Hybrid Batch Bayesian Optimization

Javad Azimi
azimi@eecs.oregonstate.edu
Oregon State University

Ali Jalali
alij@mail.utexas.edu
University of Texas at Austin

Xiaoli Fern
xfern@eecs.oregonstate.edu
Oregon State University

March 1, 2012

Abstract

Bayesian Optimization aims at optimizing an unknown non-convex/concave function that is costly to evaluate. We are interested in application scenarios where concurrent function evaluations are possible. Under such a setting, BO could choose to either sequentially evaluate the function, one input at a time and wait for the output of the function before making the next selection, or evaluate the function at a batch of multiple inputs at once. These two different settings are commonly referred to as the sequential and batch settings of Bayesian Optimization. In general, the sequential setting leads to better optimization performance as each function evaluation is selected with more information, whereas the batch setting has an advantage in terms of the total experimental time (the number of iterations). In this work, our goal is to combine the strength of both settings. Specifically, we systematically analyze Bayesian optimization using Gaussian process as the posterior estimator and provide a hybrid algorithm that, based on the current state, dynamically switches between a sequential policy and a batch policy with variable batch sizes. We provide theoretical justification for our algorithm and present experimental results on eight benchmark BO problems. The results show that our method achieves substantial speedup (up to 78%) compared to a pure sequential policy, without suffering any significant performance loss.

1 Introduction

Bayesian Optimization (BO) tries to optimize an unknown function $f(\cdot)$ by requesting a set of experiments when $f(\cdot)$ is costly to evaluate [8, 4]. In this work, we are interested in finding a point $x^* \in \mathcal{X}^d$ such that:

$$x^* = \arg\max_{x \in \mathcal{X}^d} f(x),$$

where $\mathcal{X}^d$ is our $d$-dimensional compact input space and $f(\cdot)$ is non-concave underlying function which has multiple local optima. The function $f(\cdot)$ might be the performance of a black box device characterized by input $x$. For example, in our motivating application we try to optimize the power output of nano-enhanced Microbial Fuel Cells (MFCs). MFCs [3] use micro-organisms to generate electricity. It has been shown that efficiency of generated electricity power significantly depends on the surface properties of the anode [12]. Our problem involves optimizing the surface properties of the anodes in order to maximize the output power. The goal is to develop an efficient BO algorithm for this application since running an experiment is very expensive and time consuming.

Focusing on the task of function maximization, each run of BO consists of two main steps: estimating the values of the unknown function $f(\cdot)$ via a probabilistic model such as Gaussian Process, and selecting the best next experiment(s) according to the probabilistic model via some selection criterion. The results of those experiment(s) will then added to update the probabilistic model and this cycle is repeated until we meet a stopping criterion.

Most of the proposed selection criteria in BO are sequential where only one experiment is selected at each iteration [11] [8] [14] [9]. The sequential policies usually perform very well in practice since they optimize the experiment selection at each iteration by using the maximum available information for each experiment. However, they are not time efficient in many applications where running an experiment takes a long time, and we have the capability to run multiple experiments in parallel. This motivates the batch algorithms in which more than one experiment is selected at each iteration.
Recently, [2] introduced a batch BO approach that selects a batch of k experiments at each iteration that approximates the behavior of a given sequential heuristic. [7] introduced a constant liar heuristic algorithm to select a batch of experiments based on the Expected Improvement (EI) policy. Specifically, after selecting an experiment by EI, the output of the selected point is set to a constant value. This experiment is then added to the prior and the procedure is repeated until k experiments are selected. Although these two batch algorithms [2,7] can speedup the experiment selection by a factor of k, their results show that batch selection in general performs worse than the sequential policy (EI), especially when the total number of experiments is small. This observation motivates us to introduce a Hybrid BO approach that dynamically alternates between sequential and batch selection to achieve improved time efficiency over sequential without degrading the optimization performance.

In this paper, we focus on a class of batch policies that is based on simulating a sequential policy and provide a systematic approach to analyze such batch BO policies. We analytically connect the mismatch between the BO’s probabilistic model and the underlying true function to the performance of the batch policy. We provide full characterization of simulated-based batch policies when the batch size is 2. For the purpose of illustration, consider a batch policy that selects 2 experiments. Obviously, the first experiment matches the sequential policy. The choice of the second experiment, however, will depend on what is the simulated the outcome of the first experiment. We show that the distance between the second experiment picked by a simulation-based batch policy (without the knowledge of the output of the first experiment) and the one picked by the sequential policy (with the knowledge of the output of the first experiment) is upper-bounded by a quantity that is proportional to the square root of the estimation error (of the outcome of the first experiment).

This analysis naturally gives rise to our hybrid batch/sequential algorithm. Our algorithm works as follows: At each step, given any sequential policy (EI in this paper), find the best next single experiment and estimate its possible outcome via BO’s probabilistic model (GP in this paper). Then, update the prior with that point and choose the next best single experiment and so on. We analytically show that this process can be continued until a certain stopping criterion is met. This stopping criterion measures how much a simulated experiment is going to bias our probabilistic model (mainly because of inaccuracy in estimation of the outcomes of the first experiment). If the bias is small, we continue to add more examples to our batch; and if it is large, we stop.

The proposed algorithm has the appealing property that it behaves more like a sequential policy in early stages when the number of observed experiments is small, and naturally transits to batch mode in later stages when more experiments are available. This is because the stopping criterion tends to be more stringent in early stages because the bias of the prior can be potentially large, forcing the algorithm to act sequentially. The beauty of this algorithm is that it evolves from a sequential algorithm to a batch algorithm in an optimal manner characterized by our theoretical results.

Experimental results show that the proposed algorithm can achieve up to 78% speedup over the sequential policy without degrading the performance even with a very small number of experiments. We also show that, by increasing the number of experiments, the speedup rate is increased significantly which is consistent with the theoretical results presented in the paper.

The paper is organized as follows. We introduce the Gaussian Process which is used as our model in Section 2. The proposed dynamic batch algorithm is described in Section 3. Section 4 presents the experimental results and the paper is concluded in Section 5.

2 Gaussian Process

A BO algorithm has two main ingredients: a probabilistic model for the unknown function, and, a selection criterion for choosing next best experiment(s) based on the model. We select Gaussian Process (GP) [13] as our probabilistic model and expected improvement (EI) [9] as our selection criterion. We study the properties of GP in this section and postpone the analysis of EI to the next section.

We use GP to build the posterior over the outcome values given our observation set \( O = \{x_1, y_1\} \), where, \( x_1 = \{x_1, x_2, \ldots, x_n\} \) is the set of inputs and \( y_1 = \{y_1, y_2, \ldots, y_n\} \) is the set of outcomes (of the experiment) such that \( y_j = f(x_j) \) and \( f(\cdot) \) is the underlying unknown function.

For a new input point \( x_i \), GP models the unknown output \( y_i = f(x_i) \) as a normal random variable \( y_i \sim \mathcal{N}(\mu_{x_i|O}, \sigma^2_{x_i|O}) \), with \( \mu_{x_i|O} = k(x_i, x_i)k(x_i, x_i)^{-1}y_i \) and \( \sigma^2_{x_i|O} = k(x_i, x_i) - k(x_i, x_i)k(x_i, x_i)^{-1}k(x_i, x_i) \), where, \( k(\cdot, \cdot) \) is any arbitrary kernel function.

**Definition 1.** Let \( x = \{x_1, x_2, \ldots, x_m\} \in X \setminus x_1 \) be any unobserved set of points. Let \( \hat{y} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m\} \) be our estimate of their outputs based on GP considering \( y_i|O \sim \mathcal{N}(\mu_{x_i|O}, \sigma^2_{x_i|O}) \). For any new point \( z \in X \setminus \{x_1 \cup x\} \), let \( y_z|O \sim \mathcal{N}(\mu_{x_i|O}, \sigma^2_{x_i|O}) \) and \( y_z|O, (x, \hat{y}) \sim \mathcal{N}(\mu_{x_i|O, x}, \sigma^2_{x_i|O, x}) \).
Under the GP model, the variance of a point \( z \) depends only on the location of the observed points and is independent of their outputs, i.e., \( \sigma^2_{z|O,x} = \sigma^2_{z|O} \). Therefore, we can update the variance of any point \( z \) after finalizing our new query set \( x \) without the knowledge of their true outputs \( y = f(x) \). The following theorem characterizes the change in the variance of \( z \) if we query \( x \).

**Theorem 1.** Assuming \( \Delta(\sigma_z) := \sigma^2_{z|O} - \sigma^2_{z|O,x} \) we have

\[
\Delta(\sigma_z) = (CA^{-1}B^T - k(z, x)) D (CA^{-1}B^T - k(z, x))^T,
\]

where, \( B = k(x, x_O) \), \( A = k(x_O, x_O) \), \( C = k(z, x_O) \) and \( D = (k(x, x) - BA^{-1}B^T)^{-1} \).

From a practical point of view, this theorem enables us to update the variance of \( z \) via computing the difference \( \Delta(\sigma_z) \) and add it to the previous value. This scheme is much faster than recalculating the variance of \( z \) directly. The computational bottleneck of this update is only the matrix inversion in \( D \) with complexity \( \mathcal{O}(m^3) \), considering the fact that \( k(x_O, x_O)^{-1} \) has been computed before, while the complexity of the direct variance computation is \( \mathcal{O}((n + m)^3) \).

The actual expected value \( \mu_{z|O,x} \) heavily depends on the true outputs \( y = f(x) \), which are not available. Without the knowledge of the true outputs, we make an estimation \( \hat{\mu}_{z|O,x} \) based on the GP-suggested output values \( \hat{y} \). We bound this estimation error in the next theorem.

**Theorem 2.** Let \( \gamma_z = \| (k(z, x) - CA^{-1}B^T) D \|_2 \). Then,

\[
\begin{align*}
|\mu_{z|O,x} - \hat{\mu}_{z|O,x}| & \leq \gamma_z \| y - \hat{y} \|_2, \\
|\mu_{z|O,x} - \mu_{z|O}| & \leq \gamma_z \| y - \mu_{x|O} \|_2.
\end{align*}
\]

Here, \( \| \cdot \|_2 \) is vector 2-norm. This theorem tells us that our estimation error at point \( z \) is proportional to the parameter \( \gamma_z \), which is known to us without the knowledge of \( y \). Intuitively, if \( \gamma_z \) is small, we would think that our estimation \( \hat{\mu}_{z|O,x} \) is accurate and hence, we can make our decision about the point \( z \) without knowing \( y \), i.e., before the result of experiment on \( x \) returns. This observation tells us that it is possible to do batch BO without a big loss in performance.

**Remark:** If we want to minimize our estimation error of \( \hat{\mu}_{z|O,x} \) in expectation, we should set \( \hat{y} = \mu_{x|O} \). This is in some sense trivial and even counter intuitive. One might claim that if the unknown function is upper-bounded by \( M \), then the best choice for \( \hat{y} \) is \( M \) since it increases the expected value around the optimal point in the GP model. However, this theorem shows that this choice is overly optimistic.

Previous theorem, provides a performance bound based on our estimation error on \( \hat{y} \), however, from a practical point of view, that bound cannot be computed since we do not know the exact values of \( \hat{y} \). As a practical measure, we would like to focus on the expected value of the estimation error as opposed to the error itself. Next corollary provides an upper-bound on the expected error, by simply taking expectation from the result of theorem 2.

**Corollary 1.** Let \( \theta_x := \sqrt{\sum_{i=1}^{m} \sigma^2_{x_i|O}} \), then

\[
\mathbb{E}_y [ |\mu_{z|O,x} - \mu_{z|O}| ] \leq \gamma_z \theta_x.
\]

Moreover,

\[
\mathbb{E}_y [ |\mu_{z|O,x} - \hat{\mu}_{z|O,x}| ] \leq \gamma_z ( \theta_x + \| \hat{y} - \mu_{x|O} \|_2 ).
\]

**Remark 1:** We focus on the second bound in this corollary, which has two terms. The first term \( (\gamma_z \theta_x) \) measures "how close" the point \( z \) is to \( x \). The second term captures the bias of our estimator \( \hat{y} \). According to this corollary, the best choice for \( \hat{y} \) is the mean \( \mu_{x|O} \).

**Remark 2:** This corollary entails that if for some small value of \( \epsilon \), we have

\[
\gamma_z ( \theta_x + \| \hat{y} - \mu_{x|O} \|_2 ) \leq \epsilon,
\]

then, we are guaranteed that

\[
\mathbb{E}_y [ |\mu_{z|O,x} - \hat{\mu}_{z|O,x}| ] \leq \epsilon.
\]
At any point \( \hat{y} \) where, sequential EI picks \( x_1 \), we estimate first sample that sequential EI picks, and density functions respectively.

\[ EI(x|\mathcal{O}) = \left( -u\Phi(-u) + \phi(u) \right)\sigma_{x|\mathcal{O}}, \quad (4) \]

where, \( u = (y_{\text{max}} - \mu_{x|\mathcal{O}})/\sigma_{x|\mathcal{O}} \) and \( y_{\text{max}} = \max_{y_i \in \mathcal{O}} y_i \). Also, \( \Phi(\cdot) \) and \( \phi(\cdot) \) represent standard Gaussian distribution and density functions respectively.

Our proposed algorithm selects a batch (possibly one) of samples at each iteration based on the EI policy, where the batch size is dynamically determined at each step. In particular, the algorithm will continue to select more experiments if the condition in (3) is satisfied for the select point \( z \).

To explain the algorithm, suppose we are at the beginning of the first round of the algorithm. Thus far, we have observed \( \mathcal{O} = f(x_{\mathcal{O}}) \) at some randomly chosen sample points \( x_{\mathcal{O}} \). To form our batch query, we start from an empty set of samples and gradually add the next best sample one at a time. The first sample we pick \( (x_1) \) is identical to the first sample that sequential EI picks \( x_1^* \), simply because both maximize the same objective, i.e., \( x_1 = x_1^* \). To pick our second sample, we estimate \( y_1^* = f(x_1^*) \) by some value \( \hat{y}_1 \). This estimation, changes the \( EI \) function of all unobserved points to some \( \hat{EI} \) function formulated as

\[ \hat{EI}(z|\mathcal{O}, x_1^*) = \left( -\hat{u}\Phi(-\hat{u}) + \phi(\hat{u}) \right)\sigma_{z|\mathcal{O}, x_1^*}, \]

where, \( \hat{u} = \frac{\max(y_{\text{max}}, \hat{y}_1) - \hat{\mu}_{z|\mathcal{O}, x_1^*}}{\sigma_{z|\mathcal{O}, x_1^*}} \). This is different from the true EI function:

\[ EI(z|\mathcal{O}, x_1^*) = \left( -u\Phi(-u) + \phi(u) \right)\sigma_{z|\mathcal{O}, x_1^*}, \]

where, \( u = \frac{\max(y_{\text{max}}, \hat{y}_1) - \mu_{z|\mathcal{O}, x_1^*}}{\sigma_{z|\mathcal{O}, x_1^*}} \). Obviously, optimizing \( \hat{EI} \) might not lead to the optimum of the true \( EI \). However, the next lemma shows that these two functions are close to each other for a good estimation \( \hat{y}_1 \).

**Lemma 1.** At any point \( z \), we have

\[ |EI(z|\mathcal{O}, x_1^*) - \hat{EI}(z|\mathcal{O}, x_1^*)| \leq \frac{1}{2} \left( 1 + \frac{\sigma_{z|\mathcal{O}}}{\sigma_{x_1^*|\mathcal{O}}} \right) |\hat{y}_1 - y_1^*|. \quad (5) \]

In the light of this lemma, there is hope that \( x_2 = \arg\max \hat{EI} \) (a potential batch sample from our algorithm) is close to \( x_2^* = \arg\max EI \) (the optimal sample picked by sequential policy). The next theorem bounds the error of our algorithm in terms of the second selected point in comparison to the sequential EI.

**Theorem 3.** Let \( \sigma_{\min} \) be the minimum singular value of the Hessian matrix \( \frac{d^2 EI}{dx^2} \) on the line intersecting \( x_2 \) and \( x_2^* \). Then,

\[ \|x_2 - x_2^*\|^2 \leq \frac{2}{\sigma_{\min}} \left( 1 + \frac{\max(\sigma_{x_2|\mathcal{O}}, \sigma_{x_2^*|\mathcal{O}})}{\sigma_{x_1^*|\mathcal{O}}} \right) |\hat{y}_1 - y_1^*|. \quad (6) \]
Here $x_2$ is the second point selected by our simulation based batch method without knowing the outcome of $x_1$, whereas $x_2^*$ is the second point selected by the sequential EI method after knowing the outcome of $x_1$.

**Remark 1:** The parameter $\Sigma_{\min}$ captures the curvature of the $\hat{EI}$ function around its optimal point $x_2$. This curvature cannot be zero unless $x_2^*$ is very far from $x_2$, which is very unlikely due to the closeness of their expected values (see Corollary 1).

**Remark 2:** This theorem shows that the sample estimation error is proportional to the square root of the estimation error of $y_1^*$. This means that the sample estimation is more sensitive to the output estimation error for functions taking a curvature cannot be zero unless $x_2^*$ is very far from $x_2$, which is very unlikely due to the closeness of their expected values (see Corollary 1).

This line of analysis can be extended to next samples. These results show that an algorithm based on the estimation error of $y_1^*$ is the second point selected by the sequential EI method after knowing the outcome of $x_1$.

**Algorithm 1 Hybrid Batch Expected Improvement**

**Input:** Total budget of experiments ($n_t$), maximum batch size ($n_b$), the predictor ($\hat{y}$), current observation $O = (x_\mathcal{O}, y_\mathcal{O})$ and stopping threshold $\epsilon$.

```plaintext
while $n_t > 0$ do
  $x_1^* \leftarrow \arg \max_{x \in X} EI(x|O)$.
  $A \leftarrow (x_1^*, \hat{y}_1)$, $n_t \leftarrow n_t - 1$.
  $z \leftarrow \arg \max_{x \in A} \hat{EI}(x|O \cup A)$.
  while $(\gamma_z(\theta_{x,A} + \|\hat{y}_A - \mu_{x,A}||_2) \leq \epsilon)$ and ($n_t > 0$) and $(|A| < n_b)$ do
    $A \leftarrow A \cup (z, \hat{y}_z)$, $n_t \leftarrow n_t - 1$.
    $z \leftarrow \arg \max_{x \in A} \hat{EI}(x|O \cup A)$.
  end while
  $y_A \leftarrow \text{RunExperiment}(x_A)$
  $O \leftarrow O \cup (x_A, y_A)$
end while
return $\max(y_\mathcal{O})$
```

In early stages, this algorithm behaves more like a sequential policy since the criterion for building up a batch is very hard to satisfy, mainly because $\theta_x$ is large when we have only a few samples in $O$. After collecting enough samples, the term $\theta_x$ starts decreasing and as it gets closer and closer to zero, we can select larger and larger batch sizes. Thus, the algorithm gradually transits into a batch policy while maintaining a close match to the performance to the pure sequential policy.

### 4 Experimental Results

**Benchmarks.** We consider 6 well-known synthetic benchmark functions: **Cosines** and **Rosenbrock** [1][5] over $[0, 1]^2$, **Hartman(3)** [6] over $[0, 1]^3$, **Hartman(6)** [6] over $[0, 1]^6$, **Shekel** [6] over $[3, 6]^4$ and **Michalewicz** [10] over $[0, \pi]^5$. The mathematical formulation of these function are shown in Table 1.

<table>
<thead>
<tr>
<th>Function</th>
<th>Mathematical Formulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cosines(2)</td>
<td>$1 - (u^2 + v^2 - 0.3 \cos(3\pi u) - 0.3 \cos(3\pi v))$</td>
</tr>
<tr>
<td>Rosenbrock(2)</td>
<td>$10 - 100(y - x^2)^2 - (1 - x)^2$</td>
</tr>
<tr>
<td>Hartman(3)</td>
<td>$\sum_{i=1}^4 \Omega_i \exp \left(-\sum_{j=1}^d A_{ij}(x_j - P_{ij})^2 \right)$</td>
</tr>
<tr>
<td>Shekel(4)</td>
<td>$\sum_{i=1}^m \omega_i + \sum_{j=1}^4 (x_1 - B_{ij})^2$</td>
</tr>
<tr>
<td>Michalewicz(5)</td>
<td>$-\sum_{i=1}^3 \sin(x_i) \sin \left(\frac{\sin^2(\pi x_i)}{\pi}\right)^{20}$</td>
</tr>
</tbody>
</table>

The other two real benchmarks are **Fuel Cell** and **Hydrogen**. In Fuel Cell, the goal is to maximize the generating electricity from microbial fuel cell with respect to anode’s nano structure properties, average area and average circularity. We fit a regression model on the data to build our function $f(\cdot)$ for evaluation. In Hydrogen benchmark, the data has been collected as part of a study on Hydrogen production from a particular bacteria. The goal is to maximize...
the amount of Hydrogen production by optimizing the PH Nitrogen levels of growth medium. Both Fuel cell and Hydrogen data are in $[0, 1]^2$. Their contour plots are shown in figure 2.

**Settings.** We use GP to build a posterior over our unknown function $f(\cdot)$. Our GP is based on a zero-mean prior and squared exponential kernel function $k(x, y) = \exp(-\frac{1}{2} \| x - y \|^2)$, with kernel width $l = 0.01\sum_{i=1}^{d} l_i$, where, $l_i$ is the length of the $i^{th}$ dimension [2]. Based on this kernel function, we can directly drive the next two corollaries from theorem 1.2.

**Corollary 2.** For all points $z \in X \setminus \{O \cup x\}$ such that $|z| = 1$, (where $x = \{x_i\}$), and using squared exponential kernel function, $k(x, y) = e^{-\frac{1}{2}(x-y)^2}$ as Gaussian process kernel function, then $\Delta(\sigma_z) \geq \varepsilon$ if

$$
\| z - x_i \|^2 \leq -l \ln \left( \sqrt{n} \| A^{-1}B^T \|_2 + \sigma_{x_i} \right) \sqrt{\varepsilon},
$$

where, $\| \cdot \|_2$ is the vector second norm.

This corollary entails that after selecting the first experiment $x_1^*$, the set of points $z$ such that $\Delta(\sigma_z) \geq \varepsilon$ are located inside a hyper sphere centered at $x_1^*$. The radius of the hyper sphere is a function of the variance of $x_1^*$. In other words,
those inside the circle are those whose variance is affected significantly (more than $\epsilon$) when $x_1^*$ is selected.

**Corollary 3.** Under the assumption of Corollary 2, if

$$\|z - x_1^n\|^2 \leq -\ln \sqrt{\frac{\pi \epsilon^2}{2 \sigma^2_{x1}}} - n \| A^{-1}B^T \|_2^2,$$

(7)

Similar to corollary 2, the corollary 3 represents a hyper sphere centered at $x_1^*$ and those points who are inside the hyper sphere are those whose expected values are affected by more than $\epsilon$ when $x_1^*$ is selected.

We run our algorithm on each benchmark 100 times and the simple regret is reported as the result that is the difference between the optimal value of $f(\cdot)$, known as $M$, and the output value of the predicted best experiment in $X$ after finishing the experimental procedure. In each run, the algorithm starts with 2 initial random points for 2, 3-dimensional benchmarks and 5 initial random points for higher dimensional benchmarks. This is a really hard situation to start with which is also similar to our motivating application 3. The total number of experiments $n_l$ is set to 15 for 2, 3-dimensional and 30 for the higher dimensional benchmarks and the maximum batch size at each iteration, $n_b$, is set to 5. The parameter $\epsilon$ is set to 0.02 for 2, 3-dimensional and 0.2 for higher dimensional benchmarks. We try to set our algorithm parameters very close to real applications in which we usually start with a very small number of experiments, and we are restricted with total number of experiment in which we can ask.

**Results.** Table 2 shows the performance of our algorithm on each benchmark via 6 different scenarios with respect to our estimate $\hat{y}$ which is listed as follows: 1) $\hat{y} = M$ indicating we expect to observe the best possible output in each experiment selection by EI; 2) $\hat{y} = (1 + \zeta) y_{\text{max}}$ which means each step of EI algorithm is expected to improve the best current observation by margin $\zeta$, we set the value of $\zeta$ to 0.1 in our experiment; 3) $\hat{y} = y_{\text{max}}$ such that $y_{\text{max}} = \max_{y_i \in Y_O} y_i$ is our current best observation; 4) $\hat{y} = y_{\text{max}}$ which means we set the value of $\hat{y}$ at any point $x$ with the output expectation at that point; 5) $\hat{y} = y_{\text{min}}$ such that $y_{\text{min}} = \min_{y_i \in Y_O} y_i$ is the current minimum observed output; and 6) $\hat{y} = \text{random}$ is a uniform random draw in $[y_{\text{min}}, y_{\text{max}}]$.

To illustrate the speedup over the sequential policy, we calculate the speedup as the percentage of the samples in the whole experiment that are selected to be run in parallel in the batch mode. In general, if we finish $n_l$ samples in $T$ steps, the speedup is calculated as $1 - \frac{T}{n_l}$. Clearly, the maximum speedup in our setting is $\%78$, that can be only achieved if we select 5 experiments at each and every time steps. In addition, we have reported the performance of the sequential EI and pure random policy.

Interestingly, all of the 6 proposed scenarios perform approximately the same as EI. Moreover, the results show that we are able to achieve up to $\%78$ speedup for Shekel function when $\hat{y} =  \hat{\mu}$. This is more appealing considering the fact that we started from very few samples. In these settings loosing even a few sample would degrade the performance. This shows that, regardless of our estimation for $\hat{y}$, our algorithm is able to identify when to go sequential and when to go batch. This observation states that our estimation of $\hat{y}$ does not have significant effect on the performance (simple regret), and it can only influence our speedup rate.

Interestingly, all of the 6 proposed scenarios perform approximately the same as EI. Moreover, the results show that we are able to achieve up to $\%78$ speedup for Shekel function when $\hat{y} = \hat{\mu}$. This is more appealing considering the fact that we started from very few samples. In these settings loosing even a few samples would degrade the performance. This shows that, regardless of our estimation for $\hat{y}$, our algorithm is able to identify when to go sequential and when to go batch.

Figure 2: The contour plot for FuelCell and Hydrogen.
of our knowledge. In addition, it is more practical than Matching in real application since its computational running
is the result of recently proposed approach by Azimi
8
on proposed
but without checking the stopping criterion. This scheme is similar to Ginsbourger
et al.
without degrading the performance. This result motivates for a batch approach which we call it
Approach. We select
stopping criterion is mostly met, i.e.,
That is because by setting
in algorithm
and
are expected to be maximized. On the other hand, it can be easily verified

to
is expected to be maximized while
and
is expected to be minimized.
That is because by setting
, the EI policy selects the next point very far from the selected points since the expected output value of the points which are close to previously selected points are very small and thus their EI values are minimized.
Finally, it can be seen that by setting \( \hat{y} \) to \( \hat{\mu} \), the maximum speedup among the proposed scenarios can be achieved. This is not surprising since the quantity \( \gamma \| \hat{y} - \mu_{x|O} \|_2 \) = 0 and thus the stopping bound is expected to be very small. Note that we found that the size of the batch increased as the number of experiments increases in experimental procedure. This is consistent with our theoretical results in which the stopping bounds decreases as the variances decreases.

\( \mu \)-Constant Batch Approach. Our analysis and in particular Corollary 1 indicates that setting \( \hat{y} \) to \( \mu_{x|O} \), the stopping criterion is mostly met, i.e., \( \gamma \| \theta_x \| \leq \epsilon \). Thus, a batch of experiments are requested at each most iterations without degrading the performance. This result motivates for a batch approach which we call it \( \mu \)-Constant Batch Approach. We select \( k \) (for some predefined fixed batch size \( k \)) samples at each iteration according to our algorithm but without checking the stopping criterion. This scheme is similar to Ginsbourger et al. [7]. We run this algorithm on proposed 8 benchmarks for different batch size 5 and 10. Figure 5 and 3 show the performance of proposed approach along with 5 competitive approaches: 1) Sequential EI; 2) \( \hat{y} = M \); 3) \( \hat{y} = y_{\text{max}} \); 4) \( \hat{y} = y_{\text{min}} \); and 5) Matching, which is the result of recently proposed approach by Azimi et al. [2]. We start with 5 initial experiments for 2, 3–dimensional functions and 20 for higher dimensional benchmarks. The budget is set to 30 for 2, 3–dimensional functions and 60 for higher dimensional benchmarks.

Interestingly, the results show that the proposed \( \mu \)-constant batch approach can perform very competitive to Matching approach which is the best existing batch Bayesian optimization approach in the literatures, to the best of our knowledge. In addition, it is more practical than Matching in real application since its computational running time is significantly less than Matching algorithm.

Figures 5 and 3 show that the choice of \( \hat{y} = M \) is very aggressive and the batch constant method performs poorly.
This result is consistent with the proposed results in Table 2 in which the speedup is very small. This shows that our algorithm distinguishes \( \hat{y} = M \) as a very bad estimation and then does not allow requesting batch of experiments based on this estimation. In the other hand, \( \hat{y} = \mu \) achieves the best performance comparing to other constant predictor approaches which is consistent with the results presented in Table 2. Finally, it is clear from our result that there is a trade-off between performance and speedup, but our result shows that the loose of performance is negligible in comparison to the gained speedup.

5 Conclusion

In the Bayesian optimization framework, we investigated the problem of batch query selection with the goal to maintain the performance of a sequential policy. Although our result is for general BO problems, for the sake of clarification, we focused on the task of optimizing an unknown non-convex/concave function. We made two main contributions in this paper:

Firstly, we introduce a systematic way to analyze the performance and limits of simulation-based batch BO methods by a) proving universal bounds on the bias caused by the simulation (estimation-of-outcome) error, and, and b) analyzing the selection of the second experiment when we have an estimate of the outcome of the first experiment. In all cases, we provide theoretical bounds on the error, relating the simulation error to the prediction error of the next best experiment.

Secondly, based on the analysis above, we proposed an algorithm that behaves optimally in expectation. This algorithm at each step decides whether or not to pick another query to add to the current batch, and as such dynamically determine the appropriate batch size at each step. In early iterations, our algorithm behaves more similar to the sequential policy and gradually moves toward a batch policy with variable batch sizes.

The empirical evaluation over both synthetic and real data shows substantial speedup (up to % 78) compared to the corresponding sequential policy, with little to nothing loss in optimization performance.

References


A Proof of Theorem 1

Recalling the notation introduced in the Theorem statement, we have

\[ \Delta(\sigma_z) = CA^{-1}C^T - [C \ k(z, x)] \begin{bmatrix} A & B^T \\ B & k(x, x) \end{bmatrix}^{-1} \begin{bmatrix} C^T \\ k(z, x) \end{bmatrix} \]

\[ = CA^{-1}C^T - [C \ k(z, x)] \begin{bmatrix} A^{-1} + A^{-1}B^TDBA^{-1} & -A^{-1}B^TD \\ -DBA^{-1} & D \end{bmatrix} \begin{bmatrix} C^T \\ k(z, x) \end{bmatrix} \]

\[ = (CA^{-1}B^T - k(z, x)) D (BA^{-1}C^T - k(z, x))^T. \]

This concludes the proof of the theorem.

B Proof of Theorem 2

By definition and block matrix inversion lemma, we have

\[ \mu[z|O, x] - \hat{\mu}[z|O, x] = k(z, \{x_O, x\})k(\{x_O, x\}, \{x_O, x\})^{-1} \begin{bmatrix} 0 \\ y - \hat{y} \end{bmatrix} \]

\[ = (k(z, x) - CA^{-1}B^T) D(y - \hat{y}). \]

For the second part, we have

\[ \mu[z|O] - \mu[z|O, x] = CA^{-1}y_O - [C \ k(z, x)] \begin{bmatrix} A & B^T \\ B & k(x, x) \end{bmatrix}^{-1} \begin{bmatrix} y_O \\ y \end{bmatrix} \]

\[ = CA^{-1}y_O - [C \ k(z, x)] \begin{bmatrix} A^{-1} + A^{-1}B^TDBA^{-1} & -A^{-1}B^TD \\ -DBA^{-1} & D \end{bmatrix} \begin{bmatrix} y_O \\ y^* \end{bmatrix} \]

\[ = (CA^{-1}B^T - k(z, x)) D (BA^{-1}y_O - y) \]

\[ = (CA^{-1}B^T - k(z, x)) D (\mu[z|O] - y). \]

This concludes the proof of the theorem.

C Proof of Lemma 1

Let \( \Delta_z = \max(y_{max}, y_1^*) - \mu[z|O, x_1^*] \). Using Theorem 2, we have

\[ \hat{\Delta}_z := \max(y_{max}, \hat{y}_1) - \hat{\mu}[z|O, x_1^*] \]

\[ = \max(y_{max}, y_1^*) - \mu[z|O, x_1^*] + \max(y_{max}, \hat{y}_1) - \max(y_{max}, y_1^*) \]

\[ - \frac{1}{\sigma_{z|O}^2} (k(z, x_1^*) - k(z, x_O)k(x_O, x_O)^{-1}k(x_O, x_1^*)) \hat{y}_1 - y_1^* \]

\[ = \Delta_z + \max(y_{max}, \hat{y}_1) - \max(y_{max}, y_1^*) - \rho_{z, x_1^*} \frac{\sigma[z|O]}{\sigma_{z|O}} (\hat{y}_1 - y_1^*) \]

\[ = \Delta_z + \delta_z. \]

Here, \( \rho_{z, x_1^*} \) represents the correlation coefficient between \( x \) and \( x_1 \). Thus, we have

\[ |\delta_z| \leq \left( 1 + \frac{\sigma[z|O]}{\sigma_{z|O}} \right) |\hat{y}_1 - y_1^*|. \]
By mean-value theorem, there exists \( \alpha \in [0,1] \), such that
\[
-\Delta_z \Phi \left( \frac{-\Delta_z}{\sigma_z|O,x_1^*} \right) + \sigma_z|O,x_1^* \Phi \left( \frac{-\Delta_z}{\sigma_z|O,x_1^*} \right) = -\Delta_z \Phi \left( \frac{-\Delta_z + \alpha \delta_z}{\sigma_z|O,x_1^*} \right) + \sigma_z|O,x_1^* \Phi \left( \frac{\Delta_z}{\sigma_z|O,x_1^*} \right) - \Phi \left( \frac{-\Delta_z + \alpha \delta_z}{\sigma_z|O,x_1^*} \right) \delta_z.
\]

Thus,
\[
\left| EI(z) - \widehat{EI}(z) \right| = \Phi \left( \frac{-\Delta_z + \alpha \delta_z}{\sigma_z|O,x_1^*} \right) \left| \delta_z \right| \\
\leq \frac{1}{2} \left| \delta_z \right| \leq \frac{1}{2} \left( 1 + \frac{\sigma_{x_1|O}}{\sigma_{x_1^*|O}} \right) \left| \hat{y}_1 - y_1 \right|.
\]

This concludes the Proof of Lemma.

D Proof of Theorem 3

By optimality of \( x_2 \) and \( x_2^* \), we have
\[
EI(x_2) - \widehat{EI}(x_2) \leq EI(x_2^*) - \widehat{EI}(x_2^*) \leq EI(x_2^*) - \widehat{EI}(x_2^*).
\]

Using Lemma 1, we get
\[
\left| EI(x_2^*) - \widehat{EI}(x_2) \right| \leq \frac{1}{2} \left( 1 + \frac{\max(\sigma_{x_2|O},\sigma_{x_2^*|O})}{\sigma_{x_1|O}} \right) \left| \hat{y}_1 - y_1 \right|.
\]

We can continue
\[
\widehat{EI}(x_2) - \widehat{EI}(x_2^*) \leq \left| \widehat{EI}(x_2) - EI(x_2^*) \right| + \left| EI(x_2^*) - \widehat{EI}(x_2^*) \right| \\
\leq \left( 1 + \frac{\max(\sigma_{x_2|O},\sigma_{x_2^*|O})}{\sigma_{x_1|O}} \right) \left| \hat{y}_1 - y_1 \right|.
\]

By optimality of \( x_2^* \), the derivative of EI is zero at \( x_2^* \) and Taylor series expansion yields that for some \( \alpha \in [0,1] \), we have
\[
\widehat{EI}(x_2^*) - \widehat{EI}(x_2) = \frac{1}{2} (x_2^* - x_2)^T \frac{d^2 \widehat{EI}}{dx^2} ((1 - \alpha)x_2^* + \alpha x_2) (x_2^* - x_2).
\]

Finally, we get
\[
\left\| x_2^* - x_2 \right\|_2^2 \leq \frac{2 \left| \widehat{EI}(x_2^*) - \widehat{EI}(x_2) \right|}{\Sigma_{\mathrm{min}}} (\frac{d^2 \widehat{EI}}{dx^2} ((1 - \alpha)x_2^* + \alpha x_2)) \\
\leq \frac{2}{\Sigma_{\mathrm{min}}} \left( 1 + \frac{\max(\sigma_{x_2|O},\sigma_{x_2^*|O})}{\sigma_{x_1|O}} \right) \left| \hat{y}_1 - y_1 \right|.
\]

E Proof of Corollary 2

From theorem there is an interesting finding which shows that the difference of variance of any point \( z \) in the input space after adding the point \( x^* \) to our observation set is exactly \( D(k(z,x_1^*) - BA^{-1}CT)^2 \) if we consider \( x_1^* \) as a single point. Since \( \delta_2^z - \delta_2^{x^*} > 0 \), therefore \( m \geq 0 \). In addition, when \( |x^*| = 1 \), it can be shown that \( m^{-1} = \sigma^{x^*} \).

Thus, we are interested in the points where \( \delta_2^z - \delta_2^{x^*} \geq \epsilon \geq 0 \). Therefore we have:
\[
\delta_2^z - \delta_2^{x^*} - \epsilon \geq 0 \\\ \\text{and} \\ \\ Dk(x_1^*, z)^2 - 2DCA^{-1}BT k(x_1^*, z) + (D(CA^{-1}BT))^2 - \epsilon \geq 0
\]
this is a quadratic function of \( k(x_1^*, z) \) with 2 real roots as follow:

\[
k(x_1^*, z) = \begin{cases} 
  r_1 = CA^{-1}B^T + \sqrt{\frac{\epsilon}{D}} \\
  r_2 = CA^{-1}B^T - \sqrt{\frac{\epsilon}{D}}
\end{cases}
\]

(10)

So we are interested in the region where \( k(x_1^*, z) \geq r_1 \) or \( k(x_1^*, z) \leq r_2 \). For large value of \( \epsilon \) the \( r_2 < 0 \) and since \( k(x_1^*, z) > 0 \), we are only interested in where \( k(x_1^*, z) \geq r_1 \). Therefore we have

\[
1 \geq k(x_1^*, z) = e^{-\frac{1}{1-2k}} \geq CA^{-1}B^T + \sqrt{\frac{\epsilon}{D}} \geq 0
\]

(11)

We are trying to introduce an upper bound for \( r_1 \) which is free from \( P_2 \). Clearly \( CA^{-1}B^T \leq |CA^{-1}B^T| \). Then we have,

\[
|CA^{-1}B^T| = ||CA^{-1}B^T||_2 \\
\leq ||C||_2 ||A^{-1}B^T||_2 \quad \text{Cauchy-Shwarz inequality}
\]

(12)

\[
\leq \sqrt{n} ||C||_\infty ||A^{-1}B^T||_2 \\
\leq \sqrt{n} ||A^{-1}B^T||_2 \quad \text{sinec} \ 0 \leq ||C||_\infty \leq 1
\]

Therefore we are certain about the point satisfying the following equation

\[
k(x_1^*, z) \geq \sqrt{n} ||A^{-1}B^T||_2 + \sqrt{\frac{\epsilon}{D}} \\
\| z - x^* \|^2 \leq -l \ln \left( \sqrt{n} ||A^{-1}B^T||_2 + \sqrt{\frac{\epsilon}{D}} \right)
\]

(13)

F Proof of Corollary 3

\[
\| (CA^{-1}B^T - k(x_1^*, z)) \|_\infty \sqrt{\frac{\rho}{\pi}} \| \sigma z_1^* \| \Omega_1 \geq \epsilon \]

\[
| (CA^{-1}B^T - k(x_1^*, z)) | \geq \frac{\epsilon}{\sqrt{\frac{\rho}{\pi}} \| \sigma z_1^* \| \Omega_1} \\
| (CA^{-1}B^T - k(x_1^*, z)) |^2 \geq \left( \frac{\epsilon}{\sqrt{\frac{\rho}{\pi}} \| \sigma z_1^* \| \Omega_1} \right)^2
\]

(14)

\[
(CA^{-1}B^T)^2 + k(x_1^*, z)^2 \geq \frac{\pi \epsilon^2}{2\sigma z_1^* \| \Omega_1}D^2 \\
k(x_1^*, z)^2 \geq \frac{\pi \epsilon^2}{2\sigma z_1^* \| \Omega_1}D^2 - n \| A^{-1}B^T \|^2_2 \\
\| z - x^* \|^2 \leq -l \ln \left( \frac{\pi \epsilon^2}{2\sigma z_1^* \| \Omega_1}D^2 - n \| A^{-1}B^T \|^2_2 \right)
\]

Note that \( |a - b|^2 \leq 2 \times (a^2 + b^2) \). Therefore \( E[|\mu_2|\Omega_\omega - \tilde{\mu}_2|\Omega_\omega]| \geq \epsilon \) if we have

\[
\| z - x^* \|^2 \leq -l \ln \left( \frac{\pi \epsilon^2}{2\sigma z_1^* \| \Omega_1}D^2 - n \| A^{-1}B^T \|^2_2 \right)
\]

(15)