LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
DEPARTMENT OF STATISTICS

Robust Generalizations of Stochastic Derivative-Free Optimization

Submitted in accordance with the requirements for the degree of Master of Science (M.Sc.)

Author Julian Rodemann
Supervisor Prof. Dr. Thomas Augustin
Submission July 2021

Abstract

Many derivative-free optimization methods rely on surrogate models. In this thesis, we study the effect of such models' specifications on the optimizers' convergence. We focus on the example of Gaussian processes as surrogate models in Bayesian optimization. Here, we find the prior's mean parameters to have the highest influence on convergence among all prior components. In response to this result, we propose four generalizations of Bayesian optimization that aim at rendering the method more robust towards prior mean misspecification. Two of them, the generalized lower confidence bound and a weighted maximum likelihood approach, outperform standard Bayesian optimization for specific types of problems.

Contents

Acknowledgments

I would like to thank everybody whom I pestered with questions on the matter of this thesis. First and foremost, this includes the kind and empowering supervision by Thomas Augustin. I express my gratitude to his whole research group "Foundations of Statistics and Their Applications" for providing valuable feedback during colloquia. I also thank Lars Kotthoff, Bernd Bischl, Julia Moosbauer and Martin Binder, who supervised our statistical consulting project which sparked my interest in Bayesian optimization. I especially thank Lars Kotthoff for kindly providing data for this thesis (section 6.6) and Martin Binder for taking the time to critically assess the herein proposed methods. Furthermore, I would like to express my appreciation of all the inspiring suggestions I received during ISIPTA 2021, which I could only partly consider in this thesis, from Jasper de Bock, Alessio Benavoli and Martin Jann, to name only a few. I would also like to extend my gratitude to Francesca Mangili for patiently answering questions regarding imprecise Gaussian Processes (IGPs).

Moreover, I would like to thank my family and my friends who supported me writing this thesis and, if needed, took my mind off things. Thanks to my parents, my sister and my brother for their unconditional support throughout my studies. Thanks to my loyal companions from high school times Florian Schnell and Simon Kirschner for distractions of whatever nature. I also thank Hendrik Cech for both emotional and technical (java) advice.

Notably, the set of friends and the set of people whom I bothered with statistical questions intersect. I am most grateful to my fellow students and good friends Federico Croppi, Alexander Piehler, Sebastian Fischer, Esteban Garcés Arias and Moritz Wagner for enthralling statistical discussions in the *Kunstakademie* or the institute's CIP-Pools. Without their critical assessment of my often misleading intuitive understanding, I would have neither passed all exams nor written this thesis. What is more, I would only have had half as much fun. Beyond any doubt, I owe special thanks to Federico Croppi in this regard, whose thirst for questioning the seemingly simple has helped me from *Eignungstest* to this very thesis.

I conclude by conveying my thanks to the scholarship program of Evangelisches Studienwerk Villigst for both the material and immaterial support of my studies.

List of Figures

Image Credits: If not otherwise stated, all figures are based on own computations using `ggplot2` [Wickham, 2016], `tidyverse` [Wickham et al., 2019], `RColorBrewer`, `ggsci`, `ggshadow` and `boot` [Canty and Ripley, 2021] (for CIs) in R [R Core Team, 2020]. Code to reproduce figures can be found on `www.github.com/rodemann/master-thesis-r`.

List of Tables

List of Algorithms

List of Definitions, Theorems and Proofs

1 Introduction

1.1 Scientific Motivation

> *"For this, indeed, is the main source of our ignorance – the fact that our knowledge can be only finite, while our ignorance must necessarily be infinite."*
>
> — Karl R. Popper [Popper, 2014]

Acetylsalicylic acid, marketed under the name aspirin, has been easing pain, curing headaches and reducing fever since the beginning of the 20$^{\text{th}}$ century. It has been an indispensable resource in modern societies since then. Its mechanism of action, however, was not discovered until the end of the century, starting with findings by [Vane, 1971] and extending to recent work by [Hawley et al., 2012].

In a thought-provoking essay in the *New Yorker* [Zittrain, 2019] argues that this approach to discovery, which he casually refers to as "answers first, explanations later", has become ubiquitous in machine learning (ML). He describes the *modus operandi* in ML research as discovering what works without knowing why it works, and then putting "that insight to use immediately, assuming that the underlying mechanism will be figured out later" [Zittrain, 2019]. The so-acquired burden of unexplained phenomena is named "intellectual dept" by him. Unlike in medical and other scientific areas, Zittrain argues, such theory-free advances are an intrinsic part of "statistical-correlation engines" in machine learning. He paints a bleak picture of ML's future: With a growing number of unknown mechanisms in complex systems, "the number of tests required to uncover untoward interactions must scale exponentially" [Zittrain, 2019].

Indeed, Zittrain brings up a painful subject for machine learning research. Interpreting machine learning models was long considered a research niche. This is about to change, some say, with causal inference being one of the main drivers [Peters et al., 2017]. Be that as it may, we argue the lack of interpretation is not the mere cause for increasing the intellectual credit line in machine learning, but also the hidden assumptions upon which many models rely. While influential with regard to the model's predictions, many assumptions are hardly questioned, let alone empirically tested.

The thesis at hand will demonstrate the influence of unquestioned assumptions with regard to the example of Bayesian optimization (BO), a popular stochastic derivative-

free optimization method for hyperparameter-tuning of machine learning models. As a consequence, it will outline venues of how to make BO and other derivative-free optimizers more robust towards altering their underlying assumptions. To accomplish that, a representation of partial or no knowledge about the model specification is needed. The fruitful framework of imprecise probabilities (IP) offers a way to do so for Gaussian process (GP), a functional regression approach that is essential to BO.

By this means, we will account for a set of GPs as surrogate models and thus make the optimizer more robust against their misspecification. Despite the fact that, in practice, the model is often specified arbitrarily[1], we will show that model choice can have a great influence on the optimization's convergence. One (if not *the*) founding father of the method, Jonas Močkus, has proclaimed that "the development of some system of *a priori* distributions suitable for different classes of the function f is probably the most important problem in the application of [the] Bayesian approach to (...) global optimization" [Močkus, 1975], cited after [Malkomes and Garnett, 2018].

From a philosophical point of view, such representations of non-informativeness are reminiscent of the great Socratic paradox "I know that I know nothing", handed down in Plato's *Apology* and translated by Cicero. Interestingly, recent research [Fine, 2008] questions whether Socrates actually claimed knowledge about the unknown, such that this knowledge is the only exception from his ignorance. Cicero's inaccurate translation ("I know that"), so the argument goes, clouded the view on what Plato actually delivered to posterity. His Greek original merely states that Socrates complained about the absence of true wisdom, with no additional assertions of certainty about the former.

This intriguing footnote to ancient philosophy should make all statisticians sit up and take notice. They are familiar with the age-old debate on how to represent uninformativeness in statistical models. The traditional Bayesian approach consists of the formulation of so-called non-informative priors. However, they fail at describing a situation of real prior ignorance, as they describe a *precise* prior belief [Mangili, 2016, Page 154], i.e. produce a precise probability value for any $A \subseteq \Omega$. Such priors thus often reflect *indifference* between parameter values rather than *ignorance* [Augustin et al., 2014, Page 158].[2] To put it in a philosophical way, they precisely state knowledge about the unknown, as in Cicero's wrong translation ("I know that I know nothing").[3]

[1] See default choices in popular libraries like `spearmint` (python) or `mlrMBO` (R).

[2] This concept is often referred to as the principle of indifference [Fischer, 2021]. For a comprehensive discussion of ignorance and indifference, the interested reader is referred to [Norton, 2008].

[3] Note that the Bayesian framework quite well allows for modeling ignorance about ignorance by means of hyperpriors, see section 5.2. Hyperpriors, however, again represent *precise* prior statements.

Seeking a statistical expression of Socrates' real intention, we make the case for imprecise probabilities (IP). They allow for the original representation of ignorance. That is, they can model uninformativeness without the necessity of making assured statements on its precise nature, but only on its mere existence. The interested reader can consult [Augustin et al., 2014] for a comprehensive introduction to IP.

But why, after all, is accounting for the unknown worth the methodological effort? The answers read surprisingly simple: first and foremost, because the alternative would be to make unjustified assumptions that result in false confidence, possibly wrong decisions and – in Zittrain's words – "intellectual dept". In the case of stochastic derivative-free optimizations methods such as BO this can slow down convergence, leading to suboptimal configurations of ML models for instance. Second, unexplained phenomena and resulting situations of ignorance outnumber those where (not necessarily precise) prior knowledge exists; see Popper's quote above. Throughout the history of human thought, accounting for the unknown has been an integral part of scientific considerations. More than that, following [Popper, 1992], it is *the* driving force behind scientific advancements. Popper refers to Xenophanes [Popper et al., 2013] and Socrates [Popper, 1991] as fallibilistic pioneers of his theory of critical rationalism and connects the dots between falsification in modern science and the ancient acceptance of ignorance.[4]

It might appear surprising to account for ignorance in optimization of all things relevant to ML. In his insightful introduction to machine learning, [Domingos, 2012] identifies three key components of ML models: representation, evaluation and optimization. Interestingly, each of the former two, but not the latter, is regarded as a core area of statistics. While ML practitioners turn to statisticians for advice on how to interpret the hypothesis space (representation) or make the loss function more robust (evaluation), optimization is apparently left to the numerically trained computer scientists. Yet, many popular derivative-free (hyperparameter-)optimizers heavily rely on probabilistic elements, be they advanced surrogate models or simple probability measures.

The thesis at hand will argue that in both cases, the flexibility of the optimization path can be increased by relaxing the assumptions about the probabilistic elements by means of Imprecise Probabilities (IP). Leaning on the famous quotation from [Manski, 2003], "The credibility of inference decreases with the strength of the assumptions maintained", it will be demonstrated that a relaxation of the assumptions can increase

[4] See also [Popper, 1987, page 52]: "Das Wissen im Sinne der Naturwissenschaft ist Vermutungswissen; es ist ein kühnes Raten. So behält Sokrates recht, trotz Kants verständnisvoller Einschätzung der Riesenleistung Newtons. Aber es ist ein Raten, das durch rationale Kritik diszipliniert wird."

some optimizers' modeling capacity and hence their performance.

1.2 Outline

This thesis' contributions are twofold. On the one hand, it *explores* venues of how to render optimizers more robust with regard to their model specification (section 10). On the other hand, it *exploits* the above described concept's potential with respect to the example of Bayesian optimization with Gaussian process by proposing four modifications and benchmarking them to existing predominant methods. The mental warehouse required for doing so, namely Bayesian optimization and Gaussian process, is thoroughly revised (sections 2 and 3). The main problem of how to specify the GP's prior inside BO under ignorance is analyzed in simulation studies (section 4) and then formally described (section 5), before turning to three attempts to solve it by means of IP, that are motivated, tested and discussed (section 6). Related work is compared to these proposals (section 7) and some possible extensions are briefly outlined (section 8). What is more, another method relying on likelihood estimation to render the prior specification of GP in BO more robust is put forward, empirically analyzed and critically assessed (section 9). Finally, the thesis concludes with a brief but conceptually overarching conclusion (section 11).

2 Bayesian Optimization

2.1 Basic Procedure

Bayesian optimization (BO)[5] is a popular method for optimizing functions that are expensive to evaluate and lack analytical description. BO approximates the objective function through a surrogate model (SM). In the case of real-valued covariates, Gaussian process (GP) regression is the most popular model, while random forest (RF) [Breiman, 2001] is usually preferred for categorical and mixed parameter spaces. Some authors like [Swersky et al., 2014][footnote 1] argue RF is also advantageous in case of many data points. BO scalarizes the surrogate model's mean and standard error estimates to an acquisition function (AF), that incorporates the trade-off between exploration (uncertainty reduction) and exploitation (mean optimization). The arguments of the AF's maxima[6] are eventually proposed to be evaluated. Algorithm 1 describes the basic procedure of Bayesian optimization applied on a problem of the sort: $\min_{\boldsymbol{x}\in\mathcal{X}} \Psi(\boldsymbol{x})$, where $\Psi : \mathcal{X}^p \to \mathbb{R},$, $\mathcal{X}^p$ a p-dimensional parameter space. Here and henceforth, minimization is considered without loss of generality.

Algorithm 1 Bayesian Optimization

1: create an initial design $D = \{(\boldsymbol{x}^{(i)}, \Psi^{(i)})\}_{i=1,\dots,n_{init}}$ of size n_{init}
2: **while** termination criterion is not fulfilled **do**
3: **train** a surrogate model (SM) on data D
4: **propose** $\boldsymbol{x}^{new}$ that optimizes the acquisition function $AF(SM(\boldsymbol{x}))$
5: **evaluate** Ψ on $\boldsymbol{x}^{new}$
6: **update** $D \leftarrow D \cup (\boldsymbol{x}^{new}, \Psi(\boldsymbol{x}^{new}))$
7: **end while**
8: **return** $\arg\min_{\boldsymbol{x}\in D} \Psi(\boldsymbol{x})$ and respective $\Psi(\arg\min_{\boldsymbol{x}\in D} \Psi(\boldsymbol{x}))$

Various termination criteria are conceivable with a pre-specified number of iterations being one of the most popular choices. Notably, line 4 imposes a new optimization problem, sometimes referred to as "auxiliary optimization", e.g. in [Frazier, 2018].

[5] Also called efficient global optimization (EGO) or model-based optimization (MBO). While both EGO and MBO were widespread names in the early years after [Jones et al., 1998] popularized the method, Bayesian optimization has gained ground after the much-cited paper of [Snoek et al., 2012] and is the most common name today.

[6] Without loss of generality. Notably, some software libraries internally minimize $-AF(SM(\boldsymbol{x}))$.

Compared to $\Psi(\boldsymbol{x})$, however, $AF(SM(\boldsymbol{x}))$ is analytically traceable. It is a deterministic transformation of the surrogate model's mean and standard error predictions, which are given by line 3. Thus, evaluations are cheap and optima can be retrieved through naive algorithms, such as grid search, random search or the slightly more advanced focus search[7], all of which simply evaluate a huge number of points that lie dense in $\mathcal{X}$.

2.2 Acquisition functions

There exist several acquisition functions, among which expected improvement (EI) and lower confidence bound (LCB) are the most popular. The most fundamental one is probability of improvement (PI). Definitions 1, 2 and 3 are based on [Bischl et al., 2014] as well as on [Hutter et al., 2018]. Definition 6 is based on [Cox and John, 1992].

Definition 1 (Probability of Improvement)
Let $\psi(\boldsymbol{x})$ be the surrogate model and Ψ_{min} the incumbent minimal function value. The probability of improvement (PI) of $\boldsymbol{x}$ is

$$PI(\boldsymbol{x}) = \mathbb{P}(\psi(\boldsymbol{x}) < \Psi_{min}).$$

When using a Gaussian process (GP) as SM, as assumed in what follows, the PI can be simplified. For each finite vector of function values $\Psi(\boldsymbol{x})$ we assume $\Psi(\boldsymbol{x}) \sim \mathcal{N}(\mu(\boldsymbol{x}), \mathrm{Var}(\boldsymbol{x}))$, where $\mu(\boldsymbol{x})$ is the mean function of Ψ at $\boldsymbol{x}$ and $\mathrm{Var}(\boldsymbol{x})$ is the variance function at $\boldsymbol{x}$. For our surrogate model $\psi(\boldsymbol{x})$ it is $\psi(\boldsymbol{x}) \sim \mathcal{N}\left(\widehat{\mu}(\boldsymbol{x}), \widehat{\mathrm{Var}}(\boldsymbol{x})\right)$, where $\widehat{\mu}(\boldsymbol{x}), \widehat{\mathrm{Var}}(\boldsymbol{x})$ are estimates from the posterior GP, see section 3. Since the variance function is typically estimated by the variance of the mean prediction function $\widehat{\mu}(\boldsymbol{x})$, we write $\widehat{\mathrm{Var}}(\boldsymbol{x}) = \mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))$.[8] This allows standardization of $\psi(\boldsymbol{x})$ and Ψ_{min} in $PI(\boldsymbol{x})$ as follows:

$$\mathbb{P}(\psi(\boldsymbol{x}) < \Psi_{min}) = \mathbb{P}\left(\frac{\psi(\boldsymbol{x}) - \widehat{\mu}(\boldsymbol{x})}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}} < \frac{\Psi_{min} - \widehat{\mu}(\boldsymbol{x})}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right) = \Phi\left(\frac{\Psi_{min} - \widehat{\mu}(\boldsymbol{x})}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right) \quad (1)$$

[7] Focus search iteratively shrinks the search space and applies random search, see [Bischl et al., 2017, page 7].

[8] GPs have the nice property of intrinsically estimating the posterior variance of the prediction function $\widehat{\mu}(\boldsymbol{x})$. In case of random forest (RF) as SM, additional bootstrap or jackknife-after-bootstrap is needed for variance estimation. For instance, jackknife-after-bootstrap is used in the R package `mlrMBO` [Bischl et al., 2017].

As convention dictates, Φ denotes the standard normal distribution function. Since Φ, Ψ_{min}, $\widehat{\mu}(\boldsymbol{x})$ and $\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}$ are given in line 4 of algorithm 1, it can be seen that $PI(\boldsymbol{x})$ is indeed computationally cheap to evaluate. It requires nothing but a simple function call with given arguments. Also note that the probability of improvement is 0 for already visited points, as for such points $\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))} = 0$ and $\Psi_{min} - \widehat{\mu}(\boldsymbol{x}) \leq 0$, thus

$$\Phi\left(\frac{\Psi_{min} - \widehat{\mu}(\boldsymbol{x})}{\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right) = \Phi(-\infty) = 0. \tag{2}$$

With the same line of reasoning it follows that for $\{\boldsymbol{x} : \Psi_{min} - \widehat{\mu}(\boldsymbol{x}) \leq 0\}$ the $PI(\boldsymbol{x})$ (counter-intuitively) decreases with $\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}$. This makes the PI a very exploitative AF. For a detailed theoretical analysis of the acquisition functions' impact on the trade-off between exploration and exploitation the interested reader is referred to [Rahat, 2019]. The most widely used AF is expected improvement (EI), which is closely related to PI.

Definition 2 (Expected Improvement)
Let $\psi(\boldsymbol{x})$ be the surrogate model and Ψ_{min} the incumbent minimal function value. The expected improvement of $\boldsymbol{x}$ is

$$EI(\boldsymbol{x}) = \mathbb{E}(\max\{\Psi_{min} - \psi(\boldsymbol{x}), 0\}).$$

This time, the improvement is bounded from below. Uncertainty estimates only enter if mean estimates imply real improvement. This prohibits the negative effect of increasing uncertainty for $\{\boldsymbol{x} : \Psi_{min} - \widehat{\mu}(\boldsymbol{x}) \leq 0\}$ and, thus, enforces exploration. EI was proposed by [Močkus, 1975, Pages 1-2], disguised as a utility function in a decision problem that captures the expected deviation from the extremum. It follows from this formulation that a point proposed according to expected improvement is Bayes-optimal in a given iteration. This early definition of BO with EI is very close to the modern formulation in definition 2 and algorithm 1. However, it lacked the idea of surrogate modeling and thus the simplifications that come with Gaussian processes (GPs). Namely, we can express $EI(\boldsymbol{x})$ in this case in closed form in a similar manner to equation 1:

$$EI(\boldsymbol{x}) = (\Psi_{min} - \widehat{\mu}(\boldsymbol{x}))\,\Phi\left(\frac{\Psi_{min} - \widehat{\mu}(\boldsymbol{x})}{\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right) + \sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}\,\phi\left(\frac{\Psi_{min} - \widehat{\mu}(\boldsymbol{x})}{\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right), \tag{3}$$

which can be derived by partial integration from definition 2 and where $\phi(\cdot)$ denotes the standard normal density function. The attentive reader will notice again that EI

equals 0 for points that have been visited already, just like in case of PI. What is more, it can be seen that EI is a weighted sum of (standardized) mean and standard error estimates, thus explicitly balancing exploitation and exploration. While this follows naturally from the expected deviation from the extremum and the GP assumptions in case of $EI(\boldsymbol{x})$, the same trade-off can also be gracelessly encoded by a direct weighted sum of $\widehat{\mu}(\boldsymbol{x})$ and $\sqrt{\text{Var}(\boldsymbol{x})}$ with weight τ. The AF lower confidence bound (LCB) does the latter.

Definition 3 (Lower Confidence Bound)
Let $\widehat{\mu}(\boldsymbol{x})$ and $\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}$ be the mean and standard error prediction functions of the surrogate model. The lower confidence bound[9] *of $\boldsymbol{x}$ is*

$$LCB(\boldsymbol{x}) = -\widehat{\mu}(\boldsymbol{x}) + \tau \cdot \sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}.$$

Unlike in the case of EI and PI, the user can manually guide the exploration-exploitation trade-off by setting τ. Notably, τ can also be scheduled, e.g. decreased over time. The idea is to explore $\mathcal{X}$ first, then exploit wisely selected regions in detail later. This gives rise to the so-called adaptive lower confidence bound (ALCB)[10], initially proposed by [Liu et al., 2012].

Definition 4 (Adaptive Lower Confidence Bound)
Let $\widehat{\mu}(\boldsymbol{x})$ and $\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}$ be the mean and standard error prediction functions of the surrogate model. The adaptive lower confidence bound in iteration $t \in \{1, ..., T\}$ is

$$ALCB(\boldsymbol{x})_t = -\widehat{\mu}(\boldsymbol{x}) + \tau_t \cdot \sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}, \tau_t \in \{\tau_1, ..., \tau_T\},$$

where typically $\tau_1 > \tau_2 > ... > \tau_T$.

The purely exploratory acquisition function standard error (SE) can also be regarded as a special case of LCB since it substantively corresponds to LCB with $\tau \to \infty$.

Definition 5 (Standard Error as AF)
The mere standard error prediction function $\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}$ can be used as acquisition function (AF)

$$SE(\boldsymbol{x}) = \sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}.$$

[9] We are aware that in the context of Bayesian surrogate models such as GPs, `credible` confidence bound would be the more appropriate wording, see e.g. [Benavoli et al., 2021]. However, as the SM can be any statistical model in general, we abstain from sticking to the terminology of Bayesian inference and remain general.

[10] Also called self-adaptive lower confidence bound.

It is worth mentioning that LCB was proposed by [Cox and John, 1992] in a more general way as statistical lower bound (SLB).

Definition 6 (Statistical Lower Bound)
Let $\widehat{\mu}(\boldsymbol{x})$ be the mean prediction function of the surrogate model and $MSE(\widehat{\mu}(\boldsymbol{x})) = \mathbb{E}[(\widehat{\mu}(\boldsymbol{x}) - \Psi(\boldsymbol{x}))^2]$ its mean squared error (MSE). The statistical lower bound (SLB) of $\boldsymbol{x}$ is

$$SLB(\boldsymbol{x}) = -\widehat{\mu}(\boldsymbol{x}) + \upsilon \cdot \sqrt{\widehat{\mathrm{Var}}(\boldsymbol{x})} \cdot \sqrt{MSE(\widehat{\mu}(\boldsymbol{x}))}.$$

The rationale behind the SLB is to explore regions of the parameter space, where the SM's predictions have low goodness, be it systematic (the bias) or through variation (variance). It is only logical to use the classical goodness criterion of MSE.[11] Nowadays rather unconventionally, [Cox and John, 1992] normalize the MSE as follows: $MSE(\widehat{\mu}(\boldsymbol{x})) = \mathbb{E}[(\widehat{\mu}(\boldsymbol{x}) - \Psi(\boldsymbol{x}))^2] \cdot \sigma^{-2}$, with σ^{-2} being the inverse variance of the target Ψ that they estimate through maximum likelihood (ML). Presumably, the idea behind this "normalization" (i.e. standardization) is to make the (estimated) MSE comparable to other MSEs from different data sets. For instance, take estimators of a species' mean length for two data sets, the one comprising tadpoles, the other one blue whales. To compare the two MSEs, it is desirable to standardize them since the lengths of the two species differ by orders of magnitude. In the context of BO with a single target function, however, standardization is not needed.[12] We thus proceed with the classical, unnormalized $MSE(\widehat{\mu}(\boldsymbol{x})) = \mathbb{E}[(\widehat{\mu}(\boldsymbol{x}) - \Psi(\boldsymbol{x}))^2]$.

Notably, the lower confidence bound (LCB) can be regarded a special case of statistical lower bound (SLB). This is elucidated by Theorem 1.

Theorem 1 (Relationship of LCB and SLB)
Lower confidence bound equals statistical confidence bound for unbiased $\widehat{\mu}(\boldsymbol{x})$ and $\tau = \frac{\upsilon}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}$.

Proof 1 (Relationship of LCB and SLB)
Recall that we defined $\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))$ as estimator for the variance function, i.e. $\widehat{\mathrm{Var}}(\boldsymbol{x}) = \mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))$ It then follows right from definitions 3 and 6:

[11] Furthermore, MSE corresponds to the decision-theoretic risk function with the popular quadratic loss function.

[12] Multi-objective BO is only briefly touched upon in this thesis, see section 2.5.

$$
\begin{aligned}
SLB(\boldsymbol{x}) &= -\widehat{\mu}(\boldsymbol{x}) + \upsilon \cdot \sqrt{\widehat{\mathrm{Var}}(\boldsymbol{x})} \cdot \sqrt{MSE(\widehat{\mu}(\boldsymbol{x}))} \\
&= -\widehat{\mu}(\boldsymbol{x}) + \upsilon \cdot \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))} \cdot \sqrt{MSE(\widehat{\mu}(\boldsymbol{x}))} \\
&= -\widehat{\mu}(\boldsymbol{x}) + \upsilon \cdot \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))} \cdot \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x})) + \mathbb{E}[\widehat{\mu}(\boldsymbol{x}) - \Psi(\boldsymbol{x}))]^2} \\
&= -\widehat{\mu}(\boldsymbol{x}) + \upsilon \cdot \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))} \cdot \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}, \ \textit{unbiased } \widehat{\mu}(\boldsymbol{x}) \\
&= -\widehat{\mu}(\boldsymbol{x}) + \upsilon \cdot \mathrm{Var}(\widehat{\mu}(\boldsymbol{x})) \\
&= -\widehat{\mu}(\boldsymbol{x}) + \tau \cdot \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))} \iff \tau = \frac{\upsilon}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}, \qquad \square
\end{aligned}
$$

where the third equality uses the classical bias-variance-decomposition of the MSE:

Theorem 2 (Bias-Variance-Decomposition of MSE)
For the mean squared error (MSE) of an estimator $\widehat{\mu}(\boldsymbol{x})$ for $\Psi(\boldsymbol{x})$ it holds:

$$MSE(\widehat{\mu}(\boldsymbol{x})) = \mathbb{E}[(\widehat{\mu}(\boldsymbol{x}) - \Psi(\boldsymbol{x}))^2] = \mathrm{Var}(\widehat{\mu}(\boldsymbol{x})) + \mathbb{E}[\widehat{\mu}(\boldsymbol{x}) - \Psi(\boldsymbol{x}))]^2,$$

where $\mathbb{E}[\widehat{\mu}(\boldsymbol{x}) - \Psi(\boldsymbol{x}))]$ is commonly refered to as $\mathrm{Bias}(\widehat{\mu}(\boldsymbol{x}))$.

For the sake of completeness, a proof of Theorem 2 in the general case is provided in section 2 of the appendix.

While the SLB is hardly used anymore, LCB has become one of the most popular acquisition functions. As follows from theorem 1, using LCB corresponds to implicitly assuming unbiased mean estimation. In section 5.1, we will see that this assumption does not hold for a popular prior specification. We will address this issue in detail in section 9, where an alternative prior specification approach is proposed. Beyond this issue of implicitly assuming an unbiased mean prediction, we will generalize LCB (thus also SLB) to account for model imprecision in section 6.

2.3 Acquisition Functions for Noisy Targets

In practice, target functions rarely behave like they do in analytical imagination. Evaluations of physical processes are often noisy. That is, the function has a stochastic component. Multiple evaluations of one and the same point in $\mathcal{X}$ generally result in different function values. The optimization problem changes from $\min_{\boldsymbol{x}\in\mathcal{X}} \Psi(\boldsymbol{x})$ to $\min_{\boldsymbol{x}\in\mathcal{X}} \Psi(\boldsymbol{x}) + \epsilon$, with $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ for instance. There are multiple ways of how to adapt acquisition functions to such scenarios. [Huang et al., 2006] propose the augmented expected improvement (AEI).

Definition 7 (Augmented Expected Improvement (AEI))
Assume a Gaussian process as surrogate model and a target function $\Psi(\boldsymbol{x}) + \epsilon$ *with* $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ *. Leaning on definition 2 of expected improvement (EI) and its simplification in case of Gaussian process, the augmented expected improvement (AEI) is defined as*

$$\begin{aligned}\mathrm{AEI}(\boldsymbol{x}) = (\Psi^*_{\min} - \hat{\mu}(\boldsymbol{x}))\,\Phi\left(\frac{\Psi^*_{\min} - \hat{\mu}(\boldsymbol{x}))}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right) + \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))} \times \\ \phi\left(\frac{\Psi^*_{\min} - \hat{\mu}(\boldsymbol{x})}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right) \cdot \left(1 - \frac{\sigma_\epsilon^2}{\sqrt{\sigma_\epsilon^2 + \mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right),\end{aligned}$$

where $\Psi^*_{\min}$ *is the so-called effective best solution:*

$$\Psi^*_{\min} := \min_{\boldsymbol{x} \in \mathcal{X}} \left\{\hat{\mu}(\boldsymbol{x}) - \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}\right\}.$$

The "effective best solution" makes the AF compute the expected improvement in reference to the incumbent best possible value instead of the actual evaluated incumbent best value. The alert reader might notice that in case of a noise-free target function, AEI corresponds to EI with effective best solution $\Psi^*_{\min}$ instead of $\Psi_{\min}$. This special case of AEI will be used as a benchmark AF in section 6.6.

Similar to AEI, the expected quantile improvement (EQI) [Picheny et al., 2010] does not use an actual observed Ψ value as reference point but a pre-specified quantile of the posterior distribution of $\hat{\mu}(\boldsymbol{x})$.

Definition 8 (Expected Quantile Improvement (EQI))

$$\mathrm{EQI}(\boldsymbol{x}) = (q_{\min} - q(\boldsymbol{x}))\,\Phi\left(\frac{q_{\min} - q(\boldsymbol{x})}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right) + \sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))} \cdot \phi\left(\frac{q_{\min} - q(\boldsymbol{x})}{\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}}\right),$$

where q_{min} *is the minimal quantile of the posterior GP and* $q(\boldsymbol{x})$ *is the quantile of* $\boldsymbol{x}$*.*

Just like AEI, the EQI can be used for noise-free target functions as well, and will be in section 6.6.

2.4 Convergence and Limitations

It can be proven that Bayesian optimization with EI converges to the global optimum given that the prior specification of the SM is held constant across iterations [Vazquez

and Bect, 2010] [Bull, 2011]. The latter is usually not the case in practice, see section 5, rendering the result a rather theoretical consideration. This property cannot be proven for LCB and PI. Yet, especially lower confidence bound was found a good alternative to EI in some practical problems, see [Sun et al., 2021] for instance. Its popularity might stem, among other things, from its simple interpretation, as it controls the trade-off between exploration and exploitation directly through τ. On the other hand, τ can reduce optimization performance if its choice is not suited to the problem. Determining a good configuration can be computationally expensive [Berk et al., 2018].

Proposing points is a decision problem. From such a decision-theoretic point of view, proposals under all aforementioned acquisition functions are Bayes-optimal. They result from assigning probability weights on the state space Ω (parameter space $\mathcal{X}^p$) *a priori*. This is nowadays done via the SM, which takes data into account and is thus not a real Bayesian prior.[13] [Močkus, 1975] used an actual prior and presumably thus coined the term *Bayesian* optimization. Decisions are eventually based on maximizing a utility function (the acquisition function (AF)). However, their Bayes-optimality is limited to one iteration. Future information is not taken into account. This is why AFs like expected improvement and lower confidence bound are sometimes referred to as myopic, e.g. in [Wu and Frazier, 2019]. Intuitively, it could be beneficial to evaluate in regions of the parameter space that are not expected to dominate the incumbent best point, but could change the model's predictions in a manner that helps speeding up convergence in the long run. Keep in mind that by aiming at variance reduction EI, PI and LCB *do* explore unknown regions. Yet, the rationale behind these explorations is purely based on the incumbent mean and standard error predictions. Finding a non-myopic Bayes optimal set of proposals is however NP-hard because the number of possible proposal paths grows exponentially with the iterations. This myopia is the main limitation of BO besides the prior specification problem, which will be dealt with in Section 5.

2.5 Extensions

Still, it seems desirable to take future evaluations into account. There are some AFs that aim at doing so. Knowledge gradient [Frazier et al., 2009] and entropy search [Hennig and Schuler, 2012] are both based on a "one step look ahead"-approach that

[13] This is why we consider model-based optimization (MBO) a more adequate name. However, as explained above, Bayesian optimization (BO) is the more common term (and somehow honors the pioneering work of [Močkus, 1975]). We will thus stick with the literature and call the method BO.

accounts for hypothetical changes of the SM in iteration $t+1$ as a function of the proposal in iteration t. A more recent approach called rollout [Yue and Kontar, 2020] approximates the intractable optimization (of an AF comprising multiple iterations) problem's solution directly. The main downside of non-myopic BO is the computational cost that comes with AFs that "look ahead". That is, the charming idea of replacing expensive evaluations by cheap ones of a SM is given up to some extent. The dilemma between cheap evaluations and more farsighted algorithms is generally unresolvable. To put it plainly and simply, you can't have the cake and eat it.

To improve speed, several BO libraries allow for exploiting multicore infrastructures by proposing multiple points per iteration in parallel. The `R` package `mlrMBO` [Bischl et al., 2017], for instance, provides three distinct multi-point methods.[14] Although they proceed in different ways with regard to the acquisitions function(s), the general idea is simply to evaluate multiple points at once. We will exploit multicore infrastructure in prior-mean-robust BO in section 6.4 and 6.3, too.

A more deliberate extension is multi-objective[15] optimization, where more than one objective or criterion is optimized. See [Horn and Bischl, 2016] for a well-structured overview. Typical applications comprise trade-offs such as between offspring quantity and quality in evolutionary biology or between price and longevity of consumer products or – more relevant to hyperparameter-tuning – between performance and interpretability/sparsity of a machine learning model. For k objectives $\Psi(\boldsymbol{x}) = (\Psi(\boldsymbol{x})_1, ..., \Psi(\boldsymbol{x})_k)$ the optimization problem is $\min_{\boldsymbol{x}\in\mathcal{X}} \Psi(\boldsymbol{x})$, where $\Psi : \mathcal{X}^p \to \mathbb{R}^k$. Optima of such problems are characterized by the Pareto optimality, named after Vilfredo Pareto [Lopreato, 1973]. That is, optimal parameters are in the Pareto set and optimal targets constitute the Pareto front.

Definition 9 (Pareto Set and Pareto Front)
Let $\boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathcal{X}$. The Pareto set is $P = \{\boldsymbol{x} | \nexists \tilde{\boldsymbol{x}}$ with $\tilde{\boldsymbol{x}} \preceq \boldsymbol{x}\}$, where "$\preceq$" is the Pareto domination: $a \preceq b \Leftrightarrow \forall i \in \{1, ..., k\} : \Psi(a)_i \leq \Psi(b)_i$ and $\exists i \in \{1, ..., k\} : \Psi(a)_i < \Psi(b)_i$. The Pareto front is $\Psi(P)$, i.e. the image of P under Ψ.

A popular way to tackle such problems with Bayesian optimization is to avoid the multi-objective nature of the task and scalarize $\Psi : \mathcal{X} \to \mathbb{R}^k$ to $\Psi_s : \mathcal{X} \to \mathbb{R}$ by some weighting procedure. The generic way, however, is to approximate the Pareto front by proposing points according to k separately estimated surrogate models for data sets $(\boldsymbol{x}, \Psi(\boldsymbol{x})_1), ..., (\boldsymbol{x}, \Psi(\boldsymbol{x})_k)$. One then proceeds by either defining an acquisition function

[14] See Parallelization Tutorial for `mlrMBO`.

[15] Sometimes also referred to as multi-criteria optimization.

a priori that scalarizes all surrogate models' mean and standard error predictions or by computing acquisition functions separately for all surrogate models and then scalarizing k acquisition function values by the so-called hypervolume contribution (HC). The hypervolume of points $\{a, b, c, d, e\} \in \mathcal{X}$ is defined as the set of points (pareto-)dominated (see definition 9) by $\{a, b, c, d, e\}$ and bounded above by some reference point r, see figure 1. The hypervolume contribution of a proposed point p then simply is the increase in the hypervolume by adding p to $\{a, b, c, d, e\}$.

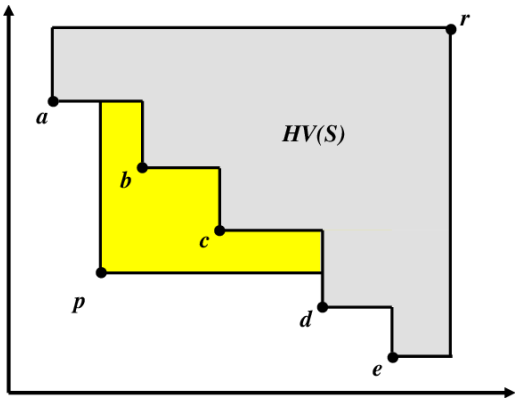

Figure 1: Hypervolume contribution (yellow) of point p, given a set $S = \{a, b, c, d, e\}$ of points, a reference point r and the corresponding hypervolume HV(S) (grey). Image credits: [Chen et al., 2020].

2.6 Applications and History

BO is a state-of-the-art method for optimizing black-box-functions. Its recent popularity stems from machine learning, where it has become one of the predominant hyperparameter optimizers [Nguyen, 2019] after the seminal work of [Snoek et al., 2012]. Even the popular reinforcement learning system AlphaGo from Google DeepMind, that beat the first human in Go, had its hyperparameters tuned with BO [Chen et al., 2018]. Thanks to its simplicity and generality, BO has proven to be useful in various fields, such as cognitive science [Shi et al., 2013], climate modeling [Abbas et al., 2014], drug discovery [Pyzer-Knapp, 2018] or even more recently in COVID-19 detection [Awal et al., 2021]. Another popular application is material science [Kotthoff, 2019] and engineering [Frazier and Wang, 2016], where the algorithm was first applied on a broader scale and enhanced by GPs as SMs [Jones et al., 1998].

It is important to distinguish between simulation-based and human-in-the-loop applications of BO. While the first deals with objective functions that can be evaluated as part of the computational procedure, the latter requires interaction with the physical world, e.g. a person setting up an experiment. In practice, however, the two domains are often

more intertwined than one might think. For instance, it is common practice in various fields to approximate the expensive true function by a (hyper-)surrogate model and then treat this model as ground truth in BO, see [Hutter, 2009] for instance. Moreover, there is a whole research area called multi-fidelity optimization that aims at combining high-fidelity data (high quality but expensive) and low-fidelity data (often simulated, low quality but cheap). See [Fernández-Godino et al., 2016] for a review of popular multi-fidelity models and methods. Section 8.2 will discuss a possible enhancement of low-fidelity optimization based on the concepts proposed in section 6.

The general idea to use decision-theoretical Bayes-criteria for optimizing unknown functions dates back to the 1970s, when Jonas Močkus published a series of papers on "Bayesian Methods For Seeking the Extremum" in Vilnius, Lithuania (at that time Soviet Union). See [Močkus, 1975] for one of the enthralling initial works including handwritten formulas or [Mockus et al., 1978] and [Močkus, 1989] for an overview. Močkus in turn borrowed from some considerations about optimizing unknown functions by [Kushner, 1962]. In an overview paper on BO, [Shahriari et al., 2015] even argue the roots of the Bayesian rationale in optimization date back even further to the 1930s. They proclaim that [Thompson, 1933] has already discovered the explore-exploit trade-off when "referring to the tension between selecting the best known treatment for every future patient (the greedy strategy) and continuing the clinical trial for longer in order to more confidently assess the quality of [...] treatments" [Shahriari et al., 2015] in the context of clinical study designs. It remains questionable, however, how strongly the explore-exploit trade-off is historically linked to BO, taking into account that it is a part of various derivative-free optimizers' rationales.

3 Gaussian Processes

As stated above, Gaussian process (GP) regressions[16] are the most common surrogate models in Bayesian optimization for continuous parameters. What is more, GP can also be used for categorical variables requiring some transformations [Garrido-Merchán and Hernández-Lobato, 2020]. For a comprehensive introduction to Gaussian processes, the reader is referred to [Rasmussen, 2003]. The main idea of functional regression based on GPs is to specify a Gaussian process *a priori* (a GP prior distribution), then observe data and eventually receive a posterior distribution over functions, from which inference is drawn, usually by mean and variance prediction. In most general terms, a GP is a

[16] Also called kriging.

stochastic process, i.e. a set of random variables, any finite collection of which has a joint normal distribution. Definition 10 specifies this as GP regression.

Definition 10 (Gaussian Process Regression)
A function $f(\boldsymbol{x})$ is generated by a Gaussian process $\mathcal{GP}\left(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')\right)$ if for any finite set of data points $\{x_1, ..., x_n\}$, the associated vector of function values $\boldsymbol{f} = (f(x_1), ..., f(x_n))$ has a multivariate Gaussian distribution: $\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Hence, GPs are fully specified by a mean function $m(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$ and a kernel[17] $k_\theta(\boldsymbol{x}, \boldsymbol{x'}) = \mathbb{E}\left[\left(f(\boldsymbol{x}) - \mathbb{E}[f(\boldsymbol{x})]\right)\left(f(\boldsymbol{x'}) - \mathbb{E}[f(\boldsymbol{x'})]\right)\right]$ such that $f(\boldsymbol{x}) \sim \mathcal{GP}\left(m(\boldsymbol{x}), k_\theta(\boldsymbol{x}, \boldsymbol{x'})\right)$ [Rasmussen, 2003, page 13]. The mean function gives the trend of the functions drawn from the GP and can be regarded as the best (constant, linear, quadratic, cubic etc.) approximation of the GP-functions. The kernel, broadly speaking, determines how the function behaves, e.g. its smoothness and periodicity. Any polynomial function is a mean function. Any finitely positive semi-definite function (definition 11) is a kernel function of a GP evaluated on a (finite) input vector.

Definition 11 (Finitely Positive Semi-Definite functions)
A function $f : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is finitely positive semi-definite if it is symmetric ($\forall \boldsymbol{x}, \boldsymbol{z} \in \mathcal{X} : f(\boldsymbol{x}, \boldsymbol{z}) = f(\boldsymbol{z}, \boldsymbol{x})$) and the matrix $\boldsymbol{K}$ formed by applying f to any finite subset of $\mathcal{X}$ is positive semi-definite, i.e. for its quadratic form it holds $\boldsymbol{x}^T \boldsymbol{K} \boldsymbol{x} \geq 0 \ \forall \boldsymbol{x} \in \mathcal{X}$.

A kernel is said to be isotropic if it is a function of the distance $\|\boldsymbol{x} - \boldsymbol{x}'\|$, conditioned on some norm, mostly the L2-Norm. Popular isotropic kernel families are linear kernels

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma_b^2 + \sigma^2(\boldsymbol{x} - c)(\boldsymbol{x}^T - c), \tag{4}$$

polynomial kernels

$$k(\boldsymbol{x}, \boldsymbol{x}') = \left(\sigma_b^2 + \sigma^2(\boldsymbol{x} - c)(\boldsymbol{x}^T - c)\right)^p, \tag{5}$$

Gaussian kernels

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\ell^2}\right), \tag{6}$$

exponential kernels

[17] Also called covariance function or kernel function.

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \cdot \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|}{2\ell}\right), \tag{7}$$

power-exponential kernels

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \cdot \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|}{2\ell}\right)^p \tag{8}$$

and Matérn-kernels

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \left(1 + \frac{\sqrt{\nu} * \|\boldsymbol{x} - \boldsymbol{x}'\|}{\ell} + \frac{\nu}{3}\left(\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|}{\ell}\right)^\rho\right) \exp\left(-\sqrt{\nu} \cdot \frac{\|\boldsymbol{x} - \boldsymbol{x}'\|}{\ell}\right), \nu, \rho \in \mathbb{R}. \tag{9}$$

In all kernels, σ^2 is the variance that can be viewed as the average distance away from the mean. In kernels with offset c, the base variance σ_b^2 additionally determines the uncertainty around c. Parameter ℓ determines the smoothness of the GP. For isotropic kernels, there even exists an exact mapping from ℓ to the expected number of upcrossings at level 0 in the unit interval (with $m(\boldsymbol{x}) = 0$, of course). Sometimes the effect of the kernel on the GP is reduced to this smoothness parameter ℓ.[18] However, as any finitely positive semi-definite function is a kernel, it can include various other parameters and represent all possible covariance structures.

Conclusively, both mean and kernel function consist of a functional form and parameters, both of which has to be specified beforehand. The effect of these four components on the BO will be assessed in section 4.

It is important to note that Gaussian process inference given the data is in principle a deterministic procedure. With a fully specified prior, the posterior distribution is a function of the prior and the data, see definition 12. However, in practice, prior parameters are often estimated from data, see section 5. For the sake of computational feasibility, sampling-based optimizers are used for this estimation. The latter makes GP a stochastic model, just like random forest (RF), the other popular surrogate model (SM) in BO. As stated above, BO is thus a stochastic optimizer that proposes different points (in general) even with the exact same initial design, surrogate model and acquisition function.[19]

[18] Also called kernel-bandwidth or length-scale parameter.

[19] Note that BO's stochastic nature also stems from the infill optimization (random search, focus search) to some extent.

Definition 12 (Gaussian Process Predictive Posterior)
Assume $m(\boldsymbol{x}) = 0$ for simplicity. Given a Gaussian process prior $\mathcal{GP}\left(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x'})\right)$, a design matrix of training data $\boldsymbol{X}$ and respective target values $\boldsymbol{f}$ as well as a design matrix $\boldsymbol{X^*}$ of test data, the posterior predictive distribution of $\boldsymbol{f^*}$ on $\boldsymbol{X^*}$ given the already evaluated training values is

$$\boldsymbol{f^*}|\boldsymbol{X^*}, \boldsymbol{X}, \boldsymbol{f} \sim \mathcal{N}\left(\boldsymbol{\Sigma^*}\boldsymbol{\Sigma}^{-1}\boldsymbol{f}, \boldsymbol{\Sigma^{**}} - (\boldsymbol{\Sigma^*})^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma^*}\right),$$

where $\boldsymbol{\Sigma^*}$ is the kernel matrix for the covariances within the training data, $\boldsymbol{\Sigma^*}$ is the kernel matrix for the covariances between training and test data and $\boldsymbol{\Sigma^{**}}$ the kernel matrix within the test data [Rasmussen, 2003].

The statement in definition 12 follows from the fact that the conditional distribution of a random variable on another random variable, both of which follow a multivariate normal, is again multivariate normal. It can be nicely seen from the mean and kernel function of the posterior predictive distribution that GP (without assuming noisy evaluations) interpolates between training points:

$$\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{f} \sim \mathcal{N}\left(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^{-1}\boldsymbol{f}, \boldsymbol{\Sigma} - (\boldsymbol{\Sigma})^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}\right) = \mathcal{N}\left(\boldsymbol{f}, \boldsymbol{0}\right), \tag{10}$$

since for predictions on training points $\boldsymbol{\Sigma^{**}} = \boldsymbol{\Sigma^*} = \boldsymbol{\Sigma}$.

It should be added that this posterior predictive process is exactly the GP surrogate model that we denoted as $\psi(\boldsymbol{x}) \sim \mathcal{N}\left(\widehat{\mu}(\boldsymbol{x}), \widehat{\mathrm{Var}}(\boldsymbol{x})\right)$ in section 2.2. Thus, $\widehat{\mu}(\boldsymbol{x}) = \boldsymbol{\Sigma}\boldsymbol{\Sigma}^{-1}\boldsymbol{f}$ and $\widehat{\mathrm{Var}}(\boldsymbol{x}) = \boldsymbol{\Sigma} - (\boldsymbol{\Sigma})^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}$.

4 Bayesian Sensitivity Analysis

The question arises quite naturally how sensitive Bayesian optimization is towards the prior specification of the GP. It is a well-known fact that classical inference from Gaussian processes is sensitive with regard to prior specification in case of small n. The less data, the more the inference relies on the prior information. For Bayesian methods this can be shown in general by expressing posterior statistics as weighted sums of likelihood and prior contributions [Rüger, 2010] as well as in the concrete case of GP [Handcock and Stein, 1993]. What is more, there exist detailed empirical studies such as [Schmidt et al., 2008] that analyze the impact of prior mean function and kernel on the posterior GP given various data sets.

We investigate to what extent this translates to BO's returned optima and convergence rates. Analyzing the effect on optima and convergence rates is closely related yet different. Both viewpoints have weaknesses: Focusing on the returned optima means conditioning the analysis on the termination criterion; considering convergence rates requires the optimizer to converge in computationally feasible time. Considering real world applications, it gets even worse: Typically, global optima are and will remain unknown, rendering convergence criteria unreliable. To avoid (at least the former two of) these downsides, we analyze the mean optimization path (MOP).

Definition 13 (Mean Optimization Path)
Given R repetitions of Bayesian optimization applied on a test function $\Psi(\boldsymbol{x})$ with T iterations each, the best incumbent target value at Iteration $t \in \{1, ..., T\}$ from repetition $r \in \{1, ..., R\}$ is $\Psi(\boldsymbol{x}^*)_{r,t}$. The mean optimization path (MOP) then is a T-dimensional vector MOP with elements

$$MOP_t = \frac{1}{R} \sum_{r=1}^{R} \Psi(\boldsymbol{x}^*)_{r,t}.$$

MOP provides a meaningful analysis of optimization performance regardless of convergence towards the optimal ground truth. Averaging is required since BO is a stochastic optimizer. Hereinafter, MOP values of different prior specifications will be compared. We condition our analysis on an initial design of size $n = 10$ due to computational reasons throughout section 4.1, i.e. it is fixed for all R repetitions. Limitations of this approach are discussed in section 4.4.

As pointed out in section 3, specifying a GP prior comes down to choosing a mean function and a kernel. Both kernel and mean function are in turn determined by a functional form (e.g. linear trend and Gaussian kernel) and its parameters (e.g. intercept, slope, σ^2 and ℓ). Hence, we vary the GP prior with regard to the mean functional form $m(\cdot)$ (section 4.1.1), the mean function parameters (section 4.1.2), the kernel functional form $k(\cdot, \cdot)$ (section 4.1.3) and the kernel parameters (section 4.1.4). We run the analysis on well-established synthetic test functions from the `R` package `smoof` [Bossek, 2017]. First, some exploratory graphical results are presented before turning to more systematic results (section 4.2). Throughout the chapter, the `R` package `mlrMBO` that builds on `DiceKriging` is used for BO and GP computations. All experiments were computed on a high performance computing cluster using 20 64-bit-cores (linux gnu). The experiments were conducted in `R` version 4.0.3 [R Core Team, 2020].

4.1 Graphical Analysis

Unlike a systematic analysis, the following graphical assessment of MOPs is of exploratory nature. It does not aim at showing that prior specification affects Bayesian optimization *on average*, but that it *can* harm BO's performance in some cases. We will see in section 4.2 that the effect strongly depends on the objective function.

4.1.1 Mean Functional Form

The most common functional forms of the mean function are constant, linear, quadratic and cubic trends. For analyzing their effect on the MOP given a target function, we run $R = 40$ repetitions of Bayesian optimization per mean functional form with $T = 20$ iterations each. Since we are interested in the mean functional form's *ceteris paribus* effect, other hyperparameters of the optimization such as the GP's kernel function (Gaussian) and the AF (expected improvement) are not varied across repetitions. The mean function and kernel parameters are estimated via maximum likelihood (as described in section 5.1). For illustrative purposes, consider the multimodal and bivariate carrom table (figure 2) and Giunta function (figure 3) from the `smoof` package as test functions.

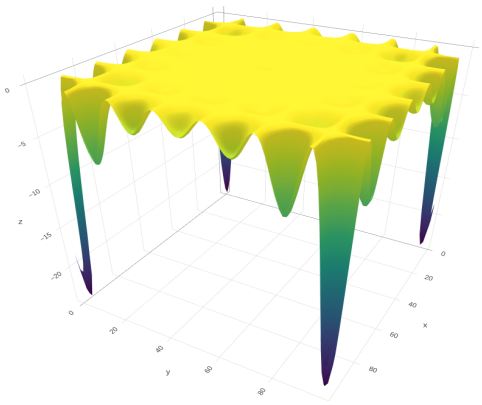

Figure 2: Carrom Table function

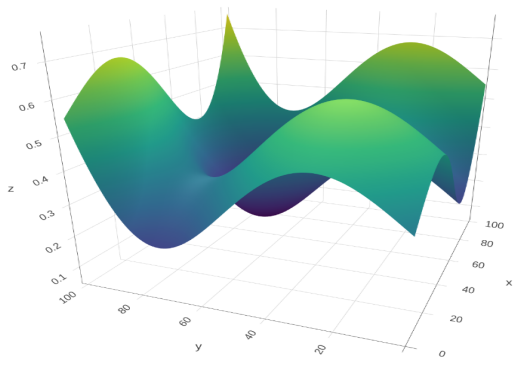

Figure 3: Giunta function

Figures 4 and 5 visualize the mean optimization paths (with $R = 40$ and $T = 20$) of BO applied on carrom table and Giunta function. In addition to the incumbent mean best target values, their 0.95-confidence-intervals are depicted as error bars. The dotted pink line represents the global optima.

Both carrom table and Giunta function have a globally good quadratic or cubic approximation, see figures 2 and 3. (Notably, this appears to be orthogonal to the functions'

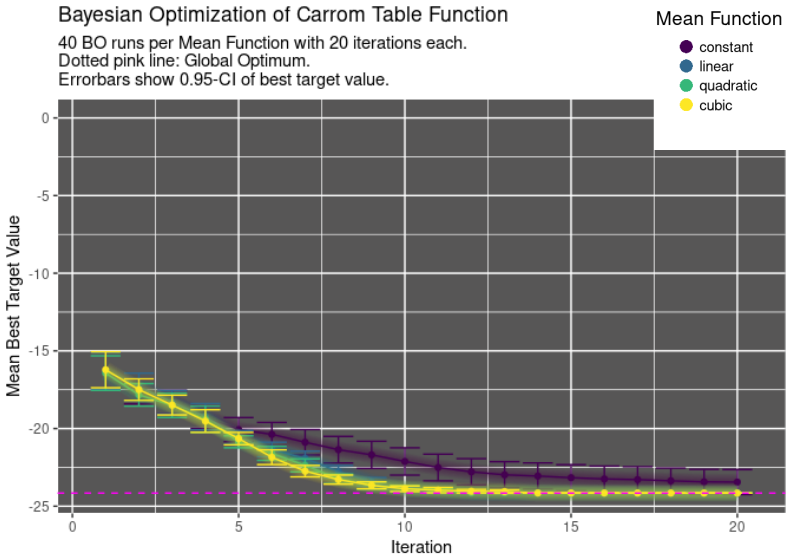

Figure 4: Effect of mean functional form on Bayesian optimization of carrom table function.

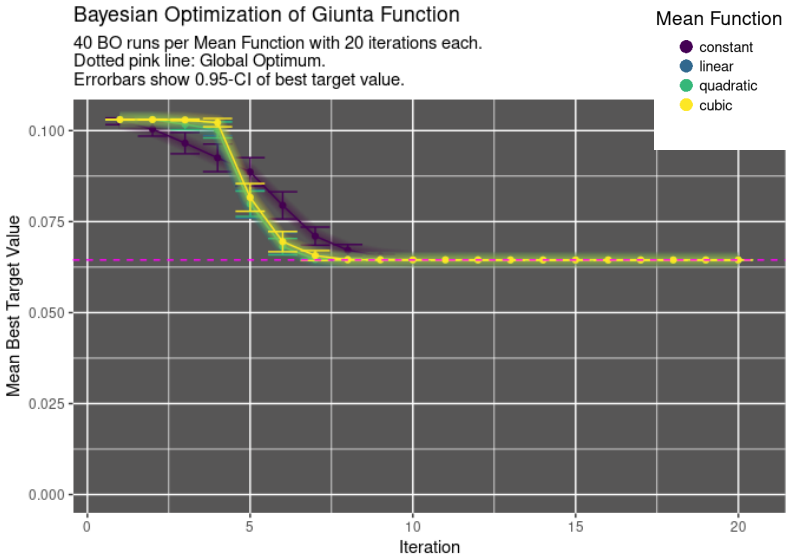

Figure 5: Effect of mean functional form on Bayesian optimization of Giunta function.

smoothness. Giunta function is much smoother than carrom table function.) It becomes evident that when dealing with such functions, trends of lower polynomial order such as constants or linear functions slow down BO's performance. Here, this can be seen by the MOPs with constant mean function. Intuitively speaking, that is because more explorations are needed for the SM to gain sufficient confidence if the prior appears not to be reliable. At a premature state of the optimization, both functions are still optimized reasonably well by the GP with constant mean. In fact, for the Giunta function's optimization, the constant mean is even slightly advantageous in iterations 1 to 4. In the long run, however, the weak approximation of the SM with constant mean

prior prevents the BO from exploiting promising regions in detail. For the carrom table function, this prior bias affects the incumbent best mean even in late iterations. The algorithm apparently has proposed points from the four promising regions, as can bee seen by comparing the vertical axis in figure 2 with the mean best target values in figure 4. However, the constant mean prior deprives the optimizer of the flexibility to dive further into these regions. The SM's mean prediction overestimates the function in these areas, which results in lower AF values, making them less "attractive". Yet, the overall effect of the mean functional form appears to be rather low in magnitude, see figures 5 and 4 again.

4.1.2 Mean Function Parameters

The graphical analysis of the mean functions parameters' effects on BO gives quite a different picture. As can be seen in figures 6 and 8, the choice of the constant mean value in the GP prior strongly influences the MOP values. This time, the functional form (constant trend) remains unchanged, while its parameter is altered in the following fashion: The mean of the function values are estimated from latin hypercube sampling (LHS) ($n = 400$ for low dimensions with adaption in higher dimensions.[20]) This mean estimate is then taken as baseline constant in the prior mean function (turquoise optimization path in figures 6 and 8). To allow cross-functional comparison of the results, these baseline means bm are then used to compute a vector of five different mean constants $(bm - 3bm, bm - 0.2bm, bm, bm + 0.2bm, bm + 3bm)^T$. Again, everything else is kept constant to allow for a *ceteris paribus* analysis.

While the effect on Bayesian optimization of carrom table function is negligible this time, the Giunta function optimization is more strongly affected than by the mean functional form, see figures 28 and 31 in the appendix. What is more, optimizations of other functions are severely slowed down by inadequate choices of the constant mean value, whereat "inadequate" does not necessarily mean "inaccurate", as the following graphical exploration will reveal.

The effect of the mean function parameter becomes especially evident in case of the six-dimensional Schwefel function from the black-box optimization benchmark (BBOB) database.

[20] We control for the curse of dimensionality (COD) by comparing mean estimates based on $n = 400$ with mean estimates based on $n = 400p$. Yet, in most cases estimates do not change severely.

$$f(\boldsymbol{x}) = -\frac{1}{6}\sum_{d=1}^{6}\left[x_d \sin\left(\sqrt{|x_d|}\right)\right] + c, \tag{11}$$

where c is a constant and $\boldsymbol{x}$ a 6-dimensional vector with entries x_d. Figure 6 shows that big deviations from the unbiased mean estimate $m(\boldsymbol{x}) = 70000$ speed up (prior mean too high) or slow down (prior mean too low) the optimization paths.

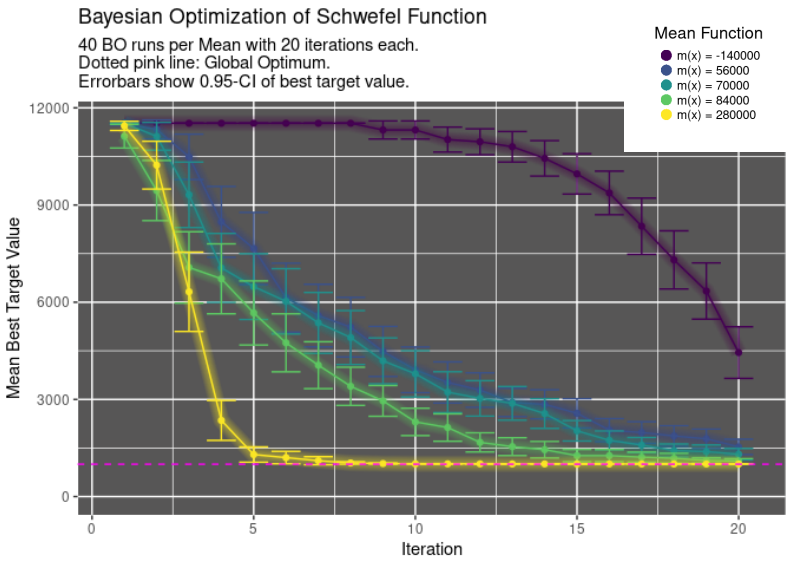

Figure 6: Effect of mean function parameters on Bayesian optimization of six-dimensional Schwefel function, see equation 11.

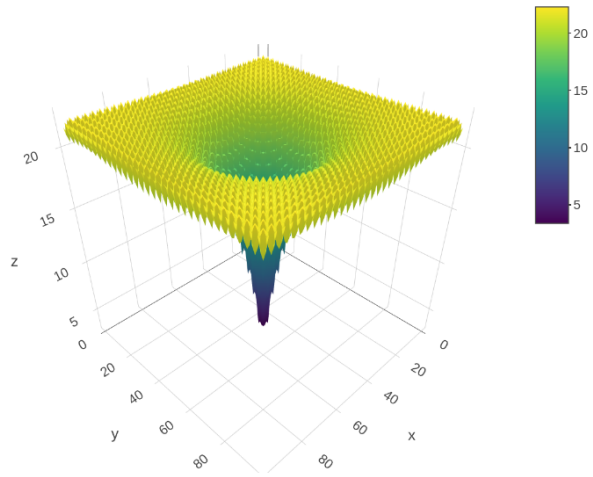

Figure 7: Bivariate Ackley function

A similar slowdown of the optimization caused by severe underestimation of the mean can be observed in case of the bivariate and highly-multimodal Ackley function (figure

7). Unlike in case of Schwefel function, however, the optimization of the bivariate Ackley function is not accelerated by a prior mean that is too high. A slightly higher mean seems beneficial, but an extraordinary high mean of 80 performs worse (figure 8). As of now, the exact nature of the prior mean's effect remains unclear. Its very existence, however, appears beyond any doubt. The systematic assessment in section 4.2 will confirm this.

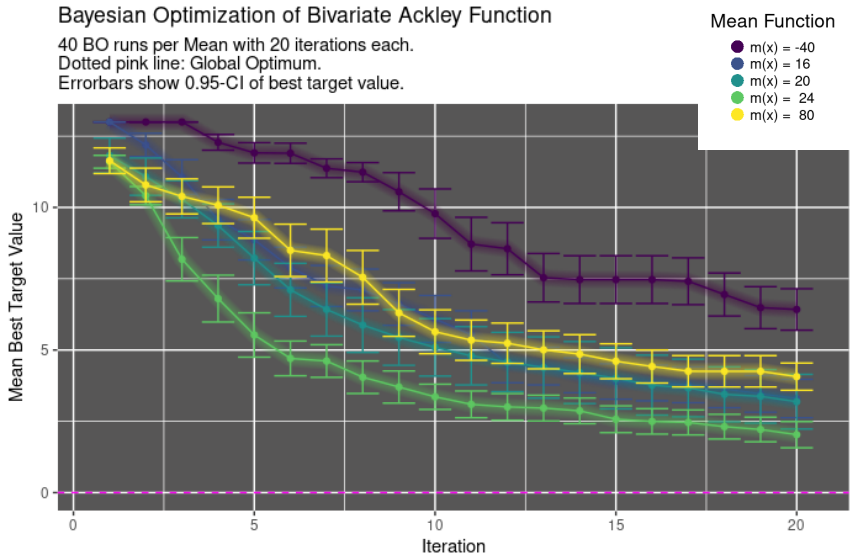

Figure 8: Effect of mean function parameters on Bayesian optimization of bivariate Ackley function.

4.1.3 Kernel Functional Form

We assess the influence of five popular kernel function families: Gaussian kernel, exponential kernel, power exponential kernel and two Matérn-kernel families ($\nu = 3, \rho = 2$ and $\nu = 5, \rho = 2$). These kernels appear throughout the literature and are available per default in `mlrMBO`. Mean function was set to $m(x) = 0$ and both the mean and kernel functions' parameters were estimated by maximum likelihood as described in section 5.1. Again, all other hyperparameters were kept constant. Figure 9 show MOP plots for the Engvall test function. Figures 32 and 33 in the appendix show MOP plots for two more test functions. As you can plainly see, the effect of the kernel functional form is generally high. Section 4.2 will confirm this. While the exponential kernel guarantees faster convergence in most of the functions we tested, such as the Bird function (figure 33) and the Engvall function (figure 9), there are also functions, for which the exponential kernel slowed down the optimization, see bivariate Ackley function in figure 32. This difference in performance caused by the kernel's functional form cannot be attributed to any specifics of the functions involved. For instance, it cannot be explained

by the functions' smoothness since Ackley and Bird function are both heavily wiggly.[21]

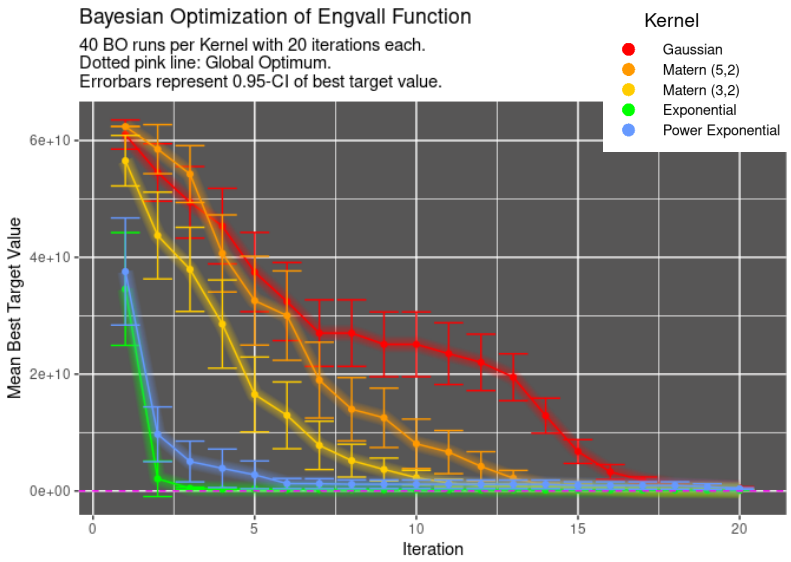

Figure 9: Effect of kernel functional form on Bayesian optimization of Engvall function. Color palette was altered to incorporate the non-ordinal nature of the attribute "kernel functional form", except for the two Matérn-kernels.

4.1.4 Kernel Function Parameters

For analyzing the kernel parameters' effects, we restrict ourselves to the kernel-bandwidth ℓ that all popular kernel functional forms share and that determines the smoothness of the prior GP. Leaning on the procedure for varying mean function parameters in section 4.1.2, we estimate an "optimal" baseline ℓ_{bm} via maximum likelihood from $n = 400$ data points obtained by LHS. This ℓ_{bm} is then used to specify a vector of five different kernel bandwidth values $(\ell_{bm} - 3\ell_{bm}, \ell_{bm} - 0.2\ell_{bm}, \ell_{bm}, \ell_{bm} + 0.2\ell_{bm}, \ell_{bm} + 3\ell_{bm})^T$. The mean function was set to $m(x) = 0$ and a Gaussian kernel was chosen as functional form. Given that the functional form has a high impact on MOPs, this appears as a rather drastic limitation. Thus, we run the same simulations for an exponential kernel and a Matérn kernel ($\nu = 5, \rho = 2$). The results do not differ, increasing our confidence in them. For each kernel functional form, a varying ℓ has almost no impact on the optimization path. The visualized MOPs all resemble figure 10.

[21] Compare the functions' visualizations in figure 7 (Ackley) as well as figures 29 and 30 (Bird and Engvall function) in the appendix.

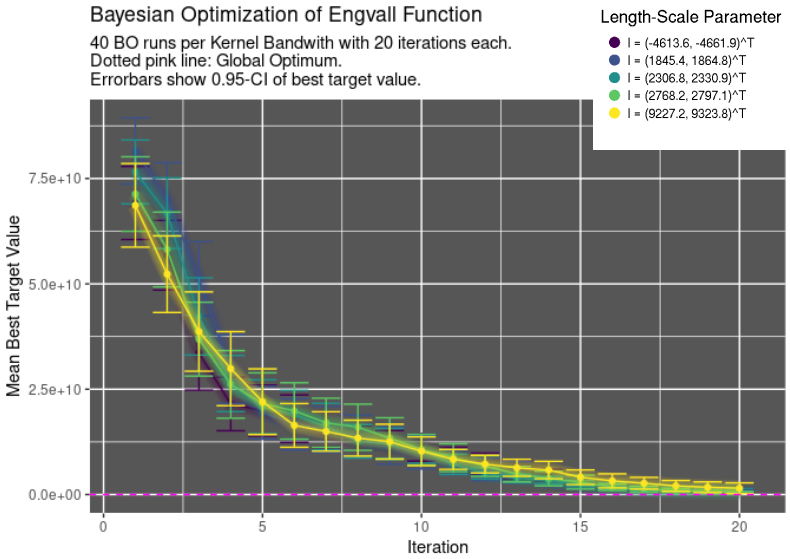

Figure 10: Effect of kernel parameter ℓ on Bayesian optimization of Engvall function.

4.2 Results

In order to test the results of the exploratory graphical analysis, we proceed in a more systematic way. 50 test functions from the `smoof` package are selected at random, stratified across the parameter space dimensions $1, 2, 3, 4$ and 7. For each of them, the sensitivity analysis is conducted with regard to mean functional form and parameters as well as with regard to kernel functional form and parameters, as described above in sections 4.1.1 - 4.1.4. Diverging from the graphical analysis, this systematic assessment is not conditioned on the initial design, i.e. it is randomly sampled anew for each of the $R = 40$ BO repetitions. This way, we make sure the results do not depend on a specific initial sample. Recall that we are interested in a systematic assessment now as opposed to the graphical analysis, where our aim was to show that an influence is possible in principle. Instead of visualizing the MOPs directly, we compute the accumulated difference (AD) for each experiment, following definition 14.

Definition 14 (Accumulated Difference of Mean Optimization Paths)
Consider an experiment comparing S different prior specifications on a test function with R repetitions per specification and T iterations per repetition. This results in a $T \times S$-matrix of MOPs as defined in definition 13 for iterations $t \in \{1, ..., T\}$ and prior specification $s \in \{1, ..., S\}$ (e.g. constant, linear, quadratic etc. trend as mean

functional form) with entries :

$$MOP_{t,s} = \frac{1}{R} \sum_{r=1}^{R} \Psi(\boldsymbol{x}^*)_{r,t,s}.$$

The accumulated difference (AD) for this experiment is:

$$AD = \sum_{t=1}^{T} \Big(\max_s MOP_{t,s} - \min_s MOP_{t,s} \Big)$$

Note that the accumulated difference (AD) is proportional to the area between the lowest and the highest graph in the visualizations of mean optimization paths, see figure 8 for instance. We use this measure to approximate the influence of prior variations with regard to the different components mentioned above.

The accumulated differences for all the 50 randomly selected test functions can be found in the appendix, table 5. The following table 1 shows the accumulated differences for some of the 50 functions. As can be seen, the AD values vary strongly across functions, compare AD values of the Bent-Cigar function with the ones of the Matyas or the Brent function, for instance. This can be explained by different levels of difficulty of the optimization problem, mainly influenced by modality ("How many local optima are there that need to be explored?") and smoothness ("How easy is it to interpolate between data points?"). The Brent function, for instance, is relatively easy to optimize due to its smoothness and uni-modality, see figure 11. Since BO converges quickly in this case regardless of the prior specification, its influence is rather negligible.

Since we are interested in an overall, systematic assessment of the prior's influence on Bayesian optimization, we sum the AD values over the stratified sample of 50 functions, see table 3. Glimpsing at table 1 again, however, reveals that this absolute sum is likely driven by some hard-to-optimize functions with generally higher AD values or by the scale of the functions' target values.[22] Thus, we standardize the AD values within each function, i.e. divide each AD value by the mean AD of the respective function. Results can be found in table 6 in the appendix. Table 3 shows the sums of the standardized AD values. It becomes evident that the optimization is affected the most by the functional form of the kernel and the mean parameters, while kernel parameters play a minor role. The mean functional form plays an important role when considering the absolute effect across all functions, while it is less important taking standardization into account.

[22]Note that AD and MOP as defined in definition 13 and 14 are not scale-invariant.

Test function (Dimension of $\mathcal{X}$)	mean functional form	mean parameters	kernel functional form	kernel parameters
Ackley (1)	23	38	67	23
Alpine N. 1 (1)	2.8	1.8	2	1.2
Alpine N. 2 (1)	0.11	0.15	0.16	0.079
Cosine Mixture (1)	0.073	0.07	0.11	0.14
Hyper-Ellipsoid (2)	0.24	2	3.7	0.0012
Six-Hump Camel Back (2)	1.3	3.3	2.9	0.71
Giunta (2)	0.16	0.29	0.1	0.00018
Carrom Table (2)	71	71	81	0.2
Brent function (2)	0.44	0.47	13	$4.4 \cdot 10^{-5}$
Trecanni (2)	1.5	3.4	7	0
Matyas (2)	0.28	0.59	2.7	0
Hartmann (3)	3	4.8	5.3	0.82
Alpine N. 2 (3)	14	25	30	4.6
Sum of Different Squares (4)	0.37	1.4	0.32	0
Bent-Cigar (4)	$3.6 \cdot 10^{9}$	$2 \cdot 10^{10}$	$7 \cdot 10^{9}$	$5.7 \cdot 10^{8}$
Deflected Corrugated Spring (7)	16	38	11	0
Sphere (7)	$1.5 \cdot 10^{2}$	$4.4 \cdot 10^{2}$	97	8.5

Table 1: Accumulated differences of MOP values for BO of selected test functions from `smoof`. Find complete results in table 5 and 6 in the appendix.

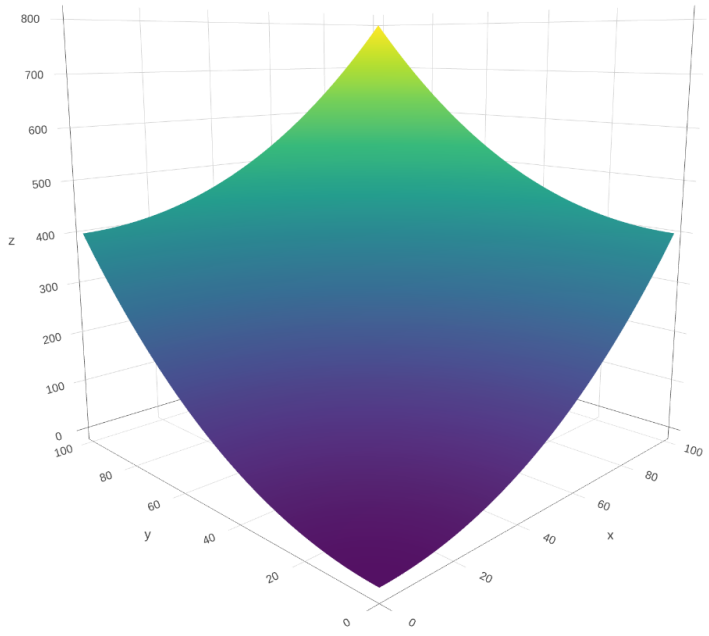

Figure 11: Bivariate Brent function

mean functional form	kernel functional form	mean parameters	kernel parameters
$1.8714 \cdot 10^{16}$	$1,0876 \cdot 10^{16}$	$1,0828 \cdot 10^{16}$	$3.5316 \cdot 10^{15}$

Table 2: Sum of accumulated differences of all 50 MOPs per prior specification. Comparisons between mean and kernel are more valid than between functional form and parameters.

mean functional form	kernel functional form	mean parameters	kernel parameters
42.49	68.20	77.91	11.40

Table 3: Sum of standardized accumulated differences of all 50 MOPs per prior specification. Comparisons between mean and kernel are more valid than between functional form and parameters.

4.3 Discussion

As expensive functions imply few data, it comes at no surprise that the GP's predictions in BO heavily depend on the prior. Our results suggest this translates to BO's convergence. It is more sensitive towards the functional form of the kernel than towards those of the mean function. It is more sensitive towards the mean function's parameters than towards those of the kernel, which appear to play a negligible role in BO's convergence.

The kernel functional form determines the general form and flexibility of the Gaussian process and thus has a strong effect on the SM's capacity to model the true functional relationship. Its importance in BO then comes at no surprise. What is more interesting, the mean parameters' effect may not only stem from the modeling capacity but also from the optimizational nature of the algorithm. While unintended in statistical modeling, a systematic under- or overestimation may be beneficial when facing an optimization problem, see also section 5.1 and 9. Further research on interpreting the effect of the GP prior's components on mean optimization paths is recommended.

4.4 Limitations

Albeit the random sample of 50 test functions was drawn from a wide range of established benchmark functions in the `smoof` package, the analysis does by far not comprise

all possible objective functions, not to mention real-world optimization problems. Additionally, as mentioned above, the graphical analysis is conditioned on an initial sample. This is not a severe limitation, since we use maximin latin hypercube sampling (LHS) that maximizes the minimum distance between design points and thereby restricts the influence of the initial sample. Anyhow, we did vary the initial design in the systematic assessment. What weighs more, the presented findings regarding kernel and mean function parameters are influenced by the degree of variation, the latter being a subjective choice. Statements comparing the influence of the functional form with the parameters are thus to be treated with caution. Yet, the comparison between kernel and mean function parameters is found valid, as both have been altered by the same factors. Similar considerations hold for comparing mean and kernel functional form.

Furthermore, interaction effects between the four prior components were not completely taken into account and partly left to further research. The reported AD values for mean parameters and mean functional form were computed using a Gaussian kernel. Since other kernels may interact differently with the mean function, the analysis was revisited using a power exponential kernel as well as a Matérn kernel. As we observe only small changes in AD values, the sensitivity analysis can be seen as relatively robust in this regard. Noteworthy, we did not control for possible interaction effects in the other direction, that is between mean functional form and parameters on the one hand with the kernel on the other hand, because the former can be (and are per default in `DiceKriging`) estimated from the data. (This, of course, is not possible in case of the kernel's functional form.) Hence, we measured the overall effect of kernel parameters and functional form conditioned on the best fit of the mean function.

5 Prior Specification Problem

In the following, we will discuss two popular approaches to specifying the prior parameters, subsumed as $\boldsymbol{\theta}$ in definition 15, of GP in BO. Within the scope of this thesis, we assume the functional form of both mean function and kernel is given. [Duvenaud, 2014] and [Duvenaud et al., 2013] address the intriguing issue of how to specify the functional form of the kernel in GP regression in general and [Malkomes and Garnett, 2018] for the case of GP inside BO, see section 7.

Definition 15 (Prior Parameter Vector)
Given functional forms $m(\cdot)$ and $k(\cdot,\cdot)$, a Gaussian process prior $\mathcal{GP}(m_\theta(\boldsymbol{x}), k_\theta(\boldsymbol{x},\boldsymbol{x}'))$ is fully specified by the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^p$, comprising parameters of both the mean function and the kernel, i.e.

$$\boldsymbol{\theta} = [\theta_{m(x),1}, ..., \theta_{m(x),p_m}, \theta_{k(x,x'),1}, ..., \theta_{k(x,x'),p_k}]^T,$$

where p_m is the number of mean function parameters, p_k is number of kernel parameters and $p_m + p_k = p$.

As BO deals with functions lacking analytical description, information about the function's smoothness, mean, periodicity and the like is scarce *a priori*. Hence, specifying the GP prior by expert knowledge is usually not an option. There are two popular approaches to this problem, maximum likelihood (ML) and integrated acquisition function (IAF). Before turning to our proposed method in section 6, we will briefly discuss these two commonly-received approaches and show their limitations. Figure 12 summarizes all case differentiations mentioned above.

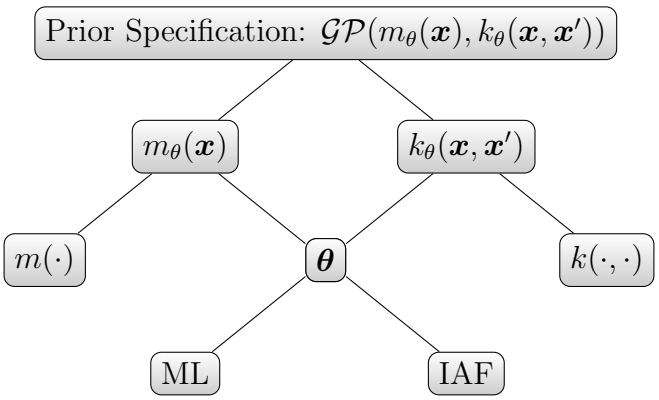

Figure 12: Summary of Section 5: The Prior Specification Problem.

5.1 Maximum Likelihood Estimation

One option is to estimate $\boldsymbol{\theta}$ from the data by maximum likelihood (ML): $\hat{\boldsymbol{\theta}}_{ML}(\boldsymbol{X}_t)$, where $\boldsymbol{X}_t$ is the design matrix of iteration t, see definition 16. Just like estimating the surrogate model violates the Bayesian paradigm of the optimization procedure, this approach violates the Bayesian paradigm of GP, i.e. specifying a prior *before* seeing the data. This "peeking at the data", however, is quite popular in practice and even the default method in many software libraries, e.g. in `mlrMBO`. Its popularity stems from classical GP Regression, see the `R` package `DiceKriging` [Roustant et al., 2012], where data is usually not sequentially augmented as in BO. Some implementations also use partial ML estimation, e.g. only for the kernel parameters, while setting mean function parameters in the actual Bayesian way *a priori*: $[\theta_{k(x,x'),1}, ..., \theta_{k(x,x'),p_k}]^T = [\widehat{\theta}_{k(x,x'),1,ML}, ..., \widehat{\theta}_{k(x,x'),p_k,ML}]^T$ and $[\theta_{m(x),1}, ..., \theta_{m(x),p_m}]^T = \boldsymbol{m}, \boldsymbol{m} \in \mathbb{R}^{p_m}$. For example $\boldsymbol{m} = 0$.

Definition 16 (Maximum Likelihood Estimation of Prior Parameters)
Consider iteration t. Given $n = n_{init} + t$ observations of covariates $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_{init}}, \boldsymbol{x}_{n_{init}+1}, ..., \boldsymbol{x}_{n_{init}+t}\}$ and the respective function values $\{\Psi(\boldsymbol{x}_1), ..., \Psi(\boldsymbol{x}_{n_{init}+1}), \Psi(\boldsymbol{x}_1), ..., \Psi(\boldsymbol{x}_{n_{init}+t})\}$, both subsumed as design matrix $\boldsymbol{X}_t$, a normal density function f and the resulting likelihood function $L(\boldsymbol{\theta}|\boldsymbol{X}_t) = \prod_{i=1}^{n} f(\boldsymbol{\theta}|\boldsymbol{x}_i, \Psi(\boldsymbol{x}_i))$, the maximum likelihood estimator for $\boldsymbol{\theta}$ is

$$\hat{\boldsymbol{\theta}}_{ML}(\boldsymbol{X}_t) = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}|\boldsymbol{X}_t).$$

Simply "plugging" ML-estimation into an optimization method like BO can result in a (globally) biased $\hat{\boldsymbol{\theta}}_{ML}(\boldsymbol{X})$. This is due to the optimizational nature of the process that proposes data points in order to find an optimal parameter configuration: $\boldsymbol{X}_t$ entails data points accumulated in certain regions that appear promising during optimization. However, for the ML-estimator $\hat{\boldsymbol{\theta}}_{ML}(\boldsymbol{X})$ to be unbiased, $\boldsymbol{X}_t$ is required to be drawn independently (*ind.*) from an identical distribution (*iden.*). This is implicitly assumed in definition 16 as follows:

$$\mathcal{L}(\boldsymbol{\theta}|\boldsymbol{X}) \overset{ind.}{=} \prod_{i=1}^{n} \ell(\boldsymbol{\theta}|\boldsymbol{x}_i, \Psi(\boldsymbol{x}_i)) \overset{iden.}{=} \prod_{i=1}^{n} f(\boldsymbol{\theta}|\boldsymbol{x}_i, \Psi(\boldsymbol{x}_i)), \tag{12}$$

where the assumption of identical distribution follows from the multivariate normal assumption of the Gaussian process (definition 10) and the fact that normal distributions are closed under marginalization.

The assumption of independence can be violated, as figure 13 illustrates. It shows the distribution of 10 proposed points from BO of a cosine mixture function with GP as surrogate model and EI as acquisition function. Albeit exploring the parameter space in three iterations, the BO has mainly focused on a promising local (and global) optimum. For illustrative purposes, now assume $p_m = 1$, a constant mean functional trend. Then θ_m for the GP in iteration t is estimated by $\hat{\theta}_m \in \hat{\boldsymbol{\theta}}_{ML}(\boldsymbol{X})$. As more and more points are proposed in the region of the global optima, $\hat{\theta}_m$ will be driven towards the global optimum around 0. It is in regions far away from the optimum, however, where $\hat{\theta}_m$ will then do some damage. In the SM in these less-explored regions the prior will dominate the likelihood in the posterior leading to a systematic underestimation of the true function. This, in turn, might make these regions more attractive as expressed by the AF as a function of the mean estimates, which by this mean might affect the optimization. Section 9 will address this hypothesis in detail.

Recall that EI is already rather exploratory compared to other acquisition functions. Only with fully exploratory AFs, consider SE from definition 5 for instance, the bias will disappear. Such an extreme form of BO, however, will hardly[23] serve the practitioner's aim of optimization given a limited budget of evaluations. At least at first sight, optimizing and understanding the unknown functional relationship through *iid* samples appear to be incompatible objectives. Yet, section 9 will demonstrate that proposals from optimization can be used to approximate an *iid* sample through a weighting procedure.

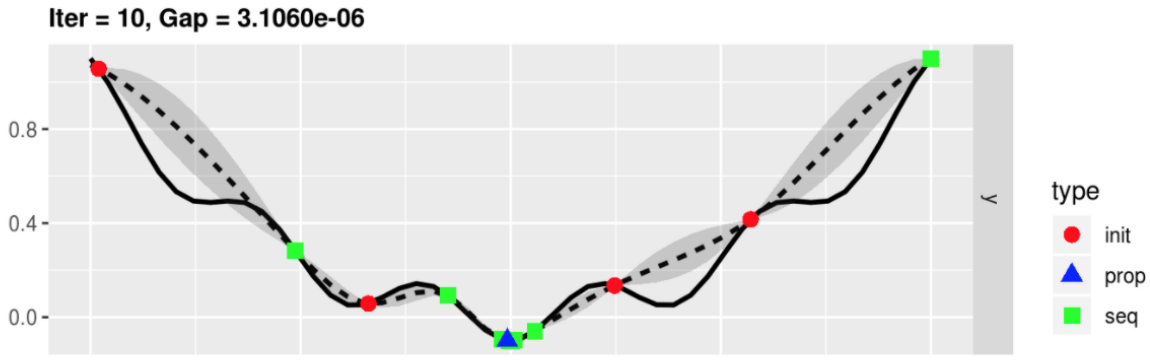

Figure 13: Four initial data points and ten sequentially proposed data points (including the blue triangle of the current iteration) by Bayesian optimization of cosine mixture function. Iter: Iteration. Gap: Distance to global optimum. Image Credits: `mlrMBO` Documentation.

Notwithstanding the above, one could argue BO is an optimizer and thus simply has

[23]There are situations, where a fully exploratory AF can be advantageous though. See section 6.6 for a practical example.

no need for an exact model of the true target function. That is, it can afford a globally biased surrogate model as long as it approximates the target well locally in promising regions. This viewpoint, however, is in stark contract to the explore-exploit rationale behind Bayesian optimization. A globally biased surrogate model will translate to a biased acquisition function (AF) through its mean estimates, as we have shown in theorem 1 for the case of LCB. This can potentially lead to more explorations than needed, which could slow down convergence. Section 9 will address this issue in detail.

5.2 Integrated Acquisition Function

Initially proposed by [Benassi et al., 2011] and popularized by [Snoek et al., 2012], the integrated acquisition function (IAF) claims a "fully-Bayesian" treatment of $\boldsymbol{\theta}$. The idea is to simply average over different specifications of $\boldsymbol{\theta}$ in order to propose points that are more robust towards $\boldsymbol{\theta}$, that is, that do not depend on a specific $\boldsymbol{\theta}$. This "integrating out" of $\boldsymbol{\theta}$ is done for the posterior. Thus, the integrated acquisition function $IAF_t(\boldsymbol{x})$ corresponds to the posterior expectation value of $AF_t(\boldsymbol{x}, \boldsymbol{\theta})$ with regard to $\boldsymbol{\theta}$, see definition 17.

Definition 17 (Integrated Acquisition Function)
Provided an acquisition function AF_t and the design matrix $\boldsymbol{X}_t$ in iteration t and a hyperprior $p(\boldsymbol{\theta})$, the integrated acquisition function accounts to

$$IAF_t(\boldsymbol{x}) = \int AF_t(\boldsymbol{x}, \boldsymbol{\theta})\, p(\boldsymbol{\theta}|\boldsymbol{X}_t)\, d\boldsymbol{\theta},$$

where the posterior $p(\boldsymbol{\theta}|\boldsymbol{X}_t)$ is derived from Bayes' Theorem, i.e. $p(\boldsymbol{\theta}|\boldsymbol{X}_t) = \frac{p(\boldsymbol{X}_i|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{X}_i)}$.

For the purpose of computational implementation, the integral $\int AF_t(\boldsymbol{x}, \boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{X}_t)d\boldsymbol{\theta}$ can be approximated by Markov chain Monte Carlo (MCMC):

$$\int AF_t(\boldsymbol{x}, \boldsymbol{\theta})\, p(\boldsymbol{\theta}|\boldsymbol{X}_t)\, d\boldsymbol{\theta} \approx \frac{1}{M} \sum_{m=1}^{M} AF_i(\boldsymbol{x}, \boldsymbol{\theta}_m), \tag{13}$$

where $\boldsymbol{\theta}_m$ is drawn from $p(\boldsymbol{\theta}|\boldsymbol{X}_t)$ by means of MCMC. First thought to be computationally too expensive, [Snoek et al., 2012] showed that the MCMC-sampling is worth the computational expense, as the resulting robust proposals $\boldsymbol{x}_t^* = \arg\max IAF_t(\boldsymbol{x})$ speed up convergence. Their python library `spearmint` [Snoek et al., 2012] popularized this approach widely. Under a decision-theoretic scope, the prior weights on the parameters from $\mathcal{X}^d$ are extended to the prior parameter vector $\boldsymbol{\theta}$ through the hyperprior

$p(\boldsymbol{\theta})$. This certainly makes this approach "more Bayesian" than the ML-estimation of $\boldsymbol{\theta}$. Yet, it is still far from "fully Bayesian", rendering the nomenclature by [Snoek et al., 2012] somewhat misleading. First, the surrogate model, i.e. the posterior GP, is still estimated from $\boldsymbol{X}_t$ in the optimization procedure. Second, it remains arguable as to what extent a hyperprior can be regarded as a "full" specification of the underlying prior. Since prior information about the function to be optimized Ψ is scarce, this holds for hyperprior information as well.

6 Prior-Mean-Robust Bayesian Optimization

6.1 Motivation

While a highly popular hyperparameter optimizer in machine learning [Nguyen, 2019], BO itself – not without a dash of irony – heavily depends on its hyperparameters, namely the GP prior specification. Bayesian sensitivity analysis (Section 4) has shown that BO's convergence is especially sensitive towards the kernel's functional form and the mean function's parameters. In response to the latter, we propose three modifications of BO (sections 6.3 to 6.5).

All modifications rely on the concept of imprecise Gaussian process (IGP), proposed in [Mangili, 2015] and more generally in [Mangili, 2016]. The proposed implementations of prior-mean-robust Bayesian optimization all use IGP as surrogate model(s) – be it explicitly or implicitly. The general idea of an IGP is to incorporate the model's imprecision regarding the choice of the prior's mean function parameter, given a constant mean function and a fully specified kernel, see definition 18. We will exploit this to possibly yield Bayesian optimization algorithms that are more robust towards misspecifications of the prior mean parameter.

The no free lunch theorem (NFL) [Wolpert and Macready, 1997] states that no (optimization) algorithm is universally better (that is, on all possible types of problems) than any other. Leaning on this famous consideration, we will discuss two hypotheses regarding prior-mean-robust BO hereinafter. First, our proposed method appears to be designed to perform worse than classical BO on a very specific type of problem but beat the method on related problems, while still being inferior to general-purpose algorithms on most types of problems. This hypothesis is derived from the fact that prior-mean-robust BO comprises a a bigger set of prior specifications than standard

BO. Second, this kind of generalization could also lead to better performance on a very specific type of functions, where the inherent imprecision of the function is increased, e.g. due to unobserved covariates.

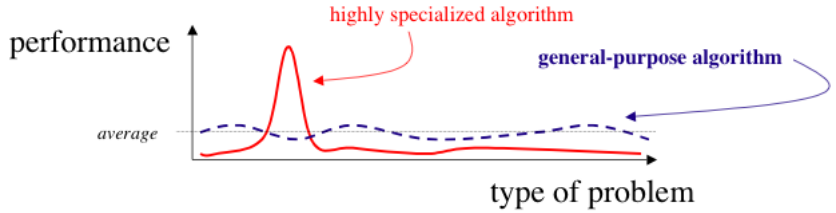

Figure 14: Visualization of the no free lunch theorem (NFL). Image Credits: Leon Fedden

In what follows, we first describe the concept of imprecise Gaussian processes and then turn to our three proposed modifications, before experimental results are presented.

6.2 Imprecise Gaussian Processes

Definition 18 (Imprecise Gaussian Process)
Consider the case of univariate regression. Given a base kernel $k_{\boldsymbol{\theta}}(x, x')$ and a degree of imprecision c, [Mangili, 2015, definition 2] defines a constant mean imprecise Gaussian process (IGP) as a set of GP priors "near ignorance".

$$\mathcal{G}_c = \left\{ GP\left(Mh, k_{\boldsymbol{\theta}}(x, x') + \frac{1+M}{c}\right) : h = \pm 1, M \geq 0 \right\} \tag{14}$$

It can be shown that $c \to 0$ yields the precise model [Mangili, 2015, page 189].

Note that the mean functional form (constant, hence $p_m = 1$) as well as both kernel functional form and its parameters do not vary in set $\mathcal{G}_c$, but only the mean parameter $\theta_{m(x),1} = Mh \in (-\infty, \infty)$. Also note that the imprecise Gaussian process (IGP) (see definition 18) is defined for univariate target functions only, i.e. $p = 1$ in $\mathcal{X}^p$. All methods proposed below are thus restricted to this univariate case. For each prior GP, a posterior GP can be inferred as described in section 3. This results in a set of posteriors and a corresponding set of mean estimates, of which the upper and lower mean estimates $\underline{\mu}(x)_c$, $\overline{\mu}(x)_c$ can be derived analytically [Mangili, 2015], see definition 20 based on definition 19.

Definition 19 (Base Kernel Matrix)
Let $k_\theta(x, x')$ be a base kernel. Analogous to definition 11, the finitely positive semi-

definite matrix $\boldsymbol{K}_n$ *is formed by applying* $k_\theta(x, x')$ *on the training data*

$$\boldsymbol{K}_n = [k_\theta(x_i, x'_j)]_{ij}.$$

We call $\boldsymbol{K}_n$ *base kernel matrix.*

Note that $\boldsymbol{K}_n$ is restricted only to be finitely positive semi-definite and not to have diagonal elements of 1. In statistical terms, $\boldsymbol{K}_n$ is a covariance matrix and not necessarily a correlation matrix. Hence, the variance $I\sigma^2$ is included, following the examples in equations 4 to 9. Diverging from [Mangili, 2015], we only consider target functions without explicit noise, thus no "nugget term" $I\sigma^2_{nugget}$ needs to be included in $\boldsymbol{K}_n$.

Definition 20 (Upper and Lower Mean Estimates of IGP)
Let x *be a scalar input, whose* $f(x)$ *is to be predicted. Then* $\boldsymbol{k}_x = [k_{\boldsymbol{\theta}}(x, x_1), ..., k_{\boldsymbol{\theta}}(x, x_n)]^T$ *is the vector of covariances between* x *and the training data. Furthermore, define* $\boldsymbol{s}_k = \boldsymbol{K}_n^{-1}\mathbb{1}_n$ *and* $\boldsymbol{S}_k = \mathbb{1}_n^T\boldsymbol{K}_n^{-1}\mathbb{1}_n$. *Then [Mangili, 2015] shows that upper and lower bounds of the posterior mean function for* $f(x)$ *can be derived. If* $|\frac{\boldsymbol{s}_k\boldsymbol{y}}{\boldsymbol{S}_k}| \leq 1 + \frac{c}{\boldsymbol{S}_k}$, *they are:*

$$\overline{\mu}(x) = \boldsymbol{k}_x^T\boldsymbol{K}_n^{-1}\boldsymbol{y} + (1 - \boldsymbol{k}_x^T\boldsymbol{s}_k)\frac{\boldsymbol{s}_k^T}{\boldsymbol{S}_k}\boldsymbol{y} + c\frac{|1 - \boldsymbol{k}_x^T\boldsymbol{s}_k|}{\boldsymbol{S}_k} \tag{15}$$

$$\underline{\mu}(x) = \boldsymbol{k}_x^T\boldsymbol{K}_n^{-1}\boldsymbol{y} + (1 - \boldsymbol{k}_x^T\boldsymbol{s}_k)\frac{\boldsymbol{s}_k^T}{\boldsymbol{S}_k}\boldsymbol{y} - c\frac{|1 - \boldsymbol{k}_x^T\boldsymbol{s}_k|}{\boldsymbol{S}_k} \tag{16}$$

If $|\frac{\boldsymbol{s}_k\boldsymbol{y}}{\boldsymbol{S}_k}| > 1 + \frac{c}{\boldsymbol{S}_k}$:

$$\overline{\mu}(x) = \boldsymbol{k}_x^T\boldsymbol{K}_n^{-1}\boldsymbol{y} + (1 - \boldsymbol{k}_x^T\boldsymbol{s}_k)\frac{\boldsymbol{s}_k^T}{\boldsymbol{S}_k}\boldsymbol{y} + c\frac{1 - \boldsymbol{k}_x^T\boldsymbol{s}_k}{\boldsymbol{S}_k} \tag{17}$$

$$\underline{\mu}(x) = \boldsymbol{k}_x^T\boldsymbol{K}_n^{-1}\boldsymbol{y} + (1 - \boldsymbol{k}_x^T\boldsymbol{s}_k)\frac{\boldsymbol{s}_k^T\boldsymbol{y}}{c + \boldsymbol{S}_k} \tag{18}$$

Definition 21 (Variance Estimates of IGP)
The corresponding variance estimate of both $\overline{\mu}(x)$ *and* $\underline{\mu}(x)$ *is*

$$\widehat{\sigma}^2_{f(x)} = k_\theta(x, x) - \boldsymbol{k}_x^T\boldsymbol{K}_n^{-1}\boldsymbol{k}_x + \frac{(1 - \boldsymbol{k}_x^T\boldsymbol{s}_k)^2}{\boldsymbol{S}_k} \tag{19}$$

Definition 22 (Credible Intervals of IGP Prediction)
For $\alpha \in [0,1]$ and z_q the q-quantile of the standard normal distribution, the $1-\alpha$ credible interval (CrI) of the mean estimate for $f(x)$ is

$$CrI_\alpha = [\underline{f}_x = \underline{\mu}(x) - z_{1-\frac{\alpha}{2}} \cdot \widehat{\sigma}^2_{f(x)}, \overline{f}_x = \underline{\mu}(x) - z_{1-\frac{\alpha}{2}} \cdot \widehat{\sigma}^2_{f(x)}].$$

[Mangili, 2015, Theorem 4] shows that $CrI_\alpha = [\underline{f}_x, \overline{f}_x]$ satisfies $\overline{P}(f(x) < \underline{f}_x) \leq \frac{\alpha}{2}$ and $\overline{P}(f(x) > \overline{f}_x) \leq \frac{\alpha}{2}$.

Figure 15 visualizes upper and lower mean function estimates as well as corresponding credible intervals of an imprecise Gaussian process trained on data generated by the first "Alpine"-function $f(x) = x \cdot \sin(x) + 0.1x$ from the package `smoof`. The prediction function including credible intervals of a precise (classical) Gaussian process with hyper-parameters estimated from the data through ML are also depicted. As can be seen by comparing predictions in $x \in [-10, -5]$ to $x \in [4, 7]$, the model imprecision $\overline{\mu}(x) - \underline{\mu}(x)$ is greater than the classical prediction uncertainty (credible interval of precise GP) in the absence of data. Here, the prior dominates the data in the posterior. The opposite holds in the abundance of data. Noteworthy, the credible intervals of $\overline{\mu}(x)$ and $\underline{\mu}(x)$ are smaller than the CrI of the precise GP.

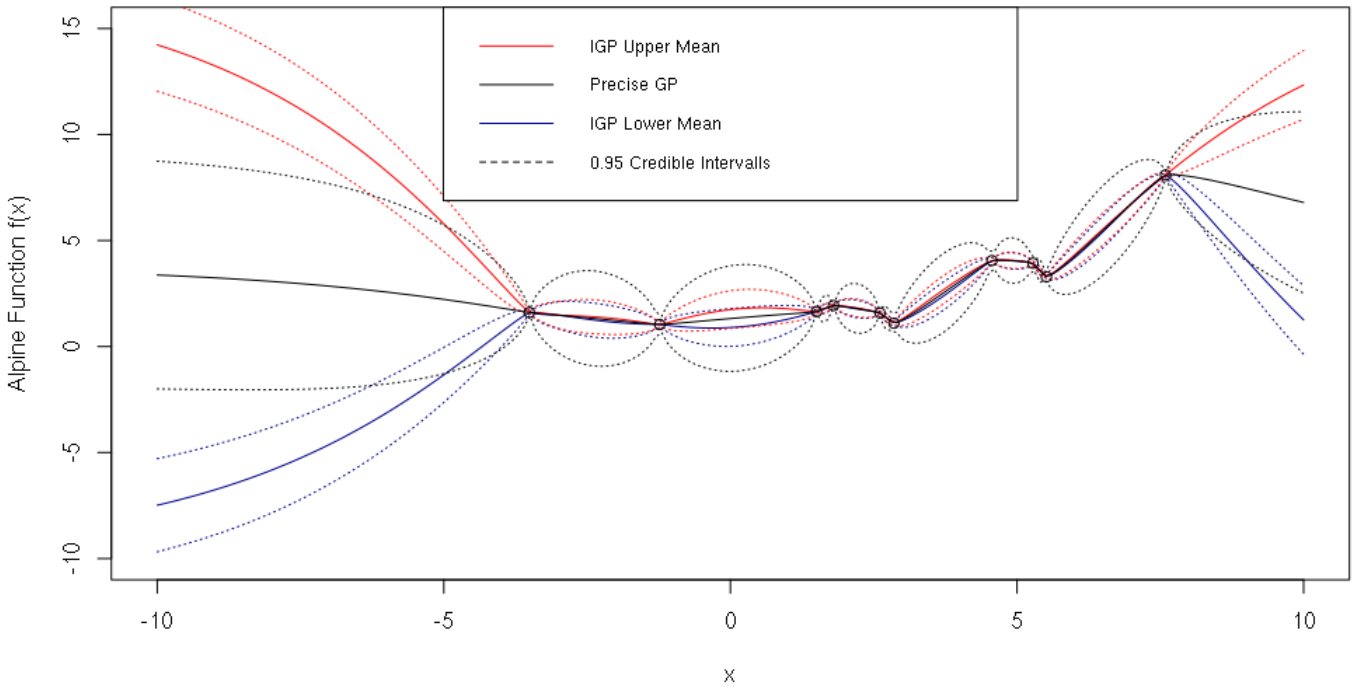

Figure 15: Upper and lower mean estimates of IGP and precise GP estimates from data generated by first alpine function.

6.3 Parallel Hedging

The most straight-forward way of rendering BO more robust towards misspecification of the prior's mean parameters is to deploy several BOs in parallel with $\underline{\mu}(x)_c$, $\overline{\mu}(x)_c$ for varying c as surrogate models. Algorithm 2 describes the procedure.

Algorithm 2 Prior-Mean-Robust Bayesian Optimization – Parallel Hedging

1: create an initial design $D = \{(\boldsymbol{x}^{(i)}, \Psi^{(i)})\}_{i=1,...,n_{init}}$ of size n_{init}
2: define budget $K+1$ of (logical) Cores with $S = \frac{K}{2} + 1$ different (I)GP models with varying c-values (one of which has $c \to 0$ for the precise model)
3: define budget B of total evaluations
4: **for each** k in K+1 **do**
5: **for** $I = \frac{B}{K+1}$ Iterations **do**
6: **train** a surrogate model (GP or $\underline{\mu}(x)_c$, $\overline{\mu}(x)_c$ of IGP_c) on data D
7: **propose** $\boldsymbol{x}^{new}$ that optimizes the acquisition function $AF(SM(\boldsymbol{x}))$
8: **evaluate** Ψ on $\boldsymbol{x}^{new}$
9: **update** $D \leftarrow D \cup (\boldsymbol{x}^{new}, \Psi(\boldsymbol{x}^{new}))$
10: **end for**
11: **save** $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in D} \Psi(\boldsymbol{x})$ and respective $\Psi(\boldsymbol{x}^*)$ as optima $(\boldsymbol{x}^*, \Psi(\boldsymbol{x}^*))_k$
12: **end for each**
13: **return** $\min_k\{(\boldsymbol{x}^*, \Psi(\boldsymbol{x}^*))_k : k \in \{1, ..., K\}\}$

In addition to the global minimum $\min_k\{(\boldsymbol{x}^*, \Psi(\boldsymbol{x}^*))_k : k \in \{1, ..., K\}\}$ across all BO runs, parallel hedging can return all $2S + 1$ optima from S imprecise surrogate models and the precise model. The $2S$ additionally proposed optima can be regarded as a hedge against getting stuck in a local optima in the regular process due to prior mean function parameter misspecification.

In line 6 of algorithm 2, the SM is trained on the data of the respective iteration. Each core uses a unique Gaussian process as surrogate model. That is, two IGPs are trained independently for each c and due to the stochastic nature of GP training (see section 3) $\underline{\mu}(x)_c$ and $\overline{\mu}(x)_c$ can be asymmetric around $\widehat{\mu}(x)$ of the precise model. We opt for this approach because its implementation is easier to integrate into the `mlr` and `mlrMBO` framework. It is also used in the batch-wise speed-up method in section 6.4. Otherwise, $\underline{\mu}(x)_c$ and $\overline{\mu}(x)_c$ would be computed in the same core reducing the budget of required cores from $K + 1$ to $S = \frac{K}{2} + 1$. We leave this to further research. However, note that

additional cores (if available, of course) come at practically no extra computational cost thanks to parallelization.

As mentioned above, IGP only varies the mean function parameter. The functional forms of mean and kernel are set manually beforehand. (We use constant trend and power-exponential kernel for our experiments in section 6.6.) The kernel parameters, however, are estimated from the data. Diverging from [Mangili, 2015], we do not use MAP but ML in order to avoid the necessity of specifying a hyperprior. `DiceKriging`'s [Roustant et al., 2012] built-in ML-optimizers from the package `rgenoud` [Mebane, Jr. and Sekhon, 2011] based on evolutionary algorithms are used to this end.

Given a limited budget of evaluations B, parallel hedging only beats classical BO if one of the $K+1$ BO runs fit the true underlying function so well and (or) speed up convergence to such a degree that outweighs the $B - \frac{B}{K+1}$ evaluations each of the $K+1$ BO runs is short of compared to classical BO. This appears rather unlikely and benchmarking results in section 6.7 will confirm this presumption. However, parallel hedging BO has an intrinsic advantage over classical BO: It provides "out-of-the-bag" sensitivity analysis. That is, one can tell from the distribution of $\{(\boldsymbol{x}^*, \Psi(\boldsymbol{x}^*))_k : k \in \{1, ..., K\}\}$ how sensitive the returned optimum is towards altering the GP prior.

What is more, parallel hedging might still be attractive from an efficiency point of view; namely in case time for each evaluation rather is the limiting factor than the budget of total evaluations B (e.g. in simulation-based applications of BO). In such setups, parallel-hedging BO might only need a little more time due to the parallelization overhead (caused by initializing parallel clusters or sessions) than BO, while the benefit of the sensitivity analysis still holds. Our implementation of IGP predictions in `R`, however, proved to be distinctly slower than the one of classical GP in `DiceKriging` that is used in `mlrMBO`. Yet, for very time-consuming evaluations of the true function (e.g. hyperparameter-tuning of huge nets in deep learning) this might be outweighed by the evaluation time.

Generally, there is still room for improvement regarding the efficiency of the implementation. Readability was preferred over speed, as the overall aim was rather a proof of concept than to write user-ready software. For instance, the IGP mean and standard error prediction could be implemented more efficiently by combining both into one function. We leave this to further projects.

6.4 Batch-Wise Speed-Up

Additionally, the budget of function evaluations can be divided into batches, after each of which the parallelization is broken and the worst performing surrogate models are thrown away. Both the remaining budget and the evaluated data is then distributed among the remaining models to speed up convergence, see pseudo-code in algorithm 3. This strategy is called successive halving (SH) and is widely popular in machine learning, see [Jamieson and Talwalkar, 2016] for instance. As an advantage over the simple parallel hedging, this batch-wise implementation tries not to waste resources in batches that do not approximate well the true underlying function.

Algorithm 3 Prior-Mean-Robust Bayesian Optimization – Batches

1: create an initial design $D = \{(\boldsymbol{x}^{(i)}, \Psi^{(i)})\}_{i=1,...,n_{init}}$ of size n_{init}
2: define initial budget $K+1$ of (logical) Cores with $S = \frac{K}{2} + 1$ different (I)GP models with varying c-values (one of which has $c \to 0$ for the precise model)
3: define budget B of total evaluations and distribute among M batches and respective number of Cores $C \in \mathbb{N}^M$ with $C = (K+1, \frac{K+1}{2}, \frac{K+1}{4}, ...)$ indexed by m.
4: compute vector of iterations per (I)GP per batch $I \in \mathbb{N}^M$ such that $\sum_{m=1}^{M} I_m \cdot C_m \leq B$
5: **for** m in M batches **do**
6: **for each** C_m **do**
7: **for** I_m Iterations **do**
8: **train** a surrogate model (GP or $\underline{\mu}(x)_c$, $\overline{\mu}(x)_c$ of IGP_c) on data D
9: **propose** $\boldsymbol{x}^{new}$ that optimizes the acquisition function $AF(SM(\boldsymbol{x}))$
10: **evaluate** Ψ on $\boldsymbol{x}^{new}$
11: **store** a surprise metric $Sur(AF(IGP_c(\mathbf{x}^{new})), \Psi(\boldsymbol{x}^{new}))$
12: **update** $D \leftarrow D \cup (\boldsymbol{x}^{new}, \Psi(\boldsymbol{x}^{new}))$
13: **end for**
14: **end for each**
15: Throw away worst $\frac{K}{2}$ batches according to averaged Sur
16: Add evaluated D from thrown away batches to remaining ones
17: **end for**
18: **return** optimum from final batch $\{(\boldsymbol{x}_i^*, \min_i \Psi(\boldsymbol{x}^*)_i \in D\}$

The surprise metric in line 11 can be specified flexibly in principle, the most natural choice being the mean prediction error, which we opt for in the following. The metric must, of course, lead to an ordinal structure on the models that allows sorting out the

worst performing ones in line 15.

This batch-wise implementation of prior-mean-robust BO can also be thought of as an attempt to cure the myopia of BO, recall section 2.4. Proposals are made not only based on the acquisition function inside the sequential BO process, but also according to a more far-sighted surprise metric that takes into account I_m iterations per batch m. However, unlike existing non-myopic BO versions such as Knowledge Gradient [Frazier et al., 2009] and Entropy Search [Hennig and Schuler, 2012], batch-wise prior-mean-robust BO looks backwards rather than ahead. Instead of accounting for hypothetical changes of the SM in iteration $t+1$ as a function of the proposal in iteration t, batch-wise prior-mean-robust BO considers the performances of the surrogate models' performances in previous iterations.

6.5 Generalized Lower Confidence Bound

Inspired by multi-objective Bayesian optimization (section 2.5) one can think of an IGP's upper and lower mean and variance predictions as individual surrogate models for different objective functions. As described in section 2.5, one approach to propose points based on various surrogate models is to scalarize their predictions by an acquisition function defined *a priori*. The herein proposed generalized lower confidence bound (GLCB) is such an acquisition function. It generalizes the popular lower confidence bound (LCB) $LCB(\boldsymbol{x}) = -\mu(\boldsymbol{x}) + \tau \cdot \sqrt{var(\mu(\boldsymbol{x}))}$ as described in definition 23 in section 2.2. The only difference to a multi-objective AF stems from the fact that the different surrogate models are estimated with regard to one and the same (single-objective) target function instead of multiple ones.

Definition 23 (Generalized Lower Confidence Bound (GLCB))
Let $\boldsymbol{x} \in \mathcal{X}$. As above, $\overline{\mu}(\boldsymbol{x})_c, \underline{\mu}(\boldsymbol{x})_c$ are the upper/lower mean estimates of an IGP with imprecision c and $\mu(\boldsymbol{x})$, $var(\mu(\boldsymbol{x}))$ are the mean and variance predictions of a precise GP. The prior-mean-robust acquisition function generalized lower confidence bound (GLCB) then is

$$GLCB(\boldsymbol{x}) = -\mu(\boldsymbol{x}) + \tau \cdot \sqrt{var(\mu(\boldsymbol{x}))} + \rho \cdot (\overline{\mu}(\boldsymbol{x})_c - \underline{\mu}(\boldsymbol{x})_c),$$

where $\tau > 0$ controls the "mean vs. data uncertainty" trade-off (degree of risk aversion) and $\rho > 0$ controls the "mean vs. model imprecision" trade-off (degree of ambiguity aversion).

Note that $\overline{\mu}(\boldsymbol{x})_c - \underline{\mu}(\boldsymbol{x})_c$ simplifies to an expression only dependent on the kernel vector between x and the training data $\boldsymbol{k}_x = [k_{\boldsymbol{\theta}}(x, x_1), ..., k_{\boldsymbol{\theta}}(x, x_n)]^T$, the base kernel matrix $\boldsymbol{K}_n$ and the degree of imprecision c, which follows from equations 17 and 18 in case $|\frac{\boldsymbol{s}_k \boldsymbol{y}}{\boldsymbol{S}_k}| > 1 + \frac{c}{\boldsymbol{S}_k}$:

$$\begin{aligned}\overline{\mu}(x) - \underline{\mu}(x) &= (1 - \boldsymbol{k}_x^T \boldsymbol{s}_k)\frac{\boldsymbol{s}_k^T}{\boldsymbol{S}_k}\boldsymbol{y} + c\frac{1 - \boldsymbol{k}_x^T \boldsymbol{s}_k}{\boldsymbol{S}_k} - (1 - \boldsymbol{k}_x^T \boldsymbol{s}_k)\frac{\boldsymbol{s}_k^T \boldsymbol{y}}{c + \boldsymbol{S}_k} \\ &= (1 - \boldsymbol{k}_x^T \boldsymbol{s}_k)\big(\tfrac{\boldsymbol{s}_k^T}{\boldsymbol{S}_k}\boldsymbol{y} + \tfrac{c}{\boldsymbol{S}_k} - \tfrac{\boldsymbol{s}_k^T \boldsymbol{y}}{c + \boldsymbol{S}_k}\big) \end{aligned} \tag{20}$$

As can be seen by comparing equations 15 and 16, in case of $|\frac{\boldsymbol{s}_k \boldsymbol{y}}{\boldsymbol{S}_k}| \leq 1 + \frac{c}{\boldsymbol{S}_k}$, the model imprecision $\overline{\mu}(\boldsymbol{x})_c - \underline{\mu}(\boldsymbol{x})_c$ even simplifies further:

$$\overline{\mu}(x) - \underline{\mu}(x) = 2c\frac{|1 - \boldsymbol{k}_x^T \boldsymbol{s}_k|}{\boldsymbol{S}_k} \tag{21}$$

In this case, the GLCB comes down to $GLCB(\boldsymbol{x}) = -\mu(\boldsymbol{x}) + \tau \cdot \sqrt{var(\mu(\boldsymbol{x}))} + 2 \cdot \rho c\frac{|1 - \boldsymbol{k}_x^T \boldsymbol{s}_k|}{\boldsymbol{S}_k}$ and the two hyperparameters ρ and c collapse to one.

In both cases, the surrogate models $\underline{\mu}(x)_c$ and $\overline{\mu}(x)_c$ do not have to be fully implemented. Only $\boldsymbol{K}_n$ and $\boldsymbol{k}_x = [k_{\boldsymbol{\theta}}(x, x_1), ..., k_{\boldsymbol{\theta}}(x, x_n)]^T$ need to be computed. GLCB can thus be plugged into standard BO as an alternative AF without further ado and much computational cost. Algorithm 4 describes the procedure.

Algorithm 4 Prior-Mean-Robust Bayesian Optimization – Generalized Lower Confidence Bound (GLCB)

create an initial design $D = \{(\boldsymbol{x}^{(i)}, \Psi^{(i)})\}_{i=1,...,n_{init}}$ of size n_{init}
specify c and ρ
while termination criterion is not fulfilled **do**
    train a precise GP on data D and obtain $\mu(\boldsymbol{x})$, $var(\mu(\boldsymbol{x}))$
    compute $\boldsymbol{K}_n$ and $\boldsymbol{k}_x = [k_{\boldsymbol{\theta}}(x, x_1), ..., k_{\boldsymbol{\theta}}(x, x_n)]^T$
    if $|\frac{\boldsymbol{s}_k \boldsymbol{y}}{\boldsymbol{S}_k}| > 1 + \frac{c}{\boldsymbol{S}_k}$ **then**
    $\overline{\mu}(x)_c - \underline{\mu}(x)_c = (1 - \boldsymbol{k}_x^T \boldsymbol{s}_k)\big(\frac{\boldsymbol{s}_k^T}{\boldsymbol{S}_k}\boldsymbol{y} + \frac{c}{\boldsymbol{S}_k} - \frac{\boldsymbol{s}_k^T \boldsymbol{y}}{c + \boldsymbol{S}_k}\big)$
    else $\overline{\mu}(x)_c - \underline{\mu}(x)_c = 2c\frac{|1 - \boldsymbol{k}_x^T \boldsymbol{s}_k|}{\boldsymbol{S}_k}$
    compute $GLCB(\boldsymbol{x}) = -\mu(\boldsymbol{x}) + \tau \cdot \sqrt{var(\mu(\boldsymbol{x}))} + \rho \cdot (\overline{\mu}(\boldsymbol{x})_c - \underline{\mu}(\boldsymbol{x})_c)$
    propose $\boldsymbol{x}^{new}$ that optimizes $GLCB(\boldsymbol{x})$
    evaluate Ψ on $\boldsymbol{x}^{new}$
    update $D \leftarrow D \cup (\boldsymbol{x}^{new}, \Psi(\boldsymbol{x}^{new}))$
end while
return $\arg\min_{\boldsymbol{x} \in D} \Psi(\boldsymbol{x})$ and respective Ψ_{min}

GLCB explicitly accounts for the surrogate model's imprecision with regard to its prior mean parameter during the optimization procedure. Just like LCB, the generalized LCB balances optimization of $\mu(x)$ and reduction of uncertainty with regard to the model's prediction variation $\sqrt{var(\mu(\boldsymbol{x}))}$ through τ. What is more, GLCB aims at reducing model imprecision caused by the prior specification, controllable by ρ. Ideally, this would allow returning optima that are robust not only towards classical prediction uncertainty but also towards imprecision of the specified model.

6.6 Experiments

All three aforementioned proposals have been implemented in `R`, see section B. Upper as well as lower mean and standard error prediction functions were integrated into the `mlr` framework as `S3` objects. They could then be plugged into `mlrMBO` [Bischl et al., 2017]. For parallelizations in algorithms 2 and 3, packages `doParallel` and `foreach` were used. `DiceKriging` [Roustant et al., 2012] was used to estimate the precise GP models.

The proposed methods have been tested on synthetic benchmark functions from the

`R` packages `smoof` [Bossek, 2017] and `soobench` [Mersmann et al., 2020] as well as on univariate target functions generated from three data sets. The first one describes the quality of experimentally produced graphene, an allotrope of carbon with (potential) use in semiconductors, smartphones and electric batteries [Wahab et al., 2020]. The second one contains historical meteorological data from Los Angeles, measured from January to December 1976 [Breiman and Friedman, 1985], while the third one comprises heartbeat time series from a cardiological study on heart arrhythmia (irregular heartbeat) [Goldberger and Rigney, 1991], see data set descriptions below.

We compare the performance of parallel hedging prior-mean-robust BO and batch-wise prior-mean-robust BO to classical BO with a budget of 90 evaluations and an initial design of size $n = 10$ generated by latin hypercube sampling (LHS). The optimization runs are repeated $n = 40$ or $n = 60$ times[24] and the mean optimization path (MOP) (see definition 13) including bootstrapped 0.95-confidence intervals are computed. The popular (and default in many libraries) expected improvement (EI) (definition 2) is deployed as acquisition function (AF) in all these set-ups. We use focus search (random search with narrowing search space) with 200 evaluated points per iteration and two maximal restarts as infill optimizer. Parallel hedging prior-mean-robust BO uses one IGP with $c = 50$ resulting in 3 models with 90 evaluations each. Batch-wise prior-mean-robust BO uses three IGPs with $c = 1, 10, 100$ resulting in 7 models in the first batch, 4 in the second and 1 in the last batch with 6, 10, 18 evaluations per model per batch. To ensure numerical stability of `DiceKriging`, a tiny noise (nugget) term has been added to the data for the precise model.[25]

We also compare generalized lower confidence bound (GLCB) (definition 6.5) to its classical counterpart lower confidence bound (LCB) (definition 3) as well as to other well-established acquisition functions, namely expected improvement (EI) (definition 2), augmented expected improvement (AEI) (definition 7), expected quantile improvement (EQI) (definition 8), adaptive lower confidence bound (ALCB) (definition 4) and standard error (SE) (definition 5). Besides their popularity in various applications, these AFs have been chosen for practical reasons – they are all fully implemented in `mlrMBO`. For pairwise comparisons of GLCB to each of them, we observe $n = 60$ BO runs with a budget of 90 evaluations and an initial design of 10 data points generated by latin hypercube sampling (LHS) each. Focus search was used as infill optimizer with 1000 evaluations per round and 5 maximal restarts.

[24] n was adjusted to bring about more precise estimates.

[25] See respective Github issue.

All experiments were computed on a high performance computing cluster using 20 64-bit-cores (linux gnu). The experiments were conducted in `R` version 4.0.3 [R Core Team, 2020].

6.6.1 Graphene Data

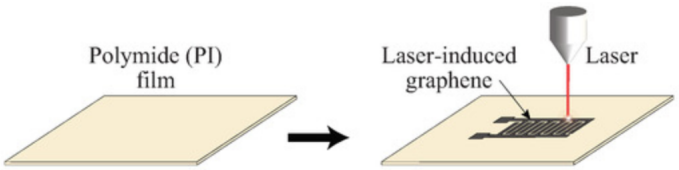

Figure 16: Producing graphene via laser irradiation. Image credits: [Kotthoff, 2019]

The graphene data set comprises $n = 210$ observations of an experimental manufacturing process of graphene. A polymide film, typically Kapton, is irradiated with laser in a reaction chamber in order to trigger a chemical reaction that results in graphene, see figure 16. The data set comprises four parameters that influence the manufacturing process such as power and time of the laser irradiation, see table 4. The target variable (to be maximized) is a measure for the quality of the induced graphene, ranging from 0.1 to 5.5.

feature	min	max	type	description
power	10	5555	real-valued	power of the laser
time	500	20210	real-valued	irradiation time
gas			categorical	gas used in the reaction chamber (Nitrogen, Argon, Air)
pressure	0	1000	real-valued	pressure in the reaction chamber
target quality	0.1	5.5	real-valued	quality of induced graphene

Table 4: Graphene data set [Wahab et al., 2020].

In order to construct a univariate target function from the data set, a random forest (RF) was trained on subsets of it (target quality and power as well as target quality and time, see figure 17). The predictions of these RFs were then used as target functions to be optimized in order to compare the proposed BO methods to existing ones on a real-world problem.

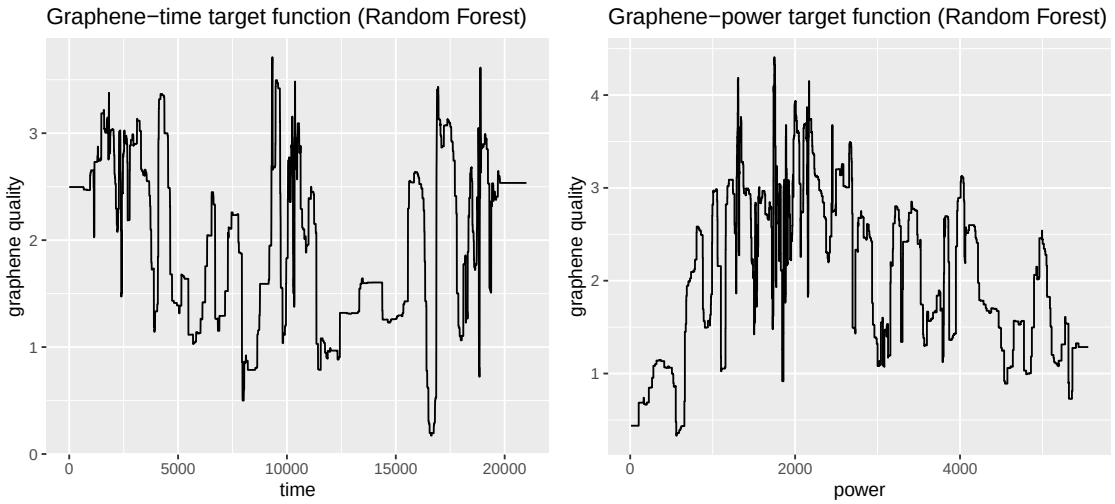

Figure 17: Univariate target functions estimated from graphene data.

6.6.2 Air Pressure Data

The second data set describes 13 meteorological variables measured 1976 in Los Angeles. It is obtained from the `R` package `mlbench` [Breiman and Friedman, 1985]. We therefrom select two variables, air pressure and humidity, and again train a RF on them. The air pressure is measured as 500 millibar pressure height, that is the height, where the air pressure reaches 500 millibar. As figure 18 shows, air pressure is used as target variable and humidity as covariate.

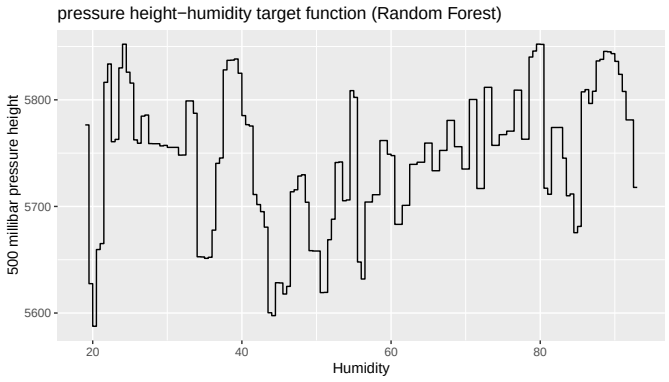

Figure 18: Air pressure as a function of humidity (RF).

6.6.3 Heartbeat Time Series

While both the graphene quality and the air pressure functions stem from multivariate data sets, we also aim at optimizing a function from a genuine univariate data set.

To this end, the heartbeat time series of two anonymous individuals from the publicly available MIT-BIH arrhythmia database [Goldberger and Rigney, 1991] were included in the analysis. A RF was fitted on the data, as figures 19 and 20 indicate.

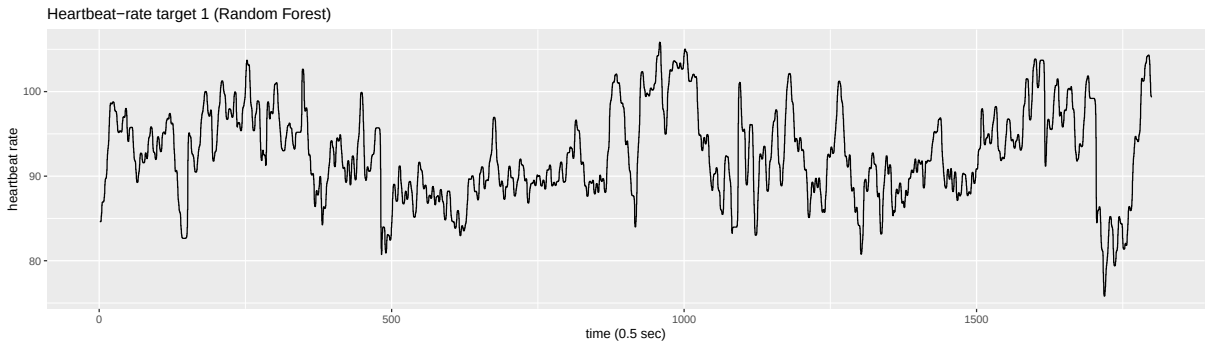

Figure 19: Heartbeat rate of individual 1 (RF).

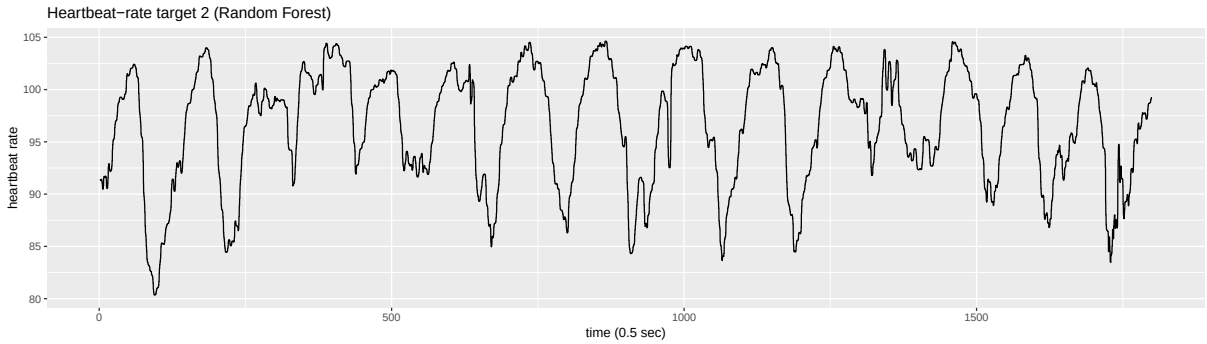

Figure 20: Heartbeat rate of individual 2 (RF).

6.6.4 Synthetic Functions

For the sake of completeness and comparability, we also benchmark the proposed methods against established synthetic test functions. While the above target functions (all based on real-world data) are rather wiggly, most synthetic functions are of smooth nature.

All available univariate functions from the `R` packages `smoof` and `soobench` have been selected. Such ones with infinite domains or target values of 0 globally were removed, ending up with 31 distinct univariate functions. Aiming at variability regarding multimodality and smoothness, we select six different target functions, see figure 21.

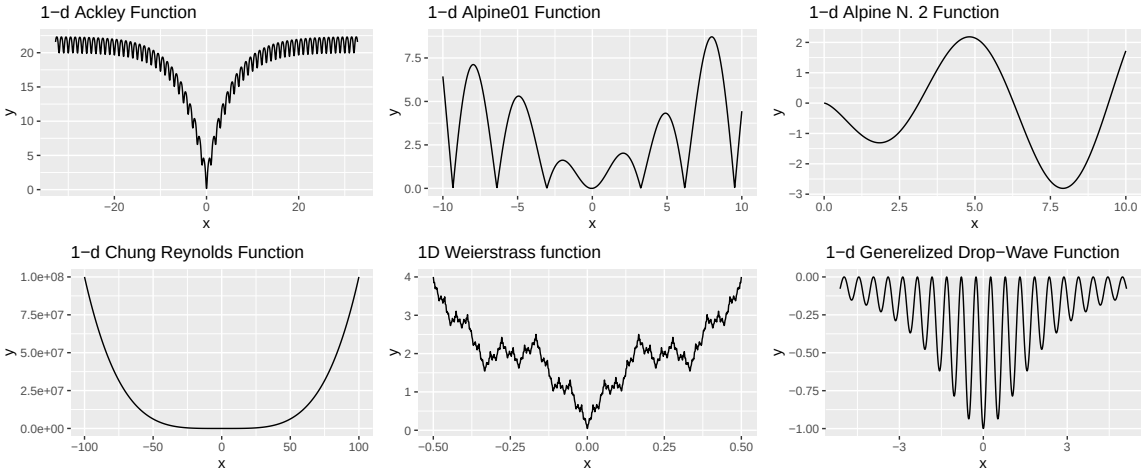

Figure 21: Selected synthetic target functions

6.7 Results

We find that parallel hedging and batch-wise speed-up prior-mean-robust BO (respective plots have white background) fail to beat classical BO, while GLCB (with varying ρ and c, dark plots) systematically finds better configurations than LCB on the graphene data set. Moreover, GLCB surpasses several other acquisition functions, including the popular AF expected improvement (EI), in late iterations. However, it does not outperform the extreme case of the purely exploratory AF standard error, i.e. LCB with $\tau \to \infty$. Additionally, GLCB is inferior to some AFs on other test functions, see section 6.7.2. Selected key findings are depicted below, while complete results can be found in the appendix, see figures 34 to 51.

6.7.1 Parallel Hedging and Batch-Wise Speed Up

Figure 22 shows mean optimization paths (see definition 13) including 95%-confidence intervals (CI) for parallel hedging and batch-wise speed-up prior-mean-robust BO compared to classic BO on the graphene-time target function. It becomes evident that both modifications – in spite of overlapping confidence intervals especially in late iterations – cannot compete with the classic BO.

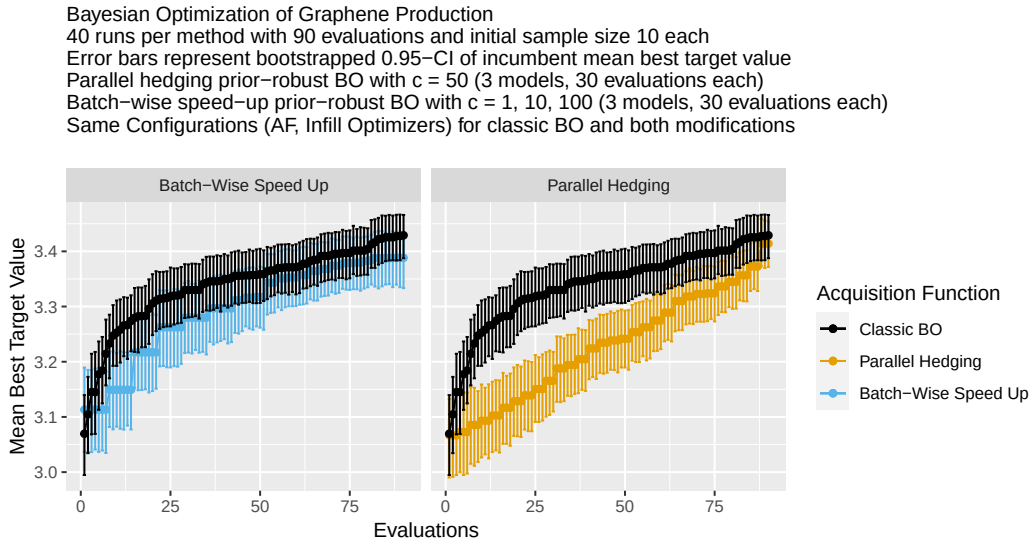

Figure 22: Benchmarking results from graphene data: Batch-wise speed-up BO (left) and parallel hedging BO (right) vs. classic BO (black).

A similar picture emerges when visualizing results from hedging and batch-wise speed up on the synthetic functions, see figure 34 in the appendix. The parallel implementations always lag behind classic BO with batch-wise speed-up performing better than parallel hedging.

6.7.2 Generalized Lower Confidence Bound (GLCB)

Since preliminary results were more promising in case of GLCB, this proposal was assessed in greater detail. In what follows, key results from graphene, air pressure and heartbeat data as well as from synthetic functions (in this order) are presented.

Figure 23 depicts mean optimization paths (MOPs) of generalized lower confidence bound (GLCB) compared to classic acquisition functions on the graphene-time target function. MOPs are shown for three different GLCB settings: $\rho = 1, c = 50$ and $\rho = 1, c = 100$ and $\rho = 10, c = 100$. Figure 23 shows that GLCB surpasses LCB (all settings) and EI ($\rho = 10, c = 100$) in late iterations. GLCB was compared to four other acquisition functions, outperforming three (AEI, ALCB and EQI) of them, see figures 35 and 36 in the appendix. As can be seen in figure 35, the highly exploratory AF standard error (SE) (definition 5) detects configurations that are as good as ($\rho = 10, c = 100$) or better than ($\rho = 1, c = 50$ and $\rho = 1, c = 100$) the ones found by GLCB. As introduced in section 2.2, this AF uses the standard error as an exclusive criterion to propose points, leading to a strategy of maximal exploration.

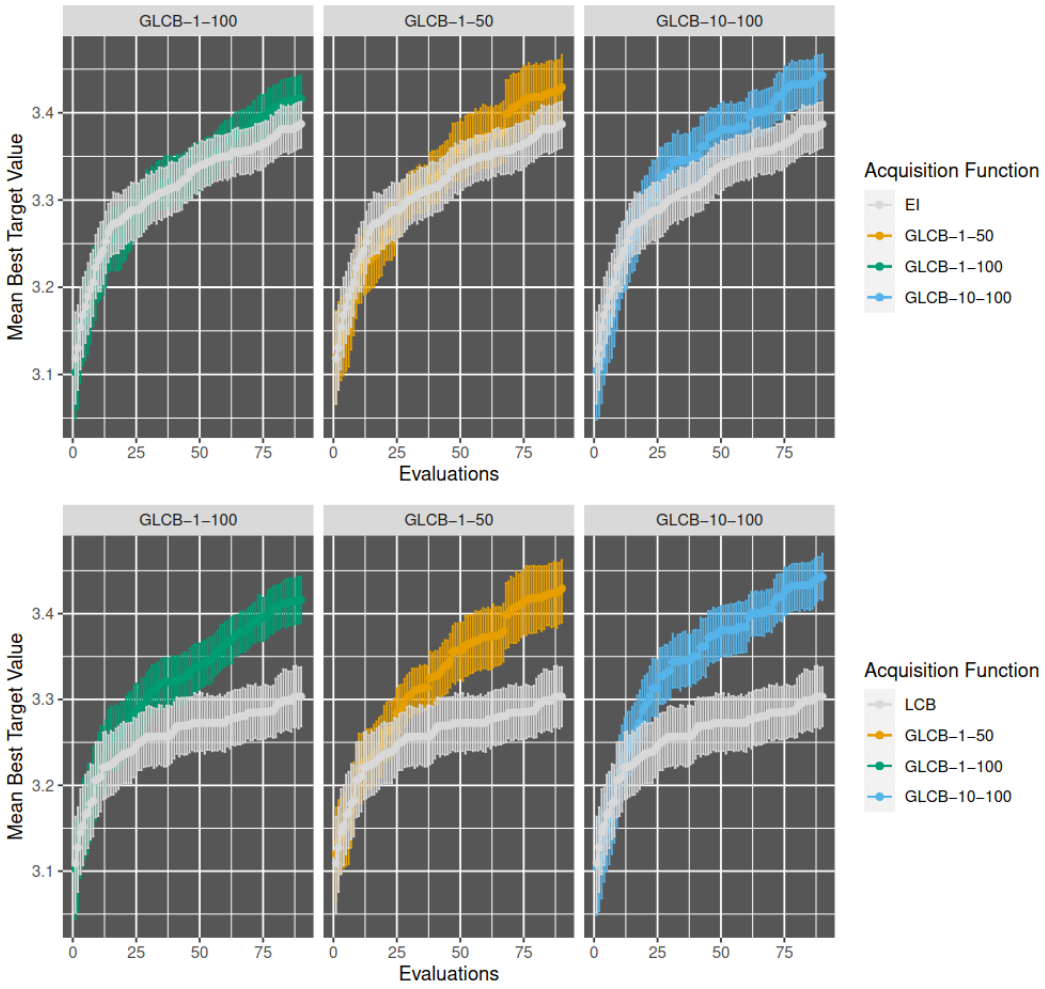

Figure 23: Benchmarking results from graphene data: generalized lower confidence bound (GLCB) vs. expected improvement (EI) and lower confidence bound (LCB). 60 runs per Acquisition Function with 90 evaluations and initial sample size 10 each. Error bars represent 0.95-CI. GLCB-1-100 means rho = 1 and c = 100. tau = 1 for all GLCBs.

Optimization of the second target function estimated from the graphene data (graphene quality depending on laser power) reveals a slightly different picture shown by figures 37 and 38 in the appendix. While GLCB still surpasses LCB for all tested values of ρ and c, it does not beat the popular EI. What is more, it is also surpassed by the rather exploitative AF EQI, while still being superior to ALCB. Interestingly, GLCB beats SE for small values of ρ and c.

On the air pressure data, however, GLCB remains unbeaten, performing as good as most classic AFs, see figures 39 and 40 in the appendix. It outperforms LCB, ALCB and AEI, while achieving similar results like SE, EQI and EI.

On the other hand, GLCB's performance on the heartbeat time series is rather disap-

pointing. Here, GLCB is only superior to AEI on both series. It achieves similar results like SE and LCB and worse results than all remaining AFs (EI, ALCB, EQI). Figures 41-44 in the appendix reveal all the details.

For five of the six synthetic test functions, GLCB is inferior to classic AFs. It is only on the drop-wave function that GLCB surpasses LCB and ALCB with sufficiently high imprecision c. Figures 45-49 in the appendix depict comparisons of GLCB to EI and LCB on Ackley, Alpine, Alpine-2, Chung-Reynolds and Weierstrass function (in this order). Figures 50 and 51 provide a more detailed assessment of benchmarking results on the drop-wave function.

6.8 Discussion

Parallel hedging and batch-wise prior-mean-robust BO do not converge faster than classic BO on any of the test functions. For the latter method, this is not a severe drawback, since it is primarily designed for providing more robust results. As mentioned in section 6.3, parallel hedging provides an "out-of-the-bag" sensitivity analysis and could be used to define a stopping criterion, see section 8. The batch-wise technique, however, explicitly tries to translate accounting for model imprecision into performance gains. The results show that it clearly misses this aim.

In the case of sequentially accounting for model imprecision in the form of the generalized lower confidence bound (GLCB), results are more promising. GLCB appears to be superior to classic AFs when confronted with multimodal and wiggly target functions – that is, not with all of them. On the graphene-power-function (figures 37 and 38) it looses to EI and EQI. Yet, on the graphene-time-function, it converges faster than or as fast as any other acquisition function in case of $\rho = 10, c = 100$. These ambiguous results show how much more research is needed to understand the determinants of optimization problems that can be solved more efficiently by accounting for model imprecision. One such determinant could be the prominent steps of both the graphene-time and the air pressure target function[26]. This, of course, needs further investigation.

What is more, the results also shed some light on the two hypotheses derived from the no free lunch theorem (NFL) in section 6.1. At least at first sight, they apparently support the second hypothesis: Accounting for model imprecision makes the BO perform better

[26] stemming from scarce data in the random forest prediction.

on a specific type of problem – of whatever nature, see above – rather than extending the range of applicable problems. Albeit, we recommend more research to test the first hypothesis, e.g. by assessing performance on functions that slightly deviate from an "ideal" function.

Less surprisingly, the degenerated and purely exploratory variant of LCB named SE competes with GLCB on many problems, with the notable exception of the graphene-power function. With $\tau \to \infty$ in the classical LCB, the exploratory part is blown up and appears to mimic the added exploratory part that aims at reduction of imprecision in GLCB, given of course that model imprecision and data uncertainty are correlated. The question of whether the latter always holds or not, should be addressed by further research. A similar consideration applies to other acquisition functions: The more exploratory they are, the more likely they compete with GLCB; see for example EI and EQI on figures 35, 36, 37 and 38.

As mentioned above, GLCB competes with SE on the heartbeat time series data. Intriguingly, these time series are the only native (unmodified) univariate data sets. The model imprecision should thus be rather low due to the lack of unobserved covariates.[27] Presumably, the problem requires some exploitation and an overall extremely high exploratory behavior may thus harm convergence. Consequently, this might be the reason why LCB with $\tau \to \infty$ is inferior to GLCB with moderate c and ρ values, see figures 41-44 in the appendix. This interpretation is also in line with the fact that GLCB is outperformed by other – more exploitative – AFs such as ALCB on the heartbeat time series. What is more, it once again supports the above mentioned presumption that model imprecision and classical data-related uncertainty correlate in some problems.

Another pattern from the results catches one's attention immediately, namely the late iterations, in which GLCB outperforms its competitors, with the exception of GLCB vs. SE on the graphene-power function. Loosely speaking, accounting for model imprecision apparently needs some time to play out its strengths. Only logically, the reduction of model imprecision needs a few iterations to impact the model's predictions that in turn impact the algorithm's proposals. This motivates an extension of our acquisition function to more complex multivariate target functions, as they usually require a higher budget of BO evaluations to be optimized. Recall that we restricted ourselves to the univariate case due to the one-dimensional nature of the imprecise Gaussian process (IGP) proposed by [Mangili, 2015]. The fruitful application of IGP in BO might initiate

[27] Needless to say, unobserved covariates can be and most certainly are still at play. However, their impact can be assumed to be rather small on heartbeat time series of a few minutes.

a more general formulation of IGP.

6.9 Limitations

It should be made clear again that the proposed BO modifications make the optimizer robust only with regard to possible misspecification of the mean function parameter given a constant trend. Albeit the Bayesian sensitivity analysis conducted in section 4 demonstrated their importance, the mean parameter is clearly not the only influential component of the GP prior in BO. For instance, the functional form of the kernel also plays a major role. The question of how to account for imprecision with regard to this prior component is briefly discussed in section 7.

Apart from this, it is important to note that all proposed methods depend on a subjectively specified degree of imprecision c. They do not account for any imaginable mean parameter vector $\boldsymbol{\theta}_m$ (the model would become vacuous). What is more, it may be difficult to interpret c and thus specify it in practical applications. Notwithstanding the above, GLCB still offers more generality than a precise choice of the mean parameter. As a matter of fact, it is easier to choose c than a precise $\boldsymbol{\theta}_m$.

7 Related Work

While there exists a vast amount of literature dealing with Bayesian optimization, merely a handful of it is explicitly concerned with robustness, not to mention model imprecision and robustness towards misspecification of the surrogate model.[28]

7.1 Robust Bayesian Optimization

In a very recent work [Makarova et al., 2021] address the issue of overfitting in tuning hyperparameters of ML models by BO. As parameters are typically optimized with regard to the training error, the (unknown) test error can increase with BO iterations while the training error (of the best incumbent configuration) still monotonically decreases. The authors show that cross-validation can mitigate this, but comes at high computational cost. As an alternative, they propose a regret-based stopping criterion

[28]The well-established field of robust optimization [Ben-Tal et al., 2009] deals with imprecise linear programming, where an analytical description of the target function - unlike in case of BO - exists.

loosely inspired by the popular regularization technique early stopping in deep learning (DL). Furthermore, [Makarova et al., 2021] present the first detailed investigation of this overfitting behavior on three types of machine learning models (linear models, boosting and random forests).

In statistics, quantile regression is a well-known alternative to mean regression. It is more robust against outliers in the response measurements than the standard linear model. [Moriconi et al., 2020] deploy quantile GP regression in BO. [Shah et al., 2014] show that Student-t processes are more flexible than Gaussian processes as prior over functions in a functional regression setting. They verify by simulation studies that Student-t processes are superior to Gaussian ones as surrogate models in Bayesian optimization on a wide range of problems. [Kirschner et al., 2020] propose a modification of Bayesian optimization that is robust towards distributional shifts of covariates, i.e. situations where the training data is sampled from a different distribution than the test data. [Nguyen et al., 2020] take a similar approach for the special case of Bayesian quadrature optimization (BQO), where the expectation of an expensive black-box integrand taken over a known probability distribution is maximized.

7.2 Model Imprecision

The above mentioned approaches certainly render BO more robust towards false confidence in its prediction due to unreliable data (underestimation of data uncertainty). However, they do not account for model imprecision. That is, robustness towards misspecification of the surrogate model is not taken into consideration. The python library with the promising name `RoBO` (robust Bayesian optimization) is robust against model misspecification only to the extent that the package provides implementations of different surrogate models and acquisition functions [Klein et al., 2017, page 2].

As far as we know, there is only one clear exception: [Malkomes and Garnett, 2018] come up with a simple, yet particularly thrilling idea: "Automating Bayesian optimization with Bayesian optimization". They suggest to optimize over a space of models in an inner loop nested inside BO. Just like in the outer loop, BO is used as an optimizer as proposed in [Malkomes et al., 2016]. The model space is defined by multiplication and addition of base kernels, see [Duvenaud et al., 2013] and [Duvenaud, 2014]. In other words, from the four components of the GP prior introduced in section 5 only the functional form of the kernel is varied, which was found to be the second-most influential component in the Bayesian sensitivity analysis conducted in section 4.

As opposed to [Malkomes and Garnett, 2018], we vary the mean function's parameter(s) since they were found most influential in the Bayesian sensitivity analysis conducted in section 4. To the best of our belief, this approach of prior-mean-robust Bayesian optimization has not appeared in the literature so far.

8 Extensions of Prior-Mean-Robust Bayesian Optimization

The thesis at hand opens several venues for future work. Based on considerations throughout this thesis, especially in section 2 and 5, the concepts proposed in section 6 can be extended to address a variety of problems.

8.1 Early Stopping in Hyperparameter-Tuning

It is usually difficult to specify the number of BO iterations needed to find a sufficiently optimal configuration beforehand. The work of [Makarova et al., 2021] on overfitting in Bayesian optimization, see section 7, demonstrates the importance of theoretically sound and computationally feasible stopping criteria in BO.

We argue that parallel hedging prior-mean-robust BO intrinsically provides a stopping criterion that might fulfill these criteria. Recall from the pseudo code in algorithm 2 that parallel hedging BO provides "out-of-the-bag" sensitivity analysis. That is, one can tell from the distribution of $\{(\boldsymbol{x}^*, \Psi(\boldsymbol{x}^*))_k : k \in \{1, ..., K\}\}$ – if accessible – at iteration t how sensitive the returned optimum is towards altering the GP prior. Hence, the degree of variation in $\{(\boldsymbol{x}^*, \Psi(\boldsymbol{x}^*))_k : k \in \{1, ..., K\}\}$ can be used as a stopping criterion. Although this variation does not stem from data uncertainty caused *inter alia* by the train-test-split in hyperparameter-tuning, but from model imprecision, it may nevertheless serve as a proxy for the algorithm's confidence in the incumbent optimum. Intuitively, such a criterion would stop the process as soon as the (classical) uncertainty related to the training data outweighs the imprecision due to prior specification. From this point on, further exploitation on the training data might cause overfitting.

To be more precise, a possible stopping criterion derived from parallel hedging would be to stop the process if $v_{stop} < \epsilon$, where

$$v_{stop} = \max_k(\boldsymbol{x}_k^*) - \min_k(\boldsymbol{x}_k^*), k \in \{1, ..., K\} \tag{22}$$

or alternatively

$$v_{stop} = \min_k(\Psi(\boldsymbol{x}^*)_k) - \max_k(\Psi(\boldsymbol{x}^*)_k), k \in \{1, ..., K\}. \tag{23}$$

For high-performance computing set-ups with many cluster, i.e. large K, one might also set

$$v_{stop} = Var(\Psi(\boldsymbol{x}^*)_k), k \in \{1, ..., K\}. \tag{24}$$

In case of small v_{stop}, confidence in the optimum is high. In case of high v_{stop}, there still might be some room for improvement in subsequent iterations. Due to time restrictions of the thesis, the stopping criterion was not implemented and tested. Further research on how it performs compared to other stopping criteria and an on the choice of ϵ is recommended.

8.2 Low Fidelity Bayesian Optimization

As outlined in section 2.6, it is common practice in many applications to approximate very expensive functions by a (hyper-)surrogate model and then treat this model as ground truth in (Bayesian) optimization. Evaluations become cheaper, while additional hyperparameters are introduced by the hyper-surrogate model, representing a second layer of model imprecision. In such scenarios, imprecise Gaussian processes (definition 18) are an attractive option in two ways. They can either be used as hyper-surrogate models or as surrogate models directly, i.e. disguised as parallel hedging prior-mean-robust BO (section 6.3) and batch-wise prior-mean-robust BO (section 6.4). Both approaches might be preferable to classic low fidelity Bayesian optimization for two reasons. First and foremost, they allow to make the optimization more robust towards the choice of the hyper-surrogate model. Second, their main disadvantage of requiring more evaluations is of smaller consequences, as evaluations are way cheaper. Section 6.6 contained an example of the latter approach to low fidelity prior-mean-robust BO already since the random forest (RF) trained on the graphene data set was treated as ground truth during optimization. However, the budget was still regarded as limiting

factor because the overall aim was to mimic optimization of an expensive-to-evaluate wiggly functional relationship. Further research particularly on the second approach, i.e. on using IGPs as hyper-surrogate models is highly recommended.

8.3 Imprecise Random Forests

As mentioned in section 2.1, random forest (RF) is a popular surrogate model in BO when dealing with categorical covariates. Following similar considerations as in the Bayesian sensitivity analysis of section 4, the effect of RF's hyperparameter *mtry* on the optimization path can be assessed and, if found influential, explicitly taken into account by means of parallelization leaning on sections 6.3 and 6.4 or a generalized acquisition function leaning on the generalized lower confidence bound (GLCB) from section 6.5.

Another approach would be to integrate already existing imprecise extensions of random forests into BO, see [Utkin et al., 2019] for an overview. Some of these extensions explicitly aim at modeling *aleatoric* uncertainty, i.e. "uncertainty due to random variation of the quantity or event being analyzed" [Balch et al., 2019][Section 1.3]. It needs to be clarified how such surrogate models relate to the general framework of BO. At first sight, their integration seems reasonable in case of categorical data, given that some formulations of BO already account for noisy data in the case of continuous parameters through nugget estimates, as briefly mentioned in section 6.2. Apart from the extensions discussed in [Utkin et al., 2019], imprecise splitting rules [Nalenz and Augustin, 2019] for decision trees could be used for trees in ensemble learners like random forests and thus also help modeling *aleatoric* uncertainty in the data.

8.4 Causal Bayesian Optimization

As explained in sections 6.3 and 6.4, imprecise Gaussian processes add some computational overhead to BO compared to precise Gaussian processes. In very high-dimensional parameter spaces, it might thus be advantageous to restrict the optimization or at least focus it on a relevant subset of the parameters. What is more, in problems like neural architecture search (NAS) (optimizing the design of neural networks[29]), the parameter space might have a hierarchical (conditional) structure. That is, the dimensionality p of $\mathcal{X}^p$ can depend on proposed optimal configurations of a

[29]Loosely speaking, NAS can be regarded as hyperparameter-tuning in deep learning.

subset of $\mathcal{X}^p$. For instance, the number of layer sizes (i.e. number of number of neurons) of course depends on the number of layers. [Swersky et al., 2014] propose a kernel functional form for such conditional parameter spaces.

Another way to reduce the computational complexity of evaluating the parameter space is to "make implicit assumptions about causal relationships explicit" [Pearl et al., 2016], i.e. to integrate causal inference. [Aglietti et al., 2020] just recently proposed causal Bayesian optimization (CBO), a method in which causal structure in $\mathcal{X}^p$ is exploited in order to reduce the number of function evaluations. This is achieved through a causal GP (see [Aglietti et al., 2020][section 3.2]) that takes into account a structural causal model of the data as well as a causal version of EI as AF that accounts for the cost of the evaluation. The above proposed generalized lower confidence bound (GLCB) could be adapted in a similar manner and thus used as AF in CBO without further ado. The causal GP prior might potentially be generalized to an IGP prior and thus integrated in parallel-hedging or batch-wise prior-mean-robust BO. Further research is needed here.

Even further, another concept from imprecise probabilities (IP) could render CBO more robust towards model imprecision. The structural causal model used in CBO could be generalized to a credal net [Cozman, 2000]. In a recent work, [Zaffalon et al., 2020] show that structural causal models can be solved by established algorithms for inference in credal nets [Augustin et al., 2014, Chapter 9.5] such as "ApproxLP" and the credal version of the variable elimination algorithm, "CVE". Interventions in CBO [Aglietti et al., 2020, algorithm 1] can then be performed by computing lower and upper bounds of $\mathbb{P}(Y|do(X = x))$ through credal sets obtained by [Zaffalon et al., 2020, algorithm 2], as in [Zaffalon et al., 2020, Example 7]. In the identifiable case, this approach yields equal lower and upper bounds corresponding to the ones computed via do-calculus. If features are only partially identifiable, the algorithm gives intervals narrower than the ones obtained by linear programming and do-calculus, see [Zaffalon et al., 2020, Example 8].

8.4.1 Feasibility Study

Implementations of "ApproxLP" and "CVE" are part of the publicly available java library `CREDICI` (`CREDal Inference for Causal Inference`) [Zaffalon et al., 2020]. The demonstration of CBO attached to [Aglietti et al., 2020], on the other hand, is written in python.[30] This is why we conduct a brief feasibility study. We test the com-

[30] Available on Github.

patibility of the two implementations using `py4j`, a python library that enables python programs to dynamically access arbitrary java objects like the classes `CredalCausalVE` or `CredalCausalAproxLP` from `CREDICI` that conduct the computations of "CVE" and "ApproxLP", respectively. Albeit adding some small computational overhead, computations of lower and upper bounds of $\mathbb{P}(Y|do(X = x))$ inside CBO could be made by `CredalCausalVE` and `CredalCausalAproxLP` in feasible time. The constraints of this thesis did not allow for a more detailed analysis, which is thus left to further research.

8.4.2 Theoretical Considerations

At first sight, causal credal nets seem somewhat counter-intuitive. Why should one be able to specify causal relations in the presence of probabilistic ambiguity? Notably, this approach is diametrically opposed to regular statistical inference, where causal interpretation is normally strictly avoided, while uncertainty is expressed by perfect stochasticity. Yet, we consider the assumption of causality in experimental set-ups (e.g. in material science) to be appropriate. In fact, causality may be an even weaker assumption than a precise probability distribution in many experimental set-ups. For instance, consider the graphene data set in section 6.6. Due to the specialist scientific knowledge gained by experiments, confounding might be excluded. Thus, the assumption that the power of the laser causes the endothermic reaction is indeed very weak. The precise nature of this probabilistic relationship, however, likely remains unknown.

In fact, imprecise causal reasoning comes very close to Pearl's notion of evolutionary advantageous causal reasoning through planning [Pearl and Mackenzie, 2018, Page 26] in human history. Still today, humans evidently base decisions on imprecise causal reasoning. Consider, for example, a hypothetical gardener: she may put a scarecrow in her yard, assuming this will cause the number of birds in her garden to decrease. However, she won't base her decision on a precise estimate of just how many birds the scarecrow will ward off. More likely, she might guess that the scarecrow will at least halve the number of birds; otherwise it would not have been worth purchasing it in the first place. In this case, then, $\mathbb{E}(X|scarecrow) \in [0, n/2]$ or $\underline{\mathbb{E}}(X|scarecrow) = 0$ and $\overline{\mathbb{E}}(X|scarecrow) = n/2$, respectively, with $\mathbb{E}(X|scarecrow)$ being the expected number of birds conditional on the presence of a scarecrow and n the previously observed number of birds. This intuitiveness behind imprecise causal reasoning *per se* does not render the concept a better problem solver. Yet, it might render credal nets more attractive from a practitioner's point of view.

9 Weighted ML Estimation of Prior Parameters

9.1 Motivation

As indicated in section 5.1, maximum likelihood (ML) estimates of prior parameters (definition 16) can be biased. This is due to the fact that the estimation is based on data from the initial design as well as data that were obtained during the optimization procedure. The latter are usually centered around local or global optima, violating the *iid.* assumption, see the example in figure 13 in section 5.1. The later the iteration, the more points in the data set usually come from such regions. This bias in the data translates to the ML estimation, see section 5.1. What is more, the bias affects not only the surrogate model's mean prediction, but also the acquisition function. As was shown in section 2.2, it follows from theorem 1 that using acquisition functions such as lower confidence bound (LCB) or statistical lower bound (SLB) corresponds to implicitly assuming unbiased mean estimation.

We address this issue by a weighting procedure. The general idea is to focus the estimation on informative points and ignore such ones that result from exploitative proposals and provide almost no new information about the true underlying function but serve a mere optimizational purpose.

Luckily, a measure for the degree of information a proposed point carries (given the SM) already exists: The estimated variance $\mathrm{Var}(\boldsymbol{x}) = \mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))$ or standard error $\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}$, respectively. The higher $\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}^*))}$, the more likely the BO explores unknown territory by proposing $\boldsymbol{x}^*$. As a consequence, the greater the reduction of uncertainty and thus the gain in information will be.

However, this absolute value alone strongly depends on the iteration. The overall (classical data) uncertainty of the SM is decreasing monotonously with the iterations. We need to compare it to $\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}))}$ of other $\boldsymbol{x}$ of the same iteration. One option is to use points that *could* have been proposed in this iteration, namely those ones that were evaluated during infill optimization. They are called candidate points henceforth. We thus compute the ratio of the proposed point's standard error $\hat{s}(\mathbf{x}^*)$ to the mean standard error of n other candidate points (standard error ratio (SER)) and its location in the distribution of candidate points' standard errors (standard error distribution value (SED)):

Definition 24 (Standard Error Ratio and Standard Error Distribution Value) *Let $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$ be the candidate points of a given iteration. Furthermore, $\hat{s}(\boldsymbol{x}) = \sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}))}$ for notational simplicity. Let $\boldsymbol{x}^* \in \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ be the proposed point of that iteration and $F_{\hat{s}(\boldsymbol{x})}(\bullet)$ the empirical distribution function of $\hat{s}(\boldsymbol{x}_1), ..., \hat{s}(\boldsymbol{x}_n)$.*

$$SER(\boldsymbol{x}^*) = \frac{\hat{s}(\boldsymbol{x}^*)}{\frac{1}{n}\sum_{i=1}^{n} \hat{s}(\boldsymbol{x}_i)}, \quad SED(\boldsymbol{x}^*) = F_{\hat{s}(\boldsymbol{x})}(\hat{s}(\boldsymbol{x}^*)).$$

Note that definition 24 depends on the infill optimizer. When focus search is used, where the sampling space is iteratively shrunken, the proposed point's standard error is not compared to the overall uncertainty of the SM, but rather to the SM in those regions that were visited extensively during infill optimization. Another way to compute SER and SED would be to compare the proposed points' standard errors[31] to those of *post hoc* randomly sampled points from the parameter space. This would allow to better understand the influence of different infill optimizers on the explore-exploit-behavior. While definition 24 is well-suited for comparing different AF, the *post hoc* approach is more global and allows analyzing the effect of both the infill criteria and the infill optimizer. Henceforth, we will use $SER(\boldsymbol{x}^*)$ or $SED(\boldsymbol{x}^*)$ as defined in definition 24, but with random search as infill optimizer, such that the definition *de facto* corresponds to random *post hoc* sampling.

To sum it up, comparing $SER(\boldsymbol{x}^*)$ or $SED(\boldsymbol{x}^*)$ of different iterations allows assessing the BO's explore-exploit behavior and the effect of the AF on it. From an information theoretical point of view, $\sqrt{\text{Var}(\widehat{\mu}(\boldsymbol{x}^*))}$ or $SER(\boldsymbol{x}^*)$ and $SED(\boldsymbol{x}^*)$ express the entropy of a data point, that is the potential gain of information when Ψ is evaluated at that point or, in other words, the (classical data driven) uncertainty reduction. As this uncertainty is associated with sparse data, $SER(\boldsymbol{x}^*)$ or $SED(\boldsymbol{x}^*)$ express the degree of solitude of the data point.[32] Leaning on the rationale behind sampling methods like latin hypercube sampling (LHS) and maximin-LHS[33], a set of such "lonely" points appears to be a reasonable approximation of an *iid.* sample.

[31] In the surrogate model of the respective iteration.

[32] Analyses in the consulting project "Interpretable Bayesian Optimization" have shown that $SER(\boldsymbol{x}^*)$ and $SED(\boldsymbol{x}^*)$ strongly correlate with minimal and mean distance of $\boldsymbol{x}^*$ to the other points.

[33] Points are sampled such that they maximize the minimal distance to already sampled points.

9.2 Weighted Maximum Likelihood

This exact property will be used to construct a weighted maximum likelihood estimator of GP's prior parameters. Definition 25 directly follows from definition 24 and the weight property that weights should sum up to 1.

Definition 25 (Importance Weights)
The importance weight w_j *of* $\boldsymbol{x}_j \in \{\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_{init}}, \boldsymbol{x}_{n_{init}+1}, ..., \boldsymbol{x}_{n_{init}+t}\}$ *is*

$$w_j = \frac{SED(\boldsymbol{x}_j)}{\sum_i^{n_{init}+t} SED(\boldsymbol{x}_i)}.$$

The question remains how to assign these weights to the observed data or their respective likelihood contribution. [Karampatziakis and Langford, 2011] discuss various ways of assigning importance weights to data points for optimization purposes. [Field and Smith, 1994] propose weighting points on the basis of the natural probabilistic scale, whose implementation appears beyond the scope of this thesis. Far more intuitive, [Wang, 2001] simply exponentiate the likelihood contributions by the according weights, see definition 26 for details.

Definition 26 (Weighted ML Estimation of Prior Parameters)
Following definition 16 (ML estimation) let $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_{init}}, \boldsymbol{x}_{n_{init}+1}, ..., \boldsymbol{x}_{n_{init}+t}\}$ *be the observed covariates and the respective function values* $\{\Psi(\boldsymbol{x}_1), ..., \Psi(\boldsymbol{x}_{n_{init}+1}), \Psi(\boldsymbol{x}_1), ..., \Psi(\boldsymbol{x}_{n_{init}+t})\}$ *in iteration* t*, both subsumed as design matrix* $\boldsymbol{X}_t$*.* f *is a normal density function and* $\{w_1, ..., w_{n_{init}+1}\}$ *are the weights as defined above. The weighted likelihood contribution of* $\boldsymbol{x}_j$ *according to [Wang, 2001] then is* $f(\boldsymbol{\theta}|\boldsymbol{x}_j, \Psi(\boldsymbol{x}_j))^{w_j}$ *and the weighted likelihood function becomes* $L(\boldsymbol{\theta}|\boldsymbol{X}_t)_w = \prod_{i=1}^n f(\boldsymbol{\theta}|\boldsymbol{x}_i, \Psi(\boldsymbol{x}_i))^{w_i}$*. Analogous to definition 16, the weighted maximum likelihood estimator for* $\boldsymbol{\theta}$ *is*

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_{WML}(\boldsymbol{X}_t) &= \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}|\boldsymbol{X}_t)_w \\
&= \arg\max_{\boldsymbol{\theta}} \prod_{i=1}^{n} f(\boldsymbol{\theta}|\boldsymbol{x}_i, \Psi(\boldsymbol{x}_i))^{w_i} \\
&= \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{n} w_i \cdot log(f(\boldsymbol{\theta}|\boldsymbol{x}_i, \Psi(\boldsymbol{x}_i))).
\end{aligned}$$

Since the latter's implementation as a fork of `DiceKriging` turned out to be computationally expensive, we restrict ourselves to a resampling approach.

Algorithm 5 Importance Resampling for ML Estimation

1: Let again $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_{init}}, \boldsymbol{x}_{n_{init}+1}, ..., \boldsymbol{x}_{n_{init}+t}\}$ be the observed covariates and the respective function values $\{\Psi(\boldsymbol{x}_1), ..., \Psi(\boldsymbol{x}_{n_{init}+1}), \Psi(\boldsymbol{x}_1), ..., \Psi(\boldsymbol{x}_{n_{init}+t})\}$ in BO iteration t, each $\boldsymbol{x}_j$ with corresponding weight $w_j \in \{w_1, ..., w_{n_{init}+t}\}$ (definition 25).
2: **for** $n_{init} + t$ Iterations **do**
3: **sample** (with replacement) $\boldsymbol{x}_j$ from $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_{init}+t}\}$ with probability w_j
4: **end for**
5: **return** $\hat{\boldsymbol{\theta}}_{ML}(\boldsymbol{X}_t) = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}|\boldsymbol{X}_t)$, where $\boldsymbol{X}_t$ is the design matrix of the sampled $\boldsymbol{x}_j$ and their respective $\Psi(\boldsymbol{x}_j)$

Ideally, this approach would push the likelihood contributions of uninformative points from thoroughly exploited regions towards 0. Points from exploratory proposals would have a relatively stronger impact. As a consequence, the deviation from the *iid.* assumption and thus the bias would be reduced.

9.3 Experiments

In a first experiment, we compare the weighted ML estimation in BO on a set of six synthetic test functions to unweighted ML, see figure 24. Algorithm 5 was implemented inside the `R` package `DiceKriging` that is in turn used inside `mlrMBO` for GP computations. SER and SED were computed by independent functions that build on `mlrMBO`, some of which have already been used in the statistical consulting project "Interpretable Bayesian Optimization". Expected improvement was used as acquisition function and random search as infill optimizer with sample size 1000. Bayesian optimizations with both weighted and unweighted ML estimates were run $n = 200$ times with 40 iterations and initial design of size 4 generated by maximin-LHS each.

Based on the results of the first experiment, we further investigate BO with weighted ML and take a closer look on its optimization of the Mexican hat function. We do this by changing the AF from expected improvement to lower confidence bound in the second experiment because the latter allows explicit guidance of the explore-exploit trade-off through varying τ. This way, we aim at assessing the impact of different exploration-exploitation-strategies. Thirdly, we choose smaller values for τ in the LCB, to take a closer look at very exploitative BO behavior. Both experiments were computed on a high performance computing cluster using 20 64-bit-cores (linux gnu). The experiments

were conducted in `R` version 4.0.3 [R Core Team, 2020]. While implementing the resampling approach by means of modifying functions of `DiceKriging`, we ran into some numerical trouble with matrix multiplications, which turned out to be connected to an already familiar problem of `DiceKriging`.[34] Iterations of the experiments, in which the respective error occurred were disregarded in the analysis, reducing the sample size of $n = 200$ slightly.

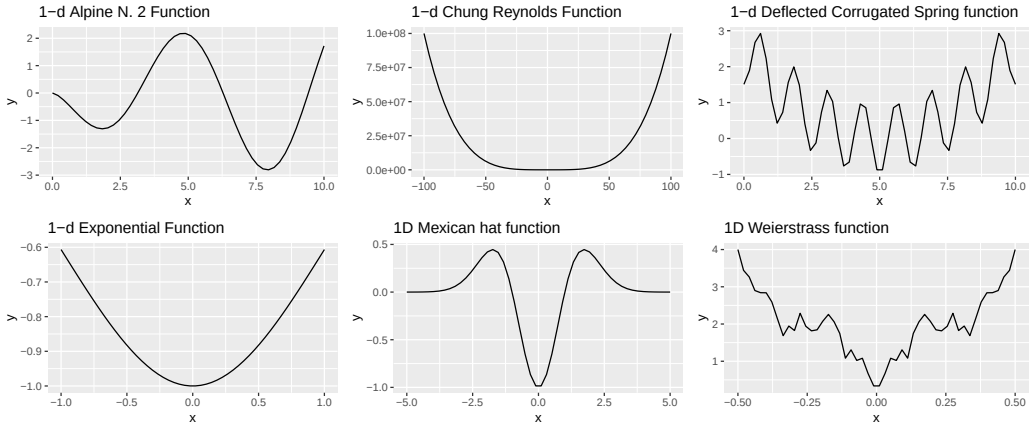

Figure 24: Synthetic test functions from package `smoof` [Bossek, 2017] that weighted ML was benchmarked on.

9.4 Results

As can be seen in figure 25, where the first experiment's results are shown, BO with weighted ML (blue) is inferior to classical unweighted ML (magenta) on the deflected corrugates spring function, the exponential function and the Weierstrass function. Only for the latter one, 0.95% confidence intervals do not overlap. Weighted ML and unweighted ML have similar mean optimization paths in case of the second alpine function and the Chung Reynolds function. It is only the Mexican hat function, where weighted ML outperforms unweighted ML albeit overlapping confidence intervals.

[34] See this issue on GitHub.

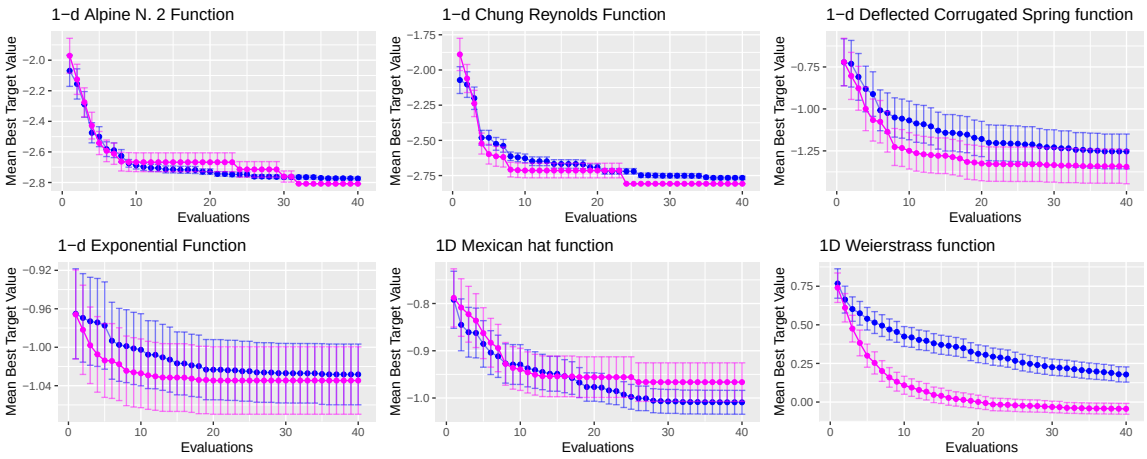

Figure 25: Benchmarking of BO with weighted ML (blue) against classic unweighted ML (magenta) on six synthetic functions. Error bars depict 0.95 % confidence intervals.

As presumed, the degree of exploration/exploitation has a strong impact on the performance of weighted ML as compared to unweighted ML inside BO. This becomes evident when assessing figure 27, which depicts the second experiment's results. For very exploratory settings (roughly $\tau \geq 5$), BO with unweighted ML clearly converges faster than BO with weighted ML. This relationship is starting to be turned upside down in case of rather exploitative BOs, as can be seen in figure 27 for $\tau \in \{1, 2\}$

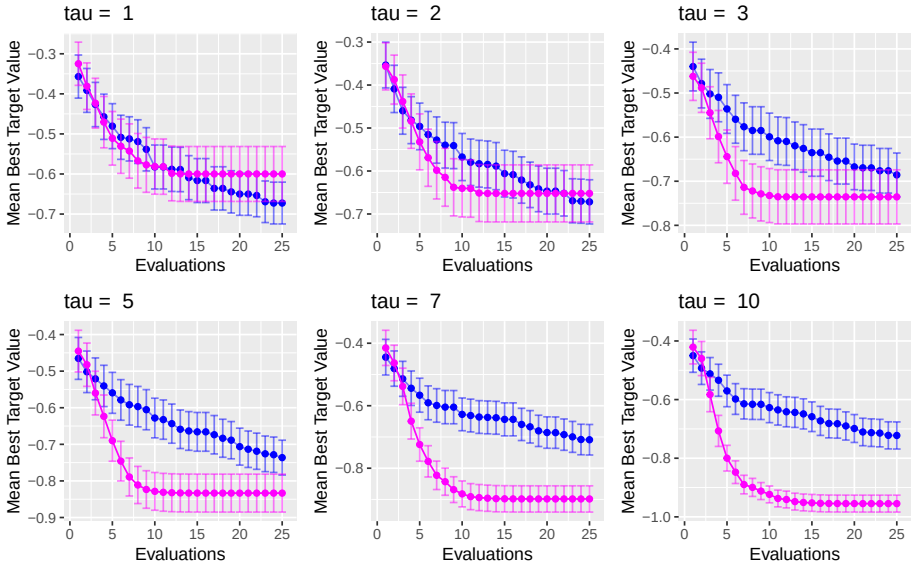

Figure 26: Benchmarking of BO with weighted ML (blue) against classic unweighted ML (magenta) on Mexican hat function with varying τ in LCB. Error bars depict 0.95 % confidence intervals.

Zooming in further reveals that for very exploitative settings ($\tau \leq 1$), BO with weighted ML clearly outperforms BO with unweighted ML. Figure 27 depicts mean optimization

paths and confidence intervals for LCB with $\tau \leq 0$. We observe how decreasing τ entails a slowdown, if not stagnation of convergence for BO with prior parameters estimated by unweighted ML. On the other hand, BO with weighted ML only slightly forfeits convergence speed. Put another way, BO with weighted ML appears to be more robust towards declines in τ, i.e. more exploitative settings. Notably, this is not observed in case of optimizing the second alpine function, see figure 52 in the appendix.

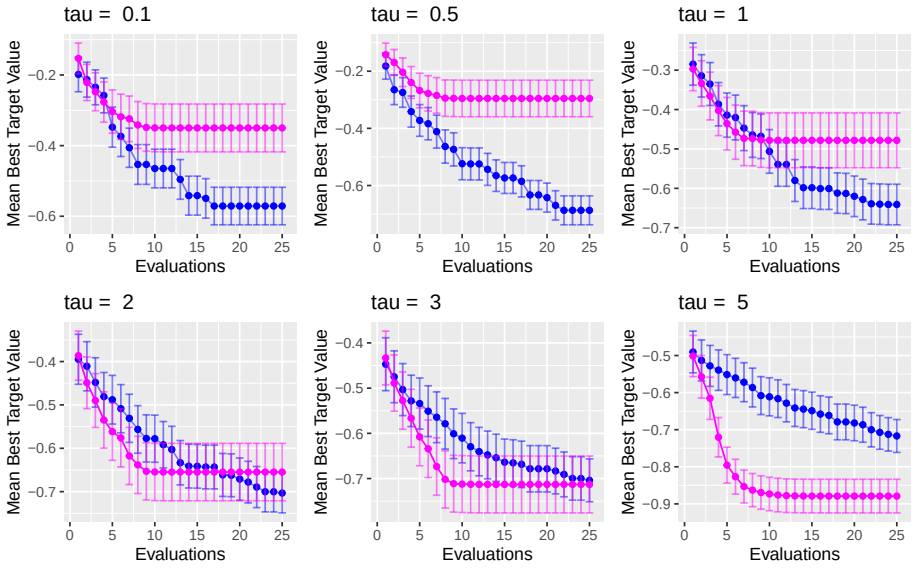

Figure 27: Benchmarking of BO with weighted ML (blue) against classic unweighted ML (magenta) on Mexican hat function with varying τ in LCB. Error bars depict 0.95 % confidence intervals.

9.5 Discussion

The slowdown in convergence of BO with exploitative AFs on the Mexican hat function can be easily explained by the shape of the function, as seen in figure 24: Some of the optimization rounds simply get stuck in the local minima at $x \in \{-5, 5\}$. Only such rounds with fortunate initial design exploit the global minima at $x = 0$. Generally, this applies to BOs with both weighted and unweighted ML prior parameter estimates. However, it happens significantly ($\alpha = 0.05$) less frequently in cases of weighted ML.

Our explanation, which of course needs further critical assessment, goes as follows: The predictive posterior GP is generally dominated by the likelihood in areas of thorough exploitation and by the prior in areas in which less proposals were made. For optimization rounds that exploit the two local minima at $x \in \{-5, 5\}$, then, the predictions in the area around the global optimum are dominated by the prior. If prior parameters are estimated from the data, the resulting predictions are biased towards the values

of the well-exploited local minima. This makes casual exploration of these areas even more unlikely. Weighted ML estimation of prior parameters mitigates this effect by diminishing the likelihood contribution of points from the local minima. Hence, BO with weighted ML is more likely to jump into these regions from time to time, and gets stuck in local optima less often. In the case of the second alpine function, see figure 52 in the appendix, the relative (that is, on a standardized scale) difference of function values from local and global minima is smaller. This is why – we presume – the bias is not as severe as in case of the Mexican hat function and as a result, the weighted ML procedure does not entail a similar improvement.

To sum it up, for (classical uncertainty related) risk-averse subjects who aim at exploitation rather than exploration, weighted ML has proven to be a better alternative than (standard) unweighted ML estimation on selected test problems. As with prior-mean-robust BO (section 6), the exact circumstances (i.e. the determinants of target functions such as the Mexican hat function) under which the method is beneficial need further assessment. As a starting point, the above described preliminary hypothesis that the difference between local and global optima determines the bias and thus the potential improvement through weighted ML can be more closely examined. Another hypothesis would be that BO does in fact exploit the global minimum, but occasionally jumps to local minima due to a posterior underestimation of these regions caused by priors estimated from a sample that is biased towards the global minimum. The mean optimization path's literal gridlock, however, contradicts this hypothesis. Such occasional jumps would slow down, but still allow global convergence.

Let us now turn to the sobering results from the other five test functions that show no improvement or even change for the worse in the case of weighted ML estimation as opposed to classical unweighted ML estimation of prior parameters. Apparently, an unbiased estimate of prior parameters is of no optimizational help in these cases. Simply put, at the end of the day BO aims at optimization instead of statistical modeling of the unknown underlying function. Nevertheless, this very behavior appears more coherent through the eyes of a statistician: the (globally) biased unweighted ML estimates are in fact unbiased on a local scale around the promising area where they were sampled from. A sound local approximation of the target function appears beneficial here, while a good global approximation might simply not serve the optimization's aim. Giving up unpromising regions early by both not proposing points within them and not aiming at a good estimate of the function in these regions might speed up convergence in such cases.

9.6 Limitations

Apart from its apparently narrow scope of fruitful applications (see above), there are only few, if any limitations to weighting the likelihood contributions of proposed points in ML estimates during BO. The computational costs of resampling appear diminutive to the human eye. Yet, they should be analyzed in a systematical manner by standard profiling techniques. Furthermore, the computational issues with matrix multiplications in `DiceKriging`, as mentioned in section 9.3, could be analyzed more closely.

9.7 Extensions

As stated in section 9.1, $\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}^*))}$ or $SER(\boldsymbol{x}^*)$ and $SED(\boldsymbol{x}^*)$ express the entropy of a data point (given a SM), that is the potential gain of information when Ψ is evaluated at that point with regard to the (classical data driven) uncertainty. Leaning on section 6, analogous considerations lead to potential information gain with regard to model imprecision, that is reduction in ambiguity. As in the generalized lower confidence bound (GLCB) from definition 23, such ambiguity can be expressed by $\overline{\mu}(\boldsymbol{x})_c - \underline{\mu}(\boldsymbol{x})_c$. The above described weighting procedure can be deployed without need for adaption. Besides that, a combination of uncertainty and imprecision weights might entail a comprehensive account for ignorance in the ML estimation.

However audacious such a combined weighting procedure may sound, it should be cautiously analyzed. The main motivation behind weighted ML is to reduce a bias in the data induced by optimization proposals rather than in the model. Nonetheless, recall that $\sqrt{\mathrm{Var}(\widehat{\mu}(\boldsymbol{x}^*))}$ or $SER(\boldsymbol{x}^*)$ and $SED(\boldsymbol{x}^*)$ measure the entropy of $\boldsymbol{x}^*$ *conditioned* on the SM. Accounting for the model imprecision might entail loosening this condition.

10 Other Stochastic Derivative-free Optimizers

Besides improving prior–mean-robust BO by means of the extensions presented above, it may be attractive to further explore possibilities of generalizing other derivative-free optimization methods with stochastic and probabilistic elements. Henceforth, evident and naive ways to generalize them are outlined. What is more, simple ways to assess their robustness towards probabilistic modifications are discussed. Two such approaches are briefly introduced in what follows. Its implementation and thorough assessment are left to future research.

10.1 Simulated Annealing

Inspired by cooling-down processes of metals and liquids, simulated annealing [Kirkpatrick et al., 1983] is a local search that uniformly samples from a hypercube or an ϵ-ball around the incumbent optimal value $\boldsymbol{x}_i$ (exploitation) and casually accepts parameters from the rejection area $\{\boldsymbol{x}_{i+1} : f(\boldsymbol{x}_{i+1}) - f(\boldsymbol{x}_i) > 0\}$ (exploration). It uses the co-called Metropolis criterion.

Definition 27 (Metropolis Criterion)
Accept parameter $\boldsymbol{x}_{i+1}$ from rejection area $\{\boldsymbol{x}_{i+1} : f(\boldsymbol{x}_{i+1}) - f(\boldsymbol{x}_i) > 0\}$ with

$$\mathbb{P}(\textit{accept}) = \exp\left(-\left(\frac{f(\boldsymbol{x}_{i+1}) - f(\boldsymbol{x}_i)}{T}\right)\right),$$

where T is the temperature of the system, which monotonically decreases with the iterations. $\mathbb{P}$ induces an exponential distribution with $\lambda = 1$.

Analogous to section 4, a sensitivity analysis may be conducted by deploying different distributions from the same distributional family (i.e. varying λ) in the metropolis criterion. Their effect on the optimization path could be assessed. Furthermore, the effect of the uniform sampling could be taken into account by analyzing the interaction of the two probability measures (uniform distribution and exponential distribution) through simulation studies similar to the ones in section 4. Once sufficiently studied, the variation of optimization paths (definition 13) induced by altering these distributional assumption could be taken into account by the optimizer. For instance, by a slowdown of the cooling process that accounts for additional exploration of areas with high model imprecision.

10.2 Evolutionary Algorithms

An evolutionary algorithm (EA) optimizes black box functions by mimicking the evolution of animal populations through natural selection. A crucial part of EA is the mutation operator, as it ensures diversity in following generations [Deb and Deb, 2014]. As in nature, mutation is assumed to be random. Besides uniformly sampling vector $\boldsymbol{x}_{i+1}$ from the parameter space, the Gauss-mutation has gained popularity: $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i \pm \boldsymbol{m}$, $\boldsymbol{m} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$. Following the same pattern as above, the effect of altering σ on the the optimization paths can be investigated. What is more, Gaussian mixture distributions might be tried out. Potentially, robust extension of the Gauss-mutation that account for model imprecision can be suggested.

Instead of deploying such mixture models, one could also sample (batch-wise) from a set of (normal) distributions, loosely leaning on ϵ-contamination-models [Chen et al., 2016] or neighborhood models, see [Huber and Strassen, 1973]. The resulting set of populations could then be ordered by a fitness function. The induced ordering can itself be imprecise, as proposed by [Abrams, 2019, Chapter 5.1] as "imprecise fitness comparison" in his insightful work on natural selection through the lens of imprecise probabilities.

In both ways it might be possible to explicitly account for model imprecision in evolutionary algorithms by including a second exploration rationale. It aims at the reduction of such imprecision as opposed to the classical exploration that is concerned with reduction of uncertainty induced by data.

This could not only increase the performance of such evolutionary-inspired optimizers, but also the accuracy of evolutionary models in biology. Here, the exploration-exploitation trade-off is a well understood phenomenon, see [Zrzavỳ et al., 2009] for instance. A popular example is the distinction between antigenic drifts (incremental change) and shifts (new influenza subtype) in influenza A viruses [Ferguson et al., 2003]. We suggest to subdivide the explorational behavior of self-conscious animals such as humans into two parts. The first one aims at acquiring data (information) about the physical surroundings of a species (uncertainty reduction). The second one intends the falsification of the species' inherent assumptions about the world such as genetic predispositions or cultural imprints (imprecision reduction). Or, in other words, it is concerned with data-free assertions that the individual has either required through inheritance or education ("nature or nurture").

11 Conclusion

In this work, we have proposed four modifications of Bayesian optimization that account for the GP's model imprecision. The latter's general effect on BO's behavior was studied in section 4. We observed a particularly strong impact of the prior's mean parameter(s); this is why our four proposals are concerned with this part of the prior specification. While the proposed parallel hedging BO (section 6.3) and batch-wise speed-up prior-mean-robust BO (section 6.4) did not accomplish their aims in empirical studies, the remaining two proposals achieved promising results. Generalized lower confidence bound (section 6.5) converged faster than or as fast as several classic AFs on real-world data sets. Weighted maximum likelihood estimation (section 9) improved BO's performance on a specific type of target function given an exploitative AF.

By and large, these results unveil the potential of accounting for model imprecision not only in BO but in stochastic derivative-free optimization methods in general. In light of this consideration, we outlined two venues for future work on imprecision in such optimizers (section 10), one of which entails a conceptual modification of explore-exploit-models in evolutionary biology. On top of that, the seemingly audacious idea of integrating model imprecision as defined in the GLCB into the weighted ML estimation was briefly touched upon (section 9.7). However, we suggest further exploitation of the proposed well-defined BO modifications first, before exploring those ideas. As insinuated in sections 6.8 and 9.5, some aspects of GLCB and weighted ML applications remain unclear, e.g. the determinants of target functions on which the methods exceed conventional approaches. A thorough understanding of GLCB and weighted ML methods may help prevent dead ends in implementing the additional ones.

All in all, the thesis at hand has demonstrated that thoroughly accounting for the unknown can increase not only statistical models' credibility [Manski, 2003] but also the performance of optimizers that deploy such models. Furthermore, it presumably reduces the amount of false confidence and "intellectual debt" [Zittrain, 2019] accumulated as a result of apparently-optimal hyperparameter settings in machine learning models. Closely leaning on [Popper, 1992] and [Popper, 1987, chapter 2], humbly accepting the unknown in stochastic derivative-free optimization might elicit scientific advancement – under some specific conditions, that is. To claim more would be presumptuous – and in turn result in false confidence.

12 References

[Abbas et al., 2014] Abbas, M., Ilin, A., Solonen, A., Hakkarainen, J., Oja, E., and Järvinen, H. (2014). Bayesian optimization for tuning chaotic systems. *Nonlinear Processes in Geophysics Discussions*, 1(2):1283–1312.

[Abrams, 2019] Abrams, M. (2019). Natural selection with objective imprecise probability. In *Proceedings of the 11. International Symposium on Imprecise Probabilities: Theories and Applications*, pages 2–13.

[Aglietti et al., 2020] Aglietti, V., Lu, X., Paleyes, A., and González, J. (2020). Causal Bayesian optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 3155–3164. PMLR.

[Augustin et al., 2014] Augustin, T., Coolen, F. P., De Cooman, G., and Troffaes, M. C. (2014). *Introduction to imprecise probabilities*. John Wiley & Sons.

[Awal et al., 2021] Awal, M. A., Masud, M., Hossain, M. S., Bulbul, A. A., Mahmud, S. M. H., and Bairagi, A. K. (2021). A novel Bayesian optimization-based machine learning framework for COVID-19 detection from inpatient facility data. *IEEE Access*, 9:10263–10281.

[Balch et al., 2019] Balch, M. S., Martin, R., and Ferson, S. (2019). Satellite conjunction analysis and the false confidence theorem. *Proceedings of the Royal Society A*, 475(2227):20180565.

[Ben-Tal et al., 2009] Ben-Tal, A., El Ghaoui, L., and Nemirovski, A. (2009). *Robust optimization*, volume 28. Princeton University Press.

[Benassi et al., 2011] Benassi, R., Bect, J., and Vazquez, E. (2011). Robust Gaussian process-based global optimization using a fully Bayesian expected improvement criterion. In *International Conference on Learning and Intelligent Optimization*, pages 176–190. Springer.

[Benavoli et al., 2021] Benavoli, A., Azzimonti, D., and Piga, D. (2021). Preferential Bayesian optimisation with skew Gaussian processes. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 1842–1850.

[Berk et al., 2018] Berk, J., Nguyen, V., Gupta, S., Rana, S., and Venkatesh, S. (2018). Exploration enhanced expected improvement for Bayesian optimization. In *Joint*

European Conference on Machine Learning and Knowledge Discovery in Databases, pages 621–637. Springer.

[Bischl et al., 2017] Bischl, B., Richter, J., Bossek, J., Horn, D., Thomas, J., and Lang, M. (2017). mlrMBO: A modular framework for model-based optimization of expensive black-box functions. *arXiv preprint asXiv:1703.03373*.

[Bischl et al., 2014] Bischl, B., Wessing, S., Bauer, N., Friedrichs, K., and Weihs, C. (2014). Moi-mbo: multiobjective infill for parallel model-based optimization. In *International Conference on Learning and Intelligent Optimization*, pages 173–186. Springer.

[Bossek, 2017] Bossek, J. (2017). smoof: Single- and multi-objective optimization test functions. *The R Journal*.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

[Breiman and Friedman, 1985] Breiman, L. and Friedman, J. H. (1985). Estimating optimal transformations for multiple regression and correlation. *Journal of the American statistical Association*, 80(391):580–598.

[Bull, 2011] Bull, A. D. (2011). Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12(10).

[Canty and Ripley, 2021] Canty, A. and Ripley, B. D. (2021). *boot: Bootstrap R (S-Plus) Functions*. R package version 1.3-28.

[Chen et al., 2016] Chen, M., Gao, C., and Ren, Z. (2016). A general decision theory for Huber's epsilon-contamination model. *Electronic Journal of Statistics*, 10(2):3752–3774.

[Chen et al., 2020] Chen, W., Ishibuhci, H., and Shang, K. (2020). Lazy greedy hypervolume subset selection from large candidate solution sets.

[Chen et al., 2018] Chen, Y., Huang, A., Wang, Z., Antonoglou, I., Schrittwieser, J., Silver, D., and de Freitas, N. (2018). Bayesian optimization in alphago. *arXiv preprint arXiv:1812.06855*.

[Cox and John, 1992] Cox, D. D. and John, S. (1992). A statistical method for global optimization. In *Proceedings of 1992 IEEE International Conference on Systems, Man, and Cybernetics*, pages 1241–1246. IEEE.

[Cozman, 2000] Cozman, F. G. (2000). Credal networks. *Artificial intelligence*, 120(2):199–233.

[Deb and Deb, 2014] Deb, K. and Deb, D. (2014). Analysing mutation schemes for real-parameter genetic algorithms. *International Journal of Artificial Intelligence and Soft Computing*, 4(1):1–28.

[Domingos, 2012] Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87.

[Duvenaud, 2014] Duvenaud, D. (2014). *Automatic model construction with Gaussian processes*. PhD thesis, University of Cambridge.

[Duvenaud et al., 2013] Duvenaud, D., Lloyd, J., Grosse, R., Tenenbaum, J., and Zoubin, G. (2013). Structure discovery in nonparametric regression through compositional kernel search. In Dasgupta, S. and McAllester, D., editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1166–1174, Atlanta, Georgia, USA. PMLR.

[Ferguson et al., 2003] Ferguson, N. M., Galvani, A. P., and Bush, R. M. (2003). Ecological and immunological determinants of influenza evolution. *Nature*, 422(6930):428–433.

[Fernández-Godino et al., 2016] Fernández-Godino, M. G., Park, C., Kim, N.-H., and Haftka, R. T. (2016). Review of multi-fidelity models. *arXiv preprint arXiv:1609.07196*.

[Field and Smith, 1994] Field, C. and Smith, B. (1994). Robust estimation: A weighted maximum likelihood approach. *International Statistical Review/Revue Internationale de Statistique*, pages 405–424.

[Fine, 2008] Fine, G. (2008). Does socrates claim to know that he knows nothing? *Oxford Studies in Ancient Philosophy*, 35:49–88.

[Fischer, 2021] Fischer, M. (2021). On the principal principle and subjective imprecise Bayesianism. Poster presented at the *12th International Symposium on Imprecise Probabilities: Theory and Application* (ISIPTA), Granada, Spain.

[Frazier et al., 2009] Frazier, P., Powell, W., and Dayanik, S. (2009). The knowledge-gradient policy for correlated normal beliefs. *INFORMS journal on Computing*, 21(4):599–613.

[Frazier, 2018] Frazier, P. I. (2018). A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*.

[Frazier and Wang, 2016] Frazier, P. I. and Wang, J. (2016). Bayesian optimization for materials design. In *Information Science for Materials Discovery and Design*, pages 45–75. Springer.

[Garrido-Merchán and Hernández-Lobato, 2020] Garrido-Merchán, E. C. and Hernández-Lobato, D. (2020). Dealing with categorical and integer-valued variables in Bayesian optimization with Gaussian processes. *Neurocomputing*, 380:20–35.

[Goldberger and Rigney, 1991] Goldberger, A. L. and Rigney, D. R. (1991). Nonlinear dynamics at the bedside. In *Theory of heart*, pages 583–605. Springer.

[Handcock and Stein, 1993] Handcock, M. S. and Stein, M. L. (1993). A Bayesian analysis of kriging. *Technometrics*, 35(4):403–410.

[Hawley et al., 2012] Hawley, S. A., Fullerton, M. D., Ross, F. A., Schertzer, J. D., Chevtzoff, C., Walker, K. J., Peggie, M. W., Zibrova, D., Green, K. A., Mustard, K. J., et al. (2012). The ancient drug salicylate directly activates amp-activated protein kinase. *Science*, 336(6083):918–922.

[Hennig and Schuler, 2012] Hennig, P. and Schuler, C. J. (2012). Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13(6).

[Horn and Bischl, 2016] Horn, D. and Bischl, B. (2016). Multi-objective parameter configuration of machine learning algorithms using model-based optimization. In *2016 IEEE symposium series on computational intelligence (SSCI)*, pages 1–8. IEEE.

[Huang et al., 2006] Huang, D., Allen, T. T., Notz, W. I., and Zeng, N. (2006). Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of global optimization*, 34(3):441–466.

[Huber and Strassen, 1973] Huber, P. J. and Strassen, V. (1973). Minimax tests and the neyman-pearson lemma for capacities. *The Annals of Statistics*, pages 251–263.

[Hutter, 2009] Hutter, F. (2009). *Automated configuration of algorithms for solving hard computational problems*. PhD thesis, University of British Columbia.

[Hutter et al., 2018] Hutter, F., Kotthoff, L., and Vanschoren, J. (2018). *Automated Machine Learning: Methods, Systems, Challenges*. Springer. In press, available at http://automl.org/book.

[Jamieson and Talwalkar, 2016] Jamieson, K. and Talwalkar, A. (2016). Non-stochastic best arm identification and hyperparameter optimization. In *Artificial Intelligence and Statistics*, pages 240–248. PMLR.

[Jones et al., 1998] Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492.

[Karampatziakis and Langford, 2011] Karampatziakis, N. and Langford, J. (2011). Online importance weight aware updates. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 392–399.

[Kirkpatrick et al., 1983] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

[Kirschner et al., 2020] Kirschner, J., Bogunovic, I., Jegelka, S., and Krause, A. (2020). Distributionally robust Bayesian optimization. In Chiappa, S. and Calandra, R., editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2174–2184. PMLR.

[Klein et al., 2017] Klein, A., Falkner, S., Mansur, N., and Hutter, F. (2017). Robo: A flexible and robust Bayesian optimization framework in python. In *NIPS 2017 Bayesian optimization workshop*.

[Kotthoff, 2019] Kotthoff, L. (2019). AI for materials science: Tuning laser-induced graphene production and beyond. *openreview.net*.

[Kushner, 1962] Kushner, H. J. (1962). A versatile stochastic model of a function of unknown and time varying form. *Journal of Mathematical Analysis and Applications*, 5(1):150–167.

[Liu et al., 2012] Liu, B., Zhang, Q., Fernández, F. V., and Gielen, G. (2012). Self-adaptive lower confidence bound: A new general and effective prescreening method for Gaussian process surrogate model assisted evolutionary algorithms. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–6. IEEE.

[Lopreato, 1973] Lopreato, J. (1973). Notes on the work of Vilfredo Pareto. *Social Science Quarterly*, pages 451–468.

[Makarova et al., 2021] Makarova, A., Shen, H., Perrone, V., Klein, A., Faddoul, J. B., Krause, A., Seeger, M., and Archambeau, C. (2021). Overfitting in bayesian Optimization: an empirical study and early-stopping solution. *arXiv preprint arXiv:2104.08166*.

[Malkomes and Garnett, 2018] Malkomes, G. and Garnett, R. (2018). Automating Bayesian optimization with Bayesian optimization. *Advances in Neural Information Processing Systems*, 31:5984–5994.

[Malkomes et al., 2016] Malkomes, G., Schaff, C., and Garnett, R. (2016). Bayesian optimization for automated model selection. In *Workshop on Automatic Machine Learning*, pages 41–47. PMLR.

[Mangili, 2015] Mangili, F. (2015). A prior near-ignorance Gaussian process model for nonparametric regression. In *ISIPTA '15: Proceedings of the 9th International Symposium on Imprecise Probability: Theories and Applications*, pages 187–196.

[Mangili, 2016] Mangili, F. (2016). A prior near-ignorance Gaussian process model for nonparametric regression. *International Journal of Approximate Reasoning*, 78:153–171.

[Manski, 2003] Manski, C. F. (2003). *Partial identification of probability distributions*. Springer Science & Business Media.

[Mebane, Jr. and Sekhon, 2011] Mebane, Jr., W. R. and Sekhon, J. S. (2011). Genetic optimization using derivatives: The rgenoud package for R. *Journal of Statistical Software*, 42(11):1–26.

[Mersmann et al., 2020] Mersmann, O., Bischl, B., Bossek, J., and Judt, L. (2020). *soobench: Single Objective Optimization Benchmark Functions*. R package version 1.9.18.

[Močkus, 1975] Močkus, J. (1975). On Bayesian methods for seeking the extremum. In *Optimization techniques IFIP technical conference*, pages 400–404. Springer.

[Močkus, 1989] Močkus, J. (1989). *The Bayesian approach to local optimization - theory and applications*. Kluwer Acad. Publ.

[Mockus et al., 1978] Mockus, J., Tiesis, V., and Zilinskas, A. (1978). The application of Bayesian methods for seeking the extremum. *Towards global optimization*, 2(117-129):2.

[Moriconi et al., 2020] Moriconi, R., Kumar, K. S., and Deisenroth, M. P. (2020). High-dimensional Bayesian optimization with projections using quantile Gaussian processes. *Optimization Letters*, 14(1):51–64.

[Nalenz and Augustin, 2019] Nalenz, M. and Augustin, T. (2019). Characterizing uncertainty in decision trees through imprecise splitting rules. Poster presented at ISIPTA '19: International Symposium on Imprecise Probability: Theories and Applications.

[Nguyen et al., 2020] Nguyen, T., Gupta, S., Ha, H., Rana, S., and Venkatesh, S. (2020). Distributionally robust Bayesian quadrature optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 1921–1931. PMLR.

[Nguyen, 2019] Nguyen, V. (2019). Bayesian optimization for accelerating hyperparameter tuning. In *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 302–305. IEEE.

[Norton, 2008] Norton, J. D. (2008). Ignorance and indifference. *Philosophy of Science*, 75(1):45–68.

[Pearl et al., 2016] Pearl, J., Glymour, M., and Jewell, N. P. (2016). *Causal inference in statistics: A primer*. John Wiley & Sons.

[Pearl and Mackenzie, 2018] Pearl, J. and Mackenzie, D. (2018). *The Book of Why*. Penguin Random House.

[Peters et al., 2017] Peters, J., Janzing, D., and Schölkopf, B. (2017). *Elements of causal inference: foundations and learning algorithms*. The MIT Press.

[Picheny et al., 2010] Picheny, V., Ginsbourger, D., and Richet, Y. (2010). Noisy expected improvement and on-line computation time allocation for the optimization of simulators with tunable fidelity.

[Popper, 2014] Popper, K. (2014). *Conjectures and refutations: The growth of scientific knowledge*. routledge.

[Popper et al., 2013] Popper, K., Mejer, J., and Petersen, A. (2013). *The world of Parmenides: essays on the Presocratic enlightenment*. Routledge.

[Popper, 1987] Popper, K. R. (1987). *Auf der Suche nach einer besseren Welt: Vorträge und Aufsätze aus dreißig Jahren*. Piper München.

[Popper, 1991] Popper, K. R. (1991). *" Ich weiß, daß ich nichts weiß-und kaum das": Karl Popper im Gespräch über Politik, Physik und Philosophie*. Number 34833. Ullstein.

[Popper, 1992] Popper, K. R. (1992). *The logic of scientific discovery*. Routledge.

[Pyzer-Knapp, 2018] Pyzer-Knapp, E. O. (2018). Bayesian optimization for accelerated drug discovery. *IBM Journal of Research and Development*, 62(6):2–1.

[R Core Team, 2020] R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

[Rahat, 2019] Rahat, A. A. (2019). Greed is good: Exploration and exploitation trade-offs in Bayesian optimisation. *ACM Transactions on Evolutionary Learning and Optimization*.

[Rasmussen, 2003] Rasmussen, C. E. (2003). Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer.

[Roustant et al., 2012] Roustant, O., Ginsbourger, D., and Deville, Y. (2012). DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization. *Journal of Statistical Software*, 51(1):1–55.

[Rüger, 2010] Rüger, B. (2010). *Test-und Schätztheorie: Band I: Grundlagen*. Walter de Gruyter.

[Schmidt et al., 2008] Schmidt, A. M., Conceição, M. d. F. d. G., and Moreira, G. A. (2008). Investigating the sensitivity of Gaussian processes to the choice of their correlation function and prior specifications. *Journal of Statistical Computation and Simulation*, 78(8):681–699.

[Shah et al., 2014] Shah, A., Wilson, A., and Ghahramani, Z. (2014). Student-t processes as alternatives to Gaussian processes. In *Artificial intelligence and statistics*, pages 877–885. PMLR.

[Shahriari et al., 2015] Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N. (2015). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175.

[Shi et al., 2013] Shi, Z., Church, R. M., and Meck, W. H. (2013). Bayesian optimization of time perception. *Trends in cognitive sciences*, 17(11):556–564.

[Snoek et al., 2012] Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25:2951–2959.

[Sun et al., 2021] Sun, G., Li, L., Fang, J., and Li, Q. (2021). On lower confidence bound improvement matrix-based approaches for multiobjective bayesian optimization and its applications to thin-walled structures. *Thin-Walled Structures*, 161:107248.

[Swersky et al., 2014] Swersky, K., Duvenaud, D., Snoek, J., Hutter, F., and Osborne, M. A. (2014). Raiders of the lost architecture: Kernels for Bayesian optimization in conditional parameter spaces. *arXiv preprint arXiv:1409.4011*.

[Thompson, 1933] Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294.

[Utkin et al., 2019] Utkin, L., Kovalev, M., Meldo, A., and Coolen, F. (2019). Imprecise extensions of random forests and random survival forests. In Bock, J. D., de Campos, C. P., de Cooman, G., Quaeghebeur, E., and Wheeler, G., editors, *Proceedings of the Eleventh International Symposium on Imprecise Probabilities: Theories and Applications*, volume 103 of *Proceedings of Machine Learning Research*, pages 404–413, Thagaste, Ghent, Belgium. PMLR.

[Vane, 1971] Vane, J. R. (1971). Inhibition of prostaglandin synthesis as a mechanism of action for aspirin-like drugs. *Nature new biology*, 231(25):232–235.

[Vazquez and Bect, 2010] Vazquez, E. and Bect, J. (2010). Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and inference*, 140(11):3088–3095.

[Wackerly et al., 2014] Wackerly, D., Mendenhall, W., and Scheaffer, R. L. (2014). *Mathematical statistics with applications*. Cengage Learning.

[Wahab et al., 2020] Wahab, H., Jain, V., Tyrrell, A. S., Seas, M. A., Kotthoff, L., and Johnson, P. A. (2020). Machine-learning-assisted fabrication: Bayesian optimization of laser-induced graphene patterning using in-situ raman analysis. *Carbon*, 167:609–619.

[Wang, 2001] Wang, S. X. (2001). *Maximum weighted likelihood estimation*. PhD thesis, University of British Columbia.

[Wickham, 2016] Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.

[Wickham et al., 2019] Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., and Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43):1686.

[Wolpert and Macready, 1997] Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

[Wu and Frazier, 2019] Wu, J. and Frazier, P. (2019). Practical two-step lookahead Bayesian optimization. *Advances in neural information processing systems*, 32:9813–9823.

[Yue and Kontar, 2020] Yue, X. and Kontar, R. A. (2020). Why non-myopic Bayesian optimization is promising and how far should we look-ahead? a study via rollout. In Chiappa, S. and Calandra, R., editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2808–2818. PMLR.

[Zaffalon et al., 2020] Zaffalon, M., Antonucci, A., and Cabañas, R. (2020). Structural causal models are (solvable by) credal networks. *arXiv preprint arXiv:2008.00463*.

[Zittrain, 2019] Zittrain, J. (2019). The hidden costs of automated thinking. *The New Yorker*.

[Zrzavỳ et al., 2009] Zrzavỳ, J., Storch, D., and Mihulka, S. (2009). *Evolution: Ein Lese-Lehrbuch*. Springer-Verlag.

A Appendix

A.1 Additional Proofs

Proof 2 (Proof of Bias-Variance-Decomposition of MSE)
Let θ be the parameter of interest, $\hat{\theta}$ its estimator.

$$
\begin{aligned}
\mathrm{MSE}(\hat{\theta}) &= \mathbb{E}_\theta\left[(\hat{\theta}-\theta)^2\right] \\
&= \mathbb{E}_\theta\left[\left(\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]+\mathbb{E}_\theta[\hat{\theta}]-\theta\right)^2\right] \\
&= \mathbb{E}_\theta\left[\left(\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]\right)^2+2\left(\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]\right)\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)+\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)^2\right] \\
&= \mathbb{E}_\theta\left[\left(\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]\right)^2\right]+\mathbb{E}_\theta\left[2\left(\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]\right)\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)\right]+\mathbb{E}_\theta\left[\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)^2\right] \\
&= \mathbb{E}_\theta\left[\left(\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]\right)^2\right]+2\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)\mathbb{E}_\theta\left[\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]\right]+\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)^2 \\
&= \mathbb{E}_\theta\left[\left(\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]\right)^2\right]+2\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)\left(\mathbb{E}_\theta[\hat{\theta}]-\mathbb{E}_\theta[\hat{\theta}]\right)+\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)^2 \\
&= \mathbb{E}_\theta\left[\left(\hat{\theta}-\mathbb{E}_\theta[\hat{\theta}]\right)^2\right]+\left(\mathbb{E}_\theta[\hat{\theta}]-\theta\right)^2 \\
&= \mathrm{Var}_\theta(\hat{\theta})+\mathrm{Bias}_\theta(\hat{\theta},\theta)^2 \qquad \square
\end{aligned}
$$

A classic of estimation theory, this deccomposition can be found in various textbooks, see [Wackerly et al., 2014] for instance.

A.2 Additional Tables

Test Function	Mean Functional Form	Mean Parameters	Kernel Functional Form	Kernel Parameters
1-d Ackley Function	23	38	67	23
1-d Alpine01 Function	2.8	1.8	2	1.2
1-d Alpine N. 2 Function	0.11	0.15	0.16	0.079
1-d Chung Reynolds Function	9.1e+03	5.4e+03	9.3e+03	5.4e+03
Cosine Mixture Function	0.073	0.07	0.11	0.14
1-d Deflected Corrugated Spring function	1.4	1.7	2.1	0.77
1-d Double-Sum Function	0.076	0.044	6.9	0.021
1-d Exponential Function	0.00036	0.0015	0.00064	0.00017
1-d Generelized Drop-Wave Function	0.7	1.4	1.7	0.59
1-d Griewank Function	1.1	0.71	1.9	0.69
2-d Hyper-Ellipsoid Function	0.24	2	3.7	0.0012
Six-Hump Camel Back Function	1.3	3.3	2.9	0.71
Price Function N. 4	1.9e+16	1.1e+16	1.1e+16	3.5e+15
Schaffer Function N. 2	0.79	1.3	0.9	0
Beale Function	27	17	20	0.76
Matyas Function	0.25	0.67	3.8	0.0014
Engvall Function	1.7e+11	3.7e+11	2.7e+11	1.2e+11
El-Attar-Vidyasagar-Dutta Function	3e+07	4.6e+07	7.7e+07	4e+07
Cube Function	2.6e+03	6.3e+03	4.8e+03	0
Holder Table Function N. 1	1.1e+02	62	74	1.8
Goldstein-Price Function	8e+02	5.2e+02	6.8e+02	0
3-d Dixon-Price function	1.5e+03	2.2e+03	9.6e+02	1.3e+03
Schaffer Function N. 2	0.32	0.9	0.94	0.078
Giunta Function	0.16	0.29	0.1	0.00018
Chichinadze Function	19	29	66	3.6
Kearfott Function	3.4	7.8	5.7	0
3-d Hartmann Function	3	4.8	5.3	0.82
3-d Alpine N. 2 Function	14	25	30	4.6
Complex Function	3.1	4	1.4	0
Carrom Table Function	71	71	81	0.2
4-d Alpine N. 2 Function	86	33	66	4.7
Adjiman Function	0.34	0.024	1.6	0.00099
Bird Function	1.4e+02	5.1e+02	1.5e+02	0
4-d Generelized Drop-Wave Function	1.5	2.1	1	0.015
Chichinadze Function	54	47	44	16
Brent Function	0.44	0.47	13	4.4e-05
Bukin Function N. 2	3.1	2.2	1.6e+02	0.089
4-d Sum of Different Squares Function	0.37	1.4	0.32	0
Bent-Cigar Function	3.6e+09	2e+10	7e+09	5.7e+08
Booth Function	13	14	47	15
Bartels Conn Function	2.2e+03	1.3e+04	4e+04	27
7-d Sphere Function	1.5e+02	4.4e+02	97	8.5
Goldstein-Price Function	5.8e+02	4e+02	4.4e+02	0
Price Function N. 2	0.36	1.7	0.84	0.21
Engvall Function	1.4e+11	5.1e+11	2.5e+11	4.7e+10
7-d Deflected Corrugated Spring function	16	38	11	0
7-d Hyper-Ellipsoid function	5e+02	1.7e+03	3.4e+02	0
Bent-Cigar Function	4.8e+10	1.5e+11	3.4e+10	0
Trecanni Function	1.5	3.4	7	0
Matyas Function	0.28	0.59	2.7	0

Table 5: Accumulated difference for BO of 50 randomly selected test functions from `smoof`, see section 4.2.

Test Function	Mean Functional Form	Mean Parameters	Kernel Functional Form	Kernel Parameters
1-d Ackley Function	0.62	1	1.8	0.61
1-d Alpine01 Function	1.4	0.94	1	0.62
1-d Alpine N. 2 Function	0.87	1.2	1.3	0.62
1-d Chung Reynolds Function	1.3	0.74	1.3	0.73
Cosine Mixture Function	0.75	0.72	1.1	1.4
1-d Deflected Corrugated Spring function	0.96	1.1	1.4	0.51
1-d Double-Sum Function	0.043	0.025	3.9	0.012
1-d Exponential Function	0.54	2.3	0.95	0.25
1-d Generelized Drop-Wave Function	0.65	1.3	1.5	0.54
1-d Griewank Function	1	0.64	1.7	0.62
2-d Hyper-Ellipsoid Function	0.16	1.3	2.5	0.00083
Six-Hump Camel Back Function	0.63	1.6	1.4	0.35
Price Function N. 4	1.7	0.99	0.99	0.32
Schaffer Function N. 2	1.1	1.7	1.2	0
Beale Function	1.7	1.1	1.2	0.047
Matyas Function	0.22	0.57	3.2	0.0012
Engvall Function	0.72	1.6	1.2	0.51
El-Attar-Vidyasagar-Dutta Function	0.62	0.96	1.6	0.82
Cube Function	0.75	1.8	1.4	0
Holder Table Function N. 1	1.8	0.99	1.2	0.028
Goldstein-Price Function	1.6	1	1.4	0
3-d Dixon-Price function	1	1.5	0.65	0.85
Schaffer Function N. 2	0.56	1.6	1.7	0.14
Giunta Function	1.2	2.1	0.73	0.0013
Chichinadze Function	0.64	0.99	2.3	0.12
Kearfott Function	0.8	1.8	1.4	0
3-d Hartmann Function	0.86	1.4	1.5	0.24
3-d Alpine N. 2 Function	0.75	1.4	1.6	0.25
Complex Function	1.5	1.9	0.64	0
Carrom Table Function	1.3	1.3	1.4	0.0036
4-d Alpine N. 2 Function	1.8	0.7	1.4	0.1
Adjiman Function	0.69	0.049	3.3	0.002
Bird Function	0.68	2.6	0.76	0
4-d Generelized Drop-Wave Function	1.3	1.8	0.88	0.013
Chichinadze Function	1.3	1.2	1.1	0.39
Brent Function	0.13	0.14	3.7	1.3e-05
Bukin Function N. 2	0.074	0.053	3.9	0.0021
4-d Sum of Different Squares Function	0.73	2.6	0.63	0
Bent-Cigar Function	0.46	2.6	0.9	0.073
Booth Function	0.59	0.61	2.1	0.67
Bartels Conn Function	0.16	0.93	2.9	0.002
7-d Sphere Function	0.84	2.6	0.56	0.049
Goldstein-Price Function	1.6	1.1	1.2	0
Price Function N. 2	0.47	2.2	1.1	0.27
Engvall Function	0.59	2.1	1.1	0.2
7-d Deflected Corrugated Spring function	1	2.3	0.65	0
7-d Hyper-Ellipsoid function	0.79	2.7	0.53	0
Bent-Cigar Function	0.83	2.6	0.59	0
Trecanni Function	0.51	1.1	2.4	0
Matyas Function	0.31	0.66	3	0

Table 6: Standardized accumulated difference (AD) for BO of 50 randomly selected test functions from `smoof`, see section 4.2.

A.3 Additional Figures

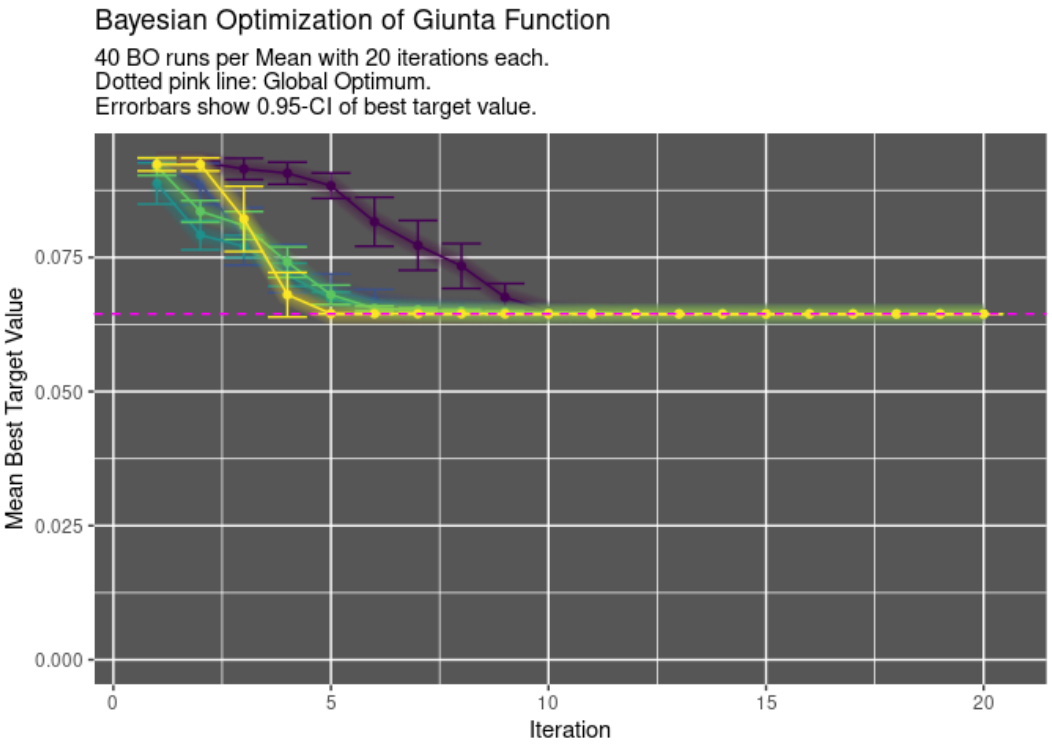

Figure 28: Effect of Mean Function Parameter on Bayesian Optimization of Giunta Function. Mean Constant varies around baseline mean (turquoise) as described in section 4.1.2.

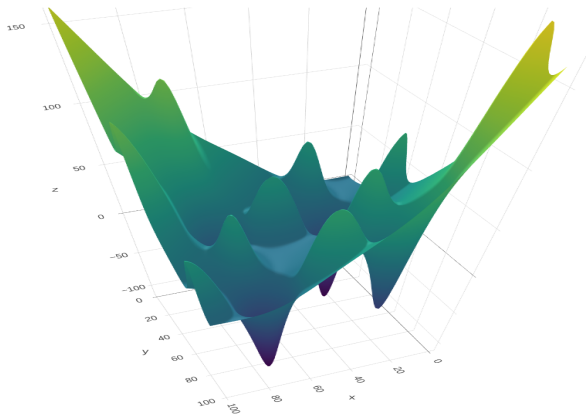

Figure 29: Bird Function

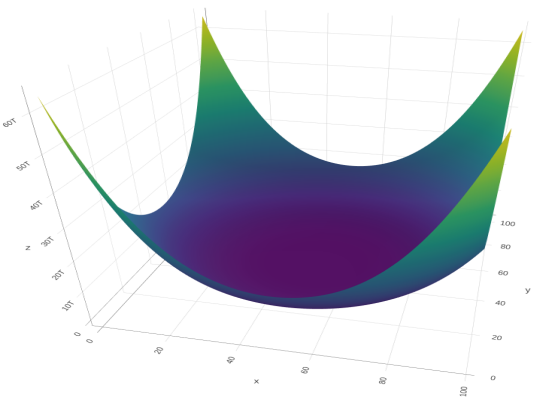

Figure 30: Engvall Function

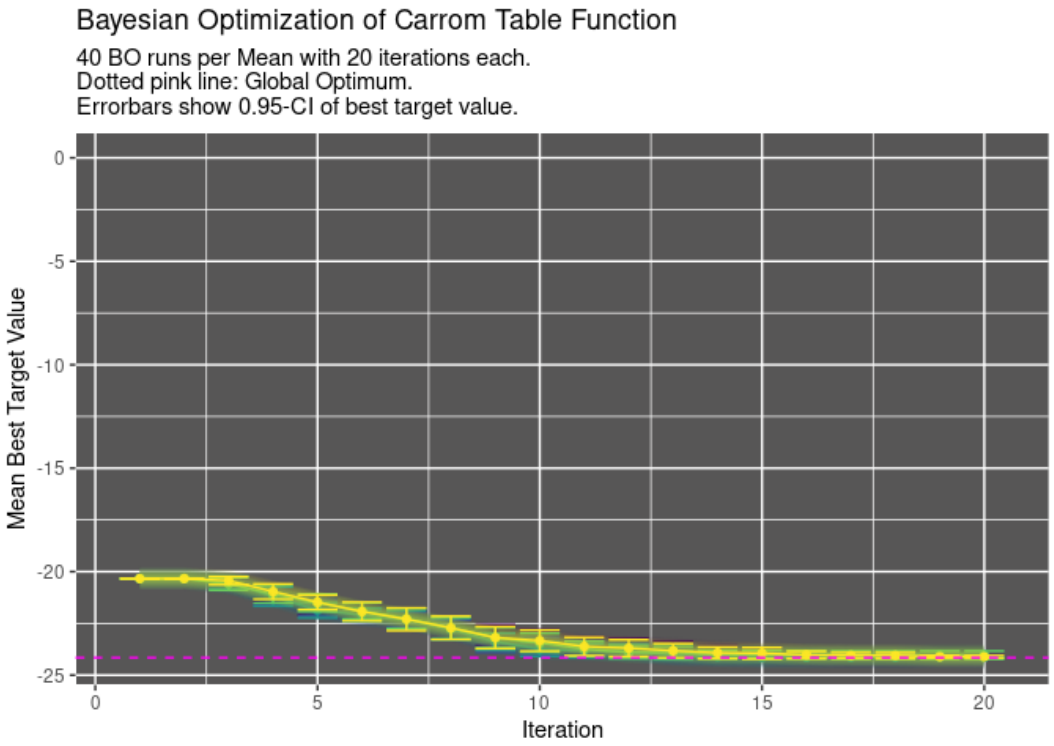

Figure 31: Effect of Mean Function Parameter on Bayesian Optimization of Carrom Table Function. Mean Constant varies around baseline mean (turquoise) as described in section 4.1.2.

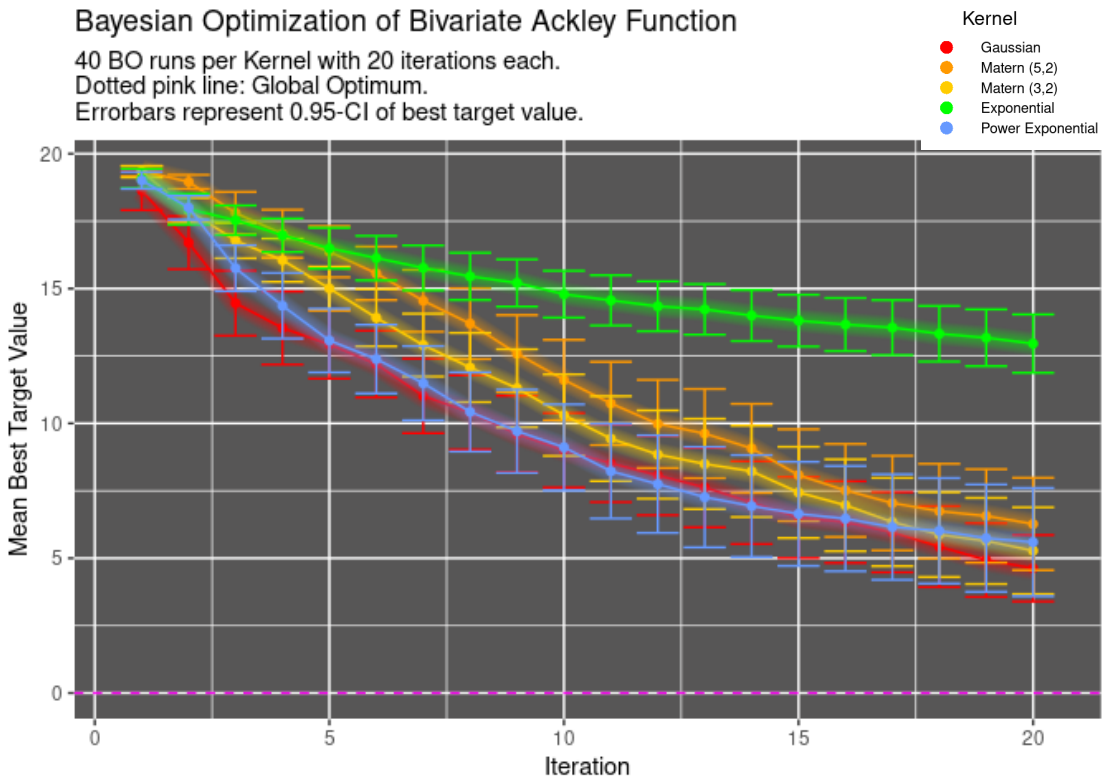

Figure 32: Effect of kernel functional form on Bayesian Optimization of bivariate ("2d") Ackley function.

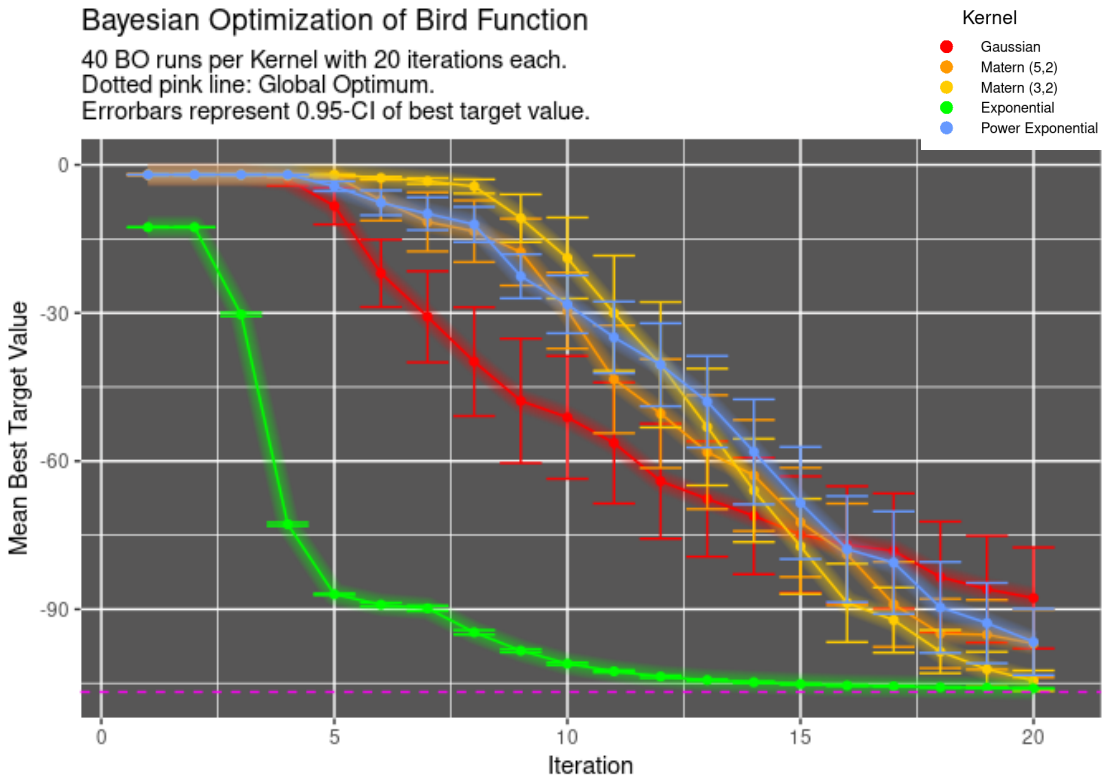

Figure 33: Effect of mean functional Form on Bayesian Optimization of Bird function.

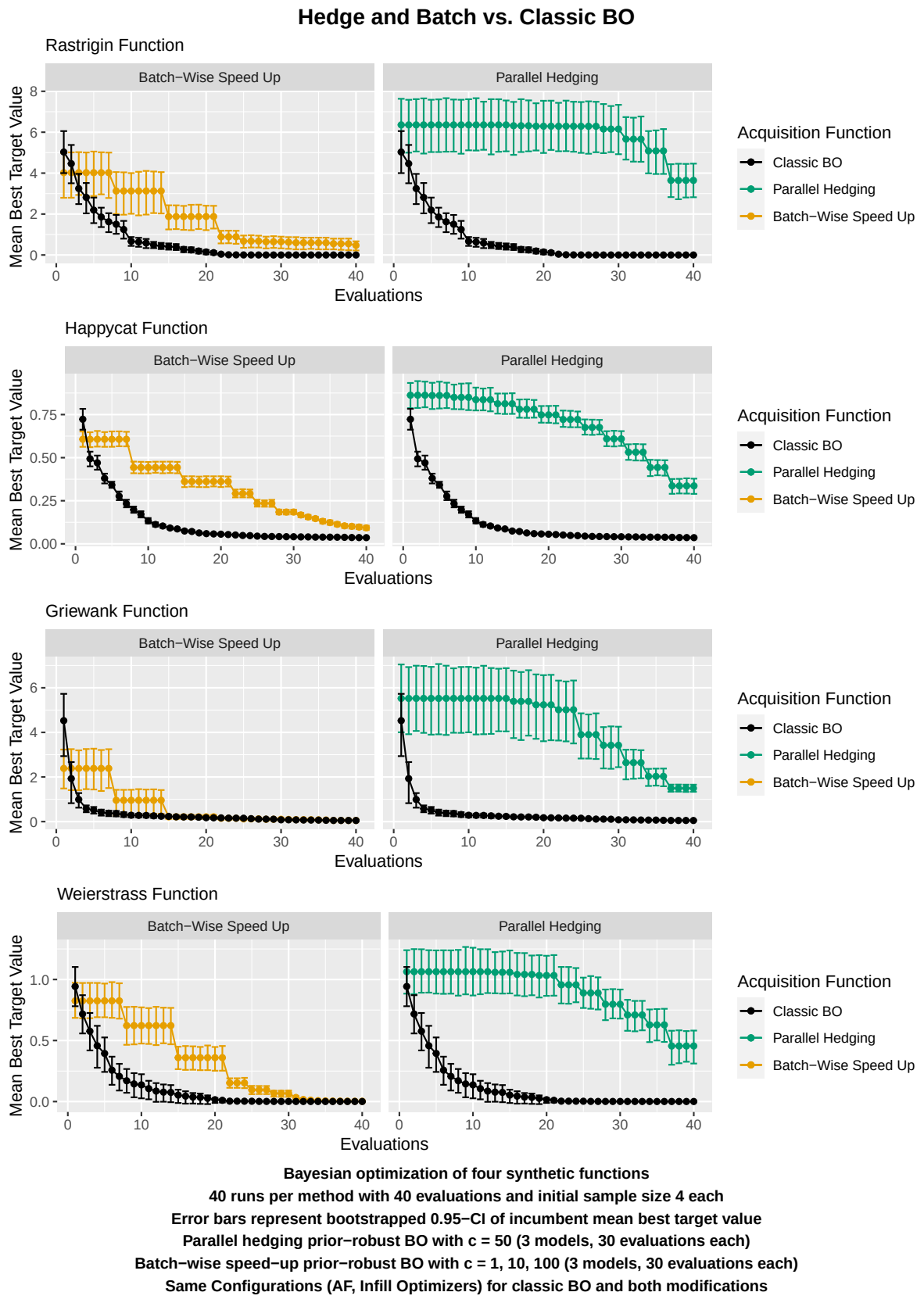

Figure 34: Benchmarking results from synthetic functions: Parallel hedging and batch-wise prior-robust BO vs. classic BO.

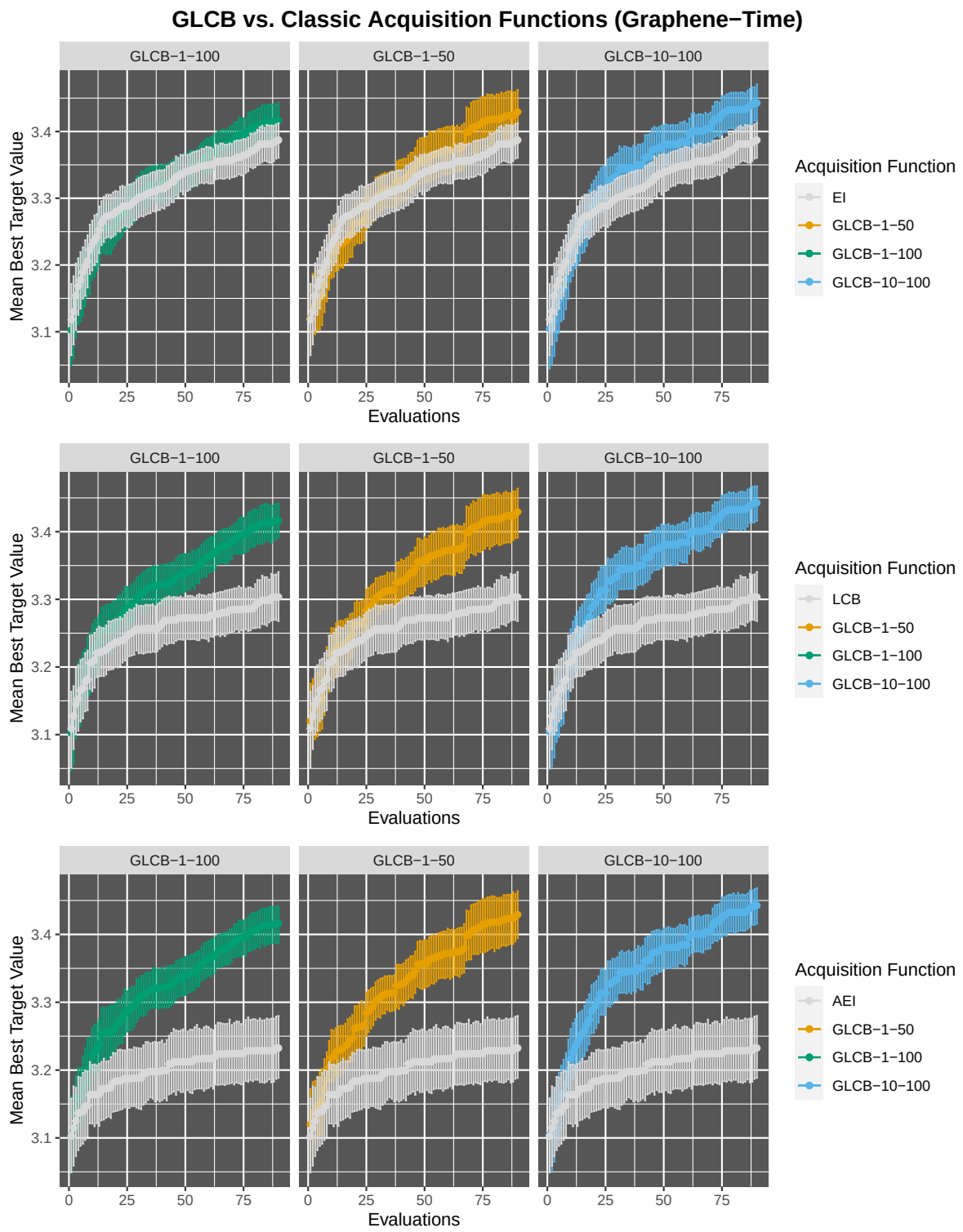

Figure 35: Graphene and time: Benchmarking results from graphene quality as function of laser irradiation time: generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (1).

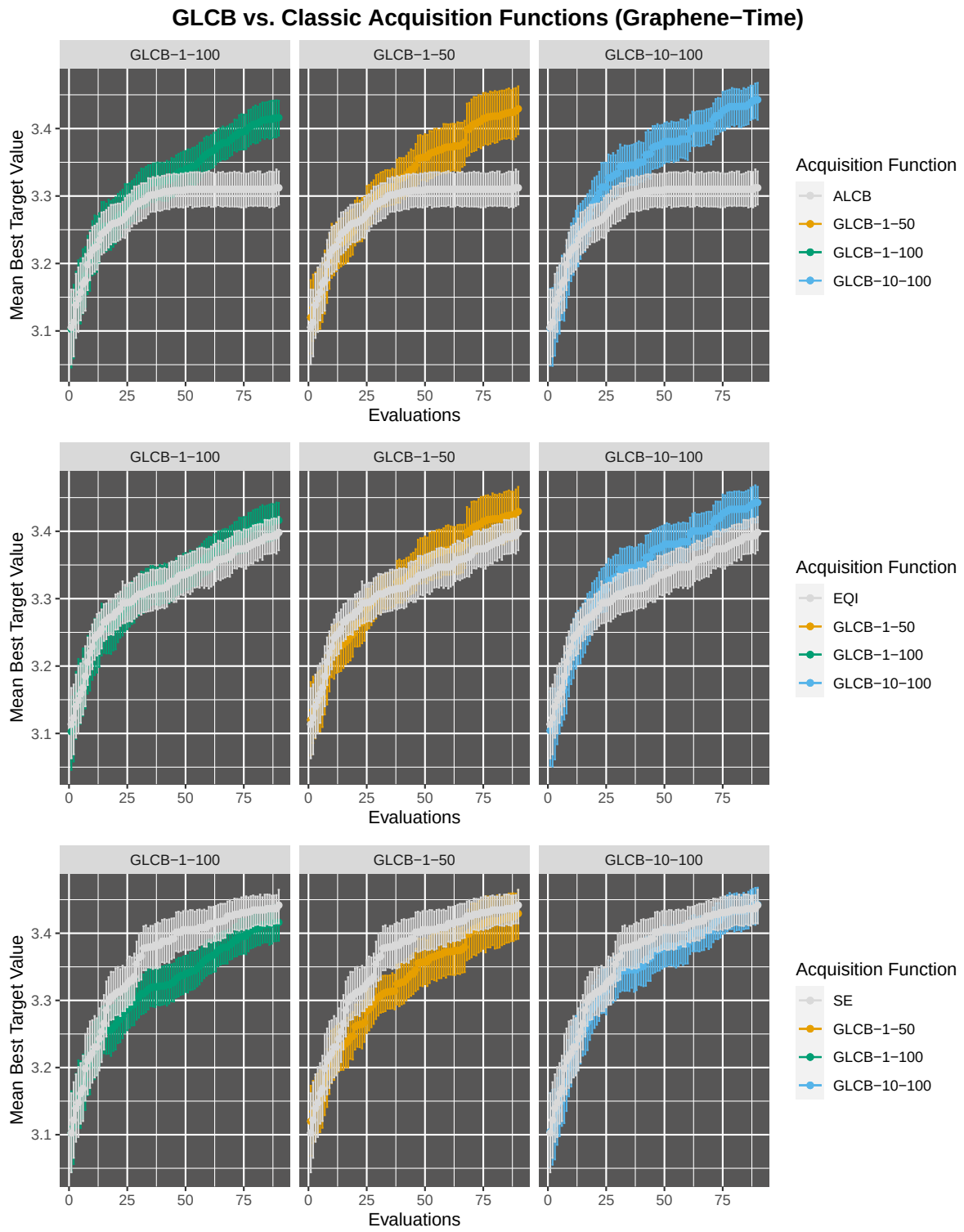

Figure 36: Graphene and time: Benchmarking results from graphene quality as function of laser irradiation time: generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (2).

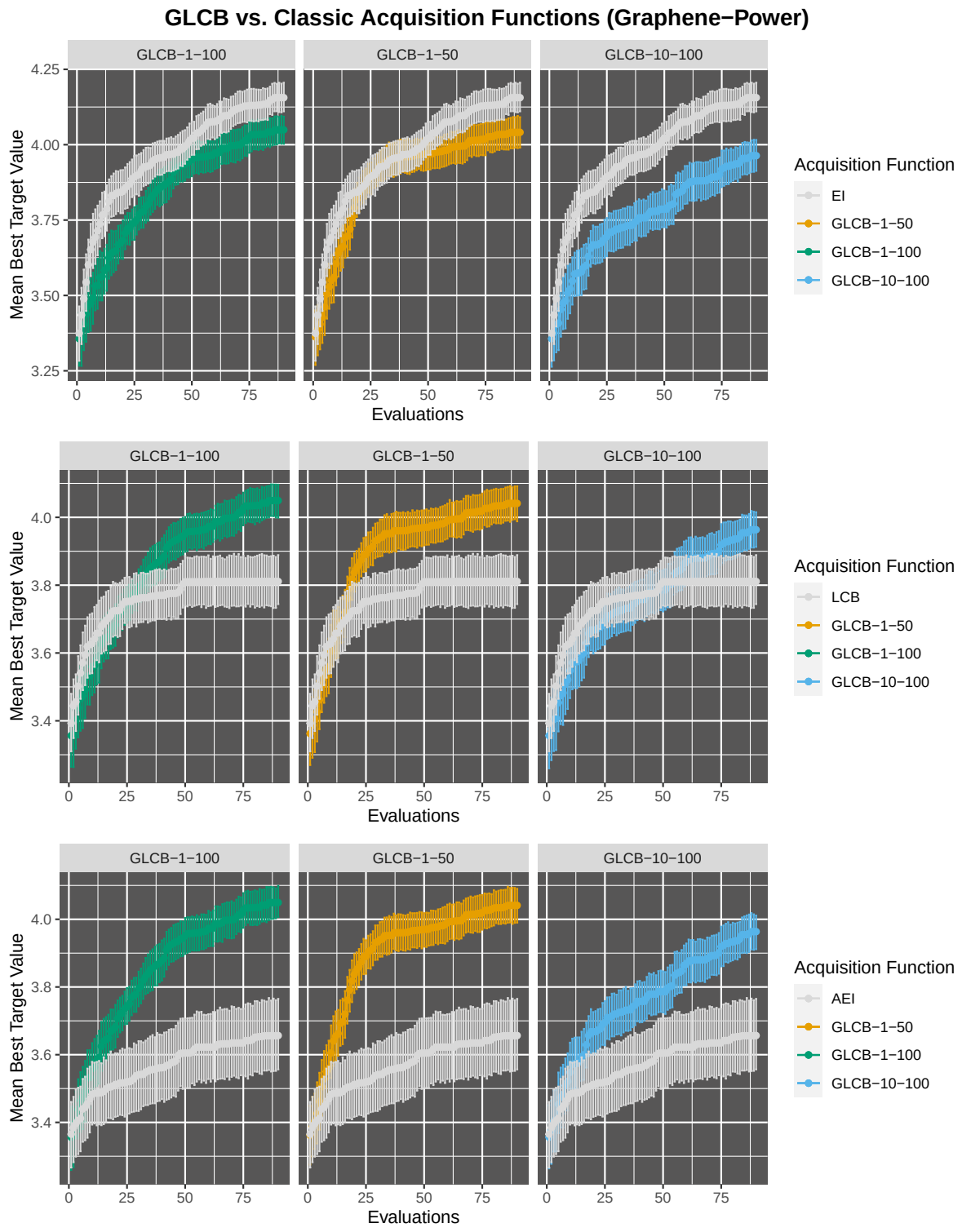

Figure 37: Graphene and power: Benchmarking results from graphene quality as function of laser irradiation power: generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (1).

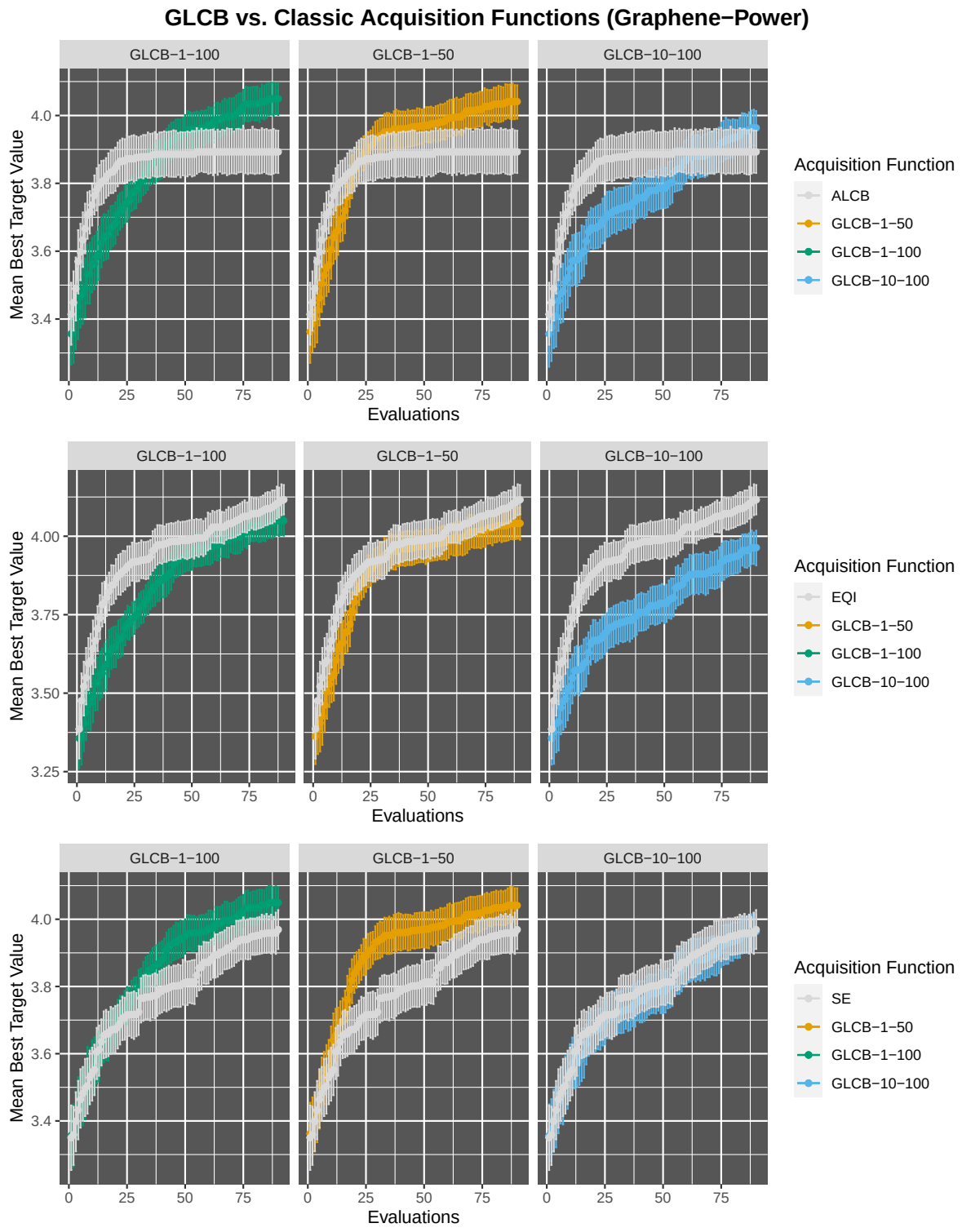

Figure 38: Graphene and power: Benchmarking results from graphene quality as function of laser irradiation power: generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (2).

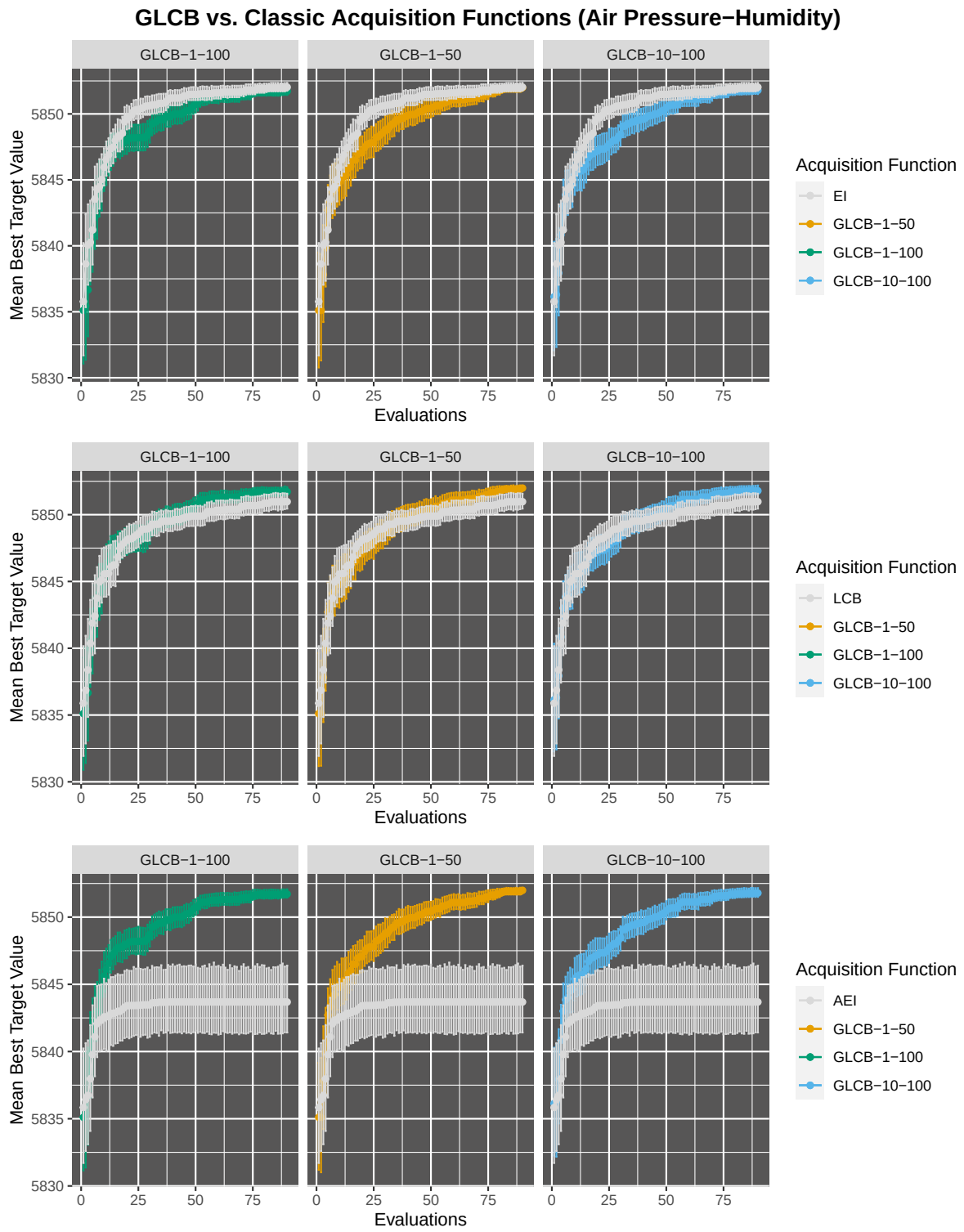

Figure 39: Air pressure in Los Angeles. Benchmarking results from 500 millibar pressure height (m) as function of humidity (%): generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (1).

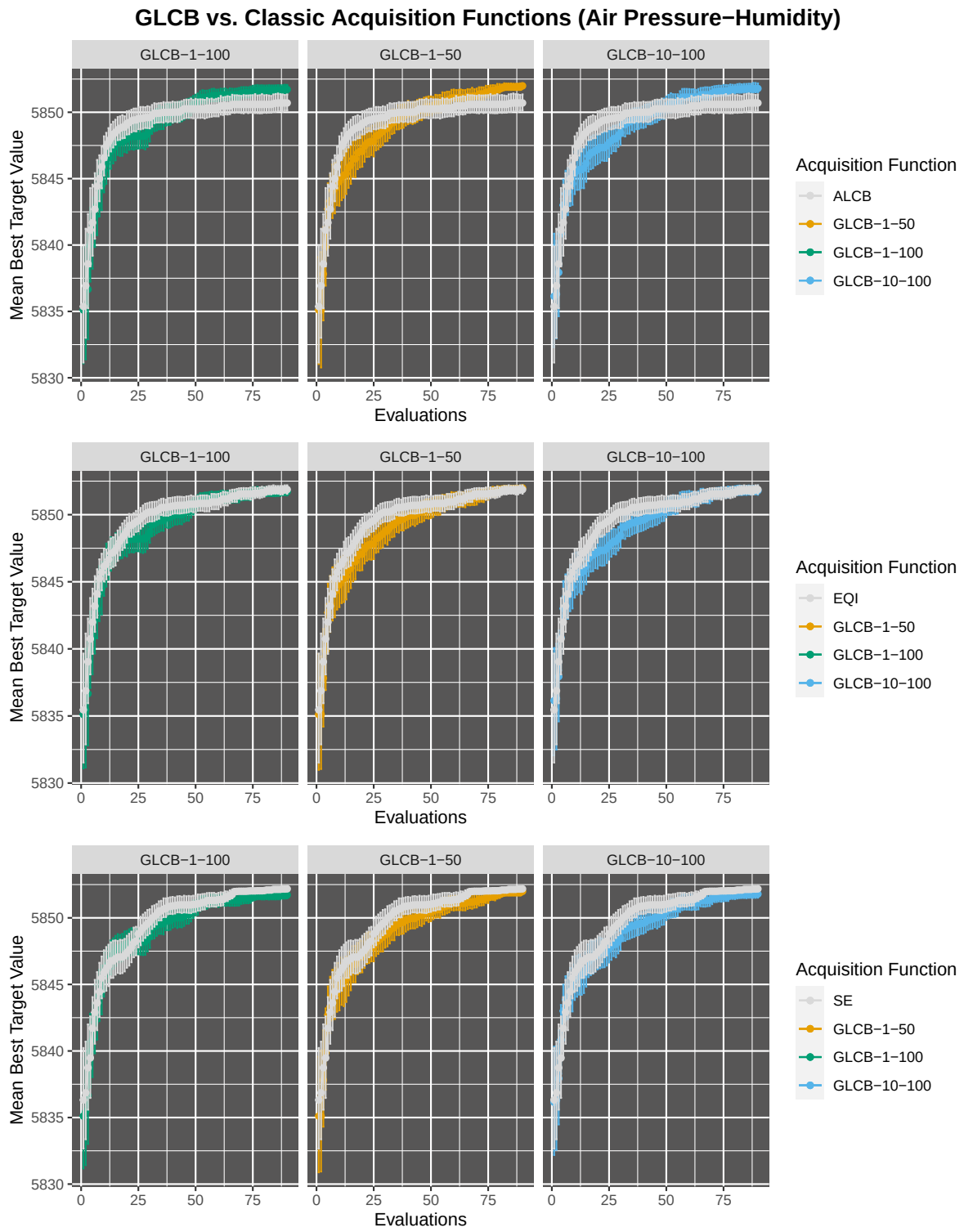

Figure 40: Air pressure in Los Angeles. Benchmarking results from 500 millibar pressure height (m) as function of humidity (%): generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (2).

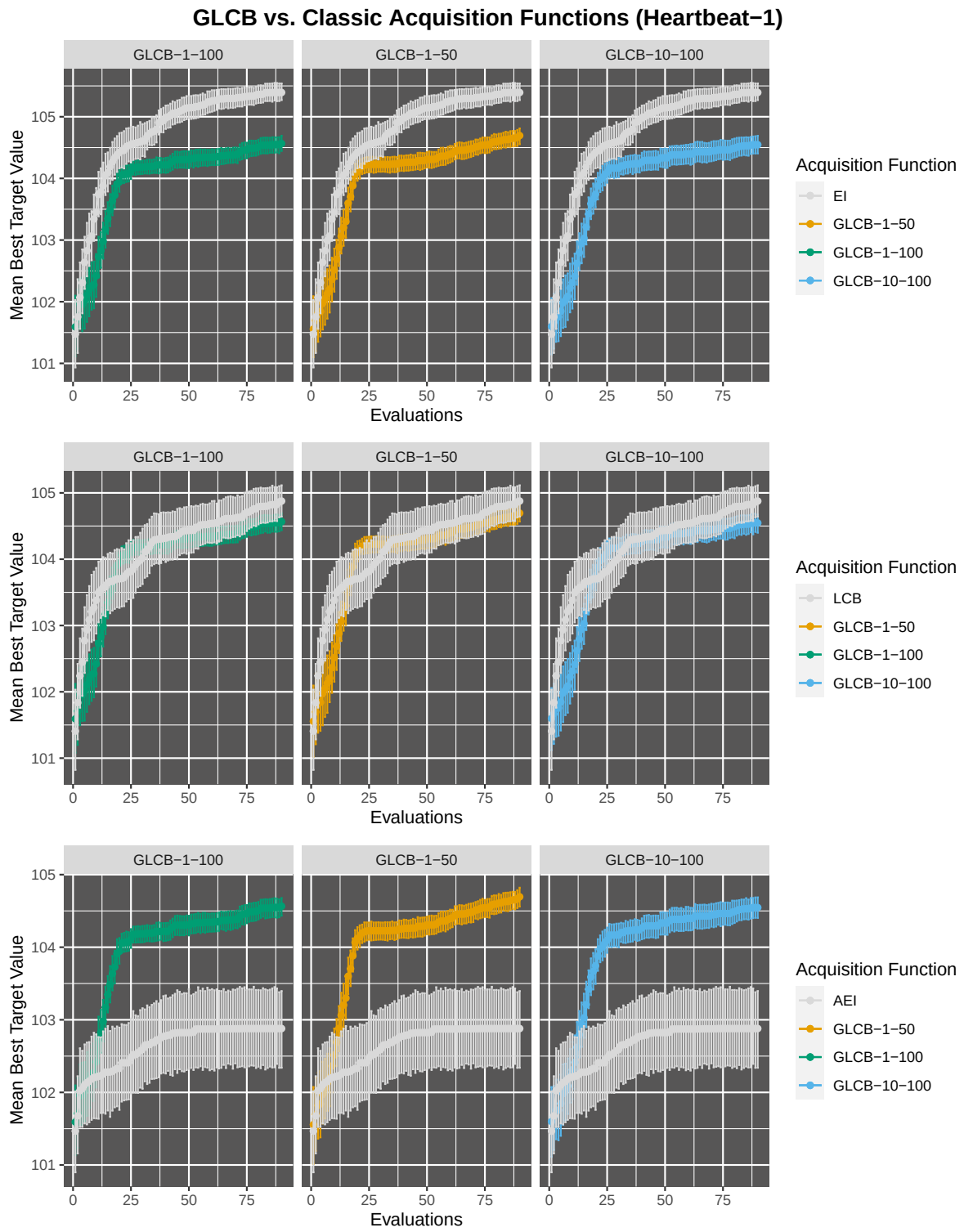

Figure 41: Benchmarking results from heartbeat (individual 1) time series: generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (1).

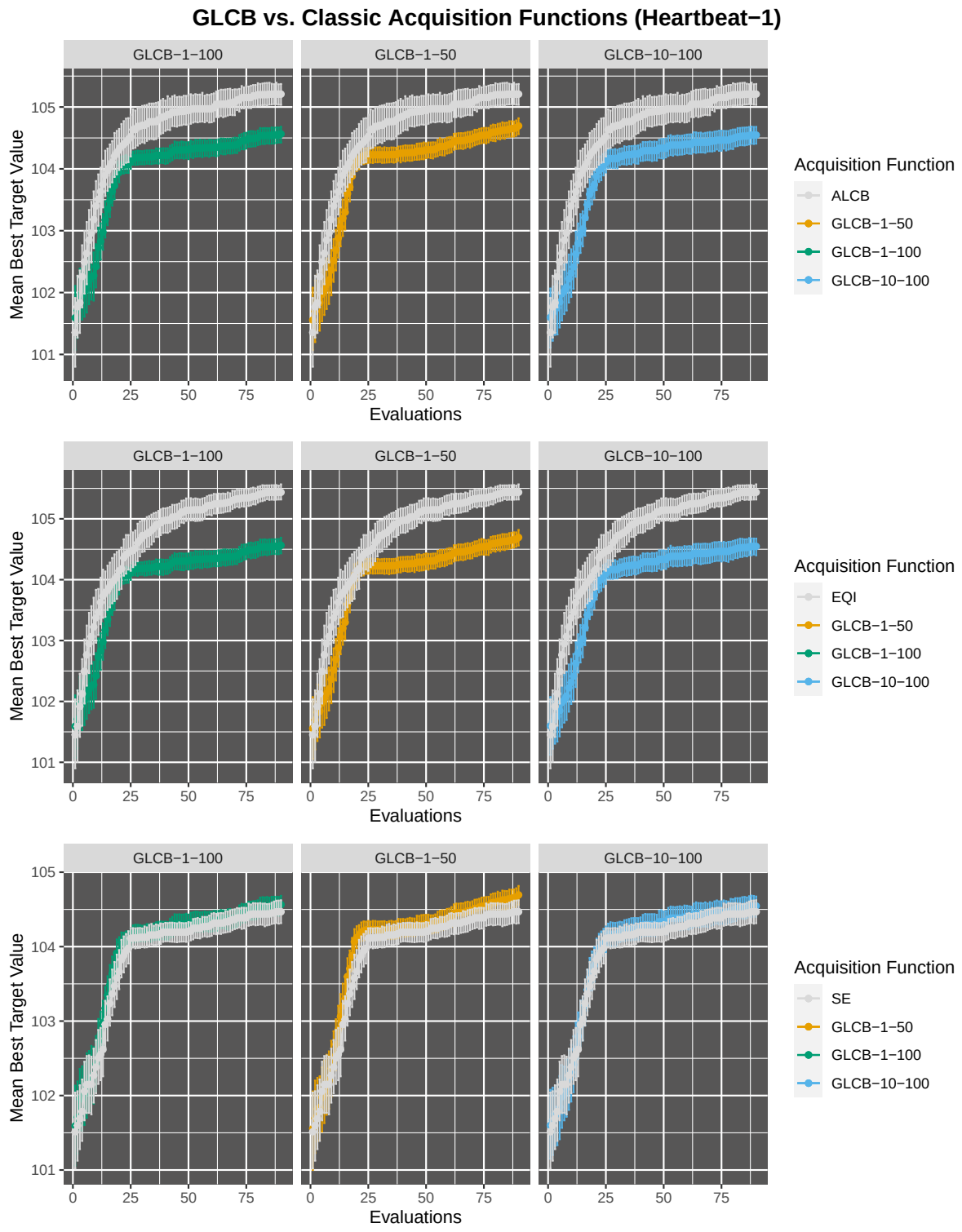

Figure 42: Benchmarking results from heartbeat (individual 1) time series: generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (2).

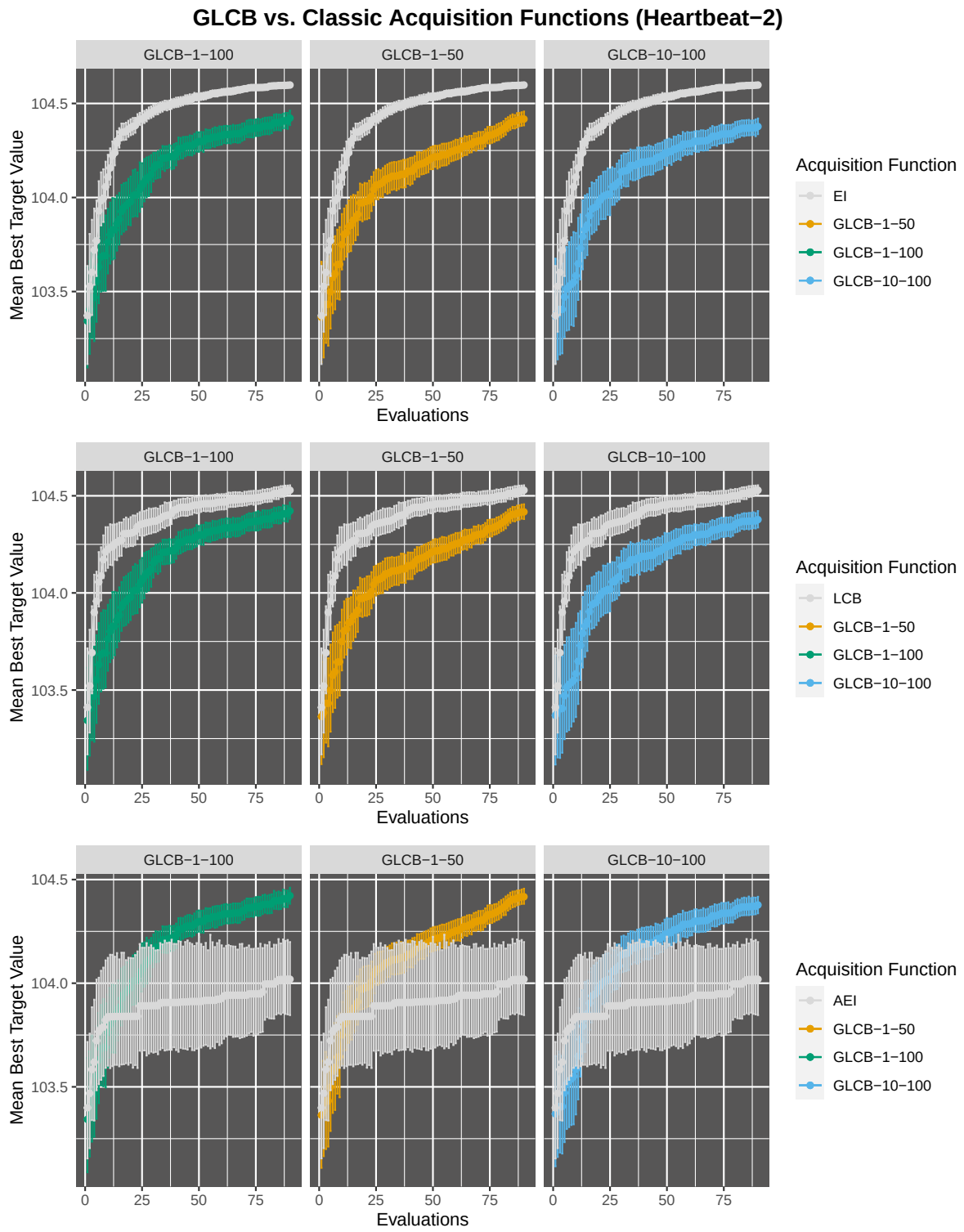

Figure 43: Benchmarking results from heartbeat (individual 2) time series: generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (1).

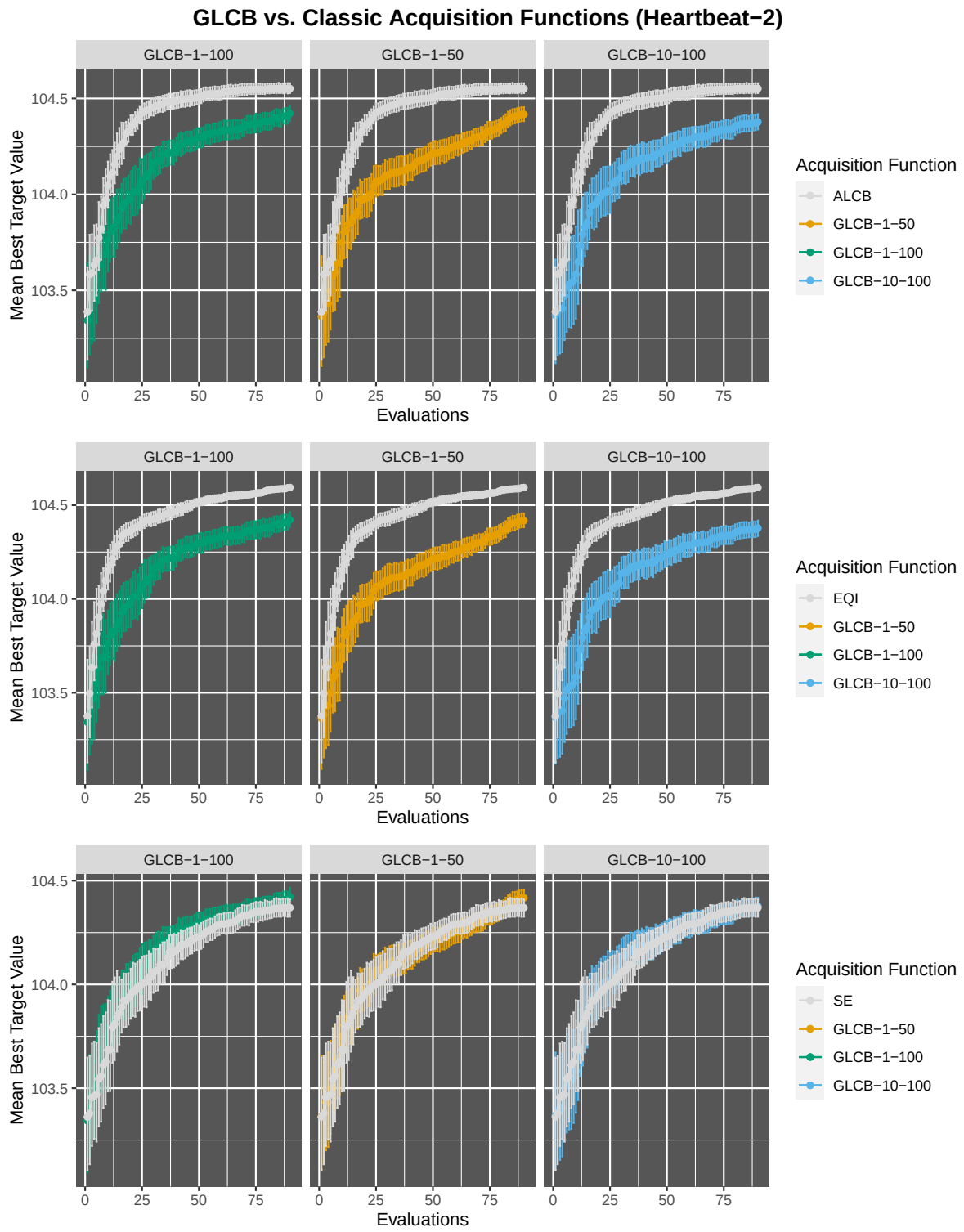

Figure 44: Benchmarking results from heartbeat (individual 2) time series: generalized lower confidence bound (GLCB) vs. several established Acquisition Functions (2).

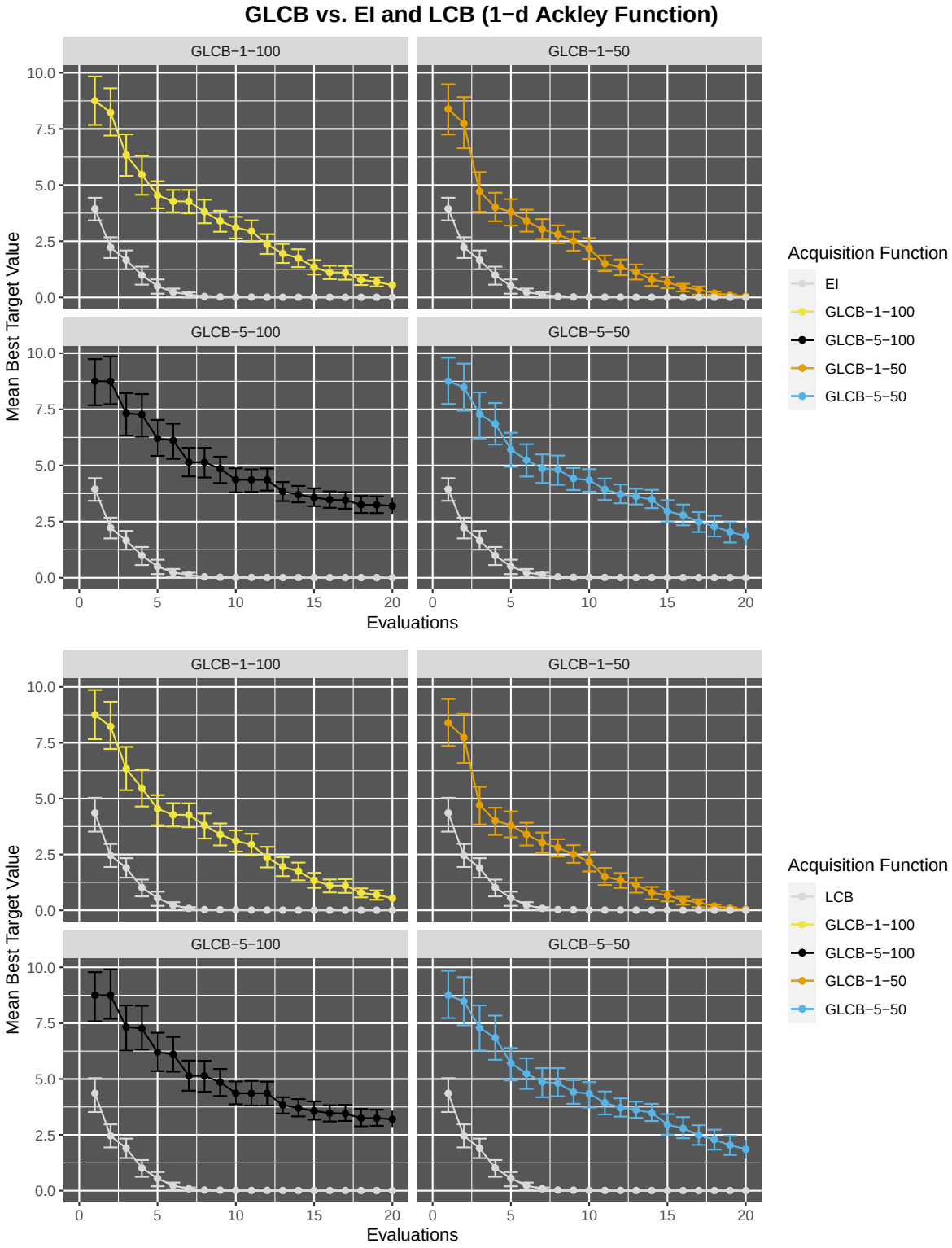

Figure 45: Benchmarking results from synthetic Ackley function: generalized lower confidence bound (GLCB) vs. expected improvement (EI) and lower confidence bound (LCB)

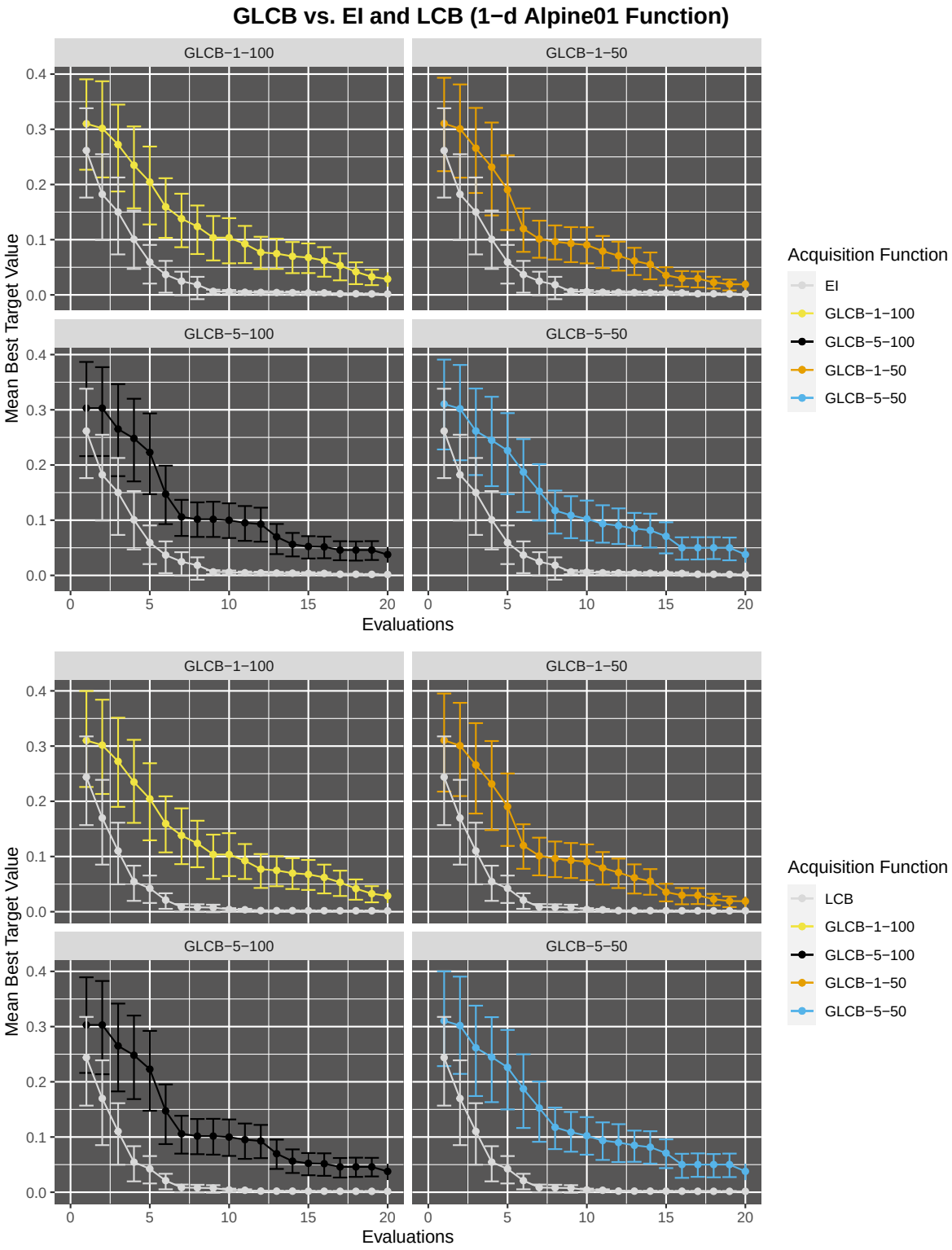

Figure 46: Benchmarking results from synthetic Alpine function: generalized lower confidence bound (GLCB) vs. expected improvement (EI) and lower confidence bound (LCB)

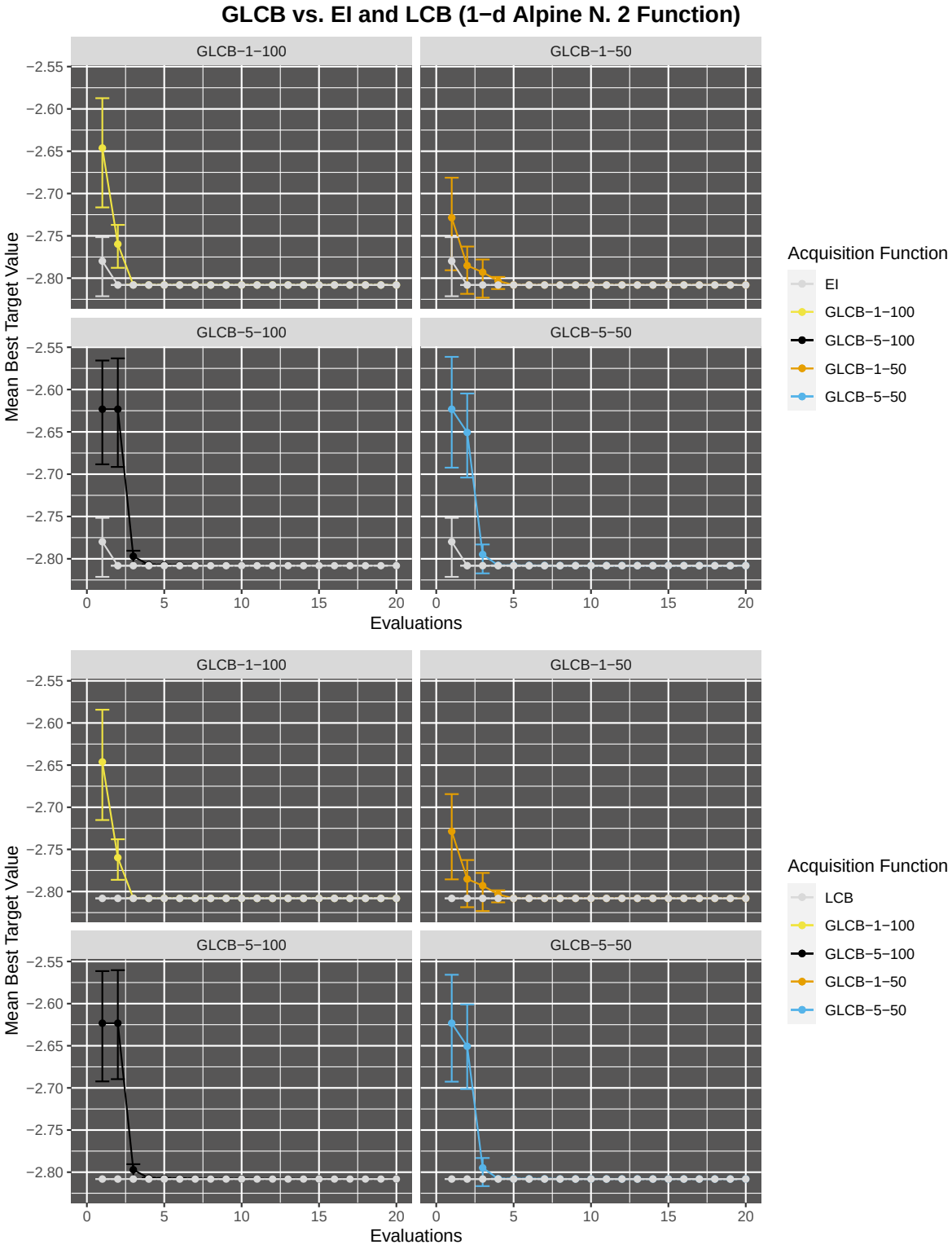

Figure 47: Benchmarking results from synthetic Alpine-2 function: generalized lower confidence bound (GLCB) vs. expected improvement (EI) and lower confidence bound (LCB)

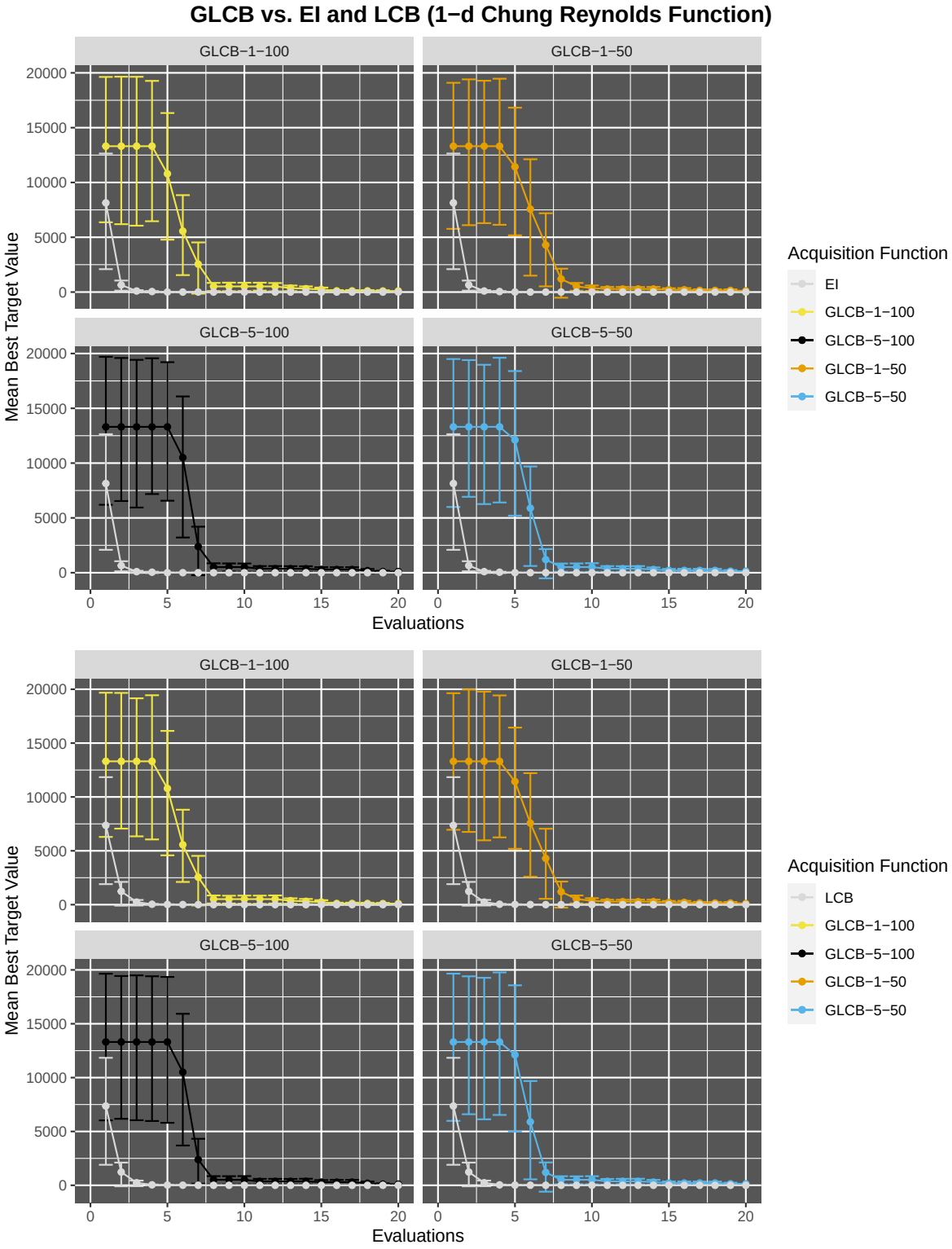

Figure 48: Benchmarking results from synthetic Chung-Reynolds function: generalized lower confidence bound (GLCB) vs. expected improvement (EI) and lower confidence bound (LCB)

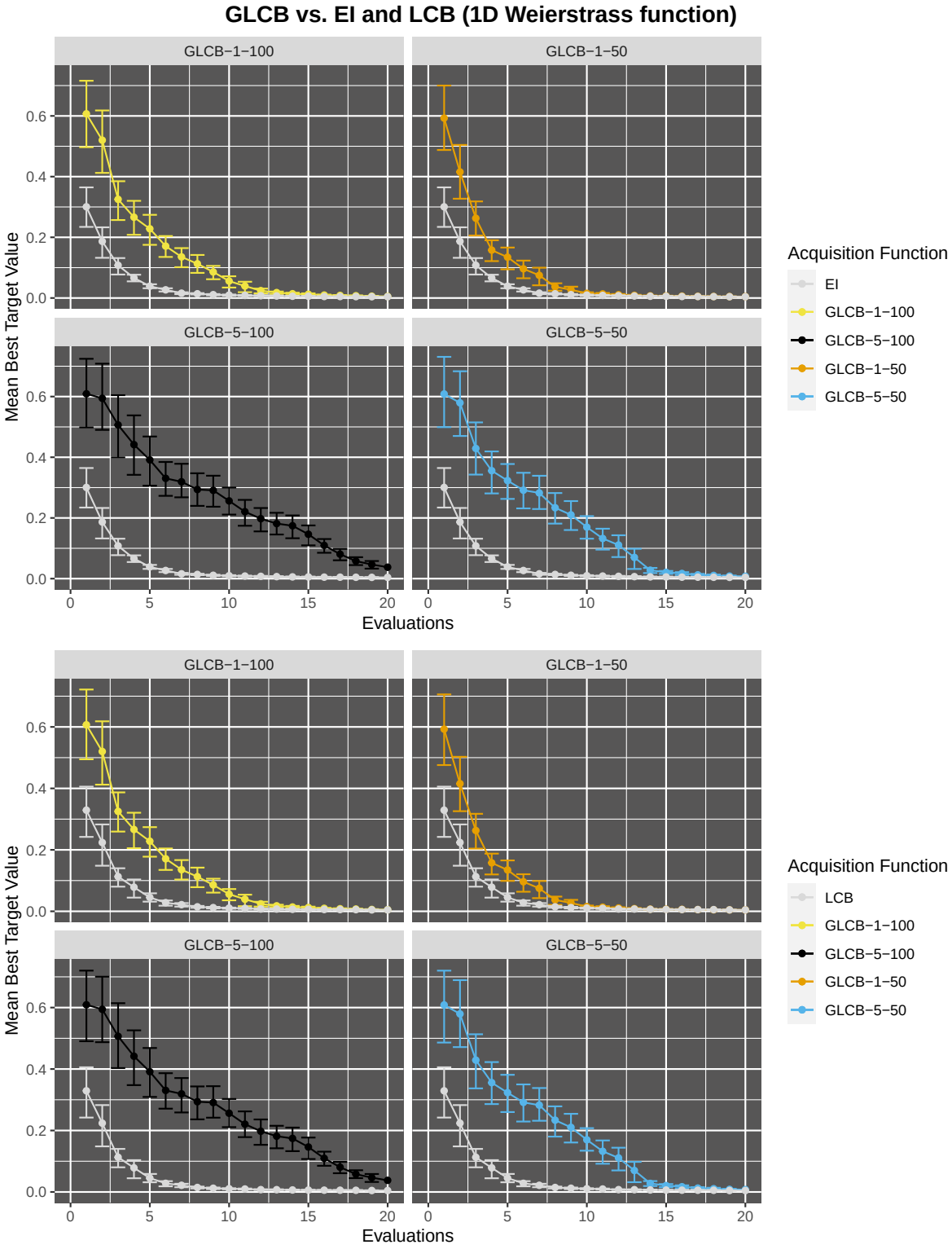

Figure 49: Benchmarking results from synthetic Weierstrass function: generalized lower confidence bound (GLCB) vs. expected improvement (EI) and lower confidence bound (LCB)

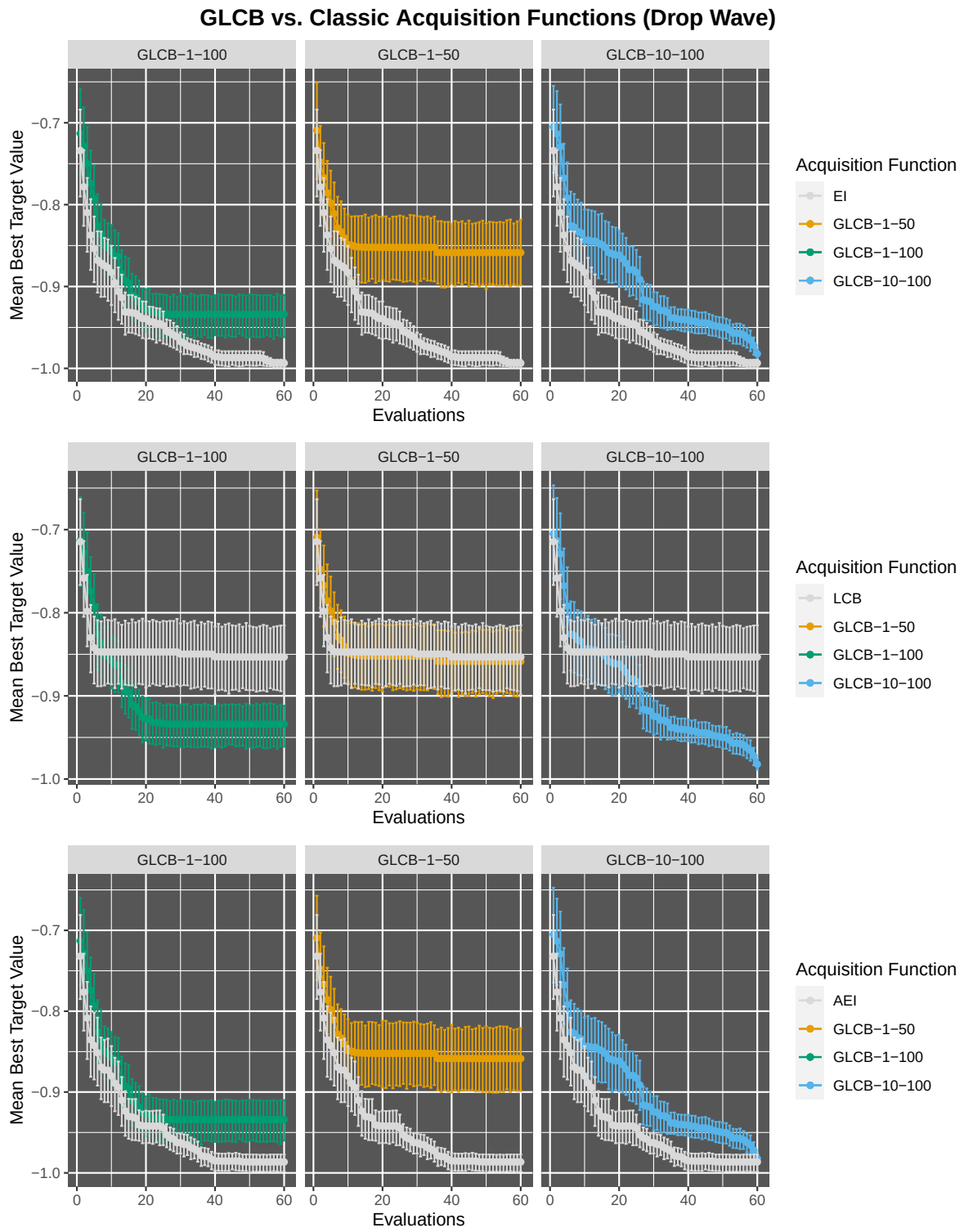

Figure 50: Benchmarking results from synthetic drop wave function: generalized lower confidence bound (GLCB) vs. established aquisition functions (1).

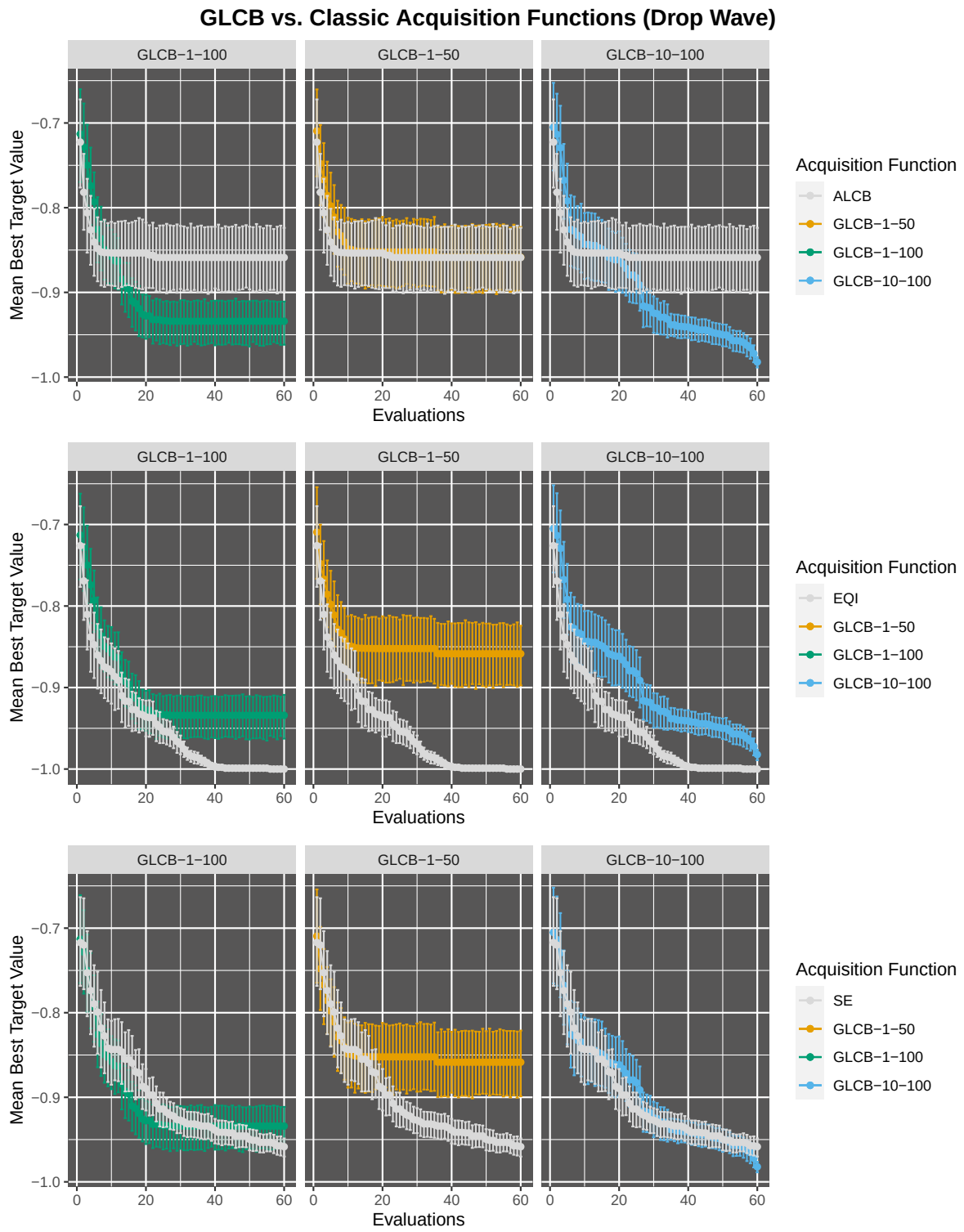

Figure 51: Benchmarking results from synthetic drop wave function: generalized lower confidence bound (GLCB) vs. established aquisition functions (2).

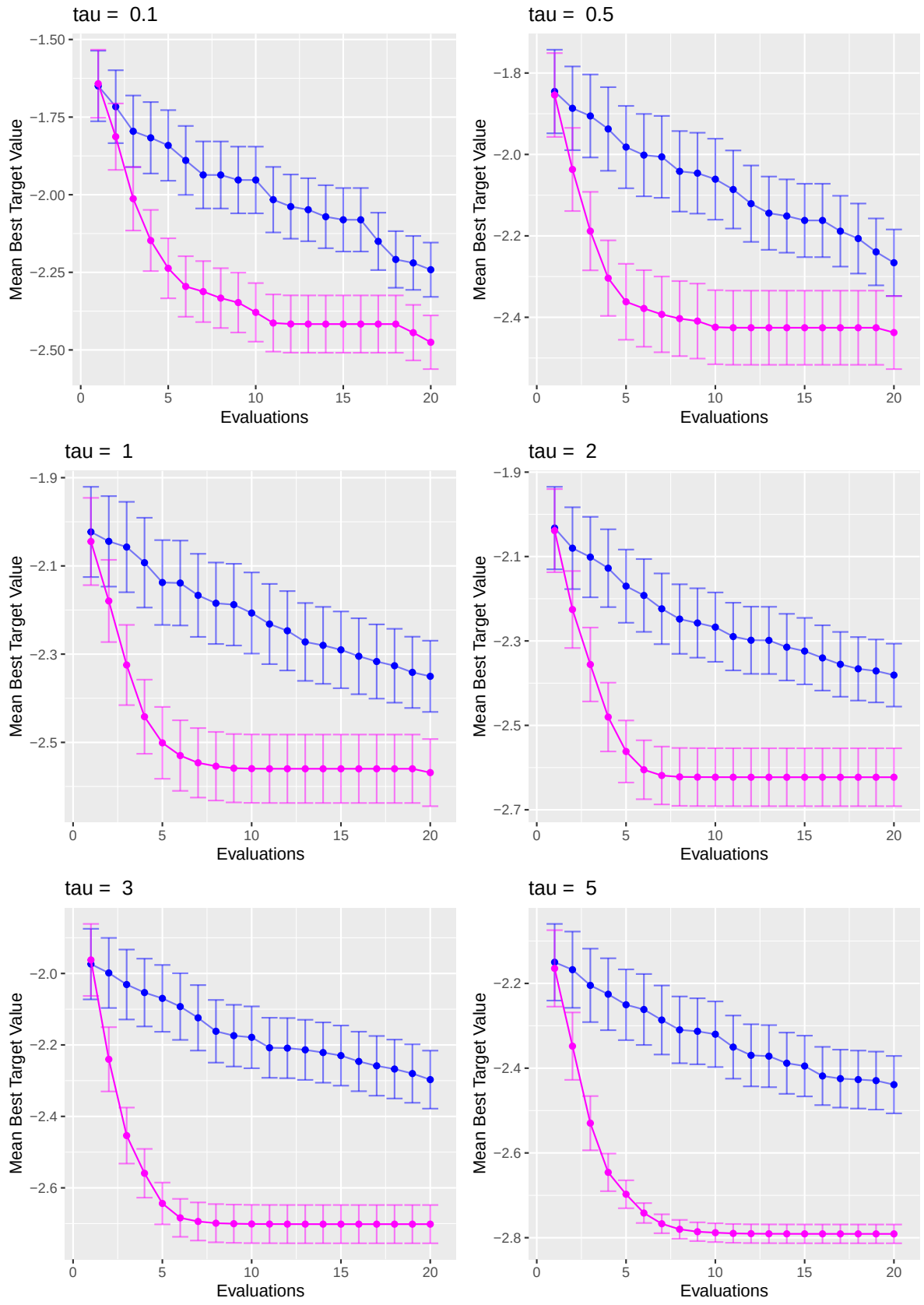

Figure 52: Benchmarking results from second alpine function: BO with weighted ML vs. BO with unweighted ML, both with LCB with varying τ as AF.

B Code and Data Availability

All code and data sets to reproduce the herein presented results, tables and figures can be found at `https://github.com/rodemann/master-thesis-r`. Access to the repository is granted upon request.

C Declaration of Academic Integrity

I hereby confirm that this master's thesis is the result of my own independent scholarly work. I further declare that I have documented all sources and material used. This thesis was not previously presented to another examination board.

Date Signature (Julian Rodemann)

D List of Abbreviations

AD accumulated difference. 26, 27, 29, 30, 85

AEI augmented expected improvement. 10, 11, 45, 50–52

AF acquisition function. 5, 7, 8, 10–14, 17, 20, 22, 33, 34, 42, 45, 49–52, 59, 61, 62, 64, 67, 72

ALCB adaptive lower confidence bound. 8, 45, 50–53

BO Bayesian optimization. 1–5, 7, 9, 12–15, 17–22, 26–29, 31–36, 39, 40, 42, 45, 46, 49, 50, 52–59, 62, 64, 65, 67–69, 72, 84, 85

BQO Bayesian quadrature optimization. 55

CBO causal Bayesian optimization. 59, 60

CI confidence interval. ix, 49

COD curse of dimensionality. 22

CrI credible interval. 38

DL deep learning. 40, 55, 58

EA evolutionary algorithm. 71

EI expected improvement. 6–8, 11, 12, 20, 33, 45, 50–53, 59, 64, 100–104

EQI expected quantile improvement. 11, 45, 50–53

GLCB generalized lower confidence bound. 42–45, 49–54, 58, 59, 69, 72, 90–106

GP Gaussian process. 2, 4–8, 11, 14–22, 25, 29, 31–33, 35, 36, 38–41, 44, 54–56, 58, 59, 63, 64, 67, 72

HC hypervolume contribution. 14

IAF integrated acquisition function. 31, 34

IGP imprecise Gaussian process. xii, xiii, 35–42, 45, 53, 54, 57–59

IP imprecise probabilities. 2–4, 59

LCB lower confidence bound. xii, 6, 8–10, 12, 42, 45, 49–53, 61, 64, 100–104

LHS latin hypercube sampling. 22, 25, 30, 45, 62, 64

MAP maximum a posteriori. 40

MCMC Markov chain Monte Carlo. 34

ML machine learning. 1, 3, 54

ML maximum likelihood. xiii, 1, 9, 20, 31, 32, 35, 38, 40, 61, 63–65, 69, 72

MOP mean optimization path. 19–22, 24–29, 45, 49, 50

MSE mean squared error. xii, 9, 10

NAS neural architecture search. 58

NFL no free lunch theorem. 35, 36, 52

PI probability of improvement. 6–8, 12

RF random forest. 5, 6, 17, 46–48, 52, 57, 58

SE standard error. 8, 33, 45, 50–53

SED standard error distribution value. 61, 62

SER standard error ratio. 61, 62

SH successive halving. 41

SLB statistical lower bound. xii, 9, 10, 61

SM surrogate model. 5, 6, 8, 9, 11–14, 17, 18, 21, 22, 29, 33–35, 39, 41, 42, 58, 61, 62, 69