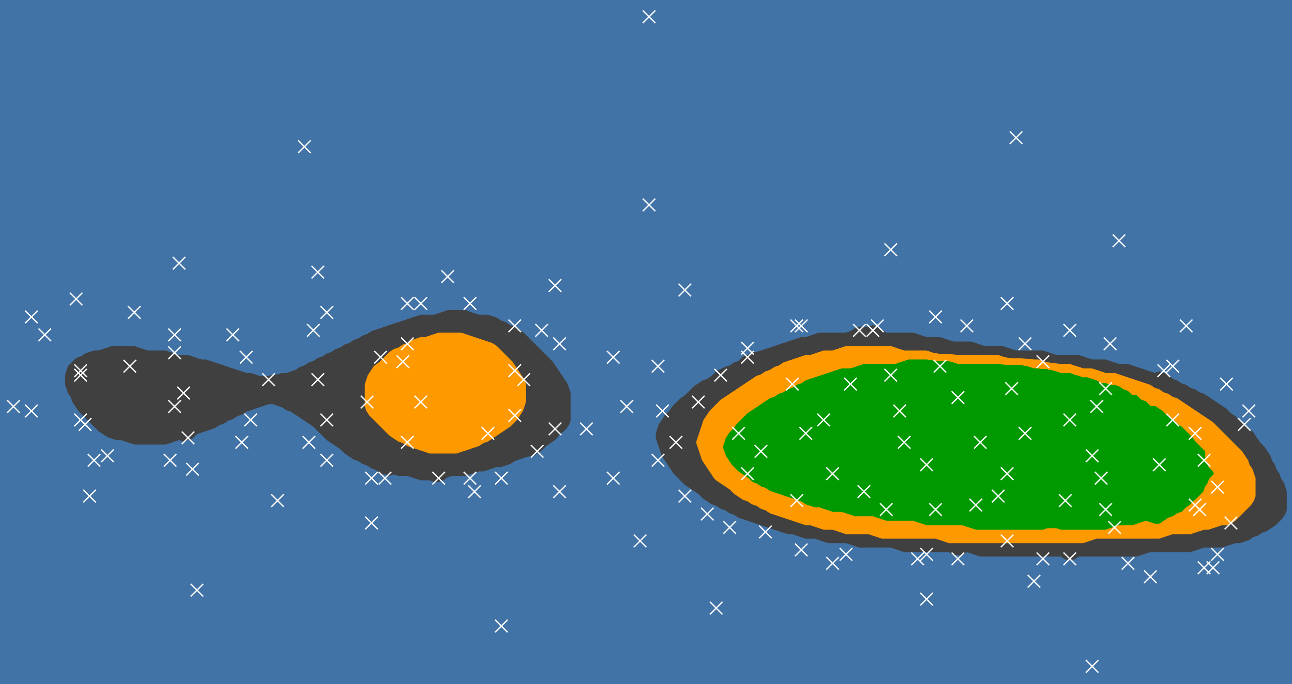
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

ETH Zurich
Department of Computer Science

M.Sc. Thesis by Alkis Gotovos
Supervisor: Andreas Krause
Assistant Professor, ETH Zurich

Active Learning for Level Set Estimation



Zurich, March 2013

Abstract

Many information gathering problems require determining the set of points, for which an unknown function takes value above or below some given threshold level. As a concrete example, in the context of environmental monitoring of Lake Zurich we would like to estimate the regions of the lake where the concentration of chlorophyll or algae is greater than some critical value, which would serve as an indicator of algal bloom phenomena. A critical factor in such applications is the high cost in terms of time, battery power, etc. that is associated with each measurement, therefore it is important to be careful about selecting “informative” locations to sample, in order to reduce the total sampling effort required.

We formalize the task of level set estimation as a classification problem with sequential measurements, where the unknown function is modeled as a sample from a Gaussian process (GP). We propose LSE, an active learning algorithm that guides both sampling and classification based on GP-derived confidence bounds, and provide theoretical guarantees about its sample complexity. Furthermore, we extend LSE and its theory to two more natural settings: (1) where the threshold level is implicitly defined as a percentage of the (unknown) maximum of the target function and (2) where samples are selected in batches. Based on the latter extension we also propose a simple path planning algorithm. We evaluate the effectiveness of our proposed methods on two problems of practical interest, namely the aforementioned autonomous monitoring of algal populations in Lake Zurich and geolocating network latency.

Acknowledgements

I consider myself very lucky to have had such an excellent supervisor as Prof. Andreas Krause, who has the gift of making complex things simple and intuitive.

I would also like to thank Nathalie Casati and Gregory Hitz for our collaboration on parts of this thesis.

Finally, thanks to the Limnological Station of University of Zurich and the Autonomous Systems Lab of ETH for providing the environmental monitoring datasets from Lake Zurich.

Contact

The best way to contact me is to search my name on the Web and retrieve my current e-mail address from my homepage. Of course, any questions or comments are welcome.

Contents

1 Introduction

Many information gathering problems require accurately determining the regions where the value of some unknown function lies above or below a given threshold level. Moreover, evaluating the function is usually a costly procedure and the measurements returned are noisy.

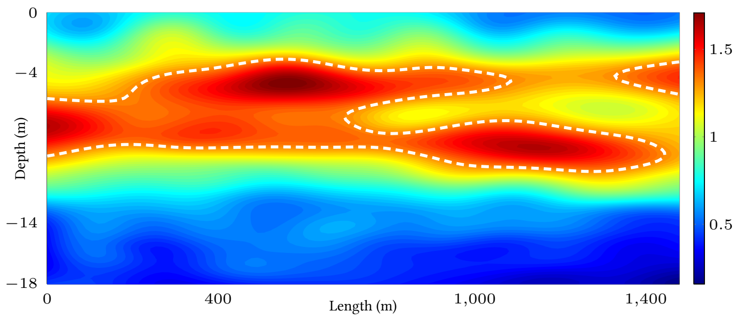
As a concrete example of such an application, consider the task of monitoring a lake environment (in our case Lake Zurich) for algal bloom, a phenomenon that is potentially harmful to other organisms of the ecosystem. One way to accomplish this is by determining the regions of the lake where the levels of algae-produced chlorophyll (see Figure 1.1a) or the concentration of the algae themselves (see Figure 1.2a) lie above some threshold value determined by field experts. These regions can be estimated by sampling at various locations of the lake using a mobile sensing device. However, each measurement is costly in terms of time and sensor battery power, therefore the sampling locations have to be picked carefully, in order to reduce the total number of measurements required.

Other example applications in the context of environmental monitoring (Rahimi et al., 2004) include estimating level sets of quantities such as solar radiation, humidity, etc., and determining the extent of hazardous phenomena, e.g. air pollution or oil spills (Galland et al., 2004). In a different category are applications that consist in determining the subset of a parameter space that represents "acceptable" hypotheses (Bryan et al., 2005) or designs (Ramakrishnan et al., 2005).

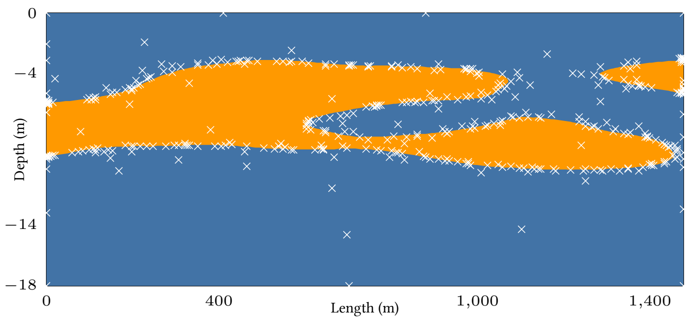
We pose the problem of estimating some function level set as a classification problem to be solved using *active learning*. In particular, we study the problem of classifying points into a super- and a sublevel set with respect to some given threshold level by making sequential decisions of the next sampling location at each time step given all previous measurements. For solving this problem, we propose the Level Set Estimation (LSE) algorithm, which utilizes Gaussian processes (Rasmussen & Williams, 2006) to model the target function and exploits inferred variance estimates to create confidence bounds and drive the selection process. We also provide an information-theoretic bound on the number of measurements needed to achieve a certain accuracy when the underlying function is sampled from a known Gaussian process.

Figure 1.1b shows an example of estimated super- and sublevel sets resulting from running LSE on the chlorophyll concentration dataset of Figure 1.1a for a specified threshold level. Intuitively, LSE works by progressively classifying points into the super- and sublevel sets and, at the same time, sampling at regions of high "ambiguity", i.e. regions where there is high uncertainty about whether the value of the function lies above or below the specified threshold level. As a consequence, the vast majority of the sampled locations are concentrated near the level set to be estimated.

We also extend the LSE algorithm to two more settings that naturally arise in practical applications. In the first setting, we do not a priori have a specific threshold level at our disposal, but would still like to perform level set estimation with respect



(a) Ground truth



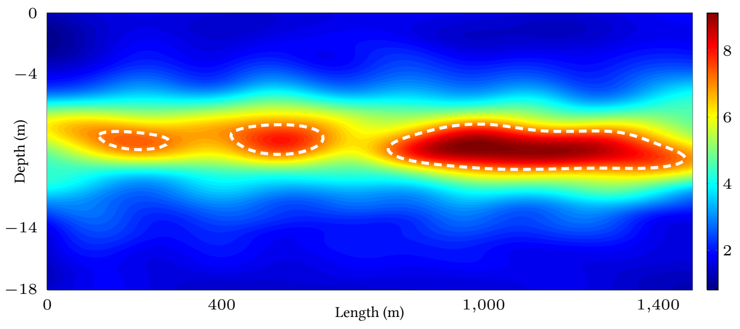
(b) Estimated level set

Figure 1.1: **(a)** Chlorophyll concentration in relative fluorescence units (RFU) inferred from 2024 measurements within a vertical transect plane of Lake Zurich (target level set at 1.3 RFU shown dashed). **(b)** Estimated classification into superlevel (orange) and sublevel (blue) sets after 405 samples (white crosses) using our proposed LSE algorithm.

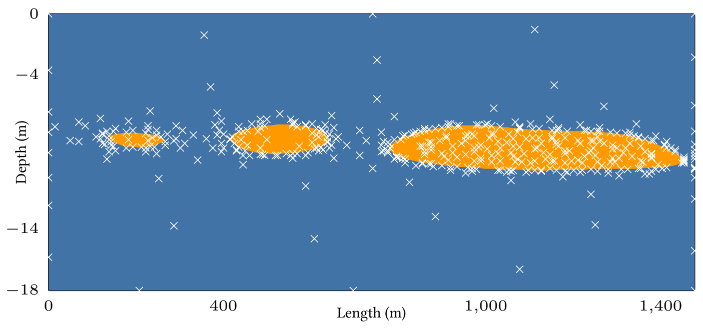
to an *implicit* level that is expressed as a percentage of the function maximum. In the environmental monitoring application outlined above, we might want to estimate regions of relatively high algae concentration, where what is considered as "high" concentration might depend on the season of the year or other environmental factors. Using an implicit level gives us a way to obtain useful information without the need to hardcode a prespecified threshold level of interest into the algorithm.

Figure 1.2b shows the estimated super- and sublevel sets resulting from running LSE on the algae concentration dataset of Figure 1.2a using an implicit threshold level of 75% of the maximum. The element of gradual classification of the explicit threshold algorithm is also present in the implicit version, but there is an additional complexity of estimating the function maximum in order to obtain accurate estimates of the implicit level. Therefore, while the algorithm still focuses on the "ambiguous" regions of the input space, at the same time it also heavily samples near the regions of the function maxima.

The second extension we propose applies to both the explicit and implicit versions of the algorithm and aims at selecting at each step a *batch* of next samples, rather than just one sample at a time. A reason for doing so is that, in problems such as the lake sensing example outlined above, apart from the cost of actually taking each measurement, we also have to take into account the cost incurred by traveling



(a) Ground truth



(b) Estimated level set

Figure 1.2: **(a)** Algae concentration in relative fluorescence units (RFU) inferred from 2024 measurements within a vertical transect plane of Lake Zurich (target level set at 75% of the maximum, i.e. $\approx$ 7 RFU, shown dashed). **(b)** Estimated classification into superlevel (orange) and sublevel (blue) sets after 440 samples (white crosses) using the implicit version of the LSE algorithm.

from one sampling location to the next. Traveling costs can be dramatically reduced, if we plan ahead by selecting multiple points at a time. Another reason is that some problems allow for running multiple function evaluations in parallel, in which case selecting batches of points can lead to a significant increase in sampling throughput.

Contributions. The main contributions of this thesis can be summarized as follows:

- We introduce the LSE algorithm for sequentially estimating level sets of unknown functions.
- We extend LSE to select samples in batches and present a straightforward application of batch sampling for path planning.
- We consider for the first time the problem of estimating level sets under implicitly defined threshold levels and propose an extension of LSE for this problem.
- We prove theoretical convergence bounds for LSE and its two extensions when the target function is sampled from a known GP.
- We evaluate LSE and its extensions on three real-world datasets and show that our proposed algorithms are competitive with the state-of-the-art.

2 Related work

As mentioned before, our work falls under the area of active learning. Traditional "passive" supervised learning is generally concerned with choosing a hypothesis (e.g. a classifier) from a set of available hypotheses, given a set of labeled instances. In many practical applications, obtaining instance labels is costly and, therefore, we would like to come up with an interactive procedure that at the same time as evaluating the candidate hypotheses also selects instances to be labeled. The hope is that focusing the labeling effort on "informative" instances will in the end significantly reduce the total sampling cost, i.e. the number of instances required to estimate an accurate solution of the problem at hand.

The particular setting that we consider for performing level set estimation is known as *pool-based active learning* (Settles, 2012). In this setting, we are given a fixed set (pool) of unlabeled instances and at each iteration the algorithm has to select the next instance to be labeled, while given access to the whole set (e.g. for evaluating some informativeness measure on each element).

For estimating level sets in this setting, Bryan et al. (2005) have proposed the *straddle* heuristic, which selects where to sample by trading off uncertainty and proximity to the desired threshold level. Similarly to our proposed algorithm they also use GPs to model the underlying function and obtain uncertainty estimates through the inferred variance. In addition, we show in Chapter 4 that our proposed ambiguity-based selection rule bears a close similarity to the straddle score. However, as we will also see, our algorithm's behavior is primarily dictated by its classification rules and, thus, does not experience the limited exploration from which straddle seems to suffer in some cases (see Chapter 6). Furthermore, no theoretical justification has been given for the use of straddle, neither for its extension to composite functions (Bryan & Schneider, 2008).

Garnett et al. (2012) consider the problem of *active search*, which is also about sequential sampling from a domain of two (or more) classes. In our case the classes would be the super- and sublevel sets with respect to the desired threshold level. In contrast to our goal of detecting the class boundaries (i.e. level set), however, their goal is to sample as many points as possible from one of the classes. The difference between our level set estimation setting and the above active search problem resembles the more fundamental difference between active learning and *online learning* (Shalev-Shwartz, 2012), where the former aims to obtain an accurate model after a number of sampling steps, while the latter aims to make accurate predictions during the sampling process.

In the context of mobile sensor networks, previous work on level set (Dantu & Sukhatme, 2007; Srinivasan et al., 2008) and boundary (Singh et al., 2006) estimation and tracking has primarily focused on controlling the movement and communication of sensor nodes, without giving much attention to individual sampling locations and the choice thereof.

A similar setting to ours in terms of sequential sampling, but different in terms of objectives, is that of *multi-armed bandit optimization.* The target there is to estimate the maximum of an underlying function by again choosing to label (evaluate) one of the bandit arms (sampling locations) at each time step. For this problem, GPs have been used both for modeling, as well as for sample selection (Brochu et al., 2010). In particular, the GP-UCB algorithm utilizes the GP-inferred mean and variance estimates to construct upper confidence bounds for each arm and uses them for driving the selection process. Using an information-theoretic approach, parts of which we utilize in the analysis of our proposed algorithms, GP-UCB has been shown to achieve sublinear regret (Srinivas et al., 2010). An extension of GP-UCB to the multi-objective optimization problem has been proposed by Zuluaga et al. (2013), who also use a similar to ours GP-based classification scheme to gradually classify points as being Pareto-optimal or not.

Existing approaches for performing multiple evaluations in parallel in the context of GP optimization, include *simulation matching* (Azimi et al., 2010), which combines GP modeling with Monte-Carlo simulations, and the GP-BUCB (Desautels et al., 2012) algorithm, which obtains similar regret bounds to GP-UCB, and from which we borrow the main idea for performing batch sample selection.

To our knowledge, there has been no previous work on actively estimating level sets with respect to implicitly defined threshold levels.

3 Preliminaries

Formal problem statement. Given a function $f : D \to \mathbb{R}$, where D is a finite[1] subset of $\mathbb{R}^d$, and a threshold level $h \in \mathbb{R}$, we define the *level set estimation problem* as the problem of classifying every point $\boldsymbol{x} \in D$ into a *superlevel set* $H = \{\boldsymbol{x} \in D \mid f(\boldsymbol{x}) > h\}$ and a *sublevel set* $L = \{\boldsymbol{x} \in D \mid f(\boldsymbol{x}) \leq h\}$.

When an explicit level is not available, we can define an *implicit threshold level* with respect to the function maximum in either absolute or relative terms. We use the relative definition in our exposition with $h = \omega \max_{\boldsymbol{x} \in D} f(\boldsymbol{x})$ and $\omega \in (0, 1)$.

In the *strictly sequential* setting, at each step $t \geq 1$ we select a point $\boldsymbol{x}_t \in D$ to be evaluated and obtain a noisy measurement $y_t = f(\boldsymbol{x}_t) + n_t$. In the *batch* setting we select B points at a time and only obtain the resulting measurements after all of the B points have been selected. This setting can be generalized by using the notion of a *feedback function* introduced by Desautels et al. (2012). In particular, we assume that there is a function $\text{fb} : \mathbb{N} \to \mathbb{N} \cup \{0\}$, such that $\text{fb}[t] \leq t - 1$ for all $t \geq 1$, and when selecting the next point at time step t, we have access to evaluated measurements up to time step $\text{fb}[t]$. For selecting batches of size B we can define $\text{fb}[1] = \ldots = \text{fb}[B] = 0$, $\text{fb}[B+1] = \ldots = \text{fb}[2B] = B$, and so on, but the feedback function also allows for defining more complex cases of delayed feedback.

Gaussian processes. Without any assumptions about the function f, attempting to estimate level sets from few samples is a hopeless endeavor. Modeling f as a sample from a Gaussian process (GP) provides an elegant way for specifying properties of the function in a nonparametric fashion. A GP is defined as a collection of random variables, any finite subset of which is distributed according to a multivariate Gaussian in a consistent way (Rasmussen & Williams, 2006). A GP is denoted as $\mathcal{GP}(\mu(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$ and is completely specified by its mean function $\mu(\boldsymbol{x})$, which can be assumed to be zero w.l.o.g., and its covariance function or kernel $k(\boldsymbol{x}, \boldsymbol{x}')$, which encodes smoothness properties of functions sampled from the GP.

Assuming a GP prior $\mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$ over f and given t noisy measurements $\boldsymbol{y}_{1:t} = [y_1, \ldots, y_t]^T$ for points in $A_t = \{x_1, \ldots, x_t\}$, where $y_i = f(\boldsymbol{x}_i) + n_i$ and $n_i \sim \mathcal{N}(0, \sigma^2)$ (Gaussian i.i.d. noise) for $i = 1, \ldots, t$, the posterior over f is also a GP and its mean, covariance, and variance functions are given by the following analytic formulae:

$$\mu_t(\boldsymbol{x}) = \boldsymbol{k}_t(\boldsymbol{x})^T \left(\boldsymbol{K}_t + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{y}_t \tag{3.1}$$

$$k_t(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}_t(\boldsymbol{x})^T \left(\boldsymbol{K}_t + \sigma^2 \boldsymbol{I}\right)^{-1} \boldsymbol{k}_t(\boldsymbol{x})$$

$$\sigma_t^2(\boldsymbol{x}) = k_t(\boldsymbol{x}, \boldsymbol{x}), \tag{3.2}$$

where $\boldsymbol{k}_t(\boldsymbol{x}) = [k(\boldsymbol{x}_1, \boldsymbol{x}), \ldots, k(\boldsymbol{x}_t, \boldsymbol{x})]^T$ and $\boldsymbol{K}_t$ is the kernel matrix of already

[1]The subsequent analysis only considers the finite domain case for simplicity, however our results can be generalized to continuous spaces as well (cf. Srinivas et al., 2010).

observed points, defined as $\boldsymbol{K}_t = [k(\boldsymbol{x}, \boldsymbol{x}')]_{\boldsymbol{x},\boldsymbol{x}' \in A_t}$.

Information gain. The information gain about a random variable X from observing a random variable Y, also known as the *mutual information* of the two variables, is defined as (Cover & Thomas, 2006)

$$I(X; Y) = H(X) - H(X \mid Y),$$

where $H(X)$ is the entropy of X and $H(X \mid Y)$ is the conditional entropy of X given Y. The information gain is symmetric, i.e. $I(X; Y) = I(Y; X)$, and, as can be seen from the above definition, quantifies the (average) reduction in our uncertainty about X when observing the value of Y (and vice versa).

We will use the information gain to quantify the reduction in uncertainty about $f \sim \mathcal{GP}(0, k(\boldsymbol{x}, \boldsymbol{x}'))$ when obtaining a set of noisy observations $\boldsymbol{y}_{1:t}$, which is

$$I(f; \boldsymbol{y}_{1:t}) = I(\boldsymbol{y}_{1:t}; f) = H(\boldsymbol{y}_{1:t}) - H(\boldsymbol{y}_{1:t} \mid f).$$

If we define $f_i = f(\boldsymbol{x}_i)$, from the fact that $\boldsymbol{y}_{1:t} = \boldsymbol{f}_{1:t} + \boldsymbol{n}_{1:t}$ is a sum of Gaussians it follows that $\boldsymbol{y}_{1:t} \sim \mathcal{N}\left(0, \boldsymbol{K}_t + \sigma^2 I\right)$, and the entropy of the noisy observations is

$$H(\boldsymbol{y}_{1:t}) = \frac{1}{2} \log\left((2\pi e)^t |\boldsymbol{K}_t + \sigma^2 I|\right). \tag{3.3}$$

Furthermore, note that the observations $\boldsymbol{y}_{1:t}$ are conditionally independent of anything else given the underlying GP values of $\boldsymbol{f}_{1:t}$ and distributed as $\boldsymbol{y}_{1:t} \mid \boldsymbol{f}_{1:t} \sim \mathcal{N}(\boldsymbol{f}_{1:t}, \sigma^2 I)$. It follows that

$$H(\boldsymbol{y}_{1:t} \mid f) = H(\boldsymbol{y}_{1:t} \mid \boldsymbol{f}_{1:t}) = \frac{1}{2} \log\left((2\pi e)^t |\sigma^2 I|\right). \tag{3.4}$$

Combining (3.3) and (3.4) we get

$$I(f; \boldsymbol{y}_{1:t}) = \frac{1}{2} \log\left(|I + \sigma^{-2} \boldsymbol{K}_t|\right)$$

The notion of information gain can be generalized to the *conditional information gain* or *conditional mutual information*, which quantifies the reduction in uncertainty about a variable X from observing the value of a variable Y when already given the value of variable Z. It is similarly defined as

$$I(X; Y \mid Z) = H(X \mid Z) - H(X \mid Y, Z).$$

We will use it to quantify the reduction in uncertainty about f when obtaining a set of additional noisy observations $\boldsymbol{y}_A$ given that we already have available observations $\boldsymbol{y}_{1:t}$, which is

$$\begin{aligned} I(f; \boldsymbol{y}_A \mid \boldsymbol{y}_{1:t}) &= H(\boldsymbol{y}_A \mid \boldsymbol{y}_{1:t}) - H(\boldsymbol{y}_A \mid \boldsymbol{y}_{1:t}, f) \\ &= H(\boldsymbol{y}_A \mid \boldsymbol{y}_{1:t}) - H(\boldsymbol{y}_A \mid f) \\ &= \ldots \\ &= \frac{1}{2} \log\left(|I + \sigma^{-2} \boldsymbol{K}_A^t|\right), \end{aligned}$$

using a similar derivation to above and defining $\boldsymbol{K}_A^t = [k_t(\boldsymbol{x}, \boldsymbol{x}')]_{\boldsymbol{x},\boldsymbol{x}' \in \boldsymbol{A}}$.

4 The LSE Algorithm

We now present our proposed Level Set Estimation (LSE) algorithm for the strictly sequential setting with explicit thresholds. LSE is similar in spirit to the GP-UCB (Srinivas et al., 2010) bandit optimization algorithm in that it uses a GP to model the underlying function and facilitates the inferred mean and variance of the GP to guide the selection of points to be evaluated.

More concretely, the inferred mean and variance of (3.1) and (3.2) can be used to construct a *confidence interval*

$$Q_t(\boldsymbol{x}) = \left[\mu_{t-1}(\boldsymbol{x}) \pm \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})\right]$$

for any point $\boldsymbol{x} \in D$, which captures our uncertainty about $f(\boldsymbol{x})$ after having already obtained noisy evaluations of f at points $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_t\}$. The parameter β_t acts as a scaling factor and its choice is discussed later. The above-defined confidence intervals serve two purposes in our algorithm: first, they allow us to judge whether a point can be classified into the super- or sublevel sets or whether the decision should be deferred until more information is available; second, they guide the sampling process towards points that are likely to be informative with respect to the desired level set.

The pseudocode of Algorithm 1 depicts in detail the operation of LSE. Our algorithm maintains a set of yet unclassified points U_t, as well as a superlevel set H_t and a sublevel set L_t, which are updated at each iteration. Furthermore, the algorithm maintains for each unclassified $\boldsymbol{x}$ a monotonically decreasing *confidence region* $C_t(\boldsymbol{x})$, which results from intersecting successive confidence intervals, i.e.

$$C_t(\boldsymbol{x}) = \bigcap_{i=1}^{t} Q_i(\boldsymbol{x}) = C_{t-1}(\boldsymbol{x}) \cap Q_t(\boldsymbol{x}).$$

Initially, all points $\boldsymbol{x} \in D$ are unclassified and the confidence regions have infinite range (lines 1–2). At each iteration, the confidence regions of all unclassified points are updated (line 7) and each of these points is either classified into one of H_t or L_t, or is left unclassified (lines 8–14). Then, the next point is selected and evaluated (lines 17–29) and the new GP posterior is computed (line 30) taking into account the newest measurement, according to equations (3.1) and (3.2). The algorithm terminates when all points in D have been classified, at which point the estimated super- and sublevel sets $\hat{H}$ and $\hat{L}$ are returned (lines 21–22).

We now discuss in more detail the issues of how points are classified and how the next point to be evaluated is selected at each step.

4.1 Algorithm details

Classification. The classification of a point $\boldsymbol{x}$ into H_t or L_t depends on the position of its confidence region with respect to the threshold level h. Intuitively, if all of

Algorithm 1 The LSE algorithm

```
Input: sample set $D$, GP prior ($\mu_0 = 0$, $k$, $\sigma_0$),
       threshold value $h$, accuracy parameter $\epsilon$
Output: predicted sets $\hat{H}$, $\hat{L}$
 1: $H_0 \leftarrow \varnothing,\ L_0 \leftarrow \varnothing,\ U_0 \leftarrow D$
 2: $C_0(\boldsymbol{x}) \leftarrow \mathbb{R}$, for all $\boldsymbol{x} \in D$
 3: $t \leftarrow 1$
 4: while $U_{t-1} \neq \varnothing$ do
 5:   $H_t \leftarrow H_{t-1},\ L_t \leftarrow L_{t-1},\ U_t \leftarrow U_{t-1}$
 6:   for all $\boldsymbol{x} \in U_{t-1}$ do
 7:     $C_t(\boldsymbol{x}) \leftarrow C_{t-1}(\boldsymbol{x}) \cap Q_t(\boldsymbol{x})$
 8:     if $\min(C_t(\boldsymbol{x})) + \epsilon > h$ then
 9:       $U_t \leftarrow U_t \setminus \{\boldsymbol{x}\}$
10:       $H_t \leftarrow H_t \cup \{\boldsymbol{x}\}$
11:     else if $\max(C_t(\boldsymbol{x})) - \epsilon \leq h$ then
12:       $U_t \leftarrow U_t \setminus \{\boldsymbol{x}\}$
13:       $L_t \leftarrow L_t \cup \{\boldsymbol{x}\}$
14:     end if
15:   end for
16:   $\boldsymbol{x}_t \leftarrow \operatorname{argmax}_{\boldsymbol{x} \in U_t}(a_t(\boldsymbol{x}))$
17:   $y_t \leftarrow f(\boldsymbol{x}_t) + n_t$
18:   Compute $\mu_t(\boldsymbol{x})$ and $\sigma_t(\boldsymbol{x})$, for all $\boldsymbol{x} \in U_t$
19:   $t \leftarrow t + 1$
20: end while
21: $\hat{H} \leftarrow H_{t-1}$
22: $\hat{L} \leftarrow L_{t-1}$
```

$C_t(\boldsymbol{x})$ lies above h, then with high probability $f(\boldsymbol{x}) > h$ and $\boldsymbol{x}$ should be moved into H_t. Similarly, if $C_t(\boldsymbol{x})$ lies below h, then $\boldsymbol{x}$ should be moved into L_t. Otherwise, we are still uncertain about the class of $\boldsymbol{x}$, therefore it should, for the moment, remain unclassified. As can be seen in the classification rules of lines 8 and 11, we relax these conditions by introducing an accuracy parameter ϵ, which trades off classification accuracy for sampling cost. The resulting classification scheme is illustrated by the example of Figure 4.1a, in which point $\boldsymbol{x}$ would be classified into H_t and point $\boldsymbol{x}''$ into L_t, while point $\boldsymbol{x}'$ would remain in U_t as unclassified. Note that LSE uses a monotonic classification scheme, meaning that once a point has been classified, it stays so until the algorithm terminates.

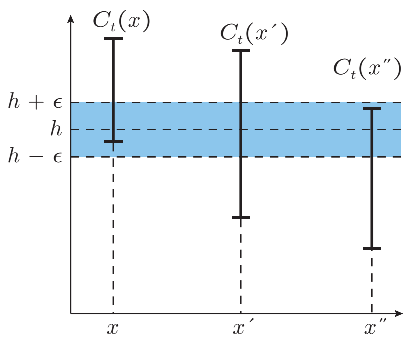
Sample selection. For selecting the next point to be evaluated at each iteration, we define the following quantity

$$a_t(\boldsymbol{x}) = \min\{\max(C_t(\boldsymbol{x})) - h, h - \min(C_t(\boldsymbol{x}))\},$$

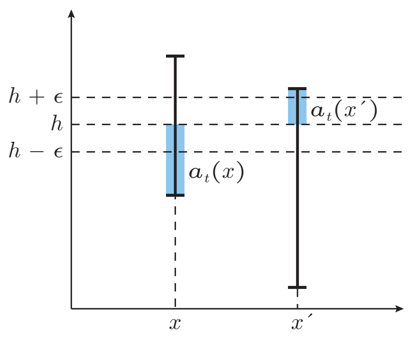
which we call classification *ambiguity*. As its name suggests, the ambiguity of a point $\boldsymbol{x}$ quantifies our uncertainty about whether $\boldsymbol{x}$ belongs to H_t or L_t. The intuition of sampling at areas of the sample space with large classification uncertainty, expecting to gain more information about the problem at hand when sampling at those areas, manifests itself in LSE by choosing to evaluate at each iteration the point with the largest ambiguity amongst the yet unclassified.

Note that, if we define the confidence region *width* of a point $\boldsymbol{x}$ as

$$w_t(\boldsymbol{x}) = \max(C_t(\boldsymbol{x})) - \min(C_t(\boldsymbol{x})),$$



(a) Confidence regions



(b) Ambiguities

Figure 4.1: **(a)** Example of the three possible configurations of confidence regions: $\boldsymbol{x}$ will be classified into H_t, $\boldsymbol{x}''$ into L_t, whereas $\boldsymbol{x}'$ will remain unclassified. **(b)** Ambiguities of two example points: $\boldsymbol{x}$ is a point of high ambiguity (close to $w_t(\boldsymbol{x})/2$), while $\boldsymbol{x}'$ has low ambiguity (close to ϵ).

it follows from the the classification rules of LSE that

$$\epsilon \leq a_t(\boldsymbol{x}) \leq w_t(\boldsymbol{x})/2, \text{ for every } \boldsymbol{x} \in U_t.$$

Ambiguity values close to $w_t(\boldsymbol{x})/2$ indicate that the confidence region of $\boldsymbol{x}$ extends almost equally above and below the threshold level h, while values close to ϵ indicate that $\boldsymbol{x}$ can almost unambiguously be classified into H_t or L_t (see Figure 4.1b).

We can make an interesting observation at this point. If we use the confidence intervals $Q_t(\boldsymbol{x})$ instead of the confidence regions $C_t(\boldsymbol{x})$ in the definition of ambiguity, we get the following quantity

$$\begin{aligned}
a'_t(\boldsymbol{x}) &= \min\{\max(Q_t(\boldsymbol{x})) - h, h - \min(Q_t(\boldsymbol{x}))\} \\
&= \min\{\mu_{t-1}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}) - h,\ h - \mu_{t-1}(\boldsymbol{x}) + \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})\} \\
&= \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}) - |\mu_{t-1}(\boldsymbol{x}) - h|.
\end{aligned}$$

For $\beta_t^{1/2} = 1.96$, this is identical to the *straddle* (Bryan et al., 2005) heuristic, which can thus be intuitively explained in terms of classification ambiguity.

Figures 4.2a to 4.2d present an example of running the LSE algorithm on a fine grid of points sampled from the inferred GP of the chlorophyll dataset shown in Figure 1.1a. Note how the sampling process focuses on the ambiguous regions around the desired level set until all points in D have been successfully classified.

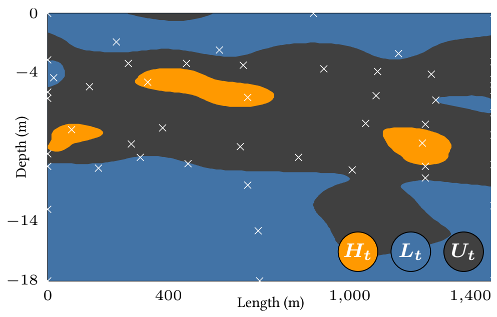
4.2 Theoretical analysis

The convergence analysis of LSE rests on quantifying the complexity of the GP prior for f in information-theoretic terms. The information gain (cf. Chapter 3) about f from observing t noisy measurements $\boldsymbol{y}_t = (y_i)_{1\leq i\leq t}$ is

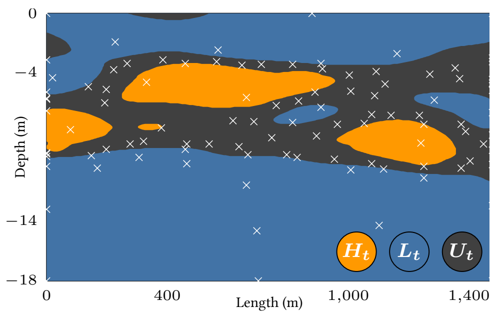
$$I(\boldsymbol{y}_t; f) = H(\boldsymbol{y}_t) - H(\boldsymbol{y}_t \mid f).$$

Srinivas et al. (2010) used the maximum information gain over all possible sets of t observations

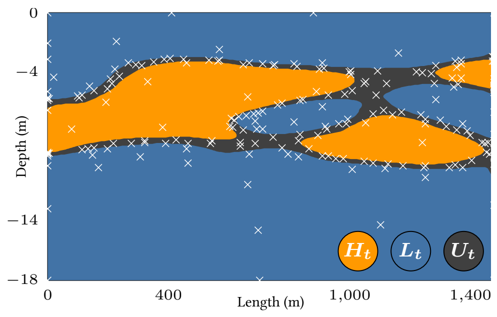
$$\gamma_t = \max_{\boldsymbol{y}_t} I(\boldsymbol{y}_t; f)$$



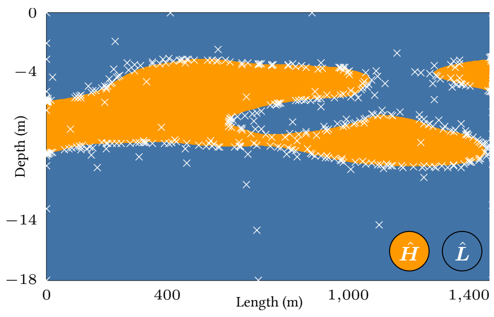
(a) $t = 50$



(b) $t = 100$



(c) $t = 200$



(d) $t = 405$ (termination)

Figure 4.2: Running LSE with $\epsilon = 0.1$ on a regular grid of 100×100 points sampled from the inferred chlorophyll concentration GP of Figure 1.1a. Regions of already classified points are shown in orange (H_t) and blue (L_t), regions of yet unclassified points (U_t) in black, and observed points ($\{\boldsymbol{x}_i\}_{1 \le i \le t}$) as white marks.

for bounding the regret of the GP-UCB algorithm. We use the same quantity to bound the number of LSE iterations required to achieve a certain classification quality.

To quantify the quality of a solution $(\hat{H}, \hat{L})$ with respect to a single point $\boldsymbol{x} \in D$ we use the misclassification loss

$$\ell_h(\boldsymbol{x}) = \begin{cases} \max\{0, f(\boldsymbol{x}) - h\}\,, & \text{if } \boldsymbol{x} \in \hat{L} \\ \max\{0, h - f(\boldsymbol{x})\}\,, & \text{if } \boldsymbol{x} \in \hat{H} \end{cases}.$$

The overall quality of a solution can then be judged by the largest misclassification loss among all points in the sample space, i.e. $\max_{\boldsymbol{x} \in D} \ell_h(\boldsymbol{x})$. Intuitively, having a solution with $\max_{\boldsymbol{x} \in D} \ell_h(\boldsymbol{x}) \le \epsilon$ means that every point $\boldsymbol{x}$ is correctly classified with respect to a threshold level that deviates by at most ϵ from the true level h; we call such a solution ϵ-*accurate*. The following theorem establishes a convergence bound for LSE in terms of γ_t for any given accuracy ϵ.

Theorem 1. *For any $h \in \mathbb{R}$, $\delta \in (0, 1)$, and $\epsilon > 0$, if $\beta_t = 2 \log(|D|\pi^2 t^2/(6\delta))$, LSE terminates after at most T iterations, where T is the smallest positive integer satisfying*

$$\frac{T}{\beta_T \gamma_T} \ge \frac{C_1}{4\epsilon^2},$$

where $C_1 = 8/\log(1 + \sigma^{-2})$.

Furthermore, with probability at least $1-\delta$*, the algorithm returns an* ϵ*-accurate solution, that is*

$$\Pr\left\{\max_{\boldsymbol{x}\in D} \ell_h(\boldsymbol{x}) \le \epsilon\right\} \ge 1-\delta.$$

The detailed proof of Theorem 1 can be found in Appendix A.1. We outline here the main steps required (cf. Figure 4.3).

Decreasing ambiguities. We show that the ambiguities of the selected points, $a_t(\boldsymbol{x}_t)$, are decreasing in t due to the maximum ambiguity selection rule and the monotonic classification scheme. Furthermore, by employing the maximum information gain γ_t, we show that $a_t(\boldsymbol{x}_t)$ decreases as $\mathcal{O}((\frac{\beta_t\gamma_t}{t})^{\frac{1}{2}})$.

Termination. We show that the classification rules guarantee that the algorithm terminates when $a_t(\boldsymbol{x}_t)$ is sufficiently small. The fact that this will eventually happen is implied by the previous step.

"Valid" confidence regions. We guarantee that $f(\boldsymbol{x})$ is included with high probability in the inferred interval $Q_t(\boldsymbol{x})$ and that this holds 1) for every $\boldsymbol{x}$ and 2) for every $t \ge 1$, which also implies that $f(\boldsymbol{x})$ is included in each confidence region $C_t(\boldsymbol{x})$. By suitably choosing the scaling parameter β_t, 1) follows from the fact that, for every $\boldsymbol{x} \in D$ and every $t \ge 1$, $f(\boldsymbol{x})$ is distributed according to $N(\mu_{t-1}(\boldsymbol{x}), \sigma_{t-1}(\boldsymbol{x}))$, since we are assuming that it is sampled from a known GP that is also used by LSE, and 2) follows from a union bound over t.

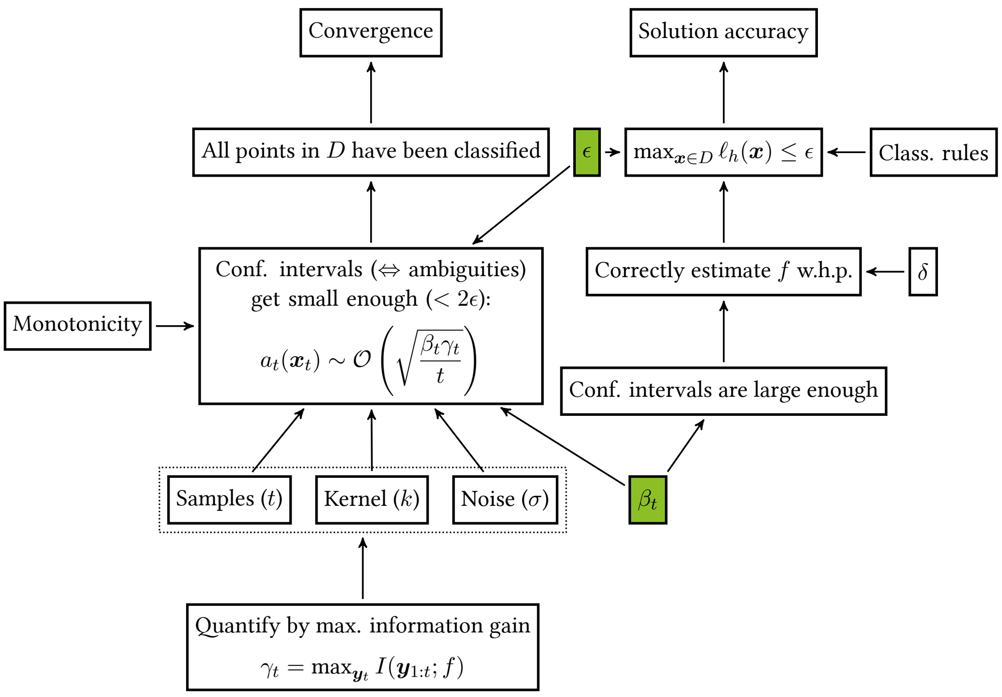

Convergence
Solution accuracy
All points in D have been classified
ϵ
$\max_{\boldsymbol{x}\in D} \ell_h(\boldsymbol{x}) \le \epsilon$
Class. rules
Conf. intervals (⇔ ambiguities) get small enough ($< 2\epsilon$): $a_t(\boldsymbol{x}_t) \sim \mathcal{O}\left(\sqrt{\frac{\beta_t\gamma_t}{t}}\right)$
Correctly estimate f w.h.p.
δ
Monotonicity
Conf. intervals are large enough
Samples (t)
Kernel (k)
Noise (σ)
β_t
Quantify by max. information gain $\gamma_t = \max_{\boldsymbol{y}_t} I(\boldsymbol{y}_{1:t}; f)$

Figure 4.3: A high-level outline of the theoretical analysis of LSE. Note the two tuning parameters in green that connect convergence and accuracy: β_t is fixed given the choice of δ, while ϵ is free and trades off sampling cost for accuracy.

Solution accuracy. We show that an ϵ-accurate solution is obtained upon termination, due to the classification rules and the “validity” of the confidence regions guaranteed in the previous step.

Note that bounds on γ_T have been established for commonly used kernels (Srinivas et al., 2010) and can be plugged into Theorem 1 to obtain concrete bounds on T. For example, for a d-dimensional sample space and a squared exponential GP kernel, $\gamma_T = \mathcal{O}((\log T)^{d+1})$. Consequently, the expression in the bound of Theorem 1 becomes $T/(\log T)^{d+2} \geq C/\epsilon^2$, where, for any given sample space and kernel hyperparameters, C depends only on the choice of δ.

5 Extensions

We now extend LSE to deal with the two problem variants introduced in Chapter 3, namely implicitly defined threshold levels and batch point selection. We highlight the key differences in the extended versions of the algorithm and the resulting implications about the convergence bound of Theorem 1.

5.1 Implicit threshold level

The substitution of the explicit threshold level by an implicit level $h = \omega \max_{\boldsymbol{x} \in D} f(\boldsymbol{x})$ requires modifying the classification rules as well as the selection rule of LSE, which results in what we call the LSE$_{\text{imp}}$ algorithm.

Since h is now an estimated quantity that depends on the function maximum, we have to take into account the uncertainty associated with it when making classification decisions. Concretely, we can obtain an *optimistic estimate* of the function maximum as $f_t^{opt} = \max_{\boldsymbol{x} \in U_t} \max(C_t(\boldsymbol{x}))$ and, analogously, a *pessimistic estimate* as $f_t^{pes} = \max_{\boldsymbol{x} \in U_t} \min(C_t(\boldsymbol{x}))$. The corresponding estimates of the implicit level are defined as $h_t^{opt} = \omega f_t^{opt}$ and $h_t^{pes} = \omega f_t^{pes}$, and can be used in a similar classification scheme to that of LSE. However, for the above estimates to be correct, we have to ensure that U_t always contains all points that could be maximizers of f, i.e. all points that satisfy $\max(C_t(\boldsymbol{x})) \geq f_t^{pes}$. For that purpose, points that should be classified, but are still possible function maximizers according to the above inequality, are kept in two sets M_t^H and M_t^L respectively, while a new set $Z_t = U_t \cup M_t^H \cup M_t^L$ is used in place of U_t to obtain the optimistic and pessimistic estimates h_t^{opt} and h_t^{pes}. The resulting classification rules are shown in lines 11–25 of Algorithm 2, where the conditions are, again, relaxed by an accuracy parameter ϵ.

In contrast to LSE, which solely focuses on sampling the most ambiguous points, in LSE$_{\text{imp}}$ it is also of importance to have a more exploratory sampling policy in order to obtain more accurate estimates f_t^{opt} and f_t^{pes}. To this end, we select at each iteration the point with the largest confidence region *width*, defined as

$$w_t(\boldsymbol{x}) = \max(C_t(\boldsymbol{x})) - \min(C_t(\boldsymbol{x})).$$

Note that, if confidence intervals were not intersected, this would be equivalent to maximum variance sampling (within Z_t).

Figures 5.1a to 5.1d present an example of running the LSE$_{\text{imp}}$ algorithm on a fine grid of points sampled from the inferred GP of the algae concentration dataset shown in Figure 1.2a. Note how, in contrast to the explicit threshold case, the sampling process now also focuses on the regions of near-maximal function value in addition to the ambiguous regions around the implied level set.

Theoretical analysis. The idea of using the maximum information gain can again be used to provide a convergence bound for the LSE$_{\text{imp}}$ algorithm. Some modifications to the analysis are required due to the difference in the classification rules, i.e. the

Algorithm 2 The LSE$_{\text{imp}}$ extension

```
Input: sample set $D$, GP prior ($\mu_0 = 0$, $k$, $\sigma_0$),
       threshold ratio $\omega$, accuracy parameter $\epsilon$
Output: predicted sets $\hat{H}$, $\hat{L}$
 1: $H_0 \leftarrow \varnothing,\ L_0 \leftarrow \varnothing,\ U_0 \leftarrow D,\ Z_0 \leftarrow D$
 2: $C_0(\boldsymbol{x}) \leftarrow \mathbb{R}$, for all $\boldsymbol{x} \in D$
 3: $t \leftarrow 1$
 4: while $U_{t-1} \neq \varnothing$ do
 5:     $H_t \leftarrow H_{t-1},\ L_t \leftarrow L_{t-1},\ U_t \leftarrow U_{t-1},\ Z_t \leftarrow Z_{t-1}$
 6:     for all $\boldsymbol{x} \in U_{t-1}$ do
 7:         $C_t(\boldsymbol{x}) \leftarrow C_{t-1}(\boldsymbol{x}) \cap Q_t(\boldsymbol{x})$
 8:         $h_t^{opt} \leftarrow \omega \max_{\boldsymbol{x} \in Z_{t-1}} \max(C_t(\boldsymbol{x}))$
 9:         $f_t^{pes} \leftarrow \max_{\boldsymbol{x} \in Z_{t-1}} \min(C_t(\boldsymbol{x}))$
10:         $h_t^{pes} \leftarrow \omega f_{pes}$
11:         if $\min(C_t(\boldsymbol{x})) + \epsilon \geq h_t^{opt}$ then
12:             $U_t \leftarrow U_t \setminus \{\boldsymbol{x}\}$
13:             if $\max(C_t(\boldsymbol{x})) < f_t^{pes}$ then
14:                 $H_t \leftarrow H_t \cup \{\boldsymbol{x}\}$
15:             else
16:                 $M_t^H \leftarrow M_t^H \cup \{\boldsymbol{x}\}$
17:             end if
18:         else if $\max(C_t(\boldsymbol{x})) - \epsilon \leq h_t^{pes}$ then
19:             $U_t \leftarrow U_t \setminus \{\boldsymbol{x}\}$
20:             if $\max(C_t(\boldsymbol{x})) < f_t^{pes}$ then
21:                 $L_t \leftarrow L_t \cup \{\boldsymbol{x}\}$
22:             else
23:                 $M_t^L \leftarrow M_t^L \cup \{\boldsymbol{x}\}$
24:             end if
25:         end if
26:     end for
27:     $Z_t \leftarrow U_t \cup M_t^H \cup M_t^L$
28:     $\boldsymbol{x}_t \leftarrow \operatorname{argmax}_{\boldsymbol{x} \in Z_t}(w_t(\boldsymbol{x}))$
29:     $y_t \leftarrow f(\boldsymbol{x}_t) + n_t$
30:     Compute $\mu_t(\boldsymbol{x})$ and $\sigma_t(\boldsymbol{x})$, for all $\boldsymbol{x} \in U_t$
31:     $t \leftarrow t + 1$
32: end while
33: $\hat{H} \leftarrow H_{t-1} \cup M_{t-1}^H$
34: $\hat{L} \leftarrow L_{t-1} \cup M_{t-1}^L$
```

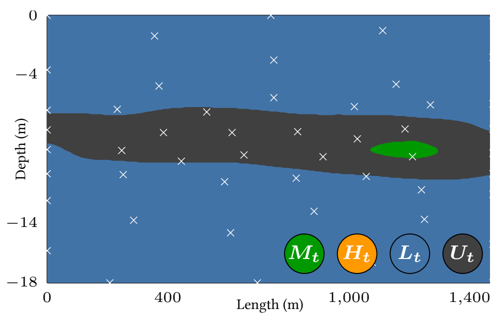
use of estimates h_t^{opt} and h_t^{pes} instead of h and the use of the extended set Z_t instead of U_t. The following theorem expresses for the LSE$_{\text{imp}}$ algorithm a similar in form bound to that of Theorem 1.

Theorem 2. *For any $\omega \in (0,1)$, $\delta \in (0,1)$, and $\epsilon > 0$, if $\beta_t = 2\log(|D|\pi^2 t^2/(6\delta))$, LSE$_{imp}$ terminates after at most T iterations, where T is the smallest positive integer satisfying*

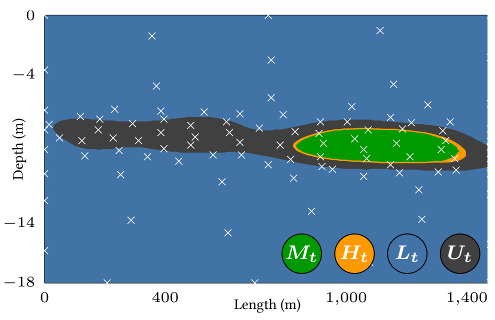
$$\frac{T}{\beta_T \gamma_T} \geq \frac{C_1(1+\omega)^2}{4\epsilon^2},$$

where $C_1 = 8/\log(1+\sigma^{-2})$.

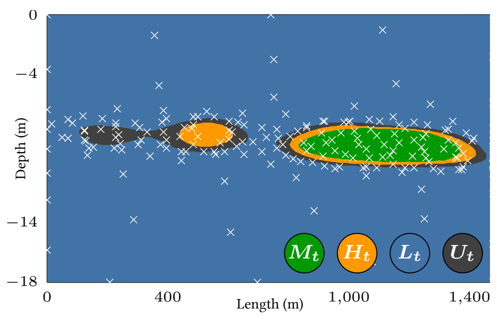
Furthermore, with probability at least $1-\delta$, the algorithm returns an ϵ-accurate



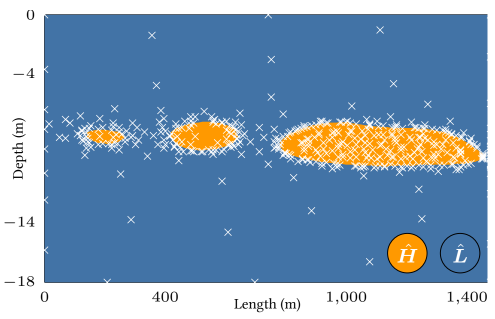
(a) $t = 50$



(b) $t = 100$



(c) $t = 200$



(d) $t = 440$ (termination)

Figure 5.1: Running LSE$_{\text{imp}}$ with $\epsilon = 0.7$ on a regular grid of 100×100 points sampled from the inferred algae concentration GP of Figure 1.2a. Regions of already classified points are shown in orange (H_t) and blue (L_t), regions of yet unclassified points (U_t) in black, regions of points that could be maximizers ($M_t = M_t^L \cup M_t^H$) in green, and observed points ($\{\boldsymbol{x}_i\}_{1 \le i \le t}$) as white marks.

solution with respect to the implicit level $h = \omega \max_{\boldsymbol{x} \in D} f(\boldsymbol{x})$, *that is*

$$\Pr\left\{\max_{\boldsymbol{x} \in D} \ell_h(\boldsymbol{x}) \le \epsilon\right\} \ge 1 - \delta.$$

Note that the sample complexity bound of Theorem 2 is a factor $(1+\omega)^2 \le 4$ larger than that of Theorem 1, and that $\omega = 0$ actually reduces to an explicit threshold of 0.

The detailed proof of Theorem 2 can be found in Appendix A.2. Here we outline the main similarities and differences compared to the proof of Theorem 1. As in the explicit threshold case, using the maximum information gain γ_t, we show that $w_t(\boldsymbol{x}_t)$ decreases as $\mathcal{O}((\frac{\beta_t \gamma_t}{t})^{\frac{1}{2}})$[1]. Furthermore, as before, the "validity" of the confidence regions is guaranteed by choosing β_t appropriately. However, the steps of proving termination and ϵ-accuracy need to be modified as follows.

Termination. Due to LSE$_{\text{imp}}$'s classification rules, in order to prove that the algorithm terminates (Lemma 12), we need to show that the gap between the optimistic h_t^{opt} and pessimistic h_t^{pes} threshold level estimates gets small enough.

[1]In fact, we could have used LSE$_{\text{imp}}$'s selection rule in LSE without any change in the convergence bound of Theorem 1. However, as we will see in the following chapter, the ambiguity-based selection rule performs slightly better in practice.

This is accomplished by bounding $h_t^{opt} - h_t^{pes}$ by a constant multiple of $w_t(\boldsymbol{x}_t)$ (Lemma 11) and using the known decreasing bound on the latter (Lemma 10).

Solution accuracy. To prove ϵ-accuracy of the returned solution with respect to the implicit threshold level, we prove that our optimistic and pessimistic estimates are valid upper and lower bounds of the true implicit level at each iteration, i.e. that $h_t^{opt} \geq h$ and $h_t^{pes} \leq h$, for all $t \geq 1$ (Lemmas 15 and 16). This is the point where keeping all possible maximizers as sampling candidates in LSE$_{\text{imp}}$ is formally required (Lemmas 13 and 14).

5.2 Batch sample selection

In the batch setting, the algorithms are only allowed to use the observed values of previous batches when selecting samples for the current batch. A naive way of extending LSE (resp. LSE$_{\text{imp}}$) to this setting would be to modify the selection rule so that, instead of picking the point with the largest ambiguity (resp. width), it chooses the B highest ranked points. However, this approach tends to select "clusters" of closely located samples with high ambiguities (resp. widths), ignoring the decrease in the estimated variance of a point resulting from sampling another point nearby.

Fortunately, we can handle the above issue by exploiting a key property of GPs, namely that the predictive variances (3.2) depend only on the selected points $\boldsymbol{x}_t$ and not on the observed values y_t at those points. Therefore, even if we do not have available feedback for each selected point up to iteration t, we can still obtain the following useful confidence intervals

$$Q_t^b(\boldsymbol{x}) = \left[\mu_{\text{fb}[t]}(\boldsymbol{x}) \pm \eta_t^{1/2}\sigma_{t-1}(\boldsymbol{x})\right],$$

which combine the most recent available mean estimate with the always up-to-date variance estimate. Here fb$[t]$ is the index of the last available observation expressed using the feedback function fb introduced in Chapter 3. Confidence regions $C_t^b(\boldsymbol{x})$ are defined as before by intersecting successive confidence intervals and are used without any further changes in the algorithms.

The pseudocode of Algorithm 3 highlights the way in which evaluation feedback is obtained in LSE$_{\text{batch}}$. Variable t_{fb} holds the latest step for which there is available feedback at each iteration and the inferred mean is updated whenever new feedback is available, as dictated by fb$[t+1]$. However, note that the inferred variance is updated at each iteration, irrespectively of available feedback. The batch extension of LSE$_{\text{imp}}$ works in a completely analogous way.

Theoretical analysis. Intuitively, to extend the convergence guarantees of the sequential algorithms we have to compensate for using outdated mean estimates by employing a more conservative (i.e. larger) scaling parameter η_t as compared to β_t. This ensures that the resulting confidence regions $C_t^b(\boldsymbol{x})$ still contain $f(\boldsymbol{x})$ with high probability.

To appropriately adjust the confidence interval scaling parameter, in their analysis for extending the GP-UCB algorithm to the batch setting Desautels et al. (2012) utilized the conditional information gain (cf. Chapter 3)

$$I(\boldsymbol{y}_A; f \mid \boldsymbol{y}_{1:\text{fb}[t]}) = H(\boldsymbol{y}_A \mid \boldsymbol{y}_{1:\text{fb}[t]}) - H(\boldsymbol{y}_A \mid f),$$

Algorithm 3 The LSE$_{\text{batch}}$ extension

```
Input: sample set $D$, GP prior ($\mu_0 = 0$, $k$, $\sigma_0$),
       threshold value $h$, accuracy parameter $\epsilon$
Output: predicted sets $\hat{H}$, $\hat{L}$
 1: $H_0 \leftarrow \varnothing,\ L_0 \leftarrow \varnothing,\ U_0 \leftarrow D$
 2: $C_0^b(\boldsymbol{x}) \leftarrow \mathbb{R}$, for all $\boldsymbol{x} \in D$
 3: $t \leftarrow 1$
 4: $t_{\text{fb}} \leftarrow 0$
 5: while $U_{t-1} \neq \varnothing$ do
 6:   $H_t \leftarrow H_{t-1},\ L_t \leftarrow L_{t-1},\ U_t \leftarrow U_{t-1}$
 7:   for all $\boldsymbol{x} \in U_{t-1}$ do
 8:     $C_t^b(\boldsymbol{x}) \leftarrow C_{t-1}^b(\boldsymbol{x}) \cap Q_t^b(\boldsymbol{x})$
 9:     if $\min(C_t^b(\boldsymbol{x})) + \epsilon > h$ then
10:       $U_t \leftarrow U_t \setminus \{\boldsymbol{x}\}$
11:       $H_t \leftarrow H_t \cup \{\boldsymbol{x}\}$
12:     else if $\max(C_t^b(\boldsymbol{x})) - \epsilon \leq h$ then
13:       $U_t \leftarrow U_t \setminus \{\boldsymbol{x}\}$
14:       $L_t \leftarrow L_t \cup \{\boldsymbol{x}\}$
15:     end if
16:   end for
17:   $\boldsymbol{x}_t \leftarrow \text{argmax}_{\boldsymbol{x} \in U_t}(a_t^b(\boldsymbol{x}))$
18:   if $\text{fb}[t+1] > t_{\text{fb}}$ then
19:     for $i = t_{\text{fb}} + 1, \ldots, \text{fb}[t+1]$ do
20:       $y_i \leftarrow f(\boldsymbol{x}_i) + n_i$
21:     end for
22:     Compute $\mu_t(\boldsymbol{x})$ for all $\boldsymbol{x} \in U_t$
23:     $t_{\text{fb}} \leftarrow \text{fb}[t+1]$
24:   end if
25:   Compute $\sigma_t(\boldsymbol{x})$ for all $\boldsymbol{x} \in U_t$
26:   $t \leftarrow t + 1$
27: end while
28: $\hat{H} \leftarrow H_{t-1}$
29: $\hat{L} \leftarrow L_{t-1}$
```

which quantifies the reduction in uncertainty about f by obtaining a number of observations $\boldsymbol{y}_A$, given that we already have observations $\boldsymbol{y}_{1:\text{fb}[t]}$ available. Following a similar treatment, we extend the convergence bound of Theorem 1 to the batch selection setting of LSE$_{\text{batch}}$ via bounding the maximum conditional information gain, resulting in the following theorem.

Theorem 3. *Assume that the feedback delay $t - \text{fb}[t]$ is at most B for all $t \geq 1$, where B is a known constant. Also, assume that for all $t \geq 1$ the maximum conditional mutual information acquired by any set of measurements since the last feedback is bounded by a constant $C \geq 0$, i.e.*

$$\max_{A \subseteq D, |A| \leq B-1} I(f; \boldsymbol{y}_A \mid \boldsymbol{y}_{1:\text{fb}[t]}) \leq C$$

Then, for any $h \in \mathbb{R}$, $\delta \in (0,1)$, and $\epsilon \geq 0$, if $\eta_t = e^C \beta_{fb[t]+1}$, LSE$_{batch}$ terminates after at most T iterations, where T is the smallest positive integer satisfying

$$\frac{T}{\eta_T \gamma_T} \geq \frac{C_1}{4\epsilon^2},$$

where $C_1 = 8/\log(1+\sigma^{-2})$.

Furthermore, with probability at least $1-\delta$*, the algorithm returns an* ϵ*-accurate solution, that is*

$$\Pr\left\{\max_{\boldsymbol{x}\in D} \ell_h(\boldsymbol{x}) \le \epsilon\right\} \ge 1-\delta.$$

The detailed proof of Theorem 3 can be found in Appendix A.3 and a completely analogous theorem can be formulated for extending $\mathsf{LSE}_{\mathsf{imp}}$ to the batch setting.

Note that, as intuitively described above, the scaling parameter η_t has to increase by a factor of e^C to compensate for the outdated mean estimates used in the confidence regions $C_t^b(\boldsymbol{x})$. The bound C on the maximum conditional information gain depends on the GP kernel ("smoother" kernel implies a lower value of C, since few measurements provide sufficient information to predict the values of f) and on the batch size B (a larger batch size implies a higher value of C, because it allows for sampling more points and gaining more information about f).

As a sidenote, Desautels et al. (2012) have shown that, initializing their GP-BUCB algorithm with a number of sequentially selected maximum variance samples, results in a constant factor increase of η_t compared to β_t, independently of B. However, we will not use initialization in our experiments.

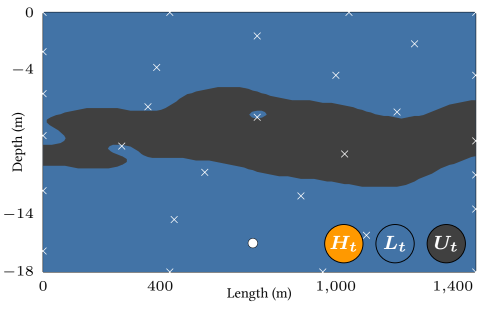
5.3 Path planning

Up to this point, we have assumed that obtaining a sample incurs a fixed cost independent of its location in the input space D. However, in practical applications like the environmental monitoring example of Chapter 1, obtaining a sample also involves moving a mobile sensor to the desired location, which implies an additional traveling cost that depends on that location. We present here a simple way for computing appropriate sampling paths based on the batch sampling extension of the previous section.

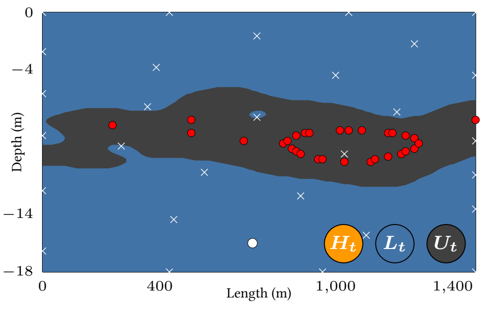
Concretely, to plan a path of length B, we use $\mathsf{LSE}_{\mathsf{batch}}$ to select a batch of B points, but, instead of sampling at those points in the order in which they were selected (lines 19–21 of Algorithm 3), we first use a Euclidean TSP solver to create a path connecting the last sampled point of the previous batch and all points of the current batch.[2]

The above algorithm effectively allows us to trade off a small decrease in the informativeness of acquired samples due to batch selection for a significant reduction in required traveling distance. Furthermore, this trade-off is controlled by the batch size. In the end, the choice of the batch size will depend on the time it takes to travel from one location to the next versus the time it takes to obtain a measurement. In cases where the actual measurement time is large, we would use a smaller batch size, which would result in less samples that are more informative, while having larger traveling paths between measurements. In contrast, if measurement time is negligible (which is the case in our environmental monitoring application), then it

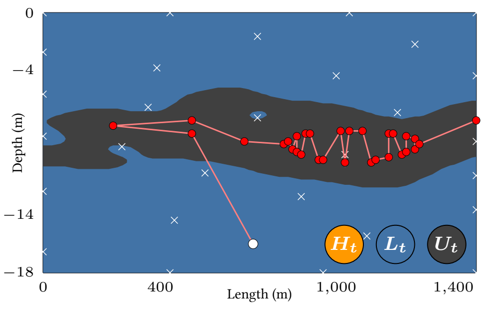
[2]An alternative would be to each time sample the first point in the TSP path and then replan using a new batch. However, this fails because it does not give enough incentive to explore areas of the input space that lie far away from the current position.



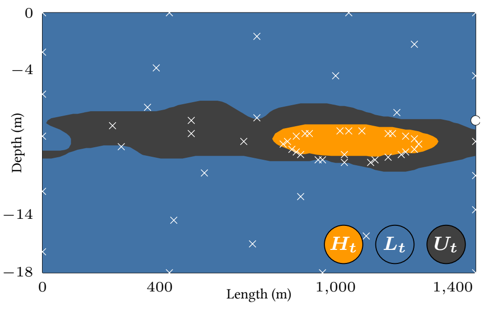
(a) After previous batch



(b) Current batch



(c) TSP path



(d) After current batch

Figure 5.2: Path planning with $\mathsf{LSE}_{\mathsf{batch}}$ using $\epsilon = 0.7$ on a regular grid of 100×100 points sampled from the inferred algae concentration GP of Figure 1.2a. **(a)** The current position, i.e. the last point of the previous batch shown as a white circle. **(b)** The current batch of $B = 30$ points shown as red circles. **(c)** TSP path connecting the points of the current batch with the current position using a TSP path. **(d)** The situation after sampling along the TSP path.

pays off to use a larger batch size in order to reduce the traveling distances, even if this results in a larger amount of total measurements.

Figure 5.2 shows an example of our path planning procedure on the algae concentration dataset for a batch size of $B = 30$. Note that, despite the simplicity of our algorithm, the computed path is reasonably spread across the ambiguous regions of the transect, although some smoothing might need to be applied if it is to be traversed by a real robotic sensor.

6 Experiments

In this section, we present the results of evaluating our proposed algorithms on three real-world datasets and compare them to the state-of-the-art in sequential level set estimation. In more detail, the algorithms and their setup are as follows.

LSE/LSE$_{\text{imp}}$: Since the bound of Theorem 1 is fairly conservative, in our experiments we used a constant value of $\beta_t^{1/2} = 3$, which is somewhat smaller than the values suggested by the theorem.

LSE$_{\text{batch}}$/LSE$_{\text{imp-batch}}$: We used somewhat larger value of $\eta_t^{1/2} = 4$ (compared to $\beta_t^{1/2} = 3$ of the sequential case) and a batch size of $B = 30$.

STR: The state-of-the-art straddle heuristic, as proposed by Bryan et al. (2005), with the selection rule $\boldsymbol{x}_t = \operatorname{argmax}_{\boldsymbol{x} \in D} (1.96\sigma_{t-1}(\boldsymbol{x}) - |\mu_{t-1}(\boldsymbol{x}) - h|)$.

STR$_{\text{imp}}$: For the implicit threshold setting, we have defined a variant of the straddle heuristic that uses at each step the implicitly defined threshold level with respect to the maximum of the inferred mean, i.e. $h_t = \omega \max_{\boldsymbol{x} \in D} \mu_{t-1}(\boldsymbol{x})$.

STR$_{\text{rank}}$/STR$_{\text{batch}}$: We have defined two batch versions of the straddle heuristic: STR$_{\text{rank}}$ selects the $B = 30$ points with the largest straddle score, while STR$_{\text{batch}}$ follows a similar approach to LSE$_{\text{batch}}$ by using the following selection rule

$$\boldsymbol{x}_t = \operatorname*{argmax}_{\boldsymbol{x} \in D} \left(1.96\sigma_{t-1}(\boldsymbol{x}) - |\mu_{\text{fb}[t]}(\boldsymbol{x}) - h|\right).$$

VAR: The maximum variance rule $\boldsymbol{x}_t = \operatorname{argmax}_{\boldsymbol{x} \in D} \sigma_{t-1}(\boldsymbol{x})$.

We implemented all algorithms in MATLAB and used an approximate TSP solver based on genetic programming[1] for the path planning evaluation.

Dataset 1: Network latency. Our first dataset consists of round-trip time (RTT) measurements obtained by "pinging" 1768 servers spread around the world (see Figure 6.1). The sample space consists of the longitude and latitude coordinates of each server, as determined by a commercial geolocation database[2]. Example applications for geographic RTT level set estimation, include monitoring global networks and improving quality of service for applications such as internet telephony or online games. Furthermore, selecting samples in batches results in significant time savings, since sending out and receiving multiple ICMP packets can be virtually performed in parallel.

We used 200 randomly selected measurements to fit suitable hyperparameters for an anisotropic Matérn-5 (Rasmussen & Williams, 2006) kernel by maximum likelihood and the remaining 1568 for evaluation. The threshold level we chose for the experiments was $h = 200$ ms.

[1] http://www.mathworks.com/matlabcentral/fileexchange/21198
[2] http://www.maxmind.com

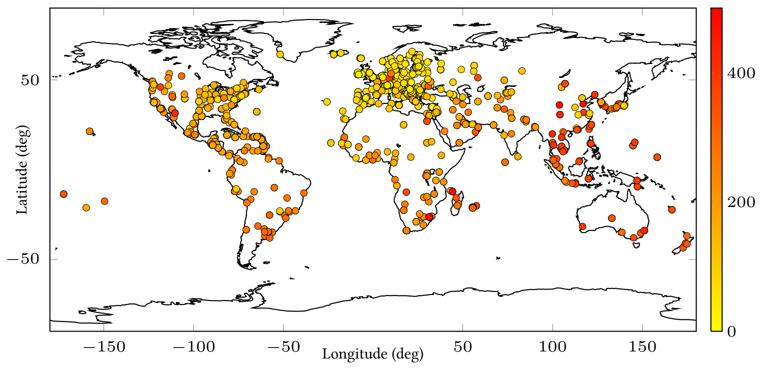


Figure 6.1: Round-trip times in milliseconds obtained by pinging from Zurich 1768 servers around the world.

Datasets 2 & 3: Environmental monitoring. Our second and third datasets come from the domain of environmental monitoring of inland waters and consist of 2024 *in situ* measurements of chlorophyll and *Planktothrix rubescens*[3] concentration respectively, which were collected by an autonomous surface vessel within a vertical transect plane of Lake Zurich (Hitz et al., 2012). As mentioned in the introduction, monitoring chlorophyll and algae concentration is useful in analyzing limnological phenomena such as algal bloom. Since the concentration levels can vary throughout the year, in addition to having a fixed threshold concentration, it can also be useful to be able to detect relative "hotspots" of chlorophyll or algae, i.e. regions of high concentration with respect to the current maximum. Furthermore, selecting batches of points can be used to plan sampling paths and reduce the required traveling distances.

In our evaluation, we used $10,000$ points sampled in a 100×100 grid from the GP posteriors that were derived using the 2024 original measurements (see Figures 1.1a and 1.2a). Again, anisotropic Matérn-5 kernels were used and suitable hyperparameters were fitted by maximizing the likelihood of two different chlorophyll and algae concentration datasets from the same lake. As illustrated in Figures 1.1a and 1.2a, we used explicit threshold levels of $h = 1.3$ RFU (relative fluorescence units) for the chlorophyll dataset and $h = 7$ RFU for the algae concentration dataset. For the implicit threshold experiments, we chose the values of ω so that the resulting implicit levels are identical to the explicit ones, which enables us to compare the two settings on equal ground.

Evaluation methodology. We assess the classification accuracy for all algorithms using the F_1-score, i.e. the harmonic mean of precision and recall, by considering points in the super- and sublevel sets as positives and negatives respectively.

Note that the execution of the algorithms is not deterministic, because ties in the next point selection rules are resolved uniformly at random. Additionally, in the case of the environmental monitoring datasets, sampling from the GP posterior inherently involves randomness because of the noise included in the GP model. For these reasons, in our evaluation we repeated multiple executions of each algorithm. In particular, STR and its extensions as well as VAR were executed 50 times on each

[3]Planktothrix rubescens is a genus of blue-green algae that can produce toxins.

dataset and the F_1-score at each iteration of each execution was computed by classifying points according to the posterior mean of the GP trained on the already selected points, i.e.

$$H_t = \{\boldsymbol{x} \in D \mid \mu_t(\boldsymbol{x}) \geq h\}$$
$$L_t = \{\boldsymbol{x} \in D \mid \mu_t(\boldsymbol{x}) < h\}.$$

On the other hand, unless explicitly stated differently, LSE and its extensions were evaluated by repeatedly executing each algorithm for a range of values of the accuracy parameter ϵ and recording the total number of samples and the F_1-score after each execution's termination (in total about 2000 executions per algorithm per dataset). The parameter ϵ was chosen to increase exponentially within a suitable range depending on the experiment (roughly between 1% and 20% of the respective threshold level).

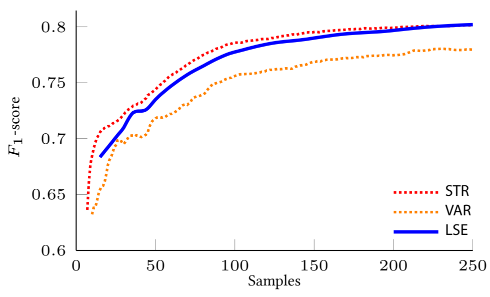
6.1 Results I: Explicit threshold level

Sequential sampling. Figures 6.2a–6.2c compare the performance of the strictly sequential algorithms on the three datasets. On the first two datasets LSE and STR are comparable in performance, with LSE doing slightly worse on the latency dataset. However, the situation is different on the third dataset, where STR performs extremely poorly. The reason is that the algorithm can get stuck in situations where, according to the straddle score, it is favorable to continue sampling points near the currently inferred level set, instead of sampling points of high variance at yet unexplored regions that lie far away from that level set.

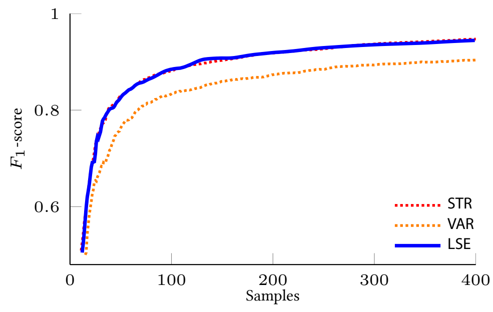
Figure 6.2d shows an example of such an execution, where STR has heavily sampled the left region of the level set, but has completely missed the part that lies on the right of the transect (cf. Figure 1.2a). In Figure 6.2g we depict as a scatter plot the detailed results of the 50 STR executions. Note that about a third of the executions do not manage to achieve an F_1-score of 0.4, even after 400 iterations, i.e. they miss the right part of the level set as explained above, and another third only achieve an F_1-score of 0.8, i.e. they miss the left part of the level set. On the other hand, as shown in Figure 6.2h, this problem never occurs in LSE, because of its classification regime, which implicitly forces exploration by excluding points that have already been classified from being sampled.

Although VAR is commonly used for estimating functions over their entire domain, it is clearly outperformed in our experiments, because it does not sufficiently focus on accurately estimating the desired level set, but rather samples almost uniformly over the sample space.

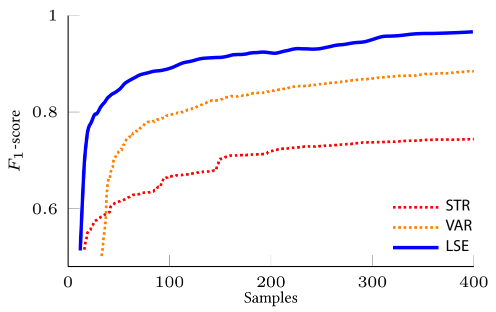
The effect of ϵ. To illustrate how the choice of ϵ affects the results of LSE, we present in Figure 6.3 two scatter plots, similar to the one in Figure 6.2h, that depict the results of running LSE on the two environmental monitoring datasets for different values of ϵ. As before, each point in the plot corresponds to the result of one execution of LSE, but in the new figures we have colored the points according to the value of ϵ used for each execution. In our theoretical analysis we have seen that ϵ represents a tradeoff parameter between classification accuracy and sampling cost. The two scatter plots give experimental evidence that this is indeed the case and, moreover, that the transition when varying ϵ is fairly smooth.



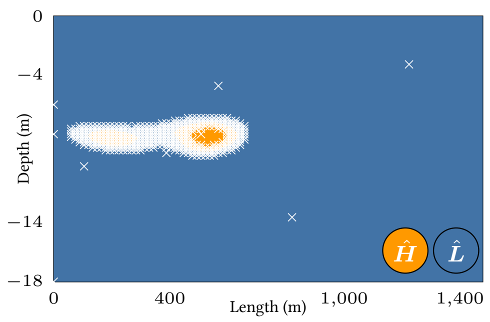
(a) [N] Mean performance



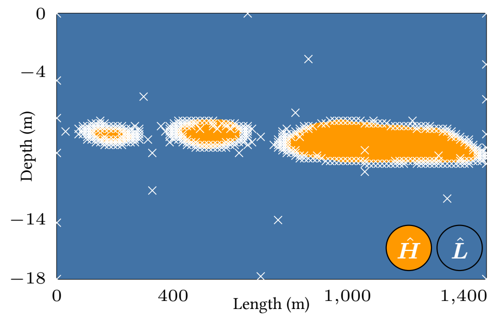
(b) [C] Mean performance



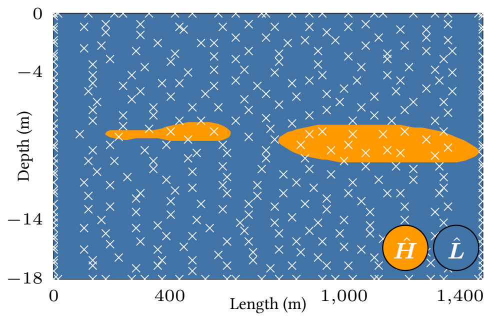
(c) [A] Mean performance



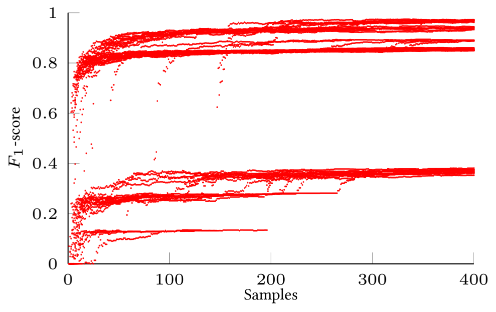
(d) [A] STR after $t = 400$ iterations



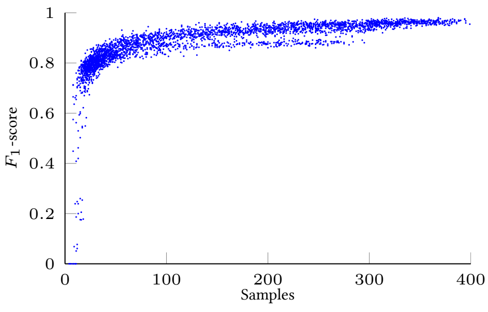
(e) [A] LSE after $t = 374$ iterations



(f) [A] VAR after $t = 400$ iterations

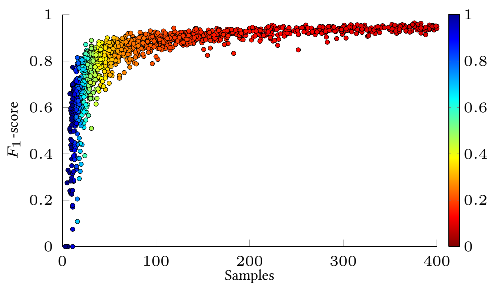


(g) [A] STR scatter plot

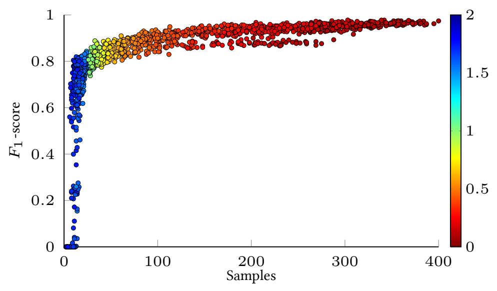


(h) [A] LSE scatter plot

Figure 6.2: Performance of explicit threshold sequential algorithms on the network latency [N], chlorophyll concentration [C], and algae concentration [A] datasets. **(a), (b)** LSE and STR are comparable and both clearly outperform VAR. **(c)** STR performs poorly due to limited exploration. **(d)** An example of a STR execution getting stuck: even after 400 iterations it has only achieved an F_1-score of 0.37. **(e)** An example of a LSE execution without any issues. **(f)** An example of a VAR execution: the sample space is sampled rather uniformly. **(g), (h)** The results of (c) in more detail: about a third of STR's executions achieve an F_1-score of less than 0.4, while LSE always achieves an F_1-score of at least 0.95.

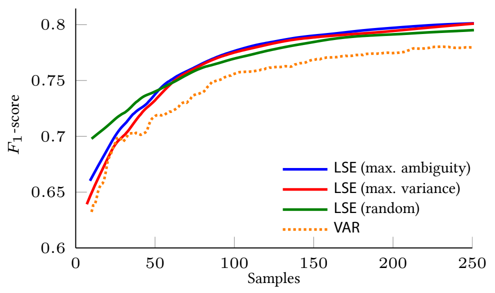


(a) [C] LSE scatter plot

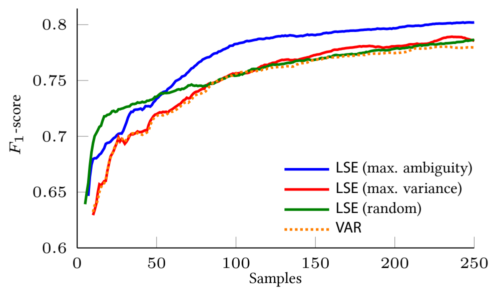


(b) [A] LSE scatter plot

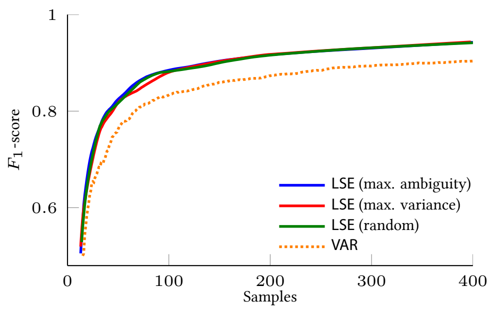
Figure 6.3: The effect of ϵ on the sampling cost and classification accuracy of LSE on the two environmental monitoring datasets.



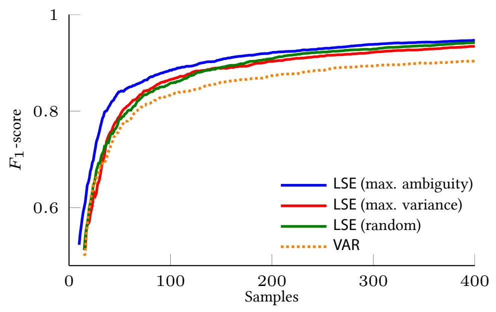
(a) [N] Varying ϵ



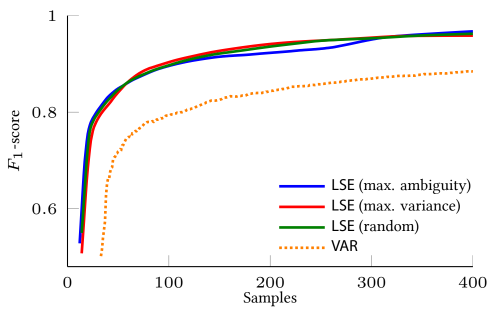
(b) [N] Fixed ϵ



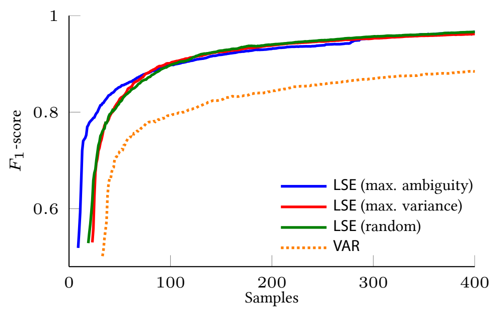
(c) [C] Varying ϵ



(d) [C] Fixed ϵ



(e) [A] Varying ϵ



(f) [A] Fixed ϵ

Figure 6.4: Performance of different LSE selection rules on the three datasets. **(a), (c), (e)** For varying ϵ there is no notable difference between the three selection rules. **(b), (d), (f)** When using a small fixed value of ϵ and evaluating during each LSE execution, the maximum ambiguity selection rule has an advantage over the other two rules.

Sample selection rules. In Figures 6.4a, 6.4c, and 6.4e we compare the performance of LSE using different sample selection rules (line 17 of Algorithm 1). In particular, we compare the maximum ambiguity rule (as presented in LSE), the maximum variance rule (which we also use in LSE_{imp}), and the rule of randomly choosing the next sample from U_t. According to the above figures, all three rules seem to be performing almost equally well on all three datasets.

To further investigate the potential effect of the sampling rules, we also ran a different set of experiments, where, instead of varying ϵ and evaluating the F_1-score after termination, we fixed ϵ to a small value ($h/50$) and evaluated the different LSE variants during their execution using the GP posterior mean to classify points into the super- and sublevel sets (as explained before for STR and VAR). In other words, we evaluated the performance obtained when the tuning of ϵ for controlling the required number of samples is ignored.

The results are presented in Figures 6.4b, 6.4d, and 6.4f. In this case, the maximum ambiguity rule outperforms the other two rules on all three datasets. However, note that in the two environmental monitoring datasets the difference is fairly small and the other two rules still perform considerably better than VAR.

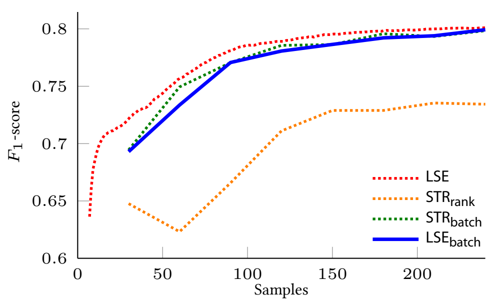
To connect the above observations with the operation of LSE, recall that the algorithm is largely based on the progressive classification of points into H_t and L_t, a process that focuses the selection of the next sample at each step on a set of "interesting" (w.r.t. to the sought after level set), yet unclassified points U_t. On the other hand, the way in which the sample is selected from U_t seems to be of secondary importance and selecting using maximum ambiguity might provide a small performance benefit in some cases.

Batch sampling. In Figures 6.5a–6.5c we show the performance of the explicit threshold batch algorithms on the three datasets. The $\text{LSE}_{\text{batch}}$ and $\text{STR}_{\text{batch}}$ algorithms, which use the always up-to-date variance estimates for selecting batches, achieve similar performance. Furthermore, there is only a slight performance penalty when compared to the strictly sequential LSE, which can easily be outweighed by the benefits of batch point selection (e.g. in the network latency application, the batch algorithms would have about $B = 30$ times higher throughput). On the other hand, as expected, the STR_{rank} algorithm performs significantly worse than the other two batch algorithms, since it selects a lot of redundant samples in areas of high straddle score, as depicted in Figure 6.5d (cf. Section 5.2).

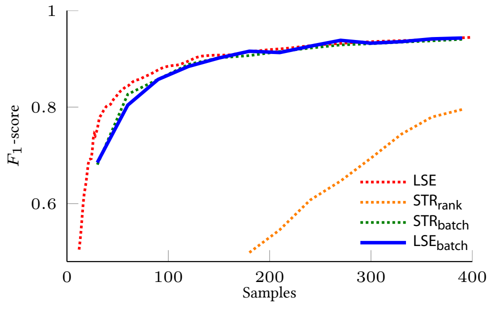
6.2 Results II: Implicit threshold level

In Figures 6.6a and 6.6b we present the results of executing the implicit threshold algorithms on the two environmental monitoring datasets. The difficulty of estimating the function maximum at the same time as performing classification with respect to the implicit threshold level is manifested in the notably larger sampling cost of LSE_{imp} required to achieve high accuracy compared to the explicit threshold experiments. As before, the batch version of LSE_{imp} is only slightly worse that its sequential counterpart.

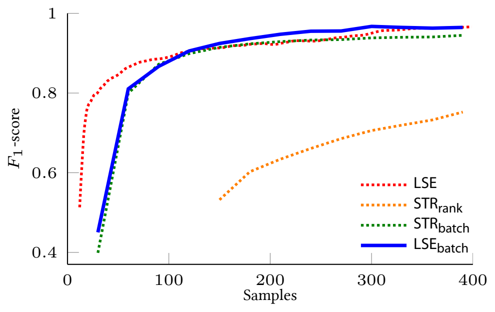
More importantly, the naive STR_{imp} algorithm fails to achieve high accuracy, as it mostly infers wrong estimates of the function maximum and never recovers, since the (modified) straddle rule gives no incentive for sampling near the maximum. This



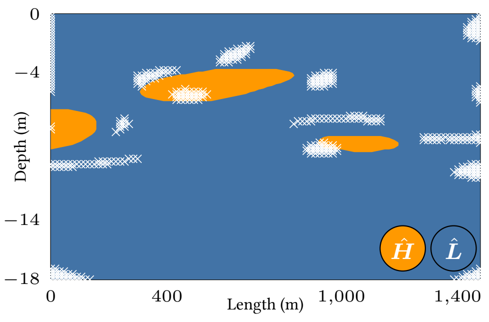
(a) [N] Mean performance



(b) [C] Mean performance



(c) [A] Mean performance

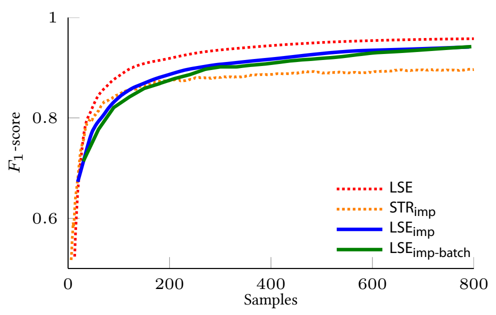


(d) [C] $\mathsf{STR}_{\mathsf{rank}}$ after $t = 400$ iterations

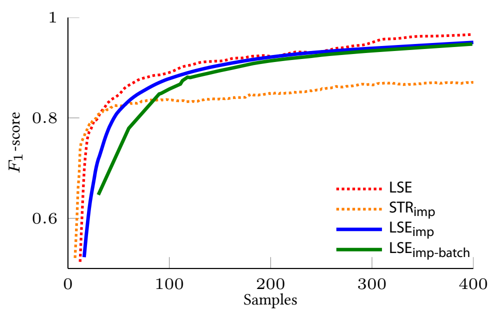
Figure 6.5: Performance of explicit threshold batch algorithms on the three datasets. **(a)–(c)** $\mathsf{LSE}_{\mathsf{batch}}$ and $\mathsf{STR}_{\mathsf{batch}}$ achieve comparable performance, which is slightly worse than that of their sequential counterparts, but significantly better than that of the naive $\mathsf{STR}_{\mathsf{rank}}$ algorithm. **(d)** The redundant sampling behavior of $\mathsf{STR}_{\mathsf{rank}}$ resulting from not using up-to-date variance estimates.

is not very clear in the case of the chlorophyll dataset, because the function is fairly smooth in the vicinity of its maxima and, thus, $\mathsf{STR}_{\mathsf{imp}}$ is usually lucky enough to obtain a rather accurate estimate of the function maximum. However, as can be seen in Figure 6.6c, in the case of the algae dataset most of the executions of $\mathsf{STR}_{\mathsf{imp}}$ achieve an F_1-score of only about 0.8, even after 400 samples. The inability of $\mathsf{STR}_{\mathsf{imp}}$ to recover from a wrong threshold estimate is illustrated by the fact that most of these executions have already achieved this F_1-score after less than 100 samples, but show no tendency for improvement thereafter. The behavior of $\mathsf{LSE}_{\mathsf{imp}}$ depicted in more detail in Figure 6.6c is very different in that it starts considerably slower, but always achieves F_1-scores of at least 0.9 after 400 samples.

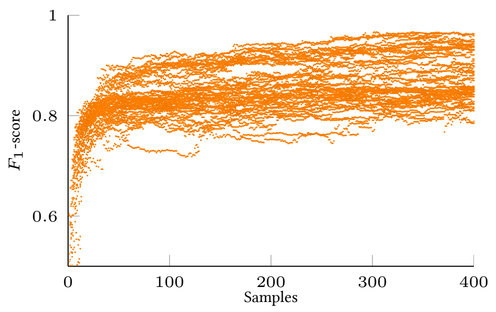
Figure 6.7c shows the estimated implicit threshold level for an example execution of $\mathsf{STR}_{\mathsf{imp}}$ on the algae concentration dataset. Note that the inferred level is smaller than the true level $h = 7$ and, even worse, seems to deviate further away from the true level over time. In contrast, as shown in Figure 6.7d, the optimistic and pessimistic estimates used by $\mathsf{LSE}_{\mathsf{imp}}$ correctly bound the true level from above and below respectively and get more accurate over time (as predicted by theory; cf. Lemma 11 and Lemmas 15–16), thus allowing for better classification results.



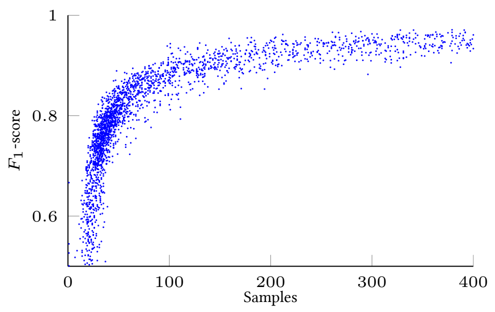
(a) [C] Mean performance



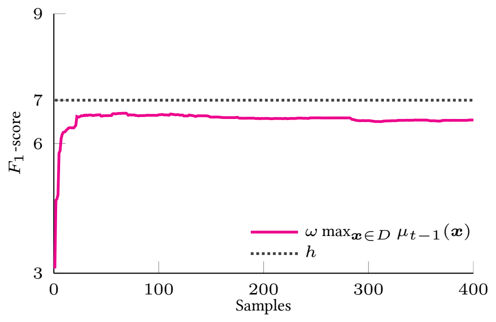
(b) [A] Mean performance



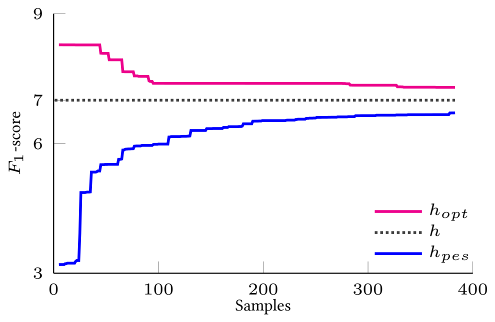
(c) [A] $\mathsf{STR_{imp}}$ scatter



(d) [A] $\mathsf{LSE_{imp}}$ scatter



(e) [A] $\mathsf{STR_{imp}}$ threshold level estimate



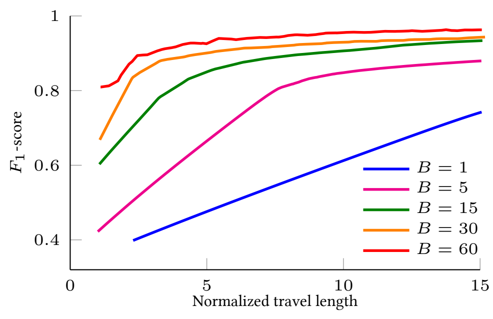
(f) [A] $\mathsf{LSE_{imp}}$ threshold level estimates

Figure 6.6: Performance of implicit threshold algorithms on the environmental monitoring datasets. **(a)**, **(b)** $\mathsf{LSE_{imp}}$ and $\mathsf{LSE_{imp\text{-}batch}}$ perform somewhat worse than their explicit threshold counterparts, while $\mathsf{STR_{imp}}$ performs notably worse. **(c)** Most of $\mathsf{STR_{imp}}$ executions fail to achieve high F_1-scores and top off at about 0.8. **(d)** $\mathsf{LSE_{imp}}$ always achieves an F_1-score of at least 0.9 after 400 iterations. **(e)** $\mathsf{STR_{imp}}$'s implicit threshold level estimate is lower than the true level and becomes worse over time. **(f)** $\mathsf{LSE_{imp}}$'s implicit threshold level estimates correctly bound the true level and converge towards it over time.

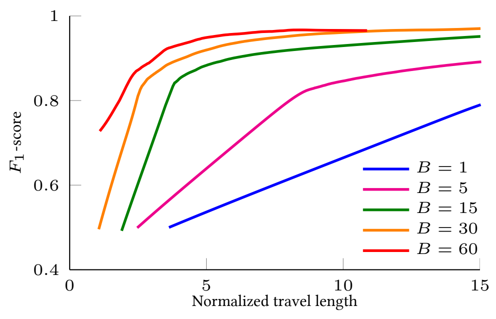
6.3 Results III: Path planning

We now present the results from applying our batch-based path planning algorithm on the two environmental monitoring datasets. Since path planning executions are costly, we used a fixed small value of ϵ for each dataset and executed the algorithm 30 times for each of five different batch sizes, while measuring total traveled path length and F_1-score after each path planning iteration (similarly to what we did in Figure 6.4).

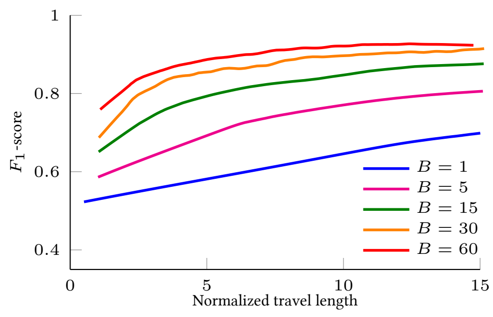
Figure 6.7 displays the dramatically reduced travel lengths by using batches of samples for path planning. As an example, in Figure 6.7b we can see that planning with $B = 30$ samples at a time achieves an F_1-score of 0.9 after a travel length of about 4 transect lengths, while planning with $B = 5$ (or even worse sequentially) does not achieve that accuracy even after 15 transect lengths. Also note that the effect of the batch size on the travel lengths seems to have a diminishing returns property: for example, increasing from $B = 5$ to $B = 15$ makes in all cases a notably larger difference than increasing from $B = 15$ to $B = 60$.



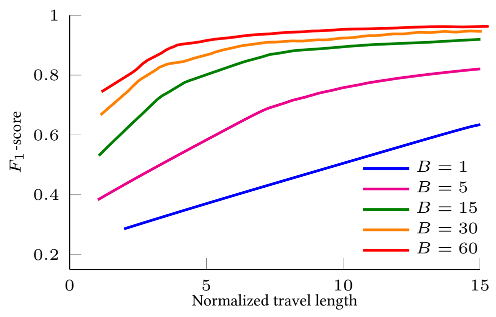
(a) [C] Explicit threshold



(b) [A] Explicit threshold



(c) [C] Implicit threshold



(d) [A] Implicit threshold

Figure 6.7: Performance of the path planning algorithm for different batch sizes on the two environmental monitoring datasets. The total lengths of the sampling paths are normalized by the lake transect length (1478 m). Note the dramatically reduced travel lengths as the batch size is increased ($B = 1$ corresponds to strictly sequential sampling).

7 Conclusion

Summary. We presented LSE, an algorithm for estimating level sets, which operates based on confidence bounds derived by modeling the target function as a GP. We considered for the first time the challenge of implicitly defined threshold levels and extended LSE to this more complex setting. We also showed how both algorithms can be extended to select samples in batches and applied this to a simple yet effective path-planning algorithm. In addition, we provided theoretical bounds on the number of iterations required to obtain an ϵ-accurate solution when the target function is sampled from a known GP. The experiments on two real-world applications showed that LSE is competitive with the state-of-the-art and in some cases shows improved performance. Furthermore, the extensions we presented are successful in handling the corresponding problem variants and perform significantly better than naive baselines. We believe our results provide an important step towards addressing complex real-world information gathering problems.

Future work. As we have seen in the analysis of LSE_{imp}, its convergence to an accurate solution is largely dictated by the gap $h_t^{opt} - h_t^{pes}$ and its rate of decrease. The algorithm as presented gives in a sense "equal weight" to sampling from ambiguous regions near the estimated level set and to sampling near potential maximizers by using maximum variance sampling among the points in Z_t. However, we expect that biasing the search early toward maximizers would result in faster accurate estimates of the maximum, leading to a more rapid decrease of the above gap and, therefore, faster overall convergence. Some preliminary experiments with using GP-UCB's selection rule every k_t iterations, where k_t starts with a small value and increases over time, justify the above claims and lead to improved performance. It would be interesting to theoretically explore such a scheme and see if any improved convergence guarantees can be proven.

The presented method for path planning via batch point selection has a number of shortcomings when applied in practice. First, since the selection of each batch is decoupled from its evaluation, there is no way to adjust the path according to the measurements obtained during its traversal. Second, while in practice we could obtain additional samples during the traversal of an edge of the path (assuming that measurements are not very costly), there is no easy way to incorporate this into our batch-based algorithm. In particular, one would have to take into account the estimated information gained from the additional samples when computing each path. Last, as can be seen in the example of Figure 5.2, the resulting TSP path may contain several abrupt changes of direction that are not achievable in practice due to kinematic constraints of the mobile sensor equipment. To deal with the above issues we are working on a graph-based path-planning algorithm, which (1) explicitly incorporates kinematic constraints into its graph structure, (2) takes into account the additional samples acquired during edge traversal when computing the path, and (3) replans after each step of the path using the newly obtained measurements.

A Detailed proofs

A.1 Proof of Theorem 1

Lemma 1. *For any $\delta \in (0,1)$, if $\beta_t = 2\log(|D|\pi_t/\delta)$, where $\sum_{t\geq 1}\pi_t^{-1} = 1$ and $\pi_t > 0$, then the following holds with probability at least $1-\delta$*

$$|f(\boldsymbol{x}) - \mu_{t-1}(\boldsymbol{x})| \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}), \ \forall \boldsymbol{x} \in D \ \forall t \geq 1.$$

In particular, we can choose $\pi_t = \pi^2 t^2/6$.

Proof. See Lemma 5.1 in (Srinivas et al., 2010). □

Corollary 1. *For any $\delta \in (0,1)$ and β_t as above, the following holds with probability at least $1-\delta$*

$$f(\boldsymbol{x}) \in C_t(\boldsymbol{x}), \ \forall \boldsymbol{x} \in D \ \forall t \geq 1.$$

Lemma 2. *The following holds for any $t \geq 1$*

$$a_t(\boldsymbol{x}_t) \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t).$$

Proof. By the definition of ambiguity

$$\begin{aligned} a_t(\boldsymbol{x}_t) &= \min\{\max(C_t(\boldsymbol{x}_t)) - h, h - \min(C_t(\boldsymbol{x}_t))\} \\ &\leq (\max(C_t(\boldsymbol{x}_t)) - \min(C_t(\boldsymbol{x}_t)))/2 \\ &\leq (\max(Q_t(\boldsymbol{x}_t)) - \min(Q_t(\boldsymbol{x}_t)))/2 \\ &= \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t). \end{aligned}$$

□

Lemma 3. *While running LSE, $a_t(\boldsymbol{x}_t)$ is nonincreasing in t.*

Proof. From the definition of the confidence region of $\boldsymbol{x}_t$ via successive intersections $C_t(\boldsymbol{x}_t) = C_{t-1}(\boldsymbol{x}_t) \cap Q_t(\boldsymbol{x}_t)$, it follows that

$$\left.\begin{array}{l} \max(C_t(\boldsymbol{x}_t)) \leq \max(C_{t-1}(\boldsymbol{x}_t)) \\ \min(C_t(\boldsymbol{x}_t)) \geq \min(C_{t-1}(\boldsymbol{x}_t)) \end{array}\right\} \Rightarrow a_t(\boldsymbol{x}_t) \leq a_{t-1}(\boldsymbol{x}_t)$$

Furthermore, from the selection rule used in LSE $\boldsymbol{x}_t = \operatorname{argmax}_{\boldsymbol{x}\in U_t}(a_t(\boldsymbol{x}))$ and the monotonicity of U_t ($U_t \subseteq U_{t-1}$), it follows that $a_{t-1}(\boldsymbol{x}_t) \leq a_{t-1}(\boldsymbol{x}_{t-1})$. □

Lemma 4. *Denoting $\boldsymbol{y}_t = (y_i)_{1\leq i\leq t}$ and $\boldsymbol{f}_t = (f(\boldsymbol{x}_i))_{1\leq i\leq t}$, the information gain for the selected points up to iteration t can be expressed in terms of the predictive variances as follows*

$$I(\boldsymbol{y}_t; \boldsymbol{f}_t) = \frac{1}{2}\sum_{i=1}^{t}\log(1 + \sigma^{-2}\sigma_{i-1}^2(\boldsymbol{x}_i)).$$

Proof. See Lemma 5.3 in (Srinivas et al., 2010). □

Lemma 5. *While running LSE with β_t as in Lemma 1, it holds that*

$$a_t(\boldsymbol{x}_t) \leq \sqrt{\frac{C_1 \beta_t \gamma_t}{4t}}, \ \forall t \geq 1,$$

where $C_1 = 8/\log(1+\sigma^{-2})$.

Proof. Similarly to Lemma 5.4 in (Srinivas et al., 2010), from Lemma 2 it follows that for any $i \geq 1$

$$\begin{aligned} a_i^2(\boldsymbol{x}_i) &\leq \beta_i \sigma_{i-1}^2(\boldsymbol{x}_i) \\ &\leq \beta_i \sigma^2 (\sigma^{-2} \sigma_{i-1}^2(\boldsymbol{x}_i)) \\ &\leq \beta_i \sigma^2 C_2 \log(1 + \sigma^{-2}\sigma_{i-1}^2(\boldsymbol{x}_i)), \end{aligned}$$

where $C_2 = \sigma^{-2}/\log(1+\sigma^{-2})$. Using Lemma 4 in the above expression, the fact that β_i is nondecreasing in i, and defining $C_1 = 8\sigma^2 C_2$, we get for any $t \geq 1$

$$\begin{aligned} C_1 \beta_t \gamma_t &\geq C_1 \beta_t I(\boldsymbol{y}_t; \boldsymbol{f}_t) \\ &\geq 4 \sum_{i=1}^{t} a_i^2(\boldsymbol{x}_i) \\ &\geq \frac{4}{t} \left(\sum_{i=1}^{t} a_i(\boldsymbol{x}_i) \right)^2 && \text{(by Cauchy-Schwarz)} \\ &= 4t \left(\frac{1}{t} \sum_{i=1}^{t} a_i(\boldsymbol{x}_i) \right)^2 \\ &\geq 4t a_t^2(\boldsymbol{x}_t) && \text{(by Lemma 3)} \end{aligned}$$

□

Lemma 6. *While running LSE, if for some $t \geq 1$, $a_t(\boldsymbol{x}_t) \leq \epsilon$, then $U_{t+1} = \varnothing$.*

Proof. Assume that $U_{t+1} \neq \varnothing$, i.e. there exists a point $\boldsymbol{x} \in U_t$, which does not meet the classification conditions (lines 12 and 15) of Algorithm 1. Consequently, that point satisfies $\max(C_{t+1}(\boldsymbol{x})) > h + \epsilon$ and $\min(C_{t+1}(\boldsymbol{x})) < h - \epsilon$. It follows that

$$\begin{aligned} \epsilon &< \min\{\max(C_{t+1}(\boldsymbol{x}_t)) - h, h - \min(C_{t+1}(\boldsymbol{x}_t))\} \\ &= a_{t+1}(\boldsymbol{x}) \\ &\leq a_t(\boldsymbol{x}) \\ &\leq a_t(\boldsymbol{x}_t), && \text{(by LSE's selection rule)} \end{aligned}$$

which contradicts the lemma's assumption. □

Corollary 2. *The LSE algorithm terminates after at most T iterations, where T is the smallest positive integer satisfying*

$$\frac{T}{\beta_T \gamma_T} \geq \frac{C_1}{4\epsilon^2}.$$

Lemma 7. *For any $h \in \mathbb{R}$ and $\delta \in (0,1)$, and $\epsilon > 0$, after running LSE with β_t as in Lemma 1, with probability at least $1-\delta$ the returned solution is ϵ-accurate, that is*

$$\Pr\left\{\max_{\boldsymbol{x}\in D} \ell_h(\boldsymbol{x}) \leq \epsilon\right\} \geq 1-\delta.$$

Proof. The lemma follows directly from Corollary 1 and the classification conditions (lines 12 and 15) of Algorithm 1. □

Theorem 1 follows by combining Corollary 2 and Lemma 7.

A.2 Proof of Theorem 2

Definition 1. *We label the inequalities that take part in LSE$_{imp}$'s classification rules as follows*

$$\min(C_t(\boldsymbol{x})) + \epsilon \geq h_t^{opt} \tag{Q1}$$
$$\max(C_t(\boldsymbol{x})) - \epsilon \leq h_t^{pes} \tag{Q2}$$
$$\max(C_t(\boldsymbol{x})) < f_t^{pes}. \tag{Q3}$$

Furthermore, we redefine here for convenience the following quantities from the main text

$$\begin{aligned}
Z_t &= U_t \cup H_t^M \cup H_t^L \\
h &= \omega \max_{\boldsymbol{x}\in D} f(\boldsymbol{x}) \\
f_t^{opt} &= \max_{Z_{t-1}} \max(C_t(\boldsymbol{x})) \\
h_t^{opt} &= \omega f_t^{opt} \\
f_t^{pes} &= \max_{Z_{t-1}} \min(C_t(\boldsymbol{x})) \\
h_t^{pes} &= \omega f_t^{pes}.
\end{aligned}$$

Lemma 8. *The following holds for any $t \geq 1$*

$$w_t(\boldsymbol{x}_t) \leq \beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t).$$

Proof. By the definition of the confidence region width

$$\begin{aligned}
w_t(\boldsymbol{x}_t) &= \max(C_t(\boldsymbol{x}_t)) - \min(C_t(\boldsymbol{x}_t)) \\
&\leq \max(Q_t(\boldsymbol{x}_t)) - \min(Q_t(\boldsymbol{x}_t)) \\
&= 2\beta_t^{1/2}\sigma_{t-1}(\boldsymbol{x}_t).
\end{aligned}$$

□

Lemma 9. *While running LSE$_{imp}$, $w_t(\boldsymbol{x}_t)$ is nonincreasing in t.*

Proof. Completely analogous to the proof of Lemma 3. □

Lemma 10. *While running LSE*$_{imp}$ *with* β_t *as in Lemma 1, it holds that*

$$w_t(\boldsymbol{x}_t) \leq \sqrt{\frac{C_1 \beta_t \gamma_t}{t}},\ \forall t \geq 1,$$

where $C_1 = 8/\log(1+\sigma^{-2})$.

Proof. Completely analogous to the proof of Lemma 5, with the only difference being a factor of 2 in the bound of Lemma 8 compared to Lemma 2. □

Lemma 11. *While running LSE*$_{imp}$

$$h_t^{opt} - h_t^{pes} \leq \omega w_t(\boldsymbol{x}_t),\ \forall t \geq 1.$$

Proof. If we define

$$\hat{\boldsymbol{x}} = \operatorname*{argmax}_{\boldsymbol{x} \in Z_{t-1}} \max(C_t(\boldsymbol{x})), \tag{A.1}$$

then by (Q3) it follows that $\hat{\boldsymbol{x}} \in Z_t$. Consequently, we get

$$\begin{aligned}
& f_t^{opt} - f_t^{pes} \\
&= \max_{\boldsymbol{x} \in Z_{t-1}} \max(C_t(\boldsymbol{x})) - \max_{\boldsymbol{x} \in Z_{t-1}} \min(C_t(\boldsymbol{x})) \\
&= \max(C_t(\hat{\boldsymbol{x}})) - \max_{\boldsymbol{x} \in Z_{t-1}} \min(C_t(\boldsymbol{x})) && \text{(by (A.1))} \\
&\leq \max(C_t(\hat{\boldsymbol{x}})) - \min(C_t(\hat{\boldsymbol{x}})) \\
&= w_t(\hat{\boldsymbol{x}}) \\
&\leq w_t(\boldsymbol{x}_t), && \text{(by } \hat{\boldsymbol{x}} \in Z_t\text{)}
\end{aligned}$$

and, therefore, $h_t^{opt} - h_t^{pes} = \omega \left(f_t^{opt} - f_t^{pes} \right) \leq \omega w_t(\boldsymbol{x}_t)$. □

Lemma 12. *While running LSE*$_{imp}$*, if for some* $t \geq 1$, $w_t(\boldsymbol{x}_t) \leq 2\epsilon/(1+\omega)$*, then* $U_{t+1} = \varnothing$.

Proof. Assume that $U_{t+1} \neq \varnothing$, i.e. there exists a point $\boldsymbol{x} \in U_t \subseteq Z_t$, which does not meet (Q1) or (Q2). Consequently, that point satisfies $\min(C_{t+1}(\boldsymbol{x})) < h_{t+1}^{opt} - \epsilon$ and $\max(C_{t+1}(\boldsymbol{x})) > h_{t+1}^{pes} + \epsilon$. It follows that

$$\begin{aligned}
2\epsilon &< h_{t+1}^{opt} - h_{t+1}^{pes} + \max(C_{t+1}(\boldsymbol{x})) - \min(C_{t+1}(\boldsymbol{x})) \\
&\leq h_{t+1}^{opt} - h_{t+1}^{pes} + w_{t+1}(\boldsymbol{x}) \\
&\leq h_{t+1}^{opt} - h_{t+1}^{pes} + w_t(\boldsymbol{x}) \\
&\leq h_{t+1}^{opt} - h_{t+1}^{pes} + w_t(\boldsymbol{x}_t) && \text{(by LSE}_{\text{imp}}\text{'s selection rule)} \\
&\leq \omega w_{t+1}(\boldsymbol{x}_{t+1}) + w_t(\boldsymbol{x}_t) && \text{(by Lemma 11)} \\
&\leq \omega w_t(\boldsymbol{x}_t) + w_t(\boldsymbol{x}_t) && \text{(by Lemma 9)} \\
&\leq (1+\omega) w_t(\boldsymbol{x}_t),
\end{aligned}$$

which contradicts the lemma's assumption. □

Corollary 3. *The LSE$_{imp}$ algorithm terminates after at most T iterations, where T is the smallest positive integer satisfying*

$$\frac{T}{\beta_T \gamma_T} \geq \frac{C_1(1+\omega)^2}{4\epsilon}.$$

Lemma 13. *While running LSE$_{imp}$, f_t^{pes} is nondecreasing in t.*

Proof. Assume that at some iteration t

$$\boldsymbol{x} = \operatorname*{argmax}_{\boldsymbol{x} \in Z_{t-1}} \min(C_t(\boldsymbol{x})). \tag{A.2}$$

Since $\min(C_t(\boldsymbol{x}))$ is nondecreasing in t, to have $f_{t+1}^{pes} < f_t^{pes}$ would mean that $\boldsymbol{x} \notin Z_t$. That, in turn, implies that $\boldsymbol{x}$ was moved to H_t or L_t, therefore (Q3) was satisfied

$$\begin{aligned}
\max_{\boldsymbol{x} \in Z_{t-1}} \min(C_t(\boldsymbol{x})) &> \max(C_t(\boldsymbol{x})) \\
&\geq \min(C_t(\boldsymbol{x})) \\
&= \max_{\boldsymbol{x} \in Z_{t-1}} \min(C_t(\boldsymbol{x})), && \text{(by (A.2))}
\end{aligned}$$

which is a contradiction and proves our lemma. □

Lemma 14. *While running LSE$_{imp}$*

$$f_t^{opt} = \max_{\boldsymbol{x} \in D} \max(C_t(\boldsymbol{x})), \forall t \geq 1.$$

Proof. The "$\leq$" follows from $Z_{t-1} \subseteq D$. Now, assume that "$<$" holds, i.e. there exists an $\boldsymbol{x} \in D \setminus Z_{t-1}$, such that

$$\max_{\boldsymbol{x} \in Z_{t-1}} \max(C_t(\boldsymbol{x})) < \max C_t(\boldsymbol{x}). \tag{A.3}$$

The fact that $\boldsymbol{x} \in D \setminus Z_{t-1}$ implies that $\boldsymbol{x}$ was moved during some iteration $i \leq t$ to H_i or L_i, therefore $\boldsymbol{x}$ satisfied (Q3) at that iteration. Putting everything together, we get

$$\begin{aligned}
\max_{\boldsymbol{x} \in Z_{t-1}} \max(C_t(\boldsymbol{x})) &< \max C_t(\boldsymbol{x}) && \text{(by (A.3))} \\
&\leq \max C_i(\boldsymbol{x}) \\
&\leq \max_{\boldsymbol{x} \in Z_{i-1}} \min(C_i(\boldsymbol{x})) && \text{(by (Q3))} \\
&\leq \max_{\boldsymbol{x} \in Z_{t-1}} \min(C_t(\boldsymbol{x})) && \text{(by Lemma 13)} \\
&\leq \max_{\boldsymbol{x} \in Z_{t-1}} \max(C_t(\boldsymbol{x})),
\end{aligned}$$

which is a contradiction and proves the lemma. □

Lemma 15. *While running LSE$_{imp}$, the following holds with probability at least $1-\delta$*

$$h_t^{opt} \geq h, \; \forall t \geq 1.$$

Proof. The following (in)equalities hold with probability at least $1-\delta$

$$\begin{aligned} h_t^{opt} &= \omega \max_{\boldsymbol{x}\in Z_{t-1}} \max(C_t(\boldsymbol{x})) \\ &= \omega \max_{\boldsymbol{x}\in D} \max(C_t(\boldsymbol{x})) && \text{(by Lemma 14)} \\ &\geq \omega \max_{\boldsymbol{x}\in D} f(\boldsymbol{x}) && \text{(by Corollary 1)} \\ &= h. \end{aligned}$$

□

Lemma 16. *While running LSE*$_{imp}$*, the following holds with probability at least* $1-\delta$

$$h_t^{pes} \leq h,\ \forall t \geq 1.$$

Proof. The following (in)equalities hold with probability at least $1-\delta$

$$\begin{aligned} h_t^{pes} &= \omega \max_{\boldsymbol{x}\in Z_{t-1}} \min(C_t(\boldsymbol{x})) \\ &\leq \omega \max_{\boldsymbol{x}\in Z_{t-1}} f(\boldsymbol{x}) && \text{(by Corollary 1)} \\ &\leq \omega \max_{\boldsymbol{x}\in D} f(\boldsymbol{x}) && \text{(by } Z_{t-1}\subseteq D) \\ &= h. \end{aligned}$$

□

Lemma 17. *For any* $\omega \in (0,1)$, $\delta \in (0,1)$, *and* $\epsilon > 0$, *after running LSE*$_{imp}$ *with* β_t *as in Lemma 1, with probability at least* $1-\delta$ *the returned solution is* ϵ*-accurate, that is*

$$\Pr\left\{\max_{\boldsymbol{x}\in D} \ell_h(\boldsymbol{x}) \leq \epsilon\right\} \geq 1-\delta.$$

Proof. From Lemma 15 and Lemma 16 it follows that, with probability at least $1-\delta$, (Q1) and (Q2) are stricter conditions than the following

$$\begin{aligned} \min(C_t(\boldsymbol{x})) + \epsilon &\geq h \\ \max(C_t(\boldsymbol{x})) - \epsilon &\leq h, \end{aligned}$$

which are identical to the ones used by LSE. Therefore, the solution of LSE$_{imp}$ achieves at least as high accuracy as the one provided for LSE by Lemma 7. □

Theorem 2 follows by combining Corollary 3 and Lemma 17.

A.3 Proof of Theorem 3

Lemma 18. *For any* $\boldsymbol{x} \in D$ *and* $t \geq 1$ *the ratio of* $\sigma_{fb[t]}(\boldsymbol{x})$ *to* $\sigma_{t-1}(\boldsymbol{x})$ *is bounded as follows*

$$\frac{\sigma_{fb[t]}(\boldsymbol{x})}{\sigma_{t-1}(\boldsymbol{x})} \leq \exp\left\{I(f; \boldsymbol{y}_{fb[t]+1:t-1} \mid \boldsymbol{y}_{1:fb[t]}\right\}.$$

Proof. See Lemma 1 in (Desautels et al., 2012). □

Lemma 19. *Assume that for all $t \geq 1$ the maximum conditional mutual information acquired by any set of measurements since the last feedback is bounded by a constant $C \geq 0$, i.e.*

$$\max_{A \subseteq D, |A| \leq B-1} I(f; \boldsymbol{y}_A \mid \boldsymbol{y}_{1:fb[t]}) \leq C. \tag{A.4}$$

Then, if $\eta_t = e^{2C} \beta_{fb[t]+1}$, the following holds with probability at least $1 - \delta$

$$f(\boldsymbol{x}) \in Q_t^b(\boldsymbol{x}),\ \forall \boldsymbol{x} \in D\ \forall t \geq 1.$$

Proof. From Lemma 18 and (A.4), it follows that for any $\boldsymbol{x} \in D$ and $t \geq 1$

$$\begin{aligned} &\frac{\sigma_{fb[t]}(\boldsymbol{x})}{\sigma_{t-1}(\boldsymbol{x})} \leq \exp\left\{I(f; \boldsymbol{y}_{fb[t]+1:t-1} \mid \boldsymbol{y}_{1:fb[t]}\right\} \leq e^C \\ \Rightarrow\ & e^C \sigma_{t-1}(\boldsymbol{x}) \geq \sigma_{fb[t]}(\boldsymbol{x}). \end{aligned}$$

Using this, the range of $Q_t^b(\boldsymbol{x})$ can be related to the range of $Q_{fb[t]+1}(\boldsymbol{x})$ as follows

$$2\eta_t^{1/2} \sigma_{t-1}(\boldsymbol{x}) = 2e^C \beta_{fb[t]+1}^{1/2} \sigma_{t-1}(\boldsymbol{x}) \geq 2\beta_{fb[t]+1}^{1/2} \sigma_{fb[t]}(\boldsymbol{x}).$$

Furthermore, $Q_t^b(\boldsymbol{x})$ and $Q_{fb[t]+1}(\boldsymbol{x})$ have the same midpoint, namely $\mu_{fb[t]}(\boldsymbol{x})$, therefore the above range inequality implies that

$$Q_t^b(\boldsymbol{x}) \supseteq Q_{fb[t]+1}(\boldsymbol{x}),\ \forall \boldsymbol{x} \in D\ \forall t \geq 1. \tag{A.5}$$

Finally, from Lemma 1 we have that

$$\begin{aligned} &\Pr\{f(\boldsymbol{x}) \in Q_{fb[t]+1}(\boldsymbol{x})\} \geq 1 - \delta,\ \forall \boldsymbol{x} \in D\ \forall t \geq 1 \\ \overset{(A.5)}{\Rightarrow}\ &\Pr\{f(\boldsymbol{x}) \in Q_t^b(\boldsymbol{x})\} \geq 1 - \delta,\ \forall \boldsymbol{x} \in D\ \forall t \geq 1. \end{aligned}$$

□

Corollary 4. *Given the assumptions of Lemma 19, the following holds with probability at least $1 - \delta$*

$$f(\boldsymbol{x}) \in C_t^b(\boldsymbol{x}),\ \forall \boldsymbol{x} \in D\ \forall t \geq 1.$$

Note that Corollary 4 is completely analogous to Corollary 1. Thus, the results of Lemmas Lemma 2–7 also hold for the case of LSE$_{\text{batch}}$, provided that η_t, as defined in Lemma 19, is used in place of β_t, which proves Theorem 3.

Bibliography

Azimi, Javad, Fern, Alan, and Fern, Xiaoli. Batch bayesian optimization via simulation matching. In *Neural Information Processing Systems (NIPS)*, 2010.

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

Bryan, Brent and Schneider, Jeff. Actively learning level-sets of composite functions. In *International Conference on Machine Learning (ICML)*, 2008.

Bryan, Brent, Schneider, Jeff, Nichol, Robert, Miller, Christopher, Genovese, Christopher, and Wasserman, Larry. Active learning for identifying function threshold boundaries. In *Neural Information Processing Systems (NIPS)*, 2005.

Cover, Thomas M. and Thomas, Joy A. *Elements of Information Theory*. Wiley-Interscience, 2006.

Dantu, Karthik and Sukhatme, Gaurav S. Detecting and tracking level sets of scalar fields using a robotic sensor network. In *International Conference on Robotics and Automation (ICRA)*, 2007.

Desautels, Thomas, Krause, Andreas, and Burdick, Joel. Parallelizing exploration-exploitation tradeoffs with gaussian process bandit optimization. In *International Conference on Machine Learning (ICML)*, 2012.

Galland, Frédéric, Rófrógier, Philippe, and Germain, Olivier. Synthetic aperture radar oil spill segmentation by stochastic complexity minimization. *IEEE Geoscience and Remote Sensing Letters (GRSL)*, 2004.

Garnett, Roman, Krishnamurthy, Yamuna, Xiong, Xuehan, Schneider, Jeff, and Mann, Richard. Bayesian optimal active search and surveying. In *International Conference on Machine Learning (ICML)*, 2012.

Hitz, Gregory, Pomerleau, François, Garneau, Marie-Eve, Pradalier, Cédric, Posch, Thomas, Pernthaler, Jakob, and Siegwart, Roland Y. Autonomous inland water monitoring: Design and application of a surface vessel. *Robotics & Automation Magazine (RAM)*, 2012.

Rahimi, Mohammad, Pon, Richard, Kaiser, William J., Sukhatme, Gaurav S., Estrin, Deborah, and Srivastava, Mani. Adaptive sampling for environmental robotics. In *International Conference on Robotics and Automation (ICRA)*, 2004.

Ramakrishnan, Naren, Bailey-Kellogg, Chris, Tadepalli, Satish, and Pandey, Varun N. Gaussian processes for active data mining of spatial aggregates. In *SIAM International Conference on Data Mining (SDM)*, 2005.

Rasmussen, Carl Edward and Williams, Christopher K. I. *Gaussian Processes for Machine Learning.* MIT Press, 2006.

Settles, Burr. *Active Learning (Synthesis Lectures on Artificial Intelligence and Machine Learning).* Morgan & Claypool Publishers, 2012.

Shalev-Shwartz, Shai. Online learning and online convex optimization. 2012.

Singh, Aarti, Nowak, Robert, and Ramanathan, Parmesh. Active learning for adaptive mobile sensing networks. In *International Conference on Information Processing in Sensor Networks (IPSN)*, 2006.

Srinivas, Niranjan, Krause, Andreas, Kakade, Sham, and Seeger, Matthias. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning (ICML)*, 2010.

Srinivasan, Sumana, Ramamritham, Krithi, and Kulkarni, Purushottam. Ace in the hole: Adaptive contour estimation using collaborating mobile sensors. In *International Conference on Information Processing in Sensor Networks (IPSN)*, 2008.

Zuluaga, Marcela, Krause, Andreas, Sergent, Guillaume, and Püschel, Markus. Active learning for multi-criterion optimization. In *International Conference on Machine Learning (ICML)*, 2013.