Exploiting Strategy-Space Diversity for Batch Bayesian Optimisation

Sunil Gupta[1] **Alistair Shilton**[1] **Santu Rana** **Svetha Venkatesh**
{sunil.gupta, alistair.shilton, santu.rana, svetha.venkatesh}@deakin.edu.au
Deakin University, Geelong, Australia

Abstract

This paper proposes a novel approach to batch Bayesian optimisation using a multi-objective optimisation framework with exploitation and exploration forming two objectives. The key advantage of this approach is that it uses a suite of strategies to balance exploration and exploitation and thus can efficiently handle the optimisation of a variety of functions with small to large number of local extrema. Another advantage is that it automatically determines the batch size within a specified budget avoiding unnecessary function evaluations. Theoretical analysis shows that the regret not only reduces sub-linearly but also by an additional reduction factor determined by the batch size. We demonstrate the efficiency of our algorithm by optimising a variety of benchmark functions, performing hyperparameter tuning of support vector regression and classification, and finally heat treatment process of an Al-Sc alloy. Comparisons with recent baseline algorithms confirm the usefulness of our algorithm.

1 Introduction

Bayesian optimisation is a sample-efficient approach to optimise expensive black-box functions. It has found enormous applications in experimental design applications where the goal is to achieve a target by setting a number of control variables - e.g. synthetic gene design (González et al., 2014), optimisation of microalgae metabolism (Ulmasov et al., 2016), sensor set selection (Garnett et al., 2010) or hyperparameter tuning (Snoek et al., 2012). It typically models a black-box function using a Gaussian process (GP) (Williams and Rasmussen, 2006) that expresses a "belief" over all possible smooth objective functions through a prior distribution. The prior is updated to derive a posterior distribution as data is observed. The search is guided by a surrogate (acquisition) function derived from the GP posterior. Its global optimum can be found because, unlike the expensive (to evaluate) objective function, it can be cheaply evaluated over the search space. Examples of acquisition function include Expected Improvement (EI) (Jones et al., 1998), GP-UCB (Srinivas et al., 2012) and Predictive Entropy Search (Hernández-Lobato et al., 2014).

To efficiently search, the GP posterior is used to make the next recommendation for function evaluation, balancing two conflicting needs at iteration t, namely the need to "explore" regions of high epistemic uncertainty in the objective function (σ_t), and the need to "exploit" regions where the mean of the objective function (μ_t) is optimum. Assuming a maxima problem, this trade-off is balanced through a weighted combination ($\mu_t + \sqrt{\beta_t}\sigma_t$), where β_t is the *trade-off* factor. Higher values of β_t move the *strategy* towards exploration, whilst lower values favor exploitation. Thus a raft of alternate strategies can be generated by varying β_t. Recently, an iteration dependent β_t scheme called GP-UCB with a statistical bound on regret has been formulated (Srinivas et al., 2012). Similar progress has also been made for EI (Bull, 2011).

In many real-world cases where parallel resources are available it may be possible to simultaneously evaluate the function at multiple settings, leading to considerable saving in function optimisation time. In such situations it is often useful to implement batch Bayesian optimisation, which recommends multiple evaluation points per iteration. Previous work has focused on identifying multiple extrema of the acquisition function as the recommendations in an iteration. Most methods select the (top) extremum as the first point in the batch, and then use diverse methods to derive the remaining extrema. For example, González et al.

[1]Contributed Equally

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84.

(2016) use a local penalisation technique to suppress each extrema after detection. Other methods update the GP with the extremum found, causing a "dip" in the acquisition function at this point and allowing other lower extrema to gain prominence. Examples include "Kriging believer" (Ginsbourger et al., 2010) and methods using "fake" or "hallucinated" observations through prediction (Azimi et al., 2012; Desautels et al., 2014). Two other efforts that differ slightly are Nguyen et al. (2016), which uses a non-parametric Bayesian method to derive the extrema without peak suppression, and Contal et al. (2013), which uses pure exploration to derive all but the first extremum.

A common thread in the existing work is the construction of an acquisition function for the sequential BO, followed by the location of all the peaks as recommendations for the batch. The problem is that all the recommendations are made by fixing a single trade-off factor β_t at each iteration t. It is our intuition that a richer approach would be to select each point of the recommended batch through alternate trade-off strategies, i.e. by using a multitude of β_t values, thus combining a plethora of exploration and exploitation strategies. The advantage of this approach is that it hedges its bets, enabling it to handle a multitude of as yet unseen situations. For example, if the function has a large number of local extrema, then a judicious approach is to tilt the strategy towards exploration - as demonstrated in Bull (2011). In other situations, a balanced approach is preferable.

The focus of this paper is the construction of a new batch Bayesian optimisation method wherein the multiple recommendations are derived using a mixture of trade-off strategies. For the first time we formulate a multi-objective optimisation for μ_t and σ_t. The Pareto front yields a set of possible recommendations, each corresponding to a solution for a particular strategy (β_t). Thus the recommendations span a set of alternate strategies across the exploration-exploitation spectrum. We rigorously analyse the convergence of this approach. We derive batch cumulative and total cumulative regret bounds and show that their growth is sub-linear, implying convergence. We also show that the regret bounds reduce by an extra factor related to the batch size. An advantage of our method is that it automatically selects the batch size within a specified budget avoiding unpromising explorations. We formulate an algorithm to implement this approach and demonstrate the efficiency of our algorithm through 3 optimisation benchmark functions, hyperparameter tuning for machine learning using real-world data (Abalone and QSAR biodegradation datasets), and finally through a real world task of optimising heat treatment process of an Al-Sc alloy to maximise the strength. We compare the performance of our method with two recent state-of-the-art multi-recommendation baselines (Batch GP-UCB (Desautels et al., 2014) and GP-UCB-PE (Contal et al., 2013)). Our main contributions are:

- A novel approach to batch Bayesian optimisation by formulating it in a multi-objective optimisation framework, where exploitation (μ_t) and exploration (σ_t) form the two objectives. This allows us to use a raft of exploitation/exploration strategies;
- Theoretical analysis of our proposed method showing that regret reduces sub-linearly with number of batches and additionally by a factor equal to average batch size;
- Evaluation of our method using three benchmark functions, hyperparameter tuning for SVM using two real-world datasets, and finally a real-world task of optimising the heat treatment process of an Al-Sc alloy to maximise the strength.

2 Problem Statement and Background

We wish to maximise a function $f : \mathbb{X} \to \mathbb{R}$ to find $x^* = \text{argmax}_{x\in\mathbb{X}} f(x)$, where $\mathbb{X} \subseteq \mathbb{R}^d$ is compact and convex. We are interested in the case where f is expensive to evaluate - say for example the outcome of a physical experiment or parameter tuning in machine learning - but for which we may, at minimal additional cost, evaluate multiple points in a single "batch".

It is assumed that the function f can be modeled using a Gaussian process (Williams and Rasmussen, 2006), i.e. $f \sim \text{GP}(\mu, k)$ is a draw from a Gaussian process characterised by a mean function $\mu : \mathbb{X} \to \mathbb{R}$ and a covariance function $k : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$, where without loss of generality it is assumed that $\mu(x) = 0$. Given observations $\mathcal{D}_t = \{(x_i, y_i) : y_i = f(x_i) + \epsilon_i, i = 1, 2, \ldots, t\}$ where $\epsilon_i \in \mathcal{N}(0, \nu^2)$ we have the posterior $f(x)|\mathcal{D}_t \sim \mathcal{N}(\mu_t(x), \sigma_t^2(x))$ with mean $\mu_t(x)$ and variance $\sigma_t^2(x)$ as:

$$\begin{aligned} \mu_t(x) &= \mathbf{k}_t^{\mathrm{T}}(x)\left(\mathbf{K}_t + \nu^2\mathbf{I}\right)^{-1}\mathbf{y}_t \\ \sigma_t^2(x) &= k(x,x) - \mathbf{k}_t^{\mathrm{T}}(x)\left(\mathbf{K}_t + \nu^2\mathbf{I}\right)^{-1}\mathbf{k}_t(x) \end{aligned} \tag{1}$$

where $\mathbf{y}_t = [y_i]_{(-,y_i)\in\mathcal{D}_t}$, $\mathbf{k}_t(x) = [k(x_i, x)]_{(x_i,-)\in\mathcal{D}_t}$ and $\mathbf{K}_t = [k(x_i, x_j)]_{(x_i,-),(x_j,-)\in\mathcal{D}_t}$ (we write $(x, -) \in \mathcal{D}_t$ if $\exists y \in \mathbb{R} : (x, y) \in \mathcal{D}_t$; and likewise $(-, y) \in \mathcal{D}_t$ if $\exists x \in \mathbb{X} : (x, y) \in \mathcal{D}_t$).

2.1 Bayesian Optimisation

Bayesian optimisation (Brochu et al., 2010) (BO) is an optimisation algorithm that aims to find the optimal solution to a function f in the fewest iterations (evaluations) possible. At every iteration t

Bayesian optimisation selects a sample x to maximise a (cheap) acquisition function $a_t : \mathbb{X} \to \mathbb{R}$ based on μ_{t-1} and σ_{t-1}. This point is then evaluated to obtain $y_t = f(x_t) + \epsilon_t$ and the GP model gets updated to incorporate this new observation, and the algorithm continues. Typical acquisition functions are Expected Improvement (EI) (Jones et al., 1998), GP-UCB (Srinivas et al., 2012) and Predictive Entropy Search (Hernández-Lobato et al., 2014). In the present paper we particularly use GP-UCB as it is amenable to theoretical convergence analysis.

2.2 Batch Bayesian Optimisation

Batch Bayesian optimisation (Contal et al., 2013; Desautels et al., 2014; Kathuria et al., 2016; Shah and Ghahramani, 2015) (batch BO) is an increasingly popular approach to the problem considered here. Batch BO operates like BO, except that at each iteration it provides a batch of recommendations $\mathcal{X}_t = (x_t^1, x_t^2, \dots x_t^{n_t})$ for evaluation - see algorithm 1. Typically the size of the batch n_t is kept fixed for every iteration, so $n_t = n$. While this may be appropriate in some situations it is not difficult to imagine many others where "padding" the batch to reach some arbitrary goal may be pointless or even counter-productive if the additional cost per recommendation in a batch is negligible but non-zero.

Algorithm 1 Generic Batch-Bayesian Optimisation

input $\mathcal{D}_0 := \{(x_i, y_i) | y_i = f(x_i) + \epsilon, i = 1, 2, \dots\}$.
1: **for** $t = 1, 2, \dots, T$ **do**
2: Select Batch $\mathcal{X}_t = (x_t^1, x_t^2, \dots, x_t^{n_t})$.
3: Perform Experiments $\mathcal{Y}_t = f(\mathcal{X}_t) + \epsilon$.
4: Update $\mathcal{D}_t := \mathcal{D}_{t-1} \cup \{(x_t^i, y_t^i) | i = 1, 2, \dots, n_t\}$.
5: **end for**

Two representative examples of batch BO are:

- GP-UCB-PE (GP-UCB pure exploration) (Contal et al., 2013): generates a single recommendation x_t^1 using the standard GP-UCB and $n-1$ recommendations $\{x_t^i\}_{i=2}^n$ using pure exploration by maximising $\sigma_{t-1}^i(x)$, where σ_{t-1}^i is the predictive variance calculated on $\mathcal{D}_{t-1}, x_t^1, x_t^2, \dots, x_t^{i-1}$, which may be calculated without $\mathcal{Y}_t$ as σ_t is independent of $\mathbf{y}_t$.

- GP-BUCB (batch GP-UCB) (Desautels et al., 2014): generates n recommendations at each iteration using a standard GP-UCB strategy by applying *hallucinated* observations $y_t^i = \mu_{t-1}(x_t^t)$ to update an intermediate GP model used to generate recommendations.

2.3 Regret

A popular measure of performance for BO algorithms is regret. Let $x^* = \text{argmax}_{x \in \mathbb{X}} f(x)$ be the optimal solution. The instantaneous regret for a single recommendation x_t is defined as $r_t = f(x^*) - f(x_t)$, and for $x_t^i \in \mathcal{X}_t$ we define $r_t^i = f(x^*) - f(x_t^i)$. For a batch $\mathcal{X}_t$ of size n_t the simple regret (Bubeck et al., 2009) is $r_t = \min_i r_t^i$. Subsequently the batch cumulative regret is defined as (Bubeck et al., 2012, 2009; Contal et al., 2013):

$$R_T^n = \textstyle\sum_{t \le T} r_t$$

which is written R_T in the non-batch case ($n_t = 1$). Following the standard practice we will demonstrate a sub-linear growth bound on R_T^n, which implies convergence $R_T^n/T \to 0$. As $\min_{t \le T}\{r_t\} \le R_T^n/T$ it follows that $\max_{i, t \le T}\{f(x_t^i)\} \to f(x^*)$ as $T \to \infty$.

Alternatively it may be appropriate to consider the total cumulative regret $R_{Tn} = \sum_{t \le T} \sum_i r_t^i$ if we wish to have low regret for all samples in all batches. Though we are primarily interested in R_T^n we provide regret bounds for our algorithm for both R_T^n and R_{Tn}.

3 Relaxed GP-UCB and the Exploration/Exploitation Tradeoff

The GP-UCB acquisition function (Srinivas et al., 2012) is:

$$a_t(x) = \begin{cases} \mu_{t-1}(x) + \sqrt{\beta_t}\sigma_{t-1}(x) & \text{if } \beta_t < \infty \\ \sigma_{t-1}(x) & \text{if } \beta_t = \infty \end{cases} \tag{2}$$

This represents a trade-off between two distinct objectives, namely exploitation of the known good regions of the system, as represented by the maximisation of the mean $\mu_t(\mathbf{x})$ of the GP model, and *exploration* of the unknown regions of the model, as represented by the maximisation of the variance $\sigma_t(\mathbf{x})$. The trade-off between these objectives is controlled by the parameter β_t - smaller β_t values will favour exploitation, while larger β_t will favour exploration. We define:

Definition 1 *A GP strategy τ is a rule for generating a sequence $\beta_1, \beta_2, \dots$, thereby also generating a sequence of acquisition functions $a_1, a_2, \dots$ via (2) and subsequently a sequence of points $x_1, x_2, \dots$ where $x_t = \text{argmax}_{x \in \mathbb{X}} a_t(x)$.*

In GP-UCB (Srinivas et al., 2012) β_t is defined as:

$$\beta_t = b_t(\pi_t) = 2\log\left(\tfrac{\eta_t}{\delta}\pi_t\right) \tag{3}$$

where:

$$\eta_t = \begin{cases} |\mathbb{X}| & \text{if } |\mathbb{X}| < \infty \\ 4\left(t^2 dbr\sqrt{\log\left(\frac{2da}{\delta}\right)}\right)^{2d} & \text{if } |\mathbb{X}| = \infty \end{cases} \tag{4}$$

where $|\mathbb{X}|$ denotes the cardinality of $\mathbb{X}$. The sequence $\pi_1, \pi_2, \ldots > 0$ is any sequence satisfying $\sum_{t=1}^{\infty} \pi_t^{-1} = 1$ such that β_t is strictly increasing - for example $\pi_t = \pi^2 t^2/6$ (as used in Srinivas et al. (2012)). In the case $|\mathbb{X}|$ is infinite it is assumed that $\mathbb{X} \subset [0, r]^d$ and that f satisfies $\Pr\{\sup_{\mathbf{x}\in\mathbb{X}} |\partial f/\partial x_j| > L\} \leq ae^{-(L/b)^2}$.

The popularity of GP-UCB arises in part from the fact that the regret R_T can be shown to grow at most sub-linearly (Srinivas et al., 2012). We note that the bound in Srinivas et al. (2012) still applies if we relax the constraint $\sum_{t=1}^{\infty} \pi_t^{-1} = 1$ and instead enforce $\sum_{t=1}^{\infty} \pi_t^{-1} = \chi \leq 1$. In this case the regret bound presented in theorems 1 and 2 in Srinivas et al. (2012) will hold with probability $1 - \chi\delta \geq 1 - \delta$.[2] Nor is it necessary that β_t be strictly increasing if we replace β_T with $\beta_{\omega_T} = \max_{t\leq T} \beta_t$ in the relevant bounds in Srinivas et al. (2012).[3] Define:

Definition 2 *A* (relaxed) GP-UCB strategy *is a GP strategy defined by* $\pi_1, \pi_2, \ldots > 0$ *via* $\beta_t = b_t(\pi_t)$ *(see (3)); where* $\sum_{t=1}^{\infty} \pi_t^{-1} = \chi \leq 1$.

Note that the regret bounds of Srinivas et al. (2012) are valid for all GP-UCB strategies. We call $\pi_t = \pi^2 t^2/6$ the standard GP-UCB strategy as it corresponds to theorems 1 and 2 in Srinivas et al. (2012). Other GP-UCB strategies include:

- For every $p > 1$ we can define a GP-UCB strategy ${}_p\tau$ via ${}_p\pi_t = \zeta(p)\, t^p$, where p controls the speed of increase of β_t and ζ is the Riemann zeta function. The standard GP-UCB strategy is then ${}_2\tau$.
- Given two GP-UCB strategies $\tilde{\tau}, \breve{\tau}$ with sequences $\tilde{\pi}_1, \tilde{\pi}_2, \ldots$ and $\breve{\pi}_1, \breve{\pi}_2, \ldots$ we can define a GP-UCB strategy τ via $\pi_t = 2\tilde{\pi}_{t/2}$ for even t, $\pi_t = 2\breve{\pi}_{(t+1)/2}$ elsewhere.
- Given a GP-UCB strategy $\tilde{\tau}$ with sequence $\tilde{\pi}_1, \tilde{\pi}_2, \ldots$ we can define a new GP-UCB strategy τ via $\pi_t = \tilde{\pi}_{t/2}$ for even t, $\tilde{\pi}_t = \infty$ elsewhere, interleaving GP-UCB and pure exploration.

etc. However it is not clear which strategy would be the best in any given situation, nor how one would go about exploring the space of possibilities.

Given the definition of GP-UCB strategies above a batch BO approach that suggests itself here is to use multiple distinct GP-UCB strategies to generate multiple recommendations. However we are once again faced with the problem of selecting which strategies to apply. Clearly we want strategies that will provide a range of *distinct* recommendations (diversity) or little advantage will be gained, but it is not clear how one can test a-priori which strategies will result in diverse recommendations. This motivates us to suggest an entirely different approach in the next section.

[2] This follows from a simple adjustment to Lemma 5.1 in Srinivas et al. (2012).

[3] This arises in Lemma 5.4 in Srinivas et al. (2012) by bounding β_t by β_{ω_T} rather than β_T.

4 Multi-Recommendation via Multi-Objective Optimisation

Multi-objective optimisation (Deb, 2001) is concerned with simultaneously optimising multiple objectives:

$$\max_{x\in\mathbb{X}} (g_1(x), g_2(x) \ldots) \tag{5}$$

Multi-objective optimisation finds a set $(x_1, x_2, \ldots, x_n)$ of Pareto-optimal solutions to (5), where x is Pareto-optimal if it is impossible to change x to increase any objective g_i without decreasing at least one other objective g_j. It can be shown (Zadeh, 1963) that any Pareto-optimal solution of (5) is in fact the solution to the scalarisation:

$$\max_{x\in\mathbb{X}} \tilde{g}(x) = \lambda_1 g_1(x) + \lambda_2 g_2(x) + \ldots \tag{6}$$

where $\lambda_1, \lambda_2, \ldots \geq 0$ are (unknown) scalarisation constants.

Consider the acquisition function $a_t(x)$ defined by (2). This is a tradeoff between two potentially conflicting objectives - exploration (maximisation of $\sigma_{t-1}(x)$) and exploitation (maximisation of $\mu_{t-1}(x)$) - *which is at heart a multi-objective optimisation problem.* Based on this observation our proposed method for multi-recommendation is to use multi-objective optimisation at each iteration t to generate a set of recommendations $(x_t^1, x_t^2, \ldots, x_t^{n_t})$ to maximise (in the Pareto sense) $(\mu_{t-1}(x), \sigma_{t-1}(x))$. All recommendations so generated correspond to some (unknown) scalarisation constant β_t^i, but without the need to directly define strategies τ^i and with multiple recommendations generated automatically. This is illustrated in Figure 1.

Our proposed algorithm is shown in Algorithm 2. There are three stages to each iteration:

1. Recommendation set generation: one recommendation x_t^{u} is generated using the standard GP-UCB strategy along with a set of additional, more exploratory recommendations that maximise $(\mu_{t-1}(x), \sigma_{t-1}(x))$ in the Pareto sense.
2. Pruning of recommendations: the recommendations produced at step 1 are pruned to ensure that no more than M recommendations are made (the maximum budget per batch).

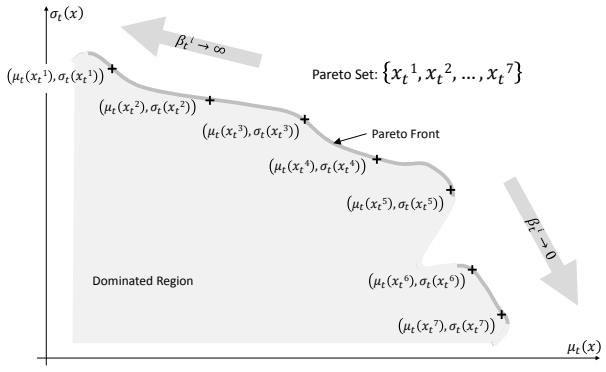

Figure 1: Using multi-objective optimisation to generate candidate recommendations. The range of the two objectives is the set $\{(\mu_t(x), \sigma_t(x))|x \in \mathbb{X}\}$, the Pareto *front* is the set of all possible Pareto optimal solutions in objective space, and the Pareto *set* is a representative set of x_t^i that dominate $\mathcal{X}_t$. The (implicit) scalarisation parameters β_t^i loosely correlate to the position of the points as shown.

3. Evaluation and model update: remaining n_t recommendations evaluated, GP models updated.

Details of these steps are given in the following sections. The key advantages of our algorithm are:

1. There is no need to painstakingly enumerate diverse, optimal GP strategies τ. The multi-objective optimiser will automatically obtain a range of Pareto-optimal recommendations each corresponding to some GP strategy.

2. The solution to multi-objective optimisation is a batch of recommendations making this a natural fit for batch BO.

3. The number of recommendations is tuned automatically by the multi-objective optimiser.

4.1 Recommendation Set Generation

With regard to recommendation set generation (stage 1) note that a standard GP-UCB recommendation x_t^{u} is always included in the recommendation batch as this will be necessary in the proof of Theorem 1 (the regret bound). The search space for multi-objective optimisation is restricted to the intersection of the relevant region (described in Contal et al. (2013)) and the set of recommendations that are more exploratory than the GP-UCB region:

$$\mathbb{X}_t^+ = \left\{ x \in \mathbb{X} \,\middle|\, \begin{array}{l} \mu_{t-1}(x) + 2\sqrt{{}_2\beta_{t+1}}\sigma_{t-1}(x) \geq y_t^\bullet \\ \sigma_{t-1}(x) \geq \sigma_{t-1}(x_t^{\mathrm{u}}) \end{array} \right\} \quad (7)$$

Algorithm 2 Proposed Batch Bayesian Optimisation

input $\mathcal{D}_0 := \{(x_i, y_i)|y_i = f(x_i) + \epsilon, i = 1, 2, \ldots\}$, maximum batch size M.

1: **for** $t = 1, 2, \ldots, T$ **do**
2: Let ${}_2\beta_t = 2\log(\eta_t \pi^2 t^2/6\delta)$ (η_t is given by (4)).
3: Select GP-UCB Recommendation:

$$x_t^{\mathrm{u}} = \mathrm{argmax}_{x\in\mathbb{X}} \left(\mu_{t-1}(x) + \sqrt{{}_2\beta_t}\sigma_{t-1}(x)\right)$$

4: Select Pareto-Optimal candidate set:

$$\mathcal{X}_t^{\mathrm{pf}} = \mathrm{argmax}_{x\in\mathbb{X}_t^+} \left(\mu_{t-1}(x), \sigma_{t-1}(x)\right)$$

where the relevant region $\mathbb{X}_t^+$ is given by 7.
5: If $|\mathcal{X}_t^{\mathrm{pf}}| > M-1$ then randomly sub-sample $\mathcal{X}_t^{\mathrm{pf}}$ to obtain $M-1$ recommendations. The result is denoted as $\mathcal{X}_t^{\mathrm{p}}$.
6: Construct Recommendation Batch as $\mathcal{X}_t = \mathcal{X}_t^{\mathrm{p}} \cup \{x_t^{\mathrm{u}}\}$ where $n_t = |\mathcal{X}_t|$.
7: Perform Experiments $y_t^i = f(x_t^i) + \epsilon \; \forall x_t^i \in \mathcal{X}_t$.
8: Update $\mathcal{D}_t := \mathcal{D}_{t-1} \cup \{(x_t^i, y_t^i)|i = 1, 2, \ldots, n_t\}$.
9: **end for**

where ${}_2\beta_t = 2\log(\eta_t \pi^2 t^2/6\delta)$ (η_t is given by (4) as per the standard GP-UCB strategy) and:

$$y_t^\bullet = \mathrm{argmax}\left(\mu_{t-1}(x) - \sqrt{{}_2\beta_t}\sigma_{t-1}(x)\right)$$

is the maximum of the lower confidence bound on f, which corresponds to the subset of $\mathbb{X}$ whose exploration is likely to have an impact on future recommendations (see Contal et al. (2013)). This approach is similar to GP-UCB-PE; however rather than restricting additional recommendation selection to recommendations that are *purely* exploratory we require only that the additional recommendations be more exploratory than the initial standard GP-UCB recommendation. This gives our algorithm more freedom to generate recommendations from a variety of strategies.

In our implementation we have used a genetic algorithm based multi-objective optimiser (Deb, 2001) (`gamultiobj` in Matlab) to generate the Pareto recommendation set $\mathcal{Y}_t^{\mathrm{pf}}$. In practice we found that similar results were obtained by replacing $\mathbb{X}_t^+$ with $\mathbb{X}$ (that is, removing all restrictions from the Pareto optimiser to allow for recommendations that are both more exploitative and more exploratory than the standard GP-UCB recommendation); however using the stricter definition allows us to bound both batch cumulative regret and total cumulative regret, so we have chosen to use this $\mathbb{X}_t^+$ unless otherwise stated.

4.2 Pruning the Recommendation Set

Give a pre-specified maximum batch size M, if more than $M-1$ recommendations are present in $\mathcal{X}_t^{\mathrm{pf}}$ then

the set is randomly sub-sampled to reduce this to $M-1$ recommendations, leaving at most M recommendations in total when the standard GP-UCB recommendation x_t^{u} is included. This steps has no impact on the regret bound presented in theorem 1.

5 Performance Bounds

In this section we present regret bounds for the proposed batch Bayesian optimisation algorithm. Our approach is comparable to Srinivas et al. (2012); Contal et al. (2013); Desautels et al. (2014) and represents a generalisation of the methods of Srinivas et al. (2012).

As previously noted in section 2.3 we define regret as per Contal et al. (2013) - specifically, for $x_t^i \in \mathcal{X}_t$ the instantaneous regret is $r_t^i = f(x^*) - f(x_t^i)$, for the batch $\mathcal{X}_t$ the simple regret is $r_t = \min_i r_t^i$, and the *batch cumulative regret* is:

$$R_T^n = \sum_{t \leq T} r_t$$

This assumes a scenario where one is interested in the best result from each batch and other samples in the batch help to explore the space and subsequently lead to faster convergence in subsequent batches. An alternative approach is to analyse the *total cumulative regret*:

$$R_{Tn} = \sum_{t \leq T} \sum_{t \leq n_t} r_t^i$$

which is useful in cases where we want to minimise regret on each sample in any batch.

As per Srinivas et al. (2012); Contal et al. (2013), and assuming the horizon T is unknown, a strategy has to be good for any number of iterations. We show that with high probability R_T^n (and also R_{Tn}) grows at most sub-linearly, and hence the minimum gap between the maximum of f and the best point evaluated so far converges to zero. Our main result is as follows:

Theorem 1 *Let $\delta \in]0,1[$. Assuming $k(x,x) \leq 1$ $\forall x \in \mathbb{X}$ and either $\mathbb{X}$ is either finite or $\mathbb{X} \subset [0,r]^d$ and f satisfies $\Pr\{\sup_{x \in \mathbb{X}} |\partial f / \partial x_j| > L\} \leq a e^{-(L/b)^2}$ then for all T:*

$$\begin{aligned} R_T^n &\leq \sqrt{\tfrac{T}{H_T} C_1 \, {}_2\beta_T \gamma_{A_T T}} + C_2 \\ R_{Tn} &\leq \sqrt{A_T T C_3 \, {}_2\beta_T \gamma_{A_T T}} + N_\infty C_2 \end{aligned} \tag{8}$$

holds with probability $\geq 1-\delta$, where $A_T = \frac{1}{T}\sum_{t \leq T} n_t$ and $H_T = \frac{T}{\sum_{t \leq T} \frac{1}{n_t}}$ are, respectively, the arithmetic and harmonic means of the batch sizes; $N_\infty = \max_{t \leq T} n_t$; $C_1 = 8/\log(1+\nu^{-2})$; $C_2 = 0$ if $|\mathbb{X}|$ is finite, $\pi^2/6$ otherwise; $C_3 = 72/\log(1+\nu^{-2})$; and $\gamma_{A_T T}$ is the max information gain obtainable from a sequence of length $A_T T$.

Proof: A complete proof is provided in the **supplementary material**. The proof proceeds as follows. In Lemma 1 we demonstrate that ordering in the (implicit) scalarisation constants β_t^i mimics the ordering in the variances $\sigma_t^i = \sigma_{t-1}(x_t^i)$ - that is, $\beta_t^j < \beta_t^k \Rightarrow \sigma_t^j \leq \sigma_t^k$ and $\sigma_t^j < \sigma_t^k \Rightarrow \beta_t^j \leq \beta_t^k$. As $\sigma_t^i \geq \sigma_{t-1}(x_t^{\mathrm{u}})$ by definition of $\mathbb{X}_t^+$, and assuming without loss of generality that $\sigma_t^1 \leq \sigma_t^2 \leq \ldots \leq \sigma_t^{n_t}$, this implies that ${}_2\beta_t = \beta_t^1 \leq \beta_t^2 \leq \ldots \leq \beta_t^{n_t}$ (Lemma 2) - that is, the Pareto-optimal recommendations are all (implicitly) generated by more exploratory strategies than the standard GP-UCB recommendation.

Having established this basic framework the remainder of the proof is modelled on Contal et al. (2013); Srinivas et al. (2012), with some modifications to account for the fact that (a) the batch size varies with iteration t and (b) the recommendations in addition to the standard GP-UCB recommendation are generated by more exploratory strategies than standard GP-UCB but not necessarily purely exploratory ones. □

This theorem demonstrates that the regret converges sublinearly for our algorithm, as is required. Comparing with respective results in Srinivas et al. (2012) we also see that the bound on R_T^n is at least as strong as standard (non-batch) GP-UCB. More generally, assuming $T \gg A_T$, we see that the performance bound on our algorithm is tighter than the comparable bound on GP-UCB by a factor of $\sqrt{H_T}$ - that is, improvement is proportional to the square-root of the harmonic mean of the sequence of batch sizes.

6 Experiments

In this section we study the empirical performance of our proposed batch Bayesian method for optimising three benchmark functions, two hyperparameter tuning tasks and for optimising heat treatment of a Aluminum-Scandium (Al-Sc) alloy. We compare our method with other batch Bayesian optimisation methods, namely, GP-BUCB (Desautels et al., 2014) and GPUCB-PE (Contal et al., 2013). We set the maximum batch size to 5 meaning our method uses a batch-size between 1 to 5 inferred automatically. We show that our method outperforms the baselines when using the same number of total experiments. We also use GP-UCB (a single recommendation algorithm) as a reference baseline to assess the additional benefit of performing batch experiments.

For function modelling, we scale each input dimension to [0,1] and use Gaussian process with squared-exponential kernel having length scale 0.1. The optimisation is initialised with $d+1$ observations where d is input space dimension. All optimisation results are reported by averaging 10 random runs along with the standard errors. For each run, the optimisation is

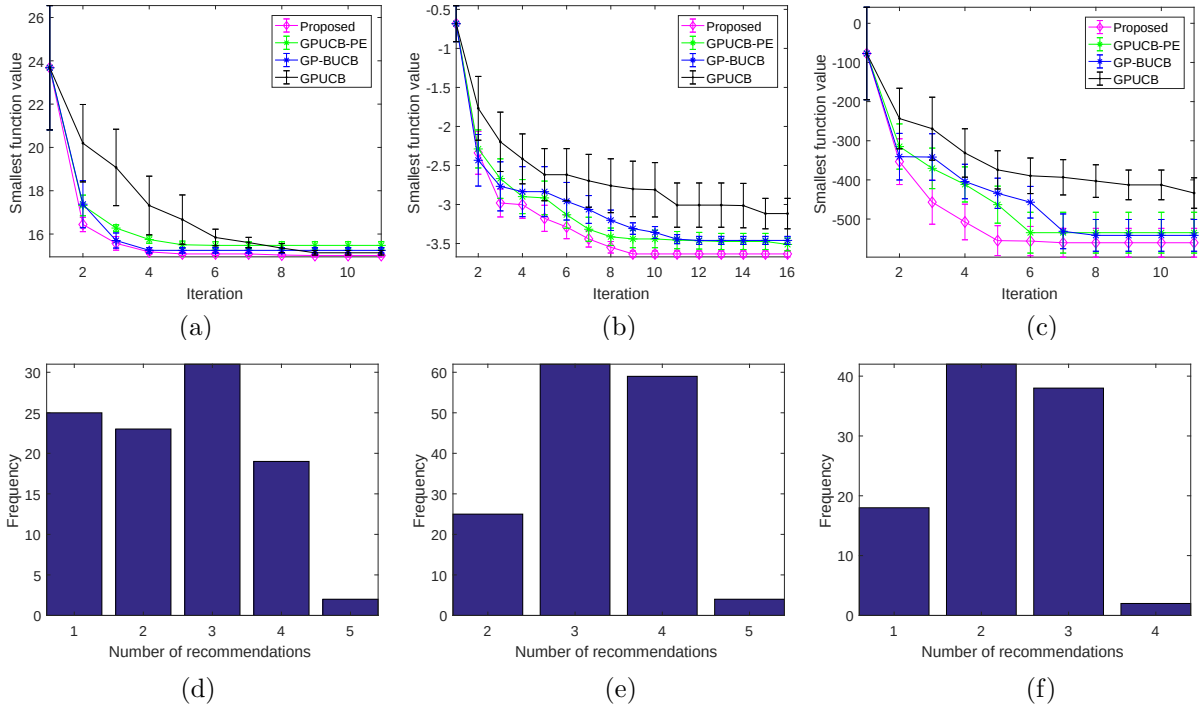

Figure 2: Results for benchmark functions. Optimisation performance shown in top row, distribution of number of recommendations in bottom row. Functions are branin ((a),(d)), Hartmann3 ((b),(e)) and Eggholder ((c),(f)). Results shown are after aggregation over 10 random initialisations.

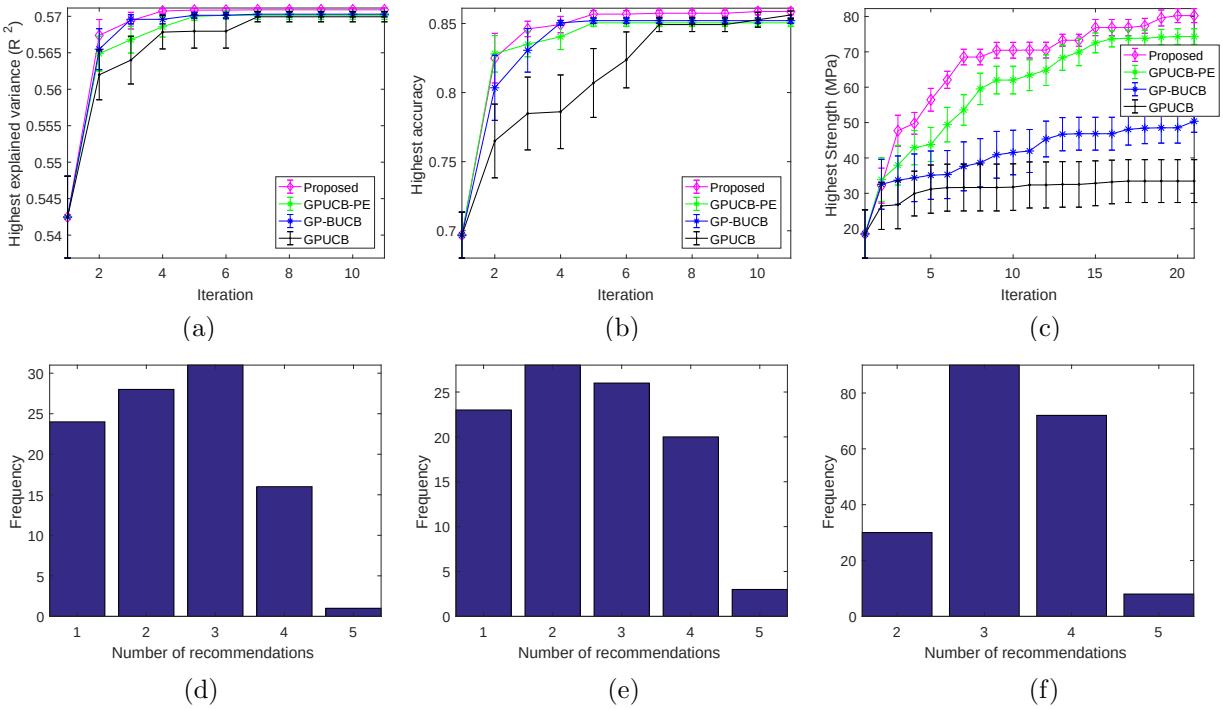

Figure 3: Experiments with hyperparameter tuning and alloy heat treatment tasks: (a) and (d) show the hyperparameter tuning results of a support vector regression model, (b) and (e) show the hyperparameter tuning results of a support vector classification model, and (c) and (f) show the heat treatment optimisation of an Al-Sc alloy. The first row depicts the optimisation performance and the second row depicts the distribution of the number of recommendations in a batch. Results shown are after aggregation over 10 random initialisations.

performed up to $5 \times d$ iterations. For optimisation of acquisition function, we use DIRECT algorithm (Jones et al., 1993).

Optimisation of Benchmark Functions

To study the convergence behaviour and optimisation efficiency of our method we use three benchmark functions commonly used to evaluate global optimisation methods, namely the *branin-Hoo*, *Hartmann3*, and *Eggholder* functions. These functions cover a variety of scenarios typically faced in global optimisation. For example, the branin function has three different global minima, the Hartmann3 has 4 local minima, while the Eggholder has plenty of local minima.

Figure 2 shows the comparison between the proposed method and the baselines. From the top row in the Figure (see subplots (a)-(c)) it can be clearly seen that the optimisation efficiency of our proposed method is either superior or comparable to the baseline multi-recommendation methods. We note that branin function is a relatively simpler function to optimise when compared to Hartmann3 and Eggholder due to the complexity brought by the number of local minima. Since the three functions are quite different in nature, the consistent superior performance of our method indicates that it can find a good balance between exploration and exploitation to be useful in a variety of situations. A further advantage of using our method is that the experimenter has to only specify the maximum batch size and the number of recommendations in a batch can be inferred automatically. The distribution of the number of recommendations used by our method is shown in the bottom row (see subplots (d)-(f)) of Figure 2. In the *supplementary*, we have also included results comparing the two variants of our method using $\mathbb{X}$ and $\mathbb{X}_t^+$; and comparisons between our method and multipoint-EI (Ginsbourger et al., 2010).

Machine Learning Hyperparameter Tuning

For hyperparameter tuning experiments, we use two machine learning models: support vector regression (SVR) and support vector classification (SVC). We use these models in combination to a radial basis kernel (RBF) giving rise to two hyperparameters. The first is the cost parameter of either ν-SVR model or C-SVC, and the second is the kernel width parameter γ of the RBF kernel. We optimise both these hyperparameters in the range of $[10^{-1}, 10^3]$. The functions are learnt in the exponent space $[-1, 3]$. For the regression task, we use the Abalone dataset (Nash et al., 1994) and for the classification task, we use QSAR biodegradation dataset (Mansouri et al., 2013) - see *http://archive.ics.uci.edu/ml/*. The data was split into 70:30 ratio for training and validation purposes. We used LibSVM Matlab library for our testbed. All the algorithms were run for 10 random runs differing in initial 3 points and the average optimisation performance is reported. Figure 3 (a) and (b) show the results of the optimisation comparing our proposed method with the baselines; and (d) and (e) the distribution of the number of recommendations. For both tasks, our proposed method slightly outperforms both baselines.

Heat Treatment Optimisation for Al-Sc Alloy

Heat treatment is a combination of heating and cooling processes applied to an alloy to achieve desired properties such as strength, hot workability etc. During the heat treatment process, a cast alloy is sequentially heated at various temperatures for different time durations. The temperatures and the exposition times significantly affect the strength of the treated alloy. Thus it is imperative to optimise this process. Since each experiment is expensive in time and many ovens may be available, we perform batch optimisation.

In our experiment, we use two-stage heat treatment process including 4 variables: Temperature at each stage and the two exposition times. To perform the experiments, we use the industrial precipitation model of Al-Sc alloy, known as Kampmann-Wagner model (Wagner et al., 1991). The search ranges are limited to [100°C, 300°C] and [1080 sec, 10800 sec] for temperature and time respectively. Figure 3 (c) and (f) show the optimisation performance of our proposed algorithm. The number of recommendations used by our algorithm is mostly 3 or 4. As seen from the figure, the use of our proposed algorithm has achieved significant improvement in alloy strength outperforming other baselines.

7 Conclusion

We presented a novel algorithm to perform batch Bayesian optimisation in a multiobjective approach for balancing exploitation/exploration requirements. The batch of recommendations is constructed using the Pareto set obtained from the multi-objective optimisation. The advantage of our approach is that it can utilise a set of alternate strategies across the exploration-exploitation spectrum. We theoretically analysed our method and showed that the regret bound is reduced by a factor that is equal to the harmonic mean of the used batch sizes. We demonstrated the efficiency of our algorithm by optimising several benchmark functions, hyperparameters tuning and alloy heat treatment optimisation.

Acknowledgements

This research was partially funded by the Australian Government through the Australian Research Council (ARC) and the Telstra-Deakin Centre of Excellence in Big Data and Machine Learning. Prof Venkatesh is the recipient of an ARC Australian Laureate Fellowship (FL170100006).

References

Javad Azimi, Ali Jalali, and Xiaoli Fern. Hybrid batch Bayesian optimization. In *Proceedings of the 29th International Conference on Machine Learning*, 2012.

Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with applications to active user modeling and heirarchical reinforcement learning. eprint arXiv:1012.2599, arXiv.org, December 2010.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *International conference on Algorithmic learning theory*, pages 23–37. Springer, 2009.

Sébastien Bubeck, Nicolo Cesa-Bianchi, and et. al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

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

Emile Contal, David Buffoni, Alexandre Robicquet, and Nicolas Vayatis. Parallel gaussian process optimization with upper confidence bound and pure exploration. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 225–240. Springer, 2013.

K. Deb. *Multi-Objective Optimization using Evoluationary Algorithms*. John Wiley & Sons, Chichester, UK, 2001.

Thomas Desautels, Andreas Krause, and Joel W. Burdick. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *Journal of Machine Learning Research*, 15(1):3873–3923, 2014.

Roman Garnett, Michael A Osborne, and Stephen J Roberts. Bayesian optimization for sensor set selection. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 209–219. ACM, 2010.

David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. Kriging is well-suited to parallelize optimization. In *Computational Intelligence in Expensive Optimization Problems*, pages 131–162. Springer, 2010.

Javier González, Joseph Longworth, David C James, and Neil D Lawrence. Bayesian optimization for synthetic gene design. *Workshop in Bayesian Optimization (in conjunction to NIPS 2014)*, 2014.

Javier González, Zhenwen Dai, Philipp Hennig, and Neil Lawrence. Batch Bayesian Optimization via Local Penalization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51, pages 648–657, 2016.

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

Donald R Jones, Cary D Perttunen, and Bruce E Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993. ISSN 1573-2878. doi: 10.1007/BF00941892.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

Tarun Kathuria, Amit Deshpande, and Pushmeet Kohli. Batched Gaussian process bandit optimization via determinantal point processes. In *Advances in Neural Information Processing Systems*, pages 4206–4214, 2016.

Kamel Mansouri, Tine Ringsted, Davide Ballabio, Roberto Todeschini, and Viviana Consonni. Quantitative structure–activity relationship models for ready biodegradability of chemicals. *Journal of chemical information and modeling*, 53(4):867–878, 2013.

Warwick J. Nash, Tracy L. Sellers, Simon R. Talbot, Andrew J. Cawthorn, and Wes B. Ford. The population biology of abalone (haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and islands of bass strait. *Sea Fisheries Division, Technical Report*, (48), 1994.

Vu Nguyen, Santu Rana, Sunil K Gupta, Cheng Li, and Svetha Venkatesh. Budgeted batch Bayesian optimization. In *International Conference on Data Mining*, pages 1107–1112. IEEE, 2016.

Amar Shah and Zoubin Ghahramani. Parallel predictive entropy search for batch global optimization of expensive objective functions. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3330–3338. Curran Associates, Inc., 2015.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Information-theoretic regret bounds for gaussian process optimization in the

bandit setting. *IEEE Transactions on Information Theory*, 58(5):3250–3265, May 2012.

Doniyor Ulmasov, Caroline Baroukh, Benoit Chachuat, Marc Peter Deisenroth, and Ruth Misener. Bayesian optimization with dimension scheduling: Application to biological systems. In *Computer Aided Chemical Engineering*, volume 38, pages 1051–1056. Elsevier, 2016.

Richard Wagner, Reinhard Kampmann, and Peter W. Voorhees. Homogeneous second-phase precipitation. *Materials science and technology*, 1991.

Christopher KI Williams and Carl Edward Rasmussen. Gaussian processes for machine learning. *the MIT Press*, 2(3):4, 2006.

Lofti Zadeh. Optimality and non-scalar-valued performance criteria. *IEEE transactions on Automatic Control*, 8(1):59–60, 1963.