Proceedings of the 2020 Winter Simulation Conference
K.-H. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, eds.

PARTITION-BASED BAYESIAN OPTIMIZATION FOR STOCHASTIC SIMULATIONS

Songhao Wang
Szu Hui Ng

Department of Industrial Systems Engineering and Management
National University of Singapore
10 Kent Ridge Crescent
Singapore 119260, SINGAPORE

ABSTRACT

Bayesian optimization (BO) is a popular simulation optimization approach. Despite its many successful applications, there remain several practical issues that have to be addressed. These includes the non-trivial optimization of the inner acquisition function (search criterion) to find the future evaluation points and the over-exploitative behavior of some BO algorithms. These issues can cause BO to select inferior points or get trapped in local optimal regions before exploring other more promising regions. This work proposes a new partition-based BO algorithm where the acquisition function is optimized over a representative set of finite points in each partition instead of the whole design space to reduce the computational complexity. Additionally, to overcome over-exploitation, the algorithm considers regions of different sizes simultaneously in each iteration, providing focus on exploration in larger regions especially at the start of the algorithm. Numerical experiments show that these features help in faster convergence to the optimal point.

1 INTRODUCTION

Computer models that simulate the stochastic behaviors of real systems are often used to help determine the optimal settings of such systems. This decision process, which utilizes the computer models, is called simulation optimization (SimOpt, or optimization via simulation). In SimOpt problems (we consider minimization problems for illustration in this paper), as the computer models capture real complex systems, the response function has no closed form and its value at each input location can only be observed via simulations. The simulations considered in this work have heteroscedastic noise, i.e., the noise of the simulations at different locations can be different. This type of problem has various applications in simulations in areas including operations research, transportation, manufacturing, supply chain management, etc (Hong and Nelson 2009; Pasupathy and Ghosh 2013). Furthermore, since no structure on the response function is assumed (e.g., non-convex, non-smooth), traditional mathematical programming techniques that require gradient information are often not suitable to solve these problems. Instead, the response function is often treated as a black-box and the typical way to solve it is to generate multiple candidate inputs iteratively and evaluate them with simulations of the computer model.

The particular SimOpt approach studied in this work, Bayesian optimization (BO), is a popular approach for black-box function optimization and has seen extensive applications in engineering design and computer science (see Shahriari et al. (2015) and Frazier (2018) for reviews). BO is a type of metamodel-based optimization algorithm, where a Gaussian process (GP) model is built with the existing observations to guide the iterative search. Such algorithms are very popular especially when each run of the simulation is expensive and the total computational budget is limited. For the stochastic SimOpt problems considered in this work, we utilize a similar two-stage procedure in our BO algorithm as in Quan et al. (2013), Jalali et al. (2017) and Pedrielli et al. (2020). The first stage is to select the next design point to evaluate and the second stage is to allocate additional replications of simulations to existing design points to reduce the noise. In each iteration, BO first builds a GP model as an approximation of the true response (see Figure 1).

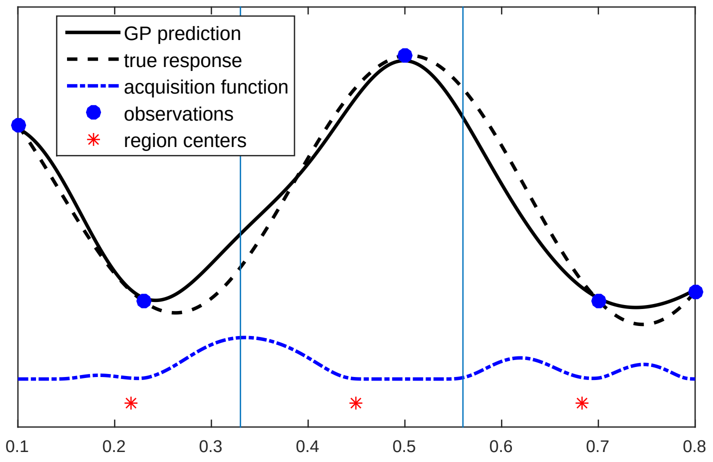

Figure 1: Illustration of BO.

Algorithm 1 Standard BO Framework of Stochastic Simulations

```
1: for $t = 1,2...$ do
2:     Searching Stage:
3:     $x_t = \arg\max_{x\in\mathscr{X}} a(x)$
4:     Evaluate response at $x_t$
5:     Allocation Stage:
6:     Allocate additional replications to existing design points
7:     Update the GP model.
8: end for
```

Although BO has many successful applications, there are a few challenges that may hinder its wider usage. Here, we highlight two key challenges.

1. Firstly, *optimizing the acquisition function in the continuous design space is not a trivial task.* This function is typically non-convex and highly non-linear. To optimize the acquisition function, some prior works suggest discretizing the design space (converting a continuous design space to a discrete one) (Jalali et al. 2017). This approach runs the risk of missing the optimal result when an improper or not-fine-enough discretization is applied. Others choose to use some direct optimization algorithms, such as DIRECT (Jones et al. 1993), random search (Bergstra and Bengio 2012) and adaptive grids (Bardenet and Kégl 2010), and gradient-based algorithms such as the multi-start gradient-based optimization algorithms (Snoek et al. 2012). Although these practical approaches are feasible, it can be expensive (like the DIRECT algorithm) or difficult to be assessed in terms of the accuracy (i.e, it is hard to tell whether the global optimizer of the acquisition function is found), which can then lead to unreliable choices of next design points in constrained time. These inferior choices will further influence the convergence results of the BO algorithm, since almost all the convergence guarantees of BO are established assuming that $a(x)$ is accurately optimized (Wang et al. 2014).

2. The second challenge of BO is *its over-exploitation characteristic with some acquisition functions*. Although many BO algorithms are designed to balance exploitation (search the current promising region) and exploration (search less explored regions for potentially better solutions) such as the EI function, when used in practice, they tend to overly exploit the current best regions before jumping out for exploration (Ranjan et al. 2011). As a result, when the budget is limited, we may lose the best solution when it spends too much time in sub-optimal regions.

To address these issues, we propose a partition-based stochastic BO (pStoBO) algorithm. In a traditional partition-based algorithm, each region in the current partition has a ***performance index*** to evaluate the performance of the points in that region. This can be the function value(s) at one or a few representative points from this region (Valko et al. 2013; Shi and Ólafsson 2000). The partition-based algorithms generally have two key stages: Expansion Stage and Evaluation Stage. The Expansion Stage is to choose the best region (with the best performance index) and then further expand it into smaller regions. Then, the Evaluation Stage is to compute the performance indices in the new regions obtained from this expansion so that these regions can be considered for expansion in the following iterations. We introduce a similar partition scheme to BO to try to address the aforementioned challenges. First, we choose the center of each region as the representative point and, different from traditional partition-based algorithms, the acquisition function value of the center as the performance index of the region. As a result, in the Expansion Step, we only optimize $a(x)$ over the region centers in the current partition, which is a finite set. This largely reduces the computational burden and addresses challenge 1. As the algorithm iterates, the design space will be partitioned into regions of different sizes. There is also the risk that the algorithm continues to focus on partitioning a specific region into finer and finer subregions, resulting in over-exploitation in that region and leaving some larger regions unexplored. To address challenge 2, in each iteration, we simultaneously consider the best region of each size to partition, enabling larger regions, that are less explored, to be explored at every iteration. Figure 2 illustrates this idea. Here we see that the best region (the one optimizing $a(x)$) of each size is partitioned in this iteration.

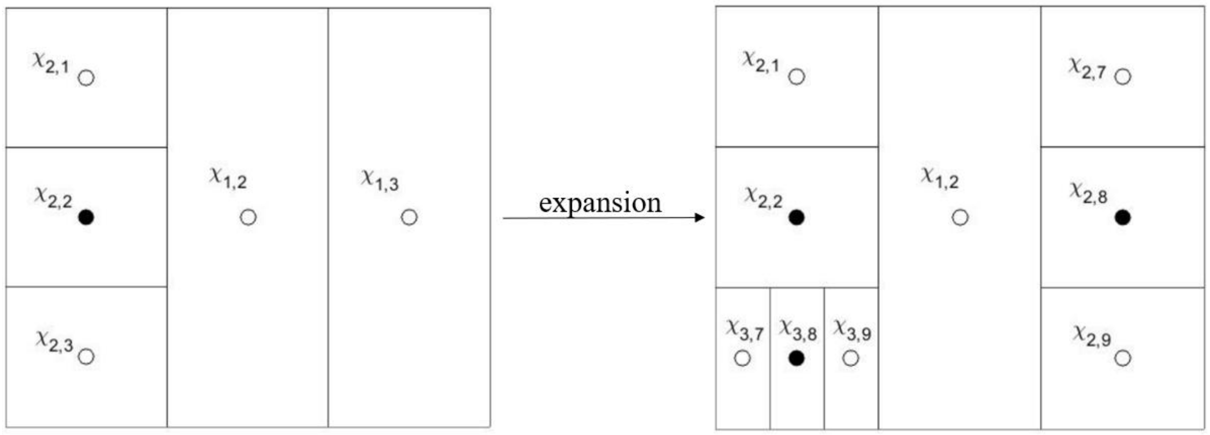

Figure 2: An example of the partitioned space in one iteration. There are two sizes of regions at the beginning of this iteration and one from each size ($\mathscr{X}_{2,3}$ and $\mathscr{X}_{1,3}$) was expanded.

As mentioned above, the performance index of each region is chosen as the $a(x)$ value at the center instead of the true function values from the points within the region. As a result, in the Evaluation Stage, there is no need to evaluate points from the newly obtained regions to compute their indices as the indices can be easily computed from the acquisition function. However, we still need to decide if some "potentially good" points should be evaluated in each iteration to update our knowledge about the global optimal as well as the response surface. We therefore keep the Evaluation Stage of traditional partition-based algorithms to decide whether to evaluate some design points. With these two 'Expansion' and 'Evaluation Stage' from the partition-based algorithm, we replace the 'Searching Stage' in the traditional stochastic BO algorithm in Algorithm 1.

The rest of this article is organized as follows. In Section 2, we review some basics for the stochastic GP model and traditional partition-based algorithms. In Section 3, we propose the pStoBO algorithm. In Section 4, some numerical examples are provided to show the efficiency of the method. In Section 5, we outline some unsolved key issues.

2 REVIEWS OF THE STOCHASTIC GP MODEL AND PARTITION-BASED APPROACHES

In this section, we first review the Stochastic GP model in Section 2.1 which is used as the metamodel in the proposed algorithms to guide the search. In Section 2.2, we review the general partition-based approach in optimization algorithms

2.1 Stochastic GP Model Basics

The standard stochastic GP model assumes that the response of stochastic simulation can be represented in the following form:

$$Y(x) = Z(x) + \varepsilon(x) = \nu(x)^T\beta + \eta(x) + \varepsilon(x),$$

where $Z(x)$ represents the noise-free response which can be further decomposed into the mean function, $\nu(x)$, and a second-order stationary GP, $\eta(x)$. $\nu(x)$ is a vector of known functions representing our belief of the mean performance of $Z(x)$. Without loss of generality, we assume little prior knowledge of $\nu(x)$ and choose a zero mean of the GP with $\nu(x) = 0$, which is a common practice in BO. In this model, $\eta(x)$ is assumed to be zero-mean with covariance $c(x_1,x_2) := \text{cov}(\eta(x_1),\eta(x_2)) = \sigma_z^2 \text{corr}(x_1,x_2)$, where $\sigma_z^2 = \text{var}(\eta(x))$. A popular choice of $\text{corr}(x_1,x_2)$ is the Gaussian correlation function: $\text{corr}(x_1,x_2) = \exp\left\{\sum_{j=1}^d \theta_j(x_{1,j} - x_{2,j})^2\right\}$, where $x_{i,j}$ is the jth coordinate of x_i and θ is the sensitivity parameter. $\varepsilon(x)$ is the random noise with mean zero and variance $\sigma_\varepsilon^2(x)$. This noise is assumed to be independent of $\eta(x)$.

The predictor $\widehat{Z}(x_0)$ and the associated predictive variance at an unknown input point x_0 can be derived as:

$$\widehat{Z}(x_0) = c(x_0)^T R^{-1}\mathscr{Y},$$

$$\widehat{s}^2(x_0) := \text{var}(\widehat{Z}(x_0)) = \sigma_z^2 - c(x_0)^T R^{-1} c(x_0),$$

where $\mathscr{Y} = (\bar{y}(x_1),...,\bar{y}(x_m))^T$ is the observations vector at design points $x_1,...,x_m$. $\bar{y}(x_i) = \frac{1}{n_i}\sum_{j=1}^{n_i} y(x_i,\xi_j)$, $i = 1,..,m$. $c(x_0)$ is the $m \times 1$ covariance vector of x_0 with existing design points whose ith entry is $c(x_0,x_i)$, $R = R_z + R_\varepsilon$ is the covariance matrix of the design points, where R_z is the $m \times m$ covariance matrix for the spatial process whose (i,j)th entry is $c(x_i,x_j)$ and R_ε is the $m \times m$ diagonal covariance matrix for the noises whose ith diagonal entry is $\text{var}(\varepsilon(x_i))$.

When used in practice, the inputs to the model is the design set D, the observation vector $\mathscr{Y}$ and the noise variance matrix R_ε, and typically, $\mathscr{Y}$ and R_ε are estimated with sample mean and sample variance from the simulation results. With these inputs, the hyperparameter θ can be obtained by maximizing the likelihood of $\mathscr{Y}$.

2.2 Review of the Partition-Based Approach in Optimization Algorithms

In parallel with BO, direct algorithms have also been developed for global black-box optimization, and the partition-based approach (Jones et al. 1993; Kleinberg et al. 2008; Pinter 1997; Munos 2011; Wang et al. 2017; Valko et al. 2013) is one of the promising ones among them. Generally, this type of algorithm assumes either global smoothness of $z(x)$ (Pinter 1997; Jones et al. 1993) or local smoothness around its global optimum (Kleinberg et al. 2008; Munos 2011). Most typically, the functions are assumed to be global or local Lipschitz (Valko et al. 2013).

A partition-based algorithm divides the input space into regions. Each region can be represented by $\mathscr{X}_{h,i}$ (see Figure 2 for an example), where the first index h is a ***size index***, indicating that the size of the region is $1/k^h$ (the size of the whole design space $\mathscr{X}$ is assumed to be 1 and it is represented as $\mathscr{X}_{0,1}$). After one partition, $\mathscr{X}_{h,i}$ will be divided into k new equal-sized ***subregions*** $\mathscr{X}_{h+1,(i-1)k+1},\dots,\mathscr{X}_{h+1,ik}$. Here, $\mathscr{X}_{h,i}$ is called the ***parent region*** of the k subregions. In Figure 2, for example, after one partition, $\mathscr{X}_{1,3}$ was partitioned into $\mathscr{X}_{2,7}$, $\mathscr{X}_{2,8}$, and $\mathscr{X}_{2,9}$.

As introduced in Section 1, in a traditional partition-based algorithm, each region is given a promising index to measure the performance of this region. For example, in DIRECT, this index is chosen as the lower bound of the region center under global Lipschitz condition (Jones et al. 1993). In the nested partition algorithm (Shi and Ólafsson 2000), in each new obtained region, a few randomly picked points will be evaluated and the performance index is chosen as the minimum value of the responses at these points. In SOO (Valko et al. 2013), the performance index is chosen as the response value at the region center. Then, the region with the best performance index will be selected to partition in the Expansion Stage and then the indices of the newly obtained regions will be computed in the Evaluation Stage. In pStoBO, we follow the work of DIRECT and SOO to choose the region center as the representative point. Specifically, from region $\mathscr{X}_{h,i}$, the center point $x_{h,i}$ is the *representative point*. The representative points of all the regions in the current partition constitute the current *center points set*, denoted as $\mathscr{S}$. For example, in Figure 2 right, $\mathscr{S}=\{x_{1,2},x_{2,1},x_{2,2},x_{2,7},x_{2,8},x_{2,9},x_{3,7},x_{3,8},x_{3,9}\}$.

The partitioned-based approaches systematically partition the design space based on the simulation results and gradually concentrate on the regions potentially containing optimal solutions. It typically utilizes the information from global and local searches and can be shown to converge under mild conditions (Shi and Ólafsson 2000). In pStoBO, we borrow this sequential partitioning approach and optimize the relevant acquisition function over the set $\mathscr{S}$ to reduce the computational burden.

3 pStoBO ALGORITHM

In this section, we present the new algorithm, pStoBO. The parameters used are listed in Table 1 and the algorithm is summarized in Algorithm 2. In pStoBO, we call the operation "dividing one region into k sub-regions" as ***region expansion*** and parameter n (defined in Table 1) denotes the total number of region expansions so far. In each iteration, as introduced before, we simultaneously consider expand and evaluate regions with different size indices. However, to keep pStoBO from partitioning into too small regions, as suggested by Valko et al. (2013), we limit the largest index with which a region can be expanded in each iteration as $h_{max}(n)$ (here n is the number of expansions at the beginning of one iteration). An example of $h_{max}(n)$ suggested by Valko et al. (2013) is $\sqrt{n}$ and this is used in our work.

Our proposed approach utilizes two new partition-based stages, the Evaluation Stage and Expansion Stage, to replace the Searching Stage in BO (Algorithm 1) as a new sampling technique. Similar to some of the partition-based algorithms like DIRECT and SOO, the representative point $x_{h,i}$ for region $\mathscr{X}_{h,i}$ is chosen as its center point and the number of sub-regions of each region is odd. Specifically, these subregions are obtained by dividing $\mathscr{X}_{h,i}$ into k parts evenly along its longest edge. Before introducing the three stages in detail in Sections 3.2 to 3.4, an overview of the pStoBO is provided in Section 3.1.

3.1 Overview of pStoBO

As introduced in Algorithm 2, there are three stages in pStoBO, where the Expansion Stage and Evaluation Stage are partition-based to select design points. We use the example in Figure 2 to better illustrate them. In each iteration (the while loop from Line 5 in Algorithm 2), pStoBO searches from the largest region to the smallest region for the current partition configuration (note that larger regions have smaller size indexes h). The Expansion Stage is to choose the promising regions by optimizing the acquisition function over the representative points in set $\mathscr{S}$. After that, the region where this point lies in will be further expanded. In the Evaluation Stage, as introduced before, we decide whether to evaluate some "potentially good" points

Algorithm 2 pStoBO Framework

```
Input: evaluation criterion, allocation rule
1: Initialization:
2: Fit a initial GP model with initial design set $D$
3: Set $\mathscr{L} = \{x_{0,1}\}$ and allocate budget to $x_{0,1}$ with given allocation rule
4: Set $n = 1$, $T = 0$
5: while true do
6:     Set $\mu_{max} = -\infty$, $T = T + 1$
7:     Set $p_1 = \min\{h|\exists j,\ s.t.\ x_{h,j} \in \mathscr{S}\}$, $p_2 = \min\{\max\{h|\exists j,\ s.t.\ x_{h,j} \in \mathscr{S}\}, h_{max}(n)\}$
8:     for $h = p_1 : p_2$ do
9:         Expansion Stage:
10:        Select $(h, j) = \arg\max_{(h,i)\in\mathscr{L}} a(x_{h,i})$
11:        if $a(x_{h,j}) \geq \mu_{max}$ then
12:            Set $\mu_{max} = a(x_{h,j})$, $I_{expansion} = 1$
13:            Add the $k$ sub-regions' representative points to $\mathscr{S}$; Remove $x_{h,j}$ from $\mathscr{S}$
14:            Set $n = n + 1$
15:        else
16:            Set $I_{expansion} = 0$
17:        end if
18:        Evaluation Stage:
19:        if $I_{expansion} = 1$ then
20:            if $a(x_{h,j}) > C(h, x_{h,j})$, $D \cap \{x_{h,j}\} = \emptyset$ then
21:                Set $I_{evaluate} = 1$, $D = D \cup \{x_{h,j}\}$
22:            end if
23:        end if
24:        Allocation Stage:
25:        if $I_{evaluate} = 1$ then
26:            Allocate budget with selected allocation rule and update $\mathscr{Y}$ and $R_\varepsilon$.
27:            $x_t = \arg\min_{x\in D} \bar{y}(x)$, $\bar{y}_t = \bar{y}(\bar{x}_t)$.
28:            Update the stochastic GP model with $D$, $\mathscr{Y}$ and $R_\varepsilon$.
29:        end if
30:    end for
31: end while
```

Table 1: Algorithm parameters list for pStoBO.

Parameter	Definition
k	number of sub-regions for each region
n	current total number of region expansions
$h_{max}(n)$	the largest size index of the regions we explore with n expansions
$\mathscr{S}$	the current center points set
p_1	the smallest size index in the current partition
p_2	the largest size index in the current partition
$I_{expansion}$	indicator of the Expansion Stage
$I_{evaluate}$	indicator of the Evaluation Stage
$C(h,x)$	the evaluation criterion in the evaluation test
x_t	the current best design point
D	current design set
T	the iteration number

to enhance our understanding of the global optimal. Straightforwardly, we can choose the centers of the region just expanded (centers of $\mathscr{X}_{2,3}$ and $\mathscr{X}_{1,3}$ in Figure 2) as these "potentially good" points as they have the best $a(x)$ values among other centers. In this stage, we conduct an evaluation test to each candidate point and only when it passes this test, we will evaluate it. The intuition of this test is to see if this point is a good proxy of the other points in that region (recall that this center is used to build the performance index of the region to represent the performance of points in this region). Since the effectiveness and utility of sampling a point are measured by $a(x)$ in BO, to check if this point is reasonably good compared to other points in the region, we compare the $a(x)$ value at this point with an evaluation criterion C built with the average value of $a(x)$ at a few randomly selected points from the region, which is an estimate of the average $a(x)$ value across the whole region (Line 20 in Algorithm 2). Only if the selected center point has a larger $a(x)$ value than that criterion, we deem it as a "*good point to evaluate*" from that region (i.e., passes the evaluation test), and be evaluated. If not, it may be more prudent to save the budget to expense later on better points from the region. As a result, there can be some partitioned regions whose representative points do not pass the evaluation test, and thus not evaluated. See the example in Figure 1, the center in the third region has a better $a(x)$ value compared with the other two regions. However, most of the points in this region have better $a(x)$ values than this center and it fails the evaluation test. We thus won't evaluate it and save the budget to evaluate other possibly much better points from this region. In Figure 2, '•' and '○' represent center points that have been/not been evaluated, respectively. Finally, in the Allocation Stage, we decide the number of replications assigned to each design point to reduce the noise.

3.2 Expansion Stage

The Expansion Stage is to expand the regions containing the best representative points.

By optimizing the acquisition function over $\mathscr{S}$, the regions containing the selected points are recognized as the current most promising regions. With good potential points in these regions, we further partition these regions into smaller subregions, providing more potential points from these regions to be considered in the next iteration, and hence, facilitating more exploitation in these promising regions.

This partitioning is done to the most promising regions regardless of whether their representative points will be evaluated (in the Evaluation Stage) for the following reasons. Firstly, if the representative points were evaluated, this indicates that the points fall in good promising regions, with good potential points. Hence, we should further partition these regions to add more potential points into $\mathscr{S}$ from these regions to be considered, further facilitating exploitation in these regions. Secondly, if the representative points were not evaluated, the identification (as the best from $\mathscr{S}$) still indicates that they are regarded as the current most promising points and regions. Moreover, the failure of the evaluation test indicates that there exist at least a few better points (w.r.t the acquisition function) than the representative point, and hence, we should further partition this region into smaller subregions whose center points may yield better results.

The partitioning strategy facilitates further exploitation in the most promising regions. To avoid over exploitation (especially at the beginning), the algorithm will by design enable the user to select the evaluation and expansion based on the region sizes. For each iteration, at the Expansion and also the Evaluation Stages of that iteration, regions of different sizes are considered separately. Hence, for each iteration, one region of every region size will be partitioned (although not all will be evaluated). This ensures that each region size will be considered, enabling larger regions, which are less explored, to be explored at every iteration. For example, in Figure 2, we choose one region with size index $h = 1$ and another with size index $h = 2$ to partition.

3.3 Evaluation Stage

The Evaluation Stage is to decide whether to evaluate the best center points.

As introduced before, to build the performance index of each region, we use the center point as a proxy to the points in the region. As a result, the best point in $\mathscr{S}$ indicates that the region it represents is likely to be better than the regions represented by the inferior points in $\mathscr{S}$. However, when the region is large, the center point may not be a good proxy. Hence, to test if it is a reasonably good point to expense simulation efforts to evaluate, we propose an evaluation test in the Evaluation Stage, and only points that pass this evaluation test will be evaluated.

The evaluation test (Line 20 in Algorithm 2) is to compare the acquisition function value at the point just selected, $a(x_{h,j})$, with an evaluation criterion (a threshold) $C(h, x_{h,j})$. This is to determine if a promising point is worthy to evaluate in comparison to the other points in its region. If not, it may be more prudent to save the simulation budget and expense it later on better points in that region. Therefore, the evaluation criteria should reflect potential function values of points in that region. Only the point with $a(x_{h,j})$ larger than the region's criterion will then be evaluated. In pStoBO, the criterion $C(h,x)$ is chosen as the averaged acquisition function value at a few randomly selected points from the considered region, which is an estimate of the averaged $a(x)$ value across the region. In this case, if $a(x_{h,j})$ is smaller than $C(h,x)$, it means that there exists a few better points (w.r.t. $a(x)$) than $x_{h,j}$ and we deem that $x_{h,j}$ is not sufficiently better than other points in the region, and thus $x_{h,j}$ is not evaluated. The budget will then be saved to evaluate better points from this region.

3.4 Allocation Stage

The Allocation Stage is to add simulation replications to existing design points to better fit the model and recognize the best from them.

Once a point is decided to be evaluated, we add it to the current design set D. pStoBO then adds simulation replications to the existing design points in each iteration. The purpose is to reduce the noise of the response estimates at the design points, which are used to build the GP model that guides the search. Moreover, as pStoBO returns the design point with the best empirical mean as its recommendation, adding replications to these points helps to more accurately identify this recommendation more accurately. As the main focus of this paper is to improve the future points selection, we do not put restrictions on the allocation schemes used in pStoBO. In the numerical study, we apply a simple equal allocation scheme to test the algorithm.

4 NUMERICAL TEST

In this section, we illustrate with a 2d example how pStoBO can address the issues of BO mentioned in Section 1. This example is taken from (Xu et al. 2010).

$$z(x) = 10\frac{\sin^6(5\pi x_1)}{2^{((100x_1-90)/50)^2}} + 10\frac{\sin^6(5\pi x_2)}{2^{((100x_2-90)/50)^2}}.$$

It is originally a maximization problem and here we minimize the negative z over $[0,1]^2$ instead. It has 25 local optima with the global optimum $z(0.9,0.9) = -20$ and two second-best optima $z(0.7,0.9) = -18.95$ and $z(0.9,0.7) = -18.95$. Sun et al. (2014) added a small noise with zero mean and variance 1 to this function and here, we add a larger noise with variance 10 for illustration.

To test pStoBO, we build a specific algorithm, pStoBO-EI. This algorithm follows Algorithm 2 with EI as the acquisition function. In pStoBO-EI, the evaluation criterion in the Evaluation Stage is chosen as:

$$C(h,x) = \frac{1}{10}\sum_{i=1}^{10} a(x_i),$$

where $x_1, ..x_{10}$ are randomly selected points from the region x falls in. The allocation rule applied here is the equal allocation rule, i.e., when $|D| = \tilde{t}$ (total number of design points), each design point is assigned with $0.25\tilde{t}$ replications.

Another two algorithms are chosen for comparison. We select eTSSO as a representative traditional stochastic BO algorithm to compare with, as, to the best of our knowledge, this is the only one with convergence guarantee for stochastic simulations. For eTSSO, we follow the recommendation of Jalali et al. (2017) to discretize $[0,1]^2$ with 1000 points chosen by the Latin Hypercube sampling technique. In addition, we also include StoSOO in the test, which is a pure partition-based algorithm that also simultaneously searches regions with multiple sizes. We note that StoSOO is a direct optimization algorithm in that no metamodel is used to enhance the search. For each of the three algorithms, we repeat the optimization process for 30 macro-replications and each macro-replication has a total budget with 10000 function evaluations. After averaging over the 30 macro-replications, we plot the gap between the current best input and the true best input w.r.t the number of function evaluations in Figure 3, and the best function value found by these algorithms at the end of the search process in Figure 4. In Figure 3, we indicate the gap between

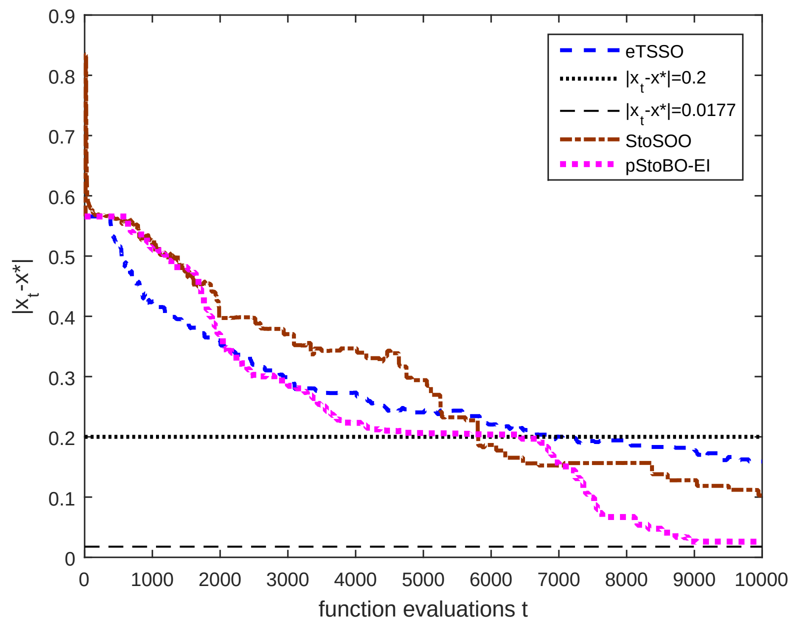

Figure 3: $|x_t - x^*|$ v.s. function evaluations.

the second optimum and the global optimum 0.2. Moreover, as in eTSSO, the space is discretized with 1000 sampling points. We also include the minimum distance between a point in the discretization with the global optimal averaged over 30 macro-replications, 0.0177. Therefore, this is the best eTSSO can get with

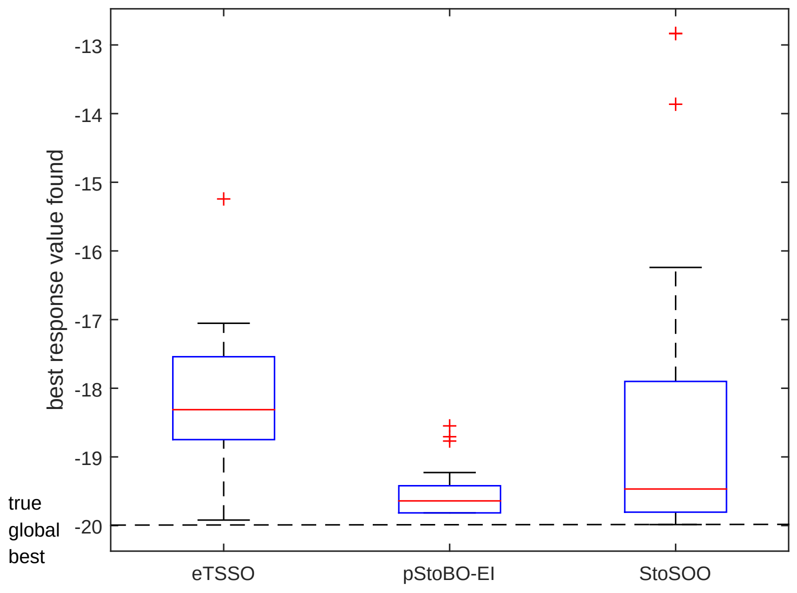

Figure 4: The function values of the best points found by the three algorithms.

the discretization. We observe that, pStoBO-EI and eTSSO perform almost equally when $t < 7000$. After that, pStoBO-EI becomes better. When t is about 6000, the $|x_t - x^*|$ line for pStoBO-EI decreases slower and more time is spent near 0.2, i.e., it searches the second-best regions for a while. However, owing to its expansion and evaluation mechanism, pStoBO-EI explores more regions and successfully goes to the global optimal regions. On the other hand, eTSSO seems stuck in suboptimal regions, even until the end. In summary, due to the expansion and evaluation mechanism of pStoBO, pStoBO-EI is better than eTSSO in exploring more regions and finding the true global optimal faster. In addition, we observe that without the help of the GP model, StoSOO takes a longer time to find the optimal regions (when $t < 6000$), which highlights the efficiency of BO algorithms. As the budget increases, the StoSOO is able to converge to the global promising regions due to its ability to search regions with multiple sizes, as does pStoBO, and performs almost the same as eTSSO but is still clearly worse than pStoBO-EI.

Comparing the final optimal values returned by these algorithms (Figure 3), we find that through further statistical tests, the function values of the best solutions found by pStoBO-EI are statistically better than eTSSO and StoSOO at a significance level of 0.05 , These observations are similar to those from Figure 3. We also observe that the variability of the results found by StoSOO is typically much larger than the remaining ones. This can be due to the lack of the guidance of the GP model in the search and without this model, StoSOO does not find the optimal regions quickly and the search of StoSOO is somewhat more ‘random’ compared with the others. As a result, StoSOO and can return much worse results from sub-optimal regions, increasing the variability. Moreover, the variability of pStoBO-EI is the smallest, indicating that this algorithm is more efficient in finding the optimal regions within constrained time and more consistent in its performance.

From this example, we can see that pStoBO can address the two issues highlighted for BO. First, it does not rely on a discretization to solve the acquisition function optimization problem. As seen in Figure 3, the best eTSSO is possible to find within the discretized points ($|x_t - x^*| = 0.0177$) has already been achieved by pStoBO-EI at the end (note that with given budget, eTSSO didn’t find this best point). Second, pStoBO with the new Expansion and Evaluation stages can help the algorithm explore more promising regions. In this example. pStoBO jumps out of local optimal regions faster than eTSSO and converges to the global optimal.

5 CONCLUSION

In this paper, we propose a new partition-based stochastic BO algorithm, pStoBO. This new algorithm optimizes the acquisition function on a finite set to reduce the computational burden and considers partitioning multiple regions with different sizes to reduce over-exploitation. With a numerical test, we observe that pStoBO can perform much better than the traditional stochastic BO algorithms for functions with multiple local optimal points.

We list several future research directions for pStoBO. First, the current pStoBO is general in terms of the acquisition functions as well as the allocation schemes that can be adopted. The impact of these functions and schemes on the algorithm performance can be further investigated. Second, the asymptotic convergence of pStoBO algorithm as well as its finite-time performance should be studied. Third, to better understand the advantages of pStoBO, a more thorough comparative study with other standard BO algorithms such as KG (Wang et al. 2020), Thompson Sampling and GP-UCB (Srinivas et al. 2010) is required. These algorithms are mainly designed for deterministic problems or problems with homogeneous noise and have been shown to be very efficient. However, adaption will be required to apply them for the stochastic simulations considered in this work.

REFERENCES

Bardenet, R., and B. Kégl. 2010. "Surrogating the Surrogate: Accelerating Gaussian-Process-Based Global Optimization with A Mixture Cross-Entropy Algorithm". In *27th International Conference on Machine Learning (ICML 2010)*, edited by J. Fürnkranz and T. Joachims, 55–62. Haifa, Israel: Omnipress.

Bergstra, J., and Y. Bengio. 2012. "Random Search for Hyper-Parameter Optimization". *Journal of Machine Learning Research* 13(Feb):281–305.

Frazier, P. I. 2018. "A Tutorial on Bayesian Optimization". *arXiv preprint arXiv:1807.02811*. https://arxiv.org/abs/1807.02811, accessed 7^{th} September.

Hong, L. J., and B. L. Nelson. 2009. "A brief introduction to optimization via simulation". In *Proceedings of the 2009 Winter Simulation Conference (WSC)*, edited by A. Dunkin, R. Ingalls, E. Yücesan, M. Rossetti, R. Hill, and B. Johansson, 75–85. Austin, TX: Institute of Electrical and Electronics Engineers, Inc.

Jalali, H., I. Van Nieuwenhuyse, and V. Picheny. 2017. "Comparison of Kriging-Based Algorithms for Simulation Optimization with Heterogeneous Noise". *European Journal of Operational Research* 261(1):279–301.

Jones, D. R., C. D. Perttunen, and B. E. Stuckman. 1993. "Lipschitzian Optimization without the Lipschitz Constant". *Journal of optimization Theory and Applications* 79(1):157–181.

Kleinberg, R., A. Slivkins, and E. Upfal. 2008. "Multi-Armed Bandits in Metric Spaces". In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, edited by R. Ladner, 681–690. Victoria, British Columbia: Association for Computing Machinery.

Munos, R. 2011. "Optimistic Optimization of A Deterministic Function Without the Knowledge of Its Smoothness". In *Advances in neural information processing systems*, edited by J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, 783–791. Granada Spain: Curran Associates, Inc.

Pasupathy, R., and S. Ghosh. 2013. "Simulation Optimization: A Concise Overview and Implementation Guide". In *Theory Driven by Influential Applications*, 122–150. Institute for Operations Research and the Management Sciences.

Pedrielli, G., S. Wang, and S. H. Ng. 2020. "An Extended Two-Stage Sequential Optimization Approach: Properties and Performance". *European Journal of Operational Research* 287(3):929–945.

Pinter, J. 1997. "Global Optimization in Action Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications". *Journal of the Operational Research Society* 79(1):157–181.

Quan, N., J. Yin, S. H. Ng, and L. H. Lee. 2013. "Simulation Optimization via Kriging: A Sequential Search Using Expected Improvement with Computing Budget Constraints". *IIE Transactions* 45(7):763–780.

Ranjan, P., R. Haynes, and R. Karsten. 2011. "A Computationally Stable Approach to Gaussian Process Interpolation of Deterministic Computer Simulation Data". *Technometrics* 53(4):366–378.

Shahriari, B., K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas. 2015. "Taking the Human Out of the Loop: A Review of Bayesian Optimization". *Proceedings of the IEEE* 104(1):148–175.

Shi, L., and S. Ólafsson. 2000. "Nested Partitions Method for Global Optimization". *Operations Research* 48(3):390–407.

Snoek, J., H. Larochelle, and R. P. Adams. 2012. “Practical Bayesian Optimization of Machine Learning Algorithms”. In *Advances in Neural Information Processing Systems 25*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, 2951–2959. Lake Tahoe: Curran Associates, Inc.

Srinivas, N., A. Krause, S. Kakade, and M. Seeger. 2010, June. “Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design”. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, edited by J. Fürnkranz and T. Joachims, 1015–1022. Haifa, Israel: Omnipress.

Sun, L., L. J. Hong, and Z. Hu. 2014. “Balancing Exploitation and Exploration in Discrete Optimization via Simulation through A Gaussian Process-Based Search”. *Operations Research* 62(6):1416–1438.

Valko, M., A. Carpentier, and R. Munos. 2013, 17–19 Jun. “Stochastic Simultaneous Optimistic Optimization”. Volume 28 of *Proceedings of Machine Learning Research*, 19–27. Atlanta, Georgia, USA: Proceedings of Machine Learning Research.

Wang, J., S. C. Clark, E. Liu, and P. I. Frazier. 2020. “Parallel Bayesian Global Optimization of Expensive Functions”. *Operations Research (published online)*.

Wang, Z., C. Li, S. Jegelka, and P. Kohli. 2017, 06–11 Aug. “Batched High-dimensional Bayesian Optimization via Structural Kernel Learning”. Volume 70 of *Proceedings of Machine Learning Research*, 3656–3664. International Convention Centre, Sydney, Australia: Proceedings of Machine Learning Research.

Wang, Z., B. Shakibi, L. Jin, and N. Freitas. 2014, 22–25 Apr. “Bayesian Multi-Scale Optimistic Optimization”. Volume 33 of *Proceedings of Machine Learning Research*, 1005–1014. Reykjavik, Iceland: Proceedings of Machine Learning Research.

Xu, J., B. L. Nelson, and J. Hong. 2010. “Industrial Strength COMPASS: A Comprehensive Algorithm and Software for Optimization via Simulation”. *ACM Transactions on Modeling and Computer Simulation (TOMACS)* 20(1):3.

AUTHOR BIOGRAPHIES

SONGHAO WANG is a research fellow in the Department of Industrial Systems Engineering and Management at National University of Singapore. He received his Ph.D. degree from the same department in 2020 and his B.S. in Modern Mechanics from University of Science and Technology of China in 2015. His research interests include simulation optimization, Bayesian optimization, experiment design and metamodeling. His email address is wshustc1993@gmail.com.

SZU HUI NG is an Associate Professor in the Department of Industrial Systems Engineering and Management at the National University of Singapore. She holds B.S., M.S., and Ph.D. degrees in Industrial and Operations Engineering from the University of Michigan. Her research interests include computer simulation modeling and analysis, design of experiments, and quality and reliability engineering. She is a member of IEEE and INFORMS and a senior member of IIE. Her email address is isensh@nus.edu.sg and her web page is https://www.eng.nus.edu.sg/isem/staff/ng-szu-hui/ .