Benefits of Monotonicity in Safe Exploration with Gaussian Processes (Supplementary Material)

Arpan Losalka[1] **Jonathan Scarlett**[1,2,3]

[1]Department of Computer Science, National University of Singapore, Singapore
[2]Department of Mathematics, National University of Singapore, Singapore
[3]Institute of Data Science, National University of Singapore, Singapore

1 PROOFS

In this section, we present the proofs for Theorem 1 and Theorem 2.

1.1 PROOF OF THEOREM 1 (REGRET BOUND)

By Lemma 1, with probability at least $1-\delta$, the following holds for all $(s,\mathbf{x}) \in \mathcal{D}$ and $t \geq 1$:

$$|\mu_{t-1}(s,\mathbf{x}) - f(s,\mathbf{x})| \leq \beta_t \sigma_{t-1}(s,\mathbf{x}) \tag{1}$$

where $\mu_{t-1}(s_t,\mathbf{x})$ and $\sigma^2_{t-1}(s_t,\mathbf{x})$ are the mean and variance of the posterior distribution. As a special case of this fact, at each round $t \geq 1$, we have

$$\mu_{t-1}(s_t,\mathbf{x}_t) - f(s_t,\mathbf{x}_t) \leq \beta_t \sigma_{t-1}(s_t,\mathbf{x}_t). \tag{2}$$

Moreover, given the description of Algorithm 1, we have the following for all t:

$$\mu_{t-1}(s_t,\mathbf{x}_t) + \beta_t \sigma_{t-1}(s_t,\mathbf{x}_t) \geq h. \tag{3}$$

(Recall that the "safe everywhere" step setting $s=1$ for all $\mathbf{x}$ will never occur when the confidence bounds are valid, since we assume that at least one point is unsafe.)

Combining the above, we can conclude that for all $t \geq 1$, with probability at least $1-\delta$,

$$\begin{aligned} r_t &= h - f(s_t,\mathbf{x}_t) \\ &\leq \mu_{t-1}(s_t,\mathbf{x}_t) + \beta_t\sigma_{t-1}(s_t,\mathbf{x}_t) - f(s_t,\mathbf{x}_t) \qquad \text{(by (3))} \\ &\leq 2\beta_t\sigma_{t-1}(s_t,\mathbf{x}_t). \qquad \text{(by (2))} \end{aligned} \tag{4}$$

Hence, we have

$$R_T = \sum_{t=1}^{T} r_t \leq 2\beta_t \sum_{t=1}^{T} \sigma_{t-1}(s_t,\mathbf{x}_t). \tag{5}$$

Now, from Lemma 4 of [Chowdhury and Gopalan, 2017], $\sum_{t=1}^{T}\sigma_{t-1}(s_t,\mathbf{x}_t) = O\big(\sqrt{T\gamma_T}\big)$. Furthermore, $\beta_T \leq B + R\sqrt{2\left(\gamma_T + 1 + \ln(1/\delta)\right)}$ (since γ_t is monotonically increasing). Hence, with probability at least $1-\delta$,

$$R_T = O\left(B\sqrt{T\gamma_T} + \sqrt{T\gamma_T\left(\gamma_T + \ln(1/\delta)\right)}\right). \tag{6}$$

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

1.2 PROOF OF THEOREM 2 (IDENTIFICATION OF SAFE BOUNDARY)

Again using Lemma 4 in [Chowdhury and Gopalan, 2017], if $(s_1, \mathbf{x}_1), (s_2, \mathbf{x}_2), \ldots, (s_T, \mathbf{x}_T)$ are the points selected by Algorithm 1, then the sum of predictive standard deviations at these points can be bounded in terms of the maximum information gain as follows:

$$\sum_{t=1}^{T} \sigma_{t-1}(s_t, \mathbf{x}_t) \leq \sqrt{4(T+2)\gamma_T}. \tag{7}$$

Using the monotonicity of β_t, we deduce that for $T \geq 2$, we have

$$\begin{aligned} \frac{1}{T}\sum_{t=1}^{T} \beta_t \sigma_{t-1}(s_t, \mathbf{x}_t) &\leq \beta_T \sqrt{4\gamma_T/T + 8\gamma_T/T^2} \\ &\leq \beta_T \sqrt{8\gamma_T/T}. \end{aligned} \tag{8}$$

Now, as per Algorithm 1, for all $\mathbf{x} \in \mathcal{D}_{\mathcal{X}}$ and $t \leq T$, $s_t^{(\mathbf{x})}$ is one of the following:

- $s_t^{(\mathbf{x})} = 0$ if it holds that $\forall s : (s, \mathbf{x}) \in \mathcal{D}, \mathrm{UCB}_{t-1}(s, \mathbf{x}) > h$;
- $s_t^{(\mathbf{x})}$ is undefined if it holds that $\forall s : (s, \mathbf{x}) \in \mathcal{D}, \mathrm{UCB}_{t-1}(s, \mathbf{x}) < h$;
- in all other cases, $s_t^{(\mathbf{x})} = \max\{s : (s, \mathbf{x}) \in \mathcal{D}, \mathrm{UCB}_{t-1}(s, \mathbf{x}) = h\}$.

Thus, we can conclude that whenever $s_t^{(\mathbf{x})}$ is defined, it satisfies

$$\overline{s}_T^{(\mathbf{x})} \geq s_t^{(\mathbf{x})} \quad \forall t \leq T. \tag{9}$$

This is because $s_t^{(\mathbf{x})}$ is defined based on UCB_{t-1}, whereas $\overline{s}_T^{(\mathbf{x})} = \max\{s : (s, \mathbf{x}) \in \mathcal{D}, \min_{1 \leq t \leq T} \mathrm{UCB}_{t-1}(s, \mathbf{x}) \leq h\}$ considers the minimum of all UCB's across t to find the maximum s.

Since M-SAFEUCB selects the point with the largest $\sigma_{t-1}(s, \mathbf{x})$ from the candidate set S_t for $t \leq T$, we have the following whenever $s_t^{(\mathbf{x})}$ is defined:

$$\beta_t \sigma_{t-1}(s_t^{(\mathbf{x})}, \mathbf{x}) \leq \beta_t \sigma_{t-1}(s_t, \mathbf{x}_t) \; \forall \mathbf{x} \in \mathcal{D}_{\mathcal{X}}. \tag{10}$$

Next, note that $s_t^{(\mathbf{x})}$ is undefined for some $\mathbf{x} \in \mathcal{D}_{\mathcal{X}}$ only if at round t, $\mathrm{UCB}_{t-1}(s, \mathbf{x}) < h \; \forall s : (s, \mathbf{x}) \in \mathcal{D}$. In this case, $\overline{s}_T^{(\mathbf{x})} = 1$ by the definition. Therefore, for all $\mathbf{x} \in \mathcal{D}_{\mathcal{X}}$ where this occurs for some t, we have $(s, \mathbf{x}) \in \hat{L}_T$ for all $s \in [0, 1]$. Hence, as long as the confidence bounds are valid, we have $l_h(s, \mathbf{x}) = 0$ for such $(s, \mathbf{x})$.

For all $\mathbf{x} \in \mathcal{D}_{\mathcal{X}}$ not satisfying the conditions of the previous paragraph, we have for all $t \leq T$ that there exists $s \in [0, 1]$ such that $\mathrm{UCB}_t(s, \mathbf{x}) \geq h$, and accordingly, $s_t^{(\mathbf{x})}$ is well-defined. In this case, we bound the maximum deviation of $f(\overline{s}_T^{(\mathbf{x})}, \mathbf{x})$ from h as follows for any $t \leq T$:

$$\begin{aligned} \Delta(\overline{s}_T^{(\mathbf{x})}, \mathbf{x}) &:= h - f(\overline{s}_T^{(\mathbf{x})}, \mathbf{x}) && (11) \\ &\leq h - f(s_t^{(\mathbf{x})}, \mathbf{x}) \qquad \text{(by (9) and monotonicity of } f) && (12) \\ &\leq 2\beta_t \sigma_{t-1}(s_t^{(\mathbf{x})}, \mathbf{x}) \qquad \text{(similar to (4))} && (13) \\ &\leq 2\beta_t \sigma_{t-1}(s_t, \mathbf{x}_t), \qquad \text{(by (10))} && (14) \end{aligned}$$

provided that the confidence bounds are valid. Since this holds for all $t \leq T$, we can average both sides over $t \in \{1, \ldots, T\}$ to obtain

$$\begin{aligned} \Delta(\overline{s}_T^{(\mathbf{x})}, \mathbf{x}) &\leq \frac{2}{T}\sum_{t=1}^{T} \beta_t \sigma_{t-1}(s_t, \mathbf{x}_t) \\ &\leq 2\beta_T \sqrt{8\gamma_T/T}, \end{aligned} \tag{15}$$

where we made use of (8). Since $\hat{L}_T = \{(s, \mathbf{x}) \in \mathcal{D} : s \leq \overline{s}_T^{(\mathbf{x})}\}$, for any $(s, \mathbf{x}) \in \hat{L}_T$, we obtain $l_h(s, \mathbf{x}) = 0$ due to the validity of the confidence bounds. On the other hand, if $(s, \mathbf{x}) \notin \hat{L}_T$, there are two sub-cases to consider:

- If $(s, \mathbf{x}) \notin \hat{L}_T$ and $\mathrm{LCB}_T(s, \mathbf{x}) > h$, then

$$l_h(s, \mathbf{x}) = \max\{0, h - f(s, \mathbf{x})\} = 0. \tag{16}$$

- If $(s, \mathbf{x}) \notin \hat{L}_T$ and $\mathrm{LCB}_T(s, \mathbf{x}) < h < \mathrm{UCB}_T(s, \mathbf{x})$, then

$$\begin{aligned} l_h(s, \mathbf{x}) &= \max\{0, h - f(s, \mathbf{x})\} && (17)\\ &\leq \Delta(\overline{s}_T^{(\mathbf{x})}, \mathbf{x}) \qquad \text{(by } s \leq \overline{s}_T^{(\mathbf{x})} \text{ and monotonicity of } f) && (18)\\ &\leq 2\beta_T \sqrt{8\gamma_T / T}. \qquad \text{(by (15))} && (19) \end{aligned}$$

Therefore, setting $\epsilon = 2\beta_T \sqrt{8\gamma_T / T}$, we have the following guarantee for M-SAFEUCB's performance on the sub-level set estimation task:

$$\mathbb{P}\left\{ \max_{s, \mathbf{x} \in \mathcal{D}} l_h(s, \mathbf{x}) \leq \epsilon \right\} \geq 1 - \delta. \tag{20}$$

Substituting β_T into the above choice of ϵ completes the proof.

2 DETAILS OF EXPERIMENTS

Gaussian Process Model: For both the synthetic data and the inverted pendulum experiments, we use a Gaussian Process with Matérn$\frac{5}{2}$ kernel to model the unknown function. We use the *Trieste* toolbox for implementation [Picheny et al., 2023], and set the length scales and variance of the kernel to be trainable. The initial variance is set by randomly sampling two points in the domain known to be safe, and computing the variance with respect to the observed function values. A log-normal prior is used for both the variance and the length scales, with a standard deviation 1. The means for the length scales are set to 0.2, and that for the variance is 3. The function values returned are noiseless, while the Gaussian Process regression model assumes a low noise level of 10^{-5} for numerical stability. We use the Trieste library for our implementations [Picheny et al., 2023].

Synthetic functions: The domains of the functions f_{syn_1} and f_{syn_2} are set to $s \in [0, 1]$ and $x \in [0, 2]$. For running the algorithms, the domain is discretised into a grid with 200 linearly spaced points in each dimension. The optimisation is run for 100 iterations for each algorithm. Each experiment is repeated 5 times, and the mean values along with the standard deviations (via error bars) are shown.

For the function f_{syn_3}, the algorithms are run for 100 iterations, with the domain discretised into a grid with 75 linearly spaced points in each dimension. The experiments are repeated 5 times, and the mean values and standard deviations (via error bars) of the average cumulative regret are shown.

Inverted Pendulum: For this experiment, we allow the initial angle of the pendulum (denoted by x) to lie in $[-2\pi + \pi/36, -\pi/36]$ (where angle 0 denotes the upright position), while the applied torque $s \in [0, 1]$. The angle θ becomes positive after the pendulum crosses the upright position. We modify the reward function $f(s, x)$ as follows:

$$f_n(s, x) = \begin{cases} -\theta_n^2(s, x) - \frac{\dot{\theta}_n^2(s,x)}{10} - \frac{s^2}{1000}, & \text{if } \theta_n \leq 0 \\ \dot{\theta}_{up}(s, x) & \text{if } \theta_n(s, x) > 0, \end{cases} \tag{21}$$

$$f(s, x) = \max_{n \leq 100} f_n(s, x), \tag{22}$$

where $\theta_n(s, x)$ and $\dot{\theta}_n(s, x)$ denote the angle and angular velocity of the pendulum at the n^{th} time step, and $\dot{\theta}_{up}(s, x)$ denotes the angular velocity of the pendulum when it crosses the upright position, starting with an initial angle and torque of x and s respectively. Note that the time step n (for simulating the motion of the pendulum) is different from the time step t (denoting the optimisation iteration).

The safety threshold is set to $f(s, x) = 0$, which can only happen when both $\theta_n(s, x)$ and $\dot{\theta}_n(s, x)$ are 0 (since s is always 0 beyond the initial time step) for some $n \leq 100$. Thus, the safety threshold denotes the condition that the pendulum is in the upright position with a zero angular velocity, resulting in the sustenance of the upright position until the end of the episode, i.e., $n = 100$.

The initial angular velocity is always set to 0, so that our assumption that $s = 0$ is a safe action is satisfied. This is because the pendulum can never swing to the upright position starting from the range of initial positions specified, unless a torque is applied. Furthermore, the initial torque is assumed to be magnified by a factor of 20 when computing the resulting motion, resulting in the possibility of unsafe actions (torque applied, s) corresponding to a large fraction of starting positions (initial angle, x).

Similar to the experiments with synthetic data, the input domain is discretised into 200 linearly spaced points along each dimension, and the results of running the three algorithms 5 times are presented in Figure 2.

Algorithm Details and Discussion: For SAFEOPT, we use the version with the Lipschitz constant L as proposed in the original paper [Sui et al., 2015]. We approximate L by calculating the gradients for a finely discretised grid of points in the input domain in each case, and take the maximum among their magnitudes. Note that for using SAFEOPT in practice, L needs to be tuned alongside β_t as a hyperparameter. We consider the "best case" here for SAFEOPT, where a close approximation of the original Lipschitz constant for the unknown function is known to the algorithm. For the version of the algorithm using an underestimate of L in the experiments, we reduce the estimated L by a factor of 2 to 5. Further, we use the techniques discussed in Section 4 of [Berkenkamp et al., 2017] to reduce the computation cost of SAFEOPT. Despite these optimisations, we found that SAFEOPT can incur more than ten times the computation cost of M-SAFEUCB in our experiments, and this difference increases with increasing input dimension.

As discussed in Section 4 of [Sui et al., 2015], we solely use the confidence intervals for guaranteeing safety, and only use the Lipschitz constant for finding potential expanders. It is important to note here that using the Lipschitz constant for determining the safe set S_t further increases the dependence of SAFEOPT on the value of L, and can lead to degradation in performance due to over-cautiousness when L is overestimated. Thus, we avoid this version of the algorithm in our experiments. We also investigated the modified SAFEOPT algorithm suggested in [Berkenkamp et al., 2017] that avoids the dependence on L altogether, but we found it to be substantially more time consuming to run.

We also note here that in the version of SAFEOPT used in our experiments as described above, overestimating L essentially makes the algorithm behave very similar to M-SAFEUCB, since the number of points included in the set of potential expanders is small (or even zero) due to over-cautiousness, while the set of maximisers remains unaffected since S_t is determined only using the confidence intervals of the GP. This leads to wasteful computation compared to M-SAFEUCB leading to a greatly increased running time for obtaining a similar performance, thus also showcasing the benefits of using M-SAFEUCB over SAFEOPT for the problem setup under consideration.

For the PREDVAR algorithm, we consider the variance of all points in the domain with $s = 0$ (since these are known to be safe), as well as the points that can be guaranteed to be safe based on UCB_{t-1} at time step t, and choose the one with the highest variance.

We also note that M-SAFEUCB is similar in spirit to the (SAFEUCB) baseline [Sui et al., 2015], which simply maximises the UCB among all points that are known (with high probability) to be safe. However, doing this naively would lead to focusing on a small region of the $\mathbf{x}$ space and ignoring the rest. M-SAFEUCB overcomes this by using the maximum-variance rule.

3 FURTHER DISCUSSION

3.1 DISCUSSION ON $\mathcal{D}_\mathcal{X}$ DEPENDENCE

To get some intuition on why a linear dependence on the domain size may arise for algorithms such as SAFEOPT (as discussed in Section 4), consider the function shown in Figure 1. Once the function reaches $h - 2\epsilon$, it may become very difficult to use the confidence bounds and Lipschitz constants (as SAFEOPT uses) to determine whether it is still safe to move further to the right. One can imagine that an algorithm ends up sampling every x (or at least most x) even if $[0, 1]$ is discretised rather finely, particularly if the Lipschitz constant is over-estimated.

On the other hand, we highlight some potential weaknesses of SAFEOPT via two perspectives as follows:

(i) If the domain is quantised very finely, then one should only expect a number of samples depending on $\frac{L}{\epsilon^2}$, rather than $\frac{|\mathcal{D}_\mathcal{X}|}{\epsilon^2}$. This is because once a given point with $f(x) = h - 2\epsilon$ has its function value known accurately (say, to within 0.5ϵ), one should be able to certify the entire surrounding region of width $O(1/L)$ as safe, rather than only the next point to the right.

(ii) One can attain a guarantee with T having $\frac{\mathcal{D}_\mathcal{X}}{\epsilon^2}$ or even $\frac{L}{\epsilon^2}$ dependence (up to logarithmic factors) using a fairly trivial

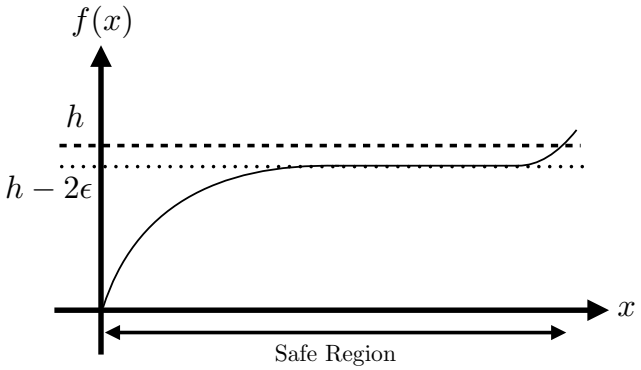

Figure 1: Example of a 1D function where expanding the known safe set (i.e., the points with $f(x) \leq h$) may be slow.

algorithm: Repeatedly sample all (known) safe points until their function values are known to within 0.5ϵ using basic concentration bounds, then expand the safe set using the Lipschitz constant, then return to repeated sampling (only for points not yet sampled), and so on. (Logarithmic terms would then arise from applying the union bound.) The resulting guarantee would even further improve SAFEOPT's guarantee due to omitting $\beta_T \gamma_T$ on the left-hand side.

Despite these limitations, we note that SAFEOPT has been an important and highly influential algorithm since its introduction, and the above discussion is only meant to highlight that its theoretical guarantees, while valuable, may leave significant room for improvement in certain scenarios.

3.2 COMPUTATIONAL CONSIDERATIONS

As stated, Algorithm 1 involves an explicit loop over all $\mathbf{x} \in \mathcal{D}_{\mathcal{X}}$. This is feasible when the domain size is small, and we adopted it in our experiments. However, such an approach may become infeasible for large or continuous domains. In such cases, one may need to rely on approximations or alternative methods, some of which we briefly discuss here.

First, if the domain is continuous, then one could rely on any *constrained black-box (non-convex) optimisation* solver to minimise the posterior variance subject to the UCB being at most h. For commonly-used kernels, the posterior variance and UCB are differentiable, which can facilitate this procedure. Moreover, to handle the possibility of points with $s = 0$ being selected, a second constrained black-box search could be performed over all $(0, \mathbf{x})$ subject to the UCB being *at least* h. The final selected point would then be the higher-variance one among the two points identified.

If no suitable black-box solver is available, or if the domain is discrete but large, then a simple practical alternative is as follows. Instead of performing a full optimisation of the acquisition function, one can randomly select a moderate number of $\mathbf{x}$ points at random (e.g., 500 or 1000) and only optimise over those. Due to the randomness, $\mathbf{x}$'s throughout the entire domain will then be considered regularly with high probability. Moreover, the efficiency could potentially be improved by ruling out certain regions early (e.g., when $s = 1$ is known to be safe). Note, however, that we do not claim any theoretical guarantees under these variations of the algorithm.

3.3 DISCUSSION OF Amani et al. [2021]

As we discussed in Section 1, the approach of [Amani et al., 2021] is based on first expanding the safe set using sufficiently many samples within an initial seed set. To highlight a limitation of this approach for certain kernels with infinite-dimensional feature spaces, consider the Matérn kernel, and suppose that the initial seed set includes a large fraction of the domain, but the function value is zero within that entire set. Since compactly supported "bump" functions are in the Matérn class [Bull, 2011], the function may contain both positive and negative bumps outside the seed set, some of which are safe and some of which are not. (Here we only assume that $f(\cdot) = 0$ is safe.) Since the function is zero within the seed set, there is no way that its samples can distinguish between these two cases.

In contrast, for finite-dimensional feature spaces (e.g., the linear or polynomial) even samples within a small seed set can indeed be sufficient to accurately learn the entire function. Finally, for the infinite-dimensional case with very rapidly decaying eigenvalues (e.g., SE kernel), the situation is somewhere in between the preceding examples; in particular,

compactly supported functions are not in the RKHS. In such scenarios, the approach of [Amani et al., 2021] may be feasible, though the precise details become somewhat complicated; certain results for infinite-dimensional settings are given in [Amani et al., 2021] accordingly.

References

Sanae Amani, Mahnoosh Alizadeh, and Christos Thrampoulidis. Regret bounds for safe Gaussian process bandit optimization. In *IEEE International Symposium on Information Theory (ISIT)*, pages 527–532, 2021.

Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. *Advances in Neural Information Processing Systems*, 30, 2017.

Adam D Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12 (Oct.):2879–2904, 2011.

Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. In *International Conference on Machine Learning*, pages 844–853. PMLR, 2017.

Victor Picheny, Joel Berkeley, Henry B. Moss, Hrvoje Stojic, Uri Granta, Sebastian W. Ober, Artem Artemev, Khurram Ghani, Alexander Goodall, Andrei Paleyes, Sattar Vakili, Sergio Pascual-Diaz, Stratis Markou, Jixiang Qing, Nasrulloh R. B. S Loka, and Ivo Couckuyt. Trieste: Efficiently exploring the depths of black-box functions with tensorflow, 2023. URL `https://arxiv.org/abs/2302.08436`.

Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with Gaussian processes. In *International Conference on Machine Learning*, pages 997–1005. PMLR, 2015.