Practical Bayesian optimization with application to tuning machine learning algorithms

by

Bobak Shahriari

B. Sc., McGill University, 2008

M. Sc., Simon Fraser University, 2011

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Computer Science)

The University of British Columbia

(Vancouver)

August 2016

Abstract

Bayesian optimization has recently emerged in the machine learning community as a very effective automatic alternative to the tedious task of hand-tuning algorithm hyperparameters. Although it is a relatively new aspect of machine learning, it has known roots in the Bayesian experimental design (Lindley, 1956; Chaloner and Verdinelli, 1995), the design and analysis of computer experiments (DACE; Sacks et al., 1989), Kriging (Krige, 1951), and multi-armed bandits (Gittins, 1979). In this thesis, we motivate and introduce the model-based optimization framework and provide some historical context to the technique that dates back as far as 1933 with application to clinical drug trials (Thompson, 1933).

Contributions of this work include a Bayesian gap-based exploration policy, inspired by Gabillon et al. (2012); a principled information-theoretic portfolio strategy, out-performing the portfolio of Hoffman et al. (2011); and a general practical technique circumventing the need for an initial bounding box. These various works each address existing practical challenges in the way of more widespread adoption of probabilistic model-based optimization techniques.

Finally, we conclude this thesis with important directions for future research, emphasizing scalability and computational feasibility of the approach as a general purpose optimizer.

Preface

This thesis is an edited volume of three separate research projects led or co-led by the author of the thesis. Two of these projects culminated in publications at the International Conference on Artificial Intelligence and Statistics (AiStats) in 2014 and 2016—namely (Hoffman et al., 2014; Shahriari et al., 2016a)—and are detailed in Chapters 6 and 4. All experiments and figures were respectively run and produced by the author, and are included in the present document with the approval of all co-authors.

Chapters 2 and 3 were inspired by a third publication—namely (Shahriari et al., 2016b)—and may exhibit some minor overlap. Similarly, Chapter 5 and Appendix D were also originally written by the author for publication in (Shahriari et al., 2016b) and therefore features significant overlap. Once again, barring some minor editing, these chapters were written, and all figures were produced, by the author.

Table of Contents

List of Tables

List of Figures

Acknowledgements

Thank you to my examiners Dale Schuurmans, Jane Z Wang, and Will J Welch for their feedback on the thesis as well as a very interesting discussion at my defence. Thank you to Mark Schmidt and Nick Harvey for serving on my supervisory committee. None of the work in this thesis would have been possible without the intellectual, emotional, and financial support of both my supervisors Alex and Nando, even though distance and time zones often worked against us. From PEP8 to GPs, so much of what I know has, one way or another, gone through Matt W Hoffman, that he holds an honorary supervisor status in my mind. Thank you to Kath and Joyce for their patience, and to Joanna McGrenere for a very supportive meeting at a particularly sensitive time.

Since creative writing was never a skill of mine, I will simply default to a somewhat chronological listing of important people who, through sustained or momentary support, have helped me complete the complicated experience that is graduate school.

It all begins with my mom and dad for making sure their son came before their irreconcilable differences. My older brother Behrak for always being available for a chat and a Corona; my younger brother Bijan who is smarter, fitter, and better looking than his brothers; my brother Behrad of whom I still have vivid fond memories. My aunt and second mother, Amé Mavach, who always demonstrated a saintly generosity and dedication toward her loved ones. My uncle Gérard and cousin Jean-Philippe for following suit and being two of the sweetest human beings I have ever known. My aunt Sima and uncle Alain for their monthly phone calls full of support, wisdom,

and humour. My oldest friend Ariane with whom I still laugh as much and as loudly as we did 26 years ago, and her warm and fuzzy family Dina, Mitra, and Hushang. My good friends Marc, Walid, and Kayvan, who are always a game of phone tag away from a deep conversation; Priscilla and Catherine, for their much needed fashion advice; and Denis, my favourite pilot.

At McGill I had the good fortune of crossing paths with two exceptional academic brothers: Charles, whose skills in the art of chit-chat are unparalleled, and Ashton, my life and chess coach. Moving to Vancouver was made easy thanks to Kevin, Monica, Franck, Ashley, Danny, Frenchie and Kate. Leaving Vancouver is extremely difficult because of Fitzy, Grahamy, Ty, Alexis, Jonesy and Angela, Jesse and SJ, Julesy and Addis, Noah and Kelly. Thank you to my labmates Matt, Misha, Ziyu, Emty, Mareija, Dave, and Sancho; and my remote fellow Nandians: Iannis, Brendan, Marcin, and Jakob. Thanks as well to a more recent friend: Mike Gelbart.

During a short stint in San Francisco, I had the absolute pleasure of living with my good friend Ashton and the one and only Bob West, whose love for a good pun may even rival my own; through them I met a fistful of tremendous human beings: Jake the wise, Hristo the Wolf, Julian the bodybuilder, Matteu the limer, Ryan the corbeau, and Fabian the mailman. While in the bay area, I worked with some very talented folks: Nathan, Dennis, Jamal, and Aaron; and I would particularly like to acknowledge Martin, Mike, and Ofer for their great mentorship and timely advice. Most of my weekends were spent enjoying the *dolce vita* with Jason, Colleen, Callie, Tarun, Joseph, Lindsay and Becca, Rajeev and Jillian, and Audrey.

When I spent a brief time in London, several people were responsible for keeping me sane, in particular I would like to thank Ogla for her incredible patience; Dave for many a necessary distraction; Ryan and John for a timely heart-to-heart at the Euston Tap that I will likely never forget; Nando, Anj, and Sienna for showing me exceptional hospitality and their pooch Picasso for countless calming walks. On the topic of hospitality, Iannis Assael certainly takes the cake by sharing his 250 sqft studio apartment for three weeks, not once making me feel unwelcome. Gabe, with a similar

heroic gesture, let me crash on an inflatable mattress in his room for a whole week.

Coming back to British Columbia, was a difficult adjustment, one that Nilima, Paul, Vijay, Mira, and Amanda have made much easier. While finishing my thesis on the hightest peak of Mount Burnaby, Nilima and her students Sebastián and Bamdad filled my final student days with stimulating discussions and barrels of laughter; in fact all three deserve acknowledgement for relentless and uncensored feedback on the thesis.

There are a few people deserving acknowledgement that do not quite fit in the chronological flow: Fitzy for being a true ally, a model for self-improvement, and always sharing a teen-aged Scotch; "Swiss" Marc Ryser for being a kindred—more hipster—spirit for as long as I've known him; Gunnar for planning a once-in-a-lifetime Icelandic adventure; Joel Sidoruk and Jean-Philippe Ramaut for being models of bravery and successful risk-taking; Femi for being a loyal friend with an infectious laugh; Jimi for being a fun and light-hearted friend; Moustapha and Roberto for being great friends in the machine learning community; and Dave Bryant for being a great companion one particularly stressful evening. Of course, no acknowledgement would be complete without a strong French contingent: Franck, Émine, Caroline, Aurélie, Benjamin.

Last but certainly not least, I would like to thank Nilima for being a mentor in mathematics, academics, integrity, generosity and charity, parenthood and friendship. Nilima, Paul, and their wonderful children Vijay and Mira have truly been—and continue to be—a surrogate family.

Chapter 1

Introduction

Model-based optimization has become a popular and successful approach to global optimization of black-box functions. In particular, the framework specializes in finding the optimizer of an unknown objective function f in a data efficient way, using a relatively low number of function evaluations. The quantification "relatively low" is deliberately vague because it will often depend on the dimensionality of the input space, the variability in the objective function, and the level of noise in the output. In general, the objective functions of interest are non-convex and multi-modal, they do not readily produce derivative information, and can be expensive to evaluate, hence the goal of data efficiency. The Bayesian approach is particularly appealing in this setting because it facilitates combining small amounts of data with prior knowledge or expertise to make predictions with uncertainties that are consistent with the prior. In fact, a positive side-effect of a Bayesian methodology is that it requires the practitioner to precisely quantify her prior knowledge[1] and encode her inductive bias into a prior probability distribution.

This approach is widely applicable to any problem where one can evaluate the objective function at arbitrary points, even when these evaluations are corrupted by measurement noise. Major successful applications of these

[1] Actually, even when there is no prior knowledge, this can be encoded using what is colloquially known as a *non-informative* prior, *e.g.,* the Jeffreys prior.

techniques include:

- ▷ design and analysis of computer experiments (Sacks et al., 1989; Jones et al., 1998; Azimi et al., 2012),
- ▷ reinforcement learning, robotics, and control (Lizotte et al., 2007; Martinez-Cantin et al., 2007; Brochu et al., 2009; Calandra et al., 2014),
- ▷ environmental monitoring and sensor networks (Garnett et al., 2010; Srinivas et al., 2010; Marchant and Ramos, 2012),
- ▷ advertising and recommender systems (Scott, 2010; Chapelle and Li, 2011),
- ▷ interactive user interfaces (Brochu et al., 2007, 2010),
- ▷ combinatorial optimization (Hutter et al., 2011; Wang et al., 2013b),
- ▷ adaptive Monte Carlo samplers (Mahendran et al., 2012; Wang et al., 2013a).

The last bullet point is a specific instance of one general area of particular interest to the machine learning community: the automatic tuning of algorithms (Bergstra et al., 2011; Snoek et al., 2012; Swersky et al., 2013; Thornton et al., 2013; Hoffman et al., 2014). In this setting, the objective function is the generalization accuracy (or error) of a learning algorithm, and the inputs are the algorithmic or hyperparameter choices. Whereas these parameters have traditionally been manually tuned and selected by domain experts, this tedious approach is not scalable and is indeed quickly becoming unwieldy, if not impossible, with the increase in complexity of modern learning algorithms. For instance, neural network architectures can be parameterized by the number of layers, the number of units per layer, the type of non-linearity at each layer, etc; while, their learning algorithms also rely on tunable parameters such as learning rates, momentum, and regularization parameters.

The Bayesian approach to optimizing an unknown objective function is to model the uncertainty over the function as randomness. With a probabilistic model for the objective, we have the powerful tools of probability theory at our disposal, as we will see in the next chapter. An agent can

then use the model as a surrogate for the objective function and carefully select where the next data point should be acquired through what is called a selection policy. These policies leverage both the model prediction and its predictive uncertainty to negotiate local and global search. This constant struggle is commonly referred to as the exploration-exploitation trade-off and is a recurring theme in optimization, machine learning, and artificial intelligence. The need for uncertainty quantification for such effective policies cannot be over-stressed; without uncertainty estimates—either Bayesian or frequentist—only very rudimentary exploration is possible, *e.g.,* ϵ-greedy algorithms. Indeed, the performance of the selection policy, *i.e.* how quickly it is able to approach the global maximizer in practice, hinges on precisely how the surrogate model prediction and uncertainty are utilized. In Chapter 3, we formally introduce selection policies and review several popular alternatives that are available in the literature.

1.1 Model-based optimization and the Bayesian approach

Consider the problem of finding a global maximizer $\mathbf{x}_\star \in \mathcal{X}$ of a given objective function $f : \mathcal{X} \mapsto \mathcal{Y}$. For the sake of generality, let us be deliberately vague about the domain, range, and the function space $\mathcal{H}$ to which f belongs, and simply express the problem as follows for now:

$$\text{find } \mathbf{x}_\star \in \arg\max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}). \tag{1.1}$$

Algorithm 1 is a very general sequential search framework, which includes many popular algorithms such as gradient ascent and Newton's method. In the following examples, we emphasize how a common optimization algorithm fits into the generic sequential scheme by using a local model of the function along with a selection criterion, yielding the next iterate.

Example 1 (Newton's method)
Assuming a deterministic function evaluations which output both the gra-

Algorithm 1 Generic sequential model-based optimization scheme

Require: –
f – unknown noisy objective function
$\mathcal{X}$ – input domain
α_t – sequence of search criteria
T – fixed query budget

1: **for** iteration t up to T **do**
2: $\quad \mathbf{x}_{t+1} \leftarrow \arg\max_{\mathcal{X}} \alpha_t$ ◁ selection
3: $\quad y_{t+1} \leftarrow f(\mathbf{x}_{t+1})$ ◁ observation
4: $\quad$ update model given $(\mathbf{x}_{t+1}, y_{t+1})$ ◁ modelling
5: **end for**
6: **return** $\hat{\mathbf{x}}_T$ ◁ recommendation

dient $\mathbf{g}_t$ and Hessian $\mathbf{H}_t$ evaluated at $\mathbf{x}_t$, the next point $\mathbf{x}_{t+1}$ is selected by maximizing the criterion

$$\alpha_t^{\mathrm{NM}}(\mathbf{x}) := f(\mathbf{x}_t) + \mathbf{g}_t(\mathbf{x} - \mathbf{x}_t) + \frac{1}{2}(\mathbf{x} - \mathbf{x}_t)^\top \mathbf{H}_t(\mathbf{x} - \mathbf{x}_t), \tag{1.2}$$

which is simply the quadratic local approximation given by Taylor's theorem, assuming appropriate continuity and differentiability.

Of course, in this case, the criterion α^{NM} can be optimized analytically at the cost of inverting the Hessian matrix, yielding the famous Newton iteration scheme:

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \mathbf{H}_t^{-1}\mathbf{g}_t \tag{1.3}$$

The last iterate is returned as the ultimate recommendation $\hat{\mathbf{x}}_T = \mathbf{x}_T$.

In this simple case, the search criterion and the model are one and the same. The quadratic model does not fit historical data $\{\mathbf{x}_s, y_s, \mathbf{g}_s, \mathbf{H}_s\}_{s<t}$, nor does it feature uncertainty quantification, which is why this algorithm makes greedy local moves. While it very quickly converges to a local optimizer close to the initial $\mathbf{x}_0$, it is well known that such a method must be randomly restarted for good performance on multimodal objective functions.

In contrast, Bayesian optimization is a probabilistic approach to this model-based optimization scheme. Figure 1.1 provides a visualization of

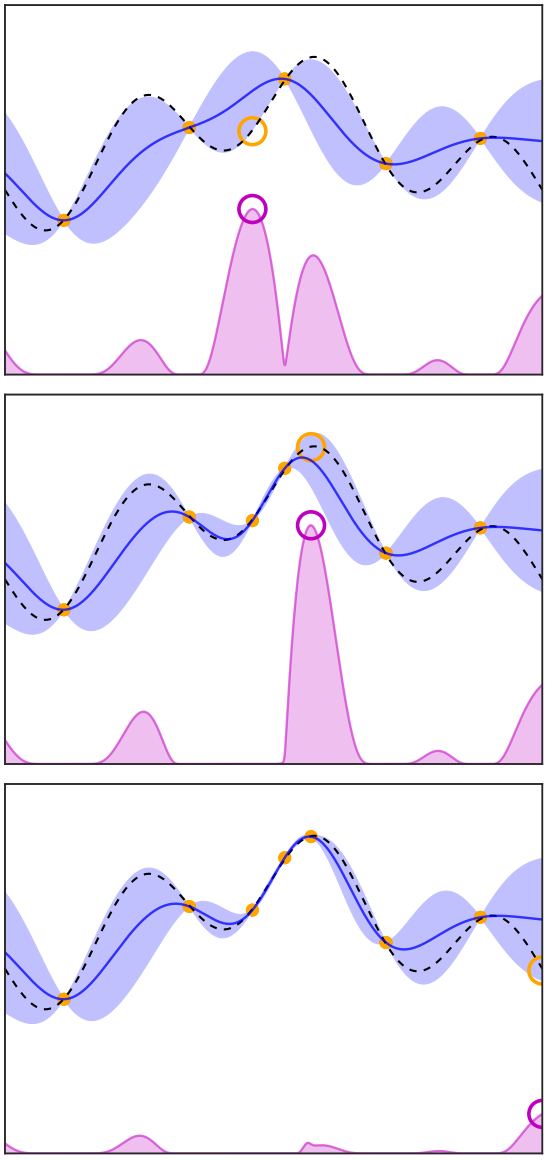


Figure 1.1: Visualization of three consecutive iterations of Bayesian optimization on a noiseless objective function (black dashed curve). As this figure is meant to provide a visual intuition, the notions of probabilistic model (blue curve and shaded area) and acquisition function (purple curve) remain purposely nebulous for now. While observed data is shown as orange dots, the purple circles indicate the maximum of the acquisition function, and the orange circles indicate the next observation.

three consecutive iterations in the sequential algorithm. In particular, we prescribe a prior belief over the possible objective functions f and then sequentially refine this model, as data are observed, via Bayesian posterior updating. The Bayesian posterior represents our updated beliefs, given *all* our observations, on the likely objective function we are optimizing. Equipped with this probabilistic model, we can sequentially induce criteria $\alpha_t : \mathcal{X} \mapsto \mathbb{R}$, similar to those in Example 1 yet more sophisticated in that they leverage the uncertainty in the posterior to guide a global exploration. In the Bayesian optimization literature, these criteria are known as acquisition functions. As we will clarify in Chapter 3, selection policies need not take the form of an acquisition function to be maximized. Though they are perhaps the most ubiquitous form of selection policy, in principle any procedure that produces an $\mathbf{x} \in \mathcal{X}$ is a selection policy, the simplest example being uniformly random search.

1.2 Historical review

Arguably the earliest work related to Bayesian optimization was that of William Thompson in 1933 where he considered the likelihood that one unknown Bernoulli probability is greater than another given observational data (Thompson, 1933). In his article, Thompson argues that when considering, for example, two alternative medical treatments, one should not eliminate the worst one based on a single clinical trial. Instead, he proposes, one should estimate the probability that one treatment is better than the other and weigh future trials in favour of the seemingly better treatment, while still, occasionally, trying the seemingly suboptimal one. Thompson rightly argues that by adopting a single treatment following a clinical trial, there is a fixed chance that all subsequent patients will be given suboptimal treatment. In contrast, by dynamically selecting a fraction of patients for each treatment, this sacrifice becomes vanishingly small.

In modern terminology, Thompson was directly addressing the exploration–exploitation trade-off, referring to the tension between selecting the best known treatment for every future patient (the greedy strategy) and contin-

uing the clinical trial to more confidently assess the quality of both treatments. This is a recurring theme not only in the Bayesian optimization literature, but also in the related fields of sequential experimental design, multi-armed bandits (whose arms correspond to differing available values of $\mathbf{x}$), and operations research.

Although modern experimental design had been developed a decade earlier by Ronald Fisher's work on agricultural crops, Thompson introduced the idea of making design choices dynamically as new evidence becomes available. This is a general strategy known as *sequential experimental design* or, in the multi-armed bandit literature, *adaptive* or *dynamic allocation rules* (Lai and Robbins, 1985; Gittins, 1979).

Meanwhile, a Master's student from the University of the Witwatersrand introduced a method now known in the resource exploration community as Kriging, inheriting its name from its original author: Krige (1951). Not only was the Kriging technique hugely successful in the mining and natural resource exploration community, but seemingly independently, a similar approach originated from a global optimization perspective through the early work of Kushner (1964); Močkus et al. (1978) which later influenced the field of design and analysis of computer experiments (DACE; Sacks et al., 1989; Jones et al., 1998; Jones, 2001).

1.3 Thesis contributions

This document is the compilation of multiple works with the common theme of addressing open practical challenges in Bayesian optimization, particularly as they relate to tuning hyperparameters of machine learning algorithms. In this section, we discuss a few motivating issues with Bayesian optimization and the contributions made towards addressing these open questions.

1.3.1 Pure exploration in multi-armed bandits

Many acquisition functions that are commonly used in Bayesian optimization have either no guarantee or guarantees with respect to a performance metric known as *cumulative regret* (*e.g.* Srinivas et al., 2010; Russo and Van Roy, 2014a). This metric is not suitable for the task of optimization because it rewards and penalizes intermediate selections. Since the task of Bayesian optimization is to find a global optimizer after a fixed budget of function evaluations[2], the *simple regret* is more appropriate because it only assesses the quality of the final recommendation; the selection strategy (the acquisition function) is therefore free to explore within the allocated budget.

Drawing from the multi-armed bandit literature (see Cesa-Bianchi and Lugosi, 2006, for a good introduction), particularly the pure exploration setting, we developed a Bayesian extension to a new bandit algorithm by Gabillon et al. (2012).

This was joint work with Matthew W. Hoffman (Hoffman et al., 2014). In this work, we discretized two Bayesian optimization tasks reformulating them as multi-armed bandit problems, and demonstrated the superior performance of our method, BayesGap, over existing acquisition functions on two real world problems: one application in traffic control and one in automatic model selection and parameter tuning.

Finally, working with noisy bandit feedback for the task of automatic model tuning and selection requires defining an objective, or reward, function. Strictly speaking, this objective function must return independent and identically distributed evaluations (these are considered random draws because they can be noise-corrupted). Therefore in running experiments for this work, we developed an experimental protocol based on a method called repeated learning-testing (see Burman, 1989, for a study of similar methods).

[2] Actually, one could also consider the same optimization problem with a fixed confidence requirement, *i.e.* the stopping criterion is defined in terms of our confidence in our recommendation. This was recently considered by Soare et al. (2014)

1.3.2 Portfolio methods

When considering Bayesian optimization as a solution, no single acquisition strategy[3] has been found to perform best on all problem instances. Hoffman et al. (2011) address this issue by proposing the use of a portfolio containing multiple acquisition strategies. The important new element of this approach is the meta-criterion by which a portfolio method selects among different strategies. This function is analogous to an acquisition function, but at a higher level. Whereas acquisition functions assign utility to points in the input space, a meta-criterion assigns utility to candidates suggested by its member acquisition functions. In the concluding chapter, we also propose a unifying view of these portfolio methods and the aforementioned BayesGap approach.

The meta-criteria considered by Hoffman et al. (2011) are based on adversarial bandit algorithms like Hedge and Exp3. These meta-criteria rely on the average performance of the member acquisition functions' past selections to decide which to select. Instead, my work on the entropy search portfolio (ESP) considers each candidate purely on how much information it is likely to provide towards locating the optimum. This work leveraged new ideas from information guided search (Villemonteix et al., 2009; Hennig and Schuler, 2012). This was joint work with Ziyu Wang and Matthew W. Hoffman (Shahriari et al., 2014) and we demonstrated the good performance of ESP across various problem domains. Interestingly, given the quality of the acquisition functions included in our portfolios we also observed surprisingly good performance with a random portfolio, which supported our claim that mixing strategies in portfolios is beneficial.

Finally, another major contribution of this work is the evaluation of an approximation to Thompson sampling for the continuous domain. This acquisition function, also known as randomized probability matching or pos-

[3] In a slight abuse of terminology, we will often use the terms acquisition function, strategy, or policy interchangeably. Similarly, selection strategy or policy also refer to the same concept. However, these should *not* to be confused with the *recommendation* strategy or policy which outputs the optimization algorithm's final estimate of the optimum. These concepts will be defined more precisely in Chapter 3.

terior sampling, had thus far been constrained to the discrete multi-armed bandit setting, but recent work by my co-author extended this acquisition function to the continuous domain. Our work in (Shahriari et al., 2014) was the first empirical evaluation of the method, which performed very well.

1.3.3 Unconstrained Bayesian optimization

Setting search domain bounds is often described as one of the main difficulties that hinders broader use of Bayesian optimization as a standard framework for hyperparameter tuning. For example, this obstacle was raised more than once at the NIPS 2014 Workshop on Bayesian optimization as one of the remaining practical hurdles in the way of more widespread adoption. While it is often possible to prescribe a reasonable range for each variable that is being optimized, treating those ranges as hard constraints can have adverse effects. Obviously, when the true global optimum of the objective function is outside of those ranges, there is no way to recover from a, possibly ill-informed, initial range. Even when the optimum lies within the bounding box (within all specified ranges) many current acquisition functions will often focus on the corners and walls of the bounding box. Corners and walls correspond to at least one and potentially many variables being in one of the two most extreme possible values. This undesired artifact becomes more and more pronounced in higher dimensions.

Our most recent work, detailed in Chapter 6, proposes an approach that accommodates unconstrained search (Shahriari et al., 2016a). By prescribing a regularizing prior mean, the search is weakly focused around the regularizer centre but is otherwise unconstrained. Our proposed method is practical and easy to add to any existing Bayesian optimization toolbox that is based on Gaussian process priors over objective functions.

1.3.4 Open-source software

Finally, many of the acquisition functions and techniques used in this thesis are included in a python package that we have released on github, and still actively develop. These repositories (`reggie` for Gaussian process models

and `pybo` for Bayesian optimization policies) include demos to facilitate broad adoption among the community. The python package `benchfunk` includes many synthetic benchmarking functions as well as python wrappers to facilitate running `pybo` on black-boxes that are not necessarily written in python. In fact, our `Interactive` wrapper allows the black-box evaluation to be entered manually by, say, a lab technician, in chemical or biological experiments for instance.

1.4 Thesis outline

Since model-based optimization solutions are composed of two main modules, namely modelling and decision-making, we will structure our review of background material into two corresponding chapters. In particular, Chapter 2 provides necessary background on probabilistic models, both the observation model, also known as the measurement or noise model, and the surrogate model, which encapsulates our belief about the unknown objective function f. On the other hand, Chapter 3 formally introduces the notion of a selection policy and acquisition functions before reviewing some common algorithms used in practice.

Chapter 4 presents our work on BayesGap, a Bayesian bandit algorithm for pure exploration. The theorem guaranteeing exponentially vanishing probability of error is stated in this chapter and the proof is relegated to Appendix A. Chapter 5 introduces the details of the entropy search portfolio (ESP) and the experiments we ran to evaluate its performance. Chapter 6 describes our latest work on unconstrained Bayesian optimization, providing intuition and empirical validation. Finally, Chapter 7 features a discussion of our contributions in the scope of the current state of Bayesian optimization as well as possible research directions for the future.

Chapter 2

Modelling

This chapter concerns fitting probabilistic models to observed data. We separate our models into two subcomponents: the observation model and the surrogate model. The former is responsible for simulating the data generation process, while the latter maintains our knowledge of the unknown underlying objective function.

Before we rigorously define any probabilistic model, we need a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, equipped with a set of outcomes Ω, a σ-algebra of measurable events $\mathcal{F}$, and a probability measure $\mathbb{P} : \mathcal{F} \mapsto [0, 1]$ associating a probability to each event in $\mathcal{F}$. See (Rosenthal, 2006) for a thorough review of basic probability theory. There exist two kinds of probability models, namely parametric and nonparametric distributions. The defining feature of a parametric distribution is that it is characterized by a fixed number of parameters, as opposed to nonparametric models, which require storage that increases with the amount of data they are fit to. While we only consider parametric observation models in this work, we consider both parametric and nonparametric surrogate models in Sections 2.2 and 2.3, respectively.

We begin this chapter with a section introducing a unifying probabilistic generative observation model for the generalized stochastic bandit, highlighting some useful examples. Next, in Section 2.2 we detail a few ways to invert these forward models to obtain a surrogate for the unknown objective function f. Finally, in Section 2.3, we introduce Gaussian process (GP)

regression: the main surrogate model of interest in this thesis.

2.1 Observation models

Since their introduction nearly a century ago (Thompson, 1933), multi-armed bandits have been extended to continuous spaces, bridging the gap between the bandit literature and that of efficient global optimization of stochastic black-box functions. We note the particularly relevant work of Srinivas et al. (2010) on GP bandits and that of Bubeck et al. (2011), who coined the term $\mathcal{X}$-armed bandits.

Due to this unifying view, we use a general notion of a stochastic bandit, which includes stochastic black-boxes with continuous input sets, such as GP bandits, and stochastic multi-armed bandits with discrete and finite input sets. Our interest in stochastic bandits in this opening section is purely as forward generative models responsible for producing the observed data. Therefore, we assume a fixed and knowable objective function f.

Definition 1 (Stochastic bandits)
Let $\mathcal{X}$ and $\mathcal{Z}$ denote input and latent sets, respectively. Let $(\mathcal{Y}, \mathcal{F}_Y)$ be a measurable set of outputs, and let $\mathcal{P}_Y$ denote the space of valid probability measures on the σ-algebra $\mathcal{F}_Y$. Finally, let $\theta_y \in \Theta_y$ be a fixed parameter. Then the *stochastic bandit* (f, ν^{θ_y}) is characterized by

- ▷ a fixed underlying objective function $f : \mathcal{X} \mapsto \mathcal{Z}$ and
- ▷ a parametric observation model $\nu^{\theta_y} : \mathcal{Z} \mapsto \mathcal{P}_Y$.

Stochastic bandit feedback: When an agent selects an input $\mathbf{x} \in \mathcal{X}$, the bandit performs the following:

$$z_{\mathbf{x}} = f(\mathbf{x}), \tag{2.1}$$

$$Y_{\mathbf{x}} \sim \nu^{\theta_y}(z_{\mathbf{x}}), \tag{2.2}$$

where a realization of the random variable $Y_{\mathbf{x}} : \Omega \mapsto \mathcal{Y}$ is returned but $z_{\mathbf{x}}$ is kept hidden from the agent, hence the "latent" qualifier.

When the input space $\mathcal{X}$ is discrete and finite, the bandit is commonly known as a multi-armed bandit, and the input space is often denoted $\mathcal{A}$, referring to the available arms, instead of $\mathcal{X}$. In what follows, $a \in \mathcal{A}$ refers specifically to an arm of a multi-armed bandit, whereas $\mathbf{x} \in \mathcal{X}$ refers to general inputs. In the multi-armed bandit setting, the idiom *pulling an arm* translates to *selecting an input*.

The term *objective* for f is deliberately general, encompassing both risk and reward functions. In the hyperparameter tuning setting for instance, we seek to minimize generalization error through noisy evaluations Y of empirical test error on minibatches of available data (*e.g.,* see Section 4.3.2). On the other hand, in reinforcement learning, an agent seeks to maximize her observed rewards. One obvious and important difference between the bandit setting and reinforcement learning is that in Definition 1 selections made by the agent do not change the input space or the reward function, whereas in reinforcement learning, picking an action can take the agent to a different state where the available actions and the underlying reward function may differ from those of the previous time step.

Note that selecting an input $\mathbf{x}$ in Definition 1, yields a realization of the single random variable $Y_{\mathbf{x}} \in \mathcal{Y}$, as opposed to an observation of the full process $\{Y_{\mathbf{x}}\}_{\mathbf{x}\in\mathcal{X}}$. This type of interaction, with partial information, is called *bandit feedback*. The *stochastic* qualifier distinguishes this type of bandit from *adversarial* bandits, where the rewards do not come from a distribution (known or unknown). This thesis will only concern stochastic bandit feedback, but the reader is referred to the very good compilation of both partial and full information results in the adversarial, also known as non-stochastic, setting by Auer et al. (1995).

Though Definition 1 allows for a large variety of observation models, in this thesis we will only consider two types, binomial and Gaussian, which we detail next.

Example 2 (Binomial bandits)
In the binomial bandit, the possible outputs are $\mathcal{Y} = \{0, \dots, n\}$, the latent set is $\mathcal{Z} = [0, 1]$, and $n \in \Theta_y = \mathbb{N}$ is a fixed, known parameter. Selecting an

input $\mathbf{x} \in \mathcal{X}$ results in an observation

$$Y_{\mathbf{x}} \sim \nu^n(z_{\mathbf{x}}) = \text{Binomial}(n, z_{\mathbf{x}}), \tag{2.3}$$

where $z_{\mathbf{x}} = f(\mathbf{x})$.

Note that, while both sets are embedded in the real line, strictly speaking, $\mathcal{Y} \neq \mathcal{Z}$, which motivates the need for distinct output and latent sets. The special case of a binomial bandit with $n = 1$ is called the Bernoulli bandit.

As we will see in Example 4, in the multi-armed binomial bandit, if we assume each z_a is sampled from an independent beta distribution for all $a \in \mathcal{A}$, the conjugacy of the beta prior and binomial observations make this forward model particularly amenable to analytic Bayesian inversion.

In the binomial bandit, all Y_a are finite integer-valued random variables, bounded above by the fixed number of trials n in the forward model ν^n. These are useful black-boxes for modelling such observations as drug success rates in clinical drug trials, click-through rates in split-testing experiments, correct classifications in hyperparameter tuning experiments, etc.

The stochastic bandit we consider next is a forward model that produce real-valued observations, which are better suited for modelling such observations as empirical test error for tuning regression algorithms, temperature measurements, and many more. Example 3 details how Gaussian bandits fit into the stochastic bandit formalism.

Example 3 (Gaussian bandits)
In the Gaussian bandit, the output and latent space are both the same compact subset $\mathcal{Y} = \mathcal{Z} \subseteq \mathbb{R}$, the observation model is additionally parameterized by the *signal noise variance* $\eta^2 \in \Theta_y = (0, \infty)$. Selecting an input $\mathbf{x} \in \mathcal{X}$ results in an observation

$$Y_{\mathbf{x}} \sim \nu^{\eta^2}(z_{\mathbf{x}}) = \mathcal{N}(z_{\mathbf{x}}, \eta^2), \tag{2.4}$$

where $z_{\mathbf{x}} = f(\mathbf{x})$.

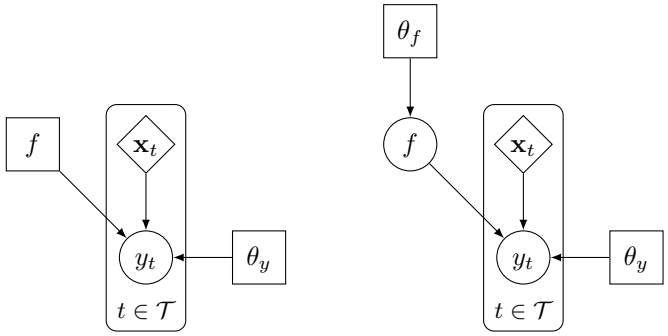


Figure 2.1: Probabilistic graphical models of bandit feedback in the stochastic (left) and the Bayesian (right) settings. Diamond shaped nodes indicate agent choices, while square and round nodes represent deterministic and random variables, respectively.

In most cases of interest, the objective function is unknown and one natural way to model our uncertainty about the function f is by treating it as random. This type of randomness is sometimes called *epistemic*, contrasting it from the uncertainty in the observations Y, for example, which is *aleatoric*. Roughly speaking, the former refers to uncertainty in a quantity that is knowable, whereas the latter describes uncertainty that is inherent. For instance, we will often assume that there exists a true objective function f which an oracle could reveal to us. On the other hand, the observations Y are inherently random due to emission or measurement noise. Of course, the line becomes blurred when we ask whether the source of noise is itself knowable, which is why we will not venture any deeper into the philosophical differences between the two types of uncertainty. Indeed, from a probability-theoretic perspective, the two types of randomness are equivalent.

Therefore in true Bayesian fashion, for the rest of the thesis, we assume the objective function is a realization of a random function $F : \Omega \mapsto \mathcal{H}$, where $\mathcal{H} \subset \{u : \mathcal{X} \mapsto \mathcal{Z}\}$ is some function space. We then prescribe an abstract prior distribution Π over F. It is abstract because, as we will see in the next section, it is only defined via the random process $\{Z_{\mathbf{x}} : \Omega \mapsto \mathcal{Z}\}_{\mathbf{x} \in \mathcal{X}}$ at arbitrary finite collections of points such that $Z_{\mathbf{x}} \equiv F(\cdot)(\mathbf{x})$, for all $\mathbf{x} \in \mathcal{X}$.

Note that since we require pointwise evalutation of realization of F, then the function space must at least satisfy $\mathcal{H} \subseteq L^\infty(\mathcal{X})$. Moreover, in order to define a $\mathcal{Z}$-valued random process, we need the measurable set $(\mathcal{Z}, \mathcal{F}_Z)$.

Definition 2 (Bayesian bandits)
Let $\mathcal{X}$, $\mathcal{Z}$, Θ_y, and $(\mathcal{Y}, \mathcal{F}_Y)$ be as in Definition 1. Let $(\mathcal{H}, \mathcal{F}_H)$ be a measurable space of functions and let $\theta_f \in \Theta_f$ be an additional fixed parameter. Then the *Bayesian bandit* $(\Pi^{\theta_f}, \nu^{\theta_y})$ is characterized by

- ▷ a distribution over objective functions $\Pi^{\theta_f} : \mathcal{F}_H \mapsto [0, 1]$ and
- ▷ a parametric observation model ν^{θ_y}, as in Definition 1.

Bayesian bandit feedback: When the Bayesian bandit is initialized, an objective function is sampled

$$f \sim \Pi^{\theta_f}, \tag{2.5}$$

and then, whenever an agent selects an input $\mathbf{x} \in \mathcal{X}$, the bandit proceeds as in Definition 1:

$$z_{\mathbf{x}} = f(\mathbf{x}), \tag{2.6}$$
$$Y_{\mathbf{x}} \sim \nu^{\theta_y}(z_{\mathbf{x}}), \tag{2.7}$$

where once again $Y_{\mathbf{x}}$ is observed and $z_{\mathbf{x}}$ is kept hidden.

In principle, we could—and eventually will—similarly treat the parameters $\theta := (\theta_f, \theta_y)$ as unknown and model them as epistemic random variables. However, for the discussion in this and the next chapter, θ is assumed to be fixed and known for a more simple exposition. Nevertheless, Appendix E outlines several ways to estimate or learn this additional parameter from observations via maximum likelihood, empirical Bayes, or Monte Carlo approaches.

2.2 Bayesian inversion with parametric models

The next two sections describe how to draw useful conclusions about the objective function f, given a dataset of observations $\mathcal{D} := \{(\mathbf{x}_s, y_s)\}_{s \leq t}$, with $(\mathbf{x}_s, y_s) \in \mathcal{X} \times \mathcal{Y}$. Depending on the observation model, there are various Markov-like inequalities in the frequentist arsenal that can be used to bound the objective function $f(\mathbf{x})$. While these inequalities often require no additional assumptions other than *iid* outcomes, extracting meaningful bounds requires many data points. When the number of evaluations is not a limiting factor, techniques such as Hoeffding (Maron and Moore, 1993) or empirical Bernstein (Mnih et al., 2008) races, or the more recent optimistic optimization techniques (Auer, 2003) are all applicable. Structural assumptions can help more efficiently propagate information from the data; see for instance the works of Dani et al. (2008); Abbasi-Yadkori et al. (2011) for frequentist approaches in the linear bandit setting, or the works of Munos (2011); Valko et al. (2013); Bubeck et al. (2011), which make various weak smoothness assumptions.

Alternatively, when data is difficult or expensive to acquire and one wishes to avoid unnecessarily repeating queries, the Bayesian framework can be a very attractive alternative. At the cost of making mild *a priori* assumptions on f, stronger conclusions can be drawn from relatively little data, including uncertainty quantification that is consistent with both the observed data and the prior assumptions. In this section we examine some parametric models and defer a discussion of Gaussian processes to the following section.

Let us first consider the discrete K-armed Bayesian bandit (Π_0, ν). In these cases, the random function F and the process $\mathbf{Z} := \{Z_a\}_{a \in \mathcal{A}}$ coincide and are simply multivariate random variables. Our inductive biases are prescribed in the form of a prior distribution Π_0, the density of which we denote $\Pi_0(\mathbf{z})$ in a slight abuse of notation. Then, for fixed $\mathcal{D} = \{(a_s, y_s)\}_{s \leq t}$, Bayesian inversion is a direct application of Bayes' rule, noting the indepen-

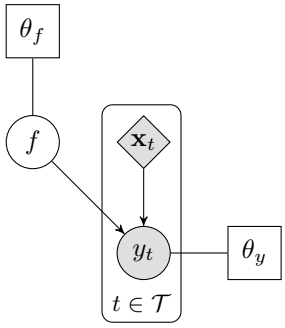


Figure 2.2: Probabilistic graphical model of the Bayesian approach to data fitting. Shaded round nodes are observed random variables and shaded diamond nodes are the corresponding choices $\mathbf{x}$.

dence of all each Y_a given a and z_a:

$$\Pi_t(\mathbf{z}) \propto \Pi_0(\mathbf{z}) \prod_{s=1}^{t} \nu(y_s; z_{a_s}), \tag{2.8}$$

where we omit the parameters θ for simplicity. The updated belief about $\mathbf{Z}$ given data $\mathcal{D}$, on the left hand side of Equation (2.8), is called the *posterior distribution*. The two factors on the right hand side of Equation (2.8) denote the prior density and the likelihood, respectively. When the corresponding two distributions form a conjugate pair, the posterior is in the same family as the prior distribution, and is readily derived analytically, see Example 4.

Example 4 (Beta-binomial multi-armed bandit inversion)
Consider the K-armed binomial bandit with n trials defined in Example 2, replacing $\mathcal{X}$ with a finite and discrete set of K arms $\mathcal{A}$. The objective function is multivariate a realization $\mathbf{z} \in [0,1]^K$ from some prior belief distribution.

Prescribe K independent beta prior distributions $Z_a \sim \Pi_0^{1,1} = \text{Beta}(1,1)$ for all $a \in \mathcal{A}$, *i.e.* $\theta_f = (1,1)$. As promised in Example 2, the conjugacy of the beta and binomial distributions yields an analytic expression with efficient computation. In particular, after observing $\mathcal{D} = \{(a_s, y_s)\}_{s=1}^t$ the

product of the densities yields:

$$\Pi_t^{1,1}(\mathbf{z}) \propto \prod_{a=1}^{K} \text{Beta}(z_a; 1, 1) \prod_{s=1}^{t} \text{Binomial}(y_s; n, z_{a_s}) \tag{2.9}$$

$$\propto \prod_{a=1}^{K} 1 \prod_{s=1}^{t} z_{a_s}^{y_s} (1 - z_{a_s})^{n - y_s} \tag{2.10}$$

$$= \prod_{a=1}^{K} z_a^{n_a} (1 - z_a)^{nt - n_a}, \tag{2.11}$$

which is clearly another beta distribution and can readily be normalized to obtain the posterior distribution for each $a \in \mathcal{A}$:

$$Z_a \mid \mathcal{D} \sim \text{Beta}(nt + n_a, 1 + nt - n_a), \tag{2.12}$$

where $n_a := \sum_{\substack{s \leq t \\ a_s = a}} y_s$ is the total number of successes observed for arm a.

Similarly, the Gaussian bandit can be inverted analytically, by prescribing a K-variate normal prior $\Pi_0(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$. In these two simple cases, the index set $\mathcal{A}$ is a subset of $\mathbb{N}$, merely numbering the arms from 1 to $K = |\mathcal{A}|$. However, the formulation in Definition 1 readily accommodates more complex index sets of the form $\mathcal{A} = \{\mathbf{x}_a \in \mathbb{R}^d\}_{a=1}^K$. Indeed, when structured feature vectors $\mathbf{x}_a$ are available for each arm, they allow us to define more complex forward models. Perhaps the simplest and most common way to exploit available feature vectors is through a linear model. These models assume the following structure:

$$f(\mathbf{x}_a) = \mathbf{w}^\mathsf{T} \mathbf{x}_a, \quad \forall \mathbf{x}_a \in \mathcal{A}. \tag{2.13}$$

Such a structure identifies each objective function with a corresponding weight vector $\mathbf{w} \in \mathbb{R}^d$. Once again, prescribing a prior distribution for f is as simple as fixing a multi-variate distribution on the measurable set $\big(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d)\big)$ to which $\mathbf{w}$ belongs. In this case, when it clear from the context, we denote the prior over functions with $\Pi_0(\mathbf{w})$.

Notice that the vector $\mathbf{w}$ may couple the reward distribution of all arms. Indeed, if we place a prior on the parameter $\mathbf{w}$, observing a realization of Y_a changes our posterior on $\mathbf{w}$, which can in turn change the distribution of $Y_{a'} \mid Y_a$ as long as $\mathbf{x}_a^\mathsf{T}\mathbf{x}_{a'} \neq 0$. However, if $d = K$ and the feature vectors are simply the orthogonal axis-aligned unit vectors $\mathcal{A} = \{\hat{e}_a\}_{a=1}^K$, then $\mathbf{w} \equiv \mathbf{z}$ and we recover the independent K-armed bandit, as in the previous examples in this chapter. Meanwhile, in the following examples we highlight two commonly studied bandits where such additional structure in the index set can be exploited.

Example 5 (linear-Gaussian multi-armed bandit inversion)
Given the K-armed Gaussian bandit in Example 3 with d-dimensional feature vectors $\mathbf{x}_a,\ \forall a \in \mathcal{A}$, and assume the linear structure in Equation (2.13).

Prescribe the d-variate normal prior distribution $\Pi_0^{\mathbf{w}_0,\mathbf{\Sigma}_0} = \mathcal{N}(\mathbf{w}_0, \mathbf{\Sigma}_0)$. Once again, from Bayes' rule it follows that, given observations $\{(\mathbf{x}_{a_s}, y_s)\}_{s=1}^t$,

$$\Pi_0^{\mathbf{w}_0,\mathbf{\Sigma}_0}(\mathbf{w}) \prod_{s=1}^t \nu^{\eta^2}(y_s; \mathbf{w}^\mathsf{T}\mathbf{x}_{a_s}) = \mathcal{N}(\mathbf{w}; \mathbf{w}_0, \mathbf{\Sigma}_0) \prod_{s=1}^t \mathcal{N}(y_s; \mathbf{w}^\mathsf{T}\mathbf{x}_{a_s}, \eta^2) \tag{2.14}$$

which is proportional to another multi-variate normal density with mean and covariance matrix:

$$\mathbf{w}_t = \mathbf{\Sigma}_t(\mathbf{\Sigma}_0^{-1}\mathbf{w}_0 + \eta^{-2}\mathbf{X}^\mathsf{T}\mathbf{y}) \tag{2.15}$$

$$\mathbf{\Sigma}_t = \left(\mathbf{\Sigma}_0 + \eta^{-2}\mathbf{X}^\mathsf{T}\mathbf{X}\right)^{-1}, \tag{2.16}$$

where the matrix $\mathbf{X}$ and vector $\mathbf{y}$ are the concatenated inputs and outputs, respectively, *i.e.* the sth row of $\mathbf{X}$ is $\mathbf{x}_{a_s}^\mathsf{T}$ and $[\mathbf{y}]_s = y_s$.

Recall that one can learn the signal noise parameter $\theta_y = \eta^2$ via a Bayesian treatment. In particular, in this case prescribing an inverse-gamma prior distribution on η^2 yields an analytic posterior distribution for the joint parameters $(\mathbf{w}, \eta^2)$. Alternatively, the unknown noise variance η^2 can be marginalized to obtain a posterior distribution for $\mathbf{w}$ alone which accounts for the uncertainty in θ_y. Note, however, that this distribution will no longer

be normal.

While the linear-Gaussian bandit is useful in settings where observations are real valued, we need a slightly different model to deal with count observations such as performance of classifiers, *e.g.,* number of correct classifications. The generalized linear model (GLM) extends linear models to such a setting and corresponds to the following structural assumption:

$$f(\mathbf{x}_a) = g^{-1}(\mathbf{w}^\mathsf{T}\mathbf{x}_a), \tag{2.17}$$

where $g : [0,1] \mapsto \mathbb{R}$ is called the *link* function, and is usually chosen for desired analytic or computational convenience and held fixed. Then for fixed link function g, we once again have a correspondence between $\mathbf{w}$ and f, so that in the following example, we may prescribe a prior Π_0 on $\mathbf{w}$ instead of f.

The use of a feature vector $\mathbf{x}_a$—if it is available—allows us to exploit possible structure, which cannot be done, for example, in the beta-binomial model because it assumes all arms are independent.

Example 6 (GLM-binomial bandits)
Consider the multi-armed binomial bandit of Example 2 with d-dimensional features for arms $\mathcal{A} = \{\mathbf{x}_a \in \mathbb{R}^d\}_{a=1}^K$, and assume a GLM structure for f.

Prescribe the d-variate normal prior distribution $\Pi_0^{\mathbf{w}_0,\mathbf{\Sigma}_0} = \mathcal{N}(\mathbf{w}_0, \mathbf{\Sigma}_0)$, then from Bayes' rule it follows that, given observations $\{(\mathbf{x}_{a_s}, y_s)\}_{s=1}^t$,

$$\begin{aligned}
\Pi_t^{\mathbf{w}_0,\mathbf{\Sigma}_0}(\mathbf{w}) &\propto \Pi_0^{\mathbf{w}_0,\mathbf{\Sigma}_0}(\mathbf{w}) \prod_{s=1}^t \nu^n \left(y_s; g^{-1}(\mathbf{w}^\mathsf{T}\mathbf{x}_{a_s})\right) && (2.18)\\
&= \mathcal{N}(\mathbf{w}; \mathbf{w}_0, \mathbf{\Sigma}_0) \prod_{s=1}^t \text{Binomial}(y_s; n, z_{a_s}) && (2.19)\\
&\propto \exp\left(-\frac{1}{2}(\mathbf{w}-\mathbf{w}_0)^\mathsf{T}\mathbf{\Sigma}_0(\mathbf{w}-\mathbf{w}_0)\right) \times && (2.20)\\
&\quad \prod_{a=1}^K g^{-1}(\mathbf{w}^\mathsf{T}\mathbf{x}_a)^{n_a} \left(1 - g^{-1}(\mathbf{w}^\mathsf{T}\mathbf{x}_a)\right)^{nt-n_a}, && (2.21)
\end{aligned}$$

where $n_a = \sum_{\substack{s \leq t \\ a_s = a}} y_s$ is the total number of successes observed for arm a.

Unlike in the beta-binomial case, this expression is not a known distribution and cannot be tractably normalized. Therefore we must resort to approximate inference methods. Markov chain Monte Carlo (MCMC) methods (Andrieu et al., 2003) approximate the posterior with a sequence of samples that, in the limit of large samples, converge to samples from the exact posterior; this is the approach taken in (Scott, 2010) with a probit link function. In contrast, the Laplace approximation fits a Gaussian distribution to the posterior by matching the curvature of the posterior distribution at the mode. For example in (Chapelle and Li, 2011), Bayesian logistic regression with a Laplace approximation was used to model click-throughs for the recommendation of news articles in a live experiment. Moment matching, expectation propagation (EP), or other variational approaches can also be used here.

2.3 Gaussian process priors

Rather than inducing a prior on f by assuming some parametric form $f_{\mathbf{w}}$ and prescribing a prior on the parameters $\mathbf{w}$ as in Examples 6 and 5, we can prescribe a prior on f directly. One simple way to define a distribution over objective functions is through a stochastic process $\{Z_{\mathbf{x}}\}_{\mathbf{x} \in \mathcal{X}}$. In particular, in this thesis we will use the Gaussian process (GP) exclusively. Though the exact GP is a nonparametric model, we discuss several scalable parametric approximations in Appendix E.

After introducing the Gaussian process, we review the GP-bandit and outline the derivation of exact inference under a Gaussian observation model ν. We then review the role of the GP covariance and mean functions, as well as that of the marginal likelihood function used in calibration. Finally, we discuss computational costs.

The Gaussian process $\mathrm{GP}(\mu_0, k_0)$ is fully characterized by its mean function $\mu_0 : \mathcal{X} \mapsto \mathbb{R}$ and its symmetric positive kernel, or covariance function, $k_0 : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}^+$. See (Rasmussen and Williams, 2006) for an extensive

treatment. Consider an arbitrary finite collection of t input points $\{\mathbf{x}_s \in \mathcal{X}\}_{s \leq t}$. In Gaussian process regression, the random vector $\mathbf{Z} := \{Z_{\mathbf{x}_s}\}_{s \leq t}$ is normally distributed as follows

$$\mathbf{Z} \sim \mathcal{N}(\mathbf{m}, \mathbf{K}) \tag{2.22}$$

where the elements of the mean vector and covariance matrix are defined as $m_i := \mu_0(\mathbf{x}_i)$ and $K_{i,j} := k_0(\mathbf{x}_i, \mathbf{x}_j)$, respectively. Let us ground the rest of our discussion in the following concrete example.

Example 7 (Gaussian process bandits with Gaussian observation models) The Gaussian process bandit with Gaussian observation model is the Bayesian bandit (Π_0, ν), where

$$\Pi_0 = \mathrm{GP}(\mu_0, k_0), \tag{2.23}$$

$$\nu = \mathcal{N}(\cdot\,, \eta^2). \tag{2.24}$$

i.e. the function f is assumed to be a sample from $\mathrm{GP}(\mu_0, k_0)$.

Although we do not explicitly indicate it, the functions μ_0 and k_0 are usually parameterized by a $\theta_f \in \Theta_f$. Similarly, we omit the dependence of ν on $\eta^2 \in \Theta_y$. See Appendix E for various ways estimate or learn the joint $\theta = (\theta_f, \theta_y)$.

When an agent selects a set of inputs $\{\mathbf{x}_s\}_{s \leq t}$, the following forward model generates the data

$$\mathbf{Z} \sim \mathcal{N}(\mathbf{m}, \mathbf{K}), \tag{2.25}$$

$$\mathbf{Y} \sim \mathcal{N}(\mathbf{Z}, \eta^2 \mathbf{I}). \tag{2.26}$$

Now consider the realizations $\mathbf{y} = \{y_s\}_{s \leq t}$, which combine with the corresponding inputs to form the set of t observations $\mathcal{D} = \{(\mathbf{x}_s, y_s)\}_{s=1}^t$ and an arbitrary test point $\mathbf{x}$ with objective value $z = f(\mathbf{x})$. According to the Gaussian process assumption, the latent variables $\mathbf{z}$ and z are jointly Gaussian; by the convolution of two Gaussian, the assumption of Gaussian observations leads to $\mathbf{y}$ and z being jointly Gaussian as well; and finally, it follows

from basic Gaussian conditioning that $z \mid \mathbf{y}$ is also normally distributed, with mean and variance depending on the test input $\mathbf{x}$ in the following way

$$\mu_t(\mathbf{x}) = \mu_0(\mathbf{x}) + \mathbf{k}(\mathbf{x})^\mathsf{T}(\mathbf{K} + \eta^2\mathbf{I})^{-1}(\mathbf{y} - \mathbf{m})\,, \text{ and} \tag{2.27}$$

$$\sigma_t^2(\mathbf{x}) = k_t(\mathbf{x}, \mathbf{x})\,, \text{ where} \tag{2.28}$$

$$k_t(\mathbf{x}, \mathbf{x}') = k_0(\mathbf{x}, \mathbf{x}') - \mathbf{k}(\mathbf{x})^\mathsf{T}(\mathbf{K} + \eta^2\mathbf{I})^{-1}\mathbf{k}(\mathbf{x}')\,, \tag{2.29}$$

where $[\mathbf{k}(\mathbf{x})]_i := k_0(\mathbf{x}_i, \mathbf{x})$. Repeating this exercise with any finite set of test points in $\mathcal{X}$ leads to a jointly Gaussian distribution with mean vector specified by the function μ_t and covariance matrix computed from the kernel k_t. Since this satisfies the definition of the Gaussian process, we can assert that the posterior distribution over the objective function $\{Z_\mathbf{x}\}_{\mathbf{x}\in\mathcal{X}} \mid \mathcal{D}$ is also a GP

$$\{Z_\mathbf{x}\}_{\mathbf{x}\in\mathcal{X}} \mid \mathcal{D} \sim \mathrm{GP}(\mu_t, k_t). \tag{2.30}$$

Formally we can say that Gaussian processes are closed under conditioning on Gaussian observations. This is an extremely useful feature of the GP, especially in Bayesian optimization, because it allows us to sequentially condition on more and more data while being guaranteed a Gaussian process posterior surrogate model of the objective function.

The posterior mean and variance evaluated at any point $\mathbf{x}$ represent the model's prediction and uncertainty, respectively, in the objective function at the point $\mathbf{x}$. These posterior functions are key ingredients in selecting the next query point $\mathbf{x}_{t+1}$ as detailed in Chapter 3.

Remark 1 (On differentiability)
Both of the posterior functions can be evaluated at arbitrary test points $\mathbf{x}$; moreover, as long as the prior mean and kernel are differentiable, the following gradients can be computed:

$$\nabla_\mathbf{x}\mu_t(\mathbf{x}) = \nabla_\mathbf{x}\mu_0(\mathbf{x}) + \nabla_\mathbf{x}\mathbf{k}(x)^\mathsf{T}(\mathbf{K} + \eta^2\mathbf{I})^{-1}(\mathbf{y} - \mathbf{m}), \text{ and} \tag{2.31}$$

$$\nabla_\mathbf{x}\sigma_t^2(\mathbf{x}) = 2\nabla_\mathbf{x}k_0(\mathbf{x}, \mathbf{x}) - 2\nabla_\mathbf{x}\mathbf{k}(\mathbf{x})^\mathsf{T}(\mathbf{K} + \eta^2\mathbf{I})^{-1}\mathbf{k}(\mathbf{x}). \tag{2.32}$$

These gradients are useful when maximizing certain acquisition functions.

See Chapter 3 for a review of acquisition functions.

2.3.1 Covariance kernels

In Gaussian process regression, the covariance function k_0 dictates the structure of the objective functions we can fit. For instance, if we expect our objective function to have some periodic structure, we can prescribe a periodic kernel; or if we expect our objective function to be smooth with long range interactions, we can prescribe a heavy tailed highly differentiable radial basis function. While kernels need not be stationary, in this thesis we focus on stationary kernels, which only depend on the difference of their arguments.

Matérn kernels are a very flexible family of stationary kernels, parameterized by a smoothness parameter $\kappa > 0$, so called because samples from a GP with such a kernel are differentiable $\lfloor \kappa - 1 \rfloor$ times (Rasmussen and Williams, 2006). The exponential kernel is a special case of the Matérn kernel with $\kappa = \frac{1}{2}$; and the squared exponential kernel is the limiting kernel when $\kappa \to \infty$. The following are the most commonly used Matérn kernels, labelled by their smoothness parameter.

$$k^{\frac{1}{2}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-r) \tag{2.33}$$

$$k^{\frac{3}{2}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\sqrt{3}r)(1 + \sqrt{3}r) \tag{2.34}$$

$$k^{\frac{5}{2}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\sqrt{5}r)(1 + \sqrt{5}r + \tfrac{5}{3}r^2) \tag{2.35}$$

$$k^{\infty}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp(-\tfrac{1}{2}r^2), \tag{2.36}$$

where $r^2 = (\mathbf{x} - \mathbf{x}')^\top \mathbf{\Lambda}^{-1} (\mathbf{x} - \mathbf{x}')$ and $\mathbf{\Lambda}$ is a diagonal matrix of d squared length scale ℓ_i^2. This family of covariance functions are therefore parameterized by an amplitude σ^2 and d length scale hyperparameters, which will contribute to the set of parameters θ_f. Covariance functions with learnable length scale parameters are also known as automatic relevance determination (ARD) kernels. Figure 2.3 provides a visualization of the kernel profiles, samples from the corresponding priors and posteriors.

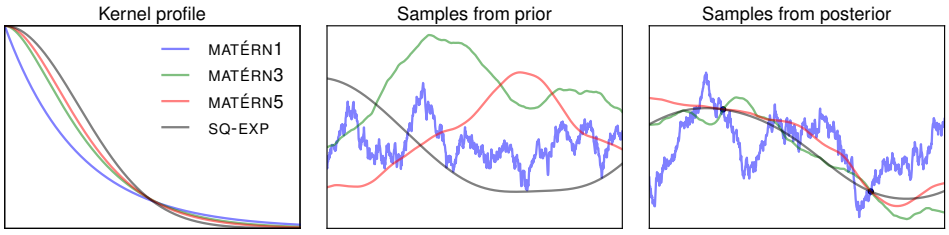


Figure 2.3: Gaussian process visualization. **Left:** Visualization of various kernel profiles. The horizontal axis represents the distance $r > 0$. **Middle:** Samples from GP priors with the corresponding kernels. **Right:** Samples from GP posteriors given two data points (black dots). Note the sharper drop in the Matérn1 kernel leads to rough features in the associated samples, while samples from a GP with the Matérn3 and 5 kernels are increasingly smooth.

2.3.2 Prior mean functions

While the prior kernel function controls the smoothness and amplitude of allowed reward functions, the prior mean provides a way to encode possible known, deterministic behaviour. In practice, this function is set to a constant $\mu_0(\mathbf{x}) \equiv \mu_0$ and inferred from data using learning techniques covered in Appendix E. Unless otherwise specified, in what follows we assume a constant prior mean function for convenience.[1] However, the prior mean function is a principled way to incorporating expert knowledge of likely objective functions or known deterministic behaviour, if they are available. Much of the following analysis can be readily extended to non-constant functions μ_0.

In recent work, the prior mean has been used to steer the search clear from the boundary. In this case, the prior mean acts as a penalty term which favours points within the boundary as opposed to the edges. This problem becomes more pronounced in high dimensions where most of the volume of the search space is in the volume along its boundary. Meanwhile, in our own work, described in detail in Chapter 6, we used the prior mean to remove the dependence on a boundary altogether.

[1]There may also be more compelling arguments for a simple constant mean (see, *e.g.*, Chen et al., 2016).

2.3.3 Marginal likelihood

Another attractive property of the Gaussian process model is that it provides an analytical expression for the marginal likelihood of the data, where marginal refers to the fact that the unknown latent function f is marginalized out. The expression for the log marginal likelihood is simply given by:

$$\begin{aligned}\log p(\mathbf{y} \mid \{\mathbf{x}_s\}_{s\le t},\ \theta) = &-\frac{1}{2}(\mathbf{y}-\mu_0)^{\mathsf{T}}(\mathbf{K}^{\sigma^2,\boldsymbol{\ell}}+\eta^2\mathbf{I})^{-1}(\mathbf{y}-\mu_0)\\ &-\frac{1}{2}\log|\mathbf{K}^{\sigma^2,\boldsymbol{\ell}}+\eta^2\mathbf{I}|-\frac{t}{2}\log(2\pi),\end{aligned} \tag{2.37}$$

where $\theta := (\sigma, \boldsymbol{\ell}, \mu_0, \eta^2)$; and the dependence on θ is finally made explicit by, for example, adding a superscript to the covariance matrix $\mathbf{K}_t^{\sigma^2,\boldsymbol{\ell}}$. The marginal likelihood is very useful in learning the hyperparameters, as we will see in Appendix E. The right hand side of Equation 2.37 can be broken into three terms: the first term quantifies how well the model fits the data, which is simply a Mahalanobis distance between the model predictions and the data; the second term quantifies the model complexity, smoother covariance matrices will have smaller determinants and therefore lower complexity penalties; finally, the last term is simply a linear function of the number of data points t, indicating that the likelihood of data tends to decrease with larger datasets unless the first term compensates accordingly.

Conveniently, as long as the kernel is differentiable with respect to its hyperparameters (not shown in the equation), the marginal likelihood can be differentiated and can therefore be optimized via gradient ascent to obtain an estimate, the so-called type II maximum likelihood (MLII) or empirical Bayes estimate. When data is scarce this can overfit the available data. In Appendix E we will review various practical strategies for learning hyperparameters which all use the marginal likelihood.

2.3.4 Computational cost

Although we have analytic expressions, exact inference in Gaussian process regression is $O(t^3)$ where t is the number of observations. This cost is due

to the inversion of the covariance matrix. In principle, the Cholesky decomposition can be computed once, saved and reused, and cheaply updated via rank-one updating techniques. In practice however the Cholesky decomposition must be recomputed every time the kernel hyperparameters θ are updated, which, depending on the learning approach and/or the learning schedule, happens often enough that the cubic cost is the bottleneck of the Bayesian optimization approach—excluding the cost of evaluating the black-box. Appendix D in the appendix reviews several ideas which propose to reduce this computational burden.

Chapter 3

Decision making

In the preceding chapter, we described some probabilistic models that can be used to represent our belief about the unknown objective function f given a set of t observations $\mathcal{D}_t$. In this chapter, we focus on the mechanisms for selecting the sequence of query points $\{\mathbf{x}_s\}_{s=1}^t$ and the final recommendation $\hat{\mathbf{x}}_t$. Naturally, selecting these points arbitrarily could be wasteful, considering the sophisticated probabilistic models developed in Chapter 2. These surrogate models can be cheaply evaluated in lieu of the objective function f, and their uncertainty quantification can be effectively leveraged to reason about where to query the function next in some optimal sense.

We refer to such mechanisms as policies or, equivalently, strategies, which we use interchangeably. We distinguish two types of policies: the *selection* policy, used to sample the query point $\mathbf{x}_t$ at iteration t in the optimization procedure; and the *recommendation* strategy used to return a point $\hat{\mathbf{x}}_t$ estimating the location of the optimizer if the procedure were stopped after iteration t.

We begin this chapter by formally defining policies and regret in the following section; then minimizing the latter yields the Bayes-optimal policy discussed in the subsequent section. The computationally intractable nature of the Bayes-optimal strategy will lead the discussion towards tractable, myopic, sub-optimal alternatives that are commonly used in model-based optimization.

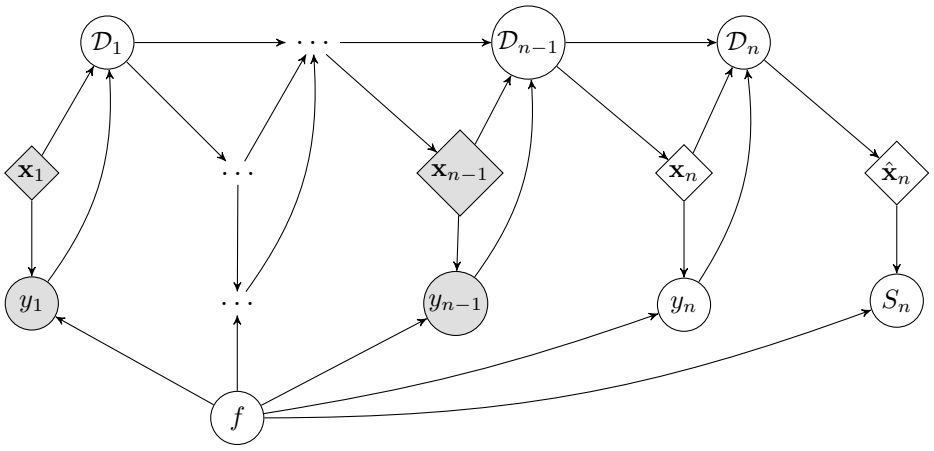


Figure 3.1: Probabilistic graphical model describing the decision problem. All nodes represent random quantities: shaded nodes are observed variables, diamond shaped choice nodes represent (possibly random) decision variables. Hyperparameters θ are omitted for reduced clutter but they are still present, as in Figure 2.1.

3.1 Definitions and notation

Until the current chapter, the inputs $\{\mathbf{x}_s\}$ were considered to be fixed, however in this chapter we will allow them to be random variables by virtue of possible dependence on previous observations but also from a possible external source of randomness—the simplest example being an ϵ-greedy type algorithm. Therefore, for the purpose of this chapter, we will capitalize the selection variables $X_s : \Omega \mapsto \mathcal{X}$ and denote realizations $\mathbf{x}_s$ as before; and similarly for the random recommendations $\hat{X}_s : \Omega \mapsto \mathcal{X}$ and their realizations $\hat{\mathbf{x}}_s$. Finally, we will denote π_s and ρ_s the distribution of X_s and $\hat{X}_s$, respectively, and refer to them as the selection and recommendation policies. In a slight abuse of notation, as it is usually clear from the context, we will use π_s to refer to both the distribution and its density, if it exists, with respect to some base measure.

At each iteration t, the purpose of a policy π_t is to somehow produce a query point $\mathbf{x}_t \in \mathcal{X}$, only using the information $\mathcal{D}_{t-1}$ that is available. Therefore, we define a policy as a sequence of distributions over the measur-

able input space $(\mathcal{X}, \mathcal{F}_X)$, which can be sampled to obtain an input $\mathbf{x}$. Note that the following formulation can also accommodate deterministic selection strategies via the Dirac δ measure. Note however, that even when a policy is a deterministic map from previous observations to a point in $\mathcal{X}$, it inherits the randomness in the observations $\{Y_{\mathbf{x}_s}\}$. Whenever it is obvious from the context, we will denote $Y_s = Y_{\mathbf{x}_s}$ for all s. Figure 3.1 depicts the probabilistic graphical model that describes the decision problem in Bayesian optimization.

Definition 3 (Valid policies)
Given a sequence of growing datasets $\{\mathcal{D}_s\}_{s\geq 1}$, such that $\mathcal{D}_{s-1} \subset \mathcal{D}_s$, one can generate the following filtration $\{\mathcal{F}_s := \sigma(\mathcal{D}_s)\}_{s\geq 1}$. Then

- ▷ a valid selection policy $\pi := \{\pi_t\}$ is a sequence of probability measures such that if $X_t \sim \pi_t$, $X_t \perp f \mid \mathcal{F}_{t-1}$; and
- ▷ a valid recommendation policy $\rho := \{\rho_t\}$ is a sequence of probability measures such that if $\hat{X}_t \sim \rho_t$, $\hat{X}_t \perp f \mid \mathcal{F}_t$.

Intuitively, the independence constraint excludes all policies whose choices depend on the objective function other than through available data. Notice the recommendation strategy has access to one additional observation. Strictly speaking, a policy includes both a selection policy π and a recommendation strategy ρ, but we will often drop the qualifiers when it is clear which type of policy we are referring to—either from the context or notation. Before reasoning about the optimality, or even the performance, of policies, let us define the following useful operator.

Definition 4 (Gap operator)
Let $u \in \mathcal{H}$ be an objective function in a function space $\mathcal{H}$, *e.g.* a reproducing kernel Hilbert space. The gap operator $\Delta : \mathcal{H} \mapsto \mathcal{H}$ is a non-linear operator such that the *gap quantity associated with* u is defined pointwise as:

$$\Delta u(\mathbf{x}) := \max_{\mathcal{X}} u - u(\mathbf{x}). \tag{3.1}$$

If f is the true objective function, the function Δf measures the sub-optimality of a point $\mathbf{x}$. In particular, $\Delta f(\mathbf{x}_\star) = 0$ and $\Delta f(\mathbf{x}) > 0, \forall \mathbf{x} \notin \arg\max_{\mathcal{X}} f$. In a slight abuse of terminology, when it is clear from the context, the function Δf is simply called the gap.

This operator allows us to express the following two notions of regret. Whereas the first evaluates the quality of each selection online, the second only evaluates selections insofar as they help locate a good final recommendation; therefore more experimental selections are not necessarily penalized, in particular, not if they provide valuable information. We will distinguish the tasks of minimizing the former and the latter as *online* and *offline optimization*, respectively.

Definition 5 (Regret)
Given an arbitrary sequence of selections $\{\mathbf{x}_s\}_{s=1}^t$, the **cumulative regret** R_t is the accumulation of the sub-optimality of all selections:

$$R_t := \sum_{s=1}^{t} \gamma^{s-1} \Delta f(\mathbf{x}_s), \quad \gamma \in (0, 1]. \tag{3.2}$$

When the discount factor $\gamma < 1$, R_t is said to be discounted. Note that $\Delta_f(\mathbf{x}_s)$ is sometimes called the **instantaneous regret** at iteration s, denoted r_s.

Finally, given an arbitrary recommendation $\hat{\mathbf{x}}_t$, the **simple regret** S_t quantifies the sub-optimality of a recommendation $\hat{\mathbf{x}}_t$:

$$S_t := \Delta f(\hat{\mathbf{x}}_t). \tag{3.3}$$

Intuitively, when the cumulative regret is asymptotically constant, we have reached the global optimum, and similarly, when simple regret asymptotically vanishes. Lai and Robbins (1985) proved that no policy can achieve better than logarithmic cumulative regret—logarithmic in the total selection budget T. On the other hand, Bubeck et al. (2009) prove that achieving the optimal cumulative regret rate prevents a policy from achieving exponentially vanishing simple regret.

Let us focus on the task of offline optimization—sometimes called the *pure exploration* setting—and briefly discuss Bayes-optimal recommendation and selection strategies.

Recommendation Recall that we wish to maximize the objective function f, which is unknown and modelled as a Gaussian process. Once the final function evaluation budget T is reached, the only quantity we can minimize is the final expected simple regret yielding the following problem:

$$\begin{aligned} \text{minimize} \quad & \mathbb{E}[\Delta f(\hat{\mathbf{x}}) \mid \mathcal{F}_T] \\ \text{s.t.} \quad & \hat{\mathbf{x}} \in \hat{\mathcal{X}}_T, \end{aligned} \tag{3.4}$$

where we allow the set of allowable recommendations $\hat{\mathcal{X}}_T \subseteq \mathcal{X}$ to be a data dependent subset of the entire input set $\mathcal{X}$. While it is perfectly reasonable to set $\hat{\mathcal{X}}_t = \mathcal{X}$ for all $t \in \mathbb{N}$, it is very common to restrict the set of allowable recommendations to the observed queries $\hat{\mathcal{X}}_t = \{\mathbf{x}_s\}_{s=1}^t$ for all $t \in \mathbb{N}$.

We can obtain a straightforward optimal recommendation strategy by expanding Equation (3.5)

$$\mathbb{E}[\Delta f(\hat{\mathbf{x}}) \mid \mathcal{F}_T] = \mathbb{E}[\max_{\mathcal{X}} f - f(\hat{\mathbf{x}}) \mid \mathcal{F}_T] \tag{3.5}$$

$$= \mathbb{E}[\max_{\mathcal{X}} f \mid \mathcal{F}_T] - \mathbb{E}[f(\hat{\mathbf{x}}) \mid \mathcal{F}_T] \tag{3.6}$$

and noticing that the first term does not depend on the recommendation $\hat{\mathbf{x}}$. Hence, the optimal recommendation strategy is

$$\hat{\mathbf{x}}_T \in \arg\max_{\hat{\mathbf{x}} \in \hat{\mathcal{X}}_T} \mathbb{E}[f(\hat{\mathbf{x}}) \mid \mathcal{F}_T] = \arg\max_{\hat{\mathbf{x}} \in \hat{\mathcal{X}}_T} \mu_T(\hat{\mathbf{x}}), \tag{3.7}$$

with ties broken arbitrarily. Note once again that $\mathbf{x}_T$ is a random variable realization by virtue of its dependence on $\mathcal{F}_T$. This recommendation yields the following expression for the simple regret:

$$S_T = \mathbb{E}[\max_{\mathcal{X}} f \mid \mathcal{F}_T] - \max_{\hat{\mathcal{X}}_T} \mu_T. \tag{3.8}$$

Selection Given a finite budget of T function evaluations, naturally the ultimate simple regret S_T is the loss that should guide the exploration. In this Bayesian setting, at any iteration t, we can express our current expectation of the ultimate loss of a given selection policy π via the following nested conditional expectations using the tower property

$$\mathbb{E}\left[S_T \mid \mathcal{F}_{t-1}\right] = \mathbb{E}\left[\mathbb{E}\left[S_T \mid \mathcal{F}_t\right] \mid \mathcal{F}_{t-1}\right]. \tag{3.9}$$

This recursive formulation of the expected ultimate reward allows us to express the deterministic Bayes-optimal selection policy $\pi^\star := \{\pi_t^\star := \delta_{\mathbf{x}_t}\}_{t=1}^T$ recursively as

$$\mathbf{x}_{t+1} \in \arg\min_{\mathbf{x}\in\mathcal{X}} \mathbb{E}\left[\mathbb{E}_{\pi^\star}\left[S_T \mid \mathcal{D}_t \cup (\mathbf{x}, Y_{\mathbf{x}})\right] \mid \mathcal{F}_t\right], \tag{3.10}$$

where ties are broken arbitrarily. The subscript $\pi^\star$ appended to the inner expectation emphasizes that $X_s \sim \pi_s^\star$ for all $s > t$.

Problem (3.10) is severely intractable, both analytically and computationally, in general. In fact, there is currently only one setting with a known computationally tractable optimal sequence due to Gittins (1979):[1] the independent Bernoulli bandit with finite query budget (or infinite budget with discounted rewards). This approach relies on casting the problem as Markov decision process and solving it using dynamic programming. If M denotes the number of possible rewards (e.g., $M = 2$ for Bernoulli arms), the computational complexity of this method is $O(TM^2)$. Therefore, it is difficult to generalize this approach to more complicated, for example continuous, reward distributions.

Consider instead, a strategy that is optimal for the single step problem, *i.e.* the optimal strategy for $T = t + 1$, then once again using the tower property

$$\mathbf{x}_{t+1} \in \arg\min_{\mathbf{x}\in\mathcal{X}} \mathbb{E}\left[\mathbb{E}\left[S_{t+1} \mid \mathcal{D}_t \cup \left(\mathbf{x}, Y_{\mathbf{x}}\right)\right] \mid \mathcal{F}_t\right] \tag{3.11}$$

[1] See also (Gittins et al., 2011).

$$= \arg\max_{\mathbf{x}\in\mathcal{X}} \mathbb{E}\left[\max_{\hat{\mathcal{X}}_t\cup\{\mathbf{x}\}} \mu_{t+1} - \mathbb{E}\left[\max_{\mathcal{X}} f \mid \mathcal{D}_t \cup \left(\mathbf{x}, Y_{\mathbf{x}}\right)\right] \mid \mathcal{F}_t\right] \tag{3.12}$$

where the outer expectation marginalizes the observation $Y_{\mathbf{x}}$, then

$$= \arg\max_{\mathbf{x}\in\mathcal{X}} \mathbb{E}\left[\max_{\hat{\mathcal{X}}_t\cup\{\mathbf{x}\}} \mu_{t+1} \mid \mathcal{F}_t\right] - \mathbb{E}\left[\max_{\mathcal{X}} f \mid \mathcal{F}_t\right] \tag{3.13}$$

$$= \arg\max_{\mathbf{x}\in\mathcal{X}} \mathbb{E}\left[\max_{\hat{\mathcal{X}}_t\cup\{\mathbf{x}\}} \mu_{t+1} \mid \mathcal{F}_t\right] \tag{3.14}$$

where in the final step, the second term drops out because it does not depend on the selection $\mathbf{x}$ and we recover a selection policy known as the Knowledge Gradient (KG; Frazier et al., 2009), often mistaken for expected improvement—a subtly different policy we discuss in the next section—due to their striking similarity. In order to understand how KG obtained its name, we may add the constant S_{t-1} from Equation (3.13), note that this does not change the selection, as S_t is independent of $\mathbf{x}_{t+1}$ since it depends on $\hat{\mathbf{x}}_t$, which must be $\mathcal{F}_t$-measurable. Then, several algebraic manipulations lead to

$$\begin{aligned}\mathbf{x}_{t+1} \in \arg\max_{\mathbf{x}\in\mathcal{X}} \mathbb{E}\left[\max_{\hat{\mathcal{X}}_t\cup\{\mathbf{x}\}} \mu_{t+1} \mid \mathcal{F}_t\right] - \mathbb{E}[\max_{\mathcal{X}} f \mid \mathcal{F}_t] \\ + \mathbb{E}[\max_{\mathcal{X}} f \mid \mathcal{F}_t] - \max_{\hat{\mathcal{X}}_t} \mu_t \qquad (3.15)\\ = \arg\max_{\mathbf{x}\in\mathcal{X}} \mathbb{E}\left[\max_{\hat{\mathcal{X}}_t\cup\{\mathbf{x}\}} \mu_{t+1} \mid \mathcal{F}_t\right] - \max_{\hat{\mathcal{X}}_t} \mu_t. \qquad (3.16)\end{aligned}$$

The first term in (3.16) is simply the expected value of the function at the recommended point after observing the function at the point $\mathbf{x}$, to which we subtract the very same estimate before observation $(\mathbf{x}, Y_{\mathbf{x}})$. Intuitively then, KG selects the point $\mathbf{x}_t$ that is expected to most increases the maximum of our prediction of the objective function f after the observation $(\mathbf{x}_t, Y_{\mathbf{x}_t})$.

Finally, notice that the knowledge gradient policy is a one-step look-ahead Bayesian strategy as it reasons about the expected effect of the next

observation $(\mathbf{x}, Y_{\mathbf{x}})$ before selecting a point. Similarly, many selection strategies can be formulated in a utilitarian framework in which the next point $\mathbf{x}_{t+1}$ maximizes some expected utility, where the expectation is taken with respect to the posterior surrogate model $f \mid \mathcal{F}_t$, and possiply future observations $Y_{t+1}, \ldots, Y_{t+q}$ in the case of a q-step look-ahead strategy.[2]

Consider once again a single-step look-ahead policy such as KG, and define a utility function $U : \mathcal{X} \times \mathcal{H} \times \mathcal{Y} \mapsto \mathbb{R}$ which maps an arbitrary query point $\mathbf{x}$ and its corresponding outcome, modelled by $Y_{\mathbf{x}}$, to a measure of quality of the experiment for a given function $f \in \mathcal{H}$, *e.g.* , how much the query would improve our posterior estimate of the optimum in KG, or how much information this query will provide towards some goal as in (Lindley, 1956). Note that U is a random variable due to the unknown function f and unobserved outcome $Y_{\mathbf{x}}$. Given some data accumulated thus far, we can marginalize both f and $Y_{\mathbf{x}}$ to obtain the expected utility of a query point $\mathbf{x}$:

$$\alpha_{t+1}(\mathbf{x}) = \mathbb{E}\left[U\left(\mathbf{x}, f, Y_{\mathbf{x}}\right) \mid \mathcal{F}_t\right], \tag{3.17}$$

and the selection policy is simply $\pi_t = \delta_{\mathbf{x}_t}$, where $\mathbf{x}_{t+1} \in \arg\max_{\mathcal{X}} \alpha_{t+1}$ and ties are broken arbitrarily. With the exception of predictive entropy search in Section 3.4, we do not consider any look-ahead expected utilities and so the expectations will only be taken over the posterior $f \mid \mathcal{F}_t$ only. Therefore, the utility U will often only take arguments in $\mathcal{X} \times \mathcal{H}$

Whereas in experimental design and decision theory, the function α is called the expected utility or criterion, in Bayesian optimization it is often called the acquisition or infill function. These acquisition functions are carefully designed to trade off exploration of the search space and exploitation of current promising areas. In the next sections, we present traditional improvement-based and optimistic acquisition functions, followed by more recent information-based approaches. We conclude the chapter with a discussion of three recommendation strategies.

[2] For simplicity, in this section we ignore the hyperparameter dependence and we defer a discussion of marginalizing hyperparameters to Section E.

3.2 Improvement-based policies

Improvement-based acquisition functions favour points that are likely to improve upon a target τ, the policies' only hyperparameter. An early strategy in the literature, due to (Kushner, 1964), is commonly known as (maximum) probability of improvement (PI or MPI). As its name suggests, this acquisition function measures the probability, according to the model, that a point $\mathbf{x}$ leads to an improvement upon τ. Since the posterior distribution of $f(\mathbf{x})$ is Gaussian, we can analytically compute this probability as follows:

$$\alpha_{t+1}^{\mathrm{PI}}(\mathbf{x}) := \mathbb{P}\left[f(\mathbf{x}) > \tau \mid \mathcal{F}_t\right] \tag{3.18}$$

Recall that $\alpha_{t+1}^{\mathrm{PI}}$ is then maximized to select the next query point. For this criterion, the utility function is simply an indicator of improvement $U^{\mathrm{PI}}(\mathbf{x}, f) := \mathbb{I}[f(\mathbf{x}) > \tau]$. From this insight it is clear that all improvements are treated equal and PI simply accumulates the posterior probability mass above (or below, when minimizing) τ at $\mathbf{x}$.

Although probability of improvement can perform very well when the target is known, in general the heuristic used for an unknown target τ causes PI to exploit quite aggressively (Jones, 2001).

One could instead measure the expected improvement (EI) which incorporates the *amount* of improvement. This new criterion, due to Močkus *et al.* (Močkus et al., 1978), corresponds to a utility called the improvement function, usually denoted I, and is defined as follows

$$U^{\mathrm{EI}}(\mathbf{x}, f) = I(\mathbf{x}, f(\mathbf{x})) := (f(\mathbf{x}) - \tau)\mathbb{I}[f(\mathbf{x}) > \tau]. \tag{3.19}$$

Note that $I > 0$ only if there is an improvement. These improvement strategies have been empirically studied in the literature (Jones et al., 1998; Jones, 2001) and recently convergence rates have been proven for EI (Bull, 2011).

Remark 2 (Analytic expressions with GP surrogates)
When using a Gaussian process surrogate model, both EI and PI yield the

following analytic expressions

$$\alpha^{\mathrm{PI}}_{t+1}(\mathbf{x}) = \Phi\left(\frac{\mu_t(\mathbf{x}) - \tau}{\sigma_t(\mathbf{x})}\right) \quad \text{and} \tag{3.20}$$

$$\alpha^{\mathrm{EI}}_{t+1}(\mathbf{x}) = (\mu_t(\mathbf{x}) - \tau)\Phi\left(\frac{\mu_t(\mathbf{x}) - \tau}{\sigma_t(\mathbf{x})}\right) + \sigma_t(\mathbf{x})\phi\left(\frac{\mu_t(\mathbf{x}) - \tau}{\sigma_t(\mathbf{x})}\right), \tag{3.21}$$

when $\sigma_t > 0$ and vanish otherwise. Here, Φ and ϕ denote the standard normal cumulative distribution function and probability density function, respectively. Analytic expressions for the derivatives of these acquisition functions are also readily derived in the appendix (Appendix C), provided the covariance kernel is differentiable. The analytic expression above allows very efficient evaluation of the acquisition function and the analytic gradients facilitate using multi-restarted quasi-Newton methods for more efficient maximization.

Remark 3 (On the choice of target τ)
Finally, although the target objective value (i.e., the best achievable objective value) is often unknown, in practice τ is adaptively set to either the best observed value $y^+ := \max_{i=1:t} y_i$ or alternatively the current incumbent $\mu_t^+ := \max_{i=1:t} \mu_t(\mathbf{x}_i)$. Whereas for PI this heuristic can lead to an overly greedy optimization (Jones, 2001), it works reasonably with EI in practice (Snoek et al., 2012). For additional exploration, an additive (or relative) minimum improvement parameter can be added (or multiplied) to y^+ to avoid excessively local selections.

3.3 Optimistic policies

Dating back to the seminal work of Lai and Robbins (1985) on the multi-armed bandit problem, the upper confidence bound criterion has been a popular way of negotiating exploration and exploitation, often with provable cumulative regret bounds. The guiding principle behind this class of strategies is to be optimistic in the face of uncertainty. Indeed, using the upper confidence for every query point $\mathbf{x}$ corresponds to effectively using a fixed probability best case scenario according to the model. Originally, the up-

per confidence was given by frequentist Chernoff–Hoeffding bounds (Auer, 2003), and more sophisticated bounds in the linear bandit setting (Dani et al., 2008; Abbasi-Yadkori et al., 2011).

More recently, the Gaussian process upper confidence bound (GP-UCB; Srinivas et al., 2010) algorithm was proposed as a Bayesian optimistic algorithm with provable cumulative regret bounds. In the deterministic case, a branch-and-bound extension to GP-UCB was proven to have exponentially vanishing regret (de Freitas et al., 2012). The GP-UCB algorithm has since been generalized to other Bayesian models by considering upper quantiles (Kaufmann et al., 2012a) instead of Equation (3.22) defined below, which is more reminiscent of frequentist concentration bounds. In the GP case, since the posterior at any arbitrary point $\mathbf{x}$ is a Gaussian, any quantile of the distribution of $f(\mathbf{x})$ is computed with its corresponding value of β_t as follows:

$$\alpha_{t+1}^{\mathrm{UCB}}(\mathbf{x}) := \mu_t(\mathbf{x}) + \beta_t \sigma_t(\mathbf{x}). \tag{3.22}$$

There are theoretically motivated guidelines for setting and scheduling the hyperparameter β_t to achieve optimal cumulative regret (Srinivas et al., 2010) and, as with τ in the improvement policies, tuning this parameter within these guidelines can offer a performance boost.

Finally, there also exist variants of these algorithms for the contextual bandits (Vanchinathan et al., 2014) and parallel querying (Desautels et al., 2014).

3.4 Information-based policies

In contrast to the acquisition functions introduced so far, information-based policies consider the posterior distribution over the unknown minimizer $\mathbf{x}_\star$, denoted $\mathbb{P}(X_\star \mid \mathcal{D}_t)$, where $X_\star$ is the random variable which models our knowledge of $\mathbf{x}_\star$. This distribution is implicitly induced by the posterior over objective functions f and is sometimes called $\mathbb{P}_\star^t$. There are two policies in this class, namely Thompson sampling and entropy search.

Though it was introduced in 1933 (Thompson, 1933), Thompson sam-

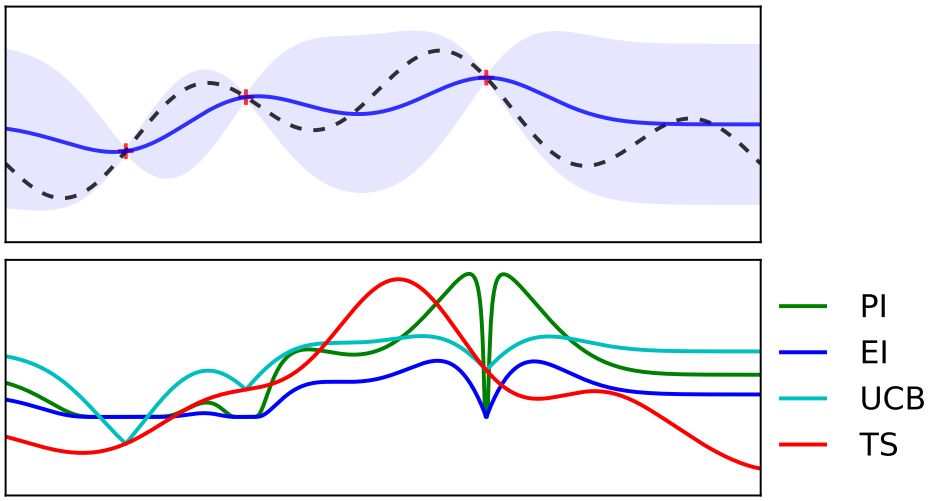


Figure 3.2: Visualizing the surrogate regression model and various induced acquisition functions. **(Top)** The true objective function is shown as a dashed line and the probabilistic regression model is shown as a blue line with a shaded region delimiting the $2\sigma_t$ credible intervals. Finally, the observations are shown as red crosses. **(Bottom)** Four acquisition functions are shown. In the case of PI, the optimal mode is much closer to the best observation as in the alternative methods, which explains its greedy behaviour. In contrast, the randomization in TS allows it to explore more aggressively.

pling has attracted renewed interest in the multi-armed bandit community, producing empirical evaluations (Scott, 2010; Chapelle and Li, 2011) as well as theoretical results (Kaufmann et al., 2012b; Agrawal and Goyal, 2013; Russo and Van Roy, 2014a). Thompson sampling (TS) is a randomized strategy which samples a reward function from the posterior and selects the arm with the highest simulated reward. Therefore the selection made by TS can be expressed as the randomized acquisition function $\mathbf{x}_{t+1} \sim \mathbb{P}^t_\star$.

However, in continuous search spaces, the analog of Thompson sampling is to draw a continuous function $f^{(t)}$ from the posterior GP and optimize it to obtain $\mathbf{x}_{t+1}$. In order to be optimized, the sample $f^{(t)}$ needs to be fixed so it can be queried at arbitrary points; unfortunately, it is not clear how to fix an exact sample from the GP. However, using recent spectral sampling

techniques (Bochner, 1959; Rahimi and Recht, 2007; Lázaro-Gredilla et al., 2010), we can draw an approximate sample from the posterior that can be evaluated at any arbitrary point $\mathbf{x}$ (Hernández-Lobato et al., 2014), which extends TS to continuous search spaces. As an acquisition function, TS can be formulated as

$$\alpha_{t+1}^{\mathrm{TS}}(\mathbf{x}) := f^{(t)}(\mathbf{x}), \text{ where } f^{(t)} \overset{s.s.}{\sim} \mathrm{GP}(\mu_0, k_0 \mid \mathcal{F}_t) \tag{3.23}$$

and $\overset{s.s.}{\sim}$ indicates approximate simulation via spectral sampling. Empirical evaluations show good performance which, however, seems to deteriorate in high dimensional problems, likely due to aggressive exploration (Shahriari et al., 2014).

Instead of sampling the distribution $\mathbb{P}_\star^t$, entropy search (ES) techniques aim to reduce the uncertainty in the location $X_\star$ by selecting the point that is expected to cause the largest reduction in entropy of $X_\star$ (Villemonteix et al., 2009; Hennig and Schuler, 2012; Hernández-Lobato et al., 2014). In terms of utility, entropy search methods are one-step look-ahead policies that use the information gain defined as follows

$$U^{\mathrm{ES}}(\mathbf{x}, Y_{\mathbf{x}}) = H(X_\star \mid \mathcal{F}_t) - H(X_\star \mid \mathcal{D}_t \cup \{(\mathbf{x}, Y_{\mathbf{x}})\}), \tag{3.24}$$

where the unknown function f is already marginalized.

In other words, ES measures the expected information gain from querying an arbitrary point $\mathbf{x}$ and selects the point that offers the most information about the unknown $\mathbf{x}_\star$. The acquisition function for ES can be expressed formally as

$$\alpha_{t+1}^{\mathrm{ES}}(\mathbf{x}) := H(X_\star \mid \mathcal{F}_t) - \mathbb{E}\left[H(X_\star \mid \mathcal{D}_t \cup \{(\mathbf{x}, Y_{\mathbf{x}})\}) \mid \mathcal{F}_t\right]$$

where $H(X_\star \mid \mathcal{F}_t)$ denotes the differential entropy of the posterior distribution $\mathbb{P}_\star^t$, and the expectation is over the distribution of the random variable $Y_{\mathbf{x}} \sim \mathcal{N}(\mu_t(\mathbf{x}), \sigma_t^2(\mathbf{x}) + \eta^2)$.

Once again, this function is not tractable for continuous search spaces $\mathcal{X}$ so approximations must be made. Early work discretized the space $\mathcal{X}$ and

computed the conditional entropy via Monte Carlo sampling (Villemonteix et al., 2009). More recent work uses a discretization of the $\mathcal{X}$ to obtain a smooth approximation to $\mathbb{P}_\star^t$ and its expected information gain (Hennig and Schuler, 2012). This method is unfortunately $O(M^4)$ where M is the number of discrete so-called representer points.

Finally, predictive entropy search (PES) removes the need for a discretization and approximates the acquisition function in $O((t+d)^3)$ time, which, for $t < n$ is of the same order as EI (Hernández-Lobato et al., 2014). This is achieved by using the symmetric property of mutual information to rewrite $\alpha_{\mathrm{ES}}(\mathbf{x})$ as

$$\alpha_{t+1}^{\mathrm{PES}}(\mathbf{x}) := H(Y_{\mathbf{x}} \mid \mathcal{F}_t) - \mathbb{E}\left[H(Y_{\mathbf{x}} \mid \mathcal{F}_t, X_\star) \mid \mathcal{F}_t\right], \tag{3.25}$$

where the expectation is now with respect to the random variable $X_\star$. The expectation can be approximated via Monte Carlo with Thompson samples; and three simplifying assumptions are made to compute $H(Y_{\mathbf{x}} \mid \mathcal{F}_t, X_\star)$, which we do not discuss here but which can be found in (Hernández-Lobato et al., 2014). Empirically, this algorithm has been shown to perform as well or better than the discretized version without the unappealing quartic term, making it arguably the state of the art in entropy search approximation.

3.5 Recommendation strategies

Finally, let us discuss recommendation strategies ρ_t. As its name suggests, at iteration t, ρ_t returns the algorithm's recommendation on where the optimum $\mathbf{x}_\star$ might be, in light of observations $\mathcal{D}_t$. Although recommendation strategies can in principle be stochastic, they are usually deterministic, *i.e.* $\rho_t := \delta_{\hat{\mathbf{x}}_t}$.

If the objective function f is truly a random draw from our prior probabilistic surrogate model, and our observations y_t are in fact derived from $f(\mathbf{x}_t)$ via our likelihood model, then by Bayes' rule, the posterior distribution $f \mid \mathcal{F}_t$ is the correct belief to hold on f. Therefore, the correct recommendation strategy ρ_t^{lat} is to select the input $\hat{\mathbf{x}}$ with the highest latent

surrogate mean $\mu_t(\mathbf{x})$ as follows

$$\hat{\mathbf{x}}_t^{\text{lat}} := \arg\max_{\hat{\mathbf{x}} \in \mathcal{X}} \mu_t(\hat{\mathbf{x}}). \tag{3.26}$$

However, in practice we rarely know the exact form of the prior and likelihood, and even if we do, exact inference may not be tractable such that $\mu_t(\mathbf{x})$ may be an approximation. In such cases of model misspecification or approximate inference, we may prefer to rely on observations rather than the surrogate model, opting for the best observation strategy ρ_t^{obs}:

$$\hat{\mathbf{x}}_t^{\text{obs}} := \arg\max_{\{\mathbf{x}_s\}_{s \le t}} y(\mathbf{x}_s). \tag{3.27}$$

Naturally, if the observations y_t are very noisy, it is not hard to see that ρ_t^{obs} can be mislead. Instead, we can restrict recommendations to inputs for which we have observations and choose between them using the smoothing posterior mean μ_t, rather than the noisy observations y_t, leading to a strategy we refer to as selecting the *incumbent*:

$$\hat{\mathbf{x}}_t^{\text{inc}} := \arg\max_{\{\mathbf{x}_s\}_{s \le t}} \mu_t(\mathbf{x}_s). \tag{3.28}$$

Note that in the noiseless case, ρ_t^{obs} and ρ_t^{inc} will return the same recommendations. Finally, note that practitioners may understandably be reluctant to trust a recommendation $\hat{\mathbf{x}}$ for which the algorithm has no corresponding observation $\hat{y}$ and therefore ρ_t^{inc} is often a good compromise between trusting the model and trusting the data.

In the multi-armed bandit setting, Bubeck et al. (2009) explored various recommendation strategies. While the *empirical best arm* strategy is analogous to ρ_t^{inc}, the other two strategies only make sense with discrete input spaces:[3] namely, the *most played arm* and the *empirical distribution of plays*, which is a randomized recommendation strategy that samples uniformly from the observed inputs $\{\mathbf{x}_s\}_{s \le t}$ with multiplicity.

[3] Strictly speaking, with continuous input spaces, atomic selection policies could allow such recommendation strategies as well.

Chapter 4

Bayesian gap-based exploration

In this chapter, I discuss a Bayesian policy that we developed for the problem of pure exploration in multi-armed bandits. Though it could be awkwardly expressed as an acquisition function, the policy presented in this chapter is not explicitly an acquisition function that is maximized. This work includes a probably approximately correct (PAC) proof and an extensive comparison and empirical study of multiple acquisition functions for pure exploration. Finally, we apply this optimization strategy to the problem of automatic model selection and hyperparameter tuning with good performance. This was joint work with Matthew W. Hoffman.

4.1 Algorithm

In this section we will introduce a gap-based solution to the Bayesian optimization problem, which we call BayesGap. This approach builds on the work of Gabillon et al. (2011, 2012), which we will refer to as UGap,[1] and offers a principled way to incorporate correlation between different arms, whereas UGap assumes all arm reward distributions are independent.

[1] Technically, UGapEb, denoting bounded horizon.

At the beginning of round t we will assume that the decision maker is equipped with high-probability upper and lower bounds $U_k(t)$ and $L_k(t)$ on the unknown mean z_k for each arm $k \in \mathcal{A} = \{1, \ldots, K\}$. While this approach can encompass more general bounds, for the linear-Gaussian bandit we consider in this work we can define these quantities in terms of the posterior mean and standard deviation $\mu_{kt} \pm \beta\sigma_{kt}$. These bounds also give rise to a confidence diameter $s_k(t) = U_k(t) - L_k(t) = 2\beta\sigma_{kt}$. We will refer to β as the exploration constant, and note that in this chapter we will use a slightly different notation $\mu_{kt} := \mu_t(k)$ and $\sigma_{kt} := \sigma_t(k)$.

Given bounds on the mean reward for each arm, we can then introduce the per-arm worst-case gap quantity

$$B_k(t) = \max_{i \neq k} U_i(t) - L_k(t), \tag{4.1}$$

which involves a comparison between the lower bound of arm k and the highest upper bound among all alternative arms. Ultimately this quantity provides an upper bound on the simple regret (see Lemma B1 in the appendix) and will be used explicitly both in the selection policy and its PAC proof. However, rather than directly selecting the arm that minimizes this gap, we will consider the two candidate arms

$$J(t) = \arg\min_{k \in \mathcal{A}} B_k(t) \text{ and} \tag{4.2}$$

$$j(t) = \arg\max_{k \neq J(t)} U_k(t). \tag{4.3}$$

We will then define the deterministic selection policy as

$$a_t = \arg\max_{k \in \{j(t), J(t)\}} s_k(t). \tag{4.4}$$

Intuitively, this strategy selects either $J(t)$, the arm that minimizes our bound on the simple regret, or $j(t)$, the best optimistic arm. We say optimistic because $j(t)$ is the index that maximizes $U_k(t)$, not μ_{kt}, and $U_k(t)$ is an optimistic upper bound on the expected reward of arm k. Between the

Algorithm 2 BayesGap

for $t = 1, \ldots, T$ **do**
  $J(t) \leftarrow \arg\min_{k \in \mathcal{A}} B_k(t)$
  $j(t) \leftarrow \arg\max_{k \neq J(t)} U_k(t)$
  $a_t \leftarrow \arg\max_{k \in \{j(t), J(t)\}} s_k(t)$
  $y_t \sim \nu_{a_t}$
  update posterior μ_{kt} and σ_{kt} given (a_t, y_t)
  update bound on H_ϵ and re-compute β
  update posterior bounds $U_k(t)$ and $L_k(t)$
end for
$\hat{t} \leftarrow \arg\min_{t \leq T} B_{J(t)}(t)$
return $\hat{a}_T \leftarrow J(\hat{t})$

two candidates $J(t)$ and $j(t)$, the arm with the highest uncertainty will be selected: the one expected to provide the most information. Finally, we will define the deterministic recommendation strategy as

$$\hat{a}_T = J\big(\arg\min_{t \leq T} B_{J(t)}(t)\big). \tag{4.5}$$

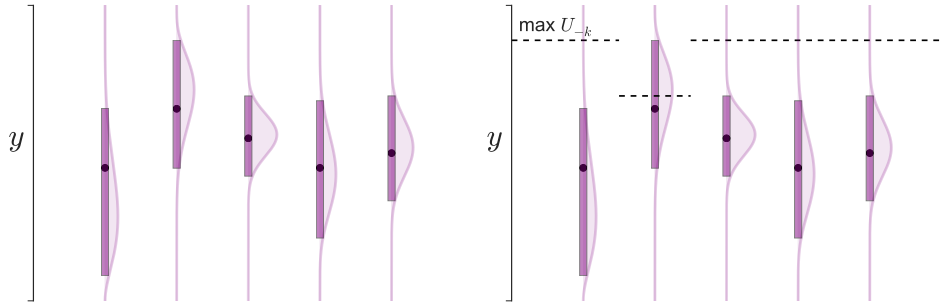
In other words, the recommended arm $\hat{a}_T \in \{J(t)\}_{t<T}$ minimizes the regret bound, over all times $t \leq T$. The reason behind this particular choice is subtle, but is necessary for the proof of the method's simple regret bound.[2] Algorithm 2 outlines the pseudo-code for BayesGap.

4.2 Theory

We now present our theoretical result concerning the case of linear–Gaussian bandits, where we ask which value of β should be used. This will be a particular case of the more general result in Theorem 2 in Appendix A.

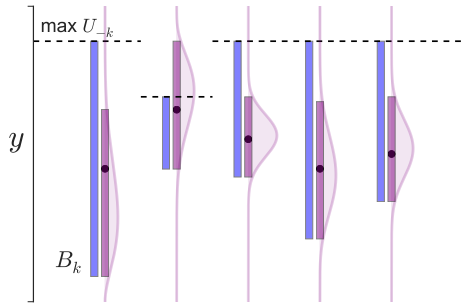
Recall the linear–Gaussian bandit introduced in Chapter 2. Selecting an arm $k \in \mathcal{A}$ from the K-armed Gaussian bandit results in an observing a realization of $\mathcal{N}(z_k, \eta^2)$, where $z_k = f(\mathbf{x}_k)$ and $\mathbf{x}_k \in \mathbb{R}^d$, $\forall k \in \mathcal{A}$ are feature

[2] See inequality (b) in Appendix A.

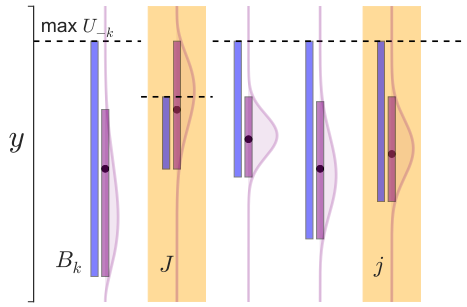


(a) Updated posterior

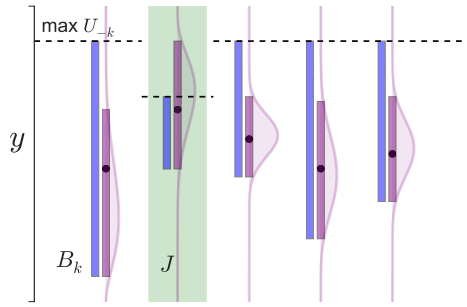
(b) Locate max upper bounds



(c) Compute gap for each arm



(d) Establish candidates (Steps 2–3)



(e) Select most uncertain candidate (Step 4)

Figure 4.1: Visualization of one iteration of the BayesGap algorithm on a 5-armed Bernoulli bandit, with references to Algorithm 2 where appropriate. Each panel includes 5 columns corresponding to the 5 arms. The unknown true values of z_k are represented by black dots, while the posterior distribution of each z_k is represented by magenta curves. The magenta bars are delimited by the bounds L_k and U_k, while the length of the blue bars quantify the gap B_k. The candidate arms J and j, and the selection are highlighted in orange and green, respectively.

vectors. Furthermore, we assume the linear structure of Equation (2.13):

$$f(\mathbf{x}_k) = \mathbf{w}^\mathsf{T}\mathbf{x}_k, \quad \forall \mathbf{x}_k \in \mathcal{A}. \tag{4.6}$$

where we prescribe an isotropic normal prior distribution $\Pi_0^{\mathbf{w}_0,\sigma} = \mathcal{N}(\mathbf{w}_0, \sigma\mathbf{I})$.

Consider the true gap quantities $\Delta_k = |\max_{i\neq k} z_i - z_k|, \ \forall k \in \mathcal{A}$. This quantity is different from the gap quantity defined in Chapter 2 only for the best arm $k_\star$, where rather than vanishing, $\Delta_{k_\star}$ measure the difference between the optimal arm and the runner up. Otherwise, for all other arms it is again a measure of sub-optimality. Given this quantity let $H_{k\epsilon} = \max(\frac{1}{2}(\Delta_k + \epsilon), \epsilon)$ be an arm-dependent hardness quantity; essentially our goal is to reduce the uncertainty in each arm to below this level, at which point with high probability we will identify the best arm. Now, given $H_\epsilon = \sum_k H_{k\epsilon}^{-2}$ we define our exploration constant as

$$\beta^2 = \frac{\frac{T-K}{\eta^2} + \frac{\kappa}{\sigma^2}}{4H_\epsilon}, \tag{4.7}$$

where $\kappa = \sum_k \|\mathbf{x}_k\|^{-2}$. We have chosen β such that with high probability we recover an ϵ-best arm, as detailed in the following theorem. This theorem relies on bounding the uncertainty for each arm by a function of the number of times that arm is pulled. Roughly speaking, if this bounding function is monotonically decreasing and if the bounds U_k and L_k hold with high probability we can then apply Theorem 2[3] to bound the simple regret S_T of BayesGap.[4]

Theorem 1

Consider a K-armed Gaussian bandit problem, horizon T, and upper and lower bounds defined as above. For $\epsilon > 0$ and β defined as in Equation (4.7), the algorithm attains simple regret satisfying $\mathbb{P}(S_T \leq \epsilon) \geq 1 - KTe^{-\beta^2/2}$.

Proof. Let $N_k(t)$ be the number of times arm $k \in \mathcal{A}$ has been pulled by

[3]The additional Theorem is in Appendix A and is a modification of that in (Gabillon et al., 2012).

[4]Recall the simple regret $S_T := \max_k z_k - z_{\hat{k}_T}$, where $\hat{k}_T$ is the recommended arm after T rounds.

round t. Using the definition of the posterior variance for arm k, we can write the confidence diameter as

$$s_k(t) = 2\beta\sqrt{\mathbf{x}_k^\mathsf{T}\Sigma_t\mathbf{x}_k} \tag{4.8}$$

$$= 2\beta\sqrt{\eta^2\mathbf{x}_k^\mathsf{T}\big(\textstyle\sum_i N_i(t-1)\,\mathbf{x}_i\mathbf{x}_i^\mathsf{T} + \frac{\eta^2}{\sigma^2}\mathbf{I}\big)^{-1}\mathbf{x}_k} \tag{4.9}$$

$$\leq 2\beta\sqrt{\eta^2\mathbf{x}_k^\mathsf{T}\big(N_k(t-1)\,\mathbf{x}_k\mathbf{x}_k^\mathsf{T} + \frac{\eta^2}{\sigma^2}\mathbf{I}\big)^{-1}\mathbf{x}_k}. \tag{4.10}$$

In the second equality we decomposed the Gram matrix $\mathbf{X}^\mathsf{T}\mathbf{X}$ in terms of a sum of outer products over the fixed vectors $\mathbf{x}_i$—letting $\mathbf{X}$ be the matrix whose rows are the vectors $\mathbf{x}_i$. In the final inequality we noted that by removing samples we can only increase the variance term, and so here we set $N_i(t-1) = 0$ for $i \neq k$. We will let the result of this final inequality define an arm-dependent bounding function $g_k : \mathbb{N} \mapsto (0, \infty)$. Letting $A = \frac{1}{N}\frac{\eta^2}{\sigma^2}$ we can simplify this quantity using the Sherman–Morrison formula as

$$g_k(N) = 2\beta\sqrt{(\eta^2/N)\mathbf{x}_k^T\big(\mathbf{x}_k\mathbf{x}_k^\mathsf{T} + A\mathbf{I}\big)^{-1}\mathbf{x}_k} \tag{4.11}$$

$$= 2\beta\sqrt{\frac{\eta^2}{N}\frac{\|\mathbf{x}_k\|^2}{A}\Big(1 - \frac{\|\mathbf{x}_k\|^2/A}{1 + \|\mathbf{x}_k\|^2/A}\Big)} \tag{4.12}$$

$$= 2\beta\sqrt{\frac{\eta^2\|\mathbf{x}_k\|^2}{\frac{\eta^2}{\sigma^2} + N\|\mathbf{x}_k\|^2}}, \tag{4.13}$$

which is monotonically decreasing in N. The inverse of this function can be solved for as

$$g_k^{-1}(s) = \frac{4\beta^2\eta^2}{s^2} - \frac{\eta^2}{\sigma^2}\frac{1}{\|\mathbf{x}_k\|^2}. \tag{4.14}$$

By setting $\sum_k g_k^{-1}(H_{k\epsilon}) = T - K$ and solving for β we then obtain the definition of this term given in the statement of the proposition. Finally, by reference to Lemma B4 in the appendix, we can see that for each k and t, the upper and lower bounds must hold with probability $1 - e^{-\beta/2}$. These last two statements satisfy the assumptions of Theorem 2 in the appendix, thus concluding the proof. □

Here we should note that while we are using Bayesian methodology to

drive the exploration of the bandit, we are analyzing this using frequentist regret bounds. This is a common practice when analyzing the regret of Bayesian bandit methods (Srinivas et al., 2010; Kaufmann et al., 2012a). We should also point out that implicitly Theorem 2 assumes that each arm is pulled at least once regardless of its bound. However, in our setting we can avoid this in practice due to the correlation between arms.

On the exploration parameter β. The proof and derivation of β given above require knowledge of the hardness quantity H_ϵ, which is unknown in most practical applications. Instead of requiring H_ϵ from the user, our approach is to adaptively bound it. Intuitively, the quantity β controls how aggressively BayesGap explores uncertain arms; indeed, β is directly proportional to the width of the uncertainty $s_k(t)$ and inversely proportional to H_ϵ. As a result, in order to initially encourage more exploration we will lower bound the hardness quantity. In particular, we can do this by upper bounding each Δ_k, using conservative posterior-dependent upper and lower bounds on z_k. In this work we use three posterior standard deviations away from the posterior mean: $\mu_{kt} \pm 3\sigma_{kt}$—we emphasize that these bounds are not the same as $L_k(t)$ and $U_k(t)$. Then a high probability upper bound on Δ_k is simply given by

$$\hat{\Delta}_{kt} = \max_{j \neq k} (\mu_{jt} + 3\sigma_{jt}) - (\mu_{kt} - 3\sigma_{kt}). \tag{4.15}$$

From this point we can recompute H_ϵ and in turn recompute β (Step 7 in Algorithm 2). For all experiments we will use this adaptive method.

Comparison with UGap. The method in this section provides a Bayesian version of the UGap algorithm which modifies the bounds used in this earlier algorithm's arm selection step. By modifying step 6 of the BayesGap pseudo-code to use either Hoeffding or Bernstein bounds we recover the UGap algorithm. Note, however, that in doing so UGap assumes independent arms with bounded rewards.

We can now compare UGap's probability of error $O(KT \exp(-\frac{T-K}{H_\epsilon}))$

with that of BayesGap $O(KT\exp(-\frac{T-K+\kappa\eta^2/\sigma^2}{H_\epsilon\eta^2}))$. With minor differences, these bounds are of the same order. First, the additional η^2 term is primarily due to the distinction between bounded and Gaussian-distributed rewards. Similarly, the σ^2 term corresponds to the concentration of the prior, and we can see that the more concentrated the prior is (smaller σ) the faster this rate is. Note, however, that the proof of BayesGap's simple regret relies on the true rewards for each arm being within the support of the prior, so one cannot increase the algorithm's performance by arbitrarily concentrating the prior. Finally, the κ term is related to the linear relationship between different arms. Additional theoretical results on improving these bounds remains for future work.

4.3 Experiments

In the following subsections, we benchmark our proposed algorithm against a wide variety of policies on two real-data applications. In Section 4.3.1, we revisit the traffic sensor network problem of Srinivas et al. (2010). In Section 4.3.2, we consider the problem of automatic model selection and algorithm configuration.

4.3.1 Application to a traffic sensor network

In this experiment, we are given data taken from traffic speed sensors deployed along highway I-880 South in California. Traffic speeds were collected at $K = 357$ sensor locations for all working days between 6AM and 11AM for an entire month. Our task is to identify the single location of least congestion, *i.e.* with the highest expected speed. This data was also used in the work of Srinivas et al. (2010) on GP-UCB.

Naturally, the readings from different sensors are correlated, however, this correlation is not necessarily only due to geographical location. Therefore specifying a similarity kernel over the space of traffic sensor locations alone would be overly restrictive. Following the approach of Srinivas et al. (2010), we construct the design matrix treating two-thirds of the available

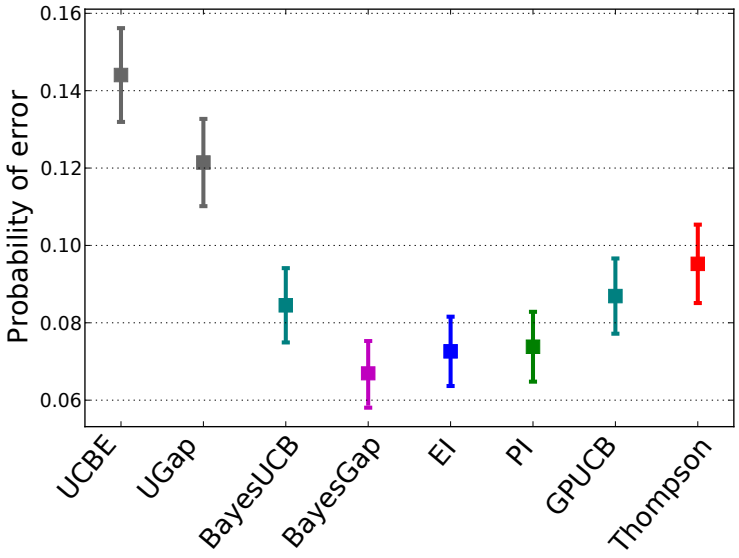


Figure 4.2: Probability of error on the optimization domain of traffic speed sensors (averaged over 840 runs). For this real data set, BayesGap provides considerable improvements over the Bayesian cumulative regret alternatives and the frequentist simple regret counterparts.

data as historical and use the remaining third to evaluate the policies. In more detail, the GP kernel matrix $G \in \mathbb{R}^{K \times K}$ is set to be empirical covariance matrix of measurements for each of the K sensor locations, the corresponding design matrix is $\mathbf{X} = VD^{\frac{1}{2}}$, where $G = VDV^T$.

Following (Srinivas et al., 2010), we estimate the noise level η of the observation model using this data. We consider the average empirical variance of each individual sensor (i.e. the signal variance corresponding to the diagonal of G) and set the noise variance η^2 to 5% of this value; this corresponds to $\eta^2 = 4.78$. We choose a broad prior with regularization coefficient $\sigma = 20$.

In order to evaluate different bandit and Bayesian optimization algorithms, we use each of the remaining 840 sensor signals (the aforementioned third of the data) as the true mean vector $\mathbf{z}$ for independent runs of the experiment. Using historical data in this way enables us to evaluate the ground truth for each run and estimate the probability that the policies return the best arm.

Other policies in this empirical comparison:

(1) UCBE (Audibert et al., 2010): variant of classical UCB of Auer (2003), substituting the $\log(t)$ exploration term in UCB with a constant of order $\log(T)$ for known horizon T;

(2) UGap (Gabillon et al., 2012): frequentist algorithm that inspired our Bayesian variant;

(3) BayesUCB and GP-UCB: Bayesian extensions of UCB introduced by Kaufmann et al. (2012a) and Srinivas et al. (2010), respectively;

(4) PI, EI, and Thompson sampling (see Chapter 3).

Note that techniques (1) and (2) attack the problem of best arm identification and use bounds which encourage more aggressive exploration. However, they do not take correlation into account. On the other hand, techniques such as (3) are designed for cumulative regret, but model the correlation among the arms. It might seem at first that we are comparing apples and oranges. However, the purpose of comparing these methods, even if their objectives are different, is to understand empirically what aspects of these algorithms matter the most in practical applications.

The results, shown in Figure 4.2, are the probabilities of error for each strategy, using a time horizon of $T = 400$. (Here we used $\epsilon = 0$, but varying this quantity had little effect on the performance of each algorithm.) By looking at the results, we quickly learn that techniques that model correlation perform better than the techniques designed for best arm identification, even when they are being evaluated in a best arm identification task. The important conclusion is that one must always invest effort in modelling the correlation among the arms.

The results also show that BayesGap does better than alternatives in this domain. This is not surprising because BayesGap is the only competitor that addresses budgets, best arm identification and correlation simultaneously.

4.3.2 Automatic machine learning

There exist many machine learning toolboxes, such as `Weka` and `scikit-learn`. However, for a great many data practitioners interested in finding the best technique for a predictive task, it is often hard to understand what each technique in the toolbox does. Moreover, each technique can have many free hyper-parameters that are not intuitive to most users.

Bayesian optimization techniques have already been proposed to automate machine learning approaches such as MCMC inference (Mahendran et al., 2012; Hamze et al., 2013; Wang et al., 2013a), deep learning (Bergstra et al., 2011), preference learning (Brochu et al., 2007, 2010), reinforcement learning and control (Martinez-Cantin et al., 2007; Lizotte, 2008), and more (Snoek et al., 2012). In fact, methods to automate entire toolboxes (`Weka`) have appeared very recently (Hutter et al., 2012), and go back to old racing strategies for model selection (Maron and Moore, 1993; Shen et al., 2011).

In this experiment, we will demonstrate BayesGap by automating regression with `scikit-learn`. Our focus will be on minimizing the cost of cross-validation in the domain of big data. In this setting, training and testing each model can take a prohibitive long time. If we are working under a finite budget, say if we only have three days before a conference deadline or the deployment of a product, we cannot afford to try all models in all cross-validation tests. However, it is possible to use techniques such as BayesGap and Thompson sampling to find the best model with high probability. In our setting, the action of "pulling an arm" will involve selecting a model, splitting the dataset randomly into training and test sets, training the model, and recording the test-set performance.

In this bandit domain, our arms will consist of five `scikit-learn` techniques and associated parameters selected on a discrete grid. We consider the following models and parameter settings for regression:

Within a class of regressors, we model correlation using a squared exponential kernel with unit length scale, i.e., $k(x, x') = \mathrm{e}^{-(x-x')^2}$. Using this kernel, we compute a kernel matrix G and construct the design matrix as before.

Model & hypers (160 total)	discretized settings
Lasso (8)	
`alpha`	$\{1, 5\} \cdot 10^{\{-4,-3,-2,-1\}}$
Random forests (64)	
`n_estimators`	$10^{\{0,1,2,3\}}$
`min_samples_split`	1, 3, 5, 7
`min_samples_leaf`	2, 6, 10, 14
linSVM (16) & rbfSVM (64)	
`C`	$10^{\{-3,-2,-1,0\}}$
`epsilon`	$10^{\{-4,-3,-2,-1\}}$
(rbf only) `gamma`	0.025, 0.05, 0.1, 0.2
K-NN (8)	
`n_neighbors`	1, 3, 5, 7, 9, 11, 13, 15

Table 4.1: Models and discretized hyperparameter settings used in the automatic machine learning model selection experiment.

When an arm is pulled we select training and test sets uniformly at random. Each set includes 10% of the data in the original dataset; the remaining 80% is ignored for this particular arm pull. We then train the selected model on the training set, and test on the test set. This specific form of cross-validation is similar to that of repeated learning-testing (Arlot et al., 2010; Burman, 1989).

We use the `wine` dataset from the UCI Machine Learning Repository, where the task is to predict the quality score (between 0 and 10) of a wine given 11 attributes of its chemistry. We repeat the experiment 100 times. We report, for each method, an estimate of the RMSE for the recommended models on each run. Unlike in the previous section, we do not have the ground truth generalization error, and in this scenario it is difficult to estimate the probability of error. Instead we report the RMSE, but remark that this is only a proxy for the error rate that we are interested in.

The performance of the final recommendations for each strategy and a fixed budget of $T = 10$ tests is shown in Figure 4.3. The results for other

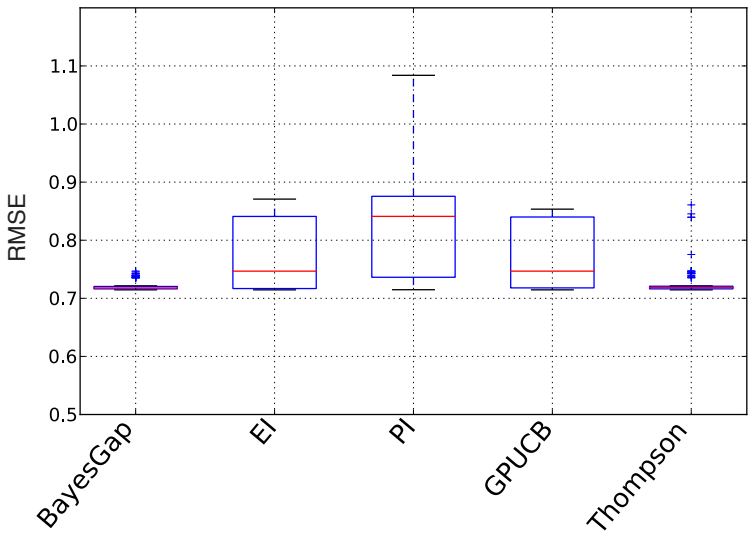


Figure 4.3: Boxplot of RMSE averaged over 100 runs with a fixed budget of $T = 10$. EI, PI, and GP-UCB get stuck in local minima.

budgets are almost identical. It must be emphasized that the number of allowed function evaluations (10 tests) is much smaller than the number of arms (160 models). Hence, frequentist approaches that require pulling all arms, e.g. UGap, are inapplicable in this domain.

The results indicate that Thompson and BayesGap are the best choices for this domain. Figure 4.4 shows the individual arms pulled and recommended by BayesGap (above) and EI (bottom), over each of the 100 runs, as well as an estimate of the ground truth RMSE for each individual model. EI and PI often get trapped in local minima. Due to the randomization inherent to Thompson sampling, it can explores more depending on the specified prior.

4.4 Conclusion

This work demonstrates the benefit of modelling correlations between arms in multi-armed bandits. Not only does modelling correlations help converge more quickly to better optima—as in the traffic sensor network results—

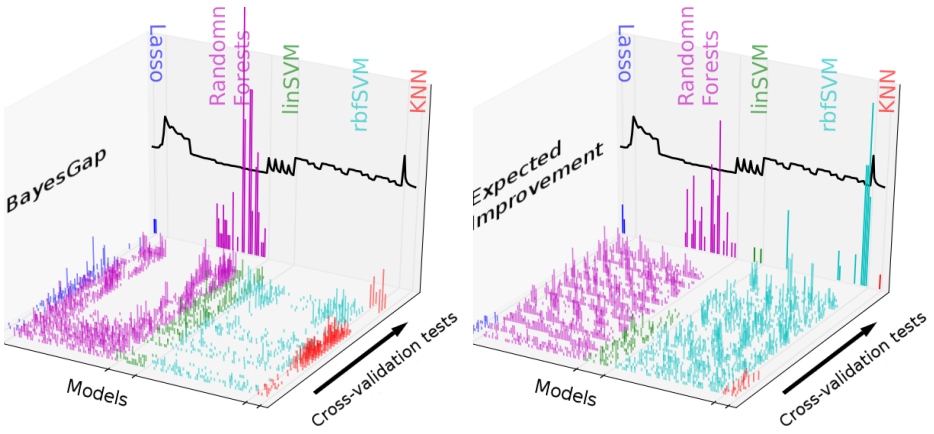


Figure 4.4: Allocations and recommendations of BayesGap (top) and EI (bottom) over 100 runs at a budget of $T = 40$ training and validation tests, and for 160 models (i.e., more arms than possible observations). Histograms along the floor of the plot show the arms pulled at each round while the histogram on the far wall shows the final arm recommendation over 100 different runs. The solid black line on the far wall shows the estimated "ground truth" RMSE for each model. Note that EI quite often gets stuck in a locally optimal rbfSVM.

but such modelling efforts accommodate budgets that are smaller than the number of available arms by generalizing observations to unobserved arms—as in the automatic model selection results.

There is room to improve the theoretical results proven in Appendix A. In particular, we emphasize two loose bounds: the union bound that contributes the factor of KT ignores both dependencies across arms and across time (or rounds); meanwhile, the inequality in Equation 4.10 is the result of a very conservative, worst case bound that effectively ignores interactions between arms.

While generalizations to continuous search spaces are not studied, it is an interesting direction for future work. Indeed, the appealing feature of these gap-based algorithms is that they always consider two different candidates and select the most uncertain one. Such an approach could allow Bayesian optimization algorithms to avoid getting stuck in local optima.

Chapter 5

Entropy search portfolio

Committing to a policy *a priori* is a challenging task, which can lead to dramatically different performance. Consider, for example, the experiments in Chapter 4 where Thompson sampling ranked very low in the traffic sensor network results, but very high in the automatic machine learning results. In this chapter, we address this practical challenge. Indeed, we investigate a hierarchical approach to decision making, whereby at every iteration t, we appeal to several policies to provide candidate query points and select a single $\mathbf{x}_t$ among them. Though this type of approach is referred to as a portfolio method (Hoffman et al., 2011), it can be interpreted as turning an optimization problem over the input space $\mathcal{X}$ into a multi-armed bandit problem over the set of arms (acquisition functions) $\mathcal{A}_t := \{\mathbf{x}_t^k := \arg\max_{\mathcal{X}} \alpha_t^k\}_{k=1}^K$.[1] In order to pick among the arms $\mathcal{A}_t$, we propose another utilitarian approach, which we distinguish from the utilitarian member policies by using the terms *meta-policy* or *meta-criterion*, denoted u_t.[2] Algorithm 3 outlines the slightly modified framework for Bayesian optimization with portfolios, and Figure 5.1 provides a visualization to help follow the discussion.

In Section 5.1.1, we outline our proposed meta-criterion, and the various approximations and practical techniques used to estimate it. An important

[1] See (Neufeld et al., 2014) for a similar bandit perspective on portfolios applied to adaptive Monte Carlo.

[2] Most of the time, since the time index is unambiguous, we will omit the subscript t to simply the notation.

Algorithm 3 Bayesian optimization with portfolios

Require: a stochastic black-box (f, ν)
initial data $\mathcal{D}_0 = \emptyset$
a portfolio $\mathcal{A} = \{\alpha^k\}_{k=1}^K$

1: **for** $t = 1, \ldots, T$ **do**
2: $\quad \mathcal{A}_t \leftarrow \{\arg\max_{\mathcal{X}} \alpha_t^k\}_{k=1}^K$ $\quad \triangleleft$ collect candidates
3: $\quad \mathbf{x}_t \leftarrow \arg\max_{\mathcal{A}_t} u(\,\cdot\,; \mathcal{D}_{t-1})$
4: $\quad y_t \sim \nu(f(\mathbf{x}_t))$
5: $\quad \mathcal{D}_t \leftarrow \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, y_t)\}$
6: **end for**
7: **return** $\hat{\mathbf{x}}_T \leftarrow \underset{\{\mathbf{x}_s\}_{s \le T}}{\arg\max}\, \mu_T$

step in the approximation relies on sampling from Equation (5.1), and in Section 5.1.2, we show how to efficiently approximately sample from this distribution, an approach that also acts as an extension of Thompson sampling to continuous domains with GP priors. Though this particular section was concurrent work by Hernández-Lobato et al. (2014), ours was the first empirical evaluation of this technique as an acquisition function. The entire algorithm, including computational complexity analysis, is summarized in Section 5.1.3 and the pseudocode is provided in Algorithm 4. Section 5.2 details several experiments and results in the domains of geostatistics and control.

5.1 Algorithm

Inspired by the works of Villemonteix et al. (2009) and Hennig and Schuler (2012), the approach we propose in this chapter is to use the reduction in entropy as a meta-criterion to select an $\mathbf{x}_t$ among the points in $\mathcal{A}_t$.

Recall that our prescribed surrogate probabilistic model induces a distribution over the location of the unknown global optimizer $\mathbf{x}_\star$. Note that this optimizer is a random variable, due to the epistemic randomness in f. Given data $\mathcal{D}$, let us define the following distribution over the measurable

input space $(\mathcal{X}, \mathcal{F}_X)$

$$\mathbb{P}_\star(\mathcal{E} \mid \mathcal{D}) := \mathbb{P}\left(\arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \in \mathcal{E} \;\middle|\; \mathcal{D}\right), \quad \forall \mathcal{E} \in \mathcal{F}_X, \tag{5.1}$$

denoting the posterior over minimizer locations, with density $p_\star(\,\cdot \mid \mathcal{D})$. Our proposed algorithm we call entropy search portfolio (ESP) then selects, among the candidate points $\mathcal{A} = \{\mathbf{x}^k := \arg\max_{\mathbf{x}\in\mathcal{X}} \alpha^k(\mathbf{x}; \mathcal{D})\}_{k=1}^K$, the one which maximizes the reduction in uncertainty. As a measure of uncertainty in the posterior distribution $\mathbb{P}_\star$, we will use the differential entropy $H[p_\star]$, and therefore propose the negative expected entropy as a meta-criterion:

$$\mathbf{x}_t := \arg\max_{\mathbf{x}^k\in\mathcal{A}} u(\mathbf{x}^k; \mathcal{D}), \text{ where} \tag{5.2}$$

$$u(\mathbf{x}^k; \mathcal{D}) := -\mathbb{E}H\left[p_\star\left(\,\cdot \;\middle|\; \mathcal{D} \cup \{(\mathbf{x}^k, Y_k)\}\right)\right], \tag{5.3}$$

where the expectation is taken over the, as yet, unobserved random variable $Y_k \sim \nu_{\mathbf{x}^k}$.

5.1.1 Estimating the expected entropy

Unfortunately, the meta-criterion expressed in Equation (5.3) is analytically intractable and requires some form of approximation in order to make the algorithm feasible.[3] We begin by estimating the expectation via Monte Carlo integration, taking N samples $y_k^{(n)} \sim p(y_k \mid \mathbf{x}^k, \mathcal{D})$ and approximating the meta-criterion as

$$u(\mathbf{x}^k; \mathcal{D}) \approx -\frac{1}{N}\sum_{n=1}^{N} H\left[p_\star(\,\cdot \mid \tilde{\mathcal{D}}_k^{(n)})\right], \tag{5.4}$$

where $\tilde{\mathcal{D}}_k^{(n)} := \mathcal{D} \cup \{(\mathbf{x}^k, y_k^{(n)})\}$. In practice one can use stratified or quasi-Monte Carlo to reduce the variance of this estimate.

We turn our attention to the entropy estimation. For continuous densi-

[3] Since this work, Hernández-Lobato et al. (2014) proposed a new and better approximation that can readily be incorporated in this work.

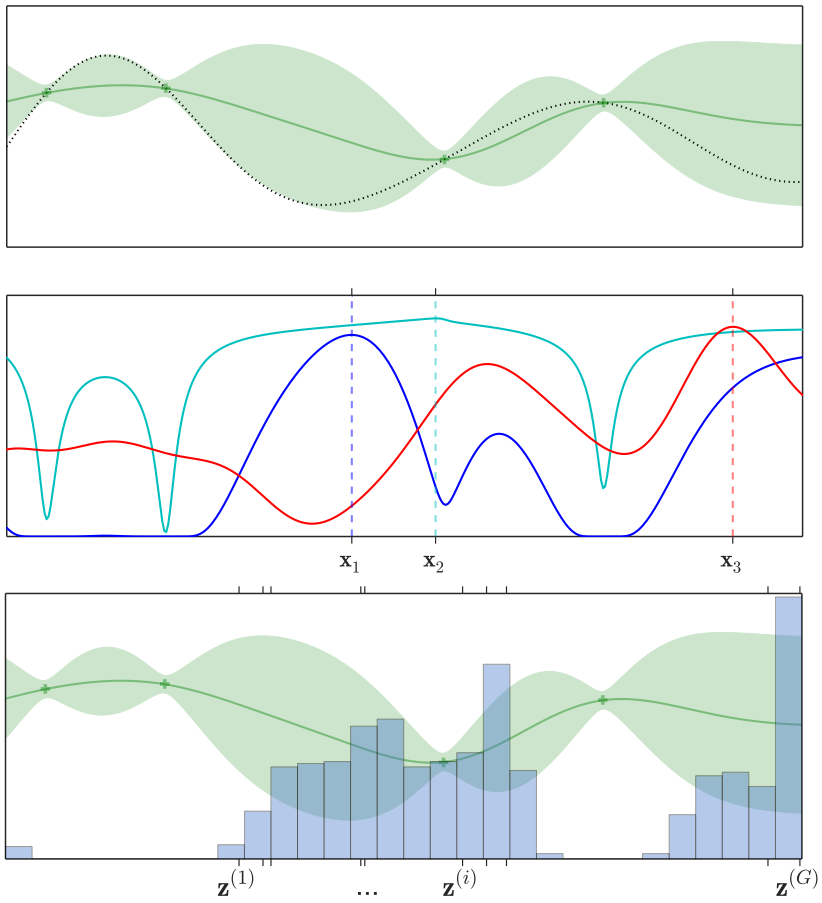


Figure 5.1: Visualization of Bayesian optimization with acquisition function portfolios. **Top:** visualization of our probabilistic model at a particular time. **Middle:** three acquisition functions, members of a portfolio, in this case EI (blue), PI (cyan), and Thompson (red). The purpose of Algorithm 4 is to select between the candidates $\{\mathbf{x}_k\}_{k=1}^3$. **Bottom:** histogram of 1000 approximate samples from $\mathbb{P}_\star$. Here we also show G draws from $\mathbb{P}_\star(\,\cdot \mid \mathcal{D})$ that we denote $\mathbf{z}^{(i)}$.

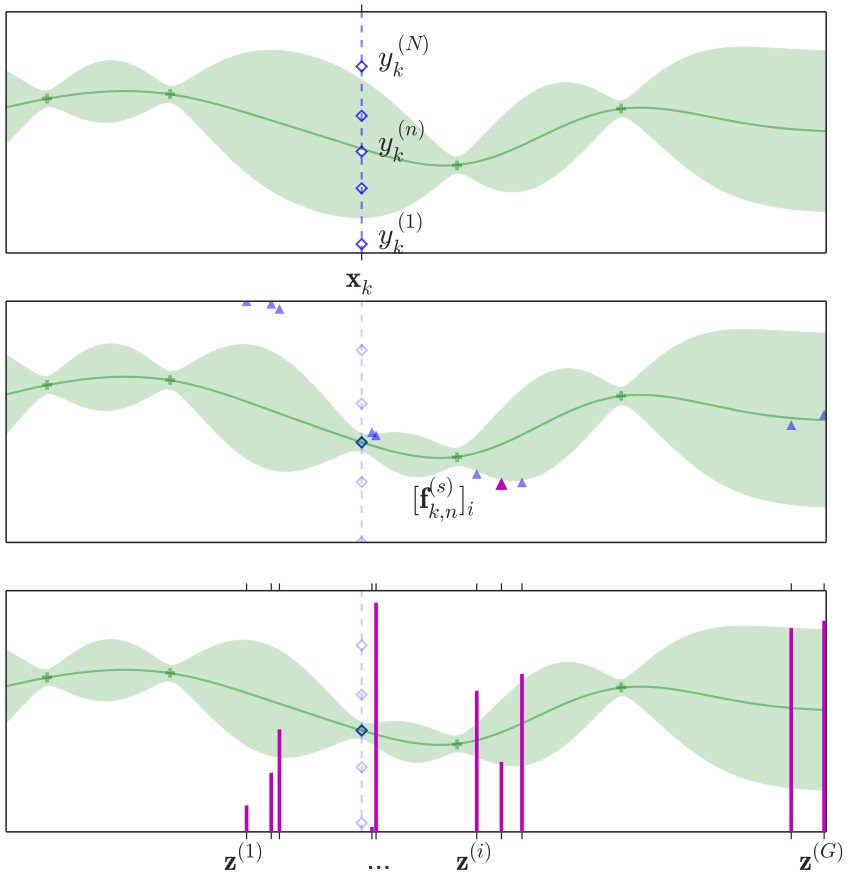


Figure 5.2: Visualization of the Entropy Search Portfolio. **Top:** for each candidate $\mathbf{x}_k$ (blue dashed line) we draw N hallucinations (blue diamonds) from the conditional. In practice this is done with quasi-Monte-Carlo to reduce variance. **Middle:** for each hallucination $y_k^{(n)}$ we augment the GP model (green line and shaded area) and sample it S times at the discrete points $\mathbf{z}^{(i)}$ to obtain the $\mathbf{f}_{kn}^{(s)}$ (blue triangles) for $s = 1, \ldots, S$. We find the minimizer of each vector $\mathbf{f}_{kn}^{(s)}$ (magenta triangle). **Bottom:** finally, we bin the S minimizers into a discrete empirical distribution $\hat{p}$ depicted here as a magenta histogram.

ties p the differential entropy can be written as

$$H[p(\mathbf{x})] := -\int_{\mathcal{X}} p(\mathbf{x}) \log p(\mathbf{x}) \, \mathrm{d}\mathbf{x}. \tag{5.5}$$

Monte Carlo integration cannot estimate this integral because the integrand $\log p_\star(\,\cdot \mid \mathcal{D})$ cannot be evaluated pointwise. Instead, we approximate this density with a discrete probability mass function $\hat{p}_\star$ and use its discrete entropy as a proxy for the differential entropy of interest in (5.5).

Let us therefore restrict our attention to a discretization of the domain $\{\mathbf{z}^{(i)}\}_{i=1}^{G} \subset \mathcal{X}$, and in an effort to maintain the flow of this section, we defer the technical details of how the space is discretized to the next section.

Given such a discretization, we can define the following probability mass function:

$$\hat{p}_\star(\mathbf{z}) := \mathbb{P}\left(Y_{\mathbf{z}} \leq Y_{\mathbf{z}^{(i)}},\ \forall \mathbf{z}^{(i)}\right), \quad \forall \mathbf{z} \in \{\mathbf{z}^{(i)}\}, \tag{5.6}$$

and similarly for the posterior distribution $\hat{p}_\star(\mathbf{z} \mid \mathcal{D})$.

Combining this discrete proxy for the entropy with the Monte Carlo integration above yields the following approximation to our meta-criterion

$$u(\mathbf{x}^k; \mathcal{D}) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{G} \hat{p}_\star(\mathbf{z}^{(i)} \mid \tilde{\mathcal{D}}_k^{(n)}) \log \hat{p}_\star(\mathbf{z}^{(i)} \mid \tilde{\mathcal{D}}_k^{(n)}). \tag{5.7}$$

Finally, we must estimate $\hat{p}_\star(\mathbf{z}^{(i)} \mid \tilde{\mathcal{D}}_k^{(n)})$. Let the multivariate random variable $\mathbf{f} := (f(\mathbf{z}^{(1)}), \ldots, f(\mathbf{z}^{(G)}))$ be the vector of latent objective function values evaluated at the representer points. Under the assumption that f is sampled from a GP, the posterior distribution of $\mathbf{f} \mid \tilde{\mathcal{D}}_k^{(n)}$ is normal; as a result, we can produce S exact samples $\mathbf{f}_{kn}^{(s)}$ from the posterior. The probabilities necessary to compute the entropy can then be approximated by the relative counts

$$\hat{p}_{kni} = \frac{1}{S} \sum_{s=1}^{S} \mathbb{I}\big[i = \operatorname*{arg\,min}_{j=1,\ldots,G} [\mathbf{f}_{kn}^{(s)}]_j\big]. \tag{5.8}$$

In other words, the number $\hat{p}_{kni}$ is the proportion of times the representer

Algorithm 4 Entropy search portfolio

Require: candidates $\{\mathbf{x}^k\}_{k=1}^K$, observations $\mathcal{D}$

$\mathbf{z}^{(i)} \sim \mathbb{P}_\star(\,\cdot \mid \mathcal{D}), \quad i = 1, \ldots, G$
for $k = 1, \ldots, K$ **do**
  for $n = 1, \ldots, N$ **do**
    $y_k^{(n)} \sim \nu(f(\mathbf{x}^k) \mid \mathcal{D})$ ◁ hallucinate next observation
    $\tilde{\mathcal{D}}_k^{(n)} \leftarrow \mathcal{D} \cup \{(\mathbf{x}^k, y_k^{(n)})\}$
    $\mathbf{f}_{kn}^{(s)} \sim p(\mathbf{f} \mid \tilde{\mathcal{D}}_k^{(n)})$ for $s = 1, \ldots, S$
    $\hat{p}_{kni} \leftarrow \frac{1}{S} \sum_s \mathbb{I}\big[i = \arg\min_j [\mathbf{f}_{kn}^{(s)}]_j\big]$
  end for
  $u_k \leftarrow \frac{1}{N} \sum_{n=1}^N \sum_{i=1}^G \hat{p}_{kni} \log \hat{p}_{kni}$
end for
$\hat{k} = \arg\max_{k=1,\ldots,K} u_k$
return $\mathbf{x}_{\hat{k}}$

point $\mathbf{z}^{(i)}$ was the minimizer among the S sampled functions with hallucinated data point $(\mathbf{x}^k, y_k^{(n)})$.

Finally, combining the estimates above yields our approximated meta-criterion

$$u(\mathbf{x}^k; \mathcal{D}) \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{G} \hat{p}_{kni} \log \hat{p}_{kni}. \tag{5.9}$$

We provide the pseudocode for this estimate of the meta-criterion in Algorithm 4 along with a corresponding visualization in Figure 5.2.

5.1.2 Discretizing $\mathcal{X}$: sampling posterior global minimizers

Discretizing the search space $\mathcal{X}$ for such entropy estimates is an established approximation technique (Villemonteix et al., 2009; Hennig and Schuler, 2012). Naturally, obtaining a good estimate depends on the particular discretization. In the work of Hennig and Schuler (2012), the discrete points $\mathbf{z}^{(i)}$ are called *representer points* and are sampled from either PI or EI, depending on the implementation; earlier, in the work of Villemonteix et al. (2009), the discretization was sampled from a latin hypercube (LHC) space filling design. Both of these works have argued the importance of both

randomization and sampling points $\{\mathbf{z}^{(i)}\}$ close to the target distribution $\mathbb{P}_\star(\,\cdot \mid \tilde{\mathcal{D}}_k)$. While the LHC approach provides a very simple discretization, it is independent of the current posterior, which can lead to very small effective discretization, *i.e.* eventually most representer points will have no effect on the estimate. On the other hand, running a slice-sampler on PI or EI is a practically challenging heuristic because these acquisition functions quickly become highly multi-modal with shrinking support.

In this work, we propose using approximate samples from the posterior distribution $\mathbb{P}_\star(\,\cdot \mid \mathcal{D})$, which we expect to be closer to the target distribution than EI or PI, by virtue of the fact that the posterior does not vary too dramatically from one observation to the next. This step corresponds to Line 1 of Algorithm 4; after this step the $\{\mathbf{z}^{(i)}\}$ are kept fixed for the remainder of the ESP subroutine, and only resampled at the next iteration of the optimization outer-loop: Algorithm 3.

We use the following two step process to sample from $\mathbb{P}_\star(\,\cdot \mid \mathcal{D})$

$$f \mid \mathcal{D} \sim \mathrm{GP}, \tag{5.10}$$

$$\mathbf{z}^{(i)} = \arg\min_{\mathcal{X}} f, \quad \forall i = 1, \ldots, G \tag{5.11}$$

where ties are broken arbitrarily. At first glance this would require constructing an infinite-dimensional object representing the function f. This could be done by sequentially constructing f while it is being optimized. However, evaluating such an f would ultimately have cost $\mathcal{O}(m^3)$ where m is the number of function evaluations necessary to find each optimum $\mathbf{z}^{(i)}$. Instead, following the work of Hernández-Lobato et al. (2014), we will sample an analytic approximation to f, which provides a closed form expression that can be differentiated and optimized efficiently. We defer the technical details of this technique to Appendix B.

Extension to Thompson sampling. Finally, this approach can also be used directly as an acquisition strategy; consider the following randomized strategy

$$\mathbf{x}_t \sim \mathbb{P}_\star(\,\cdot \mid \mathcal{D}).$$

For finite $\mathcal{X}$ this is simply the well-established Thompson sampling algorithm. However, this derivation extends Thompson sampling to continuous search spaces. This work is the first to evaluate this acquisition function empirically.

5.1.3 Algorithm summary

In Algorithm 3, we describe general pseudocode for performing Bayesian optimization using a portfolio of acquisition functions. Here line 3 denotes a general selection criterion such as the GP-Hedge approach of (Hoffman et al., 2011) or our own ESP, summarized in Algorithm 4. The computational cost of this approach will be dominated by the cubic cost of solving the linear system on line 6 of Algorithm 4. As a result the complexity will be on the order $\mathcal{O}(KNSt^3)$, however since K, N, and S are fixed *a priori*, this is only a constant-factor slower than standard Bayesian optimization algorithms and can easily be parallelized.

One consideration we have not mentioned is the selection of hyperparameters, which can have a considerable effect on the performance of any Bayesian optimization algorithm. For example, if we fix the length scale parameters Λ to be too small or large, then observing a point $(\mathbf{x}, y)$ will affect the posterior in a neighbourhood that is, in turn, too small or large around $\mathbf{x}$, respectively. This will adversely affect the optimization process, causing any Bayesian optimization algorithm to search too globally or too locally—in a sense, to explore or exploit too much.

Typical approaches to GP regression will often optimize the marginal likelihood or *evidence* as a means of setting these parameters. However, in the Bayesian optimization setting this approach is particularly ineffective due to the initial paucity of data. Even worse, such optimization can also lead to quite severe local maxima around the initial data points. In this work we instead give a fully Bayesian treatment of the hyperparameters. Let $\boldsymbol{\psi}$ denote a vector of hyperparameters which includes any kernel and likelihood parameters. Let $p(\boldsymbol{\psi} \mid \mathcal{D}) \propto p(\boldsymbol{\psi})\, p(\mathcal{D}|\boldsymbol{\psi})$ denote the posterior distribution over these parameters where $p(\boldsymbol{\psi})$ is a hyperprior and $p(\mathcal{D} \mid \boldsymbol{\psi})$

is the GP marginal likelihood. For a fully Bayesian treatment of $\boldsymbol{\psi}$ we must marginalize our acquisition strategy with respect to this posterior. The corresponding integral has no analytic expression and must be approximated using Monte Carlo.

Consider now drawing M samples $\{\boldsymbol{\psi}^{(i)}\}$ from the posterior $p(\boldsymbol{\psi}|\mathcal{D}_t)$. Often an acquisition strategy is specified with respect to some internal *acquisition function* which is optimized at every iteration in order to select $\mathbf{x}_t$. These functions can then be approximately marginalized with respect to the hyperparameter samples in order to instead optimize an integrated acquisition function. This approach was taken in (Snoek et al., 2012) for the EI and PI acquisition functions. Note that the randomized Thompson sampling strategy, introduced in Section 5.1.2, only requires a single hyperparameter sample as we can see this as a joint sample from the hyperparameters and the function minimizer.

Our approach has additional complexity in that our candidate selection criterion depends on the hyperparameter samples as well. This occurs explicitly in lines (4–6) of Algorithm 4 which sample from the GP posterior. This can be solved simply by adding an additional loop over the hyperparameter samples. Our particular use of representer points in line 1 also depends on these hyperparameters. We can solve this problem by equally allocating our representers between the M different hyperparameter samples.

5.2 Experiments

In this section we evaluate the proposed method on several problems: two synthetic test functions commonly used for benchmarking in global optimization, two real-world datasets from the geostatistics community, and two relatively high dimensional simulated control problems. We compare our method against two categories of baselines: first, Bayesian optimization based on single acquisition functions, and second, other portfolio methods based on the same ensemble of acquisition functions as ESP (but combining them in different ways).

In the first category, we included three well-known Bayesian optimization

acquisition functions, namely the EI, PI, and Thompson methods described earlier. For EI we used the implementation available in the `spearmint` package,[4] and the latter two were implemented in the `spearmint` framework. All three methods were included in the portfolios. Note that we do not compare against GPUCB (Srinivas et al., 2010) as the bounds do not apply directly when integrating over GP hyperparameters. This would require an additional derivation which we do not consider here.

In the second category, we compare ESP against the GPHedge portfolio method of Hoffman et al. (2011) and an approach which selects between different base strategies uniformly at random labelled RP (Random Portfolio).

In order to marginalize over the GP hyperparameters (*i.e.* kernel lengthscale and amplitude, and prior constant mean) we used slice sampling as implemented in the spearmint package. By default, the spearmint software outputs 10 samples at each iteration. Both improvement-based methods, EI and PI, can simply be evaluated with each sample and averaged, while in keeping with the Thompson strategy, our Thompson implementation only uses a single sample, namely the last one. As reported above, our proposed ESP method splits its 500 representer points equally among the hyperparameter samples. In other words, for each of the 10 hyperparameters we draw 50 points. In addition, in order to estimate $u(\mathbf{x}^k; \mathcal{D})$, for each of 5 simulated $y_k^{(n)}$ and each GP hyperparameter, ESP draws 1000 samples from $\mathbb{P}_\star(\,\cdot \mid \tilde{\mathcal{D}}_k^{(n)})$, and then computes and averages the entropy estimates.

In the following experiments we evaluate the performance of each algorithm as a function of the number of function evaluations. When the true optimum is known, as in the case of the Branin and Hartmann 3 experiments, we measure performance by the absolute error. For all other minimization experiments, performance is given by the minimum function value obtained up to that iteration. Each experiment was repeated 20 times with different random seeds. For each method, we plot the median performance metric over these repetitions as well as shading the band between the lower and upper quartiles.

[4] https://github.com/JasperSnoek/spearmint

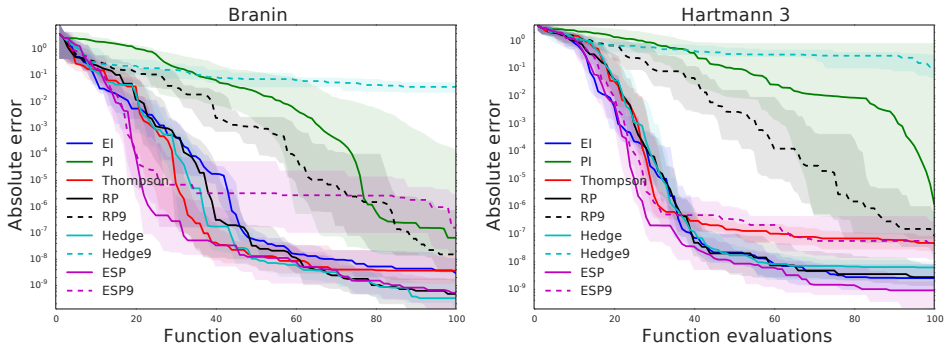


Figure 5.3: Absolute error of the best observation for the Branin and Hartmann 3 synthetic functions. The 9 additional random experts in RP9, GPHedge9, and ESP9 affect the RP and GPHedge methods much more dramatically than ESP.

5.2.1 Global optimization test functions

We begin with two synthetic functions commonly used for benchmarking global optimization methods: Branin and Hartmann 3 (Lizotte, 2008). They are two- and three-dimensional, respectively, and are both continuous, bounded, and multimodal. The experiments were run up to a final horizon of $T = 100$. Figure 5.3 reports the observed performance measured in absolute error on a logarithmic scale.

It is interesting to note that despite the poor performance of PI in this task, all portfolios still outperform their base strategies, with the exception of Thompson on Branin. As expected, ESP makes the best use of its base experts and outperforms all other methods in both examples.[5]

It is interesting to note that each of the two domains have their own champion, Thompson being a clear winner on Branin, and EI being the more attractive option on Hartmann 3. This observation motivates the use of portfolios of acquisition functions.

[5]Note that PI performs poorly in this multimodal example because of PI's greediness. This issue could in principle be addressed using a *minimum improvement* parameter (Jones, 2001; Brochu et al., 2009). However we set this parameter to its default value of 0 in our experiments since our focus in this work is in selecting between acquisition functions rather than tuning them. However, one could imagine in future work using parameterized families of strategies as a way of enlarging portfolios.

In these synthetic experiments we also demonstrate the resilience of ESP with respect to to the inclusion of poor base strategies. We do so by adding 9 random experts to each portfolio (we denote these ESP9, RP9, etc.). These so-called random experts select a point uniformly at random in the bounding box $\mathcal{X}$. We expect this sort of random search to be comparable to the other base methods in the initial stage of the optimization and eventually provide too much exploration and not enough exploitation. Note however that a few random experts in a portfolio could actually be beneficial in providing a constant source of purely exploratory candidates; precisely how many is an interesting question we do not discuss in the present work. Nevertheless, for the dimensionality and difficulty of these examples, we propose 9 additional random experts as being too many and indeed we observe empirically that they substantially deteriorate performance for all portfolios.

In particular, we see that, especially on Hartmann 3, ESP is virtually unaffected until it reaches 5 digits of accuracy. Meanwhile, the progress made by RP is hindered by the random experts which it selects as frequently as the legitimate acquisition functions. Significantly worse is the performance of GPHedge which, due to the initial success of the random experts, favours these until the horizon is reached. Note in contrast that ESP does not rely on any expert's past performance, which makes it robust to lucky guesses and time-varying expert performances. GPHedge could possibly be improved by tuning the reward function it uses internally but this is a difficult and problem dependent task.

5.2.2 Geostatistics datasets

The next set of experiments were carried out on two datasets from the geostatistics community, referred to here as Brenda and Agromet (Clark and Harper, 2008).[6] Since these datasets consist in finite sets of points, we transformed each of them into a function that we can query at arbitrary points via nearest neighbour interpolation. This produces a jagged piecewise constant function, which is outside the smoothness class of our surrogate

[6]Both datasets can be found at `kriging.com/datasets`.

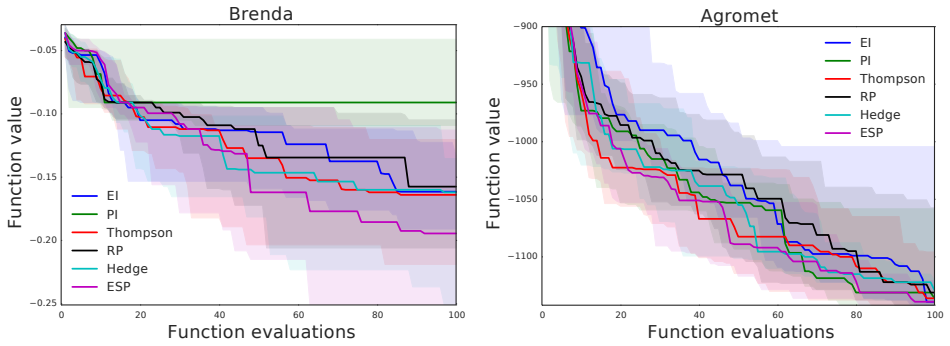


Figure 5.4: Best observed evaluation on mining datasets Brenda and Agromet. ESP outperforms the other portfolio methods while RP is dominated by all others.

models and hence a relatively difficult problem.

Brenda is a dataset of 1,856 observations in the depths of a copper mine in British Columbia, Canada, that was closed in 1990. The search domain is therefore three-dimensional and the objective function represents the quantity of underground copper. Agromet is a dataset of 18,188 observations taken in Natal Highlands, South Africa, where gold grade of the soil is the objective. The search domain here is two-dimensional.

We note that PI, which has so far been an under-achiever, is among the better strategies on Agromet, while EI is the last. This is further motivation for the use of portfolios. On both of these examples, RP performs poorly whereas GPHedge fares somewhat better. We can see that ESP performs particularly well on Brenda. On Agromet, ESP outperforms the other portfolio methods and is competitive with the best acquisition function—Thompson in the initial exploration phase, followed by PI after around 60 evaluations.

5.2.3 Control tasks

Our final experiments compare the portfolio methods on two control tasks. Walker is an eight-dimensional control problem where the inputs are fed into a simulator which returns the walking speed of a bipedal robot (Westervelt

and Grizzle, 2007). The code for the simulator is available online. This example was also used in a Bayesian optimization context by Hernández-Lobato et al. (2014). Repeller is a nine-dimensional control problem considered in previous work on Bayesian optimization portfolios (Hoffman et al., 2011, 2009). This problem simulates a particle in a two-dimensional world being dropped from a fixed starting region while accelerated downward by gravity. The task is to direct the falling particle through circular regions of low loss by placing 3 repellers with 3 degrees of freedom each, namely their position in the plane and repulsion strength. Figure 5.5 reports our results on both control tasks.

On these tasks, GPHedge performed poorly. On the other hand, RP performs well initially on both problems but its rate of improvement then slows down, particularly in Walker where it falls about 10 evaluations behind. Meanwhile, ESP is the clear top performer among the portfolio methods on Walker and catches up to RP on Repeller. We have also added the RP9, GPHedge9, and ESP9 methods in this experiment to study their sensitivity to random guesses. Indeed, both RP9 and GPHedge9 exhibit noticeably poorer performance than RP and GPHedge, while in contrast ESP9 is not significantly affected.

The good initial performance of the random portfolios and GPHedge9 could possibly be due to the high dimensionality of the search space where model-based approaches can suffer from cold start. In fact, a widely used technique suggested by Jones (2001) involves using a certain number of initial samples selected from a Latin hypercube as an initialization in order to alleviate this problem.

5.3 Conclusion

In this work, we revisited the use of portfolios for Bayesian optimization. We introduced a novel information-theoretic meta-criterion we call ESP, which can achieve performance matching or exceeding that of its best component expert. This is particularly important since we show in our experiments that the post-hoc best acquisition function varies between problem instances. We

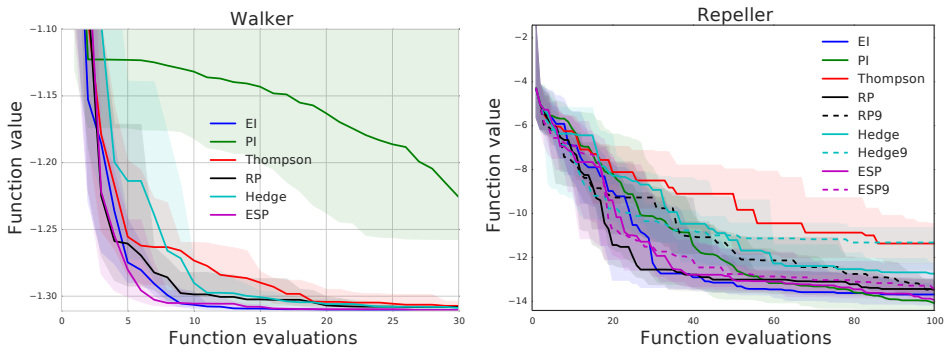


Figure 5.5: Best observed evaluation on the Walker and Repeller control tasks. Here we see ESP outperforming the other methods on Walker. On Repeller, ESP is once again competitive with the best methods and exhibits much more robustness to poor experts as evidenced by ESP9 following the performance of ESP so closely.

have also found that ESP ranks consistently high across functions of different dimensionality even though the members of its portfolio do not exhibit this behaviour. Further experiments also demonstrated that ESP is less sensitive to poorly performing experts than the other portfolio mechanisms.

We also showed that, surprisingly, a random portfolio can exhibit reasonably good performance, albeit not as good as ESP. Such a strategy is also more susceptible to the inclusion of poorly performing experts than ESP, however not as susceptible as GPHedge due to the latter algorithm's reliance on past performance as a predictor of future performance. On the other hand, ESP simply approaches vanilla entropy search as random experts are added to the portfolio.

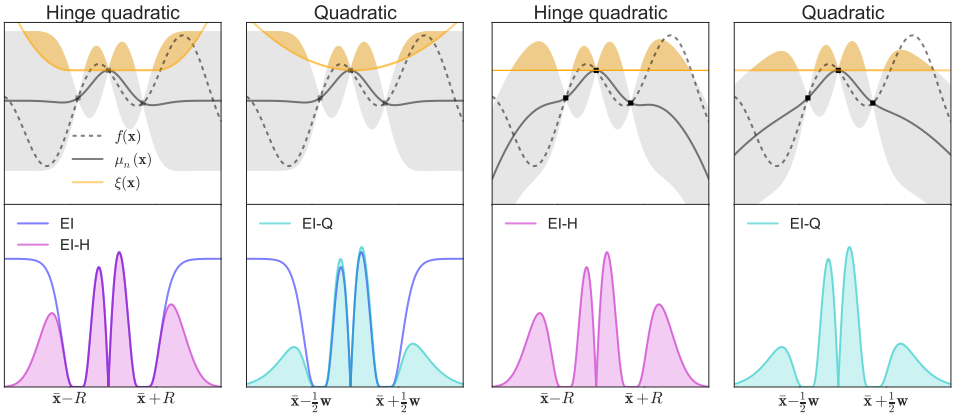
Finally, we propose a mechanism for sampling representer points which is more principled and reliable than previous approaches, which relied on slice sampling from surrogate measures. We additionally showed that this sampling mechanism can itself be used as an effective acquisition strategy, thereby extending the popular Thompson sampling approach to continuous spaces with kernel methods.

Chapter 6

Unconstrained Bayesian optimization via regularization

The established Bayesian optimization practice requires a user-defined bounding box which is assumed to contain the optimizer. However, when little is known about the probed objective function, it can be difficult to prescribe such bounds. In this work we modify the standard Bayesian optimization framework in a principled way to allow automatic, data-driven resizing of the search space.

This work is born out of the observation that the only component of standard Bayesian optimization that requires a bounding box on the search space is the maximization of the acquisition function. Therefore we propose a way to regularize the acquisition function to be a priori small in regions far to the observed data. In a first step, we achieve this by prescribing a non-constant *minimum improvement* function in expected improvement (EI; Jones et al., 1998) or probability of improvement (PI; Kushner, 1964). In a second step, we provide an equivalent view that involves a structured prior mean on the Gaussian Process. With this view, we are able to experiment with methods that are not improvement based, such as entropy search (Villemonteix et al., 2009; Hennig and Schuler, 2012; Hernández-Lobato et al., 2014) and



(a) Minimum improvement view (b) Prior mean view

Figure 6.1: Visualization of the two alternate views of regularization in Bayesian optimization. The objective function and posterior surrogate mean are represented as black dashed and solid lines, respectively, with grey shaded areas indicating $\pm 2\sigma_n$. Integrating the surrogate model above the target (orange shaded area) results in the regularized EI acquisition function (magenta and cyan). Using a non-stationary target with a constant prior mean (left) or a fixed target with a non-stationary prior mean (right) lead to indistinguishable acquisition functions, which decay at infinity.

Thompson sampling (Hernández-Lobato et al., 2014).

In this section, we introduce two methods that will result in robustness to the choice of initial bounding box. The first is our proposed approach, which can be interpreted as a regularization of current methods, while the second is a simple heuristic we call volume doubling.

6.1 Regularizing improvement policies

We motivate the regularized approach by considering improvement policies, in particular, EI. However, in the next section we show that this proposed approach can be extended more generally to all GP-derived acquisition functions, and in fact it is not difficult to apply the idea to other surrogate models, which we leave for future work.

Improvement policies are a popular class of acquisition functions that

rely on the improvement function

$$I(\mathbf{x}) = (f(\mathbf{x}) - \tau)\mathbb{I}[f(\mathbf{x}) \geq \tau] \tag{6.1}$$

where τ is some target value to improve upon. Expected improvement then compute $\mathbb{E}[I(\mathbf{x})]$ under the posterior GP.

When the optimal objective value $f_\star$ is known and we set $\tau = f_\star$, these algorithms are referred to as *goal seeking* (Jones, 2001). When the optimum is not known, it is common to use a proxy for $f_\star$, such as the value of the best observation so far, y^+; or in the noisy setting, one can use either the maximum of the posterior mean or the value of the mean prediction μ_n at the *incumbent* $\mathbf{x}^+ \in \arg\max_{\mathbf{x} \in \mathbf{x}_{1:n}} \mu_n(\mathbf{x})$.

In some cases, the above choice of target $\tau = y^+$ can lead to a lack of exploration, therefore it is common to choose a *minimum improvement* parameter $\xi > 0$ such that $\tau = (1 + \xi)y^+$ (for convenience here we assume y^+ is positive and in practice one can subtract the overall mean to make it so). Intuitively, the parameter ξ allows us to require a minimum fractional improvement over the current best observation y^+. Previously, the parameter ξ had always been chosen to be constant,[1] if not zero. In this work we propose to use a function $\xi : \mathbb{R}^d \mapsto \mathbb{R}^+$ which maps points in the space to a value of fractional minimum improvement. Following the same intuition, the function ξ lets us require larger improvements from points that are *farther* and hence acts as a regularizer that penalizes distant points. The improvement function hence becomes:

$$I(\mathbf{x}) = (f(\mathbf{x}) - \tau(\mathbf{x}))\mathbb{I}[f(\mathbf{x}) \geq \tau(\mathbf{x})], \tag{6.2}$$

where the target is now a function of $\mathbf{x}$:

$$\tau(\mathbf{x}) = (1 + \xi(\mathbf{x}))y^+; \tag{6.3}$$

the choice of $\xi(\mathbf{x})$ is discussed in the Section 6.3.

[1] Here we mean constant with respect to $\mathbf{x}$, there has been previous work on adaptively scheduling this parameters.

6.2 Extension to general policies

In the formulation of the previous section, our method seems restricted to improvement policies. However, many recent acquisition functions of interest are not improvement-based, such as GP-UCB, entropy search, and Thompson sampling. In this section, we describe a closely related formulation that generalizes to all acquisition functions that are derived from a GP surrogate model.

Consider expanding our choice of non-stationary target τ in Equation (6.3)

$$\begin{aligned} I(\mathbf{x}) &= (f(\mathbf{x}) - y^+(1+\xi(\mathbf{x})))\mathbb{I}[f(\mathbf{x}) \geq y^+(1+\xi(\mathbf{x}))] \\ &= (f(\mathbf{x}) - y^+\xi(\mathbf{x}) - y^+)\mathbb{I}[f(\mathbf{x}) - y^+\xi(\mathbf{x}) \geq y^+] \\ &= (\tilde{f}(\mathbf{x}) - y^+)\mathbb{I}[\tilde{f}(\mathbf{x}) \geq y^+] \end{aligned} \tag{6.4}$$

where $\tilde{f}$ is the posterior mean of a GP from (2.27) with prior mean $\tilde{\mu}_0(\mathbf{x}) = \mu_0(\mathbf{x}) - y^+\xi(\mathbf{x})$. Notice the similarity between (6.1) and (6.4). Indeed, in its current form we see that the regularization can be achieved simply by using a different prior mean $\tilde{\mu}_0$ and a constant target y^+. This duality can be visualized when comparing the left and right panel of Figure 6.1.

Strictly speaking, Equations (6.1) and (6.4) are not exactly equivalent. Indeed, using a surrogate GP with prior mean $\tilde{\mu}_0$, the posterior mean yields an additional term

$$-\mathbf{k}(\mathbf{x})^\mathsf{T}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\xi(\mathbf{X}), \tag{6.5}$$

where $[\xi(\mathbf{X})]_i = \xi(\mathbf{x}_i)$. This negative term will only accentuate the vanishing of expected improvement for test points $\mathbf{x}$ that are far from the regularizer centre. Indeed, in Figure 6.1 the two views produce indistinguishable regularized acquisition functions (in this case EI). However, we favour this new formulation because we can apply the same regularization to any policy which uses a Gaussian process, namely Thompson sampling and entropy search.

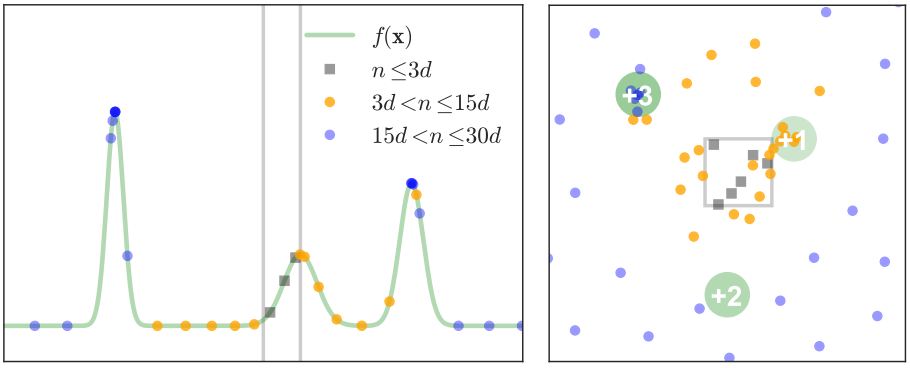


Figure 6.2: Visualization of selections $\mathbf{x}_n$ made by the EI-H algorithm on two toy problems: three Gaussian modes in one (left) and two (right) dimensions. Grey lines delimit the initial bounding box; grey square markers indicate the initial Latin hypercube points, while the orange and blue points distinguish between the first and second half of the evaluation budget of $30d$, respectively. In the two-dimensional example, the height of the Gaussians are indicated by $+1$, $+2$, and $+3$.

6.3 Choice of regularization

By inspecting Equation (3.21), we see that any *coercive*[2] prior mean function would lead to an asymptotically vanishing EI acquisition function as $||\mathbf{x}|| \to \infty$. More precisely, this is due to both Φ and ϕ vanishing as their arguments approach $-\infty$. In this work, we consider two coercive regularizing prior mean functions, namely a quadratic (Q) and an isotropic hinge-quadratic (H), defined as follows (excluding the constant bias b)

$$\xi_Q(\mathbf{x}) = (\mathbf{x} - \bar{\mathbf{x}})^\mathsf{T} \mathrm{diag}(\mathbf{w}^2)^{-1} (\mathbf{x} - \bar{\mathbf{x}}), \tag{6.6}$$

$$\xi_H(\mathbf{x}) = \mathbb{I}[\|\mathbf{x} - \bar{\mathbf{x}}\|_2 > R] \left(\frac{\|\mathbf{x} - \bar{\mathbf{x}}\|_2 - R}{\beta R} \right)^2. \tag{6.7}$$

Both of these regularizers are parameterized by d location parameters $\bar{\mathbf{x}}$, and while ξ_Q has an additional d width parameters $\mathbf{w}$, the isotropic regularizer ξ_H has a single radius R and a single β parameter, which controls the curvature

[2] By a coercive regularizer ξ we mean one that satisfies: $\lim_{||\mathbf{x}|| \to \infty} \xi(\mathbf{x}) \to -\infty$.

of ξ_H outside the ball of radius R; in what follows we fix $\beta = 1$.

Remark 4 (On fixing the prior mean hyperparameters)
We are left with the choice of centre $\bar{\mathbf{x}}$ and radius parameters R (or widths $\mathbf{w}$). Unlike the bias term b, these parameters of the regularizer are not intended to allow the surrogate to better fit the observations $\mathcal{D}_n$. In fact, using the marginal likelihood to estimate or marginalize $\psi = \{\bar{\mathbf{x}}, R, \mathbf{w}\}$, could lead to fitting the regularizer to a local mode which could trap the algorithm in a suboptimal well. For this reason, we use an initial, temporary user-defined bounding box to set ψ at the beginning of the run; the value of ψ remains fixed in all subsequent iterations.

Note that, while the bounding box in current Bayesian optimization practice is a hard constraint on the domain, our approach uses it simply to generate a regularizer which will focus the sequential search without constraining it. One clear advantage of our algorithm is that it adheres to the current user interface, allowing practitioners to continue simply specifying a reasonable range for each search dimension. This is arguably a much more natural requirement than a multidimensional centre and width parameters.

Finally, note that when users specify an arbitrary bounding box, they are in effect fixing $2d$ parameters. In that respect, our algorithm requires no more parameters than current practice, yet allows the search to progress outside of this arbitrary box, given a large enough budget.

6.4 Volume doubling

Finally, let us define the volume doubling heuristic. It consists of expanding the search space regularly as the optimization progresses, starting with an initial user-defined bounding box. This method otherwise follows the standard Bayesian optimization procedure and optimizes within the bounding box that is available at the given time step n. This approach requires two parameters: the number of iterations between expansions and the growth factor γ. Naturally, to avoid growing the feasible space $\mathcal{X}$ by a factor that is exponential in d, the growth factor applies to the volume of $\mathcal{X}$. Finally,

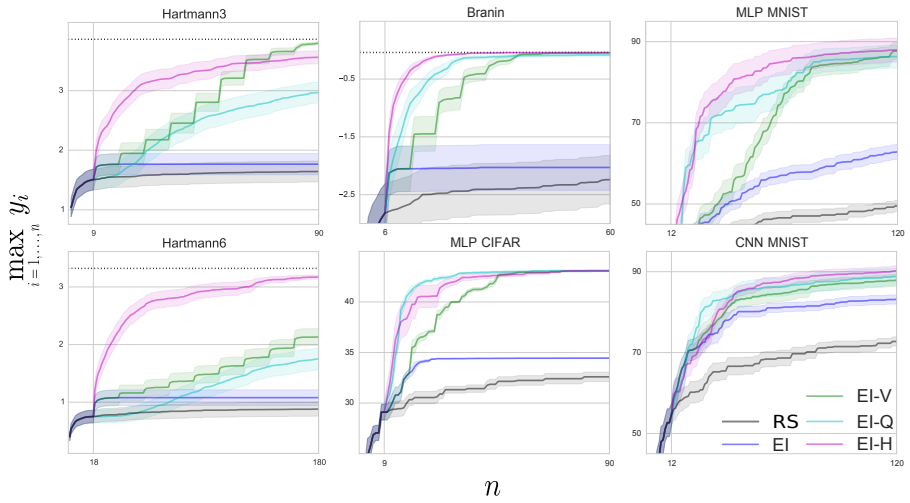


Figure 6.3: Best observations as optimization progresses. In the Hartmann experiments, the known optimum is represented as a horizontal dotted line. Plotting mean and standard error over 40 (Hartmann), 25 (MLP), and 20 (CNN) repetitions.

the expansion is isotropic about the centre of the domain. In this work, we double ($\gamma = 2$) the volume every $3d$ evaluations (only *after* an initial Latin hypercube sampling of $3d$ points).

6.5 Visualization

Before describing our experimental results, Figure 6.2 provides a visualization of EI with the hinge-quadratic prior mean, optimizing two toy problems in one and two dimensions. The objective functions simply consist of three distant Gaussian modes of varying heights and the initial bounding box is set such that it does not include the optimum. We draw attention to the way the space is gradually explored outward from the initial bounding box.

6.6 Experiments

In this section, we evaluate our proposed methods and show that they achieve the desirable behaviour on two synthetic benchmarking functions, and a simple task of tuning the stochastic gradient descent and regularization parameters used in training a multi-layered perceptron (MLP) and a convolutional neural network (CNN) on the MNIST dataset.

Experimental protocol. For every test problem of dimension d and every algorithm, the optimization was run with an overall evaluation budget of $30d$ including an initial $3d$ points sampled according to a Latin hypercube sampling scheme (as suggested in (Jones, 2001)). Throughout each particular run, at every iteration n we record the value of the best observation up to n and report these in Figure 6.3. Experiments were repeated to report and compare the mean and standard error of the algorithms: the synthetic experiments were repeated 40 times, while the MNIST experiments were repeated 25 and 20 times for the MLP and the CNN, respectively.

Algorithms. We compared the two different methods above—volume doubling and regularization—to the standard EI with a fixed bounding box. Common random seeds were used for all methods in order to reduce confounding noise. All algorithms were implemented in the `pybo` framework available on github,[3] and are labelled in the following figures as follows:

EI: Vanilla expected improvement with hyperparameter marginalization via MCMC.

EI-V: Expected improvement with the search volume doubled every $3d$ iterations.

EI-H: Regularized EI with a hinge-quadratic prior mean with $\beta = 1$ and R fixed by the circumradius of the initial bounding box.

[3] https://github.com/mwhoffman/pybo

EI-Q: Regularized EI with a quadratic prior mean where the widths $\mathbf{w}$ are fixed to those of the initial bounding box.

RS: As an additional benchmark, on the neural network tuning tasks, we considered a random selection strategy, which uniformly sampled within the user-defined bounding box.

Note that for the regularized methods EI-H/Q, the initial bounding box is only used to fix the location and scale of the regularizers, and to sample initial query points. In particular, both regularizers are centred around the box centre. For the quadratic regularizer the width of the box in each direction is used to fix $\mathbf{w}$, whereas for the hinge-quadratic R is set to the box circumradius. Once these parameters are fixed, the bounding box is no longer relevant, and more importantly, the algorithm is free to select points outside of it, see Figure 6.2 and 6.4 for example.

6.6.1 Synthetic benchmarks: Hartmann

The Hartmann 3 and 6 functions (numbers refer to their dimensionality) are standard, synthetic global optimization benchmarking test functions with known global optima. These are typically optimized in the unit hypercube $[0, 1]^d$, as we do in our Hartmann3 and 6 experiments.

In a separate experiment, indicated by an appended asterisk (*e.g.* Hartmann3$*$), we consider an initial bounding box of side length 0.2 centred uniformly at random within the unit hypercube. Each of the 40 repetitions of this experiment fixed a different such domain for all algorithms. The smaller domain has a 0.2^d probability of including the global optimum, especially unlikely in the six-dimensional problem. This experiment is meant to test whether our proposed methods are capable of useful exploration outside the initial bounding box and further compare them in such a situation.

6.6.2 MLP and CNN on MNIST

The MNIST hand-written digit recognition dataset is a very common task for testing neural network methods and architectures. Neural networks are

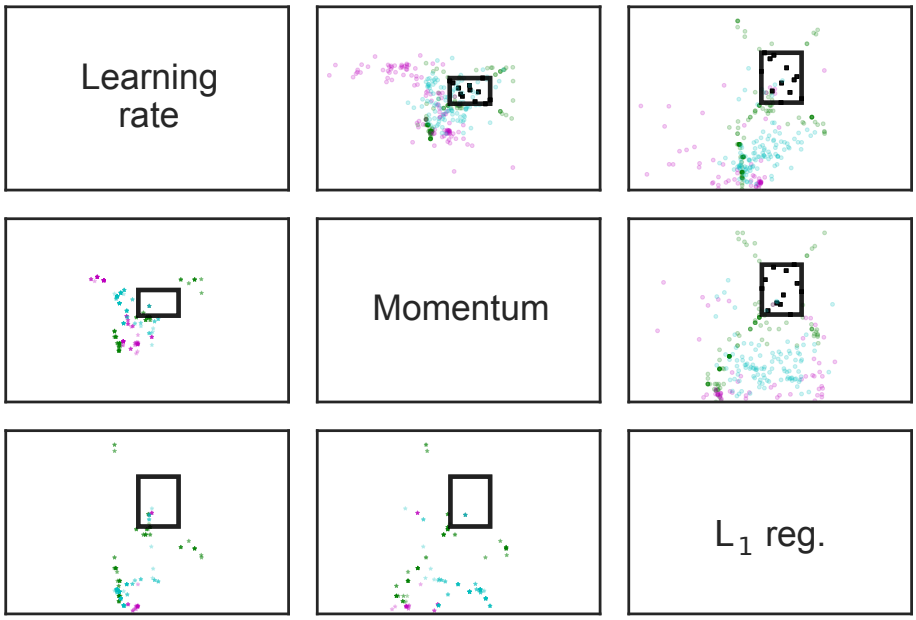


Figure 6.4: Pairwise scatterplot of selections (upper triangle) and recommendations (lower triangle) for the MLP-MNIST experiment. For example, the second plot of the first row corresponds to a scatter plot of the selected learning rates vs. momentum parameters *for a single seed*. In contrast, the first plot of the second row corresponds to a scatter plot of the recommended learning rates and momentum parameters *over all runs*. The initial bounding box and sample points (for this particular run) are shown as a black rectangle and black square dots, respectively. All other points respect the colour scheme of Figure 6.3. (the ℓ_2 regularization parameters were cropped out for space considerations.)

usually trained using some variant of stochastic gradient descent (SGD). The hyperparameters can impact both the speed of convergence and the quality of the trained network. We consider an MLP with 2048 hidden units with tanh non-linearities, and a CNN with two convolutional layers. These examples were taken from the official GitHub repository of `torch` demos.[4] The code written for this work can be readily extended to any other demo in the repository or in fact any script that can be run from the shell.

In this experiment, we optimize four parameters of the SGD optimizer,

[4]https://github.com/torch/demos

namely the learning rate and momentum, and the ℓ_1 and ℓ_2 regularization coefficients. The parameters were optimized in log space (base e) with an initial bounding box of $[-3, -1] \times [-3, -1] \times [-3, 1] \times [-3, 1]$, respectively. For each parameter setting, a black-box function evaluation corresponds to either training the MLP for 5 epochs or the CNN for 3, and returning the test set accuracy. To be clear, the goal of this experiment is not to achieve state-of-the-art for this classification task but instead to demonstrate that our proposed algorithms can find optima well outside their initial bounding boxes.

6.6.3 Results

Figure 6.3 shows that for the Hartmann tests, the proposed Bayesian optimization approaches work well in practice. The results confirm our hypothesis that the proposed methods are capable of useful exploration outside the initial bounding box. We note that when using the entire unit hypercube as the initial box, all the Bayesian optimization techniques exhibit similar performance as in this case the optimum is within the box. The Hartmann tests also show that the volume doubling heuristic is a good baseline method; and the plateaus suggest that this method warrants further study in, perhaps adaptive, scheduling strategies. Although it is less effective than EI-H as the dimensionality increases, it is nonetheless an improvement over standard EI in all cases.

The MNIST experiment shows good performance from all three methods EI-{V,H,Q}, particularly from the hinge-quadratic regularized algorithm. Indeed, when compared to the standard EI, EI-H boasts over 20% improvement in accuracy on the MLP and almost 10% on the CNN.

We believe that EI-H performs particularly well in settings where a small initial bounding box is prescribed because the hinge-quadratic regularizer allows the algorithm to explore outward more quickly. In contrast, EI-Q performs better when the optimum is included in the initial box; we suspect that this is due to the fact that the regularizer avoids selecting boundary and corner points, which EI and EI-V tend to do, as can be seen in Figure 6.4.

Figure 6.4 demonstrates that the algorithm in fact explores points outside the initially suggested bounded domain (drawn as a black rectangle). Indeed, while the green dots (EI-V) follow the corners of the growing bounding box, the magenta and cyan dots of EI-H/Q, respectively, do not exhibit this artifact.

6.7 Conclusion

In this work, we propose a new versatile approach to Bayesian optimization which is less sensitive to the initial choice of bounds because the algorithm is capable of growing its search space adaptively to an arbitrary size. Indeed, given a small initial bounding box that does not include the optimum, we have demonstrated that our approach can expand its region of interest and reach the optimum. This method has a *proximal* flavour; to select the next point, we seek a maximizer of the acquisition function that is not too far from the current neighbourhood.

Our method fits seamlessly within the current Bayesian optimization framework, and can be readily used with any acquisition function which is induced by a GP. Furthermore, all the extra hyperparameters but one (β) can readily be learned using existing hyperparameter sampling mechanisms.

Chapter 7

Discussion

This thesis contributed to both major components of the Bayesian optimization framework: modelling and decision-making. We validated our methods by extensive experimenting on ubiquitous synthetic test functions, real-world benchmarking tasks, as well as some new tasks we introduced. Moreover, throughout the thesis work, increasing effort was put on making our experiments reproducible and our software available online as free and open source.

The exposition of the background material, most of which can be found in our recent review paper (Shahriari et al., 2016b), often focused on hyperparameter tuning, which is of particular interest to the machine learning community. Nevertheless, the applicability of the method is much broader and we anticipate the model-based sequential optimization techniques reviewed in this thesis to find uses in many scientific and engineering fields.

7.1 Decision-making: unifying BayesGap and ESP

Since our work on ESP, the approach of Hernández-Lobato et al. (2014) known as PES has proven to be an efficient alternative to the Monte Carlo approximations carried out in Chapter 5. Our results would likely exhibit much less variance and be much more efficient using PES. One may ask whether it is worth maintaining a portfolio of candidates $\{\mathbf{x}^k\}$ when it is

possible to efficiently optimize the entropy directly over the entire input space $\mathcal{X}$. However, as we have seen in Chapter 4 as well as in the recent work of Russo and Van Roy (2014b) on information-directed sampling (IDS), combining regret-minimizing policies with more exploratory information-greedy approaches can be very effective. Indeed, similar to BayesGap, ESP sequentially reduces the entire search space $\mathcal{X}$ to a set of candidates, reminiscent of J and j in the BayesGap algorithm. The next query point is then selected among them using a measure of uncertainty, entropy in the case of ESP and posterior variance in the case of BayesGap.

Applied to parallel queries. The problem of optimization given a parallel query budget is often a much more realistic setting than the strictly sequential one. In many cases, the objective function can be evaluated multiple times in parallel, for instance, one often has access to a cluster of available compute nodes in order to parallelize hyperparameter tuning. There has been much recent work in batch Bayesian optimization where at each iteration the agent chooses a set of inputs to be queried next. Because this auxiliary task is a combinatorial optimization problem, most present solutions simulate the sequential process either by hallucinating pending observations (Azimi et al., 2010; Snoek et al., 2012), approximating the intractable maximum of a sequence of acquisition functions (Janusevskis et al., 2012), or conditioning on the predictive mean at pending query points (Desautels et al., 2014). This practice artificially reduces the variance at pending locations, which in turn causes these algorithms to output a diverse set of points. Moreover, for randomized acquisition functions such as Thompson sampling and its approximate extension to continuous spaces discussed in Appendix B, the algorithm can simply be run multiple times with different pseudo-random generator seeds to simply produce a batch Thompson sampling strategy. This last approach is an interesting one which, to the best of our knowledge, has yet to be studied in the literature.

For the approaches that simulate the sequential search, the acquisition functions that are used are designed to select the next query point so as to trade off exploration and exploitation for the next observation. Whereas this

makes sense when the optimizer can only run a single experiment, there may be a more optimal search strategy when multiple experiments can be run in parallel. In fact the burden of trading off exploration and exploitation is no longer on each individual experiment but on the collective, which allows the individual experiments to be more aggressively exploratory or exploitative. Our work on BayesGap and portfolio methods, provide a good way to obtain a diverse set or points, due to the different behaviours of different acquisition functions.

In practice, the recommendation strategy is to select the incumbent, which is the best observed query point (after smoothing). Such a recommendation strategy would benefit from dedicating a portion of the query batch to pure exploitation, which will result in refining the surrogate model around the current incumbent.

7.2 Contributions to the practice

Until very recently, the established Bayesian optimization practice required a user-defined bounded domain, which is assumed to contain the global optimizer. However, raised as an important remaining challenge at the 2014 NIPS workshop on Bayesian optimization, practitioners are often hesitant to prescribe a bounding box for the optimization domain because they are unwilling to completely discard large areas of the search domain. This is an unnecessary barrier to adoption that our work on unconstrained Bayesian optimization (Shahriari et al., 2016a) addressed via an intuitive regularization technique. While this shifts the difficulty of selecting a bounding box to that of choosing a regularizer, we proposed a simple and effective way to derive a regularizer from user-defined search ranges, which are often more intuitively set. Our experiments show that this approach is much more forgiving to a poor choices of search ranges as it can exceed them if necessary. Meanwhile setting hard constraints is still just as easy. Furthermore, though we experimented with expected improvement and Gaussian process models, the simplicity of the approach means that it can be extended to many more models and acquisition functions.

Indeed, that work featured an important experimental component, and the regularized methods proved effective on the task of tuning the stochastic gradient descent optimizer of two neural network architectures on two famous object recognition datasets: MNIST and CIFAR.

7.3 Future work: first order Bayesian optimization

Recently, we have also been exploring incorporating first order information, such as gradients and directional derivatives in the Bayesian optimization surrogate model. Although the use of Gaussian processes with derivative information is not new (see for example Rasmussen and Williams, 2006; Lizotte, 2008; Osborne, 2010) recent work by Maclaurin et al. (2015) has provided renewed motivation for this line of research. Indeed, in the context of tuning the hyperparameters of a neural network, they are capable of back-propagating through gradient descent iterations in order to obtain gradients of the empirical loss with respect to said hyperparameters (Maclaurin et al., 2015).

Whereas they use this so-called "hypergradient" to tune the hyperparameters via gradient descent, we are exploring how much the gradients can accelerate the convergence of Bayesian optimization. In addition, since including gradient information increases the computational complexity of exact GP inference by a factor of d^3 where d is the number of hyperparameters, we are also exploring including directional derivatives, which would increase complexity by a fixed factor independent of d. Eventually, we wish to prove convergence rates for Bayesian optimization that are better than the existing results (Srinivas et al., 2010; Bull, 2011) thanks to derivative information, which will hopefully resemble those of gradient descent.

As a practical application, consider training a recurrent neural networks (RNN). Indeed, training techniques such as backpropagation through time (BPTT) are known to produce noisy gradients and lead to error magnification due to the recurrent nature of the model. As mentioned above, Bayesian

optimization can elegantly incorporate such noisy gradients and derivatives, as well as all the observations of the empirical loss, hopefully leading to faster convergence to better (global) optima. However, the challenge for Bayesian optimization is to scale to the high-dimensional search space and the many observations such optimization will certainly require.

Fortunately, there has been much interest in compressing neural networks using Fourier or cosine transforms, *e.g.,* work by Koutnik et al. (2010) and more recently the ACDC work of Moczulski et al. (2015). Such techniques could potentially reduce the dimensionality of a neural network of a significant and useful size down to a manageable number in the hundreds and hopefully dozens. Meanwhile, in the Bayesian optimization community, Kandasamy et al. (2015) have used additive kernels successfully to optimize functions of dozens of variables; and Kandasamy's latest unpublished model, SALSA (Shrunk Additive Least Squares Approximation), seems to improve on this even further. These two approaches could be combined to train recurrent neural networks using Bayesian optimization, in other words, learn a good set of weights for the neural network. Note that this is in contrast to existing Bayesian optimization work that aims to tune the hyperparameters rather than the weights.

7.4 Final word: from academia to societal impact

Many real world implementations could—and, in our opinion, should—incorporate the Bayesian optimization framework of (i) distilling observed data into interpretable probabilistic surrogate models to facilitate (ii) principled, (near-)optimal, justifiable, and (semi-)automatic decision-making. This is often a challenging automation task because it involves intervention, and people are understandably reluctant to surrender such agency to an automatic mechanism. Hence our emphasis on interpretable models and justifiable decision-making, that will likely, at least initially, be involve some form of human oversight: driver-less cars are a timely example. Large scale projects such as kernel learning (Wilson, 2014) and the automatic statistician Duvenaud et al. (2013) are important steps towards statistical inter-

pretability. Combined with Bayesian optimization, these solutions have the potential to have a tremendous positive impact on accurate, efficient, and data-driven medical diagnostic tools to aid medical staff, and similarly for education and environmental management initiatives.

Bibliography

Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2312–2320. Curran Associates, Inc., 2011. 18, 40

S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*, 2013. 41

C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1-2):5–43, 2003. ISSN 0885-6125. 23

S. Arlot, A. Celisse, et al. A survey of cross-validation procedures for model selection. *Statistics surveys*, 4:40–79, 2010. 56

J. Audibert, S. Bubeck, and R. Munos. Best arm identification in multi-armed bandits. In *Conference on Learning Theory*, 2010. 54

P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3:397–422, 2003. 18, 40, 54

P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Symposium on Foundations of Computer Science*, pages 322–331, 1995. 14

J. Azimi, A. Fern, and X. Fern. Batch bayesian optimization via simulation matching. NIPS, 2010. 88

J. Azimi, A. Jalali, and X. Fern. Hybrid batch bayesian optimization. In *ICML*, 2012. 2

J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, pages 2546–2554, 2011. 2, 55

S. Bochner. *Lectures on Fourier integrals*. Princeton University Press, 1959. 42, 112

L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 121

E. Brochu, N. de Freitas, and A. Ghosh. Active preference learning with discrete choice data. In *Advances in Neural Information Processing Systems*, pages 409–416, 2007. 2, 55

E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical Report UBC TR-2009-23 and arXiv:1012.2599v1, Dept. of Computer Science, University of British Columbia, 2009. 2, 70

E. Brochu, T. Brochu, and N. de Freitas. A Bayesian interactive optimization approach to procedural animation design. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 103–112, 2010. 2, 55

S. Bubeck, R. Munos, and G. Stoltz. Pure exploration in multi-armed bandits problems. In *Algorithmic Learning Theory*, pages 23–37. Springer, 2009. 33, 44

S. Bubeck, R. Munos, G. Stoltz, and C. Szepesvari. X-armed bandits. *Journal of Machine Learning Research*, 12:1655–1695, 2011. 13, 18

A. D. Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879–2904, 2011. 38, 90

P. Burman. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, 76(3): 503–514, 1989. 8, 56

R. Calandra, N. Gopalan, A. Seyfarth, J. Peters, and M. Deisenroth. Bayesian gait optimization for bipedal locomotion. In *Learning and Intelligent OptimizatioN (LION8)*, pages 274–290, 2014. 2

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, 2006. 8

K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *J. of Stat. Science*, 10:273–304, 1995. ii

O. Chapelle and L. Li. An empirical evaluation of Thompson sampling. In *Advances in Neural Information Processing Systems*, pages 2249–2257, 2011. 2, 23, 41

H. Chen, J. L. Loeppky, J. Sacks, W. J. Welch, et al. Analysis methods for computer experiments: How to assess and what counts? *Statistical Science*, 31(1):40–60, 2016. 27

I. Clark and W. V. Harper. *Practical Geostatistics 2000: Case Studies*. Ecosse North America, 2008. 71

A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 7(2–3):81–227, 2011. 121

V. Dani, T. P. Hayes, and S. M. Kakade. Stochastic linear optimization under bandit feedback. In *Conference on Learning Theory*, pages 355–366, 2008. 18, 40

N. de Freitas, A. Smola, and M. Zoghi. Exponential regret bounds for Gaussian process bandits with deterministic observations. In *International Conference on Machine Learning*, 2012. 40

P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006. 128

T. Desautels, A. Krause, and J. Burdick. Parallelizing exploration-exploitation tradeoffs with Gaussian process bandit optimization. *Journal of Machine Learning Research*, 2014. 40, 88

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics letters B*, 195(2):216–222, 1987. 125

D. Duvenaud, J. R. Lloyd, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. In *Proceedings of the 30th International Conference on Machine Learning*, June 2013. 91

P. Frazier, W. Powell, and S. Dayanik. The knowledge-gradient policy for correlated normal beliefs. *INFORMS journal on Computing*, 21(4):599–613, 2009. 36

V. Gabillon, M. Ghavamzadeh, A. Lazaric, and S. Bubeck. Multi-bandit best arm identification. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 2222–2230, 2011. 45

V. Gabillon, M. Ghavamzadeh, and A. Lazaric. Best arm identification: A unified approach to fixed budget and fixed confidence. In *Advances in Neural Information Processing Systems*, pages 3212–3220, 2012. ii, 8, 45, 49, 54, 105

R. Garnett, M. A. Osborne, and S. J. Roberts. Bayesian optimization for

sensor set selection. In *ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 209–219. ACM, 2010. 2

J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 148–177, 1979. ii, 7, 35

J. C. Gittins, K. Glazebrook, and R. Weber. *Multi-armed bandit allocation indices*. Wiley, 2011. 35

R. B. Gramacy and N. G. Polson. Particle learning of Gaussian process models for sequential design and optimization. *Journal of Computational and Graphical Statistics*, 20(1):102–118, 2011. 127, 129

F. Hamze, Z. Wang, and N. de Freitas. Self-avoiding random dynamics on integer complex systems. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 23(1):9, 2013. 55

P. Hennig and C. J. Schuler. Entropy search for information-efficient global optimization. *The Journal of Machine Learning Research*, pages 1809–1837, 2012. 9, 42, 43, 60, 65, 75

J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems*. 2014. 42, 43, 60, 61, 66, 73, 75, 76, 87

M. W. Hoffman, H. Kueck, N. de Freitas, and A. Doucet. New inference strategies for solving Markov decision processes using reversible jump MCMC. In *Uncertainty in Artificial Intelligence*, pages 223–231, 2009. 73

M. W. Hoffman, E. Brochu, and N. de Freitas. Portfolio allocation for Bayesian optimization. In *Uncertainty in Artificial Intelligence*, pages 327–336, 2011. ii, 9, 59, 67, 69, 73

M. W. Hoffman, B. Shahriari, and N. de Freitas. On correlation and budget constraints in model-based bandit optimization with application to automatic machine learning. In *AI and Statistics*, pages 365–374, 2014. iii, 2, 8

F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *LION*, pages 507–523, 2011. 2, 121

F. Hutter, H. H. Hoos, and K. Leyton-Brown. Parallel algorithm configuration. In *LION*, pages 55–70, 2012. 55

J. Janusevskis, R. Le Riche, D. Ginsbourger, and R. Girdziusas. *Learning and Intelligent Optimization: 6th International Conference, LION 6, Paris, France, January 16-20, 2012, Revised Selected Papers*, chapter Expected Improvements for the Asynchronous Parallel Global Optimization of Expensive Functions: Potentials and Challenges, pages 413–418. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-34413-8. doi: 10.1007/978-3-642-34413-8_37. URL `http://dx.doi.org/10.1007/978-3-642-34413-8_37`. 88

D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *J. of Global Optimization*, 21(4):345–383, 2001. 7, 38, 39, 70, 73, 77, 82

D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *J. of Global optimization*, 13(4):455–492, 1998. 2, 7, 38, 75

K. Kandasamy, J. Schneider, and B. Poczos. High dimensional bayesian optimisation and bandits via additive models. In *International Conference on Machine Learning*, 2015. 91

E. Kaufmann, O. Cappé, and A. Garivier. On bayesian upper confidence bounds for bandit problems. In *International Conference on Artificial Intelligence and Statistics*, pages 592–600, 2012a. 40, 51, 54

E. Kaufmann, N. Korda, and R. Munos. Thompson sampling: An asymptotically optimal finite-time analysis. In *Algorithmic Learning Theory*, volume 7568 of *Lecture Notes in Computer Science*, pages 199–213. Springer Berlin Heidelberg, 2012b. 41

J. Koutnik, F. Gomez, and J. Schmidhuber. Evolving neural networks in compressed weight space. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, GECCO '10, pages 619–626, 2010. 91

D. G. Krige. A statistical approach to some basic mine valuation problems on the witwatersrand. *Journal of Chemical, Metallurgical, and Mining Society of South Africa*, 52(6), 1951. ii, 7

H. J. Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Fluids Engineering*, 86(1):97–106, 1964. 7, 38, 75

T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985. 7, 33, 39

M. Lázaro-Gredilla, J. Quiñonero-Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal. Sparse spectrum Gaussian process regression. *The Journal of Machine Learning Research*, 11:1865–1881, 2010. 42, 119, 120

D. V. Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, pages 986–1005, 1956. ii, 37

J. S. Liu and R. Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American statistical association*, 93(443):1032–1044, 1998. 127

D. Lizotte. *Practical Bayesian Optimization*. PhD thesis, University of Alberta, Canada, 2008. 55, 70, 90

D. Lizotte, T. Wang, M. Bowling, and D. Schuurmans. Automatic gait optimization with Gaussian process regression. In *Proc. of IJCAI*, pages 944–949, 2007. 2

D. Maclaurin, D. Duvenaud, and R. P. Adams. Gradient-based hyperparameter optimization through reversible learning. In *Proceedings of the 32nd International Conference on Machine Learning*, July 2015. 90

N. Mahendran, Z. Wang, F. Hamze, and N. de Freitas. Adaptive MCMC with Bayesian optimization. *Journal of Machine Learning Research - Proceedings Track*, 22:751–760, 2012. 2, 55

R. Marchant and F. Ramos. Bayesian optimisation for intelligent environmental monitoring. In *NIPS workshop on Bayesian Optimization and Decision Making*, 2012. 2

O. Maron and A. W. Moore. Hoeffding races: Accelerating model selection search for classification and function approximation. *Robotics Institute*, page 263, 1993. 18, 55

R. Martinez-Cantin, N. de Freitas, A. Doucet, and J. A. Castellanos. Active policy learning for robot planning and exploration under uncertainty. *Robotics Science and Systems*, 2007. 2, 55

V. Mnih, C. Szepesvári, and J.-Y. Audibert. Empirical Bernstein stopping. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 672–679. ACM, 2008. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390241. 18

J. Močkus, V. Tiesis, and A. Žilinskas. The application of bayesian methods for seeking the extremum. In L. Dixon and G. Szego, editors, *Toward Global Optimization*, volume 2. Elsevier, 1978. 7, 38

M. Moczulski, M. Denil, J. Appleyard, and N. de Freitas. Acdc: A structured efficient linear layer. Technical report, University of Oxford, 2015. 91

R. Munos. Optimistic optimization of a deterministic function without the knowledge of its smoothness. In *Advances in Neural Information Processing Systems*, pages 783–791, 2011. 18

I. Murray and R. P. Adams. Slice sampling covariance hyperparameters of latent Gaussian models. In *Advances in Neural Information Processing Systems*, pages 1732–1740, 2010. 125

R. M. Neal. Slice sampling. *Annals of statistics*, pages 705–741, 2003. 126

J. Neufeld, A. Gyorgy, D. Schuurmans, and C. Szepesvari. Adaptive Monte-Carlo via bandit allocation. In *Proceedings of the International Conference on Machine Learning*, ICML, 2014. 59

M. Osborne. *Bayesian Gaussian Processes for Sequential Prediction, Optimisation and Quadrature*. PhD thesis, University of Oxford, Oxford, UK, 2010. 90

J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959, 2005. 119

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pages 1177–1184, 2007. 42, 112

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006. 23, 26, 90

J. Rosenthal. *A First Look at Rigorous Probability Theory*. World Scientific, 2006. ISBN 9789812703705. 12

D. Russo and B. Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014a. 8, 41

D. Russo and B. Van Roy. Learning to optimize via information-directed sampling. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1583–1591. 2014b. 88

J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical science*, pages 409–423, 1989. ii, 2, 7

S. L. Scott. A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658, 2010. 2, 23, 41

M. Seeger, C. Williams, and N. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In *Artificial Intelligence and Statistics*, volume 9, 2003. 118, 119

B. Shahriari, Z. Wang, M. W. Hoffman, A. Bouchard-Côté, and N. de Freitas. An entropy search portfolio. In *NIPS workshop on Bayesian Optimization*, 2014. 9, 10, 42

B. Shahriari, A. Bouchard-Côté, and N. de Freitas. Unbounded Bayesian optimization via regularization. In *AISTATS*, 2016a. iii, 10, 89

B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of bayesian optimization. In *Proceedings of the IEEE*, volume 104, pages 1–28, 2016b. iii, 87

H. Shen, W. J. Welch, and J. M. Hughes-Oliver. Efficient, adaptive cross-validation for tuning and comparing models, with application to drug discovery. *The Annals of Applied Statistics*, pages 2668–2687, 2011. 55

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pages 1257–1264, 2005. 118, 119

J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, pages 2951–2959, 2012. 2, 39, 55, 68, 88, 125

M. Soare, A. Lazaric, and R. Munos. Best-arm identification in linear bandits. In *Advances in Neural Information Processing Systems*, pages 828–836, 2014. 8

N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In

International Conference on Machine Learning, pages 1015–1022, 2010. 2, 8, 13, 40, 51, 52, 53, 54, 69, 90

K. Swersky, J. Snoek, and R. P. Adams. Multi-task Bayesian optimization. In *Advances in Neural Information Processing Systems*, pages 2004–2012, 2013. 2

W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4): 285–294, 1933. ii, 6, 13, 40

C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Knowledge Discovery and Data Mining*, pages 847–855, 2013. 2

M. K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *International Conference on Artificial Intelligence and Statistics*, pages 567–574, 2009. 119

M. Valko, A. Carpentier, and R. Munos. Stochastic simultaneous optimistic optimization. In *International Conference on Machine Learning*, 2013. 18

H. P. Vanchinathan, I. Nikolic, F. De Bona, and A. Krause. Explore-exploit in top-N recommender systems via Gaussian processes. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 225–232. ACM, 2014. 40

J. Villemonteix, E. Vazquez, and E. Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *J. of Global Optimization*, 44(4):509–534, 2009. 9, 42, 43, 60, 65, 75

Z. Wang, S. Mohamed, and N. de Freitas. Adaptive Hamiltonian and Riemann manifold Monte Carlo samplers. In *International Conference on Machine Learning*, pages 1462–1470, 2013a. 2, 55

Z. Wang, M. Zoghi, D. Matheson, F. Hutter, and N. de Freitas. Bayesian optimization in high dimensions via random embeddings. In *International Joint Conference on Artificial Intelligence*, pages 1778–1784, 2013b. 2

E. Westervelt and J. Grizzle. *Feedback Control of Dynamic Bipedal Robot Locomotion*. Control and Automation Series. CRC PressINC, 2007. ISBN 9781420053722. 72

A. G. Wilson. *Covariance kernels for fast automatic pattern discovery and extrapolation with Gaussian processes*. PhD thesis, University of Cambridge, 2014. 91

Appendix A

BayesGap theory

A.1 Proof of Theorem 2

The proof of this section and the lemmas of the next section follow from the proofs of Gabillon et al. (2012). The modifications we have made to this proof correspond to the introduction of the function g_k which bounds the uncertainty s_k in order to make it simpler to introduce other models. We also introduce a sufficient condition on this bound, i.e. that it is monotonically decreasing in N in order to bound the arm pulls with respect to g_k^{-1}. Ultimately, this form of the theorem reduces the problem of of proving a regret bound to that of checking a few properties of the uncertainty model.

Theorem 2
Consider a bandit problem with horizon T and K arms. Let $U_k(t)$ and $L_k(t)$ be upper and lower bounds that hold for all times $t \leq T$ and all arms $k \leq K$ with probability $1-\delta$. Finally, let g_k be a monotonically decreasing function such that $s_k(t) \leq g_k(N_k(t-1))$ and $\sum_k g_k^{-1}(H_{k\epsilon}) \leq T - K$. We can then bound the simple regret as

$$\mathbb{P}(S_T \leq \epsilon) \geq 1 - KT\delta. \tag{A.1}$$

Proof. We will first define the event $\mathcal{E}$ such that on this event every mean is bounded by its associated bounds for all times t. More precisely we can

write this as

$$\mathcal{E} = \{\forall k \leq K, \forall t \leq T, L_k(t) \leq z_k \leq U_k(t)\}. \tag{A.2}$$

By definition, these bounds are given such that the probability of deviating from a single bound is δ. Using a union bound we can then bound the probability of remaining within all bounds as $\mathbb{P}(\mathcal{E}) \geq 1 - KT\delta$.

We will next condition on the event $\mathcal{E}$ and assume regret of the form $S_T > \epsilon$ in order to reach a contradiction. Upon reaching said contradiction we can then see that the simple regret must be bounded by ϵ with probability given by the probability of event $\mathcal{E}$, as stated above. As a result we need only show that a contradiction occurs.

We will now define $\hat{t} = \arg\min_{t \leq T} B_{J(t)}(t)$ as the time at which the recommended arm attains the minimum bound, i.e. $\hat{a}_T = J(\hat{t})$ as defined in (4.5). Let $t_k \leq T$ be the last time at which arm k is pulled. Note that each arm must be pulled at least once due to the initialization phase. We can then show the following sequence of inequalities:

$$\begin{aligned}
\min(0, s_k(t_k) - \Delta_k) + s_k(t_k) &\geq B_{J(t_k)}(t_k) && \text{(a)} \\
&\geq B_{\Omega_T}(\hat{t}) && \text{(b)} \\
&\geq R_{\Omega_T} && \text{(c)} \\
&> \epsilon. && \text{(d)}
\end{aligned}$$

Of these inequalities, (a) holds by Lemma B3, (c) holds by Lemma B1, and (d) holds by our assumption on the simple regret. The inequality (b) holds due to the definition $\hat{a}_T$ and time $\hat{t}$. Note, that we can also write the preceding inequality as two cases

$$s_k(t_k) > 2s_k(t_k) - \Delta_k > \epsilon, \quad \text{if } \Delta_k > s_k(t_k); \tag{A.3}$$

$$2s_k(t_k) - \Delta_k \geq s_k(t_k) > \epsilon, \quad \text{if } \Delta_k \leq s_k(t_k) \tag{A.4}$$

This leads to the following bound on the confidence diameter,

$$s_k(t_k) > \max(\tfrac{1}{2}(\Delta_k + \epsilon), \epsilon) = H_{k\epsilon} \tag{A.5}$$

which can be obtained by a simple manipulation of the above equations. More precisely we can notice that in each case, $s_k(t_k)$ upper bounds both ϵ and $\frac{1}{2}(\Delta_k + \epsilon)$, and thus it obviously bounds their maximum.

Now, for any arm k we can consider the final number of arm pulls, which we can write as

$$N_k(T) = N_k(t_k - 1) + 1 \leq g^{-1}(s_k(t_k)) + 1 \tag{A.6}$$

$$< g^{-1}(H_{k\epsilon}) + 1. \tag{A.7}$$

This holds due to the definition of g as a monotonic decreasing function, and the fact that we pull each arm at least once during the initialization stage. Finally, by summing both sides with respect to k we can see that $\sum_k g^{-1}(H_{k\epsilon}) + K > T$, which contradicts our definition of g in the Theorem statement. □

A.2 Lemmas

In order to simplify notation in this section, we will first introduce $B(t) = \min_k B_k(t)$ as the minimizer over all gap indices for any time t. We will also note that this term can be rewritten as

$$B(t) = B_{J(t)}(t) = U_{j(t)}(t) - L_{J(t)}(t), \tag{A.8}$$

which holds due to the definitions of $j(t)$ and $J(t)$.

Lemma B1
For any sub-optimal arm $k \neq a_\star$, any time $t \in \{1, \ldots, T\}$, and on event $\mathcal{E}$, the immediate regret of pulling that arm is upper bounded by the index quantity, i.e. $B_k(t) \geq R_k$.

Proof. We can start from the definition of the bound and expand this term

as

$$B_k(t) = \max_{i \neq k} U_i(t) - L_k(t) \tag{A.9}$$

$$\geq \max_{i \neq k} z_i - z_k = z_\star - z_k = R_k. \tag{A.10}$$

The first inequality holds due to the assumption of event $\mathcal{E}$, whereas the following equality holds since we are only considering sub-optimal arms, for which the best alternative arm is obviously the optimal arm. □

Lemma B2

For any time t let $k = a_t$ be the arm pulled, for which the following statements hold:

$$\text{if } k = j(t), \text{ then } L_{j(t)}(t) \leq L_{J(t)}(t), \tag{A.11}$$

$$\text{if } k = J(t), \text{ then } U_{j(t)}(t) \leq U_{J(t)}(t). \tag{A.12}$$

Proof. We can divide this proof into two cases based on which of the two arms is selected.

Case 1: let $k = j(t)$ be the arm selected. We will then assume that $L_{j(t)}(t) > L_{J(t)}(t)$ and show that this is a contradiction. By definition of the arm selection rule we know that $s_{j(t)}(t) \geq s_{J(t)}(t)$, from which we can easily deduce that $U_{j(t)}(t) > U_{J(t)}(t)$ by way of our first assumption. As a result we can see that

$$B_{j(t)}(t) = \max_{j \neq j(t)} U_j(t) - L_{j(t)}(t) \tag{A.13}$$

$$< \max_{j \neq J(t)} U_j(t) - L_{J(t)}(t) = B_{J(t)}(t). \tag{A.14}$$

This inequality holds due to the fact that arm $j(t)$ must necessarily have the highest upper bound over all arms. However, this contradicts the definition of $J(t)$ and as a result it must hold that $L_{j(t)}(t) \leq L_{J(t)}(t)$.

Case 2: let $k = J(t)$ be the arm selected. The proof follows the same format as that used for $k = j(t)$. □

Corollary B2
If arm $k = a_t$ is pulled at time t, then the minimum index is bounded above by the uncertainty of arm k, or more precisely

$$B(t) \leq s_k(t). \tag{A.15}$$

Proof. We know that k must be restricted to the set $\{j(t), J(t)\}$ by definition. We can then consider the case that $k = j(t)$, and by Lemma B2 we know that this imposes an order on the lower bounds of each possible arm, allowing us to write

$$B(t) \leq U_{j(t)}(t) - L_{j(t)}(t) = s_{j(t)}(t) \tag{A.16}$$

from which our corollary holds. We can then easily see that a similar argument holds for $k = J(t)$ by ordering the upper bounds, again via Lemma B2. □

Lemma B3
On event $\mathcal{E}$, for any time $t \in \{1, \ldots, T\}$, and for arm $k = a_t$ the following bound holds on the minimal gap,

$$B(t) \leq \min(0, s_k(t) - \Delta_k) + s_k(t). \tag{A.17}$$

Proof. In order to prove this lemma we will consider a number of cases based on which of $k \in \{j(t), J(t)\}$ is selected and whether or not one or neither of these arms corresponds to the optimal arm $a_\star$. Ultimately, this results in six cases, the first three of which we will present are based on selecting arm $k = j(t)$.

Case 1: consider $a_\star = k = j(t)$. We can then see that the following sequence of inequalities holds,

$$z_{(2)} \overset{(a)}{\geq} z_{J(t)}(t) \overset{(b)}{\geq} L_{J(t)}(t) \overset{(c)}{\geq} L_{j(t)}(t) \overset{(d)}{\geq} z_k - s_k(t). \tag{A.18}$$

Here (b) and (d) follow directly from event $\mathcal{E}$ and (c) follows from Lemma B2.

Inequality (a) follows trivially from our assumption that $k = a_\star$, as a result $J(t)$ can only be as good as the 2nd-best arm. Using the definition of Δ_k and the fact that $k = a_\star$, the above inequality yields

$$s_k(t) - (z_k - z_{(2)}) = s_k(t) - \Delta_k \geq 0 \tag{A.19}$$

Therefore the min in the result of Lemma B3 vanishes and the result follows from Corollary B2.

Case 2: consider $k = j(t)$ and $a_\star = J(t)$. We can then write

$$\begin{aligned} B(t) &= U_{j(t)}(t) - L_{J(t)}(t) && \text{(A.20)} \\ &\leq z_{j(t)}(t) + s_{j(t)}(t) - z_{J(t)}(t) + s_{J(t)}(t) && \text{(A.21)} \\ &\leq z_k - z_\star + 2s_k(t) && \text{(A.22)} \end{aligned}$$

where the first inequality holds from event $\mathcal{E}$, and the second holds because by definition the selected arm must have higher uncertainty. We can then simplify this as

$$\begin{aligned} &= 2s_k(t) - \Delta_k && \text{(A.23)} \\ &\leq \min(0, s_k(t) - \Delta_k) + s_k(t), && \text{(A.24)} \end{aligned}$$

where the last step evokes Corollary B2.

Case 3: consider $k = j(t) \neq a_\star$ and $J(t) \neq a_\star$. We can then write the following sequence of inequalities,

$$z_{j(t)}(t) + s_{j(t)}(t) \overset{(a)}{\geq} U_{j(t)}(t) \overset{(b)}{\geq} U_{a_\star}(t) \overset{(c)}{\geq} z_\star. \tag{A.25}$$

Here (a) and (c) hold due to event $\mathcal{E}$ and (b) holds since by definition $j(t)$ has the highest upper bound other than $J(t)$, which in turn is not the optimal arm by assumption in this case. By simplifying this expression we obtain $s_k(t) - \Delta_k \geq 0$, and hence the result follows from Corollary B2 as in Case 1.

Cases 4–6: consider $k = J(t)$. The proofs for these three cases follow the same general form as the above cases and is omitted. Cases 1 through

6 cover all possible scenarios and prove Lemma B3. □

Lemma B4

Consider a normally distributed random variable $X \sim \mathcal{N}(\mu, \sigma^2)$ and $\beta \geq 0$. The probability that X is within a radius of $\beta\sigma$ from its mean can then be written as

$$\mathbb{P}\big(|X - \mu| \leq \beta\sigma\big) \geq 1 - e^{-\beta^2/2}. \tag{A.26}$$

Proof. Consider $U \sim \mathcal{N}(0, 1)$. The probability that U exceeds some positive bound $c > 0$ can be written

$$\begin{aligned}
\mathbb{P}(U > c) &= \frac{e^{-c^2/2}}{\sqrt{2\pi}} \int_c^\infty e^{(c^2-u^2)/2}\, du \\
&= \frac{e^{-c^2/2}}{\sqrt{2\pi}} \int_c^\infty e^{-(u-c)^2/2 - c(u-c)}\, du \\
&\leq \frac{e^{-c^2/2}}{\sqrt{2\pi}} \int_c^\infty e^{-(u-c)^2/2}\, du \\
&\leq \tfrac{1}{2} e^{-c^2/2}.
\end{aligned}$$

The first inequality holds due to the fact that $e^{-c(u-c)} \leq 1$ for $u \geq c$, and the second holds since $c > 0$. Using a union bound we can then bound both sides as $\mathbb{P}(|U| > c) \leq e^{-c^2/2}$. Finally, by setting $U = (X - \mu)/\sigma$ and $c = \beta$ we obtain the bound stated above. □

Appendix B

Approximate sampling from posterior global minimizers

Given a continuous positive stationary kernel k_0, Bochner (1959) asserts the existence of its Fourier dual $s(\mathbf{w})$, the spectral density of k_0. We can normalize it, letting $p(\mathbf{w}) = s(\mathbf{w})/\alpha$ be the associated normalized density, we can write the kernel as the expectation

$$k_0(\mathbf{x}, \mathbf{x}') = \alpha \, \mathbb{E}_{p(\mathbf{w})}[e^{-i\mathbf{w}^\mathsf{T}(\mathbf{x}-\mathbf{x}')}] \tag{B.1}$$

$$= 2\alpha \, \mathbb{E}_{p(\mathbf{w},b)}[\cos(\mathbf{w}^\mathsf{T}\mathbf{x} + b)\cos(\mathbf{w}^\mathsf{T}\mathbf{x}' + b)] \,, \tag{B.2}$$

where $b \sim \mathcal{U}[0, 2\pi]$. Let $\boldsymbol{\phi}(\mathbf{x}) = \sqrt{2\alpha/m}\cos(\mathbf{W}\mathbf{x} + \mathbf{b})$ denote an m-dimensional feature mapping and where $\mathbf{W}$ and $\mathbf{b}$ consist of m stacked samples from $p(\mathbf{w}, b)$. The kernel k can then be approximated by the inner product of these features, $k_0(\mathbf{x}, \mathbf{x}') \approx \boldsymbol{\phi}(\mathbf{x})^\mathsf{T}\boldsymbol{\phi}(\mathbf{x}')$. This approach was used by (Rahimi and Recht, 2007) as an approximation method in the context of kernel methods. The feature mapping $\boldsymbol{\phi}(\mathbf{x})$ allows us to approximate the Gaussian process prior for f with a linear model $f(\mathbf{x}) = \boldsymbol{\phi}(\mathbf{x})^\mathsf{T}\boldsymbol{\theta}$ where $\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is a standard multivariate normal. By conditioning on $\mathcal{D}$, the posterior for $\boldsymbol{\theta}$ is also multivariate normal $\boldsymbol{\theta} \mid \mathcal{D} \sim \mathcal{N}(\mathbf{A}^{-1}\boldsymbol{\Phi}^\mathsf{T}\mathbf{y}, \eta^2\mathbf{A}^{-1})$ where $\mathbf{A} = \boldsymbol{\Phi}^\mathsf{T}\boldsymbol{\Phi} + \eta^2\mathbf{I}$, $\mathbf{y}$ is the vector of the output data, and $\boldsymbol{\Phi}$ is a matrix of features evaluated on the input data.

Let $\boldsymbol{\phi}^{(i)}$ and $\boldsymbol{\theta}^{(i)}$ be a random set of features and the corresponding posterior weights sampled both according to the generative process given above. They can then be used to construct the function $f^{(i)}(\mathbf{x}) = \boldsymbol{\phi}^{(i)}(\mathbf{x})^\intercal \boldsymbol{\theta}^{(i)}$, which is an approximate posterior sample of f—albeit one with a finite parameterization. We can then maximize this function to obtain $\mathbf{z}^{(i)} = \arg\min_{\mathbf{x} \in \mathcal{X}} f^{(i)}(\mathbf{x})$, which is approximately distributed according to $\mathbb{P}_\star(\,\cdot \mid \mathcal{D})$. To produce samples from this process it is also necessary to derive the spectral density for any kernel of interest. This quantity is given by the kernel's Fourier transform $s(\mathbf{w}) = \frac{1}{(2\pi)^d} \int e^{i \mathbf{w}^\intercal \boldsymbol{\tau}} k(\boldsymbol{\tau}, \mathbf{0}) \, \mathrm{d}\boldsymbol{\tau}$. For example, for the squared-exponential and Matérn kernels the normalized spectral densities take the form

$$\mathbf{w}_{\text{SE}} \sim \mathcal{N}(0, \Lambda) \tag{B.3}$$

$$\mathbf{w}_{\text{MATÉRN}} \sim \mathcal{T}(0, \Lambda, \tfrac{5}{2}), \tag{B.4}$$

i.e. these are distributed as a normal and Student's t respectively, which are easy to sample from. Recall that Λ is the diagonal matrix of length scale parameters for ARD kernels.

Appendix C

Gradients of acquisition functions

In this chapter, we provide some gradients of popular acquisition functions that came in handy when implementing the Bayesian optimization approaches.

Consider the following GP posterior mean and variance

$$\mu(\mathbf{x}) = \mathbf{k}^T(\mathbf{x})\mathbf{K}^{-1}\mathbf{y} \tag{C.1}$$

$$\sigma^2(\mathbf{x}) = k(\mathbf{x},\mathbf{x}) - \mathbf{k}(\mathbf{x})\mathbf{K}^{-1}\mathbf{k}(\mathbf{x}), \tag{C.2}$$

which we readily differentiate to obtain

$$\nabla\mu(\mathbf{x}) = \nabla\mathbf{k}^T(\mathbf{x})\mathbf{K}^{-1}\mathbf{y} \tag{C.3}$$

$$\nabla\sigma^2(\mathbf{x}) = -2\nabla\mathbf{k}^T(\mathbf{x})\mathbf{K}^{-1}\mathbf{k}(\mathbf{x}). \tag{C.4}$$

Note that in the last step we assume a stationary kernel k which means that $k(\mathbf{x},\mathbf{x})$ is some constant.

C.1 Probability of improvement

The fist acquisition function of interest is *probability of improvement* (PI), which, for a GP model, can be analytically computed yielding

$$\alpha^{\mathrm{PI}}(\mathbf{x}) := \Phi(z(\mathbf{x})) \tag{C.5}$$

where

$$z(\mathbf{x}) := \frac{\mu(\mathbf{x}) - y^+}{\sigma(\mathbf{x})}, \tag{C.6}$$

where Φ is the standard normal CDF and $y^+ = \min_i y_i$. This expression can be differentiated as follows

$$\nabla\alpha^{\mathrm{PI}}(\mathbf{x}) = \phi(z(\mathbf{x}))\nabla z(\mathbf{x}) \tag{C.7}$$

where *phi* is the standard normal pdf and

$$\nabla z(\mathbf{x}) = \nabla\left[\frac{\mu(\mathbf{x}) - y^+}{\sigma(\mathbf{x})}\right] \tag{C.8}$$

$$= \frac{\nabla\mu(\mathbf{x})\sigma(\mathbf{x}) - (\mu(\mathbf{x}) - y^+)\nabla\sigma(\mathbf{x})}{\sigma^2(\mathbf{x})} \tag{C.9}$$

$$= \frac{\nabla\mu(\mathbf{x})}{\sigma(\mathbf{x})} - \frac{1}{2}z(\mathbf{x})\frac{\nabla\sigma^2(\mathbf{x})}{\sigma^2(\mathbf{x})} \tag{C.10}$$

where in the last step we used the fact that $\nabla\sigma(\mathbf{x}) = \frac{1}{2}\frac{\nabla\sigma^2(\mathbf{x})}{\sigma(\mathbf{x})}$ in order to express the gradient in terms of the gradient of the variance, which is easier to compute. Substituting (C.10) into (C.7) we get:

$$\nabla\alpha^{\mathrm{PI}}(\mathbf{x}) = \phi(z(\mathbf{x}))\left[\frac{\nabla\mu(\mathbf{x})}{\sigma(\mathbf{x})} - \frac{1}{2}z(\mathbf{x})\frac{\nabla\sigma^2(\mathbf{x})}{\sigma^2(\mathbf{x})}\right] \tag{C.11}$$

C.2 Expected improvement

A similar acquisition function is *expected improvement* (EI); as with PI, for a GP model, EI can be analytically computed and reduces to the following simple expression:

$$\alpha^{\mathrm{EI}}(\mathbf{x}) := \sigma(\mathbf{x}) \left[z(\mathbf{x})\Phi(z(\mathbf{x})) + \phi(z(\mathbf{x})) \right], \tag{C.12}$$

with $z(x)$ defined as in (C.6). In the following, it will help to denote $A(z) = z\Phi(z) + \phi(z)$ and its first derivative $A'(z) = \Phi(z) + z\phi(z) + \phi'(z) = \Phi(z)$, where in the last step we use the property of the standard normal pdf $\phi'(z) = z\phi(z)$. Note that with this new quantity $A(z)$, the expected improvement can be written $\alpha^{\mathrm{EI}}(x) = \sigma(x)A(z(x))$. Let us now compute the gradient of this expression.

$$\nabla\alpha^{\mathrm{EI}}(\mathbf{x}) = A(z(\mathbf{x}))\nabla\sigma(\mathbf{x}) + \sigma(\mathbf{x})A'(z(\mathbf{x}))\nabla z(\mathbf{x}) \tag{C.13}$$

$$= \frac{1}{2}\frac{A(z(x))}{\sigma(x)}\nabla\sigma^2(x) + \sigma(x)\Phi(z(x)) \left[\frac{\nabla\mu(x)}{\sigma(x)} - \frac{1}{2}z(x)\frac{\nabla\sigma^2(x)}{\sigma^2(x)} \right] \tag{C.14}$$

$$= \frac{1}{2}\alpha^{\mathrm{EI}}(\mathbf{x})\frac{\nabla\sigma^2(\mathbf{x})}{\sigma^2(\mathbf{x})} + \Phi(z(\mathbf{x}))\nabla\mu(\mathbf{x}) - \frac{1}{2}z(\mathbf{x})\Phi(z(\mathbf{x}))\frac{\nabla\sigma^2(\mathbf{x})}{\sigma(\mathbf{x})} \tag{C.15}$$

$$\nabla\alpha^{\mathrm{EI}}(\mathbf{x}) = \frac{1}{2}\frac{\nabla\sigma^2(\mathbf{x})}{\sigma^2(\mathbf{x})} \left[\alpha^{\mathrm{EI}}(\mathbf{x}) - \frac{1}{2}z(\mathbf{x})\Phi(z(\mathbf{x}))\sigma(\mathbf{x}) \right] + \Phi(z(\mathbf{x}))\nabla\mu(\mathbf{x}) \tag{C.16}$$

where we have written the gradient of EI entirely in terms of the posterior quantities, μ and σ^2, and their derivatives.

C.3 Gaussian process upper confidence bound

Finally, recall the definition of GP-UCB:

$$\alpha^{\mathrm{UCB}}(\mathbf{x}) := \mu(\mathbf{x}) + \beta\sigma^2(\mathbf{x}), \tag{C.17}$$

which can be differentiated

$$\nabla \alpha^{\mathrm{UCB}}(\mathbf{x}) = \nabla \mu(\mathbf{x}) + \beta \nabla \sigma^2(\mathbf{x}). \tag{C.18}$$

Appendix D

Approximating Gaussian process inference

For large datasets, or large function evaluation budgets in the Bayesian optimization setting, the cubic cost of exact inference is prohibitive and there have been many attempts at reducing this computational burden via approximation techniques. In this chapter, we review two sparsification techniques for Gaussian processes and the alternative random forest regression.

D.1 Sparse pseudo-inputs

One early approach considered using $m < t$ *inducing pseudo-inputs* to reduce the rank of the covariance matrix to m, resulting in a significant reduction in computational cost (Seeger et al., 2003; Snelson and Ghahramani, 2005). By forcing the interaction between the t data points $\mathbf{x}_{1:t}$ and any test point $\mathbf{x}$ to go through the set of m inducing pseudo-inputs, these methods can compute an approximate posterior in $O(tm^2 + m^3)$ time. Pseudo-input methods have since been unified in a single theory based on the following overarching approximation:

$$p(f(\mathbf{x}), \mathbf{f}) = \int p(f(\mathbf{x}), \mathbf{f}, \mathbf{u}) \mathrm{d}\mathbf{u}$$

$$\approx \int q(f(\mathbf{x}) \mid \mathbf{u}) q(\mathbf{f} \mid \mathbf{u}) p(\mathbf{u}) \, \mathrm{d}\mathbf{u} = q(\mathbf{f}, \mathbf{f}_*) \tag{D.1}$$

where $\mathbf{u}$ is the vector of function values at the pseudo-inputs. All the sparse GP approximations based on pseudo-inputs can be formulated in terms of the form used for the training and test conditionals, $q(f(\mathbf{x})|\mathbf{u})$ and $q(\mathbf{f}|\mathbf{u})$, respectively (Quiñonero-Candela and Rasmussen, 2005).

In the seminal works on pseudo-input methods, the locations of the pseudo-inputs were selected to optimize the marginal likelihood of the SPGP (Seeger et al., 2003; Snelson and Ghahramani, 2005). In contrast, a variational approach has since been proposed to marginalize the pseudo-inputs to maximize fidelity to the original exact GP (Titsias, 2009) rather than the likelihood of the approximate GP.

The computational savings in the pseudo-input approach to approximating the GP comes at the cost of poor variance estimates. As can be observed in Figure D.1, the uncertainty (blue shaded area) exhibits unwanted pinching at pseudo-inputs, while it is overly conservative in between and away from pseudo-inputs. In this instance, the 10 inducing points, indicated with black crosses, were not optimized to emphasize the potential pathologies of the method. Since in Bayesian optimization we use the credible intervals to guide exploration, these artefacts can mislead our search.

D.2 Sparse spectrum

While inducing pseudo-inputs reduce computational complexity by using a fixed number of points in the search space, sparse spectrum gaussian processes (SSGP) take a similar approach to the kernel's spectral space (Lázaro-Gredilla et al., 2010). Bochner's theorem states that any stationary kernel $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x} - \mathbf{x}')$ has a positive and finite Fourier spectrum $s(\mathbf{w})$, *i.e.*

$$k(\mathbf{x}) = \frac{1}{(2\pi)^d} \int e^{-i\mathbf{w}\cdot\mathbf{x}} s(\mathbf{w}) \, \mathrm{d}\mathbf{w}. \tag{D.2}$$

Since the spectrum is positive and bounded, it can be normalized such that $p(\mathbf{w}) := s(\mathbf{w})/\nu$ is a valid probability density function. In this formulation,

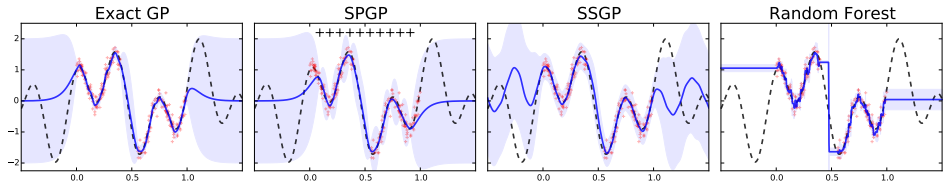


Figure D.1: Comparison of surrogate regression models

evaluating the stationary kernel is equivalent to computing the expectation of the Fourier basis with respect to its specific spectral density $p(\mathbf{w})$ as in the following

$$k(\mathbf{x}, \mathbf{x}') = \nu \, \mathbb{E}_{p(\mathbf{w})}[e^{-i\mathbf{w}\cdot(\mathbf{x}-\mathbf{x}')}]. \tag{D.3}$$

As the name suggests, SSGP approximates this expectation via Monte Carlo estimation using m samples drawn from the spectral density so that

$$k(\mathbf{x}, \mathbf{x}') \approx \frac{\nu}{m} \sum_{i=1}^{m} e^{-i\mathbf{w}^{(i)}\cdot\mathbf{x}} e^{i\mathbf{w}^{(i)}\cdot\mathbf{x}'} \tag{D.4}$$

where $\mathbf{w}^{(i)} \sim s(\mathbf{w})/\nu$. The resulting finite dimensional problem is equivalent to Bayesian linear regression with m basis functions and the computational cost is once again reduced to $O(tm^2 + m^3)$.

As with the pseudo-inputs, the spectral points can also be tuned via marginal likelihood optimization. Although this introduces a risk of overfitting, it allows for a smaller number of basis functions with good predictive power (Lázaro-Gredilla et al., 2010). Once again, in Figure D.1 we have not tuned the 80 spectral points in this way. Whereas around observed data (red crosses) the uncertainty estimates are smoother than the pseudo-inputs method, away from observations both the prediction and uncertainty regions exhibit spurious oscillations. This is highly undesirable for Bayesian optimization where we expect our surrogate model to fall back on the prior away from observed data.

D.3 Random forest surrogates

Finally, as an alternative to Gaussian processes, random forest regression has been proposed in the context of sequential model-based algorithm configuration (SMAC) (Hutter et al., 2011). Introduced in 2001 (Breiman, 2001), random forests are a class of scalable and highly parallelizable regression models that have been very successful in practice (Criminisi et al., 2011). More precisely, the random forest is an ensemble method where the weak learners are decision trees trained on random subsamples of the data (Breiman, 2001). Averaging the predictions of the individual trees produces an expressive response surfaces.

The random forest regression model gives SMAC the flexibility to handle categorical input features and to scale to large observational datasets. On the other hand, the exploration strategy in SMAC still requires an uncertainty estimate for predictions at test points. Since the random forest does not readily provide an estimate of the variance of its predictions, Hutter et al. proposed using the empirical variance in the predictions across trees in the ensemble (Hutter et al., 2011).

Although random forests are good interpolators in the sense that they output good predictions in the neighbourhood of training data, they are very poor extrapolators. Indeed, far from the data, the predictions of all trees could be identical, resulting in a poor prediction; more importantly, using the variance estimate of SMAC results in extremely confident intervals. In Figure D.1 for example, away from data the shaded area is very narrow around a very poor constant prediction. Even more troubling is the fact that in areas of missing data multiple conflicting predictions can cause the empirical variance to blow up sharply, as can be seen in Figure D.1. While Gaussian processes are also poor extrapolators (when used with local kernels), they at least produce highly uncertain predictions away from the data: a much more desirable behavior when trading off exploration and exploitation.

Finally, another drawback of random forests for Bayesian optimization is that the response surface is discontinuous and non-differentiable so gradient

based optimization methods are not applicable and SMAC must rely on a combination of local and random search when maximizing the acquisition function.

Appendix E

Hyperparameter learning

In Chapter 2, we mostly ignored the GP mean and kernel hyperparameters and assumed they were known or given. Consider the generic function $g : \mathcal{X} \times \Theta \mapsto \mathbb{R}$, where $\theta \in \Theta$ represents the hyperparameters of our GP. In the context of Bayesian optimization, this function could be our objective function or any function derived from the Gaussian process, but for concreteness, it may help to think of it specifically as the acquisition function α. We wish to marginalize out our uncertainty about θ with the following expression

$$g_t(x) := \mathbb{E}_\theta \left[g(x;\theta) | \mathcal{D}_t \right] = \int g(x;\theta) p(\mathrm{d}\theta | \mathcal{D}_t) \tag{E.1}$$

This integral is over our posterior belief over θ given observations $\mathcal{D}_t$, which can be decomposed via Bayes rule as

$$p(\theta|\mathcal{D}_t) = \frac{p(\mathbf{y}|\mathbf{x}_{1:t}, \theta) p(\theta)}{p(\mathcal{D}_t)} =: \frac{\gamma_t(\theta)}{p(\mathcal{D}_t)} \tag{E.2}$$

where we have defined γ_t as the unnormalized posterior. In this section, we review several methods used in practice to approximate the integral (E.1). We first consider simple point estimates, followed by the Markov chain Monte Carlo and sequential Monte Carlo methods, and finally, we present variational methods.

E.1 Point estimates

The simplest approach to tackling (E.1) is to fit the hyperparameter to observed data using a point estimate $\hat{\theta}_t^{\mathrm{ML}}$ or $\hat{\theta}_t^{\mathrm{MAP}}$, corresponding to type II maximum likelihood or maximum *a posteriori* estimates, respectively. The posterior is then replaced by a delta measure at the corresponding $\hat{\theta}_t$ which yields

$$\hat{g}_t(x) = g(x; \hat{\theta}_t). \tag{E.3}$$

The estimators $\hat{\theta}_t^{\mathrm{ML}}$ and $\hat{\theta}_t^{\mathrm{MAP}}$ can be obtained by optimizing the marginal likelihood or the unnormalized posterior, respectively. For certain priors and likelihoods, these quantities as well as their gradients can be computed analytically. For example, the GP regression model yields the following marginal likelihood:

$$\mathcal{L}_t(\theta) := \log p(\mathbf{y}|\mathbf{x}_{1:t}, \theta) \tag{E.4}$$

$$= -\frac{1}{2}(\mathbf{y} - \mu_0)^\mathsf{T}(\mathbf{K}_t^\theta + \sigma^2\mathbf{I})^{-1}(\mathbf{y} - \mu_0) - \frac{1}{2}\log|\mathbf{K}_t^\theta + \sigma^2\mathbf{I}| - \frac{t}{2}\log(2\pi). \tag{E.5}$$

Therefore it is common to use multi-started local optimizers such as the limiter-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method on objectives $\mathcal{L}_t$ or γ_t.

In Bayesian optimization, our uncertainty about the response surface plays a key role in guiding exploration and therefore it is important to incorporate our uncertainty about θ in the regression model. Naturally, these point estimates cannot capture this uncertainty and the following sections present common techniques for marginalizing out the hyperparameters in more principled ways.

E.2 Markov chain Monte Carlo

The common component in Monte Carlo (MC) methods is that they approximate the integral in (E.1) using N samples $\{\theta_t^{(i)}\}_{i=1}^N$ from the posterior

distribution $p(\mathrm{d}\theta|\mathcal{D}_t)$:

$$\mathbb{E}\left[g(x;\theta)|\mathcal{D}_t\right] \approx \frac{1}{N}\sum_{i=1}^{N} g(x;\theta_t^{(i)}). \tag{E.6}$$

However, in practice it is impossible to sample directly from the posterior so we have to use sampling techniques to obtain samples that are marginally distributed according to $p(\mathrm{d}\theta|\mathcal{D}_t)$. In the Markov chain Monte Carlo (MCMC) approach, a Markov chain of samples is generated where the samples are marginally distributed according to the target posterior in the limit of infinitely long chains.

In this section we introduce the *slice sampler* which is a particular MCMC algorithm used in practice for hyperparameter marginalization in Bayesian optimization (Murray and Adams, 2010; Snoek et al., 2012). Note that other MCMC methods could also be used; in particular, since the analytic gradient of the marginal likelihood can be computed, Hamiltonian Monte Carlo (HMC (Duane et al., 1987)) could be employed, but this method can be hard to tune in practice.

Consider an auxiliary random variable $u \in$ and the following joint probability distribution with density $\pi(\theta, u) = \mathbb{I}\big[0 \leq u \leq \pi(\theta)\big]$ such that

$$\int \pi(\theta, u)\mathrm{d}u = \int_0^{\pi(\theta)} \mathrm{d}u = \pi(\theta). \tag{E.7}$$

Therefore we can sample from $\pi(\mathrm{d}\theta)$ by sampling from this joint distribution and ignoring the auxiliary variable u. One way to create a Markov chain over this augmented space is by defining the following transition kernel $\kappa(\theta, u, \mathrm{d}\theta', \mathrm{d}u') = p(\mathrm{d}\theta'|u')p(\mathrm{d}u'|\theta)$, where the conditionals are defined as

$$p(\mathrm{d}u'|\theta) = \mathcal{U}_{[0,\pi(\theta)]}(\mathrm{d}u') \tag{E.8}$$

$$p(\mathrm{d}\theta'|u') = \mathcal{U}_{\{\theta':\pi(\theta')\geq u'\}}(\mathrm{d}\theta'). \tag{E.9}$$

Because this algorithm only consists of comparisons of probability densities,

we can replace the above $\pi(\theta)$ with the unnormalized $\gamma(\theta)$. Therefore, we first compute $\gamma(\theta)$ and draw a uniform random variable $u' \in [0, \gamma(\theta)]$; we then sample uniformly within the *slice* $\{\theta' : \gamma(\theta') \geq u'\}$. Even in one dimension it is difficult to find this set so in practice a width parameter w is chosen and a candidate window of width w randomly centred around the current sample θ is initialized. The endpoints θ_L and θ_U of the slice are tested to see whether $u' \geq \gamma(\theta_{L/U})$ holds, *i.e.* whether either endpoint is outside of the slice. The bounds are iteratively grown by taking steps of width w until the test holds.

Once the bounds $\theta_{L/U}$ include the slice[1], a proposal $\theta' \in [\theta_L, \theta_U]$ is uniformly sampled. If $\gamma(\theta') > u$ the proposal is accepted (and the Markov chain carries on from here), if the proposal is rejected, the bounds $\theta_{L/U}$ are updated according to whether $\theta' < \theta$ or $\theta' > \theta$, respectively. These steps are described more precisely in Algorithm 5, where we consider the slice sampler without the stepping out subroutine.

In multiple dimensions one can either sequentially sample each dimension independently, or draw a random direction and slice sample along it. Another common strategy is to partition the dimensions into blocks and sequentially slice sample along a random direction in the subspaces corresponding to each of the blocks. The latter is especially recommended when certain dimensions, *i.e.* hyperparameters are highly correlated.

[1] Actually, since the slice is not necessarily a simply connected set, the bounds are only guaranteed to include part of the slice. See (Neal, 2003) for a more rigorous proof of correctness of the slice sampling algorithm.

Algorithm 5 Slice sampling

Require: γ, the unnormalized posterior;
θ, the current sample of the Markov chain;
w, a width parameter.

Sample $u \sim \mathcal{U}[0, \gamma(\theta)]$
Sample $v \sim \mathcal{U}[0, w]$
Set $\theta_L = \theta - v$ and $\theta_U = \theta + w$
while $\theta_L < \theta_U$ **do**
  Sample $\theta' \sim \mathcal{U}[\theta_L, \theta_U]$
  if $\gamma(\theta') \geq u$ **then**
    return θ'
  else
    if $\theta' < \theta$ **then**
      Set $\theta_L = \theta'$
    else
      Set $\theta_U = \theta'$
    end if
  end if
end while
return θ'

Naturally, in practice this method is used far from the asymptotic limit and therefore the samples can be highly correlated. Furthermore, multimodal posteriors over θ can be hard to approximate due to poor mixing or short chains. The following section reviews a Monte Carlo method that addresses the mixing issue when the target distribution is in fact a slowly varying distribution, such as our posterior over θ.

E.3 Sequential Monte Carlo

In this section, the approach we take is a generalization of particle filters due to (Liu and Chen, 1998) called sequential Monte Carlo (SMC) and has been implemented for hyperparameter marginalization by (Gramacy and Polson,

2011). SMC is especially useful when one wants to compute the expectation of a sequence of functions $\{g_t\}$ with respect to a slowly varying sequence of distributions π_t. In our case, this sequence of distributions consists of the posterior distribution of θ after every observation.

The idea of SMC is to maintain a set of N particles and corresponding weights $\left\{(\theta_t^{(i)}, w_t^{(i)})\right\}$ with target distribution $p(\mathrm{d}\theta|\mathcal{D}_t)$. At time $t+1$, we evolve the particles following some transition kernel $K_t(\theta, \mathrm{d}\theta)$ and update the weights accordingly. Then the SMC approximation to (E.1) differs from (E.6) slightly and can be written as follows:

$$\mathbb{E}\left[g(x;\theta)|\mathcal{D}_t\right] \approx \sum_{i=1}^{N} g(x;\theta_t^{(i)})\hat{w}_t^{(i)}, \tag{E.10}$$

where $\hat{w}_t^{(i)}$ are the normalized weights

$$\hat{w}_t^{(i)} := \frac{w_t^{(i)}}{\sum_{j=1}^{N} w_t^{(j)}}. \tag{E.11}$$

Consider the sequence of target distributions $\pi_t(\mathrm{d}\theta) := p(\mathrm{d}\theta|\mathcal{D}_t)$ where, with a slight abuse of notation, wherever we use θ as an argument to π_t we are referring to its associated density with respect to some base measure—we use this convention throughout the paper. While we cannot compute this quantity pointwise, we can compute the unnormalized measure by applying Bayes rule

$$\pi_t(\theta) = p(\theta|\mathcal{D}_t) = \frac{p(\mathcal{D}_t|\theta)p(\theta)}{p(\mathcal{D}_t)} = \frac{\gamma_t(\theta)}{p(\mathcal{D}_t)}. \tag{E.12}$$

Note that the unnormalized posterior $\gamma_t(\theta)$ is the product of the marginal likelihood and the prior which can be computed pointwise.

The weights are sequentially updated according to

$$w_t^{(i)} = w_{t-1}^{(i)}\tilde{w}_t(\theta_{t-1}^{(i)}, \theta_t^{(i)}). \tag{E.13}$$

The incremental weight $\tilde{w}_t(\cdot,\cdot)$ depends on the transition kernel K_t that is used. Equation 31 from (Del Moral et al., 2006) states that if K_t is

an MCMC kernel with stationary distribution $\pi_t(\theta)$ then a good – though suboptimal – incremental weight to use

$$\tilde{w}_t(\theta, \theta') = \frac{\gamma_t(\theta)}{\gamma_{t-1}(\theta)} \tag{E.14}$$

which yields

$$\tilde{w}_t(\theta, \theta') = \frac{p(\mathcal{D}_t|\theta)p(\theta)}{p(\mathcal{D}_{t-1}|\theta)p(\theta)} = \frac{p(\mathcal{D}_t|\theta)}{p(\mathcal{D}_{t-1}|\theta)} = \frac{p(y_t|\mathbf{x}_t, \mathcal{D}_{t-1}, \theta)p(\mathcal{D}_{t-1}|\theta)}{p(\mathcal{D}_{t-1}|\theta)} \tag{E.15}$$

$$= p(y_t|\mathbf{x}_t, \mathcal{D}_{t-1}, \theta). \tag{E.16}$$

Since the weight of the particle $\theta' = \theta_t^{(i)}$ only depends on its parent $\theta_{t-1}^{(i)}$, it is possible to weigh and resample particles before propagating them; thus recover the particle learning approach of (Gramacy and Polson, 2011). This is sometimes referred to as the look-ahead trick because it allows us to observe the next data point before resampling the current set of particles.

Note that in the previous section we detailed a MCMC kernel with stationary distribution $\pi_t(\theta)$ and we can use the same slice sampler to propose new particles. This way, the SMC approach can be interpreted as evolving multiple MCMC chains in parallel with interactions at resampling steps.