University of California
Santa Barbara

Generalized Probabilistic Bisection for Stochastic Root-Finding

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Statistics and Applied Probability

by

Sergio Rodríguez Hernández

June 2018

The Dissertation of Sergio Rodríguez Hernández is approved.

Professor John Hsu

Professor Sang-Yun Oh

Professor Michael Ludkovski, Committee Chair

March 2018

Generalized Probabilistic Bisection for Stochastic Root-Finding

Everything I wrote is dedicated to my mom, Ma. Estela Paulina Hernández González.

Acknowledgements

First and foremost, I would like to thank my advisor Prof. Michael Ludkovski for his guidance and constant support throughout my time at the University of California, Santa Barbara (UCSB). He has been an amazing mentor and provided me with all the tools in order for me to become a researcher and better professional, as well as a great example of kindness, patience and humility. As a PhD student I couldn't have asked for a better mentor than him.

I am also very grateful to Prof. John Hsu and Prof. Sang Oh for serving as members of my doctoral committee. Furthermore, I would like to thank the entire faculty at the Statistics and Applied Probability Department at UCSB. In particular, Prof. Jean-Pierre Fouque, who encouraged me to become a graduate student at UCSB when I was an exchange student during the last year of my undergraduate studies back in 2009, and Prof. Wendy Meiring, who recommended me to join the PhD program.

Looking back to my undergraduate studies at Universidad Nacional Autónoma de México, I would like to acknowledge my former mentor, Dr. Jorge De la Vega Góngora, who I met during my internship at the Central Bank of Mexico, as well as Prof. Ruth Fuentes García, who were an inspiration to become a statistician.

Special thanks to Victor Fragoso and Saiph Savage, who became my first friends when I arrived to Santa Barbara, CA. They encouraged me to publish my first conference paper back in 2013, and are great examples of perseverance, tenacity, and generosity.

I am very thankful to all the amazing friends I had the chance to meet during my time in Santa Barbara. They helped me to overcome so many difficulties and have always been supportive and considered to me. They made my graduate studies one of the best times of my life. In particular, I want to thank Sahar Sajadieh, Rachel Engelskirger, Priyam Patel, Tobi Olofintuyi, and John and Sandra Jameson. I am also grateful to

Megan Elcheikhali and Brian Wainwright for their time and kindness to proofread the contents of this thesis.

I would also like to express my gratitude to Laura Ballesteros, who was an integral part of my PhD adventure. She allowed me to become a better human by being an example of love, kindness and generosity.

Finally, I would like to thank my family. My mother, Ma. Estela Paulina Hernández González, who raised four children by herself, and whose love has always been unconditional. Without her sacrifice and hard work I would have never come this far in my life. I love you with all my heart, *mamita querida*. Also, my three sisters: Suhail, Amaranta and Geraldina with whom I grew up in the most joyful and loving environment. They have been integral part of my development as a human and a fundamental component of who I am as a person.

Lastly, I would like to thank the National Council of Science and Technology from the Mexican Federal Government (CONACyT), as well as the University of California Institute for Mexico and the United States (UCMEXUS) for their financial support under grant CONACYT-216011.

Curriculum Vitæ
Sergio Rodríguez Hernández

Education

2018	Ph.D. Statistics and Applied Probability, UC Santa Barbara Supervisor: Professor Michael Ludkovski
2018	M.S. Electrical and Computer Engineering, UC Santa Barbara Emphasis in Control Systems and Signal Processing
2014	M.A. Statistics, UC Santa Barbara Emphasis in Mathematical Statistics
2009	B.S. Actuarial Science, Universidad Nacional Autónoma de México Exam P and FM from the Society of Actuaries, USA.

Professional Experience

2017	Data Science Intern, Amazon, Seattle, WA.
2016/2015	Data Science Intern, AOL, Santa Monica, CA.
2012	Credit Risk Analyst, HSBC Holdings, Mexico City, Mexico
2010	Statistical Consultant, National Institute of Neurology and Neurosurgery, Mexico City, Mexico.
2009	Data Science Intern, Central Bank of Mexico, Mexico City, Mexico.

Publications and Preprints

1. S. Rodriguez and M. Ludkovski, *Generalized probabilistic bisection for stochastic root-finding, arXiv preprint arXiv:1711.00843* (2017)
2. S. Rodriguez, *Blending spatial modeling and probabilistic bisection*, in Proceedings of the 2016 Winter Simulation Conference, pp. 3664-3665, IEEE Press, 2016
3. S. Rodriguez and M. Ludkovski, *Information directed sampling for stochastic root finding*, in Winter Simulation Conference (WSC), 2015, pp. 3142-3143, IEEE, 2015
4. V. Fragoso, P. Sen, S. Rodriguez, and M. Turk, *EVSAC: accelerating hypotheses generation by modeling matching scores with extreme value theory*, in Computer Vision (ICCV), 2013 IEEE International Conference on, pp. 2472-2479, IEEE, 2013

Abstract

Generalized Probabilistic Bisection for Stochastic Root-Finding

by

Sergio Rodríguez Hernández

This thesis studies the stochastic root-finding problem, which consists of estimating the point x^* that solves the equation $h(x^*) = 0$, where the function $h : (0, 1) \to \mathbb{R}$ is learned via a stochastic simulator (oracle). Instead of focusing on modeling $h(\cdot)$, we develop statistical methodologies that directly infer x^* following a fully Bayesian approach. To do so, we investigate procedures that generalize the Probabilistic Bisection Algorithm (PBA) first introduced in Horstein (1963). The PBA is a one-dimensional stochastic root-finding routine which builds an explicit Bayesian representation (i.e., a posterior density) for x^* based on the history of noisy function evaluations and sampling locations. The PBA starts by assuming that x^* is the realized value of an absolutely continuous random variable, $X^* \sim g_0$, with prior density g_0. Then, it recursively updates a posterior, g_n, leveraging the information provided by the signs (positive/negative) of the noisy function evaluations — which inform the direction where x^* is located with respect to a given location, x—. Due to observational noise, the oracle responses are correct only with probability $p(x)$. Waeber et al. (2013) showed that sampling at the median of g_n is an optimal sampling strategy and established exponential convergence of the posterior g_n to a Dirac mass at the true x^* under the very restrictive assumption that the probability of correct response $p(x)$ is known and constant for all x; however, in the most general and practical settings the latter condition no longer holds and the only way to implement the PBA is to *estimate* $p(\cdot)$.

In the first part of this thesis, we state the Generalized PBA (G-PBA), where the above assumption is relaxed to the case where the sampling distribution of the oracle is *unknown* and *location-dependent*. Namely, as in standard PBA, we rely on a knowledge state to approximate the posterior of the root location. To implement the corresponding Bayesian updating, we also carry out *inference* of $p(\cdot)$. To this end we utilize *batched querying* in combination with a variety of frequentist and Bayesian estimators based on majority vote, as well as the underlying functional responses, if available. For guiding sampling selection we propose two families of sampling policies: batched Information Directed Sampling and Randomized Quantile Sampling, which is a reminiscent of Thompson Sampling and a generalization of the median sampling as in classical PBA. The latter leads to the first main conclusion: the G-PBA is able to efficiently learn $p(\cdot)$ and X^* simultaneously.

In the second part of this thesis, we propose to leverage the *spatial* structure of a typical oracle by constructing a non-parametric statistical surrogate for $p(\cdot)$ based on binomial regression. The latter leads to the second main conclusion: surrogate modeling allows to determine the batch size for querying the oracle *adaptively* as a function of the estimated predictive uncertainty of $p(\cdot)$.

In the last part of this thesis, we present extensive numerical experiments in order to evaluate our sampling strategies (information-based or randomized). In particular we demonstrate the efficiency of randomized quantile sampling for balancing the exploration/exploitation component; moreover, we show that spatial surrogate modeling results in significant gains relative to the local estimators, as quantified by the improved quality of the resulting root estimates (namely lower absolute residuals, narrower credible intervals and dramatically higher probability coverage). Our work is motivated by the root-finding sub-routine in pricing of Bermudan financial derivatives, illustrated in the last section of this thesis.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Motivation

This thesis studies the problem of estimating the root of a function when the function is expressed implicitly through a stochastic simulation known as *stochastic root-finding problem* (SRFP).

Our interest in studying SRFPs is motivated by the root-finding sub-routine for pricing a Bermudan Put option [39]. Namely, a stochastic simulation approach (known in the literature as the Longstaff-Schwartz paradigm [38]) recursively builds noisy simulations of the *timing-value* function; defined as the difference between the *value function* and the *reward function* for any given initial price x and fixed time $t \geq 0$ [6]. This is obtained by generating forward paths of the state process and computing corresponding path-wise random reward. The realization, $Z(x)$, is the path-wise timing value $h(x)$, that is, the difference between future and immediate reward over the given trajectory, which is furthermore *unbiased* for the true $h(x)$. It is well-known that there is a unique *exercise boundary*, say x^*, so that the timing-value is zero, i.e., $h(x^*) = 0$ and one should exercise as soon as x drops below x^*, since the conditional expectation of tomorrow's reward-to-go is less than the immediate reward (frequently, a priori structure implies that the *stopping set* $(0, x^*)$ is a half-line, i.e., $h(\cdot)$ has a unique root x^*). However, since the timing-value function $h(\cdot)$ is learned with noise that arises intrinsically due to the randomness in the simulation, finding x^* effectively reduces to a SRFP.

Stochastic root-finding indeed ubiquitously in a wide range of applications. Some relevant applications of SRFPs include, for instance, the quantile estimation problem in Bio-assay experiments [18]; quality and reliability improvement [33]; sensitivity experiments [41]; and adaptive control and signal processing [12, 24, 5].

Data generating mechanism and experimental design. Throughout this thesis we work with a *black-box* view of a simulation model, where the inputs and outputs of a simulation model are observed, but the internal variables and specific functions implied by the simulation are not. In particular, we assume a data generating process of the form

$$Z(x) = h(x) + \epsilon(x), \quad x \in (0, 1); \tag{1.1}$$

where $Z(x)$ is the simulation output, $h(x)$ is a real-valued function and $\epsilon(x)$ is an additive noise component whose distribution depends upon the *sampling location* x (simulation input) but it is independent of previous evaluations. In addition, we assume that the search space is bounded, so we can scale the search region to be the interval $(0, 1)$.

We remark that the nature of the data generating mechanism (1.1) differs from the classical regression/machine-learning paradigm, where it is assumed that a data set $(x_i, Z_i(x_i))_{i=1}^N$ is available before inference about unknown parameters is performed. In general, we are interested in settings where simulation evaluation is expensive and data collection is restricted to a limited a sampling budget N and hence *experimental design* (ED) becomes critical.

The goal of ED is to *efficiently* learn the root x^* of the function h, interpreted as optimizing the simulation budget of calling (1.1) by judiciously selecting the points $x_{1:n} :=$

$(x_1, \dots, x_n)$ at which to observe $Z_{1:n} := (Z_1(x_1), \dots, Z_n(x_n))$, and in turn construct an estimator $\hat{x}_n$ whose performance[1] improves as more information is available. The latter problem falls under the rubric of Bayesian design of experiments [11], in general; and design and analysis of simulation experiments (DASE) [36], in particular. Importantly, notice the distinction between sampling locations x_n and the estimators $\hat{x}_n$, which in general need not to be the same (in contrast to deterministic root-finding algorithms where x_n coincides with the estimated root $\hat{x}_n$).

Roughly speaking, there exist two approaches to conduct an experiment, a *sequential* (adaptive) design or a passive (non-adaptive) design. In a passive design the querying locations $x_{1:n}$ are chosen prior to the experiment, whereas in a sequential design new sampling points x_{n+1} are selected based on the previous $x_{1:n}$ and $Z_{1:n}$. As pointed out in [32], the optimal election of, x_{n+1}, depends intimately on the distributional properties of the simulation outputs $Z(\cdot)$ and, naturally, no information is available before the actual experiment is conducted. Thus, the root estimates induced by a passive design (e.g., sampling uniformly over the input space) may exhibit poor optimality properties, whereas a sequential approach may be a more suitable choice to learn the root. In fact, as we show in our numeric examples presented in Chapter 4, sequential strategies outperform their non-adaptive counterpart as measured by their corresponding accuracy and uncertainty reduction about the root location.

The primary focus of the work presented below is thus to develop statistical procedures to infer the point x^* that solves $h(x^*) = 0$, by *efficiently* selecting the locations $x_{1:n}$ at which to observe simulation outputs $Z_{1:n}$ and in turn produce a high-fidelity point estimator $\hat{x}_n$ for the unknown root location x^*. To that end, we assume the existence and uniqueness of the root x^* on $(0, 1)$ so, without loss of generality, we furthermore

[1]For example, in our numeric examples presented in Chapter 4 we use several performance metrics to measure the quality of the root estimates: absolute residuals, length of confidence intervals and coverage.

suppose that the function $h(\cdot)$ in the *metamodel* (1.1) is *positive* to the left of x^* and *negative* for all $x > x^*$ (e.g., is non-increasing on $(0,1)$).

Overview of Stochastic Root-Finding Methods. Numeric schemes for solving SRFPs can be classified broadly into two main groups: stochastic approximation and sample-path methods [43].

The stochastic approximation (SA) algorithm was introduced in the seminal paper of Robbins and Monro [50]. The SA paradigm closely resembles the Newton-Raphson deterministic root-finding regime for non-linear root-finding: start at a initial point x_1 close from x^* is known to be located (usually the region at which the underlying function h is monotonic is known a priori), and then evaluate (1.1) selecting new points using the rule

$$x_{n+1} := x_n - b_n Z_n,$$

for all $n \in \mathbb{N}$ until convergence criteria are satisfied; where $(b_n)_{n\geq 1}$ is a deterministic sequence of constants. The SA strategy is in fact fully asymptotically efficient, i.e., $x_n \to x^*$ in probability as $n \to +\infty$ under regularity conditions on the tunning sequence (b_n) [32].

As mentioned in Waeber [57], one of the main drawbacks of the SA-type methods is that they only provide a point estimate $\hat{x}_n$ for x^* (which under the SA setting is equal to the sampling location x_n) without specifying any further probabilistic guarantees on the accuracy of this estimate, for example, a confidence interval for the true x^*, which is one of the main tools necessary to determine a stopping rule when sampling budget is small.

Another approach that implements the actual functional evaluations (1.1) consists of sample-path (SP) schemes [25, 54]. As mentioned in [43], the SP method is conceptually very simple and intuitive: substitute the *unknown* function h in (1.1) by a *deterministic* function obtained by observing "realizations" of an unbiased estimator Z_m for h

(for example, the average function evaluations $\bar{Z}_m(\cdot) := m^{-1}\sum_{j=1}^{m} Z_j(\cdot)$), and then solve the resulting problem as a deterministic root-finding problem (DRFP). In this sense, a practical and viable strategy is thus to learn $h(\cdot)$ directly by regressing the batched responses $\bar{Z}_{1:n} := (\bar{Z}_m(x_1), \dots, \bar{Z}_m(x_n))$ on the history of sampling locations $x_{1:n}$, i.e., build a surrogate $\hat{h}$ and then take $\hat{x} = \hat{h}^{-1}(0)$ to be the root of $\hat{h}(\cdot)$, obtained via a standard deterministic root-finding method (say Newton's method if $\hat{h}'$ is also available). In these scenarios, the practitioner is actually faced with an SRFP, but chooses, albeit implicitly, to solve it as a DRFP. Surrogate modeling offers an opportunity to import the vast machinery of emulation/meta-modeling construction which is an extensive topic in the design of simulation experiments [36], as well as in the simulation optimization and computer experiment literatures [9]. Within this context, root-finding is equivalent to contour-estimation, i.e., learning the boundary of $\{x : h(x) > 0\}$, see [47, 4, 23].

Some of the drawbacks of response surface modeling (RSM) described above is that a "good" representation for h usually does not lead to tractable models for $\hat{x}$. For example, consider a Gaussian process (GP) prior for h. A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution [48]. Then, the marginal distribution $h_*(x)$ is also Gaussian for any fixed x, however there is no closed-form expression for the distribution of $h_*^{-1}(0)$. A viable choice to overcome the latter limitation is for example build a bootstrapped empirical density for the root location based on iteratively estimating the root location as more data is available. Nevertheless, the latter heuristic ignores the dependency on the surrogate election as well as on the numeric error due to replacing the original problem of estimating the root x^* with the RSM approach as we show in our numeric examples presented in Chapter 4.

The above limitation points to the second alternative of modeling x^* itself, with h as a background latent object. In this framework, statistical inference is conducted directly on the root, considering x^* as an unknown parameter to be inferred via the realized data $Z_{1:n}$ obtained after sampling (1.1) at n locations $x_{1:n}$. A natural strategy towards constructing an estimator $\hat{x}_n$ for x^* given $Z_{1:n}$, is to follow a fully Bayesian approach: update knowledge about x^* based on *prior* information about the shape of the underlying function h (e.g., h is non-increasing) and the evidence provided by the responses $Z_{1:n}$ — whose statistical properties are governed by the distribution of the random component $\epsilon(\cdot)$ in (1.1).

1.2 Probabilistic Bisection for Stochastic Root-Finding

One promising Bayesian alternative which accounts for both the *estimation* and *design* component is the Probabilistic Bisection Algorithm (PBA), recently applied to solve one-dimensional SRFPs by Waeber et al [57].

The PBA leverages the classical *bisection search* in a noise-free setting: iteratively halve the search region and then select a subinterval in which a root must lie for further processing. In the stochastic case, however, the PBA accounts for noise in the oracle responses by considering x^* as the realization of an absolutely continuous random variable $X^* \sim g_0$ with prior density g_0 supported on $(0, 1)$. The PBA then works with the *sign* of the noisy function evaluations,

$$Y(x_n) := \text{sign}\{Z(x_n)\}; \tag{1.2}$$

which provide information as to whether x^* lies to the left or to the right of a given point x_n, in order to subsequently update a posterior density for X^*,

$$g_n(X^*) := p(X^*|Y_{1:n}, x_{1:n}), \tag{1.3}$$

that is, the conditional pdf of the root location X^* given the history of oracle responses $Y_{1:n} := (Y_1(x_1), \ldots, Y_n(x_n))$, sampling locations $x_{1:n}$ and prior g_0.

The posterior (1.3) then serves the twin purposes of guiding the election of the next sampling location x_{n+1} at which to query (1.2), as well as to provide an estimator $\hat{x}_n$ for X^* (e.g., the posterior median or mean of g_n).

Notice that due to the noise term $\epsilon(x_n)$ in the simulation outputs $Z(x_n)$, then the responses $Y(x_n) := \text{sign}\{Z(x_n)\}$ translate into potentially inaccurate oracle directions. To account for noise in (1.2) the PBA considers the probability of *correct sign*,

$$p(x_n; x^*) := \mathbb{P}(Y(x_n) = \text{sign}\{x_n - x^*\}), \quad x_n, x^* \in (0,1); \tag{1.4}$$

(henceforth referred as oracle *specificity* or *accuracy*), which is then used to update knowledge about X^* by re-weighting the current g_n proportionally to $p(x_n) \equiv p(x_n; x^*)$. Figure 1.1 shows an example of a realization of the classical PBA updating g_n (i.e., when $p(x_n)$ is known) captured at stages $n = 0, 1, 10$, implementing a linear function,

$$h_1(x) = x^* - x, \quad x \in (0,1),$$

which is henceforth our running example with root at $x^* = 1/3$. Figure 1.1 illustrates three fundamental components of the PBA:

- At each step the value of $p(\cdot)$ is required to update from g_n to g_{n+1}. The top panel of the second column in Figure 1.1 shows that the updating takes the prior g_0 and then "re-calibrates" knowledge based on whether a positive/negative response $Y(x_1)$ is observed and the corresponding probability $p(x_1)$ of that direction being correct after updating at x_1.

- The PBA is able to account for noise in oracle directions $Y(\cdot)$ by assigning non-zero probability $g_n(\cdot)$ at regions where X^* is believed to be based on the history of responses $Y_{1:n}$. For instance, notice that after $n = 10$ most of the mass of g_n is concentrated around the true x^* as illustrated in the top right panel the Figure 1.1.

- If one were to blindly apply deterministic bisection search for the stochastic case; a single wrong direction will divert the search path almost surely, as it is the case in the *positive* response observed *rightwards* x^* as seen on the bottom right panel of Figure 1.1.

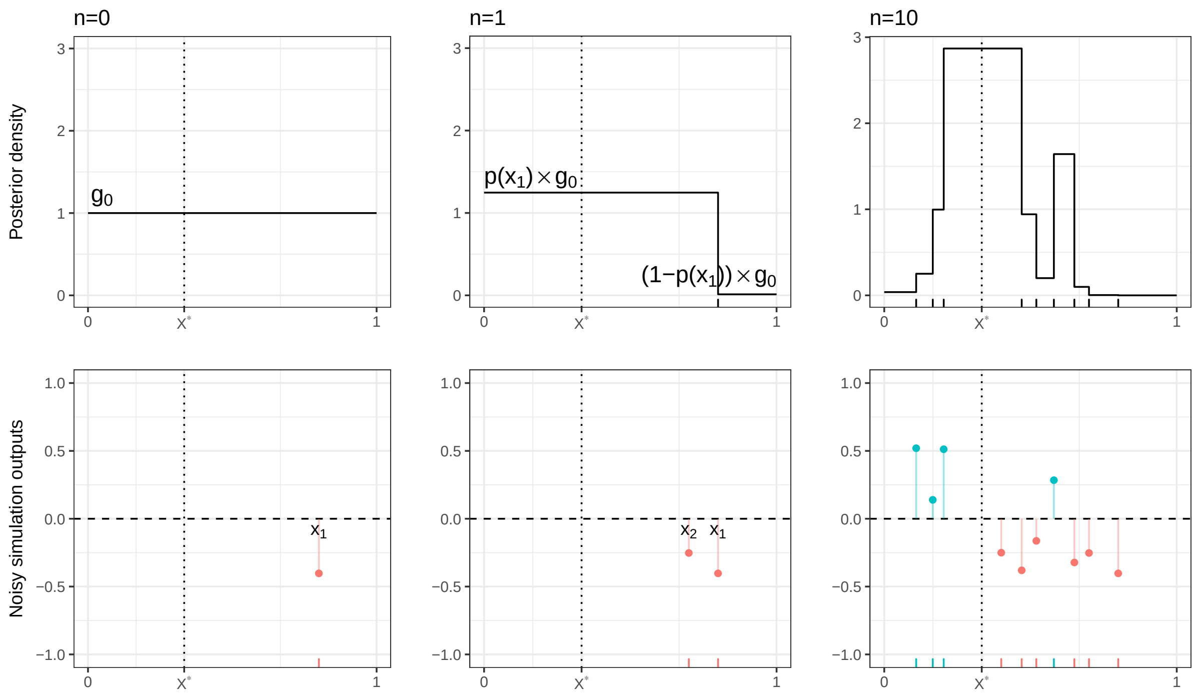

Figure 1.1: Knowledge updating using PBA with known $p(x)$ at stages $n \in \{0, 1, 10\}$ using the linear test function $h_1(x) := x^* - x$ with $x^* = 1/3$.

Classical PBA with constant and known oracle properties. Under the assumption that the oracle accuracy is *known* and *constant* (i.e., $p(x_n) \equiv p, \forall x_n$), Waeber at al. established exponential convergence of the posterior g_n to a Dirac mass at the true x^* when the next location is given by $x_{n+1} = \text{median}(g_n)$. While, in practice, these conditions are not typically going to be satisfied (as in our Bermudan Put example outlined in Section 1.1), such explicit performance guarantees are highly desirable and have not been available via RSM approaches.

In a realistic context, the oracle specificity (1.4) is unknown and spatially varying in x_n (since it intrinsically depends on $h(x_n)$ as well as on the statistical properties of $\epsilon(x_n)$ in (1.1)). Without further assumptions, the only way to employ PBA is to *estimate* $p(x_n)$. This was already noted in [57, 19] where a hypothesis-testing-inspired procedure was used to learn $p(x_n)$ en route to learning X^*. Specifically, they employed a Test of Power One (TPO) [55], which relies on repeated sampling of (1.2) to effectively boost $p(x_n)$ to a fixed accuracy level $\tilde{p}(x_n)$. However, such boosting can be very expensive in the regime where $p(x_n) = 0.5 + \delta$ for small $\delta > 0$. This highlights the second challenge with PBA: in the context of root-finding and (1.1), for x_n close to x^* we have $p(x_n) \simeq 0.5$ which implies that the oracle is *uninformative* in the neighborhood of the root. A naïve implementation of PBA leads to sampling too close to x^* and is not asymptotically convergent in the sense of the posterior collapsing to a Dirac mass at x^*.

1.3 Generalized PBA for Stochastic Root-Finding

In this thesis, we resolve the inherent challenges in the classical probabilistic bisection scheme by providing a *practically-minded* extension of PBA. We construct a class of algorithms, which we term generalized PBA (G-PBA) that can:

(P1) efficiently learn oracle properties;

(P2) aggregate collected information to construct a sequential design; and

(P3) update knowledge about the root location as new information becomes available.

To do so, similar to [57] we rely on *batched sampling* to learn $p(\cdot)$; however in contrast to the latter TPO strategy that evaluates the oracle a random amount of times in order to avoid estimating $p(\cdot)$ explicitly, we work with a *fixed* batch size $a \geq 1$. We demonstrate through the numeric examples presented in Chapter 4, fixed batching is more efficient, in particular by providing better control over the size of each batch, as well as allowing the user to control the number of design points to be explored.

Estimating oracle properties. For (P1) we propose a collection of inference methods which leverage the fundamental assumption of the PBA for SRFPs, that the distribution of the noise component $\epsilon(\cdot)$ is *symmetrical.* The latter allows for the re-parametrization of $p(x)$ as:

$$p(x) = \max\{\theta(x), 1 - \theta(x)\}, \quad \forall x \in (0, 1); \tag{1.5}$$

where

$$\theta(x) := \mathbb{P}(Y(x) = +1) \tag{1.6}$$

is the probability of observing a single positive response at x. Consequently, we will translate our inference problem of learning (1.5) into learning (1.6), to finally obtain plug-in estimators of the form

$$\hat{p}(x) = \max\{\hat{\theta}(x), 1 - \hat{\theta}(x)\}$$

for $p(x)$. Naturally, the information that is used to find $\hat{\theta}(x)$ is

$$B(x) := \sum_{j=1}^{a} 1_{\{Y_j(x)=+1\}}, \tag{1.7}$$

which counts the number of positive responses observed at x after evaluating (1.2) a-times.

In this thesis, we present two inference paradigms which intrinsically are built upon the binomial response (1.7) and whose differences reside in whether spatial correlation across sampling locations and corresponding binomial responses is leveraged or not.

The first class of estimators which do not borrow information across sampling locations is presented in Chapter 2, henceforth also referred as *local* estimators. An example of such estimators is the majority proportion estimator

$$\bar{p}(x) := \max\{\bar{B}(x), 1 - \bar{B}(x)\},$$

where $\theta(x)$ is estimated via the binomial proportion $\bar{B}(x) := B(x)/a$.

A key characteristic of the estimator $\bar{p}(x)$ is the *majority-vote* property, that is, if a positive sign is observed majoritively at x (i.e., $B(x) > \lceil a/2 \rceil$), then the probability of an accurate oracle response $p(x) > 1/2$ with high confidence. In Chapter 2 we analyze the statistical properties of $\bar{p}(x)$ such as bias and consistency.

Within the same class of local estimators, we furthermore introduce a Bayesian family of statistical procedures, $\hat{p}_{\mathscr{L}}(x)$, based on a posterior density $\pi(p|\bar{p}(x))$ for p given the majority proportion $\bar{p}(x)$ and a prior distribution for $p \sim \pi_0$. With $\pi(p|\bar{p}(x))$ in closed-form we propose a collection of optimal Bayes estimators based on several loss functions $\mathscr{L}$.

In Chapter 3, we present a second estimation paradigm which is able to borrow information across sampling locations. To do so, we deploy *spatial surrogate* models which rely on a classical binomial (logistic) regression approach: it is assumed that the locations x_i's are related to $\theta(x_i)$ via the canonical Bernoulli *link function*

$$\log\left(\frac{\theta(x_i)}{1-\theta(x_i)}\right) = \varphi(x_i), \quad i = 1, \dots, n;$$

and where $x \mapsto \varphi(x)$ is a *surrogate* model to be trained from the data $(x_i, B_i(x_i))_{i=1}^n$ for $n < N$. Specifically, we seek non-parametric regression approaches which are able to refine the regression curve in regions where more data points are placed (namely close to the root), and simultaneously give a good global fit. In particular, we implement two families of surrogates:

(A) a Gaussian random field (GRF) modeling approach (also known as Gaussian process modeling, [48]) that takes φ as a latent Gaussian process and outputs the posterior distribution $p(\varphi_*|\mathcal{D}_n)$ conditional on the data $\mathcal{D}_n := (B_{1:n}, a_{1:n}, x_{1:n})$; and

(B) a linear additive model that assumes that φ is an element of a linear space spanned by a collection of basis functions, i.e., $\varphi(x) = \sum_{j=1}^{p} \beta_j \phi_j(x)$, with the coefficients $\boldsymbol{\beta} := (\beta_1, \dots, \beta_p)$.

Both modeling schemes (A) and (B) are mathematical approximations (metamodels) capable of modeling the relation $x \mapsto \theta(x)$. *Metamodels* are particularly useful since they can be built upon available observations and updated when new data is assimilated. They can also be used to guide simulator evaluations more efficiently [51]. Given the fitted surrogate $\hat{\varphi}_n$, the estimate for $p(\cdot)$ is a plug-in estimate of the form:

$$\hat{p}_n(\cdot) := \max\{\hat{\theta}_n(\cdot), 1 - \hat{\theta}_n(\cdot)\}; \text{ where } \hat{\theta}_n(\cdot) := [1 + e^{-\hat{\varphi}_n(\cdot)}]^{-1}. \tag{1.8}$$

Binomial Gaussian Processes and Adaptive Querying. In the context of inferring $\theta(x)$ a fundamental question is to optimally determine optimally the amount of replication $a(x)$ at each location x where the batched response $B(x)$ is observed. The answer to this question is primarily driven by two concepts: (i) accuracy, as measured by successfully predicting $p(x)$ at values of x which are close from the root x^*; and (ii) allowing the algorithm to explore the search region in a way that maximize computational efficiency.

The latter idea has been already posed for the problem of design of GP surrogates in the face of heteroscedastic simulation experiments. Namely, Binois et al. [8] study the conditions under which the new element should be a replicate versus explore a single response. Analogous to the latter approach, we utilize the Binomial GP surrogate (A) to adaptively determine the batch size a_{n+1} as a function of the sampling location x_{n+1} in order to address the replication/exploration trade-off (which becomes critical at locations close from the root location).

In this context, in Chapter 3 we present an approximated adaptive replication scheme where a_{n+1} is computed as a function of the estimated predictive variance at x_{n+1}. Intuitively, this approach will employ a larger batch size at locations where the posterior uncertainty of the latent is large and a smaller batch size for locations where $\theta(\cdot)$ has already been learned sufficiently well. We again contrast this adaptive replication regime with TPO, in which querying ignores previous information leading to oversampling, whereas in our approach the replication size can be as small as $a_{n+1} = 1$ for locations where the predictive variability of $\theta(\cdot)$ is sufficiently small.

Sampling policies. A sampling policy η is a rule which maps states to actions. In the context of SRFP actions effectively correspond to selecting sampling locations x_{n+1}. As in standard PBA, we work in the sequential paradigm for (P2), selecting x_{n+1} based on a Bayesian perspective. Namely, to generalize the ideas of PBA to the setting of unknown $p(\cdot)$ we introduce a *knowledge state*. The state of a system can be described as consisting of all the information needed to make a decision, compute the objective (contributions and rewards), and compute the transition to the next state [45]. The knowledge (or belief) state f_n captures our distribution of belief about X^* that we do not know perfectly. The underlying philosophy is a Bayesian formulation of SRFP, translating the task of learning the root X^* into the language of "beliefs" encapsulated by f_n and used to quantify (posterior) uncertainty about X^*. Intuitively, f_n is a "surrogate" to the true posterior g_n that is no longer attainable due to unknown $p(\cdot)$. The knowledge state f_n is then used for the dual purposes of providing an estimate $\hat{x}_n$ of X^*, and for guiding the sequential design.

We propose a collection of sampling strategies which blend the estimation procedures $\hat{p}$ developed for (P1) with the concept of information directed sampling and quantile-sampling strategies. We also show the effectiveness of randomization for both schemes which turns out to be crucial in preventing uncontrolled error propagation in constructing the knowledge state. In Chapter 2 and Chapter 3 we compare respectively the performance of such policies using local and spatial modeling for inferring $p(\cdot)$.

Knowledge Updating. The main challenge under the G-PBA paradigm where we have unknown and location-dependent oracle accuracy, is that estimation for $p(\cdot)$

and knowledge updating about X^* must be performed *simultaneously*. Towards addressing (P3), in Chapter 2 we explicitly construct an updating mechanism

$$f_{n+a} := \Psi(f_n, x_{n+1}; \hat{p}_{n+1}, a_{n+1})$$

capable of blending both estimation and knowledge state updating about X^*.

To obtain the *knowledge transition function* $\Psi(\cdot)$ we extend the classical PBA Bayesian updating regime to construct a batched version which relies on sampling a_{n+1} replications at x_{n+1} and observing the total number of positive responses $B(x_{n+1})$ in order to construct $\hat{p}_{n+1}$. Given a_{n+1} and $\hat{p}_{n+1}$ knowledge updating occurs necessarily from f_n to f_{n+a}. This knowledge updating mechanism given by $\Psi(\cdot)$ represents some of the main contributions of this thesis and are presented in Chapter 2.

1.3.1 Summary of Contributions and Related Literature

Our G-PBA schemes are generic in that they make minimal assumptions about the underlying (1.1), and can be employed across a wide spectrum of SRFP's. To illustrate this robustness we use G-PBA on our motivating example above to learn the critical exercise threshold in the context of Regression Monte Carlo for Optimal Stopping. In that case, the behavior of (1.1) is highly non-standard, expressing strongly heteroskedastic and non-Gaussian characteristics, to which standard statistical learning procedures for $\hat{h}$ are very sensitive [39]. In contrast, the G-PBA is rather agnostic with respect to the usual homoscedasticity and Gaussianity assumptions, not least thanks to the batching substeps which allow for the Central Limit Theorem (CLT) to smooth statistical anomalies.

To provide further context for this thesis, let us recapitulate our contributions relative to existing methods. Our contributions can be traced along several directions.

First, compared to PBA, we work with *unknown* and *location-dependent* oracle accuracy $p(x)$, which requires a complete re-imagining of the algorithm, focusing on practical solutions that work well in non-asymptotic settings, where we are constrained by the budget of T available oracle calls. In particular, we contrast our strategies with the proposals in [57, 19] that employ the TPO approach to learn $p(x)$, as we will show, while TPO enjoys nice theoretical properties and is a viable alternative in terms of its asymptotic behavior, it performs poorly for a small sampling budget T.

Second, compared to simulation optimization, we develop a root-finding procedure which is built around the notion of constructing an explicit posterior density for the root, and hence, primarily operates with the knowledge state rather than a surrogate for $h(\cdot)$. This allows us to obtain and monitor the (pseudo-) credible bands for X^* which give sequential quantification of the learning performed by PBA. Thusly, we contribute to the greater stochastic root-finding toolkit.

Finally, our sequential updating and sampling strategies can be linked to the literature on active learning since they are based on the posterior uncertainty quantification of X^* rather than an h-based statistic — grounding our method in a purely information-theoretic paradigm. In that sense, we make use of an acquisition function [11] which maps previously assimilated information that is condensed by the surrogate.

Chapter 2

Generalized Probabilistic Bisection

2.1 Introduction

A complete solution to the SRFP using the PBA was provided by [59] under the key assumption that the oracle specificity (1.4) is a *known* and x-independent constant, i.e., $p(x) \equiv p, \forall x \in (0,1)$. Namely, Waeber et al. derived the equations for the posterior density

$$g_n(X^*) := p(X^*|y_{1:n}, x_{1:n}) \tag{2.1}$$

and then established that sampling at the posterior median, $x_{n+1} := G_n^{-1}(0.5)$ for all $n = 1, 2, \ldots$, is an optimal policy. More precisely, they proved that this sampling rule minimizes the expected Kullback-Leibler (KL) distance for its utility function, and most importantly achieves exponential convergence for the estimate $\hat{x}_n \equiv x_n$ towards x^*, i.e., $|\hat{x}_n - x^*| = \mathcal{O}(e^{-\alpha n})$ with an explicit expression for $\alpha > 0$. This justifies its name, as the PBA manages to effectively reduce the interval containing x^* by $\alpha\%$ at each stage. This result is truly impressive both given the noisy oracle replies and thanks to the simplicity of the sampling rule. This assumption of spatial oracle stationarity would tend to be met in applications where the transition between regions in h is abrupt.

As an example, if a city's water supply were contaminated with a dangerous chemical we would want to localize the extent of contamination as quickly as possible, and if the chemical did not dissolve well in water but instead tended to stay concentrated, we would face a situation with such abrupt transition between contaminated and uncontaminated water [45]. Moreover, PBA exemplifies the Bayesian setup: x_{n+1} is selected based on the information summarized by g_n, which also yields the point estimate $\hat{x}_n$.

Some partial results extending to the case where $p(x)$ is non-constant (but still *known*) were given in [58]. The crucial assumption that oracle properties, specifically $p(x)$ is known, is hard to justify in the context of unknown response $h(\cdot)$. For example, when the noise component in (1.1) is $\epsilon(x) \sim \mathsf{N}(0, \sigma^2(x))$, then $p(x) = \Phi\left(|h(x)|/\sigma(x)\right)$, where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of a standard Gaussian, and therefore knowledge of $p(\cdot)$ is equivalent to knowing the signal-to-noise ratio — a rather unlikely proposition. At the same time, the known-p assumption is critical to the performance: as we discuss below without further modifications the PBA might fail completely in the context of unknown $p(x)$. More sophisticated sampling strategies are needed to resolve this tension between exploitation and exploration.

To generalize the ideas of PBA to the setting of (1.1) we introduce a knowledge state, f_n, that is recursively updated and used for acquiring new samples. The underlying philosophy is a Bayesian formulation of SRFP, translating the task of learning the root into the language of "beliefs" encapsulated by f_n and used to quantify (posterior) uncertainty about X^*. Intuitively, f_n is a "surrogate" to the true posterior (2.1) that is no longer attainable due to unknown $p(\cdot)$.

A key ingredient of our approach is the use of *replications*: repeatedly evaluating the oracle $a \in \mathbb{N}$ times at a fixed sampling location x. In this sense, a is the "sample size" (aka *batch size*), that is, some measure of simulation effort that is usually well-defined depending on the context. In the context of terminating simulations, for example, a usually refers to the number of times the simulation is called in computing the estimator $\hat{p}(x)$. In non-terminating simulations, a usually refers to the "length of time" the simulation is executed when computing the estimator $\hat{p}(x)$ [43].

Replications (henceforth also referred as *batched sampling*) allows us to obtain a point estimate $\hat{p}(x)$ for $p(x)$ based on counting the total number of positive responses $B(x)$ observed at x as in (1.7), which is then used to update knowledge from f_n to f_{n+a}. Replicates decouple the problems of learning X^* and of learning $p(\cdot)$; they also boost the signal-to-noise ratio which allows faster convergence at the macro-level. Our resulting G-PBA framework learns in *parallel* X^* and $p(\cdot)$ and is summarized in Algorithm 1.

```
input : Total query budget $T$; batch size $a$ and prior distribution $f_0$ on the root.
for $n \leftarrow 0, 1, \ldots, N-1$ do
    Generate next sampling point $x_{n+1}$;
    Obtain the estimate $\hat{p}(x_{n+1})$ using the binomial response $B(x_{n+1})$;
    Update knowledge state to $f_{n+a} := \Psi\left(f_n, x_{n+1}, B(x_{n+1}); \hat{p}(x_{n+1}), a\right)$;
end
return Root estimate $\hat{x}_N$; Knowledge state $f_T$.
```

Algorithm 1: Generalized PBA.

In order to implement Algorithm 1, the G-PBA must specify:

(GPBA-I) statistical procedure $\hat{p}(x_{n+1})$ for estimating $p(x_{n+1})$ at x_{n+1}.

(GPBA-II) the mechanism to update knowledge states $\Psi : f_n \to f_{n+1}$;

(GPBA-III) the rule η for selecting $x_{n+1} = \eta(f_n)$ given f_n;

All three of the steps (GPBA-I), (GPBA-II) and (GPBA-III) require novel analysis, and are a part of the main contributions of this thesis.

(GPBA-I) Statistical procedure for estimating $p(\cdot)$. All three steps above require knowledge of $p(x)$, so proper inference of the latter is central to the G-PBA performance. The symmetrical noise distribution assumption in (1.1) implies that the oracle is “democratic”: $p(x) \geq 0.5 \forall x$, and thus there is an implicit majority-vote property in $p(x)$, whereby the estimate is based on the majoritatively observed sign of the binomial response $B(x)$. This introduces a fundamental bias which becomes especially significant when sampling close to X^* ($|h(x)|$ is small and $p(x) \simeq 0.5$).

In Section 2.2 we investigate three types of p-estimators: frequentist based on majority proportion; Bayesian based on the posterior density of p given $B(\cdot)$; and a collection of boosted estimators which directly aggregate oracle responses to construct a subsidiary signal whose *specificity* is enhanced thanks to batching.

A fundamental property of the statistical procedures developed under this setting, is that $p(x)$ is inferred based solely on the information collected at x via the summary statistic $B(x)$. In this thesis, we refer to the latter paradigm as *local* estimation as no information across sampling locations in leveraged for constructing the estimator $\hat{p}(x)$ at location x.

(GPBA-II) Knowledge Updating Procedure. To update f_n we then plug-in an estimated $\hat{p}(x)$ into a knowledge state transition function of the form

$$f_{n+a} := \Psi\left(f_n, x_{n+1}, B(x_{n+1}); \hat{p}(x_{n+1}), a\right), \quad n = 0, 1, \ldots, N-1 \quad \text{and } a \in \mathbb{N}, \tag{2.2}$$

where the sufficient statistic $B(x_{n+1})$ is defined in (1.7). The map (2.2) is the analogue of Bayesian updating when $p(x)$ is known . Note that the knowledge transition $\Psi(\cdot; \hat{p}, a)$ function is similarly batched, allowing us to make full use (while maintaining computational efficiency) of the sampled replicates. This aspect is fully addressed in Section 2.2.

(GPBA-III) Sampling Policies. Third, to select the locations x_{n+1} at each $n = 0, \ldots, N-1$ we introduce several sequential sampling policies η. The first family of *Information Directed Sampling* uses an information gain function $\mathcal{I}(x, f_n; p(x), a)$

to quantify the learning rate for X^* if a new query batch is done at x. Notice that the acquisition function $x \mapsto \mathcal{I}(x, f_n; p(x), a)$ depends on either the knowledge state f_n and the nuisance parameter $p(x)$ for each x for fixed batch size a. It is motivated by the optimality property of standard PBA in terms of maximizing the KL relative entropy between g_n and g_{n+1}.

The second family of *Quantile Sampling* is motivated by the other aspect of PBA, namely of sampling at the median of the knowledge state. Letting F_n be the CDF of f_n, we therefore propose to use the quantiles of f_n for selecting the next x_{n+1}, i.e., $x_{n+1} := F_n^{-1}(q)$, where $q \in (0, 1)$.

Another important computational adjustment that we entertain is an additional degree of randomization which serves to (a) alleviate the issue of error accumulation arising from uncertainty in estimating $p(x_n)$ and (b) enforce exploration of the state space in order to accelerate convergence to the true X^*. Our experiments demonstrate the value of such randomized sampling policies and can be viewed as analogues of similar stochastic searches in Bayesian optimization (such as Thompson sampling [53]). Full analysis of these designs is in Section 2.3.

Wall-clock and macro time. Note that due to batching G-PBA will have two different time scales: macro-iterations $n = 1, \dots, N$ corresponding to the query locations $x_{1:n}$, where N is the total number of sampling points for a fixed batch size a; and *wall-clock time*, $T = a \times N$, which counts the total number of oracle queries and hence the overall computational expense.

Estimating the root X^*. The final ingredient is the rule $\hat{x}_n$ to construct an estimate of the root based on f_n. In analogy to the classical PBA setting, in this thesis we utilize the posterior median (which we find is generally more robust than say the mean, as f_n

is often skewed or multi-modal),

$$\hat{x}_n := \text{median}(f_n). \tag{2.3}$$

2.2 Knowledge States

Consider a real-valued continuous response function $h : (0,1) \to \mathbb{R}$. For concreteness we have re-scale the (bounded) input space to the unit interval. The function h is noisily sampled via the stochastic simulator (1.1). Let X^* be the random root location and x^* its realized value at which $h(x^*) = 0$. To learn x^*, the PBA works with the signs $Y(x) := \operatorname{sign} Z(x)$, which due to the stochastic nature of the responses (1.1), are *correct* with probability $p(x)$.

Assuming that $p(x)$ is known, the next Lemma provides the analytical one-step updating equations for the posterior g_n of X^* defined in (2.1).

Lemma 2.2.1 (Updating formula for posterior density of the root location X^*). [59] Let $x \in (0,1)$, G_n be the CDF of g_n and $p(x)$ as in (1.4). Define

$$\gamma_n(x; p(x)) := p(x)[1 - G_n(x)] + [1 - p(x)]G_n(x), \tag{2.4}$$

Given a prior g_0 on X^* we have the recursion:

$$g_{n+1}(u) = \frac{1}{\gamma_n(x; p(x))} \left[p(x) 1_{\{u \geq x\}} + (1 - p(x)) 1_{\{u < x\}} \right] g_n(u), \tag{2.5a}$$

if a negative sign is observed at x, i.e., $Y_n(x) = +1$; and

$$g_{n+1}(u) = \frac{1}{1 - \gamma_n(x; p(x))} \left[(1 - p(x)) 1_{\{u \geq x\}} + p(x) 1_{\{u < x\}} \right] g_n(u), \tag{2.5b}$$

otherwise, for all $n = 0, 1, \ldots$.

Remark 1. If no prior knowledge about the root location X^* is provided, then a sensible choice is a vague prior $g_0 = \mathsf{Unif}(0,1)$. The latter is also computationally convenient, since

(2.5) then implies that g_n will be piecewise constant $\forall n$, with discontinuities precisely at the sampled $x_{1:n}$. Therefore, storage and updating of g_n becomes an $\mathcal{O}(n)$ operation in this setup.

2.2.1 Batched Querying

Abstractly, the updating (2.5) constitutes a *knowledge transition function* $\Psi : g_n \mapsto g_{n+1}$, which takes as inputs the current knowledge state g_n, the oracle response $Y_n(x)$ and its specificity $p(x)$ when queried at the point $x \in (0,1)$. To learn $p(x)$, we employ batched queries, keeping the sampling location x unchanged for $a \geq 2$ steps. Considering the resulting i.i.d. sequence of oracle responses $(Y_j(x))_{j=1}^a$, the knowledge state g_n can be recursively computed by using the update (2.5) a-times to obtain g_{n+a}. Because $p(x)$ is the same across those updates, we can simply consider the total number of *positive* oracle responses observed at x, $B(x)$, yielding an aggregated knowledge transition function from g_n to g_{n+a}.

Remark 2. The summary statistic $B(x)$ defined in (1.7) intrinsically depends on the sampling location x, as well as on the batch size a. However, in order to ease our notation we omit the dependency of $B(x)$ on a, unless necessary.

Theorem 2.2.2 (Batched Bayesian knowledge transition function). Let g_n be the current knowledge state about X^* and $p(\cdot)$ the probability of a correct oracle response. The *batched* Bayesian updating, Ψ, which maps g_n to $g_{n+a} := \Psi(g_n(u), x, B(x); p(x), a)$ is given by

$$g_{n+a}(u) = \begin{cases} c_n^{-1}(x)\left[p(x)^{B(x)}(1-p(x))^{a-B(x)}\right] \cdot g_n(u) & \text{if } 0 < x < u < 1, \\ c_n^{-1}(x)\left[(1-p(x))^{B(x)}p(x)^{a-B(x)}\right] \cdot g_n(u) & \text{if } 0 < u \leq x < 1; \end{cases} \tag{2.6}$$

for all $x \in (0,1)$ with normalizing constant

$$c_n(x) := \left[(1-p(x))^{B(x)}p(x)^{a-B(x)}\right] G_n(x) + \left[p(x)^{B(x)}(1-p(x))^{a-B(x)}\right](1-G_n(x)). \tag{2.7}$$

Proof. We will show that the updating equations (2.6) hold for any $a \in \mathbb{N}$ via mathematical induction. To do so, let x be a fixed sampling location in the support of g_n, and $B \in \{0, 1, \ldots, a\}$ to be the total number of observed positive signs after querying the oracle $a \geq 1$ times at x, and $Y_j \in \{-1, +1\}$ the j-th oracle response observed at x for $j = 1, \ldots, a$ (dropping the dependency on x and a in B and the Y_j's). Re-expressing the knowledge transition function (2.6) using indicator functions as (and disregarding the normalizing constant $c_n(x)$ in (2.7)):

$$g_{n+a}(u) \propto \left[\sum_{j=0}^{a} p^j (1-p)^{a-j} 1_{\{B=j\}}\right] g_n(u) 1_{\{u \geq x\}} + \left[\sum_{j=0}^{a} (1-p)^j p^{a-j} 1_{\{B=j\}}\right] g_n(u) 1_{\{u<x\}},$$

with $p \equiv p(x)$, we will prove that (2.6) holds for any a and fixed n. For $a = 1$ we have $\{B = 1\} = \{Y_1 = +1\}$ and $\{B = 0\} = \{Y_1 = -1\}$ and (2.6) corresponds to the updating in (2.5). We now concentrate on the case $u > x$ and inductively suppose Equation (2.6) holds for a; we now establish it for $a + 1$:

$$\begin{aligned} g_{n+(a+1)}(u) &\propto \left[(1-p)1_{\{Y_{a+1}=-1\}} + p \cdot 1_{\{Y_{a+1}=+1\}}\right] g_{n+a}(u) \\ &= \left[(1-p)1_{\{Y_{a+1}=-1\}} + p \cdot 1_{\{Y_{a+1}=+1\}}\right] \times \left[\sum_{j=0}^{a} p^j (1-p)^{a-j} 1_{\{B=j\}}\right] g_n(u) \\ &=: (A_1 + A_2) g_n(u). \end{aligned}$$

We now have

$$A_1 = \sum_{j'=0}^{a} p^{j'+1} (1-p)^{a-j'} 1_{\{B=j', Y_{a+1}=+1\}} = \sum_{j=0}^{a+1} p^j (1-p)^{a+1-j} 1_{\{B=j-1, Y_{a+1}=+1\}}.$$

Similarly we obtain $A_2 = \sum_{j=0}^{a+1} p^j (1-p)^{(a+1)-j} 1_{\{B=j, Y_{a+1}=-1\}}$, which implies that

$$\begin{aligned} A_1 + A_2 &= \sum_{j=0}^{a+1} p^j (1-p)^{(a+1)-j} \left[1_{\{B=j, Y_{a+1}=-1\}} + 1_{\{B=j-1, Y_{a+1}=+1\}}\right] \\ &= \sum_{j=0}^{a+1} p^i (1-p)^{(a+1)-j} 1_{\{B_{a+1}=j\}}. \end{aligned}$$

Analogous argument works for $u < x$. $\square$

Hence, if we furthermore define the *right scaling-factor*

$$\rho(x, B(x); p(x), a) := p(x)^{B(x)}(1 - p(x))^{a-B(x)}, \tag{2.8}$$

then the ratio

$$R^{(a)}(g_n, x, B(x); p(x)) := \rho(x, B(x); p(x), a)/c_n(x) \tag{2.9}$$

completely specifies Ψ in (2.6): given the total number of positive responses $B(x) \in \{0, 1, \dots, a\}$, the new posterior $g_{n+a}(u)$ is recovered by *scaling* the values of $g_n(u)$ for $x \leq u$ by the factor ρ from (2.8) divided by the normalizing constant $c_n(x)$ from (2.7). Hence, if $B(x) > \lfloor a/2 \rfloor$, i.e., there is favorable evidence that x^* is rightwards of x, then the mass of g_{n+a} is shifted to the right of x. In the case where $p(x) \in \{0, 1\}$, (2.8) is defined by $\rho(x, B(x); a, p(x)) := p(x)$, which effectively reduces the support of g_{n+a} by placing zero mass on one of the intervals that have x as an end-point.

Approximate Knowledge State f_n. For our G-PBA algorithms, neither (2.5) nor (2.6) are feasible, since they require the unknown $p(x)$. Nevertheless, to mimic the Bayesian updating paradigm we introduce an *approximate* knowledge state f_n which follows the transition function in (2.6) by plugging-in an appropriate estimate $\hat{p}(x)$, i.e.,

$$f_{n+a} := \Psi(f_n, x, B(x); \hat{p}(x), a); \quad a \geq 2 \text{ and } x \in (0, 1), \tag{2.10}$$

for $n = 0, \dots, N - 1$ and where $\Psi(\cdot; \hat{p}, a)$ is computed via Theorem 2.2.2, for fixed a and statistical procedure $\hat{p}$. Note that because (2.10) is necessarily an approximation, f_n does not match the true posterior g_n.

2.2.2 Frequentist and Bayesian Estimators for $p(\cdot)$

The task in this section is to perform statistical inference on the unknown (nuisance) parameter $p(x)$ required to implement Bayesian updating about X^*, by using the batched i.i.d. responses $(Y_j(x))_{j=1}^a$ observed at $x \in (0, 1)$. As mentioned above, we thus leverage the symmetry of the noise component ϵ in the underlying noise component (1.1) so $p(x)$ is re-parametrized via

$$p(x) = \max\{\theta(x), 1 - \theta(x)\}; \qquad \text{where} \quad \theta(x) := \mathbb{P}(Y(x) = +1) \tag{2.11}$$

is the *marginal* probability of observing a *positive* sign at location x. For the remainder of the section we consider a single (macro)-iteration of the overall G-PBA, treating the sampling location x as fixed and suppressed from the notation. To estimate p we construct an estimator for θ and then plug into (2.11).

From a frequentist perspective, we recall that the binomial proportion B/a is an UMVUE for θ and $B \sim \mathsf{Bin}(a, \theta)$ [10]. This yields the *majority proportion* estimator $\bar{p}$ obtained by replacing θ by B/a in (2.11):

$$\bar{p} \equiv \bar{p}(B) := \max\left\{B/a, 1 - B/a\right\}. \tag{2.12}$$

In Lemma 2.2.3, we show that $\mathbb{E}_p[\bar{p}] > p$ is necessarily biased high as soon as $p > 1/2$.

Lemma 2.2.3 (Bias of Majority proportion estimator $\bar{p}$). We have that the bias of of the majority proportion estimator $\bar{p}$ given p is

$$Bias_p(\bar{p}) := \mathbb{E}_p^B[\bar{p} - p] = \mathbb{P}_\theta(B \le \lceil a/2 \rceil - 1) - 2p\mathbb{P}_\theta(B_{a-1} \le \lceil a/2 \rceil - 2) > 0, a \ge 3 \tag{2.13}$$

Proof. For brevity, we drop the dependency on x.

$$\begin{aligned}
\mathbb{E}_p^B[\bar{p}(B)] &:= \mathbb{E}_p^B[\max\{B/a, 1 - B/a\}] \\
&= \frac{1}{a}\left\{\mathbb{E}_\theta^B[B1_{\{B \ge \lceil a/2 \rceil\}}] + \mathbb{E}_\theta^B[(a - B)1_{\{B < \lceil a/2 \rceil\}}]\right\} \\
&= \frac{1}{a}\left\{\mathbb{E}_\theta^B[B] + a\mathbb{P}_\theta[B < \lceil a/2 \rceil] - 2\mathbb{E}_\theta^B[B1_{\{B < \lceil a/2 \rceil\}}]\right\} \\
&= p + \mathbb{P}_\theta(B < \lceil a/2 \rceil) - \frac{2}{a}\mathbb{E}_\theta^B\left[B1_{\{B < \lceil a/2 \rceil\}}\right].
\end{aligned} \tag{2.14}$$

The last term is equal to

$$\mathbb{E}_\theta\Big[B1_{\{B < \lceil a/2 \rceil\}}\Big] = \sum_{i=1}^{\lceil a/2 \rceil - 1} i\binom{a}{i}p^i(1-p)^{a-i} \tag{2.15}$$

$$= ap \sum_{i=1}^{\lceil a/2 \rceil - 1} \binom{a-1}{i-1} p^{i-1}(1-p)^{(a-1)-(i-1)}$$

$$= ap\mathbb{P}_\theta(B_{a-1} \le \lceil a/2 \rceil - 2), \quad B_{a-1} \sim \mathsf{Bin}(a-1, \theta).$$

Substituting the latter quantity into (2.14) and using $Bias_p(\bar{p}(B)) := p - \mathbb{E}_\theta[\bar{p}(B)]$ yields (2.13). □

Intuitively, the bias in (2.12) is due to the possibility that the majority vote points in the wrong direction.

An alternative estimation procedure is to assign a prior for p and then construct a posterior based on the evidence (likelihood) provided by the batched responses $\bar{p}$. Using (2.12) yields the respective conditional likelihood of $\bar{p}$ as:

Lemma 2.2.4 (Likelihood function of majority proportion). Let x be a fixed sampling location at which the oracle is queried $a \ge 2$ times and B be the total number of positive responses observed at x. Then, the likelihood function of the majority proportion estimator, $\bar{p}(B) := \max\{B/a, 1 - B/a\}$, in p is given by

$$\mathbb{P}_p(\bar{p}(B) = j/a) = \begin{cases} \mathsf{Bin}(j; a, p) + \mathsf{Bin}(j; a, 1-p), & j = 0, 1, \ldots, (\lceil a/2 \rceil - 1); \\ \mathsf{Bin}(a/2; a, p), & j = \lceil a/2 \rceil; \end{cases} \tag{2.16}$$

where $\lceil a \rceil$ is the ceiling function, and $\mathsf{Bin}(j; a, \theta)$ is the probability mass function (pmf) of a binomial random variable in $a \ge 1$ independent trials and success probability θ evaluated at $j = 0, \ldots, \lceil a/2 \rceil$.

Proof. Given $B \sim \mathsf{Bin}(a, \theta(x))$ and $\theta(x) := \mathbb{P}(Y(x) = +1) = p(x)1_{\{x^* \le x\}} + (1 - p(x))1_{\{x^* > x\}}$ for $x \in (0, 1)$, we have

$$\mathbb{P}_p(\bar{p}(B) = j/a) := \mathbb{P}_p(\max\{B/a, 1 - B/a\} = j/a)$$

$$= \mathbb{P}_\theta(B = j) + \mathbb{P}_\theta(B = a - j)$$

$$= \binom{a}{j}\theta^j(1-\theta)^{a-j} + \binom{a}{a-j}\theta^{a-j}(1-\theta)^j, \quad \forall j = 0, 1, \ldots, \lceil a/2 \rceil - 1;$$

which is the sum of two binomial densities. Finally, if $j = a/2$ then $\mathbb{P}_p(\bar{p}(B) = 1/2) = \mathbb{P}_\theta(B = a/2)$ which is a single binomial density. □

Assuming a vague prior $p \sim \mathsf{Unif}(1/2, 1)$ (recall that by construction p is known to be $p \geq 1/2$) we then obtain explicitly the posterior density $\pi(p|\bar{p})$.

Theorem 2.2.5 (Posterior density of p given majority proportion $\bar{p}$). Suppose that p has prior density $\pi_0(p) = 2 \cdot 1_{\{p \in [1/2,1]\}}$. Then, for $a \geq 2$, the posterior density of p conditioning on the majority proportion (2.11) is given by

$$\pi(p|j/a) \propto \begin{cases} p^j(1-p)^{a-j} + (1-p)^j p^{a-j}, & \text{if } j = 0, 1, \ldots, (\lceil a/2 \rceil - 1); \\ p^{a/2}(1-p)^{a/2}, & \text{if } j = \lceil a/2 \rceil. \end{cases} \tag{2.17}$$

Proof.

$$\begin{aligned}\pi(p|\bar{p}(B) = j/a) &\propto \mathbb{P}_p(B = j)\pi_0(p) \\ &\propto \binom{a}{j}[p^j(1-p)^{a-j} + (1-p)^j p^{a-j}]1_{(1/2,1)}(p), \quad \text{if } j = 0, 1, \ldots, (\lceil a/2 \rceil - 1);\end{aligned}$$

with the normalizing constant $\beta_1 = \int_{1/2}^1 [p^j(1-p)^{a-j} + (1-p)^j p^{a-j}]\, dp$ which can be expressed in terms of the Beta function. □

Remark 3. Other priors (e.g. location-dependent) for p can be entertained. The $\mathsf{Uniform}$ choice is convenient both as a vague prior, and due to it matching the conjugate Beta-binomial updates.

Figure 2.1 shows the theoretical expected posterior density, $\hat{\pi}(p; x, a) := \mathbb{E}_\theta^B[\pi(p|\bar{p}(B))]$, obtained after averaging the posterior (2.17) with respect to $B(x) \sim \mathsf{Bin}(a, \theta(x))$ for batch size values $a \in \{50, 100, 250, 500\}$ and locations $x > x^*$ so that $p(x) \in \{0.5, 0.60, 0.70\}$, implementing the test function $h_1(x) = x^* - x$ presented in Section 4.1 which corresponds to a decreasing linear function with root at $x^* = 1/3$ defined for $x \in (0, 1)$. It namely shows that posterior is unimodal around the true $p(x)$; furthermore the posterior predictably tightens as a increases locating most of the posterior mass around the true $p(x)$-value.

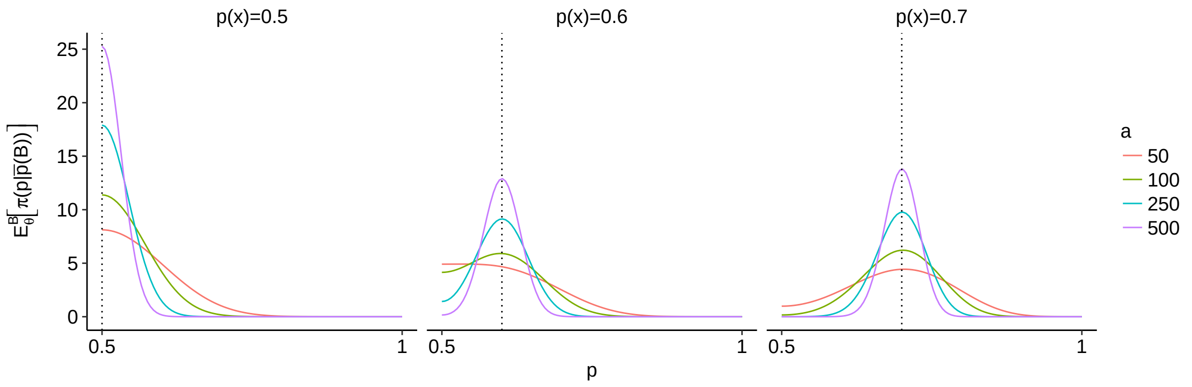

Figure 2.1: Expected posterior pdf $\hat{\pi}(p;x)$ obtained with respect to $B(x) \sim \mathsf{Bin}(a, \theta(x))$ for locations x so that $p(x) \in \{0.50, 0.60, 0.70\}$ (columns) and batch size $a \in \{50, 100, 250, 500\}$ (lines).

With $\pi(\cdot|\bar{p}(B))$ in closed-form, we can obtain a variety of estimators $\hat{p}_{\mathscr{L}}(\bar{p})$ by minimizing the Bayesian posterior expected loss for a given *loss function* $\mathscr{L}(p, \hat{p})$. Namely,

(i) *posterior mode* based on $\mathscr{L}_0(p, \hat{p}) := 1_{\{|\hat{p}-p|>\epsilon, \epsilon>0\}}$ (taking $\epsilon \downarrow 0$ as $\pi(p|\cdot)$ is unimodal),

$$\hat{p}_{\mathscr{L}_0}(\bar{p}) = \text{mode } \pi(p|\bar{p}); \tag{2.18}$$

(ii) *posterior median* based on the L_1 loss $\mathscr{L}_1(p, \hat{p}) := |p - \hat{p}|$:

$$\hat{p}_{\mathscr{L}_1}(\bar{p}) = \text{median } \pi(p|\bar{p}), \tag{2.19}$$

(iii) and *posterior mean* based on the L_2 loss $\mathscr{L}_2(p, \hat{p}) := (p - \hat{p})^2$:

$$\hat{p}_{\mathscr{L}_2}(\bar{p}) = \text{mean } \pi(p|\bar{p}) \tag{2.20}$$

Remark 4. Practically, (2.18) and (2.19) have to be computed numerically, whereas (2.20) is computed in closed form as stated in Corollary 2.2.6.

Corollary 2.2.6. The posterior mean $\hat{p}_{\mathscr{L}_2}(j/a) := \mathbb{E}_B^p[p|\bar{p}(B) = j/a]$ is computed considering two cases:

(i) If $j = 0, 1, \ldots, (\lceil a/2 \rceil - 1)/a$, then

$$\hat{p}_{\mathscr{L}_2}(j/a) := \beta_1^{-1}\Bigg\{\mathcal{B}(j+2, a-j+1)(1 - \int_0^{1/2} \mathsf{Beta}(p; j+2, a-j+1))dp \\ + \mathcal{B}(a-j+2, j+1)(1 - \int_0^{1/2} \mathsf{Beta}(p; (a-j+2, j+1)dp\Bigg\}.$$

(ii) If $j = a/2$, then

$$\hat{p}_{\mathscr{L}_2}(j/a) = \frac{\Big\{\mathcal{B}(a/2+2, a/2+1)(1 - \int_0^{1/2} \mathsf{Beta}(p; a/2+2, a/2+1))dp\Big\}}{\mathcal{B}(a/2+1, a/2+1)[1 - \int_0^{1/2} \mathsf{Beta}(p; a/2+1, a/2+1)dp]};$$

where $\mathcal{B}(a,b) := \int_0^1 u^{a-1}(1-u)^{b-1}du$ is the Beta function defined for $a, b > 0$; and $\mathsf{Beta}(u; a, b)$ is the pdf of a Beta random variable evaluated at $u \in (0,1)$.

Remark 5. The above Bayes estimators depend on *four* different parameters: the sampling location x, realized number of positive responses at x summarized via the majority proportion $\bar{p}(B(x))$; the batch size a and the loss function $\mathscr{L}$. Whenever necessary we denote such dependency explicitly by $\hat{p}_{\mathscr{L}}(\bar{p}(B(x)))$.

The left panel of Figure 2.2 shows the theoretical expected bias $Bias_p(\hat{p}(x)) := \mathbb{E}^B_{\theta,x}[p(x) - \hat{p}(B(x))]$ corresponding to the estimators (2.12), (2.18), (2.19) and (2.20); for $a = 250$ and $p \in (0.5, 1)$. Note that as $p \downarrow 0.5$, all procedures overestimate the true p, highlighting the difficulty to estimate $p(x)$ when $x \simeq x^*$. Of course, this issue is mitigated as batch size a increases. The procedures which best approximate p when $p \simeq 1/2$ are the *posterior mode*, $\hat{p}_{\mathscr{L}_0}$, and the empirical majority proportion $\bar{p}$. However, as the true p increases, $\hat{p}_{\mathscr{L}_0}$ underestimates p (the bias increases), whereas the bias in the empirical majority proportion decays uniformly. Conversely, both the posterior mean $\hat{p}_{\mathscr{L}_2}$ and median $\hat{p}_{\mathscr{L}_1}$ overestimate when $p(x) \downarrow 1/2$ and underestimate it when $p(x) \uparrow 1$.

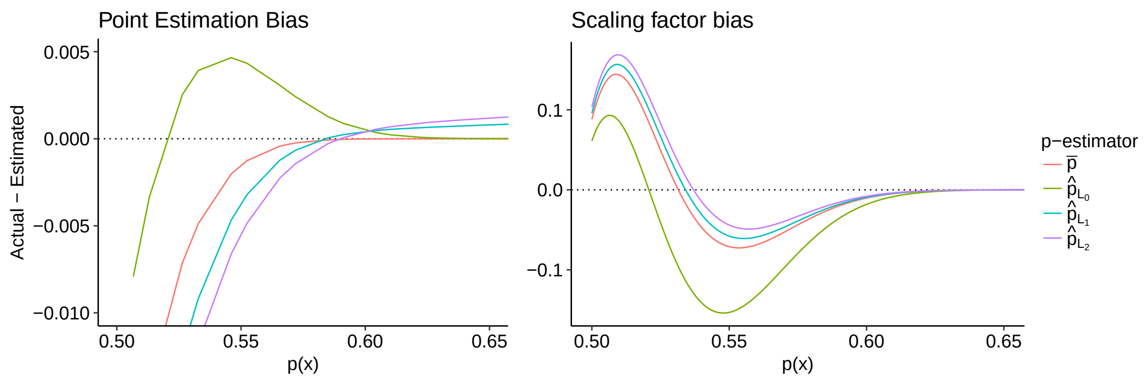

Figure 2.2: *Left:* Expected bias of $\hat{p}$-estimators with respect to the number of positive responses $B \sim \mathsf{Bin}(a, \theta)$. *Right:* Expected right scaling factor $\hat{R}^{(a)}(f_0, x, \hat{p})$ computed given $f_0 = \mathsf{Unif}(0, 1)$ and several locations $x > x^*$ so that $p(x) \in (0.50, 0.70)$ (x-axis). Both panels are for $a = 250$.

2.2.3 Bias in Knowledge States

Recall that the key component about the update f_{n+a} (obtained via the knowledge transition function Ψ) is given by the *right-scaling factor* (2.9) since it condenses all information needed in order to recover f_{n+a} given f_n. The average scaling factor integrated against the pmf of B is $\hat{R}^{(a)}(f_0, x; \hat{p}) := \mathbb{E}^B_{\theta,x}[R^{(a)}(f_0, x; B, \hat{p}(B))]$, where $B(x) \sim \mathsf{Bin}(a, \theta(x))$. The right panel of Figure 2.2 shows the expected right-scaling factor obtained given a $\mathsf{Uniform}$ prior f_0 over $(0, 1)$ and updating locations $x_1 > x^*$ labeled via their $p(x_1)$ (x-axis). Since $x_1 > x^*$, the right-scaling factor is expected to be close to zero when $p(x_1) \gg 0.5$ (since the updated f_1 would have fewer mass to the right of x_1) and conversely $\hat{R}^{(a)}(f_0, x_1) \uparrow 1$ as $p(x_1) \downarrow 0.5$ (i.e., x_1 approaches the root x^*). We observe that in the latter setting, all four statistical procedures for $\hat{p}$ tend to *overestimate* the true right-scaling factor (the expected difference $R - \hat{R}$ is negative), meaning that there is "overconfidence" that x^* is located to the right of x_1 even though in fact $p(x_1) \cong 1/2$. In particular, the two statistical procedures which seem to best resemble the true right-scaling factor when $x_1 \simeq x^*$ are the *posterior mode*, as well as the *empirical majority proportion*. Conversely, when the updating location x_1 is such that $p(x_1) > 1/2$, we see that all procedures provide an accurate description of the updated knowledge state at time $n = 1$, especially for large values of a.

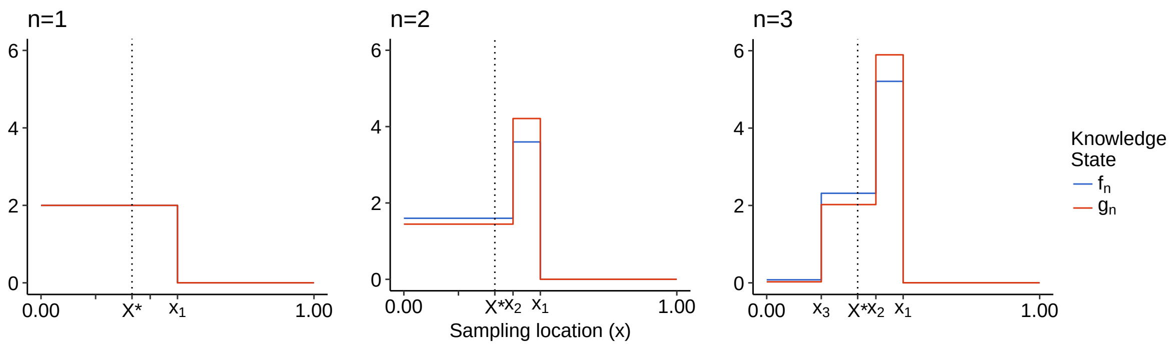

Figure 2.3: True and approximated knowledge states with the empirical proportion estimator $\bar{p}$ using three sampling locations $x_{1:3} = (0.5, 0.4, 0.2)$ and $a = 10$, using the linear function (4.1) with $x^* = 1/3$.

The approximated posterior f_{n+a} differs relative to the true posterior g_{n+a} due to the fact that f_n utilizes the estimated $\hat{p}(x_{n+1})$ whereas g_n uses the true $p(x_{n+1})$. Figure 2.3 uses the majority proportion estimator $\bar{p}$ to illustrate how the bias in $\bar{p}$ induces over/under confidence when comparing the knowledge state f_{n+a} vis-a-vis the ground truth g_{n+a}. Starting with a $f_0, g_0 \sim \mathsf{Unif}(0, 1)$ prior, we compare the true posterior g_{n+a} and its approximation f_{n+a} for $n \in \{1, 2, 3\}$ and $a = 10$, updated using the (arbitrary) locations $x_1 = 0.5$, $x_2 = 0.4$ and $x_3 = 0.2$ our running example (4.1). Note that the first two sampling locations $x_{1:2}$ are to the right of $x^* = 1/3$, whereas x_3 is leftwards of x^*.

2.2.4 Aggregation of responses

An alternative strategy for updating the knowledge state is to build a subsidiary statistic from the i.i.d. $(Y_j(x))_{j=1}^{a}$'s, whose specificity is *boosted* thanks to the batching. In other words, instead of using the a-step update $\Psi(\cdot; p, a)$ with p, we utilize a 1-step update $\Psi(\cdot; \mathscr{P}, 1)$ with an adjusted probability of correct response $\mathscr{P}$. In this case, we consider *majority-vote* statistic $\mathscr{M}(x) := 1_{\{B(x) > \lceil a/2 \rceil\}}$ [37]. Then

$$\mathscr{P}_{\mathscr{M}}(p) := \mathbb{P}_p(\mathscr{M}(x) = 1_{\{x>x^*\}}) = \sum_{j=\lceil a/2 \rceil}^{a} \binom{a}{j} p^j (1-p)^{a-j}. \tag{2.21}$$

Substituting an estimate $\hat{p}$ in (2.21) then yields $\mathscr{P}_{\mathscr{M}}(\hat{p}) = \sum_{j=\lceil a/2 \rceil}^{a} \binom{a}{j} \hat{p}^j (1-\hat{p})^{a-j}$, and the boosted update rule

$$f_{n+K} = \Psi\big(f_n, x_{n+1}, \mathscr{M}(x_{n+1}); \mathscr{P}_{\mathscr{M}}(\hat{p}(x_{n+1})), 1\big). \tag{2.22}$$

Note that since $\mathscr{M}$ only uses limited information about B, it is not sufficient for learning p. Consequently, the resulting knowledge state is not directly comparable to the full Bayesian posterior g_n; the hope is that through majority boosting we filter "noise" in B and hence mitigate the bias in $\hat{p}$.

Aggregation of Functional Responses. Assuming that the functional responses (1.1) are available, another possibility for updating the knowledge state f_n is to use the actual functional values $(Z_j(x))_{j=1}^{a}$ via the signal

$$\mathscr{S}(x) := 1_{\{\sum_{j=1}^{a} Z_j(x) > 0\}}. \tag{2.23}$$

By the CLT $\mathscr{P}_{\mathscr{S}}(h(x), \sigma(x)) := \mathbb{P}_{h,\sigma}(\mathscr{S}(x) = 1_{\{x<x^*\}}) \simeq \Phi(\sqrt{a}|h(x)|/\sigma(x))$, where $\sigma^2(x)$ is the location-dependent variance of $\epsilon(x)$. Observe that $\mathscr{P}_{\mathscr{S}}(h(x), \sigma(x))$ no longer depends on $p(x)$ but on the signal-to-noise ratio $h(x)/\sigma(x)$. A natural estimator for $\mathscr{P}_{\mathscr{S}}$ is then

$$\mathscr{P}_{\mathscr{S}}(\hat{h}_a, \hat{\sigma}_a) = \Phi(\sqrt{a}|\hat{h}_a(x)|/\hat{\sigma}_a(x)); \tag{2.24}$$

where $\hat{h}_a := \frac{1}{a}\sum_{j=1}^{a} Z_j$ and $\hat{\sigma}_a^2 := \frac{1}{a-1}\sum_{j=1}^{a}(Z_j - \hat{h}_a)^2$ are the sample mean and variance obtained for $a \geq 2$, respectively . Using the functional responses, the updated f_{n+a} is thus computed using $\mathscr{S}$ via

$$f_{n+a} = \Psi\left(f_n, x_{n+1}, \mathscr{S}(x_{n+1}); \mathscr{P}_{\mathscr{S}}(\hat{h}_a(x_{n+1}), \hat{\sigma}_a(x_{n+1})), 1\right). \tag{2.25}$$

Table 2.1: Schemes for knowledge state updating f_{n+a} based on query batches of a at location x.

Update Scheme	Sufficient Statistic	Parameters
p-estimate (2.10) using $\bar{p}$ or $\hat{p}_{\mathscr{L}}(\bar{p})$	$B = \sum_{j=1}^{a} 1_{\{Y_j=+1\}}$	p
Majority Boosting (2.22) with $\mathscr{P}_{\mathscr{M}}(\hat{p})$	$\mathscr{M} = 1_{\{B > \lceil a/2 \rceil\}}$	p
Functional Aggregation (2.25) with $\mathscr{P}_{\mathscr{S}}(\hat{h}_a, \hat{\sigma}_a)$	$\mathscr{S} = 1_{\{\sum_{j=1}^{a} Z_j > 0\}}$	h/σ

TPO Strategy. A different aggregation of Z_j's relies on hypothesis testing, specifically *statistical tests of power one* (TPO) [55]. The key idea is to use an adaptive number of replicates $a_\alpha(x)$ so as to boost the probability of correct response to level p_α, without explicitly estimating $p(x)$ [57]. Let $S(x) := \sum_{j=1}^{a} Z_j(x)$ and

$$a_\alpha(x) := \min\{k \in \mathbb{N} : |S_k(x)| \geq c_k(\alpha)\}; \tag{2.26}$$

where $(c_k(\alpha))_{k\in\mathbb{N}}$ is defined in terms of the distribution of $\epsilon(x)$ and the significance parameter $\alpha \in (0,1)$. The adaptive batch size is a_α and the resulting output is the aggregated signal which is viewed as a test statistic for inference about the positivity of the drift of the random walk $S_\cdot(x)$. The construction of $c_\cdot(\alpha)$ guarantees that $\tilde{p}(x) = \mathbb{P}(\tilde{Z}(x) = \text{sign}(x^* - x)) \geq 1 - \alpha/2$. To obtain the curved boundary $c_\cdot(\alpha)$ requires knowledge of the distribution of $Z(x)$. For example, if $Z(x) \sim \mathsf{N}(h(x), \sigma^2)$ then $c_k(\alpha) = \sigma((n+1)[\log(n+1) - 2\log\alpha])^{1/2}$.

Table 2.2 shows the average hitting time $\mathbb{E}_p[a_\alpha(x)]$ as well as its estimated standard deviation (in parentheses) for different $p(x)$ (rows) and α (columns) combinations. It illustrates that the expected batch size grows exponentially as $p(x) \downarrow 1/2$, which might be counterproductive in cases where the sampling budget is small. Indeed, instead of trying other locations, TPO will stubbornly sample the same x thousands of times.

Table 2.2: Average hitting time $\mathbb{E}[a_\alpha(x)]$ and corresponding standard deviation (in parentheses) to learn $p(x)$ using the TPO rule (2.26) with $\alpha \in \{0.05, 0.10, 0.20, 0.40\}$ for the h_1 function in (4.1) with $x^* = 1/3$. Results are based on 1,000 macro runs.

$p(x)$	$\mathbb{E}[a_{0.05}(x)]$	$\mathbb{E}[a_{0.1}(x)]$	$\mathbb{E}[a_{0.2}(x)]$	$\mathbb{E}[a_{0.4}(x)]$
0.52	4951 (3209)	4352 (3151)	3563 (2983)	2715 (2843)
0.55	692 (483)	594 (457)	456 (403)	362 (388)
0.60	159 (113)	133 (103)	105 (95)	79 (81)
0.70	34 (24)	29 (20)	23 (18)	18 (17)

2.3 Sampling Policies

Sampling is the process of selecting querying locations so that the knowledge about the root X^* can be improved. In the context of the SRFP, the challenge is that sampling close to the root yields uninformative oracle responses. More specifically, since $x \to x^*$ implies $p(x) \downarrow 1/2$, the knowledge obtained from sampling in a vicinity of x^* is minimal and the updated state f_{n+1} will change very little with relative to f_n. To resolve this challenge we investigate two classes of sampling policies that enforce *exploration* and take advantage of the full probabilistic description of the root X^* via the knowledge state f_n:

1. Information Directed Sampling (IDS): Firstly, we borrow the idea of Expected Improvement (EI), popularized in Bayesian optimization. EI constructs a one-step information gain criterion and sets x_{n+1} as the corresponding greedy maximizer. Examples of EI functions include Efficient Global Optimization (EGO) [31], Stepwise Uncertainty Reduction (SUR) [14], Expected Quantile Improvement (EQI) [44], and Integrated Mean Squared Error (IMSE) [22]. Here again we contrast the function-view strategy of emulation, which quantifies the learning of $h(\cdot)$, with the root-view strategy that quantifies learning of X^*. For the former, despite some progress on building EI measures for the level-sets and graph of $h(\cdot)$ [14, 3], these

metrics remain complex. In our view this is a fundamental conceptual hurdle arising from the mismatch between the large model space for h, and the much simpler derived quantity, i.e., the root x^*, to be learned. Moreover, to our knowledge, existing emulators for h have few tools to take advantage of the specific structure that arises in root-finding, first and foremost the fact that there is a unique x^*. In our running example of a GP emulator introduced in Chapter 1, it is very challenging to control the behavior of the level-set; see for example the ongoing efforts to build tractable monotone GP models [49]. By explicitly targeting X^* we seek the most direct path to developing effective rules for the sequential design $x_{1:n}$.

2. Posterior Quantile Sampling: Secondly, we propose another class of sampling policies which do not make explicit use of a data acquisition function such as the strategies above, but rather use solely the state variable f_n (and perhaps additional randomization) in order to select new samples. Namely, sampling locations are *quantiles* of f_n, i.e., $x_{n+1} := F_n^{-1}(q_n)$, where $F_n(\cdot)$ is the CDF of f_n and $q_n \in (0, 1)$ are the sampling quantiles, which can either be randomized or fixed. For instance, $q_n \equiv 1/2 \ \forall n \geq 1$ corresponds to the classic PBA median-sampling strategy. At the other extreme, taking $q_n \sim \mathsf{Unif}(0, 1)$ – which closely resembles Thompson Sampling [53]; new locations are chosen according to the current likelihood of X^*.

Both deterministic and randomized versions of each class are analyzed in Section 2.3.1 and Section 2.3.2, respectively.

2.3.1 Information Directed Sampling

This sampling strategy is driven by the notion of an *acquisition function* which quantifies expected information gain from a new oracle query. A common information-theoretic approach is to maximize the KL divergence between the current knowledge state f_n and its expected update f_{n+1} conditional on sampling at a given x. In the context of PBA for stochastic root-finding, the relative entropy between f_n and f_{n+1} can be interpreted as the mutual information between oracle $Y(\cdot)$ and X^*. As mentioned above, this idea is similar to entropy-maximizing EI strategies (see e.g. [27]) and leverages the explicit form of KL-divergence when $p(x)$ is known.

Lemma 2.3.1 (Expected KL divergence between g_n and g_{n+1}). [28] Let

$$D(g_{n+1}; g_n) := \int_0^1 \log_2 \left(\frac{g_{n+1}(u)}{g_n(u)} \right) \cdot g_n(u) du \tag{2.27}$$

be the KL divergence of g_n and g_{n+1} (obtained when g_n is updated at a given location x). The expected KL divergence $\mathcal{I}(x, g_n; p(x)) := \mathbb{E}_p^Y[D(g_{n+1}; g_n)|g_n, x, p(x)]$ (averaging against the pdf of $Y_{n+1}(x) \in \{-1, 1\}$) between g_{n+1} and g_n is given by

$$\begin{aligned}\mathcal{I}(x, g_n; p(x)) := &-\gamma_n(x; p(x)) \log \gamma_n(x; p(x)) - [1 - \gamma_n(x; p(x))] \log[1 - \gamma_n(x; p(x))] \\ &+ p(x) \log p(x) + (1 - p(x)) \log(1 - p(x)),\end{aligned} \tag{2.28}$$

where $\gamma_n(x; p(x))$ and $p(x)$ are given by (2.4) and (1.4), respectively.

A greedy IDS strategy then myopically maximizes the information gain (2.28). As shown in [59], this myopic sampling rule is in fact *optimal* for the global problem of reducing the expected posterior entropy of g_N when $p(\cdot)$ is a known constant. This approach has also been adopted in [28] for similar problems appearing in computer vision or, more recently in [52] for on-line optimization problems.

In analogy to the Information Directed Sampling (IDS) criterion (2.28), we introduce the *batched* expected KL divergence between g_n and $g_{n+a} := \Psi(g_n, x, B(x); p(x), a)$ for a given batch amount $a \geq 1$:

Theorem 2.3.2. Let $x \in (0, 1)$ be an arbitrary updating location and g_n the current knowledge state. The batched information criterion, that is, the expected KL divergence

$\mathcal{I}(x, g_n; p(x), a) := \mathbb{E}_p^B[D(g_{n+1}; g_n)|g_n, x, p(x), a]$ (averaging with respect the pdf of $B(x)$) between g_{n+a} and g_n is given by

$$\mathcal{I}(x, g_n; p(x), a) := \mathbb{E}\left[\log_2\left(\frac{(1-p(x))^B p(x)^{a-B}}{c_n(x, B)}\right)\right] G_n(x) \tag{2.29a}$$

$$+\,\mathbb{E}\left[\log_2\left(\frac{p(x)^{\tilde{B}}(1-p(x))^{a-\tilde{B}}}{c_n(x, \tilde{B})}\right)\right](1-G_n(x)) \tag{2.29b}$$

where (2.29a) and (2.29b) are taken with respect to $B \sim \mathsf{Bin}(a, 1-\theta(x))$ and $\tilde{B} \sim \mathsf{Bin}(a, \theta(x))$, respectively; and $c_n(x, B)$ is the normalizing constant of the updating (2.6) and G_n is the CDF of g_n.

Proof of (2.29). The expected KL divergence between the current state g_n and the updated state $g_{n+a} := \Psi(g_n, x, B; p(x), a)$ at x is obtained by averaging the KL divergence of g_{n+a} with respect to g_n, $D(g_{n+a}; g_n)$, with respect to all possible values of $B_{n+1} \in \{0, 1, \ldots, a\}$:

$$\mathcal{I}(x, g_n; p(x), a) := \mathbb{E}_p^B\left[\int_0^1 \log_2\left(\frac{g_n(u|B, x, a)}{g_n(u)}\right) g_n(u)du\right]$$

Since $\mathbb{P}_p(B = j|a, x, u) := \mathsf{Bin}(j; a, 1-\theta(x))1_{\{u\leq x\}} + \mathsf{Bin}(j; a, \theta(x))1_{\{u>x\}}$ and for $0 < u < x$ we have that $g_n(u|B, x, a) := [(1-p(x))^B p(x)^{a-B}]g_n(u)/c_n(x, B)$ (similarly for $x < u < 1$) so we end up with

$$\mathcal{I}(x, g_n; p(x), a) = \int_0^x \mathbb{E}\left[\log_2\left(\frac{(1-p(x))^B p(x)^{a-B}}{c_n(x, B)}\right)\right] g_n(u)du$$
$$+ \int_x^1 \mathbb{E}\left[\log_2\left(\frac{p(x)^{\tilde{B}}(1-p(x))^{a-\tilde{B}}}{c_n(x, \tilde{B})}\right)\right] g_n(u)du$$

which simplifies to (2.29). □

Given the acquisition function (2.29), the next sampling location is its greedy maximizer

$$x_{n+1} := \arg\sup_{x\in(0,1)} \mathcal{I}(x, f_n; p(x), a).$$

In order to further illustrate the relationship between the knowledge state g_n and the batched information-criterion $\mathcal{I}(\cdot, g_n; p(\cdot), a)$, Figure 2.4 shows the knowledge state updating using a replication amount of $a = 100$ for $n = 0, \ldots, 9$ starting with $g_0 \equiv \mathsf{Unif}(0,1)$ prior on X^*, and implementing our running example h_1 with root at $x^* = 1/3$. The dotted vertical lines correspond to the maximizer x_{n+1} of the information criterion. We notice that sampling at x_{n+1} with constant batch size $a = 100$ makes g_n to concentrate rapidly around the true root x^*, as well as the batched information criterion has typically 2 major maxima, along with a global minimum at x^* (sampling at the root provides no knowledge about its location). Moreover, the third column of Figure 2.4 shows the information criterion $x \mapsto \mathcal{I}(x, g_9; p(x), a_{10})$ for different replication values $a_{10} \in \{1, 10, 5, 100, 250\}$ given g_9 (obtained after updating g_0 with $a = 100$ at ten sampling locations). It can be seen that as the batch size increases, the information gain increases, as well as that the maximizer (dotted lines) of the associated information value does not change significantly across the different replication values (which is in part due to the posterior g_9 is already concentrated in the region where x_{n+1} is selected, as seen on the top plot in the third column).

To implement the IDS approach, two modifications are necessary. First, similar to Section 2.2, given the majority response $\bar{p}(x)$ we can obtain *a posteriori* plug-in version of (2.28) by replacing $p(x)$ by its estimate $\hat{p}(x)$, as well as the true posterior g_n by its approximation f_n, that is, $\mathcal{I}(x, f_n; \hat{p}, a)$. Second, the maximization over x can only be done *ad hoc*, since under the current estimation paradigm computing the information gain $\mathcal{I}$ can only be applied after querying the oracle a-times at x. As a work-around, we carry out the optimization over a discrete candidate set $\mathcal{S}_M(f_n) := \tilde{x}^{(n)}_{1:M}$: one picks $M \geq 2$, candidate locations $\tilde{x}^{(n)}_{1:M}$ using f_n, queries the oracle a-times at each $\tilde{x}^{(n)}_i$ and

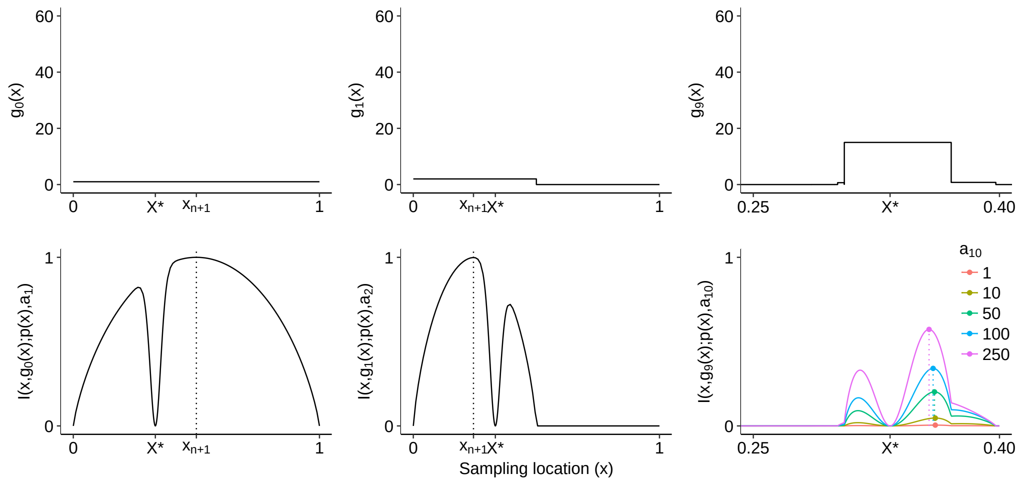

Figure 2.4: Data acquisition procedure using the batched information criterion $\mathcal{I}(\cdot, g_n; p(\cdot), a)$ starting with a Uniform prior g_0 and using the linear test function (4.1) with $x^* = 1/3$. The first row shows the true Bayesian updating g_n for $n \in \{0, 1, 9\}$ and the second row depicts the corresponding information gain function along with its maximizer x_{n+1} (vertical dotted lines). The right-bottom plot, shows the information criterion for several replication sizes $a_{10} \in \{1, 10, 50, 100, 250\}$ with the corresponding maximizers of the information criterion x_{10} (vertical dotted lines) given the updated knowledge state g_9 obtained updating g_0 with fixed size of $a = 100$ during at ten locations.

finally updates f_n at the maximizer of this criterion:

$$x_{n+1}^{\text{IDS}} := \underset{\tilde{x}_i \in \mathcal{S}_M(f_n)}{\arg\max}\, \mathcal{I}(\tilde{x}_i, f_n; \hat{p}(\tilde{x}_i), a) \tag{2.30}$$

To construct candidate sets $\mathcal{S}_M(f_n)$ we rely on the quantiles of f_n:

Deterministic IDS: The test locations $\tilde{x}_{1:M}^{(n)}$ are *fixed* posterior quantiles of f_n, i.e.,

$$\tilde{x}_i^{(n)} := F_n^{-1}(q_i). \tag{2.31}$$

Randomized IDS: The test locations are randomly chosen posterior quantiles of f_n:

$$\tilde{x}_i^{(n)} = F_n^{-1}(q_{i,n}), \qquad q_{i,n} \sim \mathsf{Unif}(0, 1). \tag{2.32}$$

Note that at each iteration n, a total of $a \times M$ queries are made (a at each $\tilde{x}_i$), of which only a are used for actual updating to f_{n+1}. Therefore, after N updates used for f_N, total wall-clock time is $T = N \times a \times M$. To minimize this inefficiency in our experiments we use $M = 2$, so that (2.30) is reduced to comparing information gain at two chosen locations $\tilde{x}^{(n)}_{1:2}$.

2.3.2 Posterior Quantile Sampling

The message of classical PBA is that one should sample at the *median* of the knowledge state g_n. However, this no longer holds when the $p(x)$ depends on the location x since $p(x) \to 1/2$ as $x \to x^*$. In fact, we show in our numeric examples that the performance of this policy does as bad as sampling uniformly over the input space in terms of uncertainty minimization. Intuitively, sampling at the median is not suitable since after a few iterations the median is located too close to the root and therefore minimal information gain is obtained (this was already pointed out in [57]).

Thus, other *posterior quantiles* are explored, taking $x_{n+1} = F_n^{-1}(q_n)$. On the one hand, quantile sampling places samples where most of the posterior mass of f_n is located (which after a few iterations will be concentrated around x^*), allowing to gradually focus on the neighborhood of x^*. On the other hand, quantile sampling is based solely on the knowledge state variable f_n and can be used *a priori* without yet having an estimate of $p(x_{n+1})$.

Systematic Quantile Sampling. Locations are selected by *systematically* iterating over $M \geq 2$ posterior quantiles $\check{q}_{0:M-1}$, fixed a priori. Then, in the n-th iteration, the next design point is

$$x^{\mathrm{SQS}}_{n+1} = F_n^{-1}(\check{q}_{(n \mod M)}). \tag{2.33}$$

We remark that the precise ordering of $\check{q}_n$'s will affect the results of Syst-Q. To balance the trade-off between exploration and exploitation we look at quantiles that are away from the median $q_n = 0.5$. Considering the shape of (2.28), a sensible rule is to consider the quartiles of f_n, i.e., $\check{q} \in \{0.25, 0.75\}$.

Randomized Quantile Sampling. The next design point is a randomly chosen quantile of the posterior distribution f_n:

$$x_{n+1}^{\mathrm{RQS}} = F_n^{-1}(U_{n+1}), \text{ where } U_{n+1} \sim \mathsf{Unif}(0,1). \tag{2.34}$$

The policy (2.34) can be interpreted as sampling at a location $X_n \sim f_n$, i.e., sampling based on the posterior distribution of X^*.

2.3.3 Batch size a

An essential tuning parameter in Algorithm 1 is the batch size $a \geq 1$ needed to learn $p(x)$ at each updating location x. Recall that the total number of learning iterations is $N := \lfloor T/a \rfloor$. Therefore, for a fixed budget T, the batch size a controls the balance between the learning of X^* and $p(\cdot)$. When a is small (thus N large), the algorithm is exploring many sampling locations to learn X^*. When a is large, the algorithm exploits the oracle in order to estimate $p(x)$ locally with high accuracy. As a result, for large values of a the estimated $\hat{p}(x_{1:n})$ is likely to be close to $p(x_{1:n})$ and therefore f_N resembles the true posterior g_N. Consequently, the probabilistic representation about X^* would be excellent (measured, for instance, in terms of the f_n-coverage). However this would come at the cost of sampling at very few sampling locations $x_{1:N}$, and the resulting limited knowledge about X^* would lead to potentially larger residuals $|\hat{x}_N - X^*|$. In contrast, for a small, the estimated $\hat{p}(x_{1:n})$ is highly biased and f_n will significantly differ from the true posterior g_n causing f_n to collapse to regions where X^* may not be located. As we show in our numerical examples, the latter case turns out to be more problematic. In particular we observe that moderately large $a \in [100, 500]$'s are necessary to obtain a reasonable f_N; otherwise the bias accumulates quickly.

Chapter 3

Blending Spatial Modeling and Probabilistic Bisection

3.1 Introduction

The Generalized PBA (G-PBA) that we developed in Chapter 2 extends the classical PBA by using the observed data to construct a point estimate for $p(x)$ as well as to simultaneously learn the root location X^*. The proposed estimators $\hat{p}(x_{n+1})$ under the G-PBA paradigm were constructed locally at x_{n+1} and did not use information from previous locations $x_{1:n}$. As such, they were robust to arbitrary specification of $p(\cdot)$ and could be viewed as making minimal assumptions about the oracle.

Surrogate modeling. In this Chapter, we build a *spatial* G-PBA by modeling the entire oracle accuracy $x \mapsto p(x)$ using a *surrogate*. The surrogate relies on tow main premises: (i) the fundamental assumption of symmetrical noise in the oracle responses (1.1), which allows to translate estimating oracle specificity $p(\cdot)$ to learning the probability of a positive response $\theta(\cdot)$ (as expressed in Equation (2.11)) and (ii) the smoothness of $x \mapsto \theta(x)$, implying that $p(x)$ and $p(x')$ should be similar when x and

x' are deemed close to each other. The spatial structure is natural in the root-finding context and provides two key benefits. On the one hand, it improves estimation of a given $p(x)$ through leveraging the knowledge acquired at previous sampling locations $x_{1:n}$. On the other hand, it enables better sampling strategies by furnishing a prediction $\hat{p}(x)$ at arbitrary x, specifically unsampled ones. In contrast, in G-PBA, $\hat{p}(x_{n+1})$ was only available *a posteriori* after sampling at x_{n+1}.

Our strategy blends the root-centric framework of PBA and the function-centric paradigm of response surface modeling (RSM). Indeed, a further alternative for solving the SRFP would be to learn the entire $\theta(\cdot)$ and then take $\hat{x} = \theta^{-1}(0.5)$ since $h(x^*) = 0 \Rightarrow \theta(x^*) = 0.5 = \mathbb{P}(\epsilon(x^*) > 0)$. Thus, stochastic root-finding can be recast as a (localized) learning task, namely contour-finding for $\theta(\cdot)$ at the level $h = 0.5$. Strategies similar to Bayesian optimization can then be employed to efficiently target this objective during sequential design. Nevertheless, several challenges are encountered with such an approach that are circumvented in PBA. First, a major feature of PBA is a full uncertainty quantification around $\hat{x}$: the algorithm provides the entire posterior distribution f_n of X^* conditional on the data. Typical RSM models return only point estimates (or pointwise credible intervals) of $\theta(x)$; the latter are difficult to "invert" into uncertainty about $\theta^{-1}(0.5)$ [3]. Second, existing design approaches for contour-learning are developed only for simple models (e.g. with zero or constant observation noise), and their performance in a complex stochastic setting like ours is poorly known. In contrast, the PBA explicitly targets the goal of reducing uncertainty of X^*. PBA moreover exploits the structural knowledge of a *unique* root to speed up estimation, an option that is not available in contour-finding. Third, contour-finding usually assumes continuous response, and nontrivial modifications (essentially "logistic" contour-finding) are necessary to handle binary $Y_n \in \{-1, 1\}$. In contrast, PBA intrinsically is designed for binomial responses.

Given the above discussion, we design a hybrid algorithm that borrows the best of both worlds. We leverage the smoothness of h that implies spatial dependence in $\theta(\cdot)$ and hence accelerates learning the oracle. At the same time, we employ the paradigm of PBA to construct the knowledge state f_n (a pseudo-posterior of X^*) that is the primary driver of sampling decisions and uncertainty quantification. For the RSM component, we rely on two key concepts. First, we investigate non-parametric architectures that have the flexibility to consistently learn the entire response $\theta(\cdot)$ and to handle non-uniform experiment designs. Specifically, we consider Gaussian Process (GP) models, as well as splines, kernel and polynomial regression. Second, we apply *batched sampling* that significantly lowers the computational overhead of surrogate construction and improves the learning of $\theta(\cdot)$. On the latter point, our work has independent interest in terms of applications of GPs. To this end, we provide several new results that to our knowledge are not available in existing literature. This include look-ahead variance formula for logistic GP, and novel active learning heuristics for logistic GPs.

Statistical modeling framework. To infer the oracle properties, we employ *logistic regression* which represents the probability of observing a positive response $\theta(x) = \mathbb{E}[1_{Z_n(x)>0}]$ via a *latent* process $\varphi(x) := \text{logit}(\theta(x))$. We remark that other link functions can be used but as we show in the sections below, the canonical Bernoulli link (i.e., logit link) is used to derive adaptive replication schemes when GPs are considered for $\varphi(\cdot)$. To achieve maximum flexibility, especially critical in our setup where the quality of the entire $x \mapsto \theta(x)$ is needed for good performance, we consider non-parametric models for $\varphi(\cdot)$. Specifically, we seek regression approaches which are able to refine the regression curve in regions where more inputs are placed (namely close to the root), but at the same time give a good global fit. Two appropriate examples we investigate are Kernel logistic regression (KLR) and Spline logistic regression (SLR). KLR behaves similarly to Support Vector Machines: data inputs are mapped to a space spanned by positive definite kernel functions, and it also happens that the loss function optimized to obtain the estimated regression curve in KLR is similar to the one in SVM for the two-class problem [62]. SLR uses the well-known cubic splines set of basis functions which are piecewise cubic polynomials defined over a pre-specified set of knot locations.

The rest this Chapter is organized as follows. In Section 3.2 we describe the model methodology used to provide an spatial estimate for $p(\cdot)$. Section 3.2.2, describes an adaptive batching/replication scheme in order to determine the number of replicates a_{n+1} given an estimated surrogate model. In Section 3.3, we describe how surrogate modeling is blended with the G-PBA and state the Spatial G-PBA in order to obtain an enhanced version of G-PBA sampling policies.

3.2 Knowledge States

In this Section we extend the estimation procedures used in the G-PBA setting developed in Chapter 2 by introducing a *surrogate* model on $p(\cdot)$ which is built upon the history of batched responses. Recall that for learning the oracle, the G-PBA produces a *local* estimate $\hat{p}(x_n)$ depending exclusively on the information observed at x_n via the use of replications. Thus, the oracle is called $a_n \geq 1$ times at the fixed sampling location x_n with the responses aggregated based on the total number $B_n \in \{0, 1, \ldots, a_n\}$ of positive signs observed at x_n:

$$B_n(x_n) := \sum_{j=1}^{a_n} 1_{\{Y_j(x_n)=+1\}}. \tag{3.1}$$

Remark 6. In contrast to the notation used in Chapter 2 we add the subscript n to the binomial batched response $B_n(\cdot)$. Moreover, the response $B_n(\cdot)$ depends intrinsically also on the replication amount a_n as well as on the sampling location x_n.

In analogy to the idea employed for the local estimators, we translate the problem of learning $p(x)$ into learning

$$\theta(x) := \mathbb{P}(Y(x) = +1), \tag{3.2}$$

the probability of observing a *positive* oracle response; and then produce a plug-in estimator for $p(\cdot)$ based on the fitted surrogate $\hat{\theta}_n$:

$$\hat{p}_n(x) := \max\{\hat{\theta}_n(x_n), 1 - \hat{\theta}_n(x)\}, \quad \forall x \in (0,1).$$

To obtain $\hat{\theta}_n(\cdot)$ we borrow information from previous sampling locations by regressing $B_{1:n} := (B_1(x_1), \dots, B_n(x_n))$ against the locations $x_{1:n}$, linking x_n to $\theta(x_i)$ via the canonical Bernoulli link function:

$$\log\left(\frac{\theta(x_i)}{1 - \theta(x_i)}\right) = \varphi(x_i), \quad i = 1, \dots, n, \tag{3.3}$$

based on a *surrogate* model $x \mapsto \varphi(x)$.

Under this setting, we consider two families for $\varphi(\cdot)$:

(A) Gaussian random field (GRF) modeling approach (also known as Gaussian process modeling, [48]) that takes φ as a latent Gaussian process (GP) and outputs the posterior distribution $p(\varphi_*|\mathcal{D}_n)$ conditional on the data $\mathcal{D}_n := (B_{1:n}, a_{1:n}, x_{1:n})$;

(B) a linear additive model that assumes that φ is an element of a linear space spanned by a collection of basis functions, i.e., $\varphi(x) = \sum_{j=1}^{p} \beta_j \phi_j(x)$, with the coefficients $\boldsymbol{\beta} := (\beta_1, \dots, \beta_p)$ fitted by, for example, penalized MLE.

Both modeling schemes (A) and (B) are mathematical approximations (metamodels) capable to model the relation $x \mapsto \theta(x)$. *Metamodels* are practically useful since they can be built upon available observations and updated when new data is assimilated. They can also be used to guide simulator evaluations more efficiently [51].

In line with the Bayesian updating procedure that introduced in Section 2.2 in Chapter 2, we carry out knowledge updating via the knowledge transition function Ψ:

$$f_{T_n + a_{n+1}} = \Psi(f_{T_n}, x_{n+1}, B_{n+1}; a_{n+1}, \hat{p}_{n+1}); \tag{3.4}$$

which maps the current knowledge state variable f_{T_n} to $f_{T_n+a_{n+1}}$ for a fixed batch size $a_{n+1} \geq 1$ and sampling location x_{n+1}. Here, $T_n := \sum_{i=1}^{n} a_i$ is the total number of oracle evaluations (i.e., *wall-clock time*) and the batch size a_{n+1} is not necessarily fixed along sampling locations. In fact, note that thanks to the surrogate $\hat{\theta}_n$ the batch size a_{n+1} may be determined *adaptively* depending on the sampling location x_{n+1} at which the current state is updated.

Below we introduce the surrogate families (A) and (B).

3.2.1 Binomial Gaussian Processes

GPs can conveniently be used to specify prior distributions for Bayesian inference in the regression context. In this case, the responses are seen as a realization of a random process whose finite dimensional distribution (fdd) follows a Multivariate Normal (MVN) distribution and whose spatial dependency is described by a (stationary) covariance function. In the case of regression with Gaussian noise, inference can be done simply in closed form, since the posterior corresponds also to a GP for a given election of covariance kernel.

Binomial Gaussian processes (B-GPs) (aka GP classification) arise naturally in the context latent variable regression. In this case, it is assumed that the binary responses are the realized value of a latent GP in which only the its sign (positive/negative) is observed. Since the data likelihood no longer corresponds to a Gaussian one, exact inference is analytically intractable and therefore approximations to the predictive posterior must be conducted. One route summarized in [42] is based on approximating the non-Gaussian posterior with a tractable Gaussian distribution. Some of the most common instances of such schemes are the Laplace Approximation [60] and Expectation Propagation [40]. Applications of such methods include: sequence annotation [1] or prostate cancer prediction [15]. Moreover, GPs are advantageous for addressing the sequential design component in order to manage the sample budget for calling the oracle $Z(x)$. In another related application, binomial GPs were used for learning an approximate globally optimal closed-loop policy in the context of approximated dynamic programming (ADP) [17]. Finally, the GP paradigm also facilitates computing the number of replicates a_{n+1} at location x_{n+1} so that the predictive uncertainty at this new location is reduced compared to the current estimate before the new sample is acquired [34, 8]. In fact, we use the latter ideas (in conjunction with the LA for binomial GPs) to give an approximate procedure for determining the number of replicates a_{n+1} in order to reduce the predictive variance of the latent GP. We now assume that the surrogate φ in (3.3) is drawn from a GP prior, $\varphi \sim \mathsf{GP}(0, \kappa_\vartheta(\cdot,\cdot))$, characterized by a *covariance kernel* function $\kappa_\vartheta(\cdot,\cdot)$ and parameterized by a vector of *hyperparameters* $\vartheta \equiv (\tau^2, l)$. For instance, one the most commonly used kernel is the 5/2-Matérn family,

$$\kappa_\vartheta(x_i, x_j) := \tau^2 \left[1 + \sqrt{5}r/l + 5r^2/(3l^2)\right] e^{-\sqrt{5}r/l} \quad r := |x_i - x_j|; \tag{3.5}$$

where $\tau^2 \geq 0$ is the intrinsic GP variance, and $l > 0$ is the length-scale, which governs how fast the correlation decreases as the distance r between inputs increases.

Binomial GPs as latent variable models. For fixed hyper-parameter ϑ, the joint distribution of the vector $\varphi_{1:n} := (\varphi_1(x_1), \dots, \varphi_n(x_n))$ is a MVN

$$\varphi_{1:n} \sim \mathsf{N}(\mathbf{0}, \boldsymbol{K}_n), \tag{3.6}$$

where $\mathbb{E}[\varphi_{1:n}|x_{1:n}] = \mathbf{0}$ is the mean vector and $\boldsymbol{K}_n \equiv Cov(\varphi_{1:n}|x_{1:n})$ is the covariance matrix with entries $\kappa_\vartheta(x_i, x_j) := Cov(\varphi_i, \varphi_j|x_{i,j})$ for all $i, j = 1, \dots, n$. Inference of $\theta(\cdot)$ is conducted in two stages. First, we compute the posterior distribution of the vector $\varphi_{1:n}$ given the training data $\mathcal{D}_n$,

$$p(\varphi_{1:n}|\mathcal{D}_n) \propto p(B_{1:n}|\varphi_{1:n}, a_{1:n})p(\varphi_{1:n}); \tag{3.7}$$

which is proportional to the binomial data likelihood $p(B_{1:n}|\varphi_{1:n}, a_{1:n})$ times the MVN prior $p(\varphi_{1:n})$ given by (3.6). Second, the posterior predictive distribution $\varphi_* \equiv \varphi_*(x)$ at a location $x \in (0, 1)$ is

$$p(\varphi_*|\mathcal{D}_n) := \int p(\varphi_*, \tilde{\varphi}_{1:n}|\mathcal{D}_n, x)d\tilde{\varphi}_{1:n}, \tag{3.8}$$

which is calculated by marginalizing the distribution of φ_* over the joint posterior distribution of $(\varphi_{1:n}, \varphi_*)$ given by (3.7). Finally, the predicted $\hat{\theta}^{GP}(x)$ is produced by averaging the inverse link function with respect to (3.8); i.e.,

$$\hat{\theta}^{GP}_n(x) := \int (1 + e^{-\varphi_*})^{-1} \cdot p(\varphi_*|\mathcal{D}_n)d\varphi_*.$$

Remark 7. Following the classical inference paradigm for binomial regression we assume that $\theta(\varphi(x_i))$ is related to the random variable $\varphi(x_i)$ via the canonical *logistic* link function (3.3). Although other link functions can be entertained (such as the probit link), we use the logistic one since this link is used to obtain closed-form expressions for adaptive replication (see Lemma 3.2.1 in Section 3.2.2).

The main challenge in computing the joint posterior (3.7) is that the MVN prior over $\varphi_{1:n}$ does not correspond to a conjugate prior for the Binomial likelihood, so either analytic approximations of integrals or solutions based on MCMC sampling are required. A commonly used method is to approximate the non-Gaussian posterior $p(\varphi_{1:n}|\mathcal{D}_n)$ with a Gaussian one via Laplace Approximation (LA).

Laplace Approximation. The Laplace method is constructed from the second order Taylor expansion of the *score function*, $\c{L}(\varphi_{1:n}) := \log p(\varphi_{1:n}|\mathcal{D}_n)$, around its mode:

$$\hat{\boldsymbol{\varphi}}_n = \arg\max_{\boldsymbol{\varphi}_n} p(\boldsymbol{\varphi}_n|\mathcal{D}_n).$$

In Appendix A.1 we show that this method yields a MVN approximation:

$$p(\cdot|\mathcal{D}_n) \simeq q(\cdot|\mathcal{D}_n, \hat{\boldsymbol{\varphi}}_n) = \mathsf{N}(\cdot; \hat{\boldsymbol{\varphi}}_n, (\boldsymbol{K}_n^{-1} + \hat{\boldsymbol{W}}_n)^{-1}), \tag{3.9}$$

where

$$\hat{\boldsymbol{\varphi}}_n := (\hat{\varphi}_{1;n}, \ldots, \hat{\varphi}_{n;n}) \tag{3.10}$$

is found numerically via Newton-Raphson using the training data $\mathcal{D}_n$; and $\hat{\boldsymbol{W}}_n$ is the Fisher Information matrix of the binomial (negative) log-likelihood. Importantly, if the canonical link is used, then the i-th entry of $\hat{\boldsymbol{W}}_n$ corresponds to the variance of the binomial response B_i at x_i:

Lemma 3.2.1. Under the Bernoulli link function (3.3), the Hessian matrix $\boldsymbol{W}_n(\varphi_{1:n}) = -\Delta l(\varphi_{1:n})$ (in the latent GP values $\varphi_{1:n}$) of negative the log-binomial likelihood, $l(\varphi_{1:n}) := \log p(B_{1:n}|a_{1:n}, \varphi_{1:n})$, is given by

$$w_{ij}(\varphi_j) = \begin{cases} a_i\theta(\varphi_i)(1-\theta(\varphi_i)), & i = j, \\ 0 & i \neq j, \quad \text{for } i, j = 1, \ldots, n. \end{cases} \tag{3.11}$$

Hence, we have that $\hat{\boldsymbol{W}}_n = \text{diag}(\hat{w}_{1;n}, \ldots, \hat{w}_{n;n})$; where $\hat{w}_{i;n} := a_i\theta(\hat{\varphi}_{i;n})(1-\theta(\hat{\varphi}_{i;n}))$ are the entries (3.11) evaluated at the estimated posterior mode (3.10). Having found the joint (3.9), the (approximated) predictive posterior density $\varphi_* \sim \mathsf{N}(m_n(x), s_n^2(x))$ is also Gaussian with mean $m_n(x) \equiv m_n(x; \hat{\boldsymbol{\varphi}}_n)$ and posterior variance $s_n^2(x) \equiv s_n^2(x; \hat{\boldsymbol{\varphi}}_n)$:

$$m_n(x) := \boldsymbol{K}_n^T \boldsymbol{K}_n^{-1} \hat{\boldsymbol{\varphi}}_n; \tag{3.12a}$$

$$s_n^2(x) := \boldsymbol{\kappa}_n^T (\boldsymbol{K}_n + \hat{\boldsymbol{W}}_n^{-1})^{-1} \boldsymbol{\kappa}_n, \tag{3.12b}$$

where $\boldsymbol{\kappa}_n := (\kappa(x, x_1), \ldots, \kappa(x, x_n))^T$ is the $n \times 1$ vector of covariances between φ_* and $\varphi_{1:n}$. The resulting point estimate for $\theta(x)$ is thus

$$\hat{\theta}_n(x) := \int (1 + e^{-\varphi_*})^{-1} \mathsf{N}(\varphi_*; m_n(x), s_n^2(x)) d\varphi_*, \quad x \in (0, 1). \tag{3.13}$$

Numerically, $\hat{\theta}_n(x)$ is obtained by approximating (3.13) via a quadrature method. In particular we use `integrate()` which is part of the core distribution of `R` and relies on the Gauss-Kronrod quadrature method [46].

Hyper-parameter estimation. The above model specification is valid for fixed hyperparameters ϑ. To optimize the latter, we consider a maximum a posteriori estimate (MAP), $\hat{\vartheta} := \arg\max_{\vartheta}\{\log q(\mathcal{D}_n|\vartheta) + \log q_0(\vartheta)\}$ based on a prior $q_0(\cdot)$. In order to obtain $\hat{\vartheta}$ we use the package `GPstuff` [56], which uses interleaved numerical optimization: at iteration m given $\hat{\vartheta}^{(m)}$, evaluate the covariance matrix $\boldsymbol{K}_n(\hat{\vartheta}^{(m)}) = (\kappa_{\hat{\vartheta}^{(m)}}(x_i, x_j))_{i,j=1}^n$ and so estimate the mode $\hat{\boldsymbol{\varphi}}_n^{(m)}$; then fix $\hat{\boldsymbol{\varphi}}_n^{(m)}$ and find $\hat{\vartheta}^{(m+1)} = \arg\max_{\vartheta} \log q(\mathcal{D}_n|\vartheta, \hat{\boldsymbol{\varphi}}_n^{(m)}) + \log q_0(\vartheta)$, where $q_0(\cdot)$ is the prior and $q(\mathcal{D}_n|\vartheta, \hat{\boldsymbol{\varphi}}_n^{(m)})$ is the data marginal log-likelihood,

$$\log q(\mathcal{D}_n|\vartheta, \hat{\boldsymbol{\varphi}}_n) = -\frac{1}{2}\hat{\boldsymbol{\varphi}}_n^T \boldsymbol{K}_n(\vartheta)^{-1}\hat{\boldsymbol{\varphi}}_n + \log p(B_{1:n}|a_{1:n}, \hat{\boldsymbol{\varphi}}_n) - \frac{1}{2}\log\{|\boldsymbol{K}_n(\vartheta)|\cdot|\boldsymbol{K}_n(\vartheta)^{-1} + \hat{\boldsymbol{W}}_n|\},$$

which is available in closed-form [48].

3.2.2 Adaptive Batching using the Posterior GP Variance

The posterior variance s_n of the surrogate quantifies the quality of learning the latent GP. It can be used to guide sampling decisions via the associated information gain regarding $\varphi(\cdot)$. This is achieved by considering the look-ahead $s_{n+1}(\cdot)$ conditional on sampling at x_{n+1}. First, we show that for binomial GPs the posterior predictive variance *does* depend on $B_{n+1}(x_{n+1})$ (i.e., the binomial response collected a posteriori at location x_{n+1}). Specifically, Equation (3.14) expresses the fact that $s^2_{n+1}(x_{n+1})$ depends on the entire $\hat{\boldsymbol{\varphi}}_{n+1}$ (computed based on $\mathcal{D}_{n+1}$).

Theorem 3.2.2. The look-ahead variance $s^2_{n+1}(x_{n+1})$ evaluated at a new location x_{n+1} under the Laplace approximation (3.12a) and (3.12b) is given by

$$s^2_{n+1}(x_{n+1}) = \left(\frac{1}{s^2_n(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n,n+1})} + \frac{1}{[a_{n+1}\cdot\theta(\hat{\varphi}_{n+1,n+1})(1-\theta(\hat{\varphi}_{n+1,n+1}))]^{-1}}\right)^{-1}. \quad (3.14)$$

To estimate $s^2_{n+1}(x_{n+1})$ using only information available at time n, we approximate the denominator of the first term in (3.14) via $s^2_n(x_{n+1}; \hat{\boldsymbol{\varphi}}_n) \simeq s^2_n(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1})$, that is, using the estimated posterior mode at time n. Furthermore, the local binomial variance in the second term of (3.14) is approximated by $a_{n+1}\hat{\theta}_n(x_{n+1})(1-\hat{\theta}_n(x_{n+1}))$, where $\hat{\theta}_n(x_{n+1})$ is obtained using (3.13).

Corollary 3.2.3. The posterior variance s^2_{n+1} at step $n+1$ is approximated by

$$s^2_{n+1}(x_{n+1}) \simeq \left(\frac{1}{s^2_n(x_{n+1}; \hat{\boldsymbol{\varphi}}_n)} + \frac{1}{a_{n+1}\hat{\theta}_n(x_{n+1})(1-\hat{\theta}_n(x_{n+1}))} \right)^{-1}, \tag{3.15}$$

where $s^2_n(x_{n+1}; \hat{\boldsymbol{\varphi}}_n)$ is the look-ahead variance from (3.12b) and $\hat{\theta}_n(x_{n+1})$ is from (3.13).

The look-ahead (approximate) variance s_{n+1} forms the basis of numerous *expected improvements* (EI) design heuristics that quantify the gain from sampling at x_{n+1}, see for example [31, 14, 44, 22]. For plain GPs, EI has been extended to the case of batched samples in Kamińsky [34] and Binois et al [7]. Here we adapt these concepts to the setting of *binomial* GPs by quantifying the reduction in posterior variance of $\varphi(x_{n+1})$ at a new location x_{n+1}. We continue to utilize LA and the logistic link function. The idea of adaptive replication is to reduce the predictive variance $s^2_{n+1}(x_{n+1}) \le \nu_n$ below a threshold value ν_n. Using the variance decomposition formula in the RHS of (3.15) and solving for a_{n+1} we have that:

$$a^{\nu}_{n+1} \ge \frac{1}{\hat{\theta}_n(x_{n+1})(1-\hat{\theta}_n(x_{n+1}))} \cdot \left(\frac{1}{\nu_n} - \frac{1}{s^2_n(x_{n+1})} \right).$$

We therefore consider the following adaptive replication scheme:

$$\hat{a}^{\nu}_{n+1} := a^{\nu}_0 \cdot 1_{\{s^2_n(x_{n+1})<\nu_n\}} + \frac{1}{\hat{\theta}_n(x_{n+1})(1-\hat{\theta}_n(x_{n+1}))} \left(\frac{1}{\nu_n} - \frac{1}{s^2_n(x_{n+1})} \right) \cdot 1_{\{s^2_n(x_{n+1})\ge\nu_n\}}. \tag{3.16}$$

Remark 8. We focus on the predictive uncertainty in the latent process φ as a measure to determine a_{n+1} – as opposed to the predictive variance of the random variable $\theta(\varphi(x_{n+1}))$. Focusing on the uncertainty of the latent GP, is a common strategy in sequential design (especially when the data likelihood is Gaussian), see for example [2, 13]. Another common measure for constructing sequential designs is the posterior predictive entropy [35] (which is the preferred uncertainty measure in the active learning framework).

3.2.3 Maximum Likelihood Binomial Regression

An alternative approach is to fit a linear surrogate of the form $\varphi(x) := \boldsymbol{\beta}^T \boldsymbol{\phi}(x)$ for a given set of p *basis functions* $\mathcal{H} = \text{span}(\phi_j : j = 1, \ldots, p)$. The coefficients $\boldsymbol{\beta} \in \mathbb{R}^p$ can be found by optimizing the *penalized* binomial log-likelihood criterion

$$\min_{\boldsymbol{\beta}^T} \sum_{i=1}^{n} \Big\{ B_i \sum_{j=1}^{p} \beta_j \phi_j(x_i) + a_i \log \left(1 + \exp(\sum_{j=1}^{p} \beta_j \phi_j(x_i)) \right) \Big\} + \frac{1}{2} \lambda \mathcal{J}(\sum_{j=1}^{p} \beta_j \phi_j); \quad (3.17)$$

where $\mathcal{J}(\varphi)$ is a penalty functional. The above specification includes the classical *logistic regression model* when the basis $\mathcal{H}$ is monomial and $\lambda = 0$.

Kernel Logistic Regression (KLR). One choice is the family of positive definite kernel functions $\phi_j(\cdot) := \kappa_{l_j}(\cdot; \xi_j)$, where each basis element $\kappa_{l_j}(\cdot, \xi_j)$ is indexed by a location parameter ξ_j and a scale parameter l_j. The corresponding space of functions $\mathcal{H}$ is a Reproducing Kernel Hilbert Space (RKHS) with penalty function $\mathcal{J}(\varphi) = ||\varphi||_2^2 = \boldsymbol{\beta}^T \boldsymbol{\Phi} \boldsymbol{\beta}$, where $\boldsymbol{\Phi}_{ij} = \phi_j(x_i)$. A popular choice is the *Gaussian radial kernel*:

$$\kappa_l(x; \xi) := \exp\left(-\frac{|x - \xi|^2}{l^2} \right). \quad (3.18)$$

Spline Logistic Regression (SLR). Another commonly used functional space $\mathcal{H}$ is the B-spline basis where the ϕ_j's are piecewise continuous functions defined in terms of a set of *knots*. Namely, an order-P spline with knots $(\xi_j)_{j=1}^p$ is a piecewise-polynomial of order P, and has continuous derivatives up to order $P - 2$. The B-spline family takes $P = 4$ and can be represented in terms of $p > 0$ basis functions: $\phi_1(x) = 1$, $\phi_2(x) = x$, and for $j = 2, \ldots, p$, $\phi_{j+1}(x) = d_j(x) - d_{j-1}(x)$, where

$$d_j(x) := \frac{(x - \xi_j)_+^3 - (x - \xi_p)_+^3}{\xi_p - \xi_j}, \quad j = 2, \ldots, p. \quad (3.19)$$

The basis coefficients $\boldsymbol{\beta}$ are fitted by penalizing the curvature of $\varphi(\cdot)$ using $\mathcal{J}(\varphi) = ||\varphi''||_2^2$ [20].

3.3 Sampling Policies

We recall that a sampling policy η is a rule which maps knowledge states to actions, namely selecting sampling locations x_{n+1}. A collection of sampling policies are considered based on the surrogate state f_{T_n} and the fitted $\hat{p}_n(\cdot)$. The sampling decision to be made at step $n+1$ concerns the new site x_{n+1} and the respective number of replicates a_{n+1}. In the spatial modeling paradigm, we consider two complementary ideas which blend the surrogate models for φ with the information about X^* contained in f_n:

(i) first select a_{n+1} and then x_{n+1}; or

(ii) choose x_{n+1} and then determine the respective a_{n+1}.

Approach (i) utilizes fixed batching $a_{n+1} \geq 1$ and selects the new site to query using an information-theoretic criterion. Namely, we use the *batched* expected Kullback-Leibler (KL) divergence between the knowledge state at T_n and $T_n + a_{n+1}$ as in (2.29). To implement this strategy, requires knowledge of the entire $x \mapsto p(x)$. This was one of the main challenges in the original G-PBA in Section 2.3, where IDS was applied ad hoc *after* estimating $p(\tilde{x}_i)$ at a set of $M \geq 2$ candidate locations $\tilde{x}_{1:M}$. However, under our spatial modeling setting, one can utilize the surrogate $\hat{p}_n$ and compute the maximizer of the batched spatial IDS criterion conditional on sampling $a_{n+1} \geq 1$ times at x_{n+1}. Then, x_{n+1} is chosen greedily as the maximizer of $\mathcal{I}(\cdot, f_n, \hat{p}_n(\cdot), a_{n+1})$, that is,

$$x^{\text{sIDS}}_{n+1} := \arg\max_{x \in (0,1)} \mathcal{I}(x, f_n; \hat{p}_n(x), a_{n+1}). \tag{3.20}$$

Practically, a numeric optimization routine is needed to find (3.20). In our experiments below we utilize the `R` package `NLopt` which implements several global constrained optimization routines [29]. In particular, we use a gradient-free deterministic-search algorithm for global optimization named DIRECT (DIviding RECTangles) [30].

For approach (ii) two different schemes are considered. Firstly, pick x_{n+1} using the RQS strategy as in Eq. (2.34). The RQS policy can be interpreted as sampling at a location $X_n \sim f_n$, i.e., sampling based on the posterior distribution of X^*. Furthermore, as discussed in Section 2.3, this policy tends to sample close to the mean of f_n but will also occasionally explore at the latter tails, capturing the trade-off between exploitation and exploration.

An attractive feature of this policy is that it relies solely on f_n so no estimation of the nuisance parameter is needed to recover x_{n+1}^{RQS}, however, under this spatial setting f_n is computed via the surrogate model $\hat{p}_n$.

Conditional on x_{n+1}, a_{n+1} is then picked to control the surrogate uncertainty at x_{n+1} according to the adaptive replication scheme (3.16).

Adaptive One-Step IDS policy. Secondly, note that (3.20) requires specifying the replication amount a_{n+1}, thus adaptive batching is not feasible for the Spatial-IDS procedure as written. To combine the adaptive replication scheme (3.16) with the IDS strategy, we use an ad hoc heuristic which first maximizes $\mathcal{I}$ using $a = 1$ to get x_{n+1} and then obtains the actual a_{n+1}^{ν}. Namely let

$$x_{n+1}^{\text{Ada-sIDS}} = \arg\max_{x \in (0,1)} \mathcal{I}(x; f_n, \hat{p}_n(x), 1). \tag{3.21}$$

3.4 Spatial Generalized PBA

Algorithm 2 specifies the main ingredients for blending surrogate modeling with probabilistic bisection. To initialize it, we use $N_0 \times a_0 = T_0 \ll T$ function evaluations to build $\hat{\varphi}_{N_0}$, picking equidistant (i.e. space-filling) sites $x_{1:N_0}$ in $(0, 1)$ and $a_0 \geq 1$ replications per site. The corresponding f_{T_0} is constructed via (2.6). Note that we first non-sequentially construct the surrogate $\hat{\varphi}_{N_0}$ using all T_0 queries, and only then compute f_{T_0}. Also note that the surrogate re-fitting step in Algorithm 2 is optional (i.e. user-controlled), since re-fitting can be expensive. In principle, re-fitting could be stopped entirely once n is large enough, which allows to keep the overhead cost of predicting $\theta_n(x)$ fixed, rather than increasing in n.

```
**PBA parameters:** Prior $f_0$; $T_0$ and $a_0 \geq 1$. Set $N_0 := T_0/a_0$;
**Surrogate initialization:** Regress $B_{1:N_0}$ on locations $x_{1:N_0}$ to obtain the
 surrogate model $\hat{\theta}_{N_0}$;
Update knowledge state starting from $f_0$ to $f_{T_0}$ given $\hat{\theta}_{N_0}$, $B_{1:N_0}$ and $x_{1:N_0}$;
$n \leftarrow N_0$, $T_n \leftarrow T_0$, $\mathcal{D}_n \leftarrow (B_{1:N_0}, a_{1:N_0})$;
while $T_n < T$ do
    Using $f_n$ generate next sampling location $x_{n+1}$ and batch size $a_{n+1}$;
    Query oracle $a_{n+1}$ times at $x_{n+1}$ to observe $B_{n+1}(x_{n+1})$;
    if (OPTIONAL) then
        Re-fit surrogate for $\hat{\theta}_{n+1}$ based on $\mathcal{D}_{n+1} = (\mathcal{D}_n, B_{n+1}, a_{n+1})$;
    else
        $\hat{\theta}_{n+1} \leftarrow \hat{\theta}_n$;
    Update knowledge state at $x_{n+1}$ $f_{n+1} \leftarrow \Psi(f_n, x_{n+1}, B_{n+1}; \hat{p}_{n+1}, a_{n+1})$ using
     $\hat{p}_{n+1} = \max\{\hat{\theta}_{n+1}(x_{n+1}), 1 - \hat{\theta}_{n+1}(x_{n+1})\}$;
    Update $T_n \leftarrow T_n + a_{n+1}$ and $n \leftarrow n + 1$;
end
return Knowledge state $f_N$ and estimator for the root location
 $\hat{x}_N = \text{median}(f_N)$;
```

Algorithm 2: Spatial Generalized-PBA.

Chapter 4

Numeric Examples

In this Chapter a series of numerical results are presented based on Monte-Carlo (MC) replications, that is, we repeatedly apply our algorithms fixing the components that the user must pick. In particular, to empirically assess our generalized PBAs (introduced in Chapter 2 and Chapter 3), we mix and match the sampling policy η (information-directed or randomized based), estimation method $\hat{p}$ for the oracle accuracy (local or spatial) and batch size a (either fixed or adaptive). We consider several metrics to quantify the quality of the root estimates induced by $(\eta, \hat{p}, a)$: absolute residuals, credible intervals and its corresponding coverage, as well as the KL divergence between the approximated knowledge state f_n and the true posterior root density g_n.

Furthermore, to focus exclusively on evaluating the sampling component of the G-PBA, the proposed policies η are benchmarked with respect to other schemes that adopt the true posterior density g_n (and hence the ground-truth oracle accuracy), such as the information-directed and Uniform sampling strategies, which correspond to the best and worst case scenarios, respectively, as well as the true RQS sampling scheme.

Our numeric examples are based on three test functions, that is, we specify the actual functional form of the unknown $h(\cdot)$ in the stochastic simulator (1.1). They illustrate different aspects and difficulties typically encountered in SRFPs, such as heteroscedasticity or zero curvature at the root location. A real-life challenging example is analyzed, which consists of estimating the optimal exercise price of a Bermudan Put financial derivative.

This Chapter is organized as follows. In Section 4.1 we state the overall experimental configuration. In particular, we define our three synthetic examples and the performance evaluation metrics. In Section 4.2 and Section 4.3, we present the numeric results for the Local and Spatial G-PBAs, respectively. Finally, in Section 4.5 we apply G-PBA in order to solve the Optimal Stopping Problem in the context of pricing a Bermudan Put financial derivative.

4.1 Experimental Setup

Synthetic Examples. In analogy to [57], we utilize the following three test functions defined for all $x \in (0, 1)$, cf. Figure 4.1:

1. The linear function,

$$h_1(x) = X^* - x, \qquad \sigma_1(x) = 0.2; \tag{4.1}$$

2. the exponential function,

$$h_2(x) = \exp\{2(X^* - x)\} - 1, \qquad \sigma_2(x) = 0.2 \cdot 1_{\{x<X^*\}} + 1 \cdot 1_{\{x>X^*\}}; \tag{4.2}$$

3. and the cubic function,

$$h_3(x) = (X^* - x)^3, \qquad \sigma_3(x) = 0.025. \tag{4.3}$$

Example (4.1) consists of a linear function whose slope is constant and different from zero at points close to the root X^*, indicating that most of the stochastic root-finding procedures should work well. The curvature of (4.2) creates an asymmetry in sampling: a measurement leftwards of X^* yields a correct response with higher probability relative to a measurement to the right of the root. Notice that the variance of the noise component $\sigma_2(x)$ also depends on the sampling location x, where $\sigma_2(x)$ is lower for locations leftwards x^* and significantly higher for $x > x^*$. Consequently, f_n is expected to be skewed. Finally, example (4.3) represents a difficult root-finding setting due to $h_3'(X^*) = 0$, which implies that $p(x) \simeq 1/2$ for x in the vicinity of X^*.

In all cases the noise term is assumed to be zero-mean Gaussian, $\epsilon(x) \sim \mathsf{N}(0, \sigma_i^2(x))$, implying that the function evaluations $Z(x)$ in (1.1) are normal random variables with mean $\mathbb{E}[Z(x)] = h_i(x)$ and variance $\mathbb{V}ar(Z(x)) = \sigma_i^2(x)$.

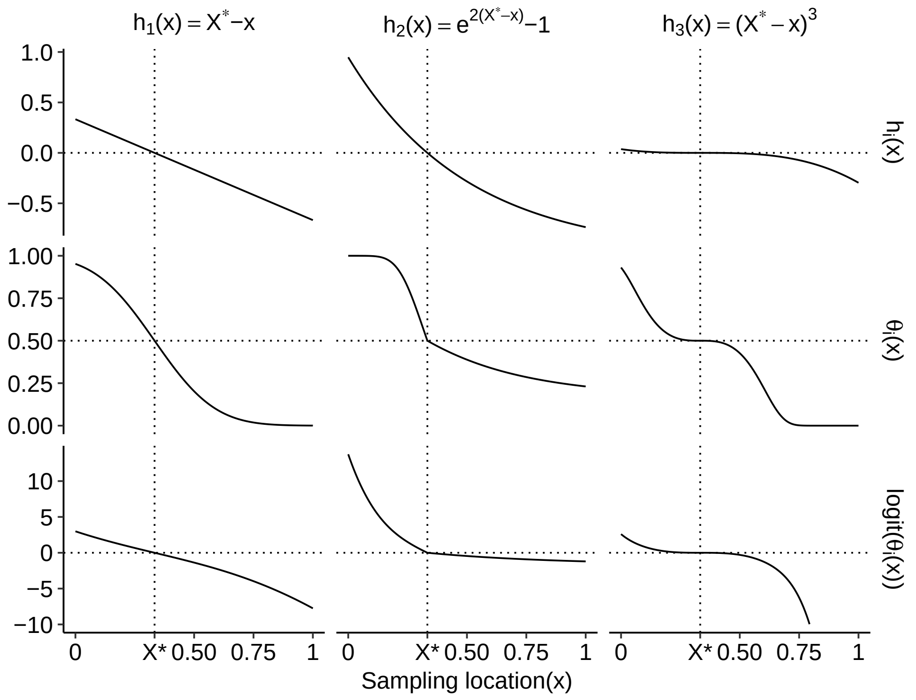

Figure 4.1: Synthetic test functions (4.1), (4.2) and (4.3).

Performance Evaluation Metrics. To evaluate the quality of the approximated knowledge state f_n for fixed $(\eta, \hat{p}, a)$, the following four performance metrics are used:

1. *Absolute residuals*: to determine the accuracy of the estimator $\hat{x}_n := \text{median}(f_n)$

we consider the L_1-residuals,

$$r_a^\eta(f_n) := |\hat{x}_{n,a}^\eta - x^*|; \tag{4.4}$$

2. *credible intervals*: to evaluate the degree of uncertainty associated with the unknown root location X^* via the length of a symmetric $(1-\alpha)\%$ credible interval (CI) between the $\alpha/2$ and $(1-\alpha/2)$ percentiles of f_n:

$$l_{a,1-\alpha}^\eta(f_n) := F_n^{-1}(1-\alpha/2) - F_n^{-1}(\alpha/2); \tag{4.5}$$

3. *coverage*: to measure the accuracy of the above CI defined as

$$c_{a,1-\alpha}(f_n) := \boldsymbol{Pr}\left\{x^* \in [F_n^{-1}(\alpha/2), F_n^{-1}(1-\alpha/2)]\right\}, \tag{4.6}$$

where the averaging in $\boldsymbol{Pr}\{\cdot\}$ is across MC runs of the algorithm. If $c_{1-\alpha}(f_n) \ll (1-\alpha)$ the coverage test indicates that f_n prematurely collapses or equivalently overstates its confidence about X^*. Namely, small CI length $l_{a,1-\alpha}$ relative to residuals r will lead to low coverage c. For both the coverage and the length of the credible interval we use $\alpha = 0.05$; and finally

4. *KL divergence*: to compare f_n to the true posterior g_n based on the sequence of the chosen locations $x_{1:n}^\eta$, we use the KL divergence denoted $D(f_n; g_n)$. Namely, since both f_n and g_n are updated at the same set of knots $\tilde{x}_{1:n}^\eta$ (sorted in increasing order), we may write $g_n(x) := \sum_{j=1}^n g(\tilde{x}_{j-1}) 1_{x\in[\tilde{x}_{j-1},\tilde{x}_j)}$ and $f_n(x) := \sum_{j=1}^n f(\tilde{x}_{j-1}) 1_{x\in[\tilde{x}_{j-1},\tilde{x}_j)}$, with $\tilde{x}_0 := 0$ and $\tilde{x}_n := 1$. We then obtain:

$$\begin{aligned} D_a^\eta(f_n; g_n) &:= \int_0^1 \log\left(\frac{f_n(x)}{g_n(x)}\right) f_n(x)dx \\ &= \sum_{j=1}^n \int_{\tilde{x}_{j-1}}^{\tilde{x}_j} \log\left(\frac{f_n(x)}{g_n(x)}\right) f_n(x)dx \\ &= \sum_{j=1}^n \log\left(\frac{f(\tilde{x}_j)}{g(\tilde{x}_j)}\right) f(\tilde{x}_j)(\tilde{x}_j - \tilde{x}_{j-1}). \end{aligned} \tag{4.7}$$

We make the usual convention that $\log(f(x)/g(x))f(x) = 0$ if $f(x) = 0$ (including when $g(x) = 0$), as well as $\log(f(x)/g(x))f(x) = +\infty$ if $g(x) = 0$ and $f(x) > 0$ [16]. Practically, to compute the average KL divergence we consider only finite values.

Monte-Carlo Iterations. Performance metrics (4.5), (4.6), (4.4), and (4.7) are estimated using a total of $MC = 100$ Monte-Carlo macro-iterations for each combination in $(\eta, \hat{p}, a)$ and a total number of simulation outputs $T = 20{,}000$. In order to make all methods comparable, we fix $X^*_{(i)} \sim \mathsf{Unif}(0, 1)$ so each estimation scheme is applied using the same root value $X^*_{(i)}$ during the i-th MC iteration, $i = 1, \ldots, MC$.

4.2 Empirical Performance of Local G-PBA

4.2.1 Parameter configuration

Sampling policies η. The sequential policies introduced in Section 2.3 are implemented as follows:

- Deterministic IDS (Det-IDS): which learns the batched information criterion (2.30) at $M = 2$ test locations $\tilde{x}_{1:M} = (F_n^{-1}(0.25), F_n^{-1}(0.75))$ (i.e., at the 25-th and 75-th quantiles $\tilde{x}_i$ of f_n) and chooses greedily the point x_{n+1} at which $\mathcal{I}(\tilde{x}_i; f_n, \hat{p}(\tilde{x}_i), a)$ is maximal;
- Randomized IDS (Rand-IDS): maximizes the batched IDS criterion (2.30) among $M = 2$ random quantiles of f_n as in (2.32).
- Randomized Quantile Sampling (RQS): $x_{n+1} \sim f_n$ as in (2.34).
- Systematic Quantile Sampling (SQS): chooses the next sampling location iterating over $M = 2$ pre-specified quantiles of f_n, viz. $\check{q}_{0:1} = (0.25, 0.75)$, that are systematically rotated using (2.33).

Local estimators for $p(\cdot)$. The frequentist and Bayesian procedures based on the majority proportion statistic $\bar{p}$, as well as the estimators based on aggregation of responses are considered:

- the empirical majority proportion $\bar{p}$ from (2.12);
- the posterior mode $\hat{p}_{\mathscr{L}_0}$ (2.18), posterior median $\hat{p}_{\mathscr{L}_1}$ (2.19), and posterior mean $\hat{p}_{\mathscr{L}_2}$ (2.20); and
- the procedures which aggregate oracle signs $\mathscr{P}_{\mathscr{M}}(\bar{p})$ (2.21) (combined with empirical proportion $\bar{p}$), as well as functional responses $\mathscr{P}_{\mathscr{S}}(\hat{h}_a, \hat{\sigma}_a)$ (2.24).

Batch size a. For the local estimation procedures specified above, the batch size remains fixed at $a \in \{100, 250\}$.

Finally, we also compare out G-PBAs to the TPO policy that follows the classical PBA sampling strategy, i.e., $x_{n+1} = F_n^{-1}(1/2)$, and performs a random number of oracle calls $a_\alpha(x_{n+1})$ based on (2.26). In order to compute the curved boundary a_α for the numeric examples we present, we plug in the true oracle sample variance $\sigma_i^2(x)$ and truncate sampling if it does not terminate by final clock-time T: $\tilde{a}(x) := \min\{T - \sum_{j=1}^{n-1} a_\alpha(x_j), a_\alpha(x)\}$ with the resulting Z-based estimator $\mathscr{P}_{\mathscr{S}_{\tilde{a}}}(\hat{h}_{\tilde{a}}(x), \hat{\sigma}_{\tilde{a}}(x))$. We consider two boosting levels $\alpha \in \{0.05, 0.4\}$.

In summary, the Local G-PBA space $(\eta, \hat{p}, a)$ consists of 4 sampling policies η, 6 estimation methods for $\hat{p}$, and 2 batch sizes a, plus two versions of the TPO procedure, for a total of $6 \times 4 \times 2 + 2 = 50$ combinations.

4.2.2 Results

We use the linear test function (4.1) as our running example to illustrate the empirical performance of the G-PBA.

Figure 4.2 compares the performance of different $(\eta, \hat{p}, a)$ schemes as a function of wall-clock budget T. In terms of estimating $p(\cdot)$, the best method is unsurprisingly the CLT approximation $\mathscr{P}_{\mathscr{S}}(\hat{h}_a, \hat{\sigma}_a)$, which directly leverages the functional responses $Z(\cdot)$. This quantifies the intuition that the Zs carry more information than the sign-based oracle responses in (1.2). As a consequence, using $\mathscr{P}_{\mathscr{S}}$ leads to lower residuals and better coverage. Specifically, it provides better recovery of the correct posterior distribution (due to smaller updating errors), which is moreover confirmed by the minimal KL divergence associated to this scheme. Among the estimators that rely only on signs of the function evaluations, two good choices are the posterior mode $\hat{p}_{\mathscr{L}_0}$ and the empirical proportion $\bar{p}$. Both of these maintain a good balance between uncertainty reduction and low absolute residuals. Recall that these procedures were shown to be conservative in over-estimating $p(x)$ and hence better at controlling the bias in the updating of f_n, cf. Section 2.2.3. This is important in the latter stages as $p(\cdot) \simeq 1/2$.

In terms of the sampling strategies, the Rand-IDS and Rand-Q policies with batch size $a = 250$ perform best for minimizing residuals. We observe that all methods struggle with coverage, indicating that f_T prematurely collapses due to "overconfidence" induced by the bias in $\hat{p}$, cf. Section 2.2.3 (which is reflected on the significantly large KL average divergence related to these methods). This effect is naturally mitigated by a larger batch size a at the cost of sampling at fewer locations. Moreover, coverage metrics for IDS methods are higher, primarily driven by the fact that they use fewer macro-iterations (since $N^{IDS} = T/(a \cdot M)$) and hence are less affected by the bias. However, this effect is not useful in practice since the IDS methods also have much wider CI's, i.e., they are conservative about X^*. We observe that Syst-Q is consistently worse than Rand-Q: they both generate similar absolute residuals, but the CI/coverage of Rand-Q is larger, indicating that it is better in approximating the true posterior g_n. Both Randomized strategies (2.32) and (2.34) perform dramatically better than the Deterministic counterpart (2.31) and (2.33) for all $\hat{p}$ in terms of minimizing both the absolute residuals and the length of the 95% CI.

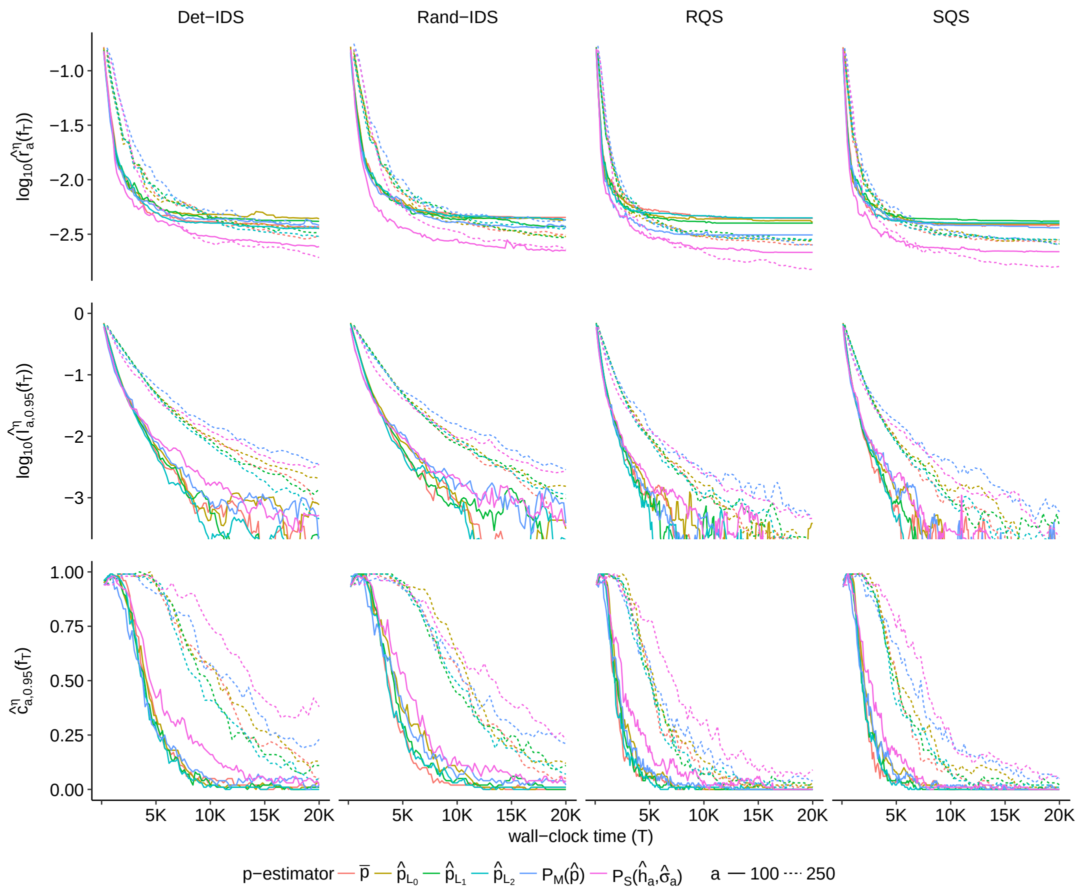

Figure 4.2: Average Monte-Carlo performance statistics applied to the linear test function h_1 using Local G-PBA.

Furthermore, Figure 4.2 indicates that the learning rate of the sampling schemes changes over iterations: the randomized methods yield a more rapid reduction in absolute residuals for T small (i.e., during the first few steps), while the systematic methods enjoy a better asymptotic improvement. This suggests a hybrid heuristic of randomizing the first few macro-iterations (exploring with Rand-Q), and then more aggressively selecting points to maximize entropy reduction (exploiting with Syst-IDS). When local estimators are deployed learning is then sub-exponential (i.e., sub-linear on the log-scale), which occurs due to $p(\cdot) \simeq 0.5$ around the root, and necessarily slows down information gains. Indeed, exponential convergence is only feasible when $p(\cdot)$ is bounded away from $1/2$. Interestingly, Figure 4.2 suggests that the CI of f_T decreases linearly in T, which is inconsistent with the above slow learning rate of X^* and subsequently ruins coverage, as the mass of the knowledge state f_T no longer includes the true X^*. It shows that the Rand-IDS method is best able to suppress this.

Table 4.1 lists the final summary statistics at $T = 20,000$ for the considered combinations $(\eta, \hat{p}, a)$ utilizing the linear test function h_1. We observe that even for this straightforward setting, a small batch size (in this case $a = 100$) is insufficient, leading to a situation whereby absolute residuals are very large and the average 95% CI length is small, so that f_T is collapsing prematurely. On the other hand, for $a = 250$, the average absolute residuals, as well as the average 95% CI length are significantly small across all η and $\hat{p}$, indicating that the associated posterior f_T is placing most of its mass near the actual root value x^*. If $a = 250$, then it can also be observed that there is a good balance between residuals and length of CI. In particular, Det-IDS and RQS behaves well in combination with the CLT estimator.

Additionally, Table 4.1 confirms the estimation method which best resembles the actual root posterior g_n, as measured by the average KL divergence (last column), is the majority proportion $\bar{p}$ as well as the posterior mode $\hat{p}_{\mathscr{L}_0}$.

Finally, the last two rows of Table 4.1 summarize the performance of TPO-PBA. This policy leads to very large batch sizes, and in this case study used just $N_\alpha = 6$ and $N_\alpha = 9$ (median) sampling locations with $\alpha = 0.05$ and $\alpha = 0.40$, respectively. As a result, TPO-PBA is not able to learn X^*, leading to average residuals and length of CI significantly larger in comparison to our G-PBA policies. Notice, however, that the KL divergence with respect to the true posterior g_n is minimized in contrast to all the other G-PBA local estimation methods, which is due to the fact that $\tilde{p}(x_i)$ is close to the the true $p(x_i)$ for the insufficient number of updating points that were selected by this policy.

Table 4.1: MC summary metrics for the test function h_1 obtained at $T = 20,000$ using the Local G-PBA. TPO policy is included in the last two rows.

η	$\hat{p}$	$\hat{r}^{\eta}_{a}(f_T)$ (10^{-2})		$\hat{l}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{c}^{\eta}_{a,0.95}(f_T)$		$\hat{D}^{\eta}_{a,0.95}(f_T; g_T)$	
		a=100	a=250	a=100	a=250	a=100	a=250	a=100	a=250
Det-IDS	$\bar{p}$	0.3692	0.2996	0.0196	0.0773	0.01	0.05	26.58	7.85
	$\hat{p}_{\mathscr{L}_0}$	0.4377	0.3576	0.0769	0.2068	0.03	0.13	21.78	6.40
	$\hat{p}_{\mathscr{L}_1}$	0.4150	0.3281	0.0223	0.1378	0.01	0.11	26.02	8.15
	$\hat{p}_{\mathscr{L}_2}$	0.3585	0.3022	0.0010	0.0452	0.00	0.05	28.80	9.35
	$\mathscr{P}_{\mathscr{M}}$	0.3817	0.3631	0.0159	0.3588	0.02	0.23	26.61	10.25
	$\mathscr{P}_{\mathscr{S}}$	0.2435	0.1928	0.0519	0.3496	0.02	0.38	22.76	10.30
Rand-IDS	$\bar{p}$	0.4509	0.3121	0.0016	0.0420	0.00	0.05	24.45	8.06
	$\hat{p}_{\mathscr{L}_0}$	0.4285	0.3003	0.0312	0.1535	0.01	0.12	22.09	6.55
	$\hat{p}_{\mathscr{L}_1}$	0.3722	0.2959	0.0005	0.0972	0.00	0.11	26.34	7.54
	$\hat{p}_{\mathscr{L}_2}$	0.4285	0.3575	0.0206	0.1124	0.01	0.07	27.65	8.11
	$\mathscr{P}_{\mathscr{M}}$	0.3608	0.4313	0.0384	0.2861	0.02	0.21	24.62	11.10
	$\mathscr{P}_{\mathscr{S}}$	0.2238	0.2292	0.0343	0.2732	0.04	0.23	25.05	10.60
RQS	$\bar{p}$	0.4422	0.2528	0.0000	0.0038	0.00	0.01	31.74	19.59
	$\hat{p}_{\mathscr{L}_0}$	0.4099	0.2735	0.0384	0.0202	0.01	0.03	32.60	17.63
	$\hat{p}_{\mathscr{L}_1}$	0.3997	0.2783	0.0000	0.0106	0.00	0.01	33.55	20.25
	$\hat{p}_{\mathscr{L}_2}$	0.4466	0.2833	0.0000	0.0006	0.00	0.00	32.17	21.22
	$\mathscr{P}_{\mathscr{M}}$	0.3113	0.2510	0.0000	0.0515	0.00	0.04	45.95	20.22
	$\mathscr{P}_{\mathscr{S}}$	0.2154	0.1516	0.0001	0.0463	0.00	0.09	41.18	18.44
SQS	$\bar{p}$	0.3829	0.2701	0.0017	0.0075	0.00	0.00	33.64	19.65
	$\hat{p}_{\mathscr{L}_0}$	0.3931	0.2812	0.0000	0.0097	0.00	0.00	33.59	16.41
	$\hat{p}_{\mathscr{L}_1}$	0.4171	0.2583	0.0000	0.0342	0.00	0.02	32.42	20.20
	$\hat{p}_{\mathscr{L}_2}$	0.4012	0.2811	0.0000	0.0077	0.00	0.01	33.86	20.60
	$\mathscr{P}_{\mathscr{S}}$	0.2186	0.1600	0.0000	0.0234	0.00	0.05	42.13	19.23
	$\mathscr{P}_{\mathscr{M}}$	0.3630	0.2556	0.0014	0.0589	0.00	0.06	42.73	20.26
TPO	$\tilde{p}_{0.05}$	16.5302		95.9121		0.736		3.5501	
	$\tilde{p}_{0.40}$	25.6349		32.2533		0.711		2.0963	

Sensitivity Analysis for the Exponential and Cubic Test Functions. Performance evaluation metrics (Tables only and not including the TPO policy) for the other test cases (4.2)–(4.3) appear in Table 4.2 and Table 4.3, respectively. Here we discuss the main take-aways.

Table 4.2: Summary metrics for the test function h_2 obtained at $T = 20,000$ using the Local G-PBA.

η	$\hat{p}$	$\hat{r}^{\eta}_{a}(f_T)$ (10^{-2})		$\hat{l}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{c}^{\eta}_{a,0.95}(f_T)$		$\hat{D}^{\eta}_{a,0.95}(f_T; g_T)$	
		a=100	a=250	a=100	a=250	a=100	a=250	a=100	a=250
Det-IDS	$\bar{p}$	0.6848	0.4418	0.0211	0.1158	0.02	0.07	27.40	8.85
	$\hat{p}_{\mathscr{L}_0}$	0.6570	0.5756	0.0090	0.3639	0.01	0.18	24.10	6.99
	$\hat{p}_{\mathscr{L}_1}$	0.7829	0.4628	0.0071	0.1454	0.01	0.09	27.29	9.38
	$\hat{p}_{\mathscr{L}_2}$	0.7154	0.4275	0.0015	0.1175	0.00	0.06	29.55	11.41
	$\mathscr{P}_{\mathscr{M}}$	0.6233	0.5598	0.0368	0.5376	0.02	0.20	26.15	10.49
	$\mathscr{P}_{\mathscr{S}}$	0.4499	0.3621	0.0059	0.2086	0.01	0.20	24.76	10.70
Rand-IDS	$\bar{p}$	0.6448	0.5433	0.0129	0.0797	0.00	0.06	26.39	8.07
	$\hat{p}_{\mathscr{L}_0}$	0.8779	0.5755	0.1166	0.2452	0.03	0.13	20.53	6.95
	$\hat{p}_{\mathscr{L}_1}$	0.6653	0.4946	0.0208	0.0413	0.01	0.05	23.87	8.73
	$\hat{p}_{\mathscr{L}_2}$	0.7814	0.6714	0.0005	0.1299	0.00	0.07	27.28	8.58
	$\mathscr{P}_{\mathscr{M}}$	0.6400	0.5222	0.0074	0.4183	0.00	0.18	29.02	11.53
	$\mathscr{P}_{\mathscr{S}}$	0.4616	0.3857	0.0386	0.5275	0.04	0.37	20.14	9.48
RQS	$\bar{p}$	0.7075	0.4846	0.0428	0.0649	0.02	0.07	27.88	12.36
	$\hat{p}_{\mathscr{L}_0}$	0.8442	0.4686	0.0477	0.0527	0.02	0.06	24.43	10.71
	$\hat{p}_{\mathscr{L}_1}$	0.7498	0.4775	0.0016	0.1313	0.00	0.06	29.67	11.64
	$\hat{p}_{\mathscr{L}_2}$	0.6312	0.5135	0.0032	0.0810	0.00	0.04	31.81	12.84
	$\mathscr{P}_{\mathscr{M}}$	0.4965	0.3578	0.0080	0.1317	0.00	0.08	37.70	17.15
	$\mathscr{P}_{\mathscr{S}}$	0.3955	0.3037	0.0391	0.1492	0.02	0.17	27.29	12.64
SQS	$\bar{p}$	0.6663	0.4772	0.0198	0.0061	0.00	0.00	30.50	12.44
	$\hat{p}_{\mathscr{L}_0}$	0.7279	0.4610	0.0026	0.1400	0.00	0.11	26.87	9.31
	$\hat{p}_{\mathscr{L}_1}$	0.5923	0.4225	0.0434	0.0649	0.01	0.03	30.41	12.89
	$\hat{p}_{\mathscr{L}_2}$	0.7567	0.5441	0.0000	0.1141	0.00	0.04	32.32	14.16
	$\mathscr{P}_{\mathscr{M}}$	0.5071	0.4422	0.0001	0.0544	0.00	0.04	42.44	18.80
	$\mathscr{P}_{\mathscr{S}}$	0.3734	0.3439	0.0265	0.1344	0.01	0.05	32.14	14.52

Among policies, Rand-Q works best for h_2 and Syst-Q for h_3 although the differences are not significant. As before, the functional response estimator $\mathscr{P}_{\mathscr{S}}(\hat{h}_a, \hat{\sigma}_a)$ performs best for root-finding, yielding lowest estimation error and highest coverage. This confirms the value of using the actual functional responses, in contrast to only its sign. Due to the more difficult setting, a larger batch size $a = 250$ is needed, representing a total of $N = 80$ sampling locations for the Q-based policies, and $N^{IDS} = 40$ for the IDS policies, respectively. Tables 4.2 and 4.3 show the complete failure of PBA when only local information is leveraged if a is too small ($a = 100$ in the Tables) whereby f_T collapses, severely underestimating the posterior uncertainty and leading to almost *zero* coverage (as depicted in the third column of the Tables).

Table 4.3: Summary metrics for the test function h_3 obtained at $T = 20,000$ using the Local G-PBA.

η	$\hat{p}$	$\hat{r}^{\eta}_{a}(f_T)$ (10^{-2})		$\hat{l}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{c}^{\eta}_{a,0.95}(f_T)$		$\hat{D}^{\eta}_{a,0.95}(f_T; g_T)$	
		a=100	a=250	a=100	a=250	a=100	a=250	a=100	a=250
Det-IDS	$\bar{p}$	5.3257	4.8835	0.0187	0.4446	0.00	0.03	33.34	11.93
	$\hat{p}_{\mathscr{L}_0}$	5.7587	5.3403	0.0001	0.3862	0.00	0.01	27.94	10.01
	$\hat{p}_{\mathscr{L}_1}$	5.7173	4.7734	0.0329	0.3032	0.00	0.03	33.16	13.76
	$\hat{p}_{\mathscr{L}_2}$	5.3107	5.0278	0.0004	0.4157	0.00	0.01	35.90	13.22
	$\mathscr{P}_{\mathscr{M}}$	5.2335	4.8161	0.0284	1.2347	0.00	0.09	28.49	11.02
	$\mathscr{P}_{\mathscr{S}}$	4.2913	4.2176	0.2149	1.9549	0.01	0.09	25.03	10.12
Rand-IDS	$\bar{p}$	5.2108	4.9102	0.0013	0.1877	0.00	0.00	34.35	12.70
	$\hat{p}_{\mathscr{L}_0}$	4.7761	4.8908	0.1397	0.6496	0.01	0.04	29.34	10.55
	$\hat{p}_{\mathscr{L}_1}$	5.5064	4.6287	0.1649	0.0345	0.01	0.00	31.40	13.39
	$\hat{p}_{\mathscr{L}_2}$	5.2753	4.9914	0.0001	0.7631	0.00	0.02	32.01	13.50
	$\mathscr{P}_{\mathscr{M}}$	5.3876	4.7527	0.5744	1.5573	0.01	0.10	29.44	12.22
	$\mathscr{P}_{\mathscr{S}}$	4.4133	4.4046	0.4867	2.8545	0.03	0.11	19.62	9.09
RQS	$\bar{p}$	5.1556	4.7262	0.0000	0.2978	0.00	0.01	37.77	15.16
	$\hat{p}_{\mathscr{L}_0}$	5.3406	4.7325	0.0001	0.7267	0.00	0.02	31.77	12.68
	$\hat{p}_{\mathscr{L}_1}$	5.2306	4.6381	0.0213	0.2493	0.00	0.01	36.03	14.32
	$\hat{p}_{\mathscr{L}_2}$	4.8320	4.4068	0.0000	0.3840	0.00	0.02	38.97	17.72
	$\mathscr{P}_{\mathscr{M}}$	5.7807	4.1977	0.0000	0.2915	0.00	0.02	41.69	17.41
	$\mathscr{P}_{\mathscr{S}}$	4.6983	3.8401	0.1229	1.3473	0.01	0.07	27.60	11.38
SQS	$\bar{p}$	4.7741	4.3860	0.0048	0.0839	0.00	0.01	39.94	16.62
	$\hat{p}_{\mathscr{L}_0}$	5.5714	4.2043	0.0005	0.5547	0.00	0.04	33.53	13.81
	$\hat{p}_{\mathscr{L}_1}$	4.8444	4.5924	0.0017	0.0875	0.00	0.00	40.46	16.96
	$\hat{p}_{\mathscr{L}_2}$	6.0304	4.7480	0.0000	0.1208	0.00	0.02	38.45	19.03
	$\mathscr{P}_{\mathscr{M}}$	5.6644	4.9692	0.0000	0.1565	0.00	0.01	46.61	19.99
	$\mathscr{P}_{\mathscr{S}}$	4.4417	4.1947	0.0010	0.5957	0.00	0.03	33.02	14.68

If only the batched response sign (1.2) at each sampling location is used to learn X^*, then there is no clear "winner" among the proposed methods. We observe that the IDS policies are less accurate (higher $\hat{r}$) but also have higher coverage. Similarly, the empirical $\bar{p}$ and posterior median $\hat{p}_{\mathscr{L}_1}$ are best for maximizing accuracy while the posterior mode $\hat{p}_{\mathscr{L}_0}$ is best for maximizing coverage. This is consistent with previous discussion that $\hat{p}_{\mathscr{L}_0}$ minimizes bias in learning $p(x) \simeq 1/2$ and is, therefore a more "conservative" approach that slows down error propagation in f_n. The majority boosting approach with $\mathscr{P}_{\mathscr{M}}$ also works quite well.

We observe that in these more challenging settings, all methods suffer from model misspecification which cause f_T to deviate from the true posterior and lead to poor statistical coverage with respect to the true root. This premature posterior collapse ranges from extremely severe ($\hat{r} \gg \hat{l}$ so the residuals are much larger than the estimated uncertainty about X^*), to moderate (coverage $\hat{c}_{0.95} \in [0.2, 0.5]$). As we will show in Section 4.3, spatial modeling of the oracle accuracy would guarantee asymptotic consistency in the sense of matching the preset coverage levels. For now our results confirm the strong sensitivity of PBA to properly estimating oracle properties and the discrepancy between the generally low residuals obtained (i.e. good root estimate) and the mediocre quantification of root uncertainty.

4.3 Empirical Performance of Spatial G-PBA

We now proceed to evaluate the performance of the Spatial G-PBA stated in Algorithm 2 that are presented in Chapter 3.

4.3.1 Parameter configuration

Sampling Policies η. We consider the three schemes that leverage spatial information:

- Spatial-IDS (sIDS) (3.20);
- Spatial-RQS (sRQS) (2.34); and
- the one-step sIDS (3.21) combined with the adaptive replication scheme a^{ν}_{n+1}.

Surrogates for $p(\cdot)$. The configuration of the non-parametric models (introduced in Section 3.2.1 and Section 3.2.3) is given as follows:

- *B-GP.* For the binomial GP (B-GP) we use the 5/2-Matérn covariance kernel (3.5). The hyper-parameters $\vartheta = (\tau^2, l)$ are estimated via a Bayesian MAP estimation procedure, placing a square root uniform prior (i.e., $q_0(\sqrt{\tau^2}) \propto 1$) on τ^2 and a Student-t prior on the length scale parameter l (both default priors for binomial GPs in `GPstuff`). Although parameter estimation can be expensive, the B-GP is *re-fitted* and *updated* every $T_n = a_n$ simulation outputs; that is, the hyper-parameters $\hat{\vartheta}$ are re-fitted and the posterior mode $\hat{\boldsymbol{\varphi}}_n$ is re-computed every time a new pair of sampling location/binomial response is observed, such that the surrogate is able to assimilate acquired information.

- *KLR.* Kernel Logistic Regression (KLR) is implemented with the Gaussian kernel basis function (3.18) using a *fixed* length scale parameter $l \equiv 1$ and centering ϕ_j at each sampling location $\xi_j \equiv x_j$, $j = 1, \ldots, n$ (implying that we use as many kernel functions as sampling points to learn φ). Since we would like to induce a surrogate model $\hat{\varphi}$ that closely resembles the local estimators $\hat{p}(\cdot)$, we use a (small) *fixed* value $\lambda = 0.01$ as the penalty parameter for optimizing (3.17). Numerically, we implement KLR as stated in Algorithm 1 of [62].

- *SLR:* We consider a *smoothing* spline logistic regression (SLR) model where the penalty coefficient $\hat{\lambda}$ (aka smoothing parameter) is estimated via Generalized Cross-Validation [20] jointly with the spline basis coefficients. In this case, the spline knots ξ_j are placed at percentiles of the sampling locations $x_{1:n}$. Thus, as the mass of f_n concentrates around x^* (and hence sampling locations $x_{1:n}$ concentrate around the root), more knots ξ_j's are also placed near X^*, making the surrogate more localized in regions where the variability of the binomial responses B_n is maximal.

- *LR.* Polynomial logistic regression with $\varphi(x) = \beta_0 + \sum_{j=1}^{5} \beta_j x^5$, a *quintic* polynomial and zero penalty $\lambda = 0$ (to enforce surrogate flexibility). We implement both the SLR and LR surrogates using the `gam()` routine from the `mgcv` package in `R` [61].

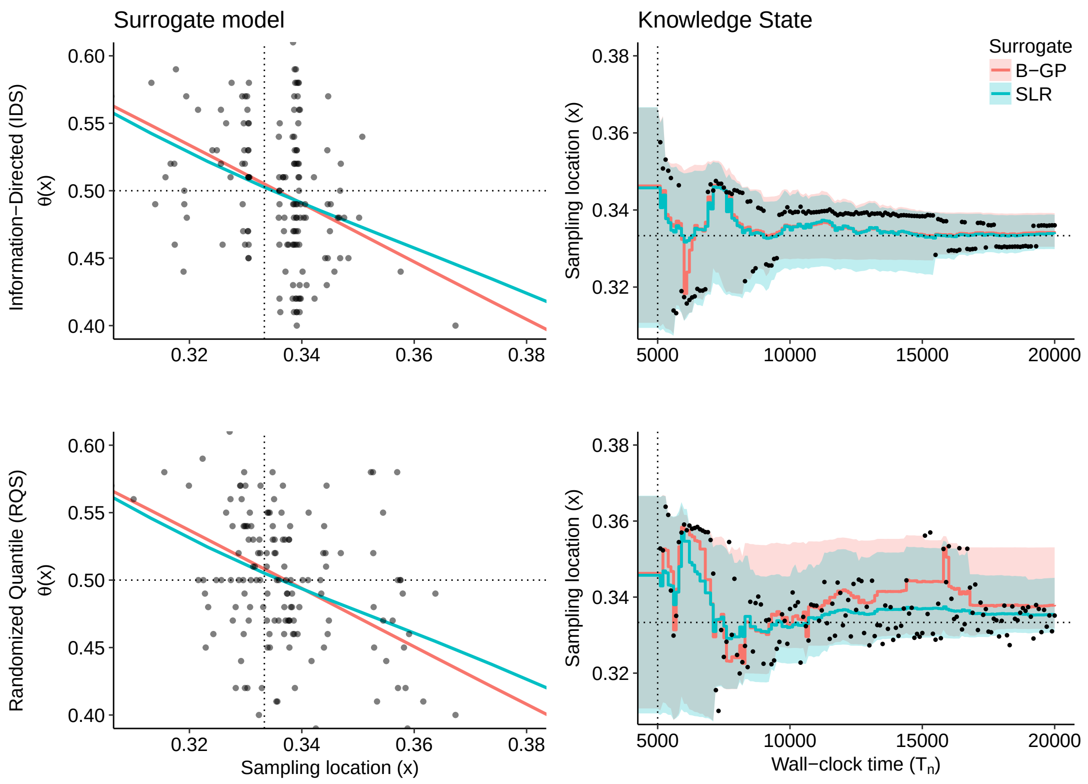

Figure 4.3: *Left*: B-GP and SLR surrogates trained on a fixed data set obtained using the sIDS policy (first row) and the sRQS policy (second row) after $T = 20,000$ iterations and batch size $a_0 = 100$. *Right*: posterior IQRs (shaded regions) of f_n, using B-GP and SLR using the sIDS policy (first row) and the sRQS policy (second row) measured in wall-clock time T_n. The corresponding root-estimates $\hat{x}_n$= median(f_n) (lines) are also shown. All plots are constructed using the linear test function (4.1).

To illustrate the above surrogates performance for h_1, Figure 4.3 shows the comparison of the fitted models using a fixed data set $\mathcal{D}_N^\eta := (B_{1:N}^\eta, a_{1:N})$ obtained created with two different η: the sIDS (first row) and the sRQS policies (second row), using the true posterior g_n and fixed batch size $a = 100$ and $T = 20{,}000$ (therefore $N = 200$

total training locations). The Figure depicts three fundamental features of our proposed spatial G-PBA: (i) the IDS strategy minimizes the posterior X^*-uncertainty across all surrogates (represented by different colors in the plot), in contrast to the RQS policy, as seen on the narrower IQR confidence bands depicted in the second column of the plot; (ii) the design of the IDS strategy brackets the true X^*, gradually squeezing the posterior f_n towards the root; and (iii) the spatial surrogates succeed in learning the true

$$\theta_1(x) := \Phi\left(-\frac{1/3 - x}{0.20}\right),$$

especially close to the root X^*, cf. the left panels of the Figure. As a result, root estimation is significantly improved and leads to reliable posterior IQRs, as depicted on the right panels of Figure 4.3.

Adaptive Replication a^{ν}_{n+1}. Recall that the batching scheme (3.16) has two parameters: the *minimum replication amount*, a^{ν}_0, and variance *thresholding sequence*, $(\nu_n)_{n\geq 1}$. In our experiments we use as minimum replication value

$$a^{\nu}_0 := 1, \tag{4.8}$$

in order to favor exploration in regions where the spatial surrogate φ already learned $p(\cdot)$ sufficiently well quantified in terms of the predictive posterior GP variance (3.12b). For the variance thresholding sequence ν_n we use the following two variants:

$$\nu^{(100)}_n := 0.1/n \text{ when } a_0 = 100 \text{ and } \nu^{(250)}_n := 0.05/n \text{ when } a_0 = 250. \tag{4.9}$$

This choice is linked to the fact that when the initial number of train locations N_0 is small, the predictive posterior GP variance s^2_n will be larger and so we take the thresholds ν_n larger, as well.

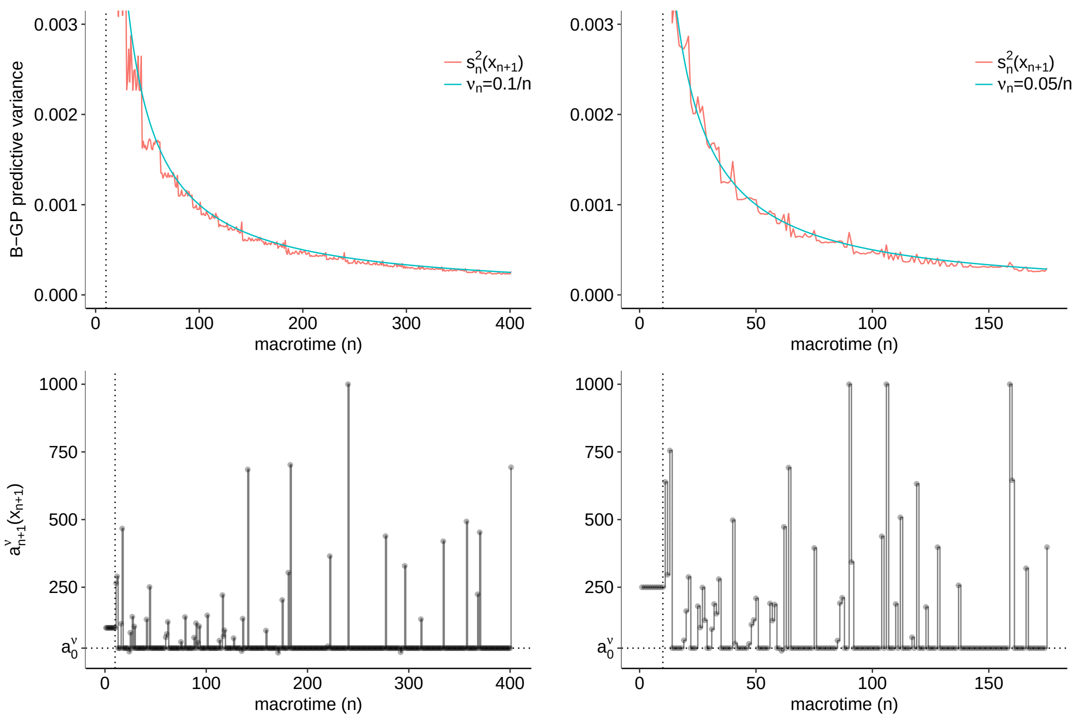

Figure 4.4: *First row*: estimated predictive variance $s_n^2(x_{n+1})$ relative to the thresholding variance sequence $\nu_n^{(100)} = 0.1/n$ (first column) and $\nu_n^{(250)} = 0.05/n$ (second column), when the initial batch size for building the B-GP is $a_0 = 100, 250$, respectively. *Second row*: adaptive replication amount $n \mapsto a^\nu_{n+1}(x_{n+1})$ (y-axis) in macro-time n (x-axis) selecting x_{n+1} using the one-step IDS criterion (3.21).

Figure 4.4 depicts the realized replication amounts $n \mapsto a^\nu_{n+1}(x_{n+1})$ using the one-step sIDS policy (3.21) applied to our running example (4.1) (during initialization, $n \le N_0 := T_0/a_0$, $a_n \equiv a_0$ is fixed). We observe that a^ν_{n+1} generally slowly decreases as n rises, although the local behavior can be quite "spiky": sometimes a large batch is required to bring $s_n^2(x_{n+1})$ below ν_n, see top panels of Figure 4.4.

Remark 9. To avoid excessive batching which could occasionally arise in our implementation, we bound a^ν_{n+1} (e.g., in Figure 4.4 the maximum batch size was restricted to 1000). On the other hand, bounding the replication size also makes it possible to manage the overall sampling budget in order to enforce exploration.

Initial batch size a_0. A key feature of Algorithm 2 is the surrogate initialization stage. In this phase, $N_0 \geq 1$ equally spaced points, $x_{1:N_0}$, are used to learn the surrogate model non-sequentially. In our experiments all surrogates are initialized using $T_0 := 5{,}000$ (i.e., 25% of total sampling budget) oracle evaluations with $a_0 \in \{100, 250\}$, which results in $N_0 := T_0/a_0 \in \{25, 10\}$ *initial training locations.*

4.3.2 Results

Table 4.4: Evaluation metrics for h_1 after $T = 20,000$ iterations using the Spatial G-PBA. The last two rows correspond to adaptive sampling schemes depending on the thresholding sequences ν_n as in (4.9).

η	$\hat{p}$	$\hat{r}^{\eta}_{a}(f_T)$ (10^{-2})		$\hat{l}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{c}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{D}^{\eta}_{a}(f_T; g_T)$	
		a_0=100	a_0=250	a_0=100	a_0=250	a_0=100	a_0=250	a_0=100	a_0=250
sIDS	B-GP	0.2241	0.1874	0.8931	0.9215	0.88	0.98	0.62	0.39
	KLR	0.2106	0.2037	0.8998	0.9496	0.95	0.96	0.57	0.38
	SLR	0.1864	0.1954	0.8669	0.8810	0.87	0.89	0.64	0.59
	LR	0.1956	0.1709	0.8852	0.8708	0.94	0.98	0.56	0.38
sRQS	B-GP	0.2230	0.1985	1.2683	1.3497	0.95	0.99	0.61	0.48
	KLR	0.2152	0.1734	1.2052	1.3843	0.99	0.99	0.51	0.41
	SLR	0.1935	0.2181	1.2027	1.2302	1.00	0.96	0.57	0.60
	LR	0.1840	0.2012	1.2543	1.3174	0.96	0.97	0.56	0.50
sIDS (ν_n)	B-GP	0.2016	0.2060	0.9730	1.0051	0.97	0.96	0.34	0.33
sRQS (ν_n)		0.3025	0.2398	1.4612	1.5013	0.99	1.00	0.16	0.22

Table 4.4 demonstrates that surrogate modeling substantially improves root estimation relative to the original G-PBA, using our running example (4.1). Indeed, we obtain significantly lower residuals (roughly half as big), narrower CI, and maintain a dramatically larger probability coverage across all sampling policies η and batch sizes a, relative to the Local G-PBAs, indicating that f_n is in fact concentrating around to the true root value. Importantly, we can see that spatial modeling leads to a very significant reduction in the average KL divergence between f_T and g_T, $\hat{D}^{\eta}_{a}(f_T; g_T)$, primarily due to the lower bias in the estimation of $p(\cdot)$ with respect to the local estimators. We note that

$\hat{D}^{\eta}_{a}(f_T; g_T)$ is consistently low across all surrogate models, indicating that the goodness-of-fit for $\theta(\cdot)$ is not overly sensitive to the choice of the surrogate type. Table 4.4 conveys the following main conclusions:

- In terms of the sampling policies, sIDS outperforms sRQS since average residuals and CI length are lower while preserving a high coverage probability.
- As mentioned above, all surrogate models yield high goodness-of-fit for $\theta(\cdot)$, as seen by the low KL divergence across all policies η and replication sizes a_0. In particular, we observe that *polynomial logistic regression* (LR) offers the best choice as it minimizes the average residuals and length of CI, as well as it matches the nominal $\hat{c}^{\eta}_{a} \approx 0.95$, confirming that f_T is close to the true posterior g_T across all η.
- For the replication regime a_n, we note a preference for $a = 250$ (with a total of $N = 80$ design sites) since this value tends to yield better learning rates about $p(\cdot)$ (and therefore about X^*) compared to $a = 100$, as measured by the average KL divergence.

Adaptive replication analysis for h_1. Figure 4.5 shows the kernel density estimate for the random variable that denotes the total the number of sampling locations N^{ν}_{T} selected using the sIDS policy across $MC = 100$ Monte-Carlo iterations for each thresholding sequences ν_n as in (4.9) (recall that due to adaptive batching the total number of sampling locations is random and determined by ν_n), applied to the test function (4.1). The vertical lines on the leftmost panel of Figure 4.5 show that the median number of total design points is 320 and 175 for the thresholding sequence $\nu^{(100)}_{n}$ and $\nu^{(250)}_{n}$, respectively. The latter gives us more insight about the effect of ν_n on efficiently exploring the search space: the more restrictive (smaller) the thresholding sequence is, the larger the replication size a^{ν}_{n} is, as confirmed by the overall shape of the estimated distribution of

N_T in the left panel of the Figure. Moreover, another important feature of the adaptive replication scheme (3.16), is that the replication size a_n^ν decreases proportionally to the absolute distance between the estimated root $\hat{x}_n$ and the actual root location X^*, as depicted by the box-plot of the distribution of a_n^ν constructed along 10 ranks of absolute distance between $\hat{x}_N$ and X^* using all MC iterations data. Finally, the right plot of the Figure confirms that the replication size decreases as more samples are placed around x^*, which is measured by the median replication size (solid line) computed across all MC runs (shaded lines) for both thresholding sequences (colors).

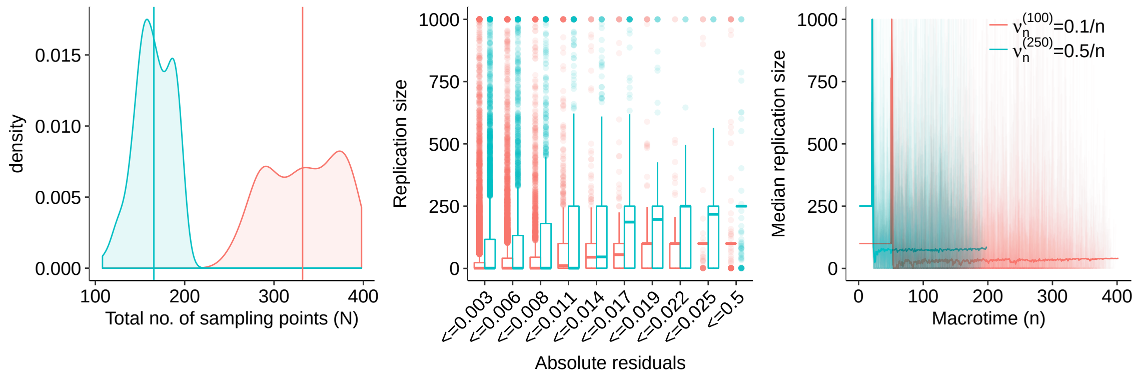

Figure 4.5: *Left*: Kernel density estimate of the distribution of the total number of design points N. *Middle*: Distribution of the replication size across 10 absolute residuals ranks. *Right*: median replication size in macro-time n. All plots corresponds to the linear test function h_1.

Empirical results for the exponential and cubic test functions. Tables 4.5 and 4.6 show the performance metrics for the test functions h_2 and h_3, respectively. As for h_1, we observe a large improvement in performance, especially in terms of the coverage probability $\hat{c}$, which was improved from $\hat{c}_a^\eta \approx 0$ to the actual nominal CI coverage value, see the fifth column of Table 2 and Table 3, respectively. The latter implies that meaning the algorithm succeeds in providing a CI for x^*. In terms of sampling policies we note that sIDS outperforms sRQS, judging by their average absolute residuals and length of CI for both h_2 and h_3. Furthermore, polynomial logistic regression (LR) continues to be the best surrogate choice combined with fixed batch of $a_n = 100$ (i.e., using a total of $N_T = 200$ design points) implying a preference for exploration in these harder problems. We note that B-GP performs worse especially for h_2, possibly due to the non-smooth behavior in θ_2 at the root (cf. Figure 4.1). Because B-GP assumes smooth response surface, it fails to properly capture such "cusp" that calls for a spatially non-stationary covariance structure.

Table 4.5: Evaluation metrics for h_2 after $T = 20,000$ iterations using the Spatial G-PBA. The last two rows correspond to adaptive sampling schemes depending on the thresholding sequences ν_n as in (4.9).

η	$\hat{p}$	$\hat{r}^{\eta}_{a}(f_T)$ (10^{-2})		$\hat{l}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{c}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{D}^{\eta}_{a}(f_T; g_T)$	
		a_0=100	a_0=250	a_0=100	a_0=250	a_0=100	a_0=250	a_0=100	a_0=250
sIDS	B-GP	0.5330	0.4439	1.1043	1.1666	0.45	0.53	1.61	1.41
	KLR	0.4537	0.4098	0.7974	0.8751	0.46	0.44	3.23	2.79
	SLR	0.4352	0.4076	1.0995	1.2555	0.67	0.80	1.21	0.86
	LR	0.3814	0.4128	1.0641	1.1795	0.60	0.56	2.08	1.61
sRQS	B-GP	0.4817	0.5162	1.4630	1.6117	0.70	0.73	1.35	1.42
	KLR	0.4602	0.5440	1.0787	1.3580	0.57	0.60	2.68	1.91
	SLR	0.3956	0.4250	1.6651	1.7434	0.82	0.83	0.93	0.86
	LR	0.4653	0.5143	1.7161	1.5902	0.79	0.67	1.40	1.34
sIDS (ν_n)	B-GP	0.5095	0.4883	1.3638	1.2129	0.49	0.52	1.62	1.83
sRQS (ν_n)		0.5586	0.5088	1.6736	1.7562	0.74	0.77	1.00	0.97

Table 4.6: Evaluation metrics for h_3 after $T = 20,000$ iterations using the Spatial G-PBA. The last two rows correspond to adaptive sampling schemes depending on the thresholding sequences ν_n as in (4.9).

η	$\hat{p}$	$\hat{r}^{\eta}_{a}(f_T)$ (10^{-2})		$\hat{l}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{c}^{\eta}_{a,0.95}(f_T)$ (10^{-2})		$\hat{D}^{\eta}_{a}(f_T; g_T)$	
		a_0=100	a_0=250	a_0=100	a_0=250	a_0=100	a_0=250	a_0=100	a_0=250
sIDS	B-GP	4.3661	4.2959	8.2188	9.7472	0.57	0.64	1.87	1.57
	KLR	4.3403	4.5771	13.0221	12.0456	0.76	0.76	1.32	1.36
	SLR	4.4470	4.6160	7.3444	7.7896	0.49	0.46	2.35	2.23
	LR	3.7645	3.6936	10.6028	10.5738	0.71	0.70	1.52	1.39
sRQS	B-GP	4.1913	4.0209	10.7298	10.8774	0.67	0.67	1.70	1.45
	KLR	3.9131	3.7121	14.2680	14.0897	0.81	0.84	1.17	0.98
	SLR	4.0451	4.1825	10.3663	10.2469	0.69	0.68	1.79	2.03
	LR	3.6513	4.1276	12.9502	11.5623	0.80	0.66	1.27	1.31
sIDS (ν_n)	B-GP	4.1540	4.2334	11.1918	11.3152	0.68	0.67	1.11	1.11
sRQS (ν_n)		4.1874	4.0915	11.6808	12.9052	0.67	0.76	1.39	1.05

4.4 Evaluating the Quality of the Design

Baseline Policies. To focus on the sampling aspect of G-PBA, we examine more closely the designs $x_{1:n}^{(a,\eta)}$ obtained from implementing the policy η (sIDS and sRQS strategies), batch size a, and knowledge state f_n. We then compute the resulting exact posterior $g_n^{(a,\eta)}$ and evaluate the corresponding absolute residual $|\,\text{median}(g_n^{a,\eta}) - x*\,|$ and length of $(1-\alpha)\%$-CI. For this analysis we consider a fixed batch size of $a = 250$. The sIDS, sRQS, and one-step sIDS strategies are benchmarked against the following *baseline* schemes which utilize the true $p(x)$, and therefore the actual posterior density g_n:

$$x_{n+1} := \arg\max_{x\in(0,1)} \mathcal{I}(x, g_n; p(x), a) \tag{IDS}$$

$$x_{n+1} := G_n^{-1}(U_{n+1}), \quad U_{n+1} \sim \mathsf{Unif}(0,1) \tag{RQS}$$

$$x_{n+1} \sim \mathsf{Unif}(0,1). \tag{Unif}$$

Remark 10. The sampling strategy (IDS) is optimal in the sense of maximizing the expected KL distance between g_n and g_{n+a}, and hence we use it as an upper bound on performance; (Unif) is a *passive policy* used as a lower bound.

To make the above baseline policies comparable with the Spatial G-PBA strategies, we implement batched sampling using the transition function (2.6) and $a = 250$. We also match the initialization step, employing $N_0 = T_0/a_0$ equidistant locations $x_{1:N_0}$ (with $T_0 = 5,000$) to construct g_{T_0}, from which (IDS), (RQS) and (Unif) are implemented.

Figure 4.6 compares the baseline strategies (dashed lines) against the Spatial G-PBA using B-GP surrogate (solid lines), applied to our running example (4.1). As expected, we observe that sIDS sampling policy better approximates the true IDS policy. Interestingly, if fixed batch is used, randomized and information-directed policies have asymptotic similar performance, as measured by the average residuals and CI length (in contrast to adaptive batching). Notice also that both Spatial G-PBA policies, dramatically outperform the (Unif) baseline strategy.

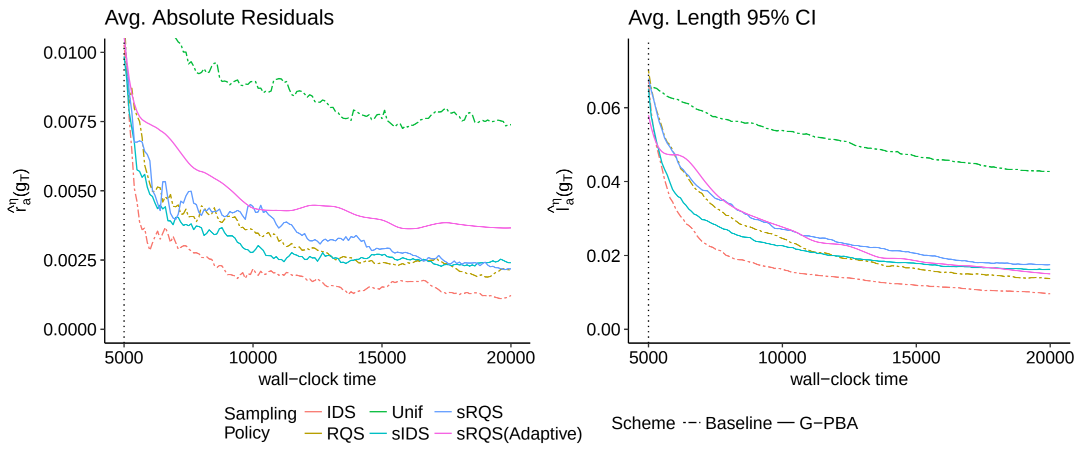

Figure 4.6: Average absolute residuals (y-axis on left panel) and average length of 95% CI (y-axis on right panel) evaluated in wall-clock time (x-axis) obtained utilizing the sampling points generated by the G-PBA policies and evaluate them into the updating model that uses the true $p(\cdot)$ with $a_0 = 250$.

4.5 Case Study: Root-Finding for Optimal Stopping

4.5.1 Introduction

Let us briefly recall a generic discrete-time optimal stopping problem on a finite horizon. Let $X \equiv X_{1:\tilde{T}}$ be a real-valued Markov process generating an information filtration $\mathcal{G} = \sigma(X_{1:t})$. Set $\mathcal{S}$ to be the collection of all $\mathcal{G}$-stopping times smaller than some given horizon $\tilde{T} < \infty$, and $H(t,x)$ the (bounded) reward function for stopping at time $t = 0, 1, \dots, \tilde{T}$. The Optimal Stopping Problem (OSP) consists of maximizing the expected reward $H(\tau, X_\tau)$ over $\tau \in \mathcal{S}$. Towards solving the OSP, define the *value function* $V(t,x) := \sup_{\tau \geq t, \tau \in \mathcal{S}} \mathbb{E}\left[H(\tau, X_\tau) | X_t = x\right]$ for any $0 \leq t \leq \tilde{T}$. Standard dynamic programming arguments imply that $V(t,x) = H(t,x) + \max\{h(x;t), 0\}$ where the function

$$h(x;t) := \mathbb{E}\big[V(t+1, X_{t+1})\,|X_t = x\big] - H(t,x), \tag{4.10}$$

is the *timing value*. It follows that the stopping decision at a given (t, x) is equivalent to comparing $V(t, x)$ and $H(t, x)$: $\boldsymbol{S}_t := \{x : \tau^*(t, x) = t\} = \{x : V(t, x) = H(t, x)\} = \{h(x; t) \leq 0\}$. Thus, it is optimal to stop immediately if and only if the conditional expectation of tomorrow's reward-to-go is less than the immediate reward. Frequently, a priori structure implies that the stopping set $\boldsymbol{S}_t$ above is a half-line, i.e., $h(\cdot; t)$ has a unique root x^*. Consequently, solving the OSP at stage t is equivalent to a root-finding problem for $h(\cdot; t)$.

A stochastic simulation approach (known in the literature as the Longstaff-Schwartz paradigm) recursively builds noisy simulators for $h(t, x)$ over $t = \tilde{T} - 1, \tilde{T} - 2, \ldots$. This is obtained by generating forward paths $x_{t:\tilde{T}}$ of the state process and computing corresponding path-wise stopping times $\tau \equiv \tau(t + 1, x_{t:\tilde{T}})$ (which rely on $\boldsymbol{S}_{t+1:\tilde{T}}$ and hence are recursively known). The realization $z_t(x_t) := H(\tau, x_\tau) - H(t, x_t)$ is the pathwise timing value, i.e., the difference between future and immediate reward over the given trajectory. By construction, $\mathbb{E}[Z_t(x_t)] = h(x_t; t)$ which matches the structure of our PBA oracle (1.1). The random component $\epsilon(x; t)$ arises intrinsically from the randomness in the trajectory $x_{t:\tilde{T}}$. Therefore, the PBA approach offers a novel algorithm to solve one-dimensional optimal stopping problems. Notably, it essentially bypasses standard value-function approximation methods, and allows to directly quantify accuracy of estimated policy $\widehat{\boldsymbol{S}}_t = [0, \hat{x}]$.

4.5.2 G-PBA for Optimal Stopping

As an illustration, we revisit the popular example of a Bermudan Put option within a discretized Black-Scholes model: the reward function is $H(t, x) := e^{-rt}(K^{Put} - x)_+$, (X_t) is a log-normal random walk and $r > 0$ is the interest rate. It is well-known that there is a unique *exercise boundary* $x^*(t) \leq K^{Put}$, and one should exercise as soon as X drops below this boundary: $\boldsymbol{S}_t = [0, x^*(t)]$.

For the practical implementation of the OSP problem introduced above, we take the parameters $K^{Put} = 40, r = 0.06, \tilde{T} = 1$ and restrict to the domain $(25, 40)$ (which is based on some mild domain knowledge, as very low stock prices are known to definitely trigger exercise). Thus, we consider the following oracle (with t fixed):

$$Z^{Put}(x) := h(x;t) + \epsilon(x;t), \qquad x \in [25, 40]; \tag{4.11}$$

where the latent function $h(\cdot;t)$ is the timing value and x is the stock price at date t.

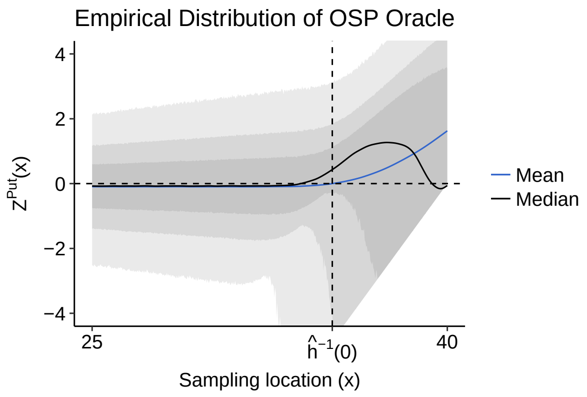

Figure 4.7: Bermudan Put oracle distribution: the fitted mean response $\hat{h}$ (blue line), median $\hat{q}_Z^{0.50}(x)$ (black line) and empirical quantiles (at 1%, 10%, 25%, 75%, 90% and 99% levels, different shaded areas). The root estimate $\hat{h}^{-1}(0;t) \simeq 35.125$ (dashed vertical line) is obtained using Newton-Raphson on the off-line surrogate $\hat{h}$.

Figure 4.7 shows an estimate $\hat{h}(\cdot;t)$ of (4.10), as well as the distribution of $\epsilon(\cdot;t)$. The plot was obtained by fitting an off-line smoothing spline model to 500 pointwise estimates $\hat{h}(x_i;t)$ (equidistant in $(25, 40)$), each obtained from 20,000 oracle calls, i.e., a total of $T = 10^7$ function evaluations. A deterministic root finding procedure (Newton-Raphson) was run to estimate $x^* \simeq 35.1249$ (vertical dashed line) based on the latter $\hat{h}$. This estimate of the root is used as the ground truth in the sequel, although notably it comes without any standard error, being based on a point estimate of $h(\cdot;t)$.

This case-study (4.11) violates the basic PBA assumption of a symmetric noise distribution. Instead, Figure 4.7 demonstrates that $\epsilon(x)$ is right-skewed and heavy-tailed and, in particular, $p(x^*) < 0.5$. Because PBA in fact searches for the point x^*_{med} such that $p(x^*_{med}) = 0.5$, direct use of (4.11) will return the root x^*_{med} of the *median* $\hat{q}^{0.50}_Z(x) := \hat{F}^{-1}_Z(0.50; x)$ (black line in the Figure) rather than the root x^*.

Pre-averaging oracle responses. To resolve the skewness issue, we use a *pre-averaging* procedure that considers the sign of an average of $\zeta > 1$ oracle evaluations,

$$\bar{Y}^{Put}_\zeta(x) := \text{sign}\{\bar{Z}_\zeta(x)\}, \quad \bar{Z}_\zeta(x) := \zeta^{-1} \textstyle\sum_{l=1}^{\zeta} Z_l(x). \tag{4.12}$$

The G-PBA now works with (4.12) and to estimate the corresponding probability of correct response

$$p^{Put}_\zeta(x) := \mathbb{P}(\bar{Y}^{Put}_\zeta(x) = \text{sign}\{x^* - x\}), \tag{4.13}$$

a batch of $\tilde{a} \geq 1$ oracle evaluations in required, considering the signal

$$B^{Put}_\zeta(x) := \sum_{j=1}^{\tilde{a}} 1_{\{\bar{Y}^{Put}_{\zeta,j}(x)=+1\}}.$$

Denoting by a, the total number of oracle queries at x, we have $\tilde{a} = a/\zeta$ for the effective number of replicates to query $\bar{Y}^{Put}_\zeta(x)$. Note that pre-averaging is not needed for functional response aggregation.

The principal role of ζ is to alleviate the skewness of $\epsilon(\cdot)$ from (4.11). Pre-averaging also has the side effect of boosting the signal-to-noise ratio, and hence, boosting the probability of observing correct oracle responses p^{Put}_ζ similar to the majority-vote estimator in (2.21). Overall, the choice of ζ is governed by the above aim of making $\epsilon(\cdot)$ symmetric, as well as the trade-off between sampling many locations to find x^* vis-a-vis proper probabilistic updating of f_n. Our analysis suggests that if the Local G-PBAs are to be used, then take ζ as small as feasible and keep the effective number of replications $\tilde{a}$ relatively large. Conversely, if spatial modeling is used, one then can substantially increase the pre-averaging value, and then use an effective number of replicates $\tilde{a}$ as low as $\tilde{a} = 1$ so that statistical anomalies are alleviated while enforcing exploration.

Lastly, since our earlier analysis assumed a decreasing response, the sign of $Z^{Put}(x)$ is flipped in the sequel.

Baseline root estimation using a RSM approach. As mentioned in Chapter 1, Section 1.1, a practical and viable alternative to find the root of an unknown noisily sampled function, $h(\cdot)$, is to use a response surface modeling (RSM) approach so that a regression/surrogate model $\hat{h}(\cdot)$ is fitted on $h(\cdot)$ in order to subsequently estimate the root via a deterministic root finding (DRF) routine (Newton-Raphson) on $\hat{h}(\cdot)$.

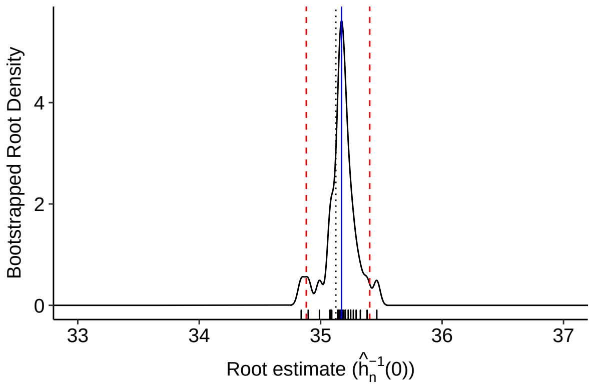

Figure 4.8: Empirical distribution of root estimates, where each point is obtained using a deterministic-root procedure. The final root estimate (blue line) is given by the median of the estimated density, and a 95% CI is provided by the difference between the 2.5% and 97.5% empirical quantiles (red dashed lines). The ground-truth root is given by $\hat{h}^{-1}(0; t) \simeq 35.125$ (dashed vertical line).

To illustrate the RSM setting, we consider the OSP simulator (4.11) and we build a GP *regression* model with a Matérn covariance kernel (3.5); regressing the pre-averaged responses $\bar{Z}_{1:n}^{Put} := (\bar{Z}_1^{Put}(x_1), \ldots, \bar{Z}_n^{Put}(x_n))$ (with $\bar{Z}_i(x_i) := a^{-1} \sum_{j=1}^{a} Z_j(x_i)$ for $x \in (25, 40)$ and $a = 500$), on the history of sampling locations $x_{1:n}$. Fixing the total number of simulation outputs to T = 20,000 and initializing the GP over 10 equally spaced sampling locations on $(25, 40)$, the corresponding estimate of the root will be given by

$\hat{x}_n := \hat{h}_n^{-1}(0)$, where $\hat{h}_n(\cdot)$ is the posterior predictive GP mean [48]. Numerically, $\hat{x}_n$ is obtained using `uniroot()` routine in `R` which employs Newton-Raphson. Furthermore, for this example, the next querying location is also the current root estimate, $x_{n+1} := \hat{h}_n^{-1}(0)$, so sampling concentrates at regions close from where x^* is believed to lie. In the case that at any given iteration a unique root is not found (due to multiple crossings in GP predictive mean), we then consider the median of all previous locations, $x_{n+1} = \text{median}(x_{1:n})$ as new querying location instead. After all T function evaluations are depleted, a final root estimate under the RSM scheme is provided by the median of the history of root estimates, i.e., $\hat{x}_N := \text{median}(\hat{x}_{1:N})$.

Figure 4.8 shows the empirical kernel density constructed on $\hat{x}_{1:n}$. An uncertainty measure for the root x^* can be obtained using a 95% CI via the estimated quantiles (red dashed lines in the Figure). Although this empirical approach can be used as a first order approximation, it relies heavily on the election of the surrogate model $\hat{h}(\cdot)$. Repeating the RSM procedure for $MC = 100$ iterations, we observe that the average absolute residuals are 0.5466, as well as the average length of CI given by 2.5064. In particular, the probability coverage is 31%, indicating that this particular approach fails to provide a reliable estimator for the root uncertainty due to the large mismatch with the nominal coverage of 95%. We now proceed to apply Spatial G-PBA towards solving the OSP problem.

4.5.3 Results

For the numeric evaluation of the Bermudan Put example described in Section 4.5, we employ the inference methods that showed optimal behavior in terms of minimizing overall root uncertainty. In particular, we apply our Local G-PBA using the estimator $\mathscr{P}_{\mathscr{S}}$ (which relies on functional aggregation of oracle responses), as well as all spatial surrogates φ as described in Section 4.3.1.

Furthermore, we continue to use a total simulation budget[1] of $T = 20,000$. To learn the exercise boundary, x^*, we implement the pre-averaged simulator (4.12) using $\zeta := 25$, i.e., each oracle response will consist upon $\zeta = 25$ averaged function evaluations as in (4.12).

Spatial G-PBA for OSP. Algorithm 2 is implemented as follows. We use $a \equiv 500$ total number of oracle calls per sampling location, and hence the effective replication size for learning (4.13) is reduced to $\tilde{a} := a/\zeta = 20$. To initialize all spatial surrogates we use N_0 locations placed equidistantly on the input space $(25, 40)$, and we continue to deploy 25% of the total simulation budget T, implying that total number of design points used to train the initial surrogate, $\hat{\varphi}_{N_0}$, is

$$N_0 := (0.25 \times T)/a = 10.$$

Finally, the remaining $(N_T - N_0)$ design points are chosen applying the sIDS and sRQS strategies combined with the B-GP, KLR, SLR and LR surrogate models.

Adaptive batching is implemented with a minimum replication amount of a_0^ν :=1. The latter is due to the fact that a large pre-averaging value is needed in order to alleviate the skewness of $Z^{Put}(x)$, and therefore we will keep the replication size as *low* as possible in order to incentivize exploration – especially at regions where $\theta_\zeta(\cdot)$ has been learned sufficiently well. As variance thresholding sequence we use $\nu_n^{Put} := 0.50/n$, where the effective number of replicates under this scheme is given by $\tilde{a}_{n+1}^\nu = \min\{a_{n+1}^\nu, a/\zeta\}$ for $n = n_0, n_0 + 1, \ldots$ (we bound a_{n+1}^ν so that the maximum number of function evaluation is still no more than $a = 500$ per querying location).

[1]Due to the non-standard noise component and very low signal-to-noise ratio, this is a difficult root-finding problem, comparable to test case h_3; in particular the simulation budget is quite low.

Table 4.7: Average Monte-Carlo performance statistics after $T = 20,000$ simulation outputs. The Local G-PBA policy RQS is presented in the last row of the table coupled with the functional aggregation procedure $\mathscr{P}_{\mathscr{S}}$.

η	$\hat{p}$	$\hat{r}^{\eta}_{a}(f_T)$	$\hat{l}^{\eta}_{a,0.95}(f_T)$	$\hat{c}^{\eta}_{a,0.95}(f_T)$ (in %)
sIDS	B-GP	0.3210	0.8903	69.00
	KLR	0.3598	0.6351	53.33
	SLR	0.3158	0.9878	77.00
	LR	0.2753	1.0687	88.50
sRQS	B-GP	0.2988	1.4064	86.00
	KLR	0.3121	0.8209	62.00
	SLR	0.3180	1.2005	74.50
	LR	0.2913	1.4039	90.50
sIDS (ν_n)	B-GP	0.2225	0.6944	80.00
sRQS (ν_n)		0.3011	1.1121	76.47
Rand-Q	$\mathscr{P}_{\mathscr{S}}$	0.271	2.0880	97.00

Table 4.7 shows the average residuals, length of CI, and coverage probability of the Spatial G-PBA schemes compared against the baseline root location $\hat{x}^* \simeq 35.1249$. This time, adaptive replication with the one-step sIDS policy (3.21) is the best-performing scheme. One reason could be that it allows for more sampling locations (median number of sampling locations was median$(N_T) = 55$, as opposed to 40 for the fixed a_n schemes). Among the rest, sIDS policy coupled with the quintic polynomial logistic regression model (LR) performs sufficiently well, consistent with our findings in the earlier synthetic experiments. Relative to the non-spatial G-PBA (which implements functional aggregation via the estimator $\mathscr{P}_{\mathscr{S}}$ and the Rand-Q sampling policy given by (2.24) and (2.34), respectively) two important improvements are noted: (i) much better coverage probabilities, indicating the gains in learning $p(\cdot)$ and hence maintaining a reliable knowledge state; (ii) lower residuals (below 0.25 while they were about 0.35).

Chapter 5

Conclusions

We have developed a collection of numerical schemes that aim to solve the Stochastic Root-Finding Problem (SRFP). To do so, we generalized the probabilistic bisection algorithm (PBA) to the setting where the statistical properties of the oracle responses are unknown and location-dependent.

The main ingredients of our generalized PBA (G-PBA) paradigm can be summarized in three different components:

(C1) The explicit construction of estimators for the oracle accuracy $p(\cdot)$ (i.e., the probability of observing a correct response) by using batched sampling at each querying location;

(C2) the introduction of novel sampling strategies that are capable of maximizing the information about the root location at each stage; and

(C3) the explicit construction of a knowledge state variable, which condenses our beliefs about the root location, and serves the dual purposes of estimating the root and selecting the next location at which to query the oracle.

The above three components constitute the main contributions of this work, and naturally extend the classical PBA for SRFPs presented in Waeber et al. [58].

Below we outline in detail the main conclusions with regard to (C1), (C2), and (C3).

For (C1), we introduced a family of estimators for $p(\cdot)$ that do not use information from previous locations. As such, we demonstrated their robustness to arbitrary specification of $p(\cdot)$. Through our extensive numeric examples, we showed that they can be viewed as making minimal assumptions about the oracle. Additionally, we documented the significant advantage of using functional responses (via functional aggregation) relative to utilizing just the signs of the responses. Otherwise, we concluded that the empirical majority proportion or a conservative Bayes-like are good choices.

Moreover, we presented a structured extension of the above statistical local procedures. In particular, we used *spatial* surrogate modeling for $p(\cdot)$ in order to incorporate knowledge acquired from previous sampling locations. This was achieved by regressing the observed (batched) responses on the previously seen sampling locations. We then demonstrated that such blend of a regression-type paradigm with G-PBA, improved the accuracy in the root estimation. Namely, absolute residuals actually decay faster than their corresponding local G-PBA methods and, most importantly, the probability coverage is improved from zero to the actual nominal values, relative to the methods which only use local information.

For (C2), one take-away is the advantage of Randomized Quantile sampling (RQS) against Information Gain approaches. Namely, we observed that selecting locations randomizing according to f_n, minimized the need to construct a high-fidelity estimator for $p(\cdot)$ during the design construction, and hence made more efficient use of oracle queries. Our RQS design coupled with G-PBA, mimics the success of *Thompson sampling* in other learning contexts, since it is an efficient heuristic for balancing the exploration/exploitation trade-off, as well as it improves the learning rate in the early stages by better exploring the posterior of X^*. Nevertheless,if Spatial G-PBA is used, we showed that the spatial structure yields two key benefits in terms of sampling policies. Firstly, given the surrogate, the Information Gain approaches, namely the IDS criterion, $\mathcal{I}$ can be predicted for any x, allowing direct optimization of next querying site selection like in standard PBA. Secondly, Binomial GPs allows adaptive batching schemes to automatically fine-tune exploration by reducing replication amounts in regions where $p(\cdot)$ is already learned well. Our numeric experiments confirmed the advantages of Spatial G-PBA relative to the our Local G-PBA, with the new algorithm inducing more accurate root estimates and better quantifying the posterior uncertainty about the root.

Looking ahead, one motivation for considering PBA in the context of SRFP is its Bayesian flavor that allows in particular to apply informative priors f_0 as a way to warm-start the root search. This offers one way to lift PBA, which is intrinsically limited to one-dimensional setting, to higher-dimensions. The analogue of SRFP in two-dimensions is noisy (zero-)contour-finding, which can be viewed as a collection of root-finding problems in the first coordinate x_1, indexed by the second coordinate x_2. Assuming the zero-contour is smooth one may then try to solve for a few $x^*(x_2)$ and then "connect the dots" through interpolation (or a further surrogate model). Such searches can be made efficient with G-PBA by using $f_N(\cdot; x_2)$ as a basis for an informative prior $f_0(\cdot; x_2')$ at a new x_2'. We leave such investigations to future research.

Appendix A

Additional Results

A.1 Binomial GPs and Laplace Approximation

Binomial log-likelihood Gradient and Hessian. The number of positive responses $B_i := \sum_{j=1}^{a_i} 1_{\{Z_i>0\}}$ after $a_i \geq 1$ replicates at location x_i, follows a binomial distribution $B_i|a_i, \varphi_i, \overset{iid}{\sim} \mathsf{Bin}(a_i, \theta(\varphi_i))$ with log-likelihood function (in the latent value φ_i):

$$\log p(B_{1:n}|\varphi_{1:n}, a_{1:n}) = \sum_{j=1}^{n} \left\{ \log \binom{a_i}{B_i} + B_i \log \theta(\varphi_i) + (a_i - B_i) \log[1 - \theta(\varphi_i)] \right\}.$$

Using the Bernoulli link function $\theta(\varphi_i) = (1 + e^{-\varphi_i})^{-1}$ which implies that $\theta'(\varphi_i) := \theta(\varphi_i)[1 - \theta(\varphi_i)]$, the corresponding gradient vector $\boldsymbol{u}_n(\varphi_{1:n}) := \nabla l(\varphi_{1:n})$ is given by

$$\begin{aligned} u_i(\varphi_i) &= B_i \frac{\theta'(\varphi_i)}{\theta(\varphi_i)} - (a_i - B_i) \frac{\theta'(\varphi_i)}{1 - \theta(\varphi_i)} && \text{(A.1)} \\ &= B_i[1 - \theta(\varphi_i)] + (B_i - a_i)\theta(\varphi_i) \\ &= B_i - a_i\theta(\varphi_i), \quad i = 1, \ldots, n. && \text{(A.2)} \end{aligned}$$

Differentiating $\boldsymbol{u}_n$ again yields the $n \times n$ Hessian matrix $\boldsymbol{W}_n(\varphi_{1:n}) = -\Delta \log p(B_{1:n}|\varphi_{1:n}, K_{1:n})$ as specified in (3.11).

Normal Approximation to the Joint Posterior Distribution. By Bayes' rule the posterior $p(\varphi_{1:n}|\mathcal{D}_n)$ is proportional to the binomial likelihood $p(B_{1:n}|\varphi_{1:n}, a_{1:n}, x_{1:n})$ times the zero-mean GP prior $p(\varphi_{1:n}|x_{1:n})$. Taking the log of the unnormalized joint posterior we obtain

$$\begin{aligned}\mathscr{L}(\varphi_{1:n}) &\propto \log p(B_{1:n}|\varphi_{1:n}, a_{1:n}, x_{1:n}) + \log p(\varphi_{1:n}|x_{1:n}) \\ &:= \log p(B_{1:n}|\varphi_{1:n}, a_{1:n}, x_{1:n}) - \frac{1}{2}\varphi_{1:n}^T \boldsymbol{K}_n^{-1}\varphi_{1:n} - \frac{1}{2}\log|\boldsymbol{K}_n| - \frac{n}{2}\log 2\pi. \quad \text{(A.3)}\end{aligned}$$

Denote by $\hat{\boldsymbol{\varphi}}_n := \arg\max_{\boldsymbol{\varphi}_n} \mathscr{L}(\boldsymbol{\varphi}_n) = \arg\max_{\boldsymbol{\varphi}_n} p(\boldsymbol{\varphi}_n|\mathcal{D}_n)$. Expanding $\mathscr{L}(\cdot)$ around $\hat{\boldsymbol{\varphi}}_n$ gives $\mathscr{L}(\boldsymbol{\varphi}_n) = \mathscr{L}(\hat{\boldsymbol{\varphi}}_n) + \frac{1}{2}(\boldsymbol{\varphi}_n - \hat{\boldsymbol{\varphi}}_n)^T[\Delta\mathscr{L}(\hat{\boldsymbol{\varphi}}_n)](\boldsymbol{\varphi}_n - \hat{\boldsymbol{\varphi}}_n) + \cdots$; where the linear term in the expansion is zero because the log-posterior density has zero derivative at its mode. As discussed in [21], the remainder terms of higher order fade in importance relative to the quadratic term when $\boldsymbol{\varphi}_n$ is close to $\hat{\boldsymbol{\varphi}}_n$ and the sample size n is large.

Taking first and second partial derivatives of $\mathscr{L}(\varphi_{1:n})$ with respect to $\varphi_{1:n}$ and combining with (A.2)-(3.11) we obtain:

$$\nabla\mathscr{L}(\varphi_{1:n}) = \boldsymbol{u}_n(\varphi_{1:n}) - \boldsymbol{K}_n^{-1}\varphi_{1:n}, \quad \text{(A.4)}$$

$$\Delta\mathscr{L}(\varphi_{1:n}) = -\boldsymbol{W}_n(\varphi_{1:n}) - \boldsymbol{K}_n^{-1}; \quad \text{(A.5)}$$

At the mode of $\mathscr{L}(\varphi_{1:n})$ we have

$$\nabla\mathscr{L}(\hat{\boldsymbol{\varphi}}_n) = \boldsymbol{0} \quad \Rightarrow \quad \hat{\boldsymbol{\varphi}}_n = \boldsymbol{K}_n\boldsymbol{u}_n(\hat{\boldsymbol{\varphi}}_n) \quad \text{(A.6)}$$

as a self-consistent equation for $\hat{\boldsymbol{\varphi}}_n$. Next, the Hessian of the score function $\Delta\mathscr{L}(\varphi_n)$ is interpreted as the inverse covariance matrix, leading to the Gaussian approximation $q(\cdot|\mathcal{D}_n)$ to the true posterior $p(\cdot|\mathcal{D}_n)$

$$q(\cdot|\mathcal{D}_n) \equiv \mathsf{N}(\cdot; \hat{\boldsymbol{\varphi}}_n, (\boldsymbol{K}_n^{-1} + \boldsymbol{W}_n(\hat{\boldsymbol{\varphi}}_n))^{-1}), \quad \text{(A.7)}$$

that is, $q(\cdot|\mathcal{D}_n, \hat{\boldsymbol{\varphi}}_n)$ is a Gaussian distribution with mean $\hat{\boldsymbol{\varphi}}_n$ and covariance matrix $\boldsymbol{\Sigma}_n \equiv (\boldsymbol{K}_n^{-1} + \boldsymbol{W}_n(\hat{\boldsymbol{\varphi}}_n))^{-1}$.

Since $\boldsymbol{u}_n(\boldsymbol{\varphi}_n^*)$ is a non-linear function in $\boldsymbol{\varphi}_n$, $\nabla\mathscr{L}(\boldsymbol{\varphi}_n^*) = \mathbf{0}$ cannot be solved directly. In order to solve (A.6), a numeric approximation, $\hat{\boldsymbol{\varphi}}_n$, for $\boldsymbol{\varphi}_n^*$ is obtained using an iterative updating procedure based on classical Newton-Raphson search.

Predictive distribution. The approximated predictive pdf $\varphi_* := \varphi_*(x)$ at a test location $x \in (0,1)$ is Gaussian $\varphi_* \sim \mathsf{N}(m_n(x;\hat{\boldsymbol{\varphi}}_n), s_n^2(x;\hat{\boldsymbol{\varphi}}_n))$ with the mean $m_n(x;\hat{\boldsymbol{\varphi}}_n)$ given by:

$$\begin{aligned} m_n(x;\hat{\boldsymbol{\varphi}}_n) &:= \int \mathbb{E}[\varphi(x)|\tilde{\varphi}_{1:n}]p(\tilde{\varphi}_{1:n}|\mathcal{D}_n)d\tilde{\varphi}_{1:n} \\ &= \boldsymbol{\kappa}_n^T \boldsymbol{K}_n^{-1} \int \tilde{\varphi}_{1:n} p(\tilde{\varphi}_{1:n}|\mathcal{D}_n)d\tilde{\varphi}_{1:n} \\ &= \boldsymbol{\kappa}_n^T \boldsymbol{K}_n^{-1}\mathbb{E}[\varphi_{1:n}|\mathcal{D}_n] \simeq \boldsymbol{\kappa}_n^T \boldsymbol{K}_n^{-1}\hat{\boldsymbol{\varphi}}_n; \end{aligned} \tag{A.8}$$

where $\boldsymbol{\kappa}_n^T \equiv (\kappa(x_1,x),\ldots,\kappa(x_n,x))$, matching (3.12a). Likewise, the approximated predictive variance, $s_n(x;\hat{\boldsymbol{\varphi}}_n) \equiv \mathbb{V}ar(\varphi(x)|\mathcal{D}_n,\hat{\boldsymbol{\varphi}}_n,x)$, is given by (cf. (3.12b)):

$$\begin{aligned} s_n(x;\hat{\boldsymbol{\varphi}}_n) &:= \mathbb{E}[\mathbb{V}ar(\varphi(x)|\varphi_{1:n},x_{1:n})|\mathcal{D}_n] + \mathbb{V}ar(\mathbb{E}[\varphi(x)|\varphi_{1:n},x_{1:n},x]|\mathcal{D}_n) \\ &= \mathbb{E}[\tau^2 - \boldsymbol{\kappa}_n^T\boldsymbol{K}_n^{-1}\boldsymbol{\kappa}_n|\mathcal{D}_n] + \mathbb{V}ar(\boldsymbol{\kappa}_n^T\boldsymbol{\kappa}_n^{-1}\varphi_{1:n}|\mathcal{D}_n) \\ &= \tau^2 - \boldsymbol{\kappa}_n^T\boldsymbol{K}_n^{-1}\boldsymbol{\kappa}_n + \boldsymbol{\kappa}_n^T\boldsymbol{K}_n^{-1}\mathbb{V}ar(\varphi_{1:n}|\mathcal{D}_n)\boldsymbol{K}_n^{-1}\boldsymbol{\kappa}_n \\ &\simeq \tau^2 - \boldsymbol{\kappa}_n^T\boldsymbol{K}_n^{-1}\boldsymbol{\kappa}_n + \boldsymbol{\kappa}_n^T\boldsymbol{K}_n^{-1}(\boldsymbol{K}_n^{-1} + \boldsymbol{W}_n(\hat{\boldsymbol{\varphi}}_n))^{-1}\boldsymbol{K}_n^{-1}\boldsymbol{\kappa}_n \\ &= \tau^2 - \boldsymbol{\kappa}_n^T(\boldsymbol{K}_n + \boldsymbol{W}_n(\hat{\boldsymbol{\varphi}}_n)^{-1})^{-1}\boldsymbol{\kappa}_n, \end{aligned}$$

where the last line is true via the matrix inversion lemma applied to $(\boldsymbol{K}_n + \boldsymbol{W}_n(\hat{\boldsymbol{\varphi}}_n)^{-1})^{-1}$:

$$\begin{aligned} \boldsymbol{\kappa}_n^T(\boldsymbol{K}_n + \boldsymbol{W}_n(\hat{\boldsymbol{\varphi}}_n)^{-1})^{-1}\boldsymbol{\kappa}_n &= \boldsymbol{\kappa}_n^T\{\boldsymbol{K}_n^{-1} - \boldsymbol{K}_n^{-1}(\boldsymbol{K}_n^{-1} + \boldsymbol{W}_n(\hat{\boldsymbol{\varphi}}_n))^{-1}\boldsymbol{K}_n^{-1}\}\boldsymbol{\kappa}_n \\ &= \boldsymbol{\kappa}_n^T\boldsymbol{K}_n^{-1} - \boldsymbol{\kappa}_n^T(x)\boldsymbol{K}_n^{-1}(\boldsymbol{K}_n^{-1} + \boldsymbol{W}_n(\hat{\boldsymbol{\varphi}}_n))^{-1}\boldsymbol{K}_n^{-1}\boldsymbol{\kappa}_n; \end{aligned}$$

A.2 Predictive Variance Decomposition for Binomial GPs

Theorem 3.2.2. Set

$$\hat{\boldsymbol{\varphi}}_{n+1} \equiv (\hat{\varphi}_{1;n+1}, \ldots, \hat{\varphi}_{n+1;n+1})$$

to be the $(n+1)$-dimensional estimated mode based on training data $\mathcal{D}_{n+1}$ obtained at locations $x_{1:n+1}$; and let

$$\hat{\boldsymbol{W}}_{n+1;n+1} := \mathrm{diag}\{\hat{w}_{1;n+1}, \ldots, \hat{w}_{n+1;n+1}\}, \quad \hat{w}_i \equiv w_i(\hat{\varphi}_{i;n+1})$$

be the Hessian matrix (3.11) evaluated at elements of the estimated posterior $\hat{\boldsymbol{\varphi}}_{n+1}$. Then, we have that the covariance matrix $\boldsymbol{\Sigma}_{n+1} \equiv (\boldsymbol{K}_{n+1} + \hat{\boldsymbol{W}}_{n+1})^{-1})$ of the joint approximated posterior (A.7) can be partitioned as:

$$\boldsymbol{\Sigma}_{n+1} = \begin{pmatrix} \boldsymbol{\Sigma}_{1:n;n+1} & \boldsymbol{\kappa}_n^* \\ (\boldsymbol{\kappa}_n^*)^T & \tau^2 + \hat{w}_{n+1;n+1}^{-1} \end{pmatrix}; \tag{A.9}$$

where $\boldsymbol{\kappa}_n^* := (\kappa(x_1, x_{n+1}), \ldots, \kappa(x_n, x_{n+1}))^T$ is a $n \times 1$ column vector of covariances of $\varphi_{1:n}$ against φ_{n+1}, and $\tau^2 = \kappa(x_{n+1}, x_{n+1})$ is a non-negative scalar. Applying the Matrix Inversion Theorem [26], the inverse of (A.9) is given by:

$$\begin{aligned}
\boldsymbol{\Sigma}_{n+1}^{-1} &= \begin{pmatrix} \boldsymbol{\Sigma}_{1:n;n+1}^{-1} + (\boldsymbol{\Sigma}_{1:n;n+1}^{-1}\boldsymbol{\kappa}_n^*)(\boldsymbol{\Sigma}_{n;n+1}^{-1}\boldsymbol{\kappa}_n^*)^T a^{-1} & -\boldsymbol{\Sigma}_{1:n;n+1}^{-1}\boldsymbol{\kappa}_n^* a^{-1} \\ -(\boldsymbol{\Sigma}_{1:n;n+1}^{-1}\boldsymbol{\kappa}_n^*)^T a^{-1} & a^{-1} \end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{\Sigma}_{n;n+1}^{-1} & 0 \\ 0 & 0 \end{pmatrix} + a^{-1} \begin{pmatrix} (\boldsymbol{\Sigma}_{n;n+1}^{-1}\boldsymbol{\kappa}_n^*)(\boldsymbol{\Sigma}_{n;n+1}^{-1}\boldsymbol{\kappa}_n^*)^T & -\boldsymbol{\Sigma}_{n;n+1}^{-1}\boldsymbol{\kappa}_n^* \\ -(\boldsymbol{\Sigma}_{n;n+1}^{-1}\boldsymbol{\kappa}_n^*)^T & 1 \end{pmatrix},
\end{aligned}$$

where the scalar is $a = \hat{w}_{n+1}^{-1} + s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{n;n+1})$, since

$$\begin{aligned}
a &:= [(\tau^2 + \hat{w}_{n+1;n+1}^{-1}) - (\boldsymbol{\kappa}_n^*)^T \boldsymbol{\Sigma}_{n;n+1}^{-1} \boldsymbol{\kappa}_n^*] \\
&= [\hat{w}_{n+1;n+1}^{-1} + (\tau^2 - (\boldsymbol{\kappa}_n^*)^T \boldsymbol{\Sigma}_{n;n+1}^{-1} \boldsymbol{\kappa}_n^*)]
\end{aligned}$$

$$= [\hat{w}_{n+1}^{-1} + s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n,n+1})].$$

Substituting the expression for $\boldsymbol{\Sigma}_{n+1}^{-1}$ obtained above in the predictive variance formula (3.12b), we have that the posterior predictive variance at time $n+1$ and given the data set $\mathcal{D}_{n+1}$ is:

$$\begin{aligned}
s_{n+1}^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{n+1}) &:= \tau^2 - (\boldsymbol{\kappa}_n^*)^T \boldsymbol{\Sigma}_{n+1}^{-1} \boldsymbol{\kappa}_n^*, \quad \boldsymbol{\kappa}_n^* \equiv (\kappa(x_1, x_{n+1}), \dots, \kappa(x_n, x_{n+1})) \\
&= \tau^2 - \boldsymbol{u}^T \left\{ \begin{pmatrix} \boldsymbol{\Sigma}_{1:n;n+1}^{-1} & 0 \\ 0 & 0 \end{pmatrix} + a^{-1} \begin{pmatrix} \boldsymbol{v}\boldsymbol{v}^T & -\boldsymbol{v} \\ -\boldsymbol{v}^T & 1 \end{pmatrix} \right\} \boldsymbol{u} \\
&= \tau^2 - \boldsymbol{u}^T \begin{pmatrix} \boldsymbol{\Sigma}_{n;n+1}^{-1} & 0 \\ 0 & 0 \end{pmatrix} \boldsymbol{u} \\
&- a^{-1} \boldsymbol{u}^T \begin{pmatrix} \boldsymbol{v}\boldsymbol{v}^T & -\boldsymbol{v} \\ -\boldsymbol{v}^T & 1 \end{pmatrix} \boldsymbol{u} \\
&= \left[\tau^2 - (\boldsymbol{\kappa}_n^*)^T \boldsymbol{\Sigma}_{n;n+1}^{-1} \boldsymbol{\kappa}_n^*\right] - a^{-1}[b^2 - b\tau^2 - \tau^2 b + \tau^4], \quad b_{1\times 1} \equiv (\boldsymbol{\kappa}_n^*)^T \boldsymbol{v} \\
&= s_n^2(x_{n+1}) - a^{-1}(\tau^2 - b)^2;
\end{aligned}$$

where we set $\boldsymbol{v}_{n\times 1} \equiv \boldsymbol{\Sigma}_{1:n;n+1}^{-1} \boldsymbol{\kappa}_n^*$ and let $\boldsymbol{u}_{(n+1)\times 1} \equiv (\boldsymbol{\kappa}_n^* \ \tau^2)^T$ be the concatenation of the vector $\boldsymbol{\kappa}_n^*$ and the scalar τ^2. Simplifying the expression above, we then obtain that the posterior predictive variance at time $n+1$ can be expressed by:

$$\begin{aligned}
s_{n+1}^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{n+1}) &:= s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1}) - a^{-1}(\tau^2 - (\boldsymbol{\kappa}_n^*)^T \boldsymbol{\Sigma}_{1:n;n+1}^{-1} \boldsymbol{\kappa}_n^*)^2 \\
&= s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1}) - a^{-1}(s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1}))^2 \\
&= s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1})(1 - \frac{s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1})}{\hat{w}_{n+1;n+1}^{-1} + s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1})}) \\
&= \frac{s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1}) \hat{w}_{n+1;n+1}^{-1}}{w_{n+1;n+1}^{-1} + s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1})} \\
&= \left(\frac{1}{s_n^2(x_{n+1}; \hat{\boldsymbol{\varphi}}_{1:n;n+1})} + \frac{1}{\hat{w}_{n+1;n+1}^{-1}} \right)^{-1}.
\end{aligned}$$

Finally, we notice that

$$\begin{aligned}\hat{w}_{n+1;n+1} &:= K_{n+1}\theta(\hat{\varphi}_{n+1;n+1})(1-\theta(\hat{\varphi}_{n+1;n+1})) \\ &= K_{n+1}\cdot v_{n+1}(\theta(\hat{\varphi}_{n+1,n+1}))\end{aligned}$$

which leads to (3.14). □

Bibliography

[1] Yasemin Altun, Thomas Hofmann, and Alexander J Smola. Gaussian process classification for segmenting and annotating sequences. In *Proceedings of the twenty-first international Conference on Machine Learning*, page 4. ACM, 2004.

[2] Bruce Ankenman, Barry L Nelson, and Jeremy Staum. Stochastic kriging for simulation metamodeling. *Operations research*, 58(2):371–382, 2010.

[3] Dario Azzimonti, Julien Bect, Clément Chevalier, and David Ginsbourger. Quantifying uncertainties on excursion sets under a Gaussian random field prior. *SIAM/ASA J. Uncertainty Quantification*, 4(1):850–874, 2016.

[4] Julien Bect, David Ginsbourger, Ling Li, Victor Picheny, and Emmanuel Vazquez. Sequential design of computer experiments for the estimation of a probability of failure. *Statistics and Computing*, 22(3):773–793, 2012.

[5] Albert Benveniste, Michel Métivier, and Pierre Priouret. *Adaptive algorithms and stochastic approximations*, volume 22. Springer Science & Business Media, 2012.

[6] Dimitri P Bertsekas, Dimitri P Bertsekas, Dimitri P Bertsekas, and Dimitri P Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena scientific Belmont, MA, 1995.

[7] Mickaël Binois, Robert B. Gramacy, and Mike Ludkovski. Practical heteroskedastic Gaussian process modeling for large simulation experiments. *Journal of Computational and Graphical Statistics*, 0(ja):1–41, 2018.

[8] Mickaël Binois, Jiangeng Huang, Robert B. Gramacy, and Mike Ludkovski. Replication or exploration? Sequential design for stochastic simulation experiments. *Technometrics*, 0(ja):1–43, 2018.

[9] Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[10] George Casella and Roger L Berger. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

[11] Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical Science*, pages 273–304, 1995.

[12] Han-Fu Chen. *Stochastic approximation and its applications*, volume 64. Springer Science & Business Media, 2006.

[13] Xi Chen and Qiang Zhou. Sequential design strategies for mean response surface metamodeling via stochastic kriging with adaptive exploration and exploitation. *European Journal of Operational Research*, 262(2):575–585, 2017.

[14] Clément Chevalier, Julien Bect, David Ginsbourger, Emmanuel Vazquez, Victor Picheny, and Yann Richet. Fast parallel kriging-based stepwise uncertainty reduction with application to the identification of an excursion set. *Technometrics*, 56(4):455–465, 2014.

[15] Wei Chu, Zoubin Ghahramani, Francesco Falciani, and David L Wild. Biomarker discovery in microarray gene expression data with Gaussian processes. *Bioinformatics*, 21(16):3385–3393, 2005.

[16] Thomas M. Cover and Joy A. Thomas. *Elements of Information heory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[17] Marc P Deisenroth, Jan Peters, and Carl E Rasmussen. Approximate dynamic programming with Gaussian processes. In *American Control Conference, 2008*, pages 4480–4485. IEEE, 2008.

[18] David John Finney et al. *Statistical method in biological assay*, volume 8. JSTOR, 1952.

[19] Peter I Frazier, Shane G Henderson, and Rolf Waeber. Probabilistic Bisection Converges Almost as Quickly as Stochastic Approximation. *arXiv preprint arXiv:1612.03964*, 2016.

[20] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The Elements of Statistical Learning*, volume 1. Springer series in Statistics New York, 2001.

[21] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*, volume 2. Taylor & Francis, 2014.

[22] Robert B. Gramacy and Herbert K H. Lee. Adaptive Design and Analysis of Supercomputer Experiments. *Technometrics*, 51(2):130–145, May 2009.

[23] Robert B Gramacy and Mike Ludkovski. Sequential design for optimal stopping problems. *SIAM Journal on Financial Mathematics*, 6(1):748–775, 2015.

[24] J Harold, G Kushner, and Yin George. Stochastic Approximation Algorithms and Applications, 1997.

[25] Kevin Healy and Lee W Schruben. Retrospective simulation response optimization. In *Simulation Conference, 1991. Proceedings., Winter*, pages 901–906. IEEE, 1991.

[26] Harold V Henderson and Shayle R Searle. On deriving the inverse of a sum of matrices. *SIAM Review*, 23(1):53–60, 1981.

[27] José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems*, pages 918–926, 2014.

[28] Bruno Jedynak, Peter I Frazier, Raphael Sznitman, et al. Twenty questions with noise: Bayes optimal policies for entropy loss. *Journal of Applied Probability*, 49(1):114–136, 2012.

[29] Steven G Johnson. The NLopt nonlinear-optimization package, 2014.

[30] Donald R Jones, Cary D Perttunen, and Bruce E Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.

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

[32] V Roshan Joseph, Yubin Tian, and CF Jeff Wu. Adaptive designs for Stochastic Root-Finding. *Statistica Sinica*, 17(4):1549, 2007.

[33] V Roshan Joseph and CFJ Wu. Operating window experiments: a novel approach to quality improvement. *Journal of Quality Technology*, 34(4):345, 2002.

[34] Bogumił Kamiński. A method for the updating of stochastic kriging metamodels. *European Journal of Operational Research*, 247(3):859–866, 2015.

[35] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Active learning with Gaussian processes for object categorization. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[36] Jack PC Kleijnen et al. *Design and analysis of Simulation Experiments*, volume 20. Springer, 2008.

[37] Louisa Lam and Ching Y Suen. A theoretical analysis of the application of majority voting to pattern recognition. In *Pattern Recognition, 1994. Vol. 2-Conference B: Computer Vision &amp; Image Processing., Proceedings of the 12th IAPR International. Conference on*, volume 2, pages 418–420. IEEE, 1994.

[38] Francis A Longstaff and Eduardo S Schwartz. Valuing American options by simulation: a simple least-squares approach. *The Review of Financial Studies*, 14(1):113–147, 2001.

[39] Mike Ludkovski. Kriging Metamodels and Experimental Design for Bermudan Option Pricing. *Journal of Computational Finance*, to appear.

[40] Thomas P Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 362–369. Morgan Kaufmann Publishers Inc., 2001.

[41] Barry T Never. AD-Optimality-Based Sensitivity Test. *Technometrics*, 36(1):61–70, 1994.

[42] Hannes Nickisch and Carl Edward Rasmussen. Approximations for binary Gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078, 2008.

[43] Raghu Pasupathy and Sujin Kim. The Stochastic Root-Finding Problem: Overview, Solutions, and Open Questions. *ACM Trans. Model. Comput. Simul.*, 21(3):19:1–19:23, February 2011.

[44] Victor Picheny, David Ginsbourger, Yann Richet, and Gregory Caplin. Quantile-based optimization of noisy computer experiments with tunable precision. *Technometrics*, 55(1):2–13, 2013.

[45] Warren B Powell and Ilya O Ryzhov. *Optimal learning*, volume 841. John Wiley & Sons, 2012.

[46] R Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2016.

[47] Pritam Ranjan, Derek Bingham, and George Michailidis. Sequential experiment design for contour estimation from complex computer codes. *Technometrics*, 50(4):527–541, 2008.

[48] Carl Edward Rasmussen. Gaussian processes in Machine Learning. In *Advanced lectures on Machine Learning*, pages 63–71. Springer, 2004.

[49] Jaakko Riihimäki and Aki Vehtari. Gaussian processes with monotonicity information. In *AISTATS*, volume 9, pages 645–652, 2010.

[50] Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

[51] Olivier Roustant, David Ginsbourger, and Yves Deville. The DiceKriging package: kriging-based metamodeling and optimization for computer experiments. In *Book of abstract of the R User Conference*, 2009.

[52] Dan Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. In *Advances in Neural Information Processing Systems*, pages 1583–1591, 2014.

[53] Daniel Russo and Benjamin Van Roy. An information-theoretic analysis of Thompson sampling. *Journal of Machine Learning Research*, 17(68):1–30, 2016.

[54] Alexander Shapiro. Asymptotic analysis of stochastic programs. *Annals of Operations Research*, 30(1):169–186, 1991.

[55] David Siegmund. *Sequential analysis: tests and confidence intervals*. Springer Science & Business Media, 1985.

[56] Jarno Vanhatalo, Jaakko Riihimäki, Jouni Hartikainen, Pasi Jylänki, Ville Tolvanen, and Aki Vehtari. Gpstuff: Bayesian modeling with Gaussian processes. *Journal of Machine Learning Research*, 14(Apr):1175–1179, 2013.

[57] Rolf Waeber. *Probabilistic Bisection Search for Stochastic Root-Finding*. PhD thesis, Cornell University, 2013.

[58] Rolf Waeber, Peter I Frazier, and Shane G Henderson. A Bayesian approach to Stochastic Root Finding. In *Proceedings of the 2011 Winter Simulation Conference (WSC)*, pages 4033–4045. IEEE, 2011.

[59] Rolf Waeber, Peter I Frazier, and Shane G Henderson. Bisection search with noisy responses. *SIAM Journal on Control and Optimization*, 51(3):2261–2279, 2013.

[60] Christopher KI Williams and David Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

[61] Simon N Wood. mgcv: GAMs and generalized ridge regression for R. *R News*, 1(2):20–25, 2001.

[62] Ji Zhu and Trevor Hastie. Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14(1):185–205, 2005.