also referred to as kernel - measures the similarity of expected function values of x, x'.

$$\begin{aligned} m(x) &= \mathbb{E}[f(x)], \\ k(x, x') &= \mathbb{E}[(f(x) - \mu(x))(f(x') - \mu(x'))] \end{aligned} \tag{2.9}$$

The Gaussian Process for some real process f is thereby denoted by

$$f(x) \sim \mathcal{GP}\left(\mu(x), k(x, x')\right). \tag{2.10}$$

The use of GPs in BO aims at generating predictive distributions over functions where each predicted function value $f(x_*)$ itself is a GD with some mean $m(x_*)$ and variance $k(x_*, x_*)$. To receive such predictive distributions from a GP, bayesian inference is employed which fits a probability model - here the GP - to a set of observed data. Note that the modeled process f is assumed to be of infinite dimension [1], making GPs a non-parametric model as it moreover grows with observed data. Furthermore, GPs learn the real process f based on provided observation allowing the categorization of GPs as a supervised machine-learning method [1].

The process of fitting a GP on a set of observed data and providing predictions on unobserved input can be separated in three major steps which are individually presented next: Prior Distributions, Observations, Posterior Distributions.

Prior Distribution

Before having any observations available, the model needs to be set up with some prior beliefs about the process. As the GP is fully characterized by its mean and covariance function, setting those imposes a prior distribution. In the remainder of this work, the mean function is set to zero ($\mu(x) = 0$). For now, the covariance function is chosen to be the one-dimensional Sqaured Exponential (SE) function

$$k_{SE}(x, x') = \exp\left(-\frac{(x - x')^2}{2\ell^2}\right). \tag{2.11}$$

The lengthscale ℓ is called the *hyperparameter* which can be set according to prior beliefs about the process. The choice of kernel functions and the tuning of their hyperparameters will be discussed in the following section. Equation 2.11 represents the one-dimensional case, the extension to the n-dimensional case is simply given by [12]

$$k_{\text{SE}}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \exp\left(-\frac{1}{2}(\boldsymbol{x}_p - \boldsymbol{x}_q)^\top \Lambda^{-1}(\boldsymbol{x}_p - \boldsymbol{x}_q)\right), \quad \boldsymbol{x}_p, \boldsymbol{x}_q \in \mathbb{R}^d. \tag{2.12}$$

The length scales are now collected in the diagonal matrix $\Lambda = diag([l_1^2, ..., l_n^2])$. Despite the simplicity of the choice of mean and covariance function, the resulting GP shows to be quite expressive [12]. The prior is determined by the choice of the mean and covariance function, simply stating that the function values of f are samples of a normal distribution

$$f(x) \sim \mathcal{N}(0, k_{\text{SE}}(x, x')), x, x' \in \mathbb{R}. \tag{2.13}$$

Observations

Since prior beliefs about a process are not enough to sufficiently model a real process, observations are collected. Those are then incorporated into the GP to allow improved posterior distributions. Relevant for this work are noisy observations, meaning there is no access to exact function values $f(x)$ but only noise corrupted measurements $y_i = f(x) + \epsilon$. Importantly, the noise is assumed to be independent, identically distributed (i.i.d.) Gaussian noise [1]. This means that the noise terms ϵ do not depend on the input or time at which observations are made. The noise is hence a normal distribution with zero mean and variance σ_ϵ^2.

$$\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2) \tag{2.14}$$

Now, assume the presence of n training points $(x_1, ..., x_n) \in \mathbb{R}^d$ and n target points $(y_1, ..., y_n) \in \mathbb{R}$. The observations of function values y_i are collected in the target vector $\boldsymbol{y} = (y_1, ..., y_n)^T$.

Posterior Distributions

After incorporating a prior distribution and gathering observations, one can move on to generating posterior distributions. To learn about some real process $f : \mathcal{X} \mapsto \mathbb{R}$ given N observations $\boldsymbol{y}$ where $\mathcal{X} \subset \mathbb{R}^d$, Bayes' rule [13] is fundamental as it provides the posterior distribution

$$p(f|\boldsymbol{y}) = \frac{p(f)p(\boldsymbol{y}|f)}{p(\boldsymbol{y})}. \tag{2.15}$$

Note that $p(\boldsymbol{y})$ can be thought of as a normalizing constant, also referred to as the *marginal likelihood*. It does not depend on the real process f, thus it does not influence the shape of the posterior distribution $p(f|\boldsymbol{y})$. What governs the posterior distribution is the *likelihood* $p(\boldsymbol{y}|f)$ and the prior $p(f)$. The likelihood quantifies the relationship between observations $\boldsymbol{y}$ and the unknown process f.

Translating this to the use of GPs, Bayes' rule yields the posterior distribution with input space as

$$p(f \mid \mathcal{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y} \mid f, \mathcal{X})p(f)}{p(\boldsymbol{y} \mid \mathcal{X})}. \tag{2.16}$$

Exploiting the observations $\boldsymbol{y}$, the prior distribution has been updated to the posterior distribution, expressing beliefs about the real process f. In other words, the prior distribution has been conditioned on the observations. This can be considered as reducing the space of functions the prior distributions assumed to potentially match the real process f by removing those functions inconsistent with the observations [1].

A graphical representation in figure 2.1 shows three randomly sampled functions from the prior distributions on the left plot whereas the right plot shows three randomly sampled

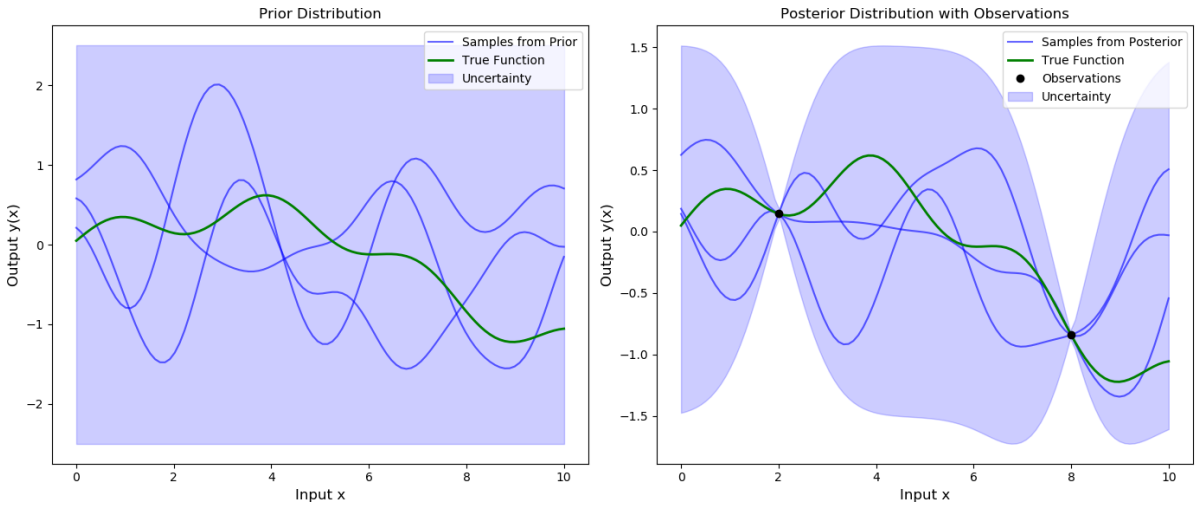


Figure 2.1: Three prior and posterior samples for $f \sim \mathcal{GP}(0, k_{SE}(x, x'))$

functions from the posterior distribution. Note that only samples from the posterior distribution are considered that comply with the two observations added.

To predict function values at unseen inputs, the mean and covariance function of the posterior distribution are used. Consider $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^T, \boldsymbol{x} \in \mathcal{X}$ to contain the training data, i.e. the inputs of to the generated observations $\boldsymbol{y}$. The vector $X_* \subset \mathcal{X}$ then contains the test points along which predictions shall be made. Applying Bayes' rule for the posterior distribution from 2.16 to the unseen function values $\boldsymbol{f}_*$ yields in

$$p(\boldsymbol{f}_*|X, \boldsymbol{y}, X_*) = \frac{p(\boldsymbol{y}|X, \boldsymbol{f}_*, X_*)p(\boldsymbol{f}_*)}{p(\boldsymbol{y}|X, X_*)}. \tag{2.17}$$

It is important to mention that the result of Bayes' rule itself is an infinite distribution over functions. However, the likelihood expressed by $p(\boldsymbol{y}|X, \boldsymbol{f}_*, X_*)$ is a finite dimension distribution, embodying the noise assumption σ_ϵ^2, since it corresponds to the product of all likelihoods of $\boldsymbol{f}_*$ according to

$$p(\boldsymbol{y}|\boldsymbol{f}_*, X) = \prod_{i=1}^{n} p(y_i|f_*(x_i)) = \prod_{i=1}^{n} \mathcal{N}(y_i|f_*(x_i), \sigma_\epsilon^2) = \mathcal{N}(\boldsymbol{y}|f_*(X), \sigma_\epsilon^2 \boldsymbol{I}). \tag{2.18}$$

What makes the posterior distribution an infinite distribution is the prior $p(\boldsymbol{f}_*)$, since the infinite dimensional SE kernel is chosen [12]. The joint distribution of the observations $\boldsymbol{y}$ and test points $\boldsymbol{f}_*$ according to the prior are given by

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K(X, X) + \sigma_\epsilon^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right). \tag{2.19}$$

Here K denotes Gram matrix of the input vectors X and X_*. Thus, it contains the kernel evaluations of all inputs contained in X and X_*, respectively.

What Bayes' rule in equation 2.17 does, is called *conditioning* the joint prior distribution on the observations which results in

$$\begin{aligned} \boldsymbol{f}_*|X, \boldsymbol{y}, X_* \sim &\mathcal{N}(K(X_*, X)[K(X, X) + \sigma_\epsilon^2 I]^{-1}\boldsymbol{y}, \\ &K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma_\epsilon^2 I]^{-1}K(X, X_*)). \end{aligned} \tag{2.20}$$

Thus, the mean and covariance functions of the predictive distribution [1] are given by

$$\begin{aligned} \bar{\boldsymbol{f}}_* \equiv \mathbb{E}[\boldsymbol{f}_*|X, \boldsymbol{y}, X_*] &= K(X_*, X)[K(X, X) + \sigma_\epsilon^2 I]^{-1}\boldsymbol{y}, \\ \text{cov}(\boldsymbol{f}_*) &= K(X_*, X_*) - K(X_*, X)[K(X, X) + \sigma_\epsilon^2 I]^{-1}K(X, X_*). \end{aligned} \tag{2.21}$$

Note that the noise assumption σ_ϵ^2 is incorporated by adding the diagonal matrix $\sigma_\epsilon^2 I$ which follows from the assumption about i.i.d. noise, meaning the noise of each observation is a normal distribution sample as in 2.14 and independent for each observation n [1].

To simplify those expressions, assume only one test point $\boldsymbol{x}_*$ and replace $K = K(X, X)$ and $\boldsymbol{k}_* = K(\boldsymbol{x}_*, X)$.

$$\begin{aligned} \bar{f}_* &= \boldsymbol{k}_*^T(K + \sigma^2 I)^{-1}\boldsymbol{y}, \\ \mathbb{V}[f_*] &= k(x_*, x_*) - \boldsymbol{k}_*^T(K + \sigma^2 I)^{-1}\boldsymbol{k}_* \end{aligned} \tag{2.22}$$

In that case, $\bar{f}_*$ is the predicted mean value at x_* and $\mathbb{V}[f_*]$ denotes the variance.

2.2.1 Kernel Choice and Hyperparameter Tuning

The previous section introduced kernel functions when setting a GP prior by demonstrating the SE kernel briefly. This section presents more elaborate theoretical foundations of kernel functions, types of kernels and methods for tuning their hyperparameters in the scope of Gaussian Process Regression (GPR).

As mentioned afore, the GP prior is entirely determined by its mean and covariance function. Whereas the mean function encodes information about the function values of some process, the covariance function provides a measure of similarity between data points. The concept *similarity* is crucial here, as it states that inputs x which are close are likely to result in function values $f(x)$ that are similar. Therefore covariance functions in GPR can be used to incorporate beliefs about how similar outputs are given their inputs.

A more general perspective of a kernel is a function k of two arguments that maps a pair of inputs $x, x' \in \mathcal{X} \subset \mathbb{R}^d$ into a real number, thus

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}. \tag{2.23}$$

An important property of real kernels is their symmetry stating that $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}', \boldsymbol{x})$ [1]. For a given set of inputs $[\boldsymbol{x}_i, ..., \boldsymbol{x}_n]$ for $i = 1, ..., n$ and a kernel k, the *Gram matrix* K can be computed which is also referred to as the *covariance matrix*. Containing the

evaluation of the kernel function k at a set of inputs, it must be positive semidefinite (PSD) in order to assure mapping into real positive numbers which then can be interpreted in a meaningful way as a similarity measure of inputs in GPs. For a symmetric $n \times n$ Gram matrix K to be PSD, it requires that all eigenvalues are non-negative which is equivalent to

$$\boldsymbol{v}^T K \boldsymbol{v} \geq 0, \forall \boldsymbol{v} \in \mathbb{R}^n. \tag{2.24}$$

Another important concept is the *stationarity* of kernel functions. For example, the SE kernel from equation 2.12 is stationary as it depends on the difference of inputs $\boldsymbol{x} - \boldsymbol{x}'$, making it invariant to translations in the input space. In other words, the covariance of inputs depends solely on their relative distance in the input space and not their absolute position. This is pivotal for their application in GPs since thereby assumptions about the true function contained in the kernel are consistent among the entire input space.

Another relevant concept is the mean square (MS) differentiability. Assume f to be a sample from a GP with kernel k. The mean square derivative of f in the i-th direction is then defined as

$$\frac{\partial f(\boldsymbol{x})}{\partial x_i} = \text{l.i.m.}_{h \to 0} \frac{f(\boldsymbol{x} + h\boldsymbol{e}_i) - f(\boldsymbol{x})}{h}, \tag{2.25}$$

when the limit in mean square (l.i.m.) exists. Note that $\boldsymbol{e}_i$ is the eigenvector in the i-th direction. The covariance function of that derivative can also be expressed by the kernel as

$$\frac{\partial^2 k(\boldsymbol{x}, \boldsymbol{x}')}{\partial x_i \partial x_i'}. \tag{2.26}$$

Within the frame of stationary processes, it can be stated that if the $2k$-th order partial derivative $\partial^2 k(\boldsymbol{x})/\partial^2 x_{i1} ... \partial^2 x_{ik}$ exists, then the k-th order derivative $\partial^k f(\boldsymbol{x})/\partial x_{i1} ... \partial x_{ik}$ exists for all $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_k$, $\boldsymbol{x} \in \mathbb{R}^D$ as a mean square limit. This is particularly relevant for GPs as smoothness assumptions about the modeled process f can be made by the kernel choice as the MS differentiability of the kernel allows to infer the MS differentiability of the stochastic process f. Note that MS differentiability can only be proven if MS continuity is given which can be assured for stationary kernels by checking continuity at $k(\mathbf{0})$ [1].

Given those theoretical foundations, prominent types of one-dimensional, stationary kernel functions are discussed. To simplify expressions, the variable r is introduced for the distance of applied inputs

$$r = x - x'. \tag{2.27}$$

The Squared Exponential Kernel Function

First, one of the most widely applied kernels is presented, the SE kernel. It can be written as

$$k_{SE}(r) = \sigma_f^2 \exp\left(-\frac{r^2}{2\ell^2}\right). \tag{2.28}$$

In contrast to its introduction in equation 2.11, another hyperparameter is introduced, the signal variance σ_f^2. It functions simply as a constant factor to the kernel function's output.

Interestingly, the SE kernel is infinitely differentiable due to the exponential term. Infinite differentiability is an important and distinct property as it imposes strong correlation between close function values and their high order derivatives. The concept of MS differentiability allows thereby to state that functions sampled from a GP with an SE kernel also exhibit mean square derivatives of all orders making them very smooth [1].

Despite the fact that the SE kernel is widely used in practical GP applications, such strong smoothness assumptions are described as unrealistic for real physical systems [14]. Moreover, imposing strong smoothness assumptions can become particularly problematic in high order GPs [15].

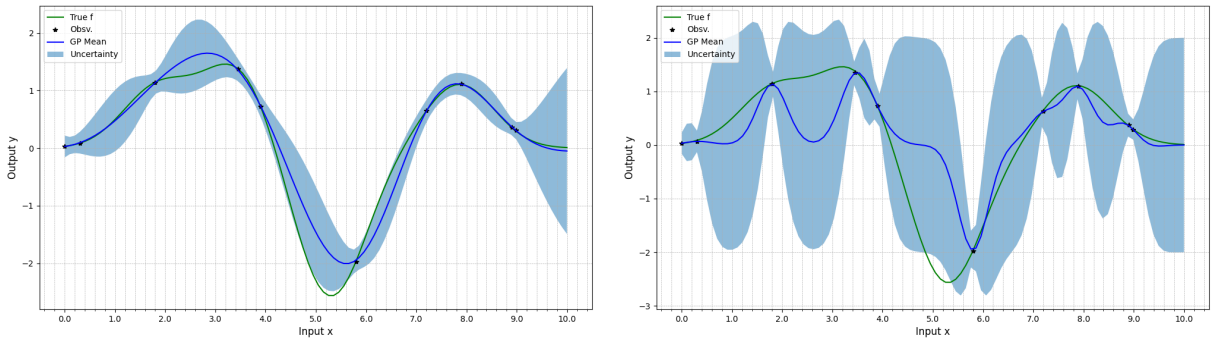
Next the influence of SE kernel's hyperparameters are analyzed by comparing results contained in figure 2.2. They depict the predictive distribution of a GP with zero-mean ($\mu(x) = 0$) and an SE kernel according to equation 2.28 with constant signal variance $\sigma_f^2 = 1$. Moreover, function values are made at the presence of i.i.d. gaussian noise of variance $\sigma_\epsilon^2 = 0.01$ resulting in noise-perturbed observations. The upper and lower bounds on confidence regions $u(x)$ and $l(x)$ are constructed by adding and subtracting two times the corresponding standard deviation.

$$\begin{aligned} u(x) &= \mu(x) + 2 \cdot \sigma(x) \\ l(x) &= \mu(x) - 2 \cdot \sigma(x) \end{aligned} \tag{2.29}$$

The lengthscale ℓ can be tuned to quantify prior belief about the similarity of inputs, i.e. how quickly function values change. The signal variance σ_f^2 scales the output of the kernel as it simply multiplies the exponential term. It does not account for how quickly function values change but rather under which magnitude.

Comparing figure 2.2a to figure 2.2b, the difference in both is created by the reduced lengthscale. While the mean function in figure 2.2a changes less rapidly as in figure 2.2b, it can be seen that decreased lengthscales assume low similarity while increased lengthscale incorporate high similarity. Therefore, the uncertainty between observed points increases.

Thus, hyperparameters can be an important tool for matching the GP's mean prediction with the the ground truth when sufficient knowledge in form of observations is available.



(a) $\ell = 1$, $\sigma_f = 1$, $\sigma_\epsilon = 0.01$ (b) $\ell = 0.3$, $\sigma_f = 1$, $\sigma_\epsilon = 0.01$

Figure 2.2: Predictive Distributions under Varying SE Kernel Hyperparameters

The Matérn Kernel Functions

The next type of stationary kernel function is the *Matérn* class of kernel functions. It is generally formulated by

$$k_{\text{Matérn}}(r) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu} r}{\ell} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu} r}{\ell} \right), \tag{2.30}$$

with ν and ℓ being positive parameters. The function K_ν represents the modified *Bessel* function which exhibits exponential decay for large inputs r [16]. Therefore, inputs that vary strongly will be quantified with low similarity. Especially the exponential decay for large inputs qualifies the Matérn kernel for the use in GPR. Moreover, it can be stated that sampled functions f from a GP utilizing the Matérn kernel are k-times MS differentiable if and only if $\nu > k$ holds [17]. Consequently, ν can be considered as a tuning parameter for the expected smoothness of the true function in the scope of GPR. This suggests that the Matérn kernel is able to impose weaker smoothness assumptions than the SE kernel. The hyperparameter ℓ is equivalent to the lengthscale discussed in the SE kernel. Note that for $\nu \to \infty$, one obtains the SE kernel [14].

The choice of ν is thus essential. The Matérn functions become particularly simple when $\nu = p + 1/2$ for p being a positive integer. In that case, ν becomes a positive half integer and the Matérn kernel yields in a product of an exponential term and polynomial of order p.

$$k_{\nu=p+\frac{1}{2}}(r) = \exp\left(-\frac{\sqrt{2\nu} r}{\ell} \right) \frac{\Gamma(p+1)}{\Gamma(2p+1)} \sum_{i=0}^{p} \frac{(p+i)!}{i!(p-i)!} \left(\frac{\sqrt{8\nu} r}{\ell} \right)^{p-i} \tag{2.31}$$

The most interesting cases of ν in machine learning have shown to be $\nu = 3/2$ and $\nu = 5/2$. The reason is, kernel functions with $\nu = 1/2$ - also known as the *Ornstein-Uhlenbeck* covariance function - result in very rough processes. On the other hand, for $\nu \geq 7/2$, one gets a large degree of differentiability yielding once again in strict smoothness assumptions [1].

Rational Quadratic Kernel Functions

The Rational Quadratic (RQ) kernel can be considered as an infinite sum of SE kernel with weighted lengthscales ℓ.

$$k_{RQ} = \left(1 + \frac{r^2}{2\alpha_{RQ}\ell^2}\right)^{-\alpha_{RQ}} \tag{2.32}$$

The hyperparameters here are the lengthscale $\ell > 0$ and the signal variance σ_f^2 which functions in the same manner as for SE and Matérn. The RQ kernel exhibits a third hyperparameter $\alpha_{RQ} > 0$. Especially useful in the multidimensional case, α_{RQ} can thus be used to assess weights on different lengthscales. Note that the RQ kernel results in infinite MS differentiability of a process f for all α_{RQ}. Furthermore, for $\alpha_{RQ} \to \infty$ the RQ kernel becomes the SE kernel [1].

Marginal Likelihood for Hyperparameter Selection

As demonstrated the optimal hyperparameter selection for a GP is a non-trivial task. An efficient method to compute optimal hyperparameters by exploiting certain properties of GPs and the Bayesian Inference principle will be presented now.

Generally, the non-parametric nature of GPs makes it hard to understand what the actual parameters of the model are. Most interesting models in Machine Learning require the computation of integrals over the parameters space which is analytically intractable. Fortunately, such integrals form an exception in the scope of GPs as they are analytically tractable while resulting models with promising flexibility [1]. To demonstrate that, recall equation 2.16 resulting from Bayesian inference. The denominator was introduced as the *marginal likelihood*, serving as a normalizing constant. Here, the expression is extended to include the dependency of a set of hyperparameters $\boldsymbol{\theta}$, given observations $\boldsymbol{y}$ from the input space X.

$$p(f \mid \boldsymbol{y}, X, \boldsymbol{\theta}) = \frac{p(\boldsymbol{y} \mid X, f)p(f \mid \boldsymbol{\theta})}{p(\boldsymbol{y} \mid X, \boldsymbol{\theta})} \tag{2.33}$$

The corresponding integral formulation follows

$$p(\boldsymbol{y} \mid X, \boldsymbol{\theta}) = \int p(\boldsymbol{y} \mid X, f)p(f \mid \boldsymbol{\theta})\, df. \tag{2.34}$$

Thus, the marginal likelihood can be understood as the probability of observing outputs y depending on the choice of hyperparameters $\boldsymbol{\theta}$. To compute that integral, one can exploit the fact that observations y are Gaussian, i.e. sampled from a normal distribution as $y \sim \mathcal{N}(0, K + \sigma_\epsilon^2 I)$ [18]. That avoids the explicit computation of the integral and allows the application of log to yield in

$$\log p(\boldsymbol{y} \mid X, \boldsymbol{\theta}) = \underbrace{-\frac{1}{2}\boldsymbol{y}^T K_{\boldsymbol{y}}^{-1}\boldsymbol{y}}_{a}\underbrace{-\frac{1}{2}\log|K_{\boldsymbol{y}}|}_{b}\underbrace{-\frac{n}{2}\log 2\pi}_{c}. \tag{2.35}$$

Note that the matrix K_y incorporates covariance matrix of noise-free observations K_f in addition to the observation noise by $K_y = K + \sigma^2 I$. Equation 2.35 is a pivotal expression, as it allows an ocular interpretation of the marginal likelihood. Term a is referred to as the *data-fit*, being solely dependent on the observations and the covariance in place. Term b represents the *complexity penalty* by depending only on training inputs through the determinant of the covariance matrix $|K_y|$. Term c can be considered as a normalizing constant [1].

To take advantage of that expression for hyperparameter selection, recall that the marginal likelihood expresses the probability of observations y given some hyperparameter set $\boldsymbol{\theta}$. It is thus intuitive to construct the partial derivatives of equation 2.35 in order to identify the optimal hyperparameters [1].

$$\begin{aligned}\frac{\partial}{\partial\theta_j}\log p(\boldsymbol{y} \mid X, \boldsymbol{\theta}) &= \frac{1}{2}\boldsymbol{y}^T K^{-1}\frac{\partial K}{\partial\theta_j}K^{-1}\boldsymbol{y} - \frac{1}{2}\mathrm{tr}\left(K^{-1}\frac{\partial K}{\partial\theta_j}\right)\\ &= \frac{1}{2}\mathrm{tr}\left((\alpha\alpha^T - K^{-1})\frac{\partial K}{\partial\theta_j}\right), \quad \text{where} \quad \alpha = K^{-1}\boldsymbol{y}\end{aligned} \tag{2.36}$$

The computational complexity is hereby dominated by the inverse of the covariance matrix K. However, gradient-based methods such as the `Adam` or the `L-BFGS` optimizers perform reasonably well [19].

2.2.2 Reproducing Kernel Hilbert Spaces in Gaussian Processes

Kernel hyperparameters were demonstrated as a tool to tune the predicted mean for increased matching of the true function when numerous observations are present. However, incorporating prior knowledge in the absence of numerous observations, the modeling of the true function becomes an 'ill-posed' problem. A hereby pivotal approach to overcome this issue results from the Reproducing Kernel Hilbert Space (RKHS) of a kernel which can be used to add assumptions about the true function by bounding the space of possible functions in a sophisticated way. Thus, the general concept of RKHS will be presented.

Following a general definition [1], assume $\mathcal{H}$ to be a Hilbert Space of real functions f defined on an input set $\mathcal{X}$. The Hilbert Space $\mathcal{H}$ is an RKHS with an inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ if there exists a function $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ with the properties:

1. For every x, $k(x, x')$ as a function of x' belongs to $\mathcal{H}$
2. k has the *reproducing property* $\langle f(\cdot), k(\cdot, x)\rangle_{\mathcal{H}} = f(x)$

Moreover, the *Moore-Aronszajn* theorem states that the kernel choice k uniquely determines the RKHS and vice versa [20]. Assuming both properties are given, the RKHS *norm* of a function f is given by

$$||f||_{\mathcal{H}} = \sqrt{\langle f, f \rangle_{\mathcal{H}}}. \tag{2.37}$$

For further analysis of the RKHS some kernel properties must be discussed for which *Mercer's Theorem* is introduced. Assume a positive definite kernel k that is continuous and square-integrable on a compact domain $\mathcal{X}$. An eigenfunction expansion as

$$k(x, x') = \sum_{i=1}^{\infty} \lambda_i \phi_i(x) \phi_i(x') \tag{2.38}$$

is allowed where

$$\int_{\mathcal{X}} k(x, x') \phi_i(x') \, dx' = \lambda_i \phi_i(x) \tag{2.39}$$

holds. Here, ϕ_i are the orthonormal eigenfunctions of k in $\mathcal{L}^2(\mathcal{X})$ and λ_i are the corresponding eigenvalues [21]. Now, consider the linear combination of eigenfunction of some real functions f and g.

$$f(x) = \sum_{i=1}^{N} f_i \phi_i(x), \quad g(x) = \sum_{i=1}^{N} g_i \phi_i(x) \tag{2.40}$$

Assuming a Hilbert Space $\mathcal{H}$ consisting of those two linear combinations of eigenfunctions, an inner product $\langle f, g \rangle_{\mathcal{H}}$ is defined.

$$\langle f, g \rangle_{\mathcal{H}} = \sum_{i=1}^{N} \frac{f_i g_i}{\lambda_i} \tag{2.41}$$

Note that for achieving a finite norm, i.e. $||f||_{\mathcal{H}} < \infty$ the sequence of coefficients f_i must decay quickly. Same holds for the norm of g. Practically, this requires a certain degree of smoothness on the space containing those functions. This can be exploited in the GP setting to add assumptions about the true function that is to be modelled. For that purpose, it must be shown that the constructed Hilbert space is the RKHS with the associated kernel k, thus the formerly stated properties must be given. To verify the reproducing property, the inner product of some function f from the Hilbert Space and the kernel k must yield in function f itself. This can be easily demonstrated as

$$\langle f(\cdot), k(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}} = \sum_{i=1}^{N} \frac{f_i \lambda_i \phi_i(\boldsymbol{x})}{\lambda_i} = f(\boldsymbol{x}). \tag{2.42}$$

holds. Consequently, the given Hilbert Space is the unique RKHS associated with kernel k. Referring back to the GP setting, the concept of RKHS is used to display the true function f as a weighted sum of the kernel k.

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x, x_i), \quad n \in \mathbb{N}, x_i \in \mathcal{X}, \alpha_i \in \mathbb{R} \tag{2.43}$$

Composing f in that way, the reproducing property is again fulfilled as

$$\begin{aligned}\langle k(\cdot, \boldsymbol{x}), f(\cdot)\rangle_{\mathcal{H}} &= \left\langle k(\cdot, x), \sum_{i=1}^{n} \alpha_i k(\cdot, x_i) \right\rangle_{\mathcal{H}} \\ &= \sum_{i=1}^{n} \alpha_i \langle k(\cdot, x), k(\cdot, x_i)\rangle_{\mathcal{H}} \\ &= \sum_{i=1}^{n} \alpha_i k(x, x_i)\end{aligned} \tag{2.44}$$

holds, recalling $\langle k(\cdot, x), k(\cdot, x_i)\rangle_{\mathcal{H}} = k(x, x_i)$. Referencing to the posterior mean function from 2.22 for a single observation translated to the noise-free case as

$$f_* = \boldsymbol{k}^T K^{-1} \boldsymbol{f}, \tag{2.45}$$

one can assert $\alpha = K^{-1}\boldsymbol{f}$ to yield $f_* = f(x_*)$ according to equation 2.43. Finally, the posterior mean function is within the RKHS of a kernel k used in the GP.

However, it is interesting to note that f has been considered a fixed, deterministic function. For comparison, let f be a sample of a zero-mean GP with kernel k as in equation 2.10. The eigenbasis expansion of k from equation 2.38 allows to define f sampled from a GP as

$$f(x) = \sum_{i=1}^{\infty} f_i \phi_i(x) \tag{2.46}$$

where the coefficients f_i are sampled from a normal distribution as

$$f_i \sim \mathcal{N}(0, \lambda_i). \tag{2.47}$$

Attempting to compute the expectation of the RKHS norm, one reaches

$$\mathbb{E}\Big[\|f\|_{\mathcal{H}}^2\Big] = \sum_{i=1}^{\infty} \frac{\mathbb{E}[f_i^2]}{\lambda_i} = \sum_{i=1}^{\infty} 1 \tag{2.48}$$

as the expectation of squared random variable is equal to its variance, hence $\mathbb{E}[f_i^2] = \lambda_i$. Finally, the expectation of the RKHS norm formally diverges, therefore f as a stochastic GP sample is not a member of the associated RKHS [1].

Later chapters will elaborate on the use of RKHS norm bounds of the true function f to scale uncertainty in optimization algorithms.

2.3 Bayesian Optimization

The general goal of any optimization problem is to efficiently allocate resources in order to identify optimal solutions as quickly as possible. A variety of methods for optimizing known functions exist such as gradient based methods, however optimization becomes more

challenging when the function to be optimized is not known. Bayesian Optimization (BO) has emerged as a promising approach to efficiently solve optimization problems assuming black-box functions, especially in cases where evaluations are costly to compute [22]. For this purpose, BO determines inputs whose output is expected to reveal valuable information. Those inputs are then evaluated revealing more knowledge about the underlying function. Consequently, for an efficient BO algorithm, it is desired to keep the amount of evaluated sub-optimal solution minimal.

To further elaborate on the principles of BO, a simple optimization problem statement follows. Let f be some real, unknown function defined on the input space $\mathcal{X}$ upon which it is sought to maximize.

$$\max_{x \in \mathcal{X}} f(x) \tag{2.49}$$

As this is a widely defined optimization problem, some restrictions will be applied for the remainder of this work. First, assume f maps one-dimensional $\mathbb{R}$-valued inputs from the compact set $\mathcal{X}$ to $\mathbb{R}$-valued, one-dimensional outputs.

$$f : \mathcal{X} \to \mathbb{R}, \mathcal{X} \subset \mathbb{R}^d. \tag{2.50}$$

Second, assume Lipschitz continuity of f, meaning that a constant $L \geq 0$ exists for which

$$\forall x, x' \in \mathcal{X}, |f(x) - f(x')| \leq L \cdot ||x - x'|| \tag{2.51}$$

holds. This allows to state that a solution x^* exists which is the supremum of f [23].

$$\exists x^* \in \mathcal{X} \quad \text{s.t.} \quad f(x^*) = \sup_{x \in \mathcal{X}} f(x) \tag{2.52}$$

For solving the optimization problem of equation 2.49 within a finite amount of sub-optimal evaluations, ξ-optimality is considered. Thus, the solution x^* is considered as optimal when the distance between the real optimal value $f(x_{opt})$ and the computed optimal value $f(x^*)$ is below some threshold ξ.

$$|f(x_{opt}) - f(x^*)| < \xi \tag{2.53}$$

Nonetheless, the amount of evaluations can still be considerably large applying classical algorithms [23]. The power of BO lays in reducing the amount of evaluations by modeling the function itself rather than solely searching for the optimal solution. This is done by stating prior beliefs about the underlying function and generating observations of its behavior via input evaluations. Moreover, laws of probability are exploited to combine the prior beliefs and observations to update the generated model of the function. A BO algorithm is mainly characterized by the type of probabilistic surrogate model and the acquisition function that queries the next sample for evaluation to reveal the best information required for further improving the model. GPs rose to the most prominent

type of surrogate models in BO. The choice of acquisition functions however is more diverse and influences dominantly the performance of BO algorithms. Some are now presented.

Maximum Probability Improvement

Primary work on BO attempted to optimize a one-dimensional, $\mathbb{R}$-valued function by using a specific type of GPs called *Wiener process* to model the unknown function [24]. The chosen acquisition function analyzes the posterior probability that the function value of the next observation is greater than the current best, i.e. highest observation. Note that the magnitude of improvement however is not considered [22]. Originally, no noise-perturbed observations were assumed but the availability of exact function values. Here, the acquisition is modified to consider noisy observations $y_1, ..., y_n$ at corresponding input points $x_1, ..., x_n$ with y_{max} being the highest observation. The *improvement* I of some input x can be given by

$$I(x) = \max(y(x) - y_{max}, 0). \tag{2.54}$$

Recalling that the true function is modeled by a surrogate GP model allows to state that each observation is a sample from a normal distribution whose posterior mean $\mu_n(x)$ and variance $\sigma_n^2(x)$ are computed by the surrogate model.

$$y_n(x) \sim \mathcal{N}(\mu_n(x), \sigma_n^2(x)) \tag{2.55}$$

The *maximum probability of improvement* (MPI) acquisition criterion now states that the subsequent sample input x_{n+1} is chosen which exhibits the largest probability on increasing the highest observation so far

$$x_{n+1} = \arg\max_z P[Y_z \geq y_{max} + \zeta(n)]. \tag{2.56}$$

Hereby, Y_z denotes all observations that have not been made yet. Applying the standard normal cumulative density function $\Phi(z_0)$, with $z_0 = (y_{max} - \mu_n(x))/\sigma_n(x)$ to evaluate this probability [25], one can compute the subsequent query point x_{n+1} as

$$x_{n+1} = \arg\min_x \left(\frac{(y_{max} - \mu_n(x) + \zeta(n))^2}{\sigma_n(x)^2} \right), i = 0, ..., n. \tag{2.57}$$

The iteration-dependent constant $\zeta(n)$ can be set to balance the trade-off between choosing inputs with large uncertainty, or those with large posterior mean values [25]. If $\zeta(n) \to \infty$, the criterion is reduced to choosing the input of largest uncertainty. Setting $\zeta(n) \to 0$ yields in sampling inputs with largest expected observation.
Nonetheless, the heuristic choice of $\zeta(n)$ often leads to aggressive exploitation around the current best observation such that the algorithm can get easily stuck in local optima [2].

Expected Improvement

Further work on relevant acquisition function has been conducted by applying the *Expected Improvement* (EI) criterion [26]. It rises from the idea that MPI only considers the probability of a function value being closer to the current best observation but not the extent of improvement. The (EI) however, maximizes the amount of improvement, not only maximizing the probability of arbitrary improvement. Following the formulation of improvement in equation 2.54, the expected improvement $EI(x)$ of some input x is

$$EI(x) = \mathbb{E}[I(x)|\boldsymbol{y}], \tag{2.58}$$

with vector $\boldsymbol{y}$ containing all observations. Similarly to the approach in MPI, one can apply the density and cumulative distribution functions $\phi(x)$, $\Phi(x)$ respectively [27], to yield in

$$EI(x) = (y(x) - y_{max})\Phi\Big(\frac{y(x) - y_{max}}{\sigma_n^2}\Big) + \sigma_n^2\phi\Big(\frac{y(x) - y_{max}}{\sigma_n^2}\Big) \tag{2.59}$$

where y_{max} denotes the best observations so far. The subsequent query point x_{n+1} is thereby

$$x_{n+1} = \arg\max_x EI(x). \tag{2.60}$$

Finally, the EI acquisition function exhibits better performance in balancing the choice of samples with large uncertainty or large posterior mean [28]. In contrast to MPI that is dependent on heuristic parameter settings, convergence rates for EI acquisition functions have been proven [29]. However, MPI can outperform EI in settings where given initial samples are close to the global optimum [30].

Upper Confidence Bound

Interesting results from research on the so-called *Multiarmed Bandit (MAB)* problem ([31], [32]) have built the ground for major advancements in BO. The MAB problem originates from the idea of playing on multiple slot machines where each is considered as one arm, not knowing the individual reward distributions. The goal is to maximize the cumulative reward over a finite time horizon by always choosing the machine with highest reward considering previous observations of reward distributions which are initially unknown.

In that context, the *Exploration-Exploitation dilemma* was stated, pointing at the trade-off between choosing a machine which has returned high rewards in previous observations (exploitation) and those whose reward distributions are less known (exploration) [31]. Hence, there will be a regret $R(T)$ which measures the difference between the optimal decision strategy and the one in place after T iterations [33].

This has been the basis for the introduction of the *Upper Confidence Bound* (UCB) decision policy ([34]) based on the *Upper Confidence Index* [35]. For a play machine j with average

reward μ_j based on previous observations, the next machine is determined by maximizing the expression

$$\mu_j + \sqrt{\frac{2 \cdot ln(n)}{n_j}}$$

where n depicts the amount of overall plays and n_j the amount of plays on machine j.

This paved the way for the GP-UCB method [7] which translates UCB to GP-based BO. The question which arm to choose from the MAB problem becomes the question which query point to evaluate on the continuous, unknown function f to optimize it as quickly as possible, given some observations. The maximizing of the standard deviation $\sigma(x)$ is initially assumed

$$x_t = \arg\max_x \sigma(x). \tag{2.61}$$

That however is considered to be wasteful since global uncertainty is decreased and not particularly for potential maxima. Another approach is to maximize along the expected reward which is equal to evaluating inputs with high mean values $\mu(x)$, aiming to increase the highest observation so far.

$$x_t = \arg\max \mu(x) \tag{2.62}$$

As stated afore, this results again in convergence to local optima. To overcome both issues, GP-UCB combines both approaches from equations 2.61 and 2.62 by

$$x_t = \arg\max \mu(x) + \beta^{1/2}\sigma(x). \tag{2.63}$$

To decide on whether to maximize the mean or the standard deviation illustrate the exploration-exploitation dilemma. Therefore, the input with the highest upper confidence bound, dependent on $\beta^{1/2}$ is chosen for evaluation and added to the surrogate model. The scaling term β can be used to either stress exploration or exploitation. Large β correspond to increasing the influence of uncertainty, hence focusing on exploration whereas small β decrease the influence of uncertainty, prioritizing large observations. The GP-UCB method has shown favorable theoretical guarantees such as bounds on cumulative regret [7] in sequential decision-making problems or even on contextual regret [36]. In addition to its capability of tuning the exploration-exploitation dilemma, it theoretically excels MPI and EI. However, empirical studies have shown that EI is capable of outperforming GP-UCB in a direct optimization task [37].

In conclusion, acquisition functions dominate the efficiency of BO algorithms by exhibiting drawbacks linked to the exploration-exploitation dilemma.

2.4 Safe Exploration in Bayesian Optimization

The relevance of a suitable choice of acquisition function has been shown to be pivotal. However, their applicability to constraint optimization problems has not been highlighted yet.

The optimization problem formulated in 2.49 considers the whole input space when optimizing the underlying function f. However, in many practical applications this is not the case as some input may yield in unstable or undesired behavior. An apparent example is the field of robotics. Let f be some sort of performance measure of a robot and let $\boldsymbol{x} \in \mathbb{R}^d$ be the control parameters of that robot. Control parameters are known to have limitation w.r.t stability, reducing the set of possible values $\boldsymbol{x}$. Not only robotic applications are subject this type of constraints [38] but also BO applications in the medical field, e.g. for finding optimal parameters for Deep Brain Stimulation [5]. Recall that the relation between the performance measure and the optimal control parameters is unknown a priori. Otherwise, one could simply reduce the considered input space by the known, unsafe set of parameters. Hence, safety constraints need to be formulated. Simplifying to the one-dimensional case, those can be stated by

$$g_i(x) \geq 0, x \in \mathbb{R}^d, i = 1, ..., q \tag{2.64}$$

where the function g describes context dependent constraints. Notice that by adding constants, any constraint satisfaction can be imposed as $g_i(x) \geq 0$. Thus the algorithm considers solely inputs x that satisfy this equation. The constrained optimization problem can then be formulated as

$$\max_{x \in \mathcal{X}} f(x) \quad \text{subject to} \quad g_i(x) \geq 0 \quad \forall i = 1, ..., q. \tag{2.65}$$

An illustrative example can be given by assuming a BO setting where the underlying true function f is modeled by a GP as

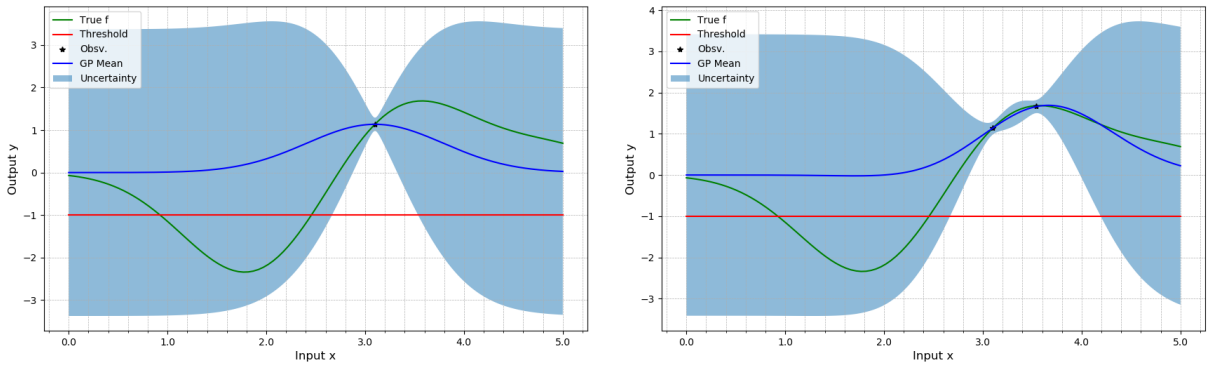
$$f(x) \sim \mathcal{GP}\left(\mu(x), k(x, x')\right) \tag{2.66}$$

where the mean $\mu(x)$ is initially set to zero and the squared exponential kernel from equation 2.12 is chosen as the covariance function k. Furthermore, let h be a safety threshold by stating that all evaluated inputs must yield in observations higher than h. Thus, the safety constraint function g can be given by

$$g(x) = f(x) - h \geq 0, \quad h \in \mathbb{R}. \tag{2.67}$$

Given an initial safe input $x = 3.1$, the predictive distribution is shown in figure 2.3a.

The red line represents the imposed constraint $h = -1$, meaning that no input x_i can be chosen which potentially yields in an observation $y(x_i) < h$. The green function represents the ground truth which is to be optimized. An example of a subsequent observation that satisfies the constraint is shown in 2.3b, exhibiting a second observation at $x = 3.54$



(a) Predictive distribution after one observation (b) Predictive distribution after two observation

Figure 2.3: Gaussian Process based Safe Bayesian Optimization with two subsequent observations

that is well above the safety threshold h. Assume the blue shaded area to represent the uncertainty, constructed by scaling the standard deviation from the GP at each input.

By presenting an example like that, the obvious question of how to choose safe, subsequent query points when their true function value is unknown, arises. First, it must be assured that the uncertainty width includes the true function as seen in figure 2.3. Second, the acquisition function responsible for querying the new sample input must choose inputs whose true, unknown output is above the safety threshold. The following chapter will elaborate on strategies to fulfill these two requirements.

3 Methodology

This chapter discusses approaches to assure the true functions containment within the uncertainty bounds and associated algorithms to query safe input values. First, prominent algorithms for safety-critical BO are presented. Second, the mentioned scaling of the standard deviation from the GP in section 2.4 will be elaborated on. Finally, efficient practical implementations of safe BO algorithms with varying uncertainty bound construction will be demonstrated.

3.1 Considered Algorithms

Various algorithmic approaches have been discussed for general Bayesian Optimization in section 2.3. It has been shown that general BO incorporating the EI acquisition function outperforms implementations with MPI in most cases. However, both do not incorporate safety constraints making them unsuitable for further consideration. `GP-UCB` excels both approaches in terms of handling the exploration-exploitation dilemma, however the afore presented version of `GP-UCB` did not consider safety constraints either. The first seminal work on Safe Bayesian Optimization using GPs is given by the development of the `SafeOpt` algorithm [3]. It exhibits safety constraint satisfaction while properly handling the exploration-exploitation dilemma. Moreover, the work on `SafeOpt` came with an extension of the `GP-UCB` algorithm from section 2.3 to handling safety constraints, called `Safe-UCB`. The `SafeOpt` algorithm exhibited better convergence than `Safe-UCB` in an experimental setting [3]. However, as this work aims at utilizing different uncertainty scaling strategies, the convergence of `Safe-UCB` and `SafeOpt` will be compared.

3.1.1 SafeUCB Algorithm

First, the `Safe-UCB` algorithm is presented as an extension of `GP-UCB` with additional safety constraint handling. It aims at maximizing some unknown function f without sampling inputs whose corresponding output is below the safety threshold. Thus, let $f : \mathcal{X} \to \mathbb{R}$ be the unknown function (ground truth) with d-dimensional inputs $\boldsymbol{x} \in \mathbb{R}^d$. Let $h \in \mathbb{R}$ be some constant representing the safety-threshold that must not be violated, then the constrained optimization problem can be written as

$$\max_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}) \quad \text{subject to} \quad y(\boldsymbol{x}) \geq h. \tag{3.1}$$

Note that no precise function values $f(\boldsymbol{x})$ can be determined but only noise perturbed observations $y(\boldsymbol{x}) = f(\boldsymbol{x}) + \epsilon$. It is desired to optimize f by sequentially querying safe inputs to learn the ground truth. The surrogate model for the ground truth f is a Gaussian Process, specifying mean and variance for all inputs at each iteration n. Given is a set of n observations $\mathcal{D}_n = \{\boldsymbol{x}_i, y_i\}_{i=1}^n$, the input space $\mathcal{X}$ and a GP defined by

$$f(\boldsymbol{x}) \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x'})), \quad \boldsymbol{x}, \boldsymbol{x'} \in \mathcal{X}. \tag{3.2}$$

Compliant with predictive distributions given in equation 2.20, the mean and variance of the predicted output of input $\boldsymbol{x}^*$ at iteration n is given by

$$\begin{aligned}\mu_n(\boldsymbol{x}^*) &= \boldsymbol{k}_n(\boldsymbol{x}^*)\Big(K_n + I_n\sigma_\epsilon^2\Big)^{-1}\boldsymbol{y}_n,\\ \sigma_n^2(\boldsymbol{x}^*) &= k(\boldsymbol{x}^*,\boldsymbol{x}^*) - \boldsymbol{k}_n(x^*)^\top\Big(K_n + I_n\sigma_\epsilon^2\Big)^{-1}\boldsymbol{k}_n(\boldsymbol{x}^*).\end{aligned} \tag{3.3}$$

Note that the gram matrix K_n carries the entries $[K_n]_{(i,j)} = k(\boldsymbol{x}_i, \boldsymbol{x}_j), i, j \in 1, ..., n$. Moreover, the vector $\boldsymbol{k}_n(\boldsymbol{x}^*) = [k(\boldsymbol{x}^*, \boldsymbol{x}_1), ..., k(\boldsymbol{x}^*, \boldsymbol{x}_n)]$ is composed by the covariances between the subsequent input $\boldsymbol{x}^*$ and the training data.

As mentioned earlier, any algorithm that wishes to optimize an unknown function in a safety-critical setting requires an initial set of safe inputs which is hereby denoted as $\mathcal{S}_0$. Throughout the optimization process, this set needs to be expanded in order to identify the global maximum as it is unlikely to be given in $\mathbb{S}_0$. To do so, a classification of inputs whose unseen observations are expected to be safe is necessary. This is achieved by the help of GP predictive distributions. Let u_n and l_n be upper and lower bounds, respectively, derived from the GP as

$$\begin{aligned}u_n(\boldsymbol{x}) &= \mu_n(\boldsymbol{x}) + \beta_n\sigma_n(\boldsymbol{x}),\\ l_n(\boldsymbol{x}) &= \mu_n(\boldsymbol{x}) - \beta_n\sigma_n(\boldsymbol{x}),\end{aligned} \tag{3.4}$$

where β is some scaling factor that is defined later. An input $\boldsymbol{x}$ can then be classified as safe with high probability if its lower bound is above the required safety threshold h. Hence, the safe set after n iterations is given by

$$\mathcal{S}_n = \Big\{\boldsymbol{x} \in \mathcal{X} \quad | \quad l_n(\boldsymbol{x}) \geq h\Big\}. \tag{3.5}$$

Sampling a query point $\boldsymbol{x}^*$ from this set will yield in a safe observation with high probability. To graphically demonstrate the safe set, see figure 3.1 where a one dimensional ground truth f is given.

The green bar on the input axis indicates the safe set $\mathcal{S}_1$ that is constructed according to equation 3.5 after computing the predictive distribution based on the initial safe set $\mathcal{S}_0 = [6.5, 7.0]$. The lower bounds of inputs from that set are all above or equal to the safety threshold. Thereby, the most conservative output prediction of the so far unseen inputs is considered to classify safeness.

However, sampling from that safe set is not sufficient for the goal of optimization. Hence, among the safe set, the next query point is chosen to be the one that maximizes the upper confidence bound. From a GP with mean and covariance functions as in equation 3.3, the next query point x_n is given by

$$\boldsymbol{x_n} = \arg\max_{\boldsymbol{x}} \mu_{n-1}(\boldsymbol{x}) + \beta^{1/2}\sigma_{n-1}(\boldsymbol{x}). \tag{3.6}$$

The choice of the scaling factor β is crucial for the containment of the true function in the

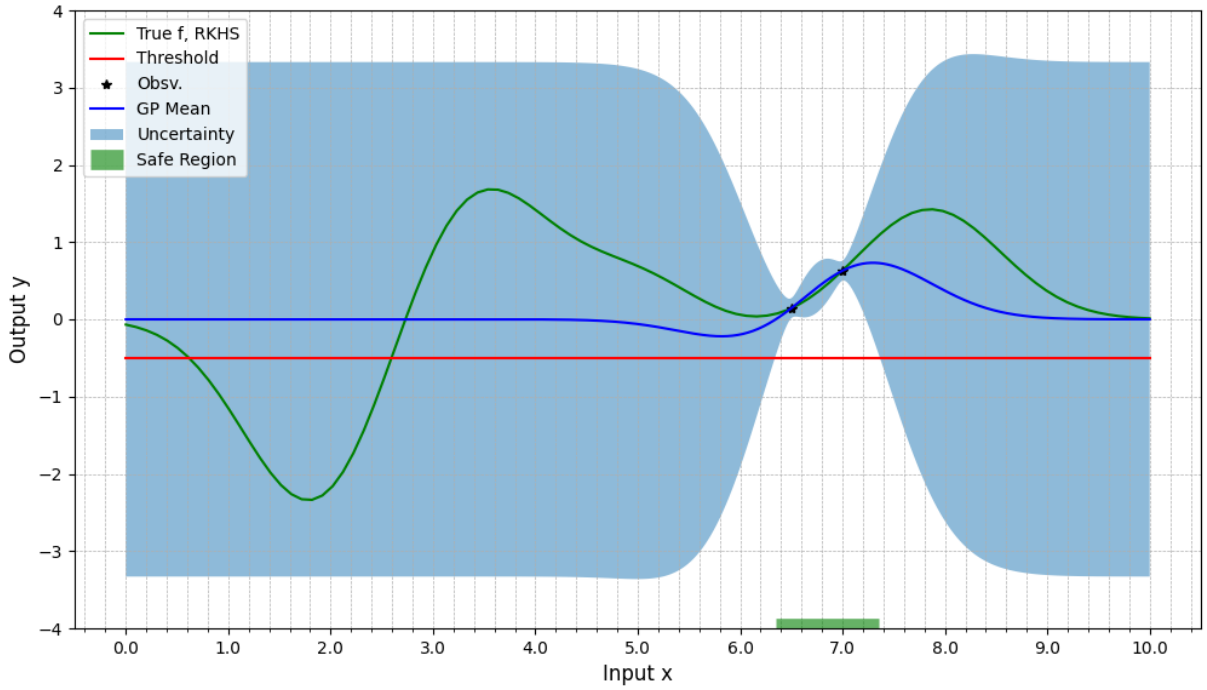
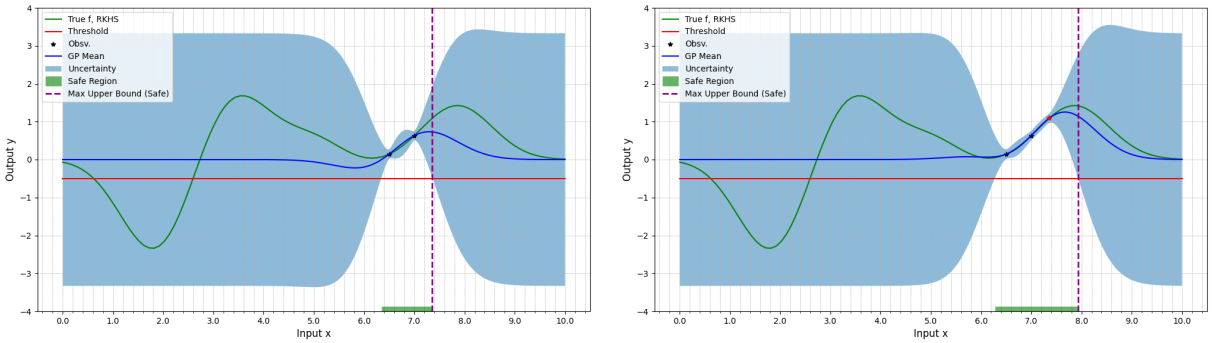


Figure 3.1: Predictive Distribution of Initial Safe Set for `SafeUCB`

uncertainty bounds. Its computation will be elaborated on in further sections, assuming β to be given for now.

Figure 3.2 depicts the safe set in addition to the sampling criterion. The dotted, vertical line points at the input that exhibits the largest upper bound among the safe set. The observation marked by the red star in figure 3.2b corresponds thereby to the input indicated by the vertical line in figure 3.2a. The next input that would be evaluated in iteration 2 is henceforth $x_2 = 7.92$ as seen in figure 3.2b.



(a) Safe Sampling Choice from Iteration 0 (b) Safe Sampling Choice from Iteration 1

Figure 3.2: Indicated Safe Set and Sampling Choice according to `SafeUCB`

The major drawback of `SafeUCB` is its lack to handle exploration as it solely maximizes the upper bound. Therefore, it can easily get stuck in a local maximum as the algorithm

cannot query inputs which low predicted observations which however would expand the safe set more towards the global maximum at $x = 3.54$. This issue will be elaborately addressed and demonstrated in chapter 5.

3.1.2 `SafeOpt` Algorithm

Previous sections have demonstrated the need for a BO algorithm that efficiently optimizes an unknown function f satisfying safety constraints and handling the exploration-exploitation dilemma. The `SafeOpt` algorithm is derived from the well-known problem of sequential decision-making under uncertainty when optimizing an unknown function [3]. Safety, as well as noise perturbed observations are considered allowing the formulation of a constrained optimization problem as in equation 2.65.

To give a mathematical description of the `SafeOpt` algorithm, this work follows the modified version of `SafeOpt` [38] as the dependency of Lipschitz constants from the initial publication [3] is omitted, simplifying the required classification of inputs which then can be directly drawn from the underlying GP surrogate model.

The optimization problem that is solved by `SafeOpt` is equal to the one for `SafeUCB`, hence consider

$$\max_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}) \quad \text{subject to} \quad y(\boldsymbol{x}) \geq h. \tag{3.7}$$

Moreover, the same formulation for the underlying GP model, the predictive distributions and the formulation of lower and upper bounds hold as in equations 3.2, 3.3, 3.4 respectively.

Classification of safe, maximiser and expander sets

First, the `SafeOpt` algorithm requires a classification of the safeness of inputs. This is handled equivalently as for `SafeUCB`, hence

$$\mathcal{S}_n = \Big\{ \boldsymbol{x} \in \mathcal{X} \quad | \quad l_n(\boldsymbol{x}) \geq h \Big\}. \tag{3.8}$$

However, it is not possible to assure convergence to an optimum by simply sampling from the safe set, hence further classifications are made.
To exploit knowledge about safe inputs with predicted uncertainty bounds, inputs are chosen which are expected to obtain high observations. They are called *maximizers*, i.e. those inputs whose upper bound is larger than the largest lower bound within the safe set.

$$\mathcal{M}_n = \Big\{ \boldsymbol{x} \in \mathcal{S}_n | u_n(\boldsymbol{x}) \geq \max_{\boldsymbol{x'} \in \mathcal{X}} l_n(\boldsymbol{x'}) \Big\} \tag{3.9}$$

Figure 3.3 depicts the predictive distribution of some random initial safe set $\mathcal{S}_0 = [6.3, 6.5, 6.6, 7.0, 7.5]$ with indicated safe set and maximizer set. Note that the largest lower bound corresponds to $l_0(x = 7.5) = 1.12$. Thus, all inputs with an upper bound above that value are contained in the maximizer set.

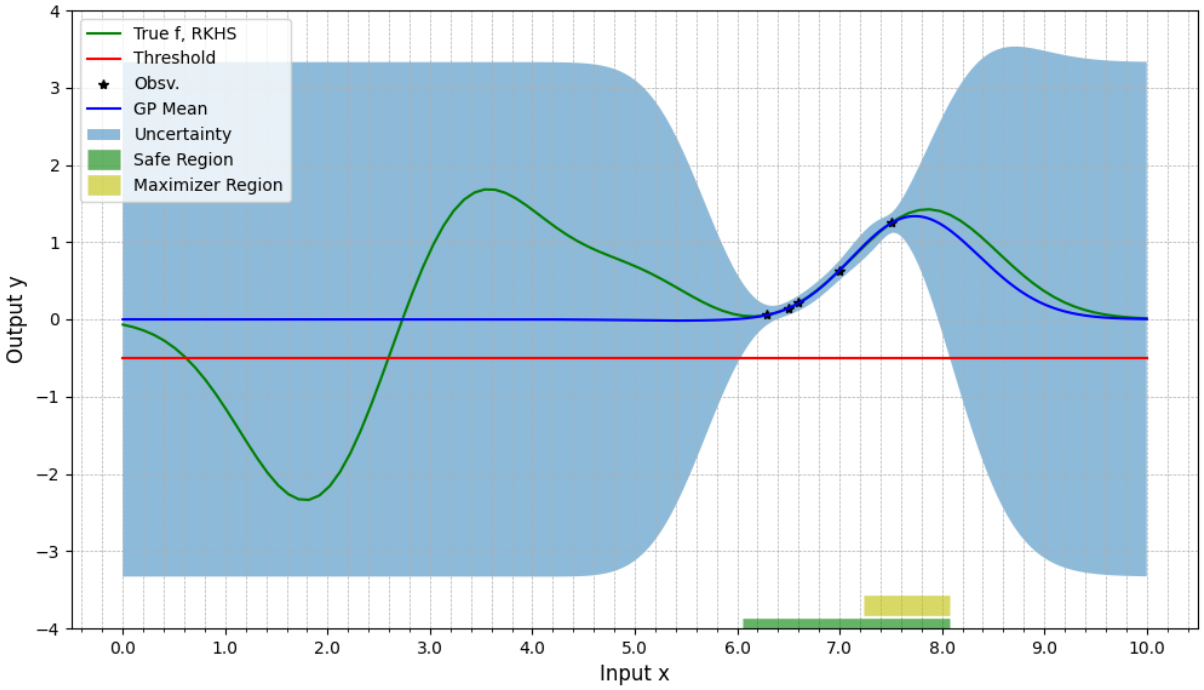


Figure 3.3: Safe Set and Maximizer Set from a Predictive Distribution for `SafeOpt`

As exposed earlier, methods that predominantly aim at maximizing the output value and neglecting expansion of safe sets, may get stuck in local optima. For a more advanced optimization result, a third set is introduced. While the maximizer set accounts for exploitation, the *expander* set $\mathcal{G}_n$ incorporates exploration by containing all safe inputs which increase the size of the safe set with high probability. Mathematically, this can be expressed by

$$\begin{aligned} \mathcal{G}_n &= \Big\{ \boldsymbol{x} \in \mathcal{S}_n \quad | \quad g_n(\boldsymbol{x}) > 0 \Big\}, \\ g_n(\boldsymbol{x}) &= |\boldsymbol{x'} \in \mathcal{X} \notin \mathcal{S}_n \quad | \quad l_{n,(\boldsymbol{x},u_n(\boldsymbol{x}))}(\boldsymbol{x'}) \geq h|. \end{aligned} \tag{3.10}$$

To explain the principle of this characterization, see figure 3.4. The purple bars above the the x axis indicate the set of inputs that are classified as expanders. The predictive distribution is computed based on the same initial safe set as used for the maximizer demonstration.

If one adds the upper bound of an input from that set to the GP model as an observation, there will exists some input $\boldsymbol{x}'$ that has been unsafe before the enlargement of the model and became safe thereby. The expression $l_{n,(\boldsymbol{x},u_n(\boldsymbol{x}))}(\boldsymbol{x'})$ from equation 3.10 representing the lower bound of an unsafe input became larger than the safety threshold, thus safe due to adding the upper bound of a safe input which hence classifies as an expander. Note that the expander set from figure 3.4 is disconnected since inputs which are located in the vicinity of already observed inputs are unlikely to further expand the safe set.

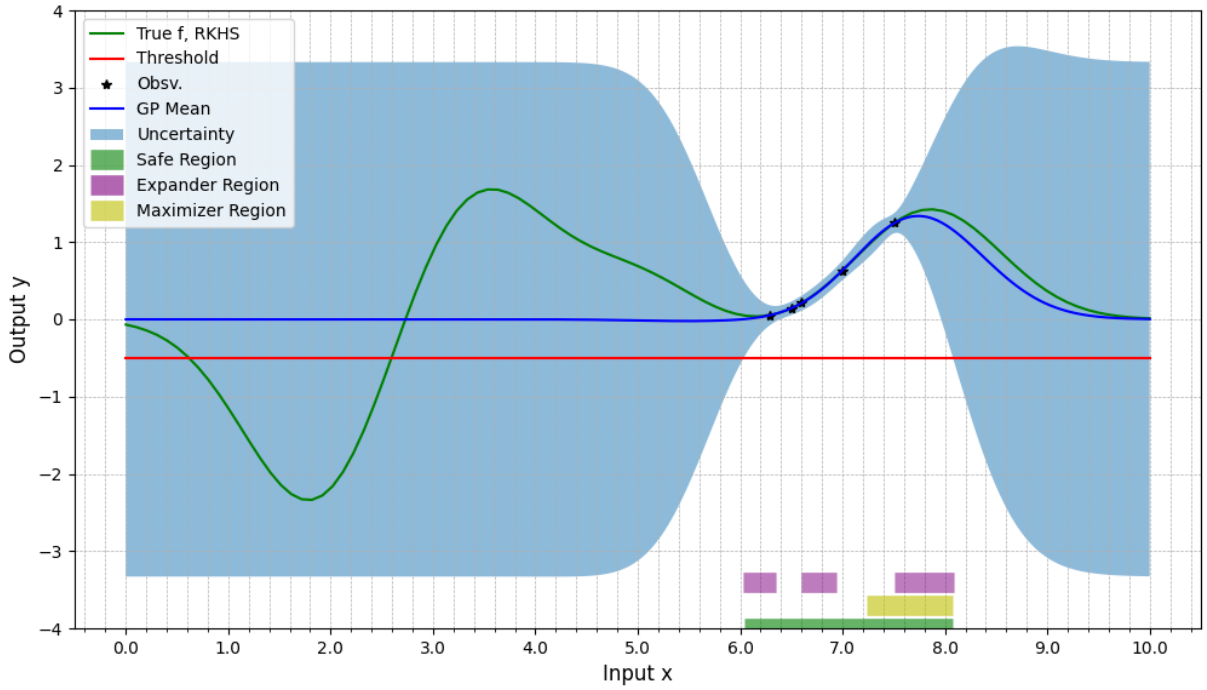


Figure 3.4: Safe, Maximizer and Expander from a Predictive Distribution for `SafeOpt`

Sampling criterion

After constructing those sets, a query point $\boldsymbol{x}_n$ is determined. The sampling criterion of the `SafeOpt` algorithm suggests to choose the input with most uncertainty among the union set of maximizers and expanders. Since those two can only contain safe inputs, safety constraints are fulfilled. Finally, the function f will be evaluated at input $\boldsymbol{x_n}$ according to

$$\begin{aligned} \boldsymbol{x}_n &= \underset{\boldsymbol{x} \in \mathcal{G}_n \cup \mathcal{M}_n}{\arg\max}\, w_n(\boldsymbol{x}), \\ w_n(\boldsymbol{x}) &= u_n(\boldsymbol{x}) - l_n(\boldsymbol{x}). \end{aligned} \tag{3.11}$$

3.2 Construction of Uncertainty Bounds

The size of uncertainty bounds during a safe BO application is key to the efficiency of the algorithm. Large uncertainty bounds, i.e. conservative predictions reduce the safe set, requiring the algorithm to query more inputs for optimizing the ground truth, while smaller ones accelerate the optimization by potentially violating safety constraints. Particularly relevant for Safe Bayesian Optimization are theoretical guarantees that the ground truth is contained in the uncertainty bounds, such that no unsafe inputs are chosen for evaluation. The afore mentioned scaling factor β that functions as a weight on the standard deviation when constructing upper and lower bounds according to equation 3.4 is thereby crucial.

Existing methods to construct uncertainty bounds strongly depend on the assumptions

made about the ground truth. This work is focussed on demonstrating and comparing two prominent ones, the frequentist and bayesian view. Frequentist approaches assume the ground truth f to be deterministic, fixed but unknown. Moreover, it is assumed that this function exhibits some RKHS norm that is bounded by some positive real constant B which can be used in various ways to compute error bounds.

Bayesian settings however assume f to be a sample from a GP with some kernel k. Thus, the ground truth is not considered to be deterministic and fixed. Another important assumption amongst others is the Lipschitz continuity of the kernel and the ground truth from which probabilistic uniform error bounds can be constructed.

The resulting uncertainty bound construction under both views will be presented in the following.

A derivation of β for the `SafeOpt` algorithm [3] is realized by the use of the maximum information gain γ_n [7] which quantifies the reduction in uncertainty by revealing observations of the ground truth. This frequentist approach assumes the true function f to exhibit a bounded RKHS norm B w.r.t. the chosen kernel of the underlying GP.

Another approach of scaling uncertainty results from the assumption that the unknown function has some bound B on the RKHS norm induced by the chosen kernel. The uncertainty width is then computed by a B-dependent scaling factor β_n and the standard deviation σ_n of the GP at each input.

Further work on uncertainty scaling is based on uniform error bounds resulting from Lipschitz continuity assumption for the kernel and thereby also for the posterior mean function. Those three strategies are presented and compared in the following.

3.2.1 Frequentist Uncertainty Bounds

The uncertainty bound construction from the initially proposed `SafeOpt` algorithm [3] is realized by the use of the maximum information gain γ_n [7] which quantifies the reduction in uncertainty by revealing observations of the ground truth. Moreover, some RKHS norm bound B of the true function is implemented which indicates that a frequentist approach assuming some fixed but unknown deterministic true function f is in place. However, the paper erroneously states that f is a sample from a GP [3] which is not possible since sample functions from GPs do not exhibit finite RKHS norms as demonstrated in chapter 2. It is important to note that true functions analyzed in most learning-based control settings are of deterministic nature which is compliant with the frequentist view where uncertainty is solely introduced via data generation (observations) and not due to randomness in the underlying physical system itself. In what follows, different frequentist strategies to bound the error of predictions from a GP are described. All of them are based on Assumption 1.

Assumption 1 *The true function $f : \mathcal{X} \mapsto \mathbb{R}$ with $\mathcal{X} \in \mathbb{R}^d$ is a member of the RKHS and the RKHS norm of f is bounded by some positive real constant B, such that $f \in RKHS, ||f||_{\mathcal{H}} \leq B$ with k being the associated kernel from the GP model.*

First, the approach of bounding uncertainty from the initial `SafeOpt` proposal is presented

[3]. The aforementioned information gain γ_t originates from information theory. When an unknown objective f is to be maximized given some observations $\boldsymbol{y}$, the informativeness of a set of potential sampling points is considered for which the *information gain* can be used. It describes the *mutual information*

$$I(\boldsymbol{y}; f) = H(\boldsymbol{y}) - H(\boldsymbol{y}|f) \tag{3.12}$$

between observations and the unknown function [7]. The entropy $H(X)$ measures the uncertainty of some random variable X [39]. Thus, equation 3.12 quantifies the reduction of uncertainty about the unknown function f when observations $\boldsymbol{y}$ become available.

Now, assuming that f is modeled by a GP, then its observations $\boldsymbol{y}$ are each a random variable from a normal distribution.

$$\begin{aligned} f(\boldsymbol{x}) &\sim \mathcal{GP}(\mu(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')) \\ \boldsymbol{y} &\sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \end{aligned} \tag{3.13}$$

The entropy H of observations $\boldsymbol{y}$ is then

$$H(\boldsymbol{y}) = \frac{1}{2}\log|2\pi\mathrm{e}\boldsymbol{\Sigma}|, \tag{3.14}$$

allowing the formulation of mutual information as

$$I(\boldsymbol{y}; f) = \frac{1}{2}\log|I + \sigma_\epsilon^{-2}K|. \tag{3.15}$$

The matrix K contains the covariances $k(\boldsymbol{x}, \boldsymbol{x}')$ whereas σ represents the assumptions about noisy observations.

Finally, assume the availability of a set of observations $\boldsymbol{y}_A \in \mathcal{A}$. The information gain is dependent on the sample size N and quantifies the maximally obtainable mutual information from N observations.

$$\gamma_n = \max_{|\mathcal{A}| \leq n} I(\boldsymbol{y}_A; f) \tag{3.16}$$

One reason why this has been considered an efficient approach, is the provability of a bounded information gain γ_t ([7], [3]), given a finite input set. The resulting uncertainty bound scaling is then given by

$$\beta_n = 2B + 300\gamma_n \log^3(n/\delta). \tag{3.17}$$

Here, B is the bound on the RKHS norm of ground truth f, associated by the kernel k and $1 - \delta$ is the probability that the ground truth is contained in such uncertainty bounds.

It is important to note that the maximum information gain exhibits rather conservative behavior. In safe BO, observations are collected in order to learn the ground truth by

expanding the safe set and exploiting the current best observations. As the information gain quantifies the reduction of uncertainty after new observations are provided, the maximum information gain corresponds always to the observation that achieved the highest reduction of uncertainty. However, throughout the execution of a safe BO algorithm like `SafeOpt`, there are not only observations added that drastically reduce uncertainty. Hence, conservatism is increased as the maximum information gain among all observations is applied to construct the uncertainty bounds after an observation which itself exhibits a low information gain. Despite the existence of upper bounds on the information gain, the scaling factor β still grows with sample size. Therefore, uncertainty bounds are overestimated and the efficiency of the safe BO algorithm suffers seriously.

One way to improve the results from the maximum information gain usage is to incorporate self-normalizing concentration inequalities [9]. Those refine the maximum information gain by a log-determinant expression of the gram matrix resulting from training data, introducing real data-dependency in contrast to the maximum information gain.

However, practical control applications have shown that the dependency on information gains as in [7] or even in the significantly improved case of [9] is too conservative for algorithmic use in control applications [40] which often leads to a heuristic choice of scaling [4]. Considering strict safety requirements, heuristically chosen scaling factors for uncertainty bounds destroy all theoretical safety guarantees. Hence, other ways of construction uncertainty bounds need to be considered for reliably ensuring that f is contained in the predictions such that no unsafe input is chosen for evaluation.

Another existing frequentist approach is able to increase upon conservatism and practical computability by further developing the use of concentration inequalities to generate practical and rigorous uncertainty bounds [8]. The work is motivated by the need of an error bound function $\nu(\boldsymbol{x})$ from which the error for any input $\boldsymbol{x}$ can be computed.

$$|f(\boldsymbol{x}) - \mu(\boldsymbol{x})| \leq \nu(\boldsymbol{x}),\ \forall \boldsymbol{x} \in \mathcal{X}. \tag{3.18}$$

The posterior mean and variance function is denoted by $\mu(\boldsymbol{x})$ and $\sigma^2(\boldsymbol{x})$, respectively. The error itself is simply defined by the difference of the true function value $f(\boldsymbol{x})$ and its predictive mean from the GP.

To construct the function $\nu(\boldsymbol{x})$, some assumptions must be discussed. One of the key assumptions considers the true function f to be a member of the RKHS associated by the kernel k from the GP model in place. Moreover, the RKHS norm of the true function is assumed to be bounded by some positive constant, i.e. $\|f\|_k \leq B$. However, it should be noted that determining an RKHS upper bound B that handles the trade-off between large error bounds for ensuring the containment of the true function and low error bounds for reducing conservatism skillfully, is a difficult task. So far, there are no generalized strategic methods to compute B based on prior knowledge of the unknown true function.

To present the novel frequentist error bounds, consider a zero-mean GP with covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$ and training data $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N], \forall \boldsymbol{x} \in \mathcal{X}$. Recall that assumption 1 holds. For any failure probability $0 < \delta < 1$, the error from equation 3.18 is bounded by

$$\mathbb{P}\Big[|\mu(\boldsymbol{x}) - f(\boldsymbol{x})| \leq B\sigma(\boldsymbol{x}) + \eta(\boldsymbol{x}),\, \forall \boldsymbol{x} \in \mathcal{X}\Big] \geq 1 - \delta, \tag{3.19}$$

with

$$\eta(\boldsymbol{x}) = R\|(K + \lambda I)^{-1}\boldsymbol{k}(\boldsymbol{x})\| \times \sqrt{N + 2\sqrt{N}\sqrt{\log\left[\frac{1}{\delta}\right]} + 2\log\left[\frac{1}{\delta}\right]}. \tag{3.20}$$

The Gram matrix containing the kernel evaluations of the training data X is denoted by K. The vector $k(\boldsymbol{x})$ consists of the kernel evaluations of the observed input $\boldsymbol{x}$ among X. Note that the authors assume $\mathbb{R}$-valued independent R-subgaussian random variables for the noise in observations whereas $\lambda \geq 0$ corresponds to the noise assumptions made in the GP.

Furthermore, specified error bound construction exists even in the case of model misspecification, i.e. when the hyperparameters of the kernel are not set exactly as required to match the ground truth. Those findings show that the difference in error bounds behaves moderately in the presence of increasing misspecification [8]. However, this is not of concern for this work.

In summary, the presented error bound construction is less computationally expensive and results in tighter bounds as other existing frequentist methods like maximizing the information gain [7], [9]. The applicability for learning-based control tasks in the presence of safety constraints is thus increased.

3.2.2 Bayesian Uncertainty Bounds

Another approach of constructing reliable uncertainty bounds rises from the assumption of the true function f to be a sample from a GP. Moreover, Lipschitz continuity within GP-based Bayesian Optimization is assumed [10]. This strategy is fundamentally motivated by the necessity of reliable and explicitly quantifiable model errors in safety-critical settings. Contrarily to deterministic models, the probabilistic nature of GP Regression problems is exploited as they inherently exhibit probabilistic measures of uncertainty.

Existing probabilistic uniform error bounds often rely on strongly restrictive assumptions such as the challenging computation of complex constants like the information gain [7] making them unsuitable for safety-critical applications in control. Furthermore, other types of uncertainty scaling require the unknown function to be a member of the RKHS associated by the kernel in the applied GP, as seen in section 3.2.1. Reliable methods to estimate RKHS bounds of an unknown function do not exist and are thus impractical. Therefore, probabilistic error bounds that are independent on RKHS norm bounds are now presented which require a set of assumptions.

Assumption 2 *The true function $f(\boldsymbol{x}) : \mathcal{X} \mapsto \mathbb{R}$ is a sample from a zero mean GP with kernel $k(\boldsymbol{x}, \boldsymbol{x'})$.*

It is important to note that assumption 2 is essential to distinguish this approach from the frequentist one discussed in the previous section. Sample functions from a GP do not exhibit finite RKHS norms as demonstrated in section 2, prohibiting the use of the afore mentioned RKHS norm B. Another required assumption is the Lipschitz continuity of the kernel.

Assumption 3 *The applied kernel k is at least four times differentiable and Lipschitz continuous with Lipschitz constant L_k on the compact vector space $(\mathcal{X}, ||\cdot||_p)$, with $p \in \mathbb{N}$.*

Most commonly used kernel like the SE kernel or the Matérn kernel exhibit Lipschitz continuity [1], hence assumption 3 imposes weak restrictions. The Lipschitz constant of a kernel can thereby be computed by

$$L_k := \max_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{X}} \left\| \begin{bmatrix} \frac{\partial k(\boldsymbol{x},\boldsymbol{x}')}{\partial x_1} & \dots & \frac{\partial k(\boldsymbol{x},\boldsymbol{x}')}{\partial x_d} \end{bmatrix}^T \right\|. \tag{3.21}$$

Nonetheless, assumption 3 is powerful as it can be proven that Lipschitz continuity of kernels results in Lipschitz continuity of the posterior distributions and samples of a GP with such kernels [41]. Thus, a modulus of continuity ω_μ for the posterior mean can be defined as

$$|\mu(\boldsymbol{x}) - \mu(\boldsymbol{y})| \leq \omega_{\boldsymbol{\mu}}(||\boldsymbol{x} - \boldsymbol{y}||_p), \tag{3.22}$$

and equivalently for the posterior standard deviation σ

$$|\sigma(\boldsymbol{x}) - \sigma(\boldsymbol{y})| \leq \omega_{\boldsymbol{\sigma}}(||\boldsymbol{x} - \boldsymbol{y}||_p). \tag{3.23}$$

For computing and bounding such moduli of continuity, the input space $\mathcal{X}$ is discretized into a finite set $\mathcal{I}$. This is done by constructing a set of equivalence classes where each member $[x]$ of $\mathcal{X}$ can be defined by

$$[x] := \{x' \in \mathcal{X} \mid \|x - x'\|_p \leq \tau\}. \tag{3.24}$$

Due to the compactness of $\mathcal{X}$ and the non-negativity of τ, which can be thought of as a grid constant, the cardinality of the constructed set $\mathcal{I}$ is finite, $|\mathcal{I}| < \infty$. Applying this discretization thus allows to compute the moduli of continuity depending on τ. Moreover, both can be bounded by leveraging the containment of the posterior mean and standard deviation in the kernel associated RKHS. The resulting upper bounds are

$$\begin{aligned} \omega_\mu(\tau) &\leq \sqrt{2\tau L_k}||\boldsymbol{\mu}||_{\mathcal{H}}, \\ \omega_\sigma(\tau) &\leq \sqrt{2\tau L_k}. \end{aligned} \tag{3.25}$$

Note that $||\boldsymbol{\mu}||_{\mathcal{H}}$ denotes the RKHS norm of the posterior mean which can be computed by

$$||\boldsymbol{\mu}||_{\mathcal{H}} = \sqrt{\boldsymbol{y}^T(K + \sigma_\epsilon I)^{-1}K(K + \sigma_\epsilon I)^{-1}\boldsymbol{y}}. \tag{3.26}$$

resulting from the computation of the posterior mean evaluated along the training data X as

$$\mu(X) = K(K + \sigma_\epsilon^2 I)\boldsymbol{y}, \tag{3.27}$$

where K denotes the Gram matrix containing the kernel evaluations of the training data. Finally, for a failure probability of $\delta \in (0, 1)$ the model error can be bounded [11].

$$\mathbb{P}\Big[|f(\boldsymbol{x}) - \mu_N(\boldsymbol{x})| \leq \sqrt{\beta(\tau)}\sigma_N(\boldsymbol{x}) + \gamma(\tau),\ \forall \boldsymbol{x} \in \mathcal{X}\Big] \geq 1 - \delta \tag{3.28}$$

The scaling factor of the standard deviation β is hereby solely dependent on the discretization applied on the compact input space and is defined as

$$\beta = 2 \log\left(\frac{|\mathcal{I}|}{\delta}\right). \tag{3.29}$$

For instance, a one-dimensional, real-valued input space $[0, 10]$ with a grid constant of $\tau = 0.01$, requires $|\mathcal{I}|$ intervals of radius τ to cover the whole input space for which $|\mathcal{I}|$ can be computed as

$$|\mathcal{I}| = \frac{10 - 0}{2\tau} = 500. \tag{3.30}$$

Moreover, the additive term γ is defined by

$$\gamma(\tau) = L_f \tau + \omega_\mu(\tau) + \sqrt{\beta}(\tau)\omega_\sigma. \tag{3.31}$$

Recall that Lipschitz continuity of samples is provided by assumption 3. Consequently, the Lipschitz constant of the true function L_f must be determined. For many control applications, prior domain knowledge allows a formulation of L_f for example based on simplified first or second order physical models. However, methods to estimate L_f without domain knowledge exist but are excluded in this work [42].

Note that the influence of the term $\gamma(\tau)$ can be designed to be negligible compared to $\sqrt{\beta(\tau)}\sigma(\boldsymbol{x})$ by choosing τ arbitrarily small as $\gamma(\tau)$ strongly depends on τ and $\beta(\tau)$ exhibits logarithmic growth depending on τ. Moreover, even conservative approximations for ω_μ, L_f and ω_{σ_N} have a weak influence on the error bounds from equation 3.28 as $\beta(\tau)$ hardly grows logarithmically with decreasing τ [10].

The presented approach - which is hereafter referred to as the *Bayesian* view - of computing uniform error bounds surpasses existing frequentist strategies that are based on information gains as presented earlier, especially in terms of computational cost but also w.r.t. conservatism. The exploitation of Lipschitz continuity of the kernel and posterior distributions has shown to result in tighter, yet reliable error bounds [10].

However, it is not apparent nor provided by the literature whether bounds resulting from the bayesian view surpass the performance of error bounds constructed according to [8], which are hereafter referred to as the *Frequentist* view.

As Lipschitz constants in general are considered a conservative approach, one can infer that error bounds resulting from the assumption of the bayesian view, incorporated in safe BO algorithms perform weaker than error bounds from the frequentist view. To provide experimental validation for assessing which type of error bounds yields in more efficient optimization algorithms and to what extent, the error bounds from the frequentist and bayesian view will be practically implemented and compared w.r.t. convergence and safety constraint satisfaction of the associated BO algorithms.

For better comparability of both views, table 1 summarizes the assumptions in place and the error bounds whose performance will be compared.

	Assumptions	Error Bounds
Frequentist	$f \in \text{RKHS}$ $\|f\|_{\mathcal{H}} \leq B$ (Assumption 1)	$B\sigma(\boldsymbol{x}) + \eta(\boldsymbol{x})$ (Eq. 3.19, 3.20)
Bayesian	$f(\cdot) \sim \text{GP}(0, k(\cdot, \cdot))$ f, k are Lip. continuous with L_f, L_k (Assumptions 2, 3)	$\sqrt{\beta(\tau)}\sigma(\boldsymbol{x}) + \gamma(\tau)$ (Eq. 3.28, 3.29, 3.31)

Table 1: Overview of Frequentist and Bayesian Assumptions and Error Bounds

The choice of algorithms, kernels and other relevant parameters is elaborately discussed in the following section.

4 Experimental Setup

The previous sections demonstrated approaches to scale uncertainty in a safety-critical setting. As frequently mentioned in literature, uncertainty scaling based on maximizing the information gain performs poorly due to large computational effort and strong conservatism. Thus, this work aims at comparing the remaining strategies resulting from the bayesian and improved frequentist view, summarized in table 1. In the former, the true function is assumed to be a sample of a GP exhibiting Lipschitz continuity, whereas the true function in the latter is assumed to be a member of the RKHS associated by the kernel choice. To quantify the associated conservatism, an experimental setup is presented which is desired to produce valid data used for the comparison of efficiency. Moreover, core characteristics of the presented algorithms will be examined.

For comparing both settings, they are each implemented in a modified `SafeOpt` and `SafeUCB` algorithm as introduced in section 3.1.2 and 3.1.1, respectively. Further experimental settings such as the choice of the kernel or true function are now demonstrated.

4.1 Practical Implementation of Applied Algorithms

First, the practical implementation of `SafeOpt` and `SafeUCB` will be demonstrated. They were demonstrated in their initial proposition [3] in sections 3.1.2 and 3.1.1. However, they will be modified within the frame of the applied experimental setup to simplify and thus accelerate computation. For the remainder of this work, one-dimensional inputs $x \in \mathbb{R}$ are considered.

4.1.1 `SafeOpt`

Recalling the presentation of the `SafeOpt` algorithm in section 3.1.2, the applied algorithm complies with the construction of first safe sets, then maximizer and expander sets. While section 3.1.2 demonstrated that the subsequent query point x_n is chosen from the union set of maximizers and expander by finding the largest uncertainty among all inputs, the applied SafeOpt algorithm here follows another idea. The union set of maximizers and expanders can become quite large, especially for multidimensional settings. Thus, computing the uncertainty for all members of the union set is linked to large computational effort [38]. One approach to overcome that issue was to merge derivative-free optimization methods with safe learning by posing additional optimization problems that allow the application of particle swarm optimization (PSO) [43]. However, achieving efficient performance with PSO methods requires auxiliary adjustment which relies on heuristics.

To avoid that issue, another strategy is applied here. Instead of sampling from the union set or applying PSO or other direct-search methods, separate optimization problems for the maximizer and expander classification are formulated. Each optimization problem finds the input with highest uncertainty from the set of maximizers and separately from the set of expanders [44]. Those separate optimization problems follow the same idea of

maximizers and expanders as introduced in section 3.1.2, but are formulated otherwise. Expressing the maximizer classification as an optimization problem yields in

$$\begin{aligned} \max_{x \in \mathcal{X}} \quad & w_n(x) && \text{(4.1a)} \\ \text{subject to} \quad & l_n(x) \geq h, && \text{(4.1b)} \\ & u_n(x) \geq l^*. && \text{(4.1c)} \end{aligned}$$

Note that l^* stands for the largest lower bound as

$$l^* = \max_x l_n(x). \tag{4.2}$$

Note that only inputs can solve this optimization problem which are also a member of the maximizer set as defined in equation 3.9. The inputs must be safe, imposed by the first constraint and their upper bound must be larger than the largest lower bound, imposed by the second constraint. This formulations thus avoids the expensive construction of the maximizer set. Similarly, the optimization problem for the expander problem is given by

$$\begin{aligned} \max_{x, x' \in \mathcal{X}} \quad & w_n(x) && \text{(4.3a)} \\ \text{subject to} \quad & l_n(x) \geq h, && \text{(4.3b)} \\ & l_n(x') < h, && \text{(4.3c)} \\ & l_{n(x, u_n(x))}(x') \geq h. && \text{(4.3d)} \end{aligned}$$

This optimization problem works with two variables in contrast to the maximizer optimization problem where solely one safe x is considered. The expander problem requires one safe x according to equation 4.3b and one unsafe input x'. Recall that $l_{n(x,u_n(x))}(x')$ represents the lower bound of the previously unsafe input x' after adding the upper bound $u_n(x)$ of the safe input as an observation to the GP model.

Practically, those optimization problems are handled by the `gen_candidates_scipy` method from the `botorch` library which generally requires satisfaction of constraints c_i to be expressed as $c_i(x) > 0$. Therefore, the constraints are implemented as

Maximizer constraints:
$l_n(x) - h \geq 0,$
$u_n(x) - l^* \geq 0,$

Expander constraints:
$l_n(x) - h \geq 0, l_n(x') - h < 0,$
$l_{n(x,u_n(x))}(x') - h \geq 0.$

Moreover, a solver for those optimization problems must be chosen that is capable of handling the non-linear inequality constraints in place. Thus, the gradient-based *Sequential Least Square Programming* (SLSQP) method is chosen as it is generally able to solve constrained nonlinear optimization problems. It requires convergence criteria which are

here appointed by a maximum of $M_{SLSQP} = 100$ iterations and a tolerance of $f_{tol} = 0.01$. Hence, the solver stops either if 100 iterations have been performed or when the current solution x_m for each optimization problem from eq. 4.1, 4.3 varies by less than f_{tol} from the previous one x_{m-1}.

Another property of optimization problems with nonlinear inequality constraints is the required availability of initial conditions that fulfill the constraints. Consequently, a method to determine such constraint fulfilling initial conditions $x_{ic,exp}, x_{ic,max}$ must be provided. A manual choice of initial conditions during execution would compromise the comparability of the frequentist and bayesian setting. As it is expected that both setting produce different uncertainty bounds, the sets from which one could choose initial conditions vary. To guard the comparability of efficiency, a randomized choice is applied. Maximizer and expander sets are computed in each iteration of the algorithm from which each, one input is randomly chosen and given to the optimizer as the starting point. By that, satisfaction of initial conditions $x_{ic,exp}, x_{ic,max}$ is assured and due to the randomized nature of choice, the influence of manually interfering can be counteracted by evaluating multiple algorithm executions and drawing a mean of the results. Recall that $x_{ic,exp}$ must contain one safe input x and one unsafe input x' for which equation 3.10 must hold.

Ultimately, this optimization-based formulation of `SafeOpt` yields a solution x^*_{exp} from the expander problem and x^*_{max} from the maximizer problem. The corresponding uncertainties w_n are compared and the subsequent query point x_n is the one with the higher uncertainty, hence

$$x_n = \underset{x^*_{exp}, x^*_{max}}{\arg\max} \{w_n(x^*_{exp}), w_n(x^*_{max})\}. \tag{4.4}$$

Finally, a stopping criterion for the `SafeOpt` algorithm must be set. Within the frame of this work, the goal is to find the global maximum of the unknown function f. Thus, a convergence criterion is needed which classifies the observation $y(x_n)$ of the input x_n determined by the algorithm as optimal. Therefore, the regret R is introduced by $R(n) = y_n - f_{max}$ where y_n is the observation after iteration n and f_{max} denotes the true global maximum. Now, one might assess some tolerance similar to what has been done for the SLSQP solver such that optimality is achieved when $R(n) < h_{opt}$. However as the observations are subject to noise, it is likely to not precisely match the true global optimum f_{max}. Hence, the optimality threshold h_{opt} must be carefully chosen which is here done by

$$h_{opt} = 3 \cdot \sigma_\epsilon. \tag{4.5}$$

Recalling that the noise is parameterized by its variance σ^2_ϵ, a confidence region around $R(n) = 0$ is formulated that considers the regret as a random variable with zero mean and the same variance as the present noise. A scaled standard deviation σ_ϵ hence results in a confidence region around zero containing the true regret at a probability of

$$P\Big[-3\sigma_\epsilon \leq R(n) \leq +3\sigma_\epsilon\Big] \approx 0.9973. \tag{4.6}$$

Thus, convergence is reached when

$$R(n) < h_{opt}. \tag{4.7}$$

A summary of the applied `SafeOpt` algorithm is depicted in Algorithm 1.

Algorithm 1 Applied `SafeOpt` Algorithm

Input: Initial safe set $\mathcal{S}_0 = \{x_0, x_1, ..., x_k\} \subset \mathcal{X}$
Output: Optimized candidate x_*
Set $n \leftarrow 1$, compute observations $\boldsymbol{y} = \{y(x_i)\}_{i=1,...,n}$
repeat
  Using $\mathcal{S}_n$ and y_n, compute GP with lower and upper bounds $l_n(x)$, $u_n(x)$
  Construct $\mathcal{M}_n$ from eq. 3.9, set $x_{ic,max} \leftarrow$random sample of $\mathcal{M}_n$
  Construct $\mathcal{G}_n$ from eq. 3.10, set $x_{ic,exp} \leftarrow$random sample of $\mathcal{G}_n$
  $l^* \leftarrow$Solve eq. 4.2
  $x^*_{max} \leftarrow$Solve eq. 4.1
  $x^*_{exp} \leftarrow$Solve eq. 4.3
  Compute $w_n(x^*_{exp})$, $w_n(x^*_{max})$
  Set $x_n \leftarrow \arg\max_{x^*_{exp}, x^*_{max}} \{w_n(x^*_{exp}), w_n(x^*_{max})\}$
  Compute $y_n = f(x_n) + \epsilon$
  Set $y_n \leftarrow \boldsymbol{y} \cup \{y_n\}$
  Set $\mathcal{S}_n \leftarrow \mathcal{S}_{n-1} \cup \{x_n\}$
  Set $n \leftarrow$n+1
until $R(n) < h_{opt}$

4.1.2 `SafeUCB`

The `Safe-UCB` algorithm follows section 3.1.1 while applying the same modifications as given for `SafeOpt`. Thus, the subsequent safe input x_n is determined by the highest upper confidence bound as stated in equation 3.6. That is now expressed in terms of a separate optimization problem to avoid the computation of upper confidence bounds $u(x)$ among the entire safe set. The optimization problem in place is hence

$$\max_{x \in \mathcal{X}} \quad u_n(x) \tag{4.8a}$$

$$\text{subject to} \quad l_n(x) \geq h. \tag{4.8b}$$

Equivalently to the explanation given for the applied `SafeOpt` algorithm, the nonlinear inequality constraints need to be formulated in a way that satisfaction of the constraint is achieved by an output larger than zero.

$$l_n(x) - h \geq 0. \tag{4.9}$$

To solve this problem the SLSQP method is applied, requiring once again an initial condition x_{ic} satisfying the non-linear inequality constraint. Here, the choice of a constraint fulfilling initial condition is less complex as for `SafeOpt` as solely safeness is required. Consequently, the initial condition is randomly chosen from the safe set $\mathcal{S}_n$. Moreover, the same convergence criteria from equation 4.7 holds. An algorithmic overview is presented in algorithm 2.

Algorithm 2 Applied `Safe-UCB` Algorithm

Input: Initial safe set $\mathcal{S}_0 = \{x_0, x_1, ..., x_k\} \subset \mathcal{X}$, safety threshold h,
Output: Optimized candidate x_*

Set $n \leftarrow 1$, compute observations $\boldsymbol{y}_n = \{y(x_i)\}_{i=1,...,n}$
repeat
    Using $\mathcal{S}_n$ and y_n, compute GP with lower and upper bounds $l_n(x)$, $u_n(x)$
    x_{ic} ←random sample from $\mathcal{S}_n$
    x_n ←Solve eq. 4.8
    Compute $y_n = f(x_n) + \epsilon$
    Set y_n ←$\boldsymbol{y} \cup \{y_n\}$
    Set $\mathcal{S}_n$ ←$\mathcal{S}_{n-1} \cup \{x_n\}$
    Set n ←n+1
until $R(n) < h_{opt}$

4.2 Construction of a Reference True Function and Gaussian Process

This work considers safety-critical BO of a fixed, continuous, real and one-dimensional function $f : \mathcal{X} \mapsto \mathbb{R}$ over the input domain $\mathcal{X} \in \mathbb{R}$. The goal is to find the global maximum while minimizing the amount of iterations. To reliably compare the efficiency of error bounds resulting from the frequentist and bayesian view with different safe BO algorithms, the same ground truth function f shall be applied.

While the frequentist view assumes f to be a member of the RKHS of the associated GP kernel, the Bayesian setting assumes Lipschitz continuity of f while being a sample of a GP. Therefore, a ground truth function is chosen that is a member of an associated RKHS, bounded by some RKHS norm B and additionally Lipschitz continuous. Moreover, the applied kernel must be Lipschitz continuous. Note that not all functions from an RKHS are necessarily Lipschitz continuous. Therefore, it is required to choose a true function f that in summary exhibits the following properties based on the kernel k used in the GP:

1. $f \in$ RKHS with $||f||_{\mathcal{H}} \leq B$
2. f is Lipschitz continuous with Lipschitz constant L_f
3. k is Lipschitz continuous with Lipschitz constant L_k

Now, to generate a one-dimensional function f that satisfies the listed requirements, recall that the function

$$x \mapsto \sum_{n=1}^{N} \alpha_n k(x_n, x) \tag{4.10}$$

is contained in the RKHS associated by kernel k [45]. The parameters $\alpha_1, ..., \alpha_N \in \mathbb{R}$ and $x_1, ..., x_N \in \mathcal{X}$, for $N \in \mathbb{N}$ are hereafter referred to as weights and centers, respectively. This allows to sample a function f from the RKHS by randomly sampling centers and weights that is defined by

$$f(x) = \sum_{n=1}^{N} \alpha_n k(x_n, x). \tag{4.11}$$

The prescribed RKHS norm of such a function f is computed by

$$B = \sqrt{\boldsymbol{\alpha}^T K \boldsymbol{\alpha}}, \quad \boldsymbol{\alpha} \in \mathbb{R}^N, \tag{4.12}$$

with Gram matrix K contains the kernel evaluations of all center values $x_1, ..., x_N$. Thereafter, one can scale the weights $\boldsymbol{\alpha}$ in order to impose a specific RKHS norm B on a generated function f. To do so, $\tilde{\boldsymbol{\alpha}} \in \mathbb{R}^N$ is appointed with prescribed values and applying then

$$\boldsymbol{\alpha} = \frac{B}{\sqrt{\tilde{\boldsymbol{\alpha}} K \tilde{\boldsymbol{\alpha}}}} \tilde{\boldsymbol{\alpha}} \tag{4.13}$$

in equation 4.11 results in a function f that exhibits RKHS norm $||f||_{\mathcal{H}} = B$.

Consequently, the construction of such desired true functions depends on the kernel choice which also determines the GP prior. For increased numerical validation of the results, two different kernels are applied. Namely, the SE kernel and the RQ kernel. Similar to the presentation of SE and RQ kernels in chapter 2, the implementations from `gpytroch` result in

$$\begin{aligned} k_{SE}(x, x') &= \sigma_f \exp\left(-\frac{(x - x')^2}{2\ell^2}\right), \\ k_{RQ}(x, x') &= \sigma_f \left(1 + \frac{(x - x')^2}{2\alpha\ell^2}\right)^{-\alpha}. \end{aligned} \tag{4.14}$$

Thus, two true functions depending on the kernel will be constructed. One for the SE kernel resulting in f_{SE} and one for the RQ kernel, f_{RQ}. The mean function of the corresponding GPs is without loss of generality set to zero, resulting in

$$\begin{aligned} f_{SE}(x) &\sim \mathcal{GP}\left(0, k_{SE}(x, x')\right), \\ f_{RQ}(x) &\sim \mathcal{GP}\left(0, k_{RQ}(x, x')\right). \end{aligned} \tag{4.15}$$

In order to numerically generate functions f_{SE} and f_{RQ} that are members of the RKHS associated k_{SE} and k_{RQ} respectively, the kernel hyperparameters must be set for both. Note that hyperparameter tuning is not considered, thus the hyperparameters are initially set and kept constant. The default values from the `gpytorch` implementation of the SE and RQ kernel are applied.

To finalize the setting of the GP prior, noise assumption must be incorporated. It is assumed that no precise function values are available, rather noise-perturbed observations. Hereby, i.i.d. gaussian noise is implemented with a noise variance $\sigma_\epsilon^2 = 0.001$ such that observations are computed by $y(x) = f(x) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$. In real world settings, the present noise when generating inputs is usually not known thus the GP assumes a noise variance that is likely larger than the true noise variance in place. However, setting the assumed noise variance in the GP equal to the actual noise variance does not degenerate the comparability of efficiency of the different error bound constructions. Thus, the GP is equipped with a noise assumption that is the afore mentioned σ_ϵ.

Finally, the resulting true function from applying the SE kernel is portrayed in figure 4.1 and for the RQ kernel in figure 4.2. Both base on the numerical values summarized in table 2. Their choice is explained by requiring a ground truth that reveals typical problems in safety-critical BO.

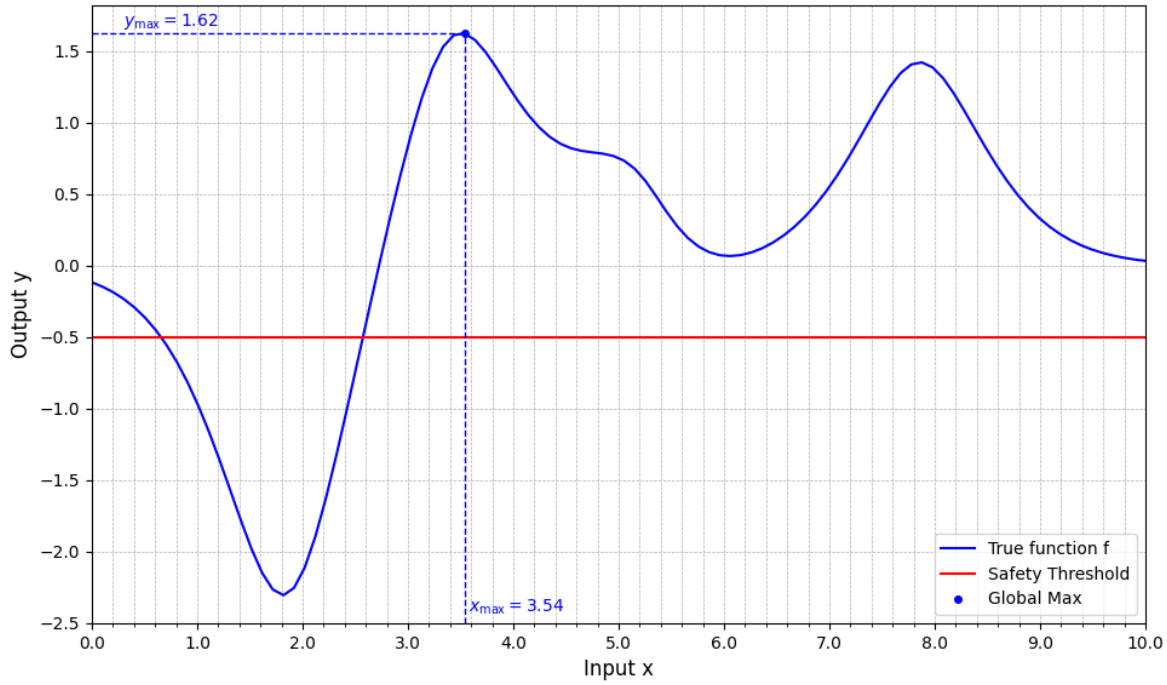


Figure 4.1: True function f_{SE} generated from SE kernel

On the one hand, various local maxima should exist such that the performance of applied algorithms can be evaluated along its capability of finding global maxima in the presence of various local ones. On the other hand, the function should exhibit, together with the chosen safety constraint, a reasonable ratio between safe and unsafe input. Therefore, the

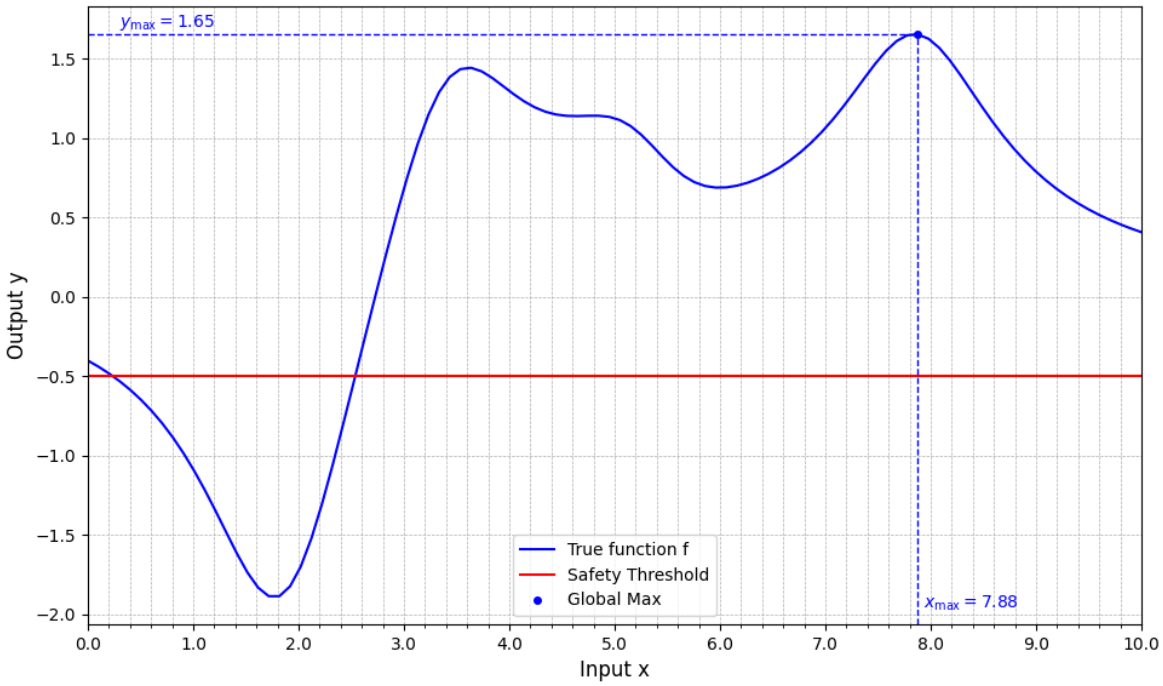


Figure 4.2: True function f_{RQ} generated from RQ kernel

Centers	Weights	RKHS Norm Bound	Kernel Hyperparameters		Safety Threshold
			RQ	SE	
$x_1 = 1.8557$	$\alpha_1 = -1.9098$				
$x_2 = 3.4076$	$\alpha_2 = 1.3832$				
$x_3 = 5.2115$	$\alpha_3 = 1.0477$	$B = 4$	$\ell_{RQ} = 0.6931$	$\ell_{SE} = 0.6931$	$h = -0.5$
$x_4 = 5.4738$	$\alpha_4 = -0.7144$		$\alpha_{RQ} = 0.6931$	$\sigma_{f,SE} = 0.6931$	
$x_5 = 7.8554$	$\alpha_5 = 1.1028$		$\sigma_{f,RQ} = 0.6931$		

Table 2: Parameter settings of Kernels and True Function

one-dimensional safety constraint was set to $h = -0.5$ with an input space $\mathcal{X} = [0, 10]$. The kernel hyperparameters are initially set and not modified by training of the model throughout execution.

In the next step, the choice of the initial safe set $\mathcal{S}_0$ that is given to the GP is discussed. Looking at the true function resulting from the SE kernel of figure 4.1, one can see that the safe set is disconnected. One part of the overall safe set can be roughly defined by $\mathcal{S}_a = [0, 0.6]$ and the other by $\mathcal{S}_b = [2.6, 10]$. This is relevant for the choice of the initial safe set as safety-critical BO algorithms are not capable of finding the global maximum with a finite amount of iterations if the initial safe set does not contain the input associated

with the global maximum. For the algorithm to move between disconnected safe set, evaluations of unsafe inputs would be required which is here prevented. Thus, the initial safe set $\mathcal{S}_0$ must be chosen from $\mathcal{S}_b$. As a starting point, the input $x_0 = 6.5$ is chosen with its noisy observation $y_0(x_0) = f(x_0) + \epsilon$. This training data point $\{x_0, y_0\}$ is given to the GP from equation 4.15 from which the first predictive distribution can be drawn. Recall that i.i.d. gaussian noise is considered, sampled from a normal distribution parameterized by $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ with $\sigma_\epsilon^2 = 0.001$.

For the true function from the RQ kernel, the same requirements hold including noise properties. Hence, an initial safe input is chosen from the safe set that contains the input corresponding to the global maximum which is hereby chosen to be $x_0 = 5.6$. The resulting observation is added to the GP model from which the first predictive distribution can be computed.

4.3 Parameterization of the Bayesian and Frequentist Setting

To finally compare the efficiency of both algorithms, relevant parameters introduced in the presentation of frequentist and bayesian error bounds need to be determined.

Recalling the frequentist error bound from equation 3.19 with equation 3.20, the RKHS norm bound B dominantly describes the uncertainty scaling. At the same, this is the most challenging parameter to be set as translating domain knowledge about some process f to a reliable and applicable RKHS norm such that $||f||_{\mathcal{H}} \leq B$ is particularly complicated. For this work however, ideal assumptions will be made to emphasize the goal of comparing the efficiency of both algorithms under the frequentist and bayesian setting. Therefore, the practical implementation of the frequentist settings works with an RKHS bound assumption equivalent to the real RKHS bound $B = 4$ that was used to generate the ground truth. Secondly, the parameters λ and R need to be addressed as the remaining parts of equations solely depend on training data and kernel choice. The former represents the noise assumptions made in the GP model that was earlier introduced by the GP's `likelihood.noise` variable. The parameter R characterizes the observation noise. For the presented frequentist error bounds, the noise is not restricted to be i.i.d. gaussian noise but is assumed to be $\mathbb{R}$-valued independent R-subgaussian noise. However, this work assumes known i.i.d. gaussian noise which allows to set R in accordance to the ideal noise assumption of the GP model to $R = \lambda$.

Next, the more elaborate parameter allocation in the bayesian is discussed. Recalling from section 3.2.2, the Lipschitz constant of the kernel L_k must be initially computed according to

$$L_k := \max_{x,x' \in \mathcal{X}} \left\| \begin{bmatrix} \frac{\partial k(x,x')}{\partial x_1} & \dots & \frac{\partial k(x,x')}{\partial x_d} \end{bmatrix}^T \right\|. \tag{4.16}$$

As this expression does not depend on training data or other process variables, the Lipschitz constant of the SE and RQ kernels is initially computed and kept constant yielding in $L_{k,SE} = 0.6065$ and $L_{k,RQ} = 0.4214$.

Afterwards, the Lipschitz constant of the posterior mean function L_{μ_N} is computed which depends on L_k, kernel evaluations along the training set and the observations y_n. Thus, this parameter must be reevaluated after each enlargement of the training set. The modulus of continuity ω_{σ_N} depends on the grid constant τ. As demonstrated in section 3.2.2, that constant can be chosen arbitrarily which is herein set to $\tau = 0.0001$. Since the scaling factor β is only dependent on τ, the input space $\mathcal{X}$ and the failure probability δ, one can initially set β as well. For a failure probability of $\delta = 0.05$ that holds for both settings, the scaling factor results in $\beta(\tau) = 29.0173$.

The choice of the Lipschitz constant of the true function L_f is in practice the most delicate as it requires advanced domain knowledge or reliable first or second order physical models. This work aims dominantly at the comparison between the different error bound settings and henceforth applies ideal parameter settings. Thus, the actual Lipschitz constants from the generated true functions f_{SE} and f_{RQ} are used resulting in $L_{f,SE} = 3.6821$ and $L_{f,RQ} = 2.6591$.

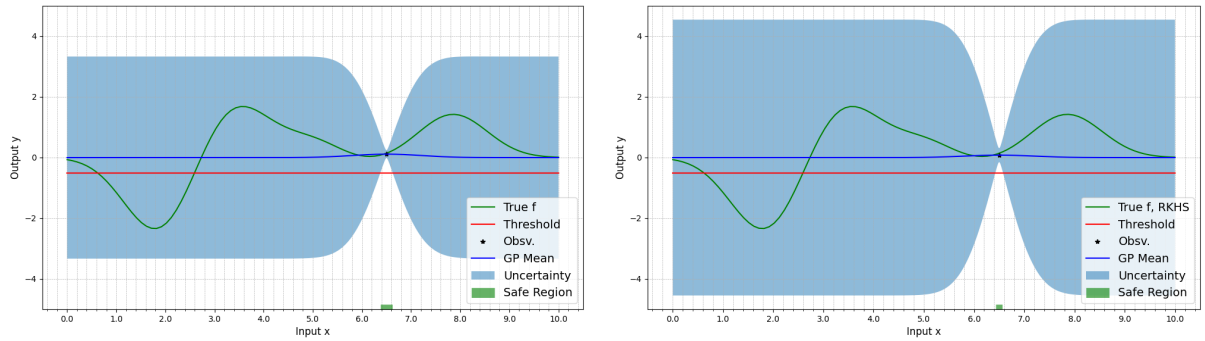
In conclusion, both algorithms are executed with a failure probability of $\delta = 0.05$, over the input space $\mathcal{X} = [0, 10]$ given the remaining parameters listed in tabled 3.

<table>
<tr><th colspan="2">Frequentist</th><th colspan="2">Bayesian</th></tr>
<tr><td colspan="2">$\lambda = 0.001$
$R = \lambda$
$B = 4$</td><td colspan="2">$\tau = 1 \times 10^{-4}$
$\beta = 29.0173$</td></tr>
<tr><th>SE</th><th>RQ</th><th>SE</th><th>RQ</th></tr>
<tr><td>$S_0 = [6.5]$</td><td>$S_0 = [5.6]$</td><td>$S_0 = [6.5]$
$L_{k,SE} = 0.6065$
$L_{f,SE} = 3.6821$</td><td>$S_0 = [5.6]$
$L_{k,RQ} = 0.4214$
$L_{f,RQ} = 2.6591$</td></tr>
<tr><td colspan="4">$\mathcal{X} = [0, 10]$, $\delta = 0.05$, $\sigma_\epsilon^2 = 0.001$</td></tr>
</table>

Table 3: Applied Kernel Dependent Numerical Values for Frequentist and Bayesian View

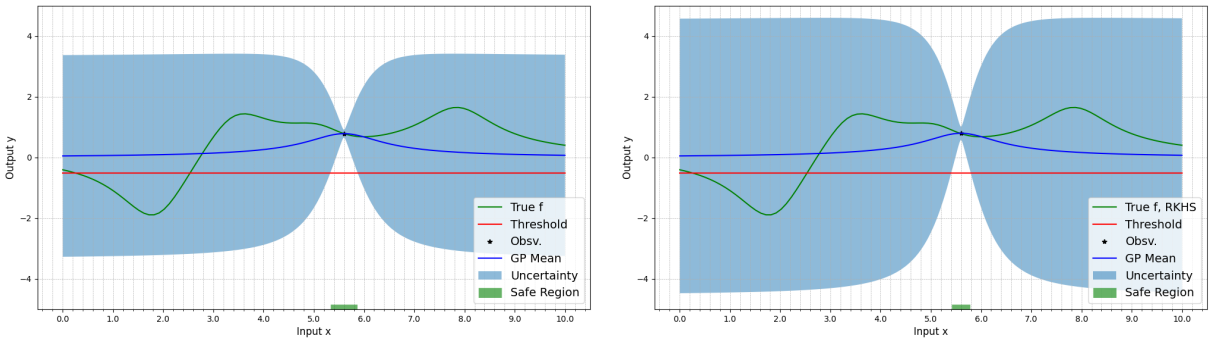
Figure 4.3 shows the predictive distributions after one observation at the initial safe input from f_{SE}. Note the difference of uncertainty size in both settings. The resulting safe set from the frequentist setting consist of $\mathcal{S}_{SE,f,1} = [6.3664, 6.6266]$ whereas the safe set from the bayesian setting is slightly smaller, given by $\mathcal{S}_{SE,b,1} = [6.4064, 6.5966]$.

Similarly, figure 4.4 depicts the predictive distributions after one observation at the initial safe input from f_{RQ}. Here, the safe set from the predictive distribution resulting from frequentist error bounds is $\mathcal{S}_{RQ,f,1} = [5.3353, 5.8659]$ and from the bayesian error bounds $\mathcal{S}_{RQ,b,1} = [5.4054, 5.7958]$. Thus, the prediction are again more conservative in the bayesian



(a) Frequentist View (b) Bayesian View

Figure 4.3: Predictive Distributions after one Observation from f_{SE}



(a) Frequentist View (b) Bayesian View

Figure 4.4: Predictive Distributions after one Observation from f_{RQ}

view as $|\mathcal{S}_{RQ,f,1}| > |\mathcal{S}_{RQ,b,1}|$ suggesting increased conservatism of the bayesian error bounds independent of kernel choice and the true function.

5 Results and Discussion

This chapter presents and discusses the results from the experimental setup. As explained in chapter 3, the focus lays on the comparison between the frequentist and bayesian setting of error bound construction. In addition to that, the general behavior of the `SafeOpt` and `SafeUCB` algorithms will be compared w.r.t. to gloabal convergence and safety constraint satisfaction. To increase statistical validity, both the SE and RQ kernel are implemented in each algorithm with the frequentist and bayesian error bounds. Consequently, eight algorithms are available for analysis.

5.1 Comparison of `SafeOpt` and `SafeUCB`

The previous chapters have elaborately discussed various techniques to implement safe BO algorithms. The `SafeOpt` and `SafeUCB` algorithms have been pointed out to likely achieve safety constraint satisfaction while efficiently optimizing the ground truth to reveal the global maximum. To demonstrate the algorithmic behavior, example executions of both will be illustrated. For that purpose, the frequentist error bounds are effective while the GP is equipped with the previously defined SE kernel. Both algorithms thus optimize the corresponding true function f_{SE} depicted in figure 4.1.

Both algorithms are provided with the same initial safe input $x_0 = 6.5$ together with its noisy observation $y(x_0)$ from which the first predictive distribution can be computed, see figure 5.1.

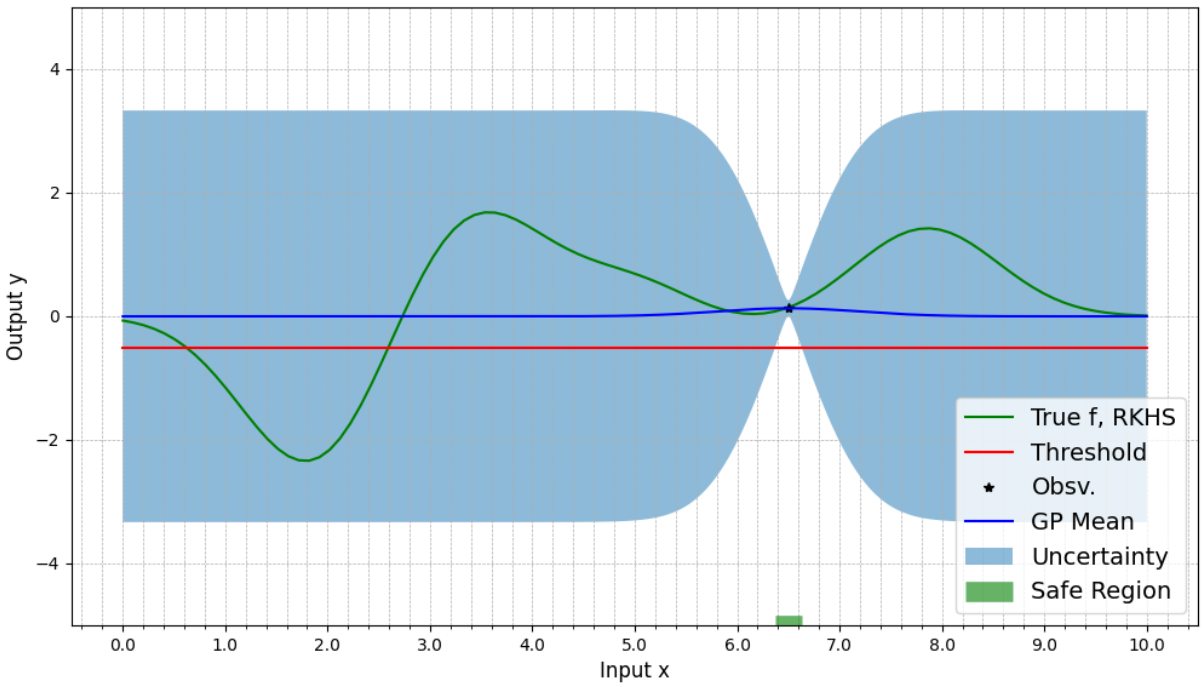
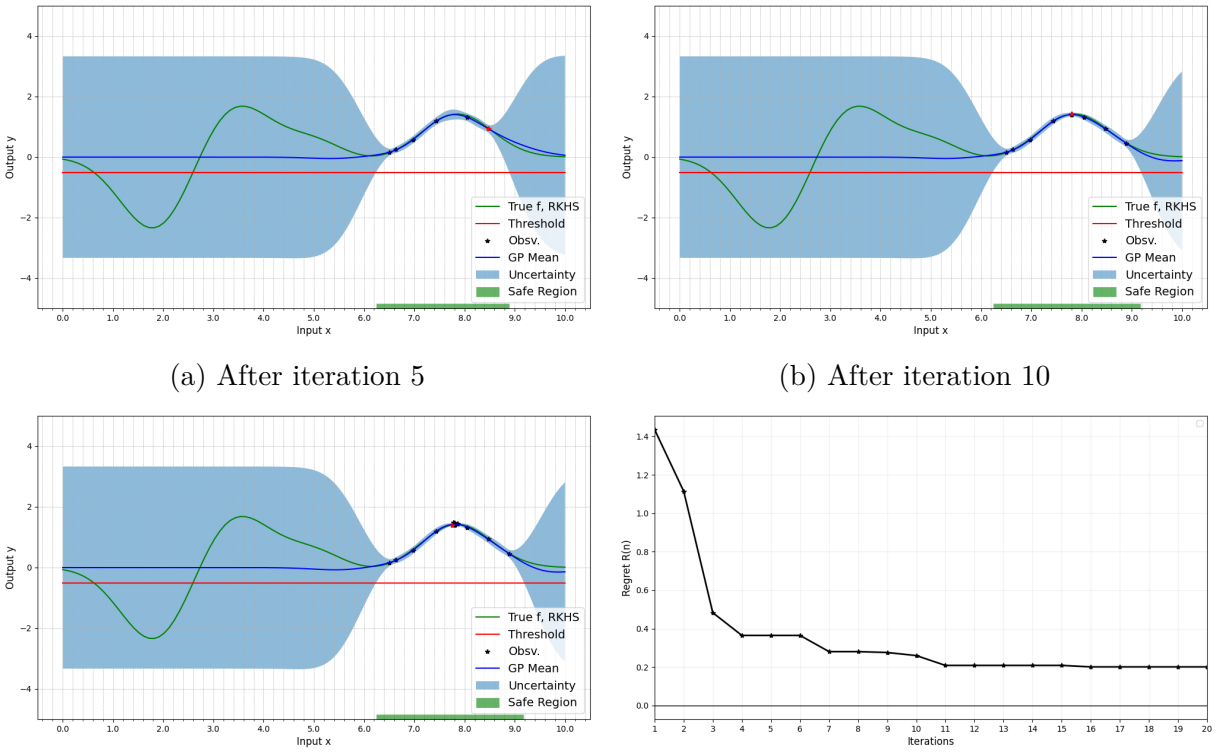


Figure 5.1: Initial Predictive Distribution, Frequentist Setting with SE Kernel

Now, the behavior of the `SafeUCB` algorithm is demonstrated by analyzing informative iterations depicted in figure 5.2.



(a) After iteration 5

(b) After iteration 10

(c) After iteration 20

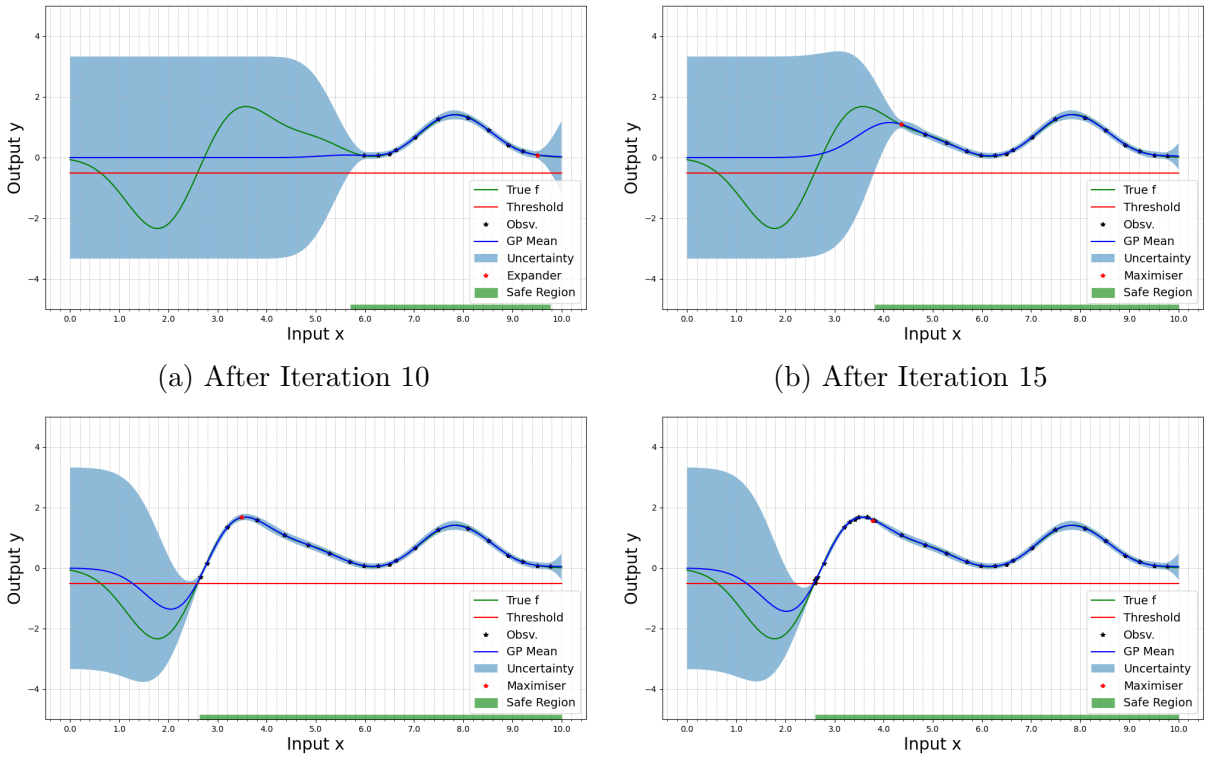
(d) Convergence Plot

Figure 5.2: Predictive Distributions from `SafeUCB` Algorithm after 5, 10, 20 Iterations and Convergence Plot

From figure 5.2a one can see how the algorithm rapidly explores the local maximum at $x = 7.8$, requiring solely five iterations. By that, the safe set has increased as indicated by the green bar above the input axis. Another five iterations later, figure 5.2b depicts how the safe set has further increased towards its right limit. At the same time, the algorithm attempts to further maximize the observations around the local maximum as indicated by the observation with the red star. Those generally indicate the observation that has been added lastly. Now, the nature of `SafeUCB` becomes apparent, as figure 5.2c reveals a high amount of observations located at the local maximum. Thereby, no further exploration of safe inputs has been conducted as the safe set has not been increased compared to figure 5.2b. One can hence conclude that the algorithm got stuck at the local maximum and is unable to further optimize the function leading eventually to finding the global maximum at $x = 3.54$. This can equivalently be demonstrated by the convergence plot in figure 5.2d where the regret is plotted against the iterations. Note how the regret remains approximately constant after iteration eleven at a regret of $R(n) = 0.201$. The strongest decay in regret took place until iteration four. The remaining change is rather due to the present noise than actual optimization. By accurately examining the local maximum, a high density of observations can be found from which some are higher than the actual

local maximum. Finally, the convergence towards the local maximum is explained by the fact that `SafeUCB` maximizes the upper confidence bound. If the safe set does not contain inputs whose corresponding upper bound is above the current best observation, the algorithm will always persist at this current best observation. Particularly in the presence of noise as even observed inputs keep exhibiting an upper confidence bound.

Next, the results from `SafeOpt` are presented in the same manner. Recalling that the same initial predictive distribution is given as in figure 5.1, relevant iterations are depicted in figure 5.3.



(a) After Iteration 10

(b) After Iteration 15

(c) After Iteration 20

(d) After Iteration 30

Figure 5.3: Predictive Distributions from `SafeOpt` Algorithm after 5, 15, 20 and 30 Iterations

The first ten iterations depicted in figure 5.3a reveals the exploration of the local maximum while the safe set has been extended almost to the upper limit of the input space at $x = 10$. Five iterations later, exploratory character of `SafeOpt` is illustrated as the safe set kept expanding by on the one hand reaching the upper limit and on the other by approaching the global maximum. The local minimum at $x = 6.2$ has thus been overcome in contrast to `SafeUCB`. In figure 5.3c, the observation resulting from iteration 20 is shown to be in the near vicinity of the global maximum at $x = 3.54$. Moreover, the left limit of the safe set indicated by the intersection of the safety threshold and the true function at $x = 2.59$ has been closely approached. During the following ten iterations, the algorithm attempts to further expand the safe set as numerous observations around $x = 2.59$ can be found in figure 5.3d. Furthermore, light is shed on the exploiting character of `SafeOpt` that comes

to action when exploration is exhausted as the density of observations around the global maximum is increased. Note that no inputs $x > 4$ have been sampled for observation by the algorithm, as it successfully found the global maximum. This examination is supported by the convergence plot depicted in figure 5.4. Note that the best observation is generated after iteration 20 resulting in an approximately constant regret $R(n)$ afterwards.

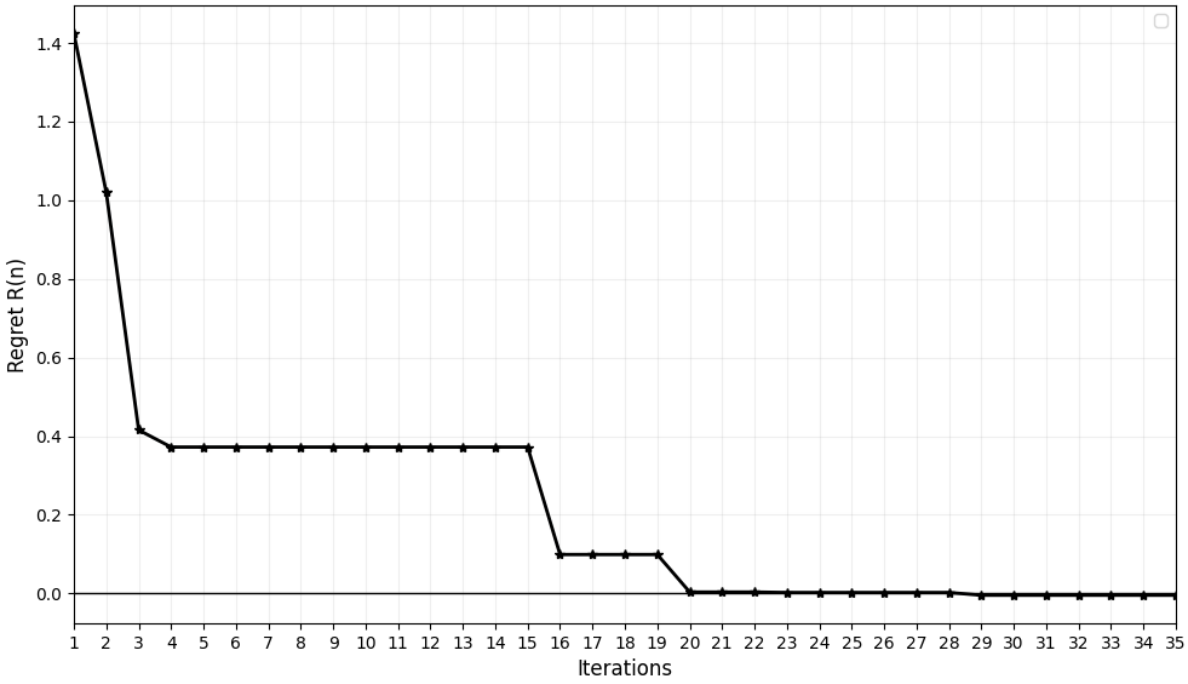


Figure 5.4: Convergence of `SafeOpt` Example Execution

Ultimately, the `SafeOpt` algorithm was able to determine the location of the global maximum due to its well handled trade-off of exploration and exploitation. In contrast, `SafeUCB` failed to do so suggesting that `SafeOpt` is more reliable in terms of assured convergence towards the global maximum. However, note that both algorithm did not violate safety constraints at any iteration demonstrating a successful implementation of both.

5.2 Conservatism in the Frequentist and Bayesian Setting

The convergence rates (if existent) of the `SafeOpt` and `SafeUCB` algorithms depending on the error bound constructions from the frequentist and bayesian settings are now of interest. Hereby, the incorporation of the SE kernel is initially analyzed. Afterwards the same algorithms equipped with the RQ kernel are considered to see if they validate the conclusions from the results generated by applying the SE kernel.

As afore mentioned, randomness is introduced in the experimental setup due to the randomized choice of initial conditions for the separate optimization problems of the `SafeOpt` and `SafeUCB` algorithms. Thus, each algorithm with each frequentist and bayesian

error bounds is executed 100 times to account for the randomness. Conservatism due to frequentist and bayesian settings is examined by computing the mean convergence of those 100 executions.

5.2.1 Optimizing the True Function from the SE Kernel

The mean convergence plots of running `SafeOpt` with frequentist and bayesian error bounds on the SE kernel are depicted in figures 5.5 and 5.6, respectively.

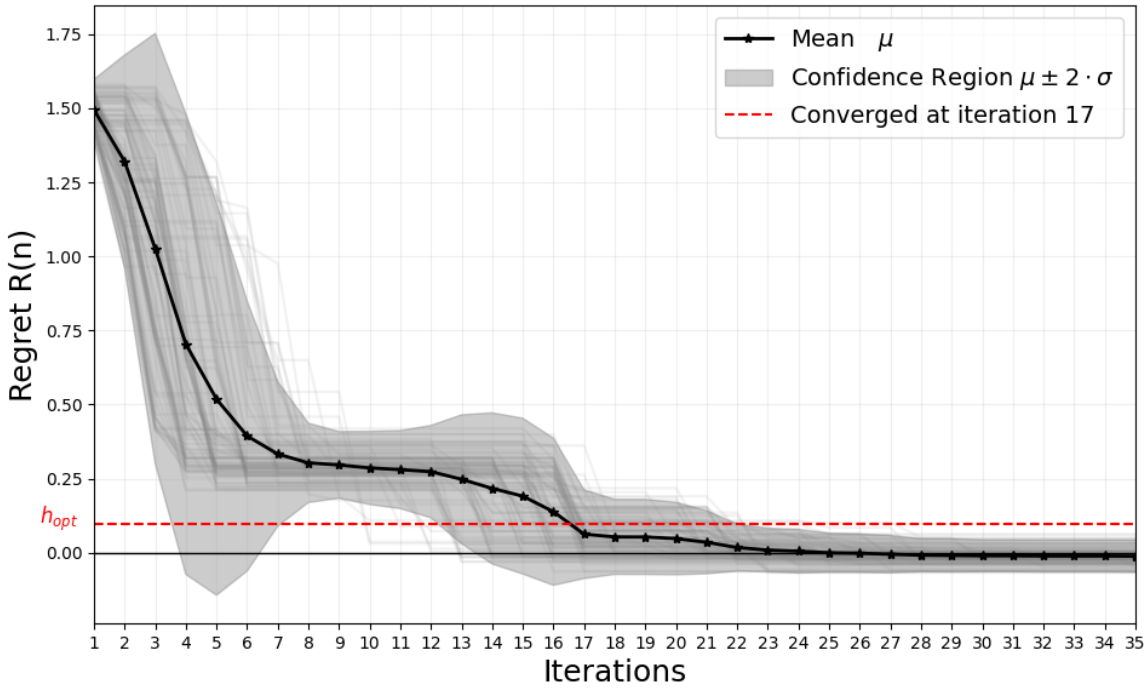


Figure 5.5: Mean Convergence of Frequentist `SafeOpt` with SE Kernel

Both plots portray the mean convergence μ at each iteration by the black dotted line. The light black lines depict the individual executions. The shaded area corresponds to the confidence region computed by adding and subtracting the associated standard deviation σ.

For the frequentist setting, figure 5.5 exhibits a steep decay of regret until iteration eight reaching $R(8) = 0.29$ suggesting that the local maximum has first been explored. Next, another significant decay can be observed between iterations 12 and 17 where the algorithm further approach the global maximum. Finally, at iteration 17 the convergence criterion defined in section 4.1 is reached as the regret drops below the optimality threshold $R(n = 17) < h_{opt}$. During the remaining iterations the regret is further decreased. A note on the individual execution must be made. Some of light black lines reach a regret that satisfies optimality already after significantly fewer iterations. One example can be seen around iteration ten. However, other reach convergence after significantly more iterations as the light black lines exhibit decay above the h_{opt}. Thus, the choice of initial conditions

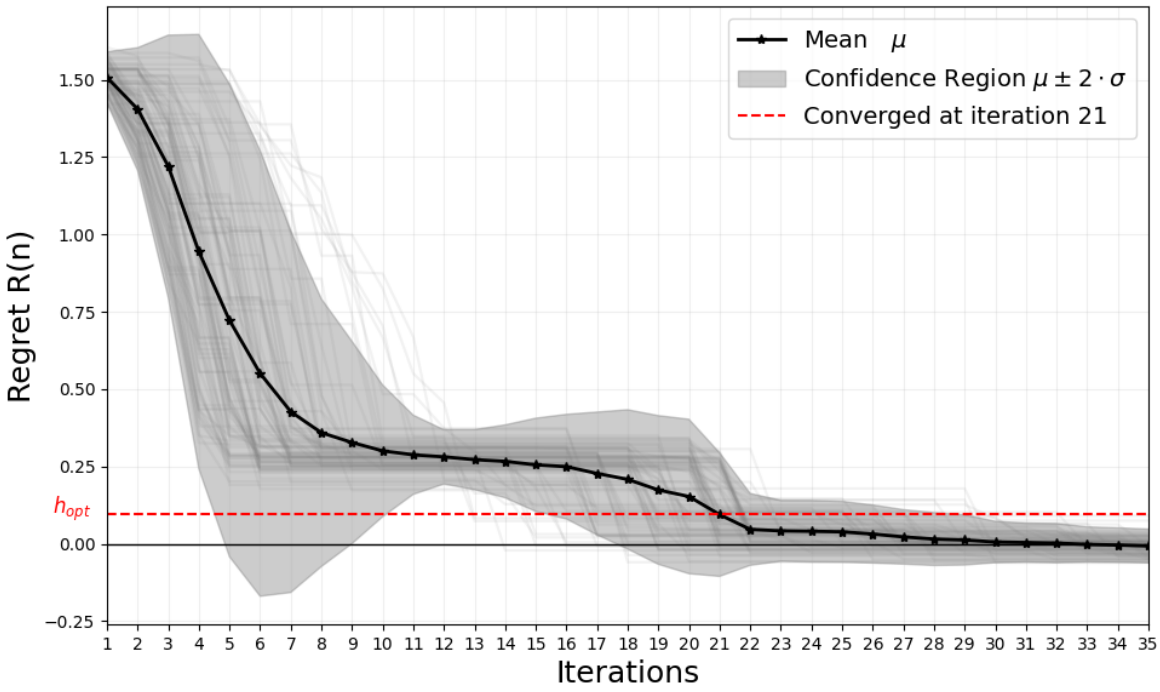


Figure 5.6: Mean Convergence of Bayesian `SafeOpt` with SE Kernel

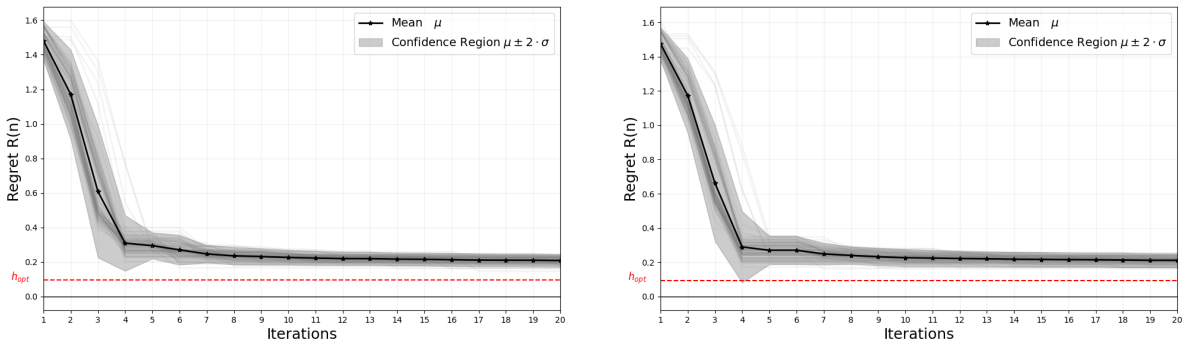
in this experimental setup heavily influences the convergence supporting the need for a statistical performance evaluation.

In the bayesian setting, figure 5.6 exhibits a slightly less steep decay reaching a regret of $R(n = 8) = 0.34$ after iteration eight compared to the frequentist convergence plot. However, similarly to the frequentist results the algorithm exhibits a low decay of regret between iteration nine and 16 before decreasing more rapidly towards its convergence at iteration 21. For the setup that applies the SE kernel, the frequentist error bounds result in a convergence rate that is thus 19.05% faster than the bayesian setting allowing the conclusion that the bayesian error bounds yield increased conservatism throughout the entire algorithm execution.

To reason this increased conservatism of the bayesian setting, its assumptions are recalled. First, it assumes the optimized ground truth f to be a sample of a GP. That assumption prohibits the use of any RKHS bound dependent error bound construction as the RKHS norm of a GP sample diverges to infinity. Secondly, the bayesian setting assumes Lipschitz continuity of the true function and the kernel function, providing their respective Lipschitz constant. As stated afore, Lipschitz based methods tend to increase conservatism naturally. Within the frame of error bounds for safe BO algorithm, this has been supported by the statistical results of this work. The application of Lipschitz constants in error bounds can be thought as some sort of 'worst case scenario'. The Lipschitz constant of the true function L_f and the kernel L_k are generated by computing the largest change of function values w.r.t. to their inputs. Thus, the highest change rate is applied among the entire input domain when computing input dependent error bounds. Subsets of the input domain

exhibiting much lower change in their respective function values are hence treated with the same assumption of change as the highest change among the entire set.

For completion, the mean convergences of the `SafeUCB` algorithm equipped with the SE kernel are depicted in figure 5.7.



(a) Frequentist

(b) Bayesian

Figure 5.7: Mean Convergence of Frequentist and Bayesian `SafeUCB` with SE Kernel

The demonstration of an example execution of `SafeUCB` with the SE kernel under the frequentist setting has revealed that it failed to converge globally. The mean convergence in figure 5.7a confirms that finding. Moreover, the same holds for the `SafeUCB` with the SE kernel under the bayesian setting as no convergence is achieved in figure 5.7b.

5.2.2 Optimizing the True Function from the RQ Kernel

Next, the results from applying the RQ kernel are examined to see if the findings so far can be supported. Figures 5.8 and 5.9 depict the mean convergence of the frequentist and bayesian setting applied in `SafeOpt` in the same manner as described for the SE kernel.

Comparing both mean convergences from the RQ to the results of the SE kernel, one can note that the decay of regret is generally smoother when applying the RQ kernel. Deviations in decay are more constant which is unlikely due to the RQ kernel itself but rather due to the true function f_{RQ} in combination with the choice of initial safe input which was appointed to be $x_{0,RQ} = 5.6$. Furthermore, it can be observed that the frequentist mean regret in figure 5.8 converges at iteration 14 while the bayesian mean regret reaches convergence four iterations later, see figure 5.9. Thus, frequentist `SafeOpt` converges 22.21% faster than its bayesian counterpart exhibiting a similar performance advantage as for the SE kernel.

For completion, the results from the `SafeUCB` algorithm with the RQ kernel are demonstrated. Figure 5.10 thereby depicts the mean convergence from executing `SafeUCB` with frequentist and bayesian error bounds using the RQ kernel.

In contrast to the frequentist and bayesian `SafeUCB` with the SE kernel, the application of the RQ kernel results in convergence for both settings. However, it is interesting to note that only the computed mean converges not the entire set of individual executions.

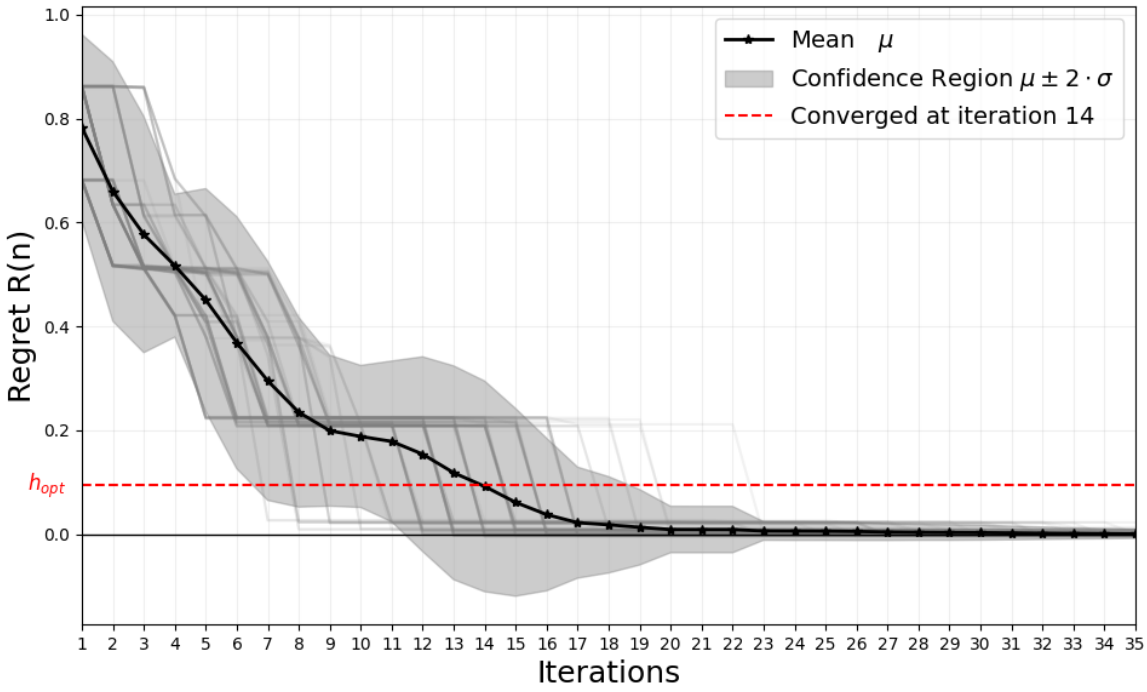


Figure 5.8: Mean Convergence of Frequentist `SafeOpt` with RQ Kernel

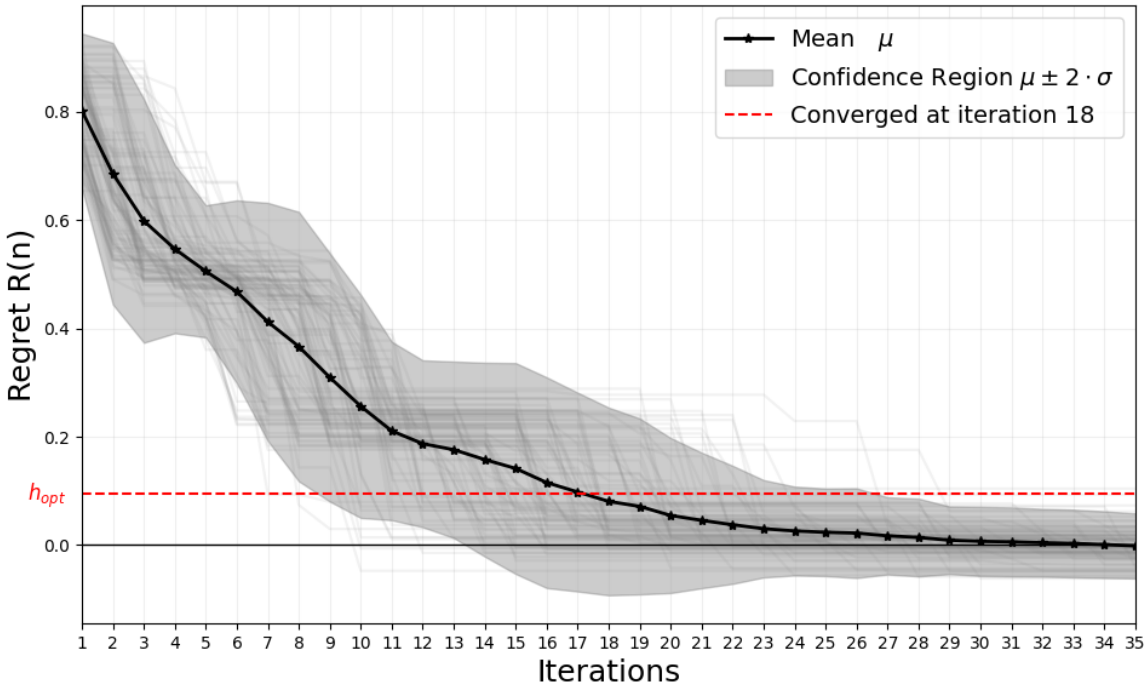
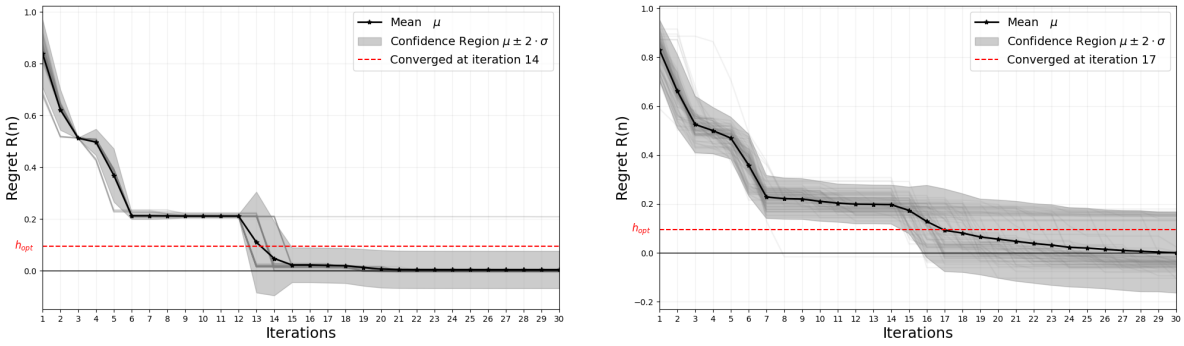


Figure 5.9: Mean Convergence of Bayesian `SafeOpt` with RQ Kernel

Examining the light black lines in figure 5.10, the regret of some is well above the convergence criterion h_{opt} at roughly $R(n) \approx 0.2$. This is most likely due to the randomized



(a) Frequentist

(b) Bayesian

Figure 5.10: Mean Convergence of Frequentist and Bayesian `SafeUCB` with RQ Kernel

choice of initial conditions which appear to yield in varying global convergence behavior. Nonetheless, the bayesian setting exhibits slower convergence than the frequentist setting as the former mean convergence is reached at iteration 17 and for the latter at iteration 14 thus complying with previous findings.

For a overview of the obtained results, table 4 contains the convergence iteration of each combination of setting, algorithm and true function depending on kernel choice.

Setting	**Algorithm**	**Kernel / True Function**	**Convergence Iteration**
Frequentist	Safe Opt	k_{SE}, f_{SE}	17
		k_{RQ}, f_{RQ}	14
	Safe UCB	k_{SE}, f_{SE}	None
		k_{RQ}, f_{RQ}	14
Bayesian	Safe Opt	k_{SE}, f_{SE}	21
		k_{RQ}, f_{RQ}	18
	Safe UCB	k_{SE}, f_{SE}	None
		k_{RQ}, f_{RQ}	17

Table 4: Comparison of Algorithms and Convergence Iterations

In conclusion, frequentist error bounds yield in a significant faster mean convergence as its bayesian counterpart when implemented in `SafeOpt` independently on kernel choice. The convergence of frequentist and bayesian `SafeUCB` in this experimental setup was shown to depend on the kernel choice and the resulting ground truth. For the SE kernel and its associated true function f_{SE}, no convergence was achieved applying `SafeUCB`. In contrast, frequentist and bayesian `SafeUCB` have reached convergence when optimizing f_{RQ} resulting from the RQ kernel. Hereby, the frequentist setting exhibited a 17.65% faster convergence. However, convergence was not attained for each execution but solely when considering the

mean convergence giving rise to the influence of randomized choice of initial conditions. It is hence concluded that bayesian error bounds perform weaker in terms of convergence rates than frequentist bounds due to their conservative assumptions about the ground truth and the associated incorporation of Lipschitz constants. Nonetheless, it must be noted that from a practitioner's point of view, the assessment of relevant parameters for the bayesian error bounds is a less complex task than for frequentist bounds. This is due to lacking methods to reliably bound the RKHS norm of an unknown, continuous function. Finally, no executed algorithm has evaluated unsafe inputs at any iteration certifying a successful implementation of `SafeOpt` and `SafeUCB` w.r.t. various settings and kernel choices.

6 Conclusion and Outlook

This work has investigated efficient and safe exploration methods for Gaussian Processes in constrained Bayesian Optimization. A comparative analysis of frequentist and bayesian assumptions about the ground truth resulting in different formulations of uncertainty quantification was conducted in order to asses their introduced conservatism and thus impact on efficiency.

An elaborate literature review revealed reasonable choices of safe BO algorithms to implement for this work, namely the `SafeOpt` and `SafeUCB` algorithm. Their safety constraint satisfaction and global convergence behavior has been examined under the frequentist and bayesian setting implementing two different kernels for increased numerical validation, the SE and RQ kernel.

Frequentist uncertainty quantification is based on assuming the true function to be a member of the kernel associated RKHS with bounded RKHS norm whereas the bayesian setting leverages Lipschitz continuity by assuming the ground truth to be a sample of the GP in place and thus deriving Lipschitz continuity of posterior distributions and samples. The application of Lipschitz continuity and their respective constants gave rise to expecting increased conservatism and reduced efficiency for the bayesian setting. Therefore, the experimental setup aimed to reveal whether or not this assumption is justifiable and if so, to what extent is efficiency, measured by convergence rates, reduced.

The findings indicate that the frequentist setting yields in less conservative uncertainty bounds and thus accelerated convergence rates in comparison to the bayesian setting. This holds for the experimental results of executing `SafeOpt` independently of kernel choice. The frequentist `SafeOpt` equipped with the SE kernel required 19.05% and 22.21% for the RQ kernel fewer iterations than its bayesian counterpart to converge. Contrarily, the `SafeUCB` algorithm exhibited strong limitations in global convergence as it failed to convergence for the frequentist and bayesian setting incorporating the SE kernel. Global convergence has been achieved when applying the RQ, however only globally by computing the mean convergence. Individual executions have shown to converge locally. The necessity for reliable global convergence by proper handling of the exploration-exploitation dilemma has thus been underscored.

An important conclusion of this work is the significant impact of balancing conservatism by uncertainty quantification on the efficiency of safe BO algorithms. While frequentist assumptions result in superior optimization performance, their practical applicability is challenged by the estimation of the required RKHS norm bound of real processes. The parameterization of the bayesian setting requires advanced domain knowledge but is nonetheless more intuitive exhibiting practical advantages over the frequentist setting.

Looking ahead, this work forms a solid base for further progress in experimental validation of frequentist and bayesian uncertainty quantification strategies. As this experimental setup introduced some randomness, the experiments conducted should be extended to a larger scale evaluating an increased amount of varying true functions and kernels to ideally support the findings of this work.

Moreover, further research could be conducted to further tighten the demonstrated error bounds as margins between uncertainty bounds and local or global optima even under ideal assumption remain notably large.

In conclusion, this work contributes to the progressive development of safe and efficient BO methods by systematically quantifying differences in uncertainty quantification technique and providing empirical insight into their performance. Thereby, the results emphasize the necessity of properly choosing methods of uncertainty bound construction to reach adequate performance within the broader applicability of BO in constrained optimization tasks.

References

[1] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005, ISBN: 026218253X.

[2] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization", *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016. DOI: 10.1109/JPROC.2015.2494218.

[3] Y. Sui, A. Gotovos, J. Burdick, and A. Krause, "Safe exploration for optimization with Gaussian processes", in *32nd Int. Conf. Mach. Learn. (ICML)*, ser. Proceedings of Machine Learning Research, vol. 37, Lille, France: PMLR, Jul. 2015, pp. 997–1005.

[4] F. Berkenkamp, A. P. Schoellig, and A. Krause, "Safe controller optimization for quadrotors with gaussian processes", *Proc. of the IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 491-496*, pp. 491–496, Sep. 3, 2015. DOI: 10.1109/icra.2016.7487170. arXiv: 1509.01066 [cs.RO].

[5] E. R. Cole, M. J. Connolly, M. Ghetiya, *et al.*, "Safe-opt: A bayesian optimization algorithm for learning optimal deep brain stimulation parameters with safety constraints.", eng, *Journal of neural engineering*, vol. 21, 4 Aug. 2024.

[6] J. Menn, P. Pelizzari, M. Fleps-Dezasse, and S. Trimpe, "Lipschitz safe bayesian optimization for automotive control", Jan. 2025. DOI: 10.48550/ARXIV.2501.12969. arXiv: 2501.12969 [eess.SY].

[7] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design", Dec. 21, 2009. DOI: 10.1109/TIT.2011.2182033. arXiv: http://arxiv.org/abs/0912.3995v4 [cs.LG].

[8] C. Fiedler, C. Scherer, and S. Trimpe, "Practical and rigorous uncertainty bounds for gaussian process regression", May 2021.

[9] S. R. Chowdhury and A. Gopalan, "On kernelized multi-armed bandits", in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, Jun. 2017, pp. 844–853. [Online]. Available: https://proceedings.mlr.press/v70/chowdhury17a.html.

[10] A. Lederer, J. Umlauft, and S. Hirche, "Uniform error bounds for Gaussian process regression with application to safe control", in *Advances in Neural Information Pro-cessing Systems*, Jun. 2019, pp. 659–669.

[11] J. O. Luebsen and A. Eichler, "An analysis of safety guarantees in multi-task bayesian optimization", Mar. 2025. DOI: 10.48550/ARXIV.2503.08555. arXiv: 2503.08555 [cs.LG].

[12] M. P. Deisenroth, "Efficient reinforcement learning using gaussian processes", Ph.D. dissertation, 2010, 205 pp., ISBN: 978-3-86644-569-7. DOI: 10.5445/KSP/1000019799.

[13] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis*. Chapman and Hall/CRC, Nov. 2013, ISBN: 9780429113079. DOI: 10.1201/b16018.

[14] M. L. Stein, *Interpolation of Spatial Data*. Springer New York, 1999, ISBN: 9781461214946. DOI: 10.1007/978-1-4612-1494-6.

[15] A. Hadji and B. Szábo, "Can we trust bayesian uncertainty quantification from gaussian process priors with squared exponential covariance kernel?", Apr. 2019. DOI: 10.48550/ARXIV.1904.01383. arXiv: 1904.01383 [math.ST].

[16] M. Abramowitz and I. Stegun, *Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables* (Applied mathematics series). Dover Publications, 1965, ISBN: 9780486612720. [Online]. Available: https://books.google.de/books?id=MtU8uP7XMvoC.

[17] Z. Chen and B. Wang, "How priors of initial hyperparameters affect gaussian process regression models", *Neurocomputing*, vol. 275, pp. 1702–1710, 2018, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2017.10.028. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092523121731679X.

[18] V. Lalchand, W. Bruinsma, D. Burt, and C. E. Rasmussen, "Sparse gaussian process hyperparameters: Optimize or integrate?", in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 16 612–16 623. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/69c49f75ca31620f1f0d38093d9f3d9b-Paper-Conference.pdf.

[19] Z. Wang, G. E. Dahl, K. Swersky, *et al.*, "Pre-trained gaussian processes for bayesian optimization", *Journal of Machine Learning Research, 25(212):1-83, 2024. URL http://jmlr.org/papers/v25/23-0269.html*, Sep. 2021. DOI: 10.48550/ARXIV.2109.08215. arXiv: 2109.08215 [cs.LG].

[20] N. Aronszajn, "Theory of reproducing kernels", *Transactions of the American Mathematical Society*, vol. 68, no. 3, pp. 337–404, 1950, ISSN: 00029947, 10886850. [Online]. Available: http://www.jstor.org/stable/1990404 (visited on 02/13/2025).

[21] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley, "Reproducing kernel hilbert space, mercer's theorem, eigenfunctions, nyström method, and use of kernels in machine learning: Tutorial and survey", Jun. 2021. DOI: 10.48550/ARXIV.2106.08443. arXiv: 2106.08443 [stat.ML].

[22] R. Garnett, "Introduction", in *Bayesian Optimization*. Cambridge University Press, 2023, pp. 1–14.

[23] R. Horst and P. Pardalos, *Handbook of Global Optimization* (Nonconvex Optimization and Its Applications). Springer US, 2013, ISBN: 9781461520252. [Online]. Available: https://books.google.de/books?id=yBDaBwAAQBAJ.

[24] H. J. Kushner, "A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise", *Journal of Basic Engineering*, vol. 86, pp. 97–106, 1964. [Online]. Available: https://api.semanticscholar.org/CorpusID:62599010.

[25] D. J. Lizotte, "Practical bayesian optimization", AAINR46365, Ph.D. dissertation, CAN, 2008, ISBN: 9780494463659.

[26] J. Močkus, "On bayesian methods for seeking the extremum", in *Optimization Techniques IFIP Technical Conference Novosibirsk, July 1–7, 1974*. Springer Berlin Heidelberg, 1975, pp. 400–404, ISBN: 9783540374978. DOI: 10.1007/3-540-07165-2_55.

[27] D. Zhan and H. Xing, "Expected improvement for expensive optimization: A review", *Journal of Global Optimization*, vol. 78, no. 3, pp. 507–544, 2020, ISSN: 1573-2916. DOI: 10.1007/s10898-020-00923-x. [Online]. Available: https://doi.org/10.1007/s10898-020-00923-x.

[28] H. Wang, "Modifications of PI and EI under gaussian noise assumption in current optima", in *Proceedings of The 2nd International Conference on Signal Processing and Machine Learning, Virtual Event, Oxford, United Kingdom, May 12-18, 2022*, Y. Xing and T. Ratnarajah, Eds., ser. CEUR Workshop Proceedings, vol. 3150, CEUR-WS.org, 2022, pp. 85–97. [Online]. Available: http://ceur-ws.org/Vol-3150/paper2.pdf.

[29] A. Bull, "Convergence rates of efficient global optimization algorithms", *Journal of Machine Learning Research*, vol. 12, Jan. 2011.

[30] A. A. Pawar and U. Warbhe, "Optimizing bayesian acquisition functions in gaussian processes", Nov. 2021. DOI: 10.48550/ARXIV.2111.04930. arXiv: 2111.04930 [cs.LG].

[31] H. Robbins, "Some aspects of the sequential design of experiments", *Bulletin of the American Mathematical Society*, vol. 58, no. 5, pp. 527–535, 1952, ISSN: 1088-9485. DOI: 10.1090/s0002-9904-1952-09620-8.

[32] C. Mallows and H. Robbins, "Some problems of optimal sampling strategy", *Journal of Mathematical Analysis and Applications*, vol. 8, no. 1, pp. 90–103, 1964, ISSN: 0022-247X. DOI: https://doi.org/10.1016/0022-247X(64)90087-3. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0022247X64900873.

[33] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules", *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 4–22, Mar. 1985, ISSN: 0196-8858. DOI: 10.1016/0196-8858(85)90002-8.

[34] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem", *Machine Learning*, vol. 47, no. 2, pp. 235–256, May 2002, ISSN: 1573-0565. [Online]. Available: https://doi.org/10.1023/A:1013689704352.

[35] R. Agrawal, "Sample mean based index policies with o(log n) regret for the multi-armed bandit problem", *Advances in Applied Probability*, vol. 27, no. 4, pp. 1054–1078, 1995, ISSN: 00018678. [Online]. Available: http://www.jstor.org/stable/1427934 (visited on 01/23/2025).

[36] A. Krause and C. Ong, "Contextual gaussian process bandit optimization", in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., vol. 24, Curran Associates, Inc., 2011. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2011/file/f3f1b7fc5a8779a9e618e1f23a7b7860-Paper.pdf.

[37] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms", Jun. 2012. DOI: 10.48550/ARXIV.1206.2944. arXiv: 1206.2944 [stat.ML].

[38] F. Berkenkamp, A. Krause, and A. P. Schoellig, "Bayesian optimization with safety constraints: Safe and automatic parameter tuning in robotics.", eng, *Machine learning*, vol. 112, pp. 3713–3747, 10 2023.

[39] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley, Apr. 2005, ISBN: 9780471748823. DOI: 10.1002/047174882x.

[40] C. Fiedler, J. Menn, L. Kreisköther, and S. Trimpe, "On safety in safe bayesian optimization", Mar. 19, 2024. arXiv: http://arxiv.org/abs/2403.12948v1 [cs.LG].

[41] S. Ghosal and A. Roy, "Posterior consistency of gaussian process prior for nonparametric binary regression", *Annals of Statistics 2006, Vol. 34, No. 5, 2413-2429*, vol. 34, no. 5, Feb. 23, 2007, ISSN: 0090-5364. DOI: 10.1214/009053606000000795. arXiv: math/0702686 [math.ST].

[42] S. Grünewälder, J.-Y. Audibert, M. Opper, and J. Shawe–Taylor, "Regret bounds for gaussian process bandit problems", in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Y. W. Teh and M. Titterington, Eds., ser. Proceedings of Machine Learning Research, vol. 9, Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 273–280. [Online]. Available: https://proceedings.mlr.press/v9/grunewalder10a.html.

[43] R. R. Duivenvoorden, F. Berkenkamp, N. Carion, A. Krause, and A. P. Schoellig, "Constrained bayesian optimization with particle swarms for safe adaptive controller tuning", *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 11 800–11 807, 2017, 20th IFAC World Congress, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2017.08.1991. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896317326290.

[44] M. Zagorowska, E. C. Balta, V. Behrunani, A. Rupenyan, and J. Lygeros, "Efficient sample selection for safe learning*", *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 10 107–10 112, 2023, 22nd IFAC World Congress, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2023.10.882. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896323012624.

[45] I. Steinwart and A. Christmann, *Support Vector Machines* (Information Science and Statistics). Springer New York, 2008, ISBN: 9780387772424. [Online]. Available: https://books.google.de/books?id=HUnqnrpYt4IC.