A Framework for Multi-fidelity Modeling in Global Optimization Approaches

Zelda B. Zabinsky[1(✉)], Giulia Pedrielli[2], and Hao Huang[3]

[1] University of Washington, Seattle, WA 98195, USA
zelda@u.washington.edu
[2] Arizona State University, Tempe, AZ, USA
Giulia.Pedrielli@asu.edu
[3] Yuan Ze University, Taoyuan City, Taiwan
haohuang@saturn.yzu.edu.tw

Abstract. Optimization of complex systems often involves running a detailed simulation model that requires large computational time per function evaluation. Many methods have been researched to use a few detailed, high-fidelity, function evaluations to construct a low-fidelity model, or surrogate, including Kriging, Gaussian processes, response surface approximation, and meta-modeling. We present a framework for global optimization of a high-fidelity model that takes advantage of low-fidelity models by iteratively evaluating the low-fidelity model and providing a mechanism to decide when and where to evaluate the high-fidelity model. This is achieved by sequentially refining the prediction of the computationally expensive high-fidelity model based on observed values in both high- and low-fidelity. The proposed multi-fidelity algorithm combines Probabilistic Branch and Bound, that uses a partitioning scheme to estimate subregions with near-optimal performance, with Gaussian processes, that provide predictive capability for the high-fidelity function. The output of the multi-fidelity algorithm is a set of subregions that approximates a target level set of best solutions in the feasible region. We present the algorithm for the first time and an analysis that characterizes the finite-time performance in terms of incorrect elimination of subregions of the solution space.

Keywords: Global optimization · Multi-fidelity models · Meta-models · Probabilistic Branch and Bound · Gaussian processes

1 Introduction

Complex systems are often represented and evaluated by means of a detailed, high-fidelity simulation model that requires significant computational time to execute. Since the high-fidelity model is time consuming to run, a low-fidelity

This work has been supported in part by the National Science Foundation, Grant CMMI-1632793.

G. Nicosia et al. (Eds.): LOD 2019, LNCS 11943, pp. 335–346, 2019.
https://doi.org/10.1007/978-3-030-37599-7_28

model is often constructed that takes far less computational time to run, but whose output is affected by error. For example, in engineering design, the high-fidelity model may involve a finite element analysis with a fine grid, whereas a low-fidelity version may use a coarse grid in the finite element analysis. The coarse grid is faster to execute, but provides less accurate performance metrics of the design. As another example, in manufacturing, a high-fidelity model may include a detailed discrete-event simulation with a complicated network of queues, whereas a low-fidelity version may be an analytical Markov chain model that is constructed by making simplifying assumptions.

An interesting and relevant question is how to make use of low-fidelity models to increase the likelihood of determining a solution, or set of solutions, that achieve good high-fidelity performance. The importance of the problem is well documented by the rich literature on the topic across different areas of engineering and computer science [4,7,11,14,16,20]. Most of the approaches can be brought back to the large category of Bayesian optimization methods.

Bayesian optimization is a well-established approach [12,13,17] to optimizing a complex system described by a potentially multi-modal, non-differentiable, black-box objective function such as the high-fidelity model we refer to. A Bayesian method starts with an a priori distribution, commonly a Gaussian distribution with a special covariance matrix, that represents the unknown objective function. Given several function evaluations of the objective function, the posterior (conditional) distribution of the objective function is updated. A tutorial on Bayesian optimization is given in [2]. We follow the basic procedure of updating the spatial covariance matrix using the observed function values.

There are several alternatives to Gaussian distributions to describe the objective function. In particular, radial basis functions have shown a remarkable success [19], additive Gaussian Processes that assume a dependency structure among the co-variates exists and can be learned [3,8]. Moreover, embeddings have been investigated in order to tackle the problem of scalability of model-based approaches [9,18].

Our approach is to use the statistical power of Gaussian distributions to relate the low-fidelity model to the high-fidelity model, and thus use fewer high-fidelity function evaluations. Several other papers have used a combination of low- and high-fidelity models, however, our approach is unique in that it embeds the Gaussian process into Probabilistic Branch and Bound [5,23], which provides a statistical confidence interval on how close the solutions obtained in the algorithm are to the global optimum.

In the literature relevant to this work, Xu et al. [21] proposed MO^{2}TOS (Multi-Fidelity Optimization with Ordinal Transformation and Optimal Sampling) that relies on the concept of Ordinal Transformation (OT). OT is a mapping $\mathbb{X} \rightarrow \mathcal{H}$, where $\mathbb{X}$ is a d-dimensional discrete space and $\mathcal{H}$ is a one-dimensional rank space constructed by associating to each point of $\mathbb{X}$, the rank computed according to the evaluation returned by the low-fidelity model. This mapping, as defined by the authors, can be applied to any finite space $\mathbb{X}$. Once the mapped space is computed, the solutions are grouped in subsets defined using

$\mathcal{H}$, and sampled according to the Optimal Sampling (OS) scheme. The theoretical analysis performed by the authors provides properties that a low-fidelity model should have to guarantee an improved performance of the proposed algorithm with respect to a benchmark version not using any low-fidelity information. Xu et al. [22] further extends the previous contribution by proposing an innovative optimal sampling methodology that maximizes the estimated probability of selecting the best solution. In [6], the authors extend the framework to continuous optimization by proposing a novel *additive model* that captures the relationship between the high- and low-fidelity functions and trying to sample mostly with the low-fidelity model. A potential drawback of this approach is that it assumes that a unique additive model can be used that fits the function across the entire solution space. In the direction to consider different behaviors, in [10], a multi-fidelity algorithm for global optimization was introduced that used Probabilistic Branch and Bound (PBnB) [5,23] to approximate a level set for the low-fidelity model, and under assumptions of consistency between the low-fidelity and high-fidelity models, the paper showed an increased probability of sampling high-quality solutions within a low-fidelity level set.

In this paper, we relax the assumption in [10] that the low-fidelity level set and high-fidelity level set need to intersect and the assumption of a unique model in [6], by combining the work in [10] with [6]. The new approach derives several predictive models of the original high-fidelity function using both high- and low-fidelity evaluations, in the subregions identified by PBnB. Specifically, when the predictive model(s) fails a certification test, it indicates that either more high-fidelity observations are needed, or that the subregion should be branched into smaller subregions. When the predictive model is good, we use it to decide which subregion should be explored more to discover global optima. A theoretical analysis of this new algorithm provides a probability of correctly focusing on good subregions on any iteration k, providing new finite-time results.

2 Framework

We consider an optimization problem with a high-fidelity black-box function f_H

$$\min_x \ f_H(x) \tag{1}$$

$$\text{subject to } x \in S$$

where $S \subset \Re^d$ is the feasible region, and $f_H : S \to \Re$. We also consider a low-fidelity model, $f_L : S \to \Re$, and assume that the computation time to evaluate $f_L(x)$ is much less than that to evaluate $f_H(x)$.

We are interested in determining near-optimal solutions in a target set that consists of the best δ-quantile of solutions, which can be defined as a level set bounded by a quantile $y_H(\delta, S)$,

$$y_H(\delta, S) = \arg\min_y \{P(f_H(X) \leq y | x \in S) \geq \delta\}, \text{ for } 0 < \delta < 1, \tag{2}$$

where X is uniformly distributed over S. Using $y_H(\delta, S)$, the target level set is defined as

$$L_H(\delta, S) = \{x \in S : f_H(x) \leq y_H(\delta, S)\}, \text{ for } 0 < \delta < 1. \tag{3}$$

Similarly, we define $y_L(\delta, S)$ and $L_L(\delta, S)$ as quantile and target set associated with the low-fidelity model, respectively. We note that for quantile level δ, $\delta = \frac{\nu(L_H(\delta,S))}{\nu(S)} = \frac{\nu(L_L(\delta,S))}{\nu(S)}$, where $\nu(\cdot)$ is the d-dimensional volume (i.e., Lebesgue measure) of a set.

The goal of the algorithm introduced in this paper is to approximate the target level set $L_H(\delta, S)$ using relatively few high-fidelity function evaluations and allow many more low-fidelity function evaluations.

2.1 Using Statistical Learning to Bridge High and Low Fidelity Models

As is common in Bayesian optimization [2], we use Gaussian processes as a framework to provide a statistical relationship between the low-fidelity function and the high-fidelity function, so that we can use the low-fidelity function evaluations to predict improving regions and focus the execution of the high-fidelity function. We next briefly summarize a Gaussian process modeling framework and introduce notation.

Within the Gaussian process modeling framework, a function $f(x)$ is interpreted as a realization from an infinite family of random functions, i.e., a stationary Gaussian process $Y(x)$. The statistical model estimating the behavior of the unknown function Y is characterized in terms of an optimal predictor $\hat{y}(x)$ and predictive error $s^2(x)$ based on a set of k observations, $x_i \in S$, with function evaluations $f(x_i)$, for $i = 1, \ldots, k$. We let $\mathbb{X}$ represent the set of observed locations $\mathbb{X} = \{x_1, \ldots, x_k\}$, and let $\bar{f}$ be a k-vector of the observed function evaluations $f(x_i)$, for $i = 1, \ldots, k$.

The statistical model of Y, conditional on the sampled observations, is:

$$Y(x) \sim \mathcal{N}\left(\hat{y}(x), s^2(x)\right) \tag{4}$$

for any $x \in S$ that has not yet been observed,[1] such that:

$$\hat{y}(x) := \hat{\mu} + \mathbf{c}^T \mathbf{K}^{-1}\left(\bar{f} - \hat{\mu}1\right) \quad \text{and} \quad \hat{\mu} := \frac{1^T \mathbf{K}^{-1} \bar{f}}{1^T \mathbf{K}^{-1} 1} \tag{5}$$

and

$$s^2(x) := \tau^2 \left(1 - \mathbf{c}^T \mathbf{K}^{-1} \mathbf{c} + \frac{\left(1 - 1^T \mathbf{K}^{-1} \mathbf{c}\right)^2}{1^T \mathbf{K}^{-1} 1}\right). \tag{6}$$

Here, 1 is a k-vector having all elements equal to 1, $\mathbf{K}$ is a $k \times k$ spatial variance-covariance matrix parameterized over ϕ, and ϕ is a d-dimensional vector of weighting parameters, obtained upon the sampling of k points.

[1] For an observed point x_i, $\hat{y}(x_i) = f(x_i)$ and $s^2(x_i) = 0$.

The scale correlation coefficients in the d-dimensional vector ϕ are sensitivity parameters that control how fast the correlation decays with the *distance* between two observed points (x_i, x_j). Given k observations, the $d+1$ parameters (τ and $\phi_i, i = 1, \ldots, d$) are estimated using maximum likelihood [15].

The spatial variance-covariance matrix often appears in its exponential form, where the (i, j)th element of $\mathbf{K}$ is

$$\mathbf{K}\left(\mathbb{X}, \phi\right)_{i,j} = \exp\left[-\sum_{l=1}^{d} \phi_l \left(x_{i,l} - x_{j,l}\right)^t\right] \tag{7}$$

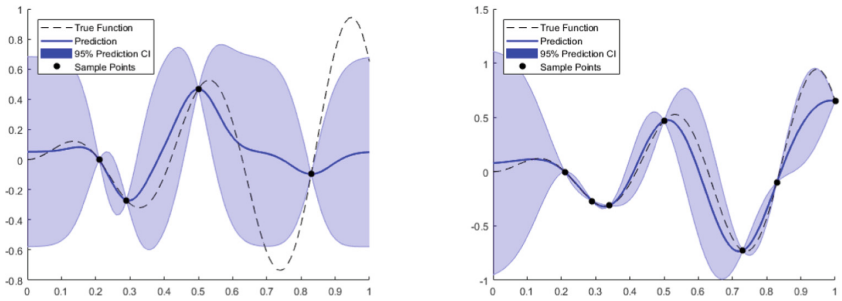
and $x_{i,l}$ is the lth element of x_i. The parameter t controls the smoothness of the response. When $t = 1$, Eq. (7) is known as the exponential correlation function, and when $t = 2$ it is known as the Gaussian correlation function [15]. In this paper, we adopt the Gaussian correlation function, with $t = 2$.

The k-vector $\mathbf{c}$ contains the spatial correlation between the prediction point $x \in S$ and the sampled locations $x_j,\ j = 1, \ldots, k$, where the jth element of $\mathbf{c}$ uses the lth element of the prediction point x_l and the sampled location $x_{j,l}$:

$$\mathbf{c}\left(x, \mathbb{X}, \phi\right)_j = \exp\left[-\sum_{l=1}^{d} \phi_l \left(x_l - x_{j,l}\right)^t\right] \tag{8}$$

where, again, we let $t = 2$.

Given k observations, it is straight-forward to build a Gaussian process model $Y(x)$ with predictor $\hat{y}\left(x\right)$ as in (5) and predictive error $s^2(x)$ as in (6) (Fig. 1).



(a) Illustration using four observations. (b) Illustration using seven observations.

Fig. 1. Illustration of a one-dimensional function $f(x)$ with a predictor function $\hat{y}(x)$ and predictive error $s^2(x)$, using four observations in (a) and seven observations in (b).

Our approach to determine a relationship between the high-fidelity function f_H and the low-fidelity function f_L is to build several Gaussian process models based on high- and low-fidelity function observations. Following the approach in [6], we let $\mathbb{X}_H$ represent the set of observations that are sampled in S and only

evaluated with high-fidelity (computationally expensive) function evaluations; let $\mathbb{X}_L$ represent the set of observations which are sampled and only evaluated with the low-fidelity (very fast computation) function; and let $\mathbb{X}_B$ be the set of points independently sampled that are evaluated with both high- and low-fidelity functions. The observations in $\mathbb{X}_B$ are used to estimate the bias using both cheap and expensive function evaluations, as a type of training set.

Under a Gaussian process multi-fidelity perspective, consider the following high-fidelity, low-fidelity, and bias statistical models used in the algorithm:

$$\begin{aligned} Y_{\cdot}(x) &\sim \mathcal{N}\left(\hat{y}_{\cdot}(x), s_{\cdot}^2(x)\right) \\ \hat{y}_{\cdot}(x) &:= \hat{\mu}_{\cdot} + \mathbf{c}_{\cdot}^T \mathbf{K}_{\cdot}^{-1}\left(\bar{f}_{\cdot} - \hat{\mu}_{\cdot} 1\right) \\ s_{\cdot}^2(x) &:= \tau_{\cdot}^2 \left(1 - \mathbf{c}_{\cdot}^T \mathbf{K}_{\cdot}^{-1} \mathbf{c}_{\cdot} + \frac{\left(1 - 1^T \mathbf{K}_{\cdot}^{-1} \mathbf{c}_{\cdot}\right)^2}{1^T \mathbf{K}_{\cdot}^{-1} 1}\right) \end{aligned} \tag{9}$$

where the subscript $\cdot$ is replaced by either H, L, or B indicating which function evaluations are used in the statistical model (high-fidelity, low- fidelity or bias, respectively). Specifically, the high-fidelity model uses $\bar{f}_H(x_i) = f_H(x_i)$ for $x_i \in \mathbb{X}_H$, the low-fidelity model uses $\bar{f}_L(x_i) = f_L(x_i)$ for $x_i \in \mathbb{X}_L$, and the bias model uses $\bar{f}_B(x_i) = f_H(x_i) - f_L(x_i)$ for $x_i \in \mathbb{X}_B$.

The statistical relationship between the high- and low-fidelity models is:

$$Y_{H|L}(x) = Y_L(x) + Y_B(x) \tag{10}$$

and use the set of observations $\mathbb{X}_B$ to reconstruct the prediction of the low-fidelity model conditional on the high-fidelity simulations performed, namely,

$$Y_{L|H}(x) = Y_H(x) - Y_B(x) \tag{11}$$

for $x \in S$. This model will be at the basis of the certificate that we use to decide whether or not evaluations in high-fidelity are required.

Our objective is to embed this statistical modeling into the partitioning logic of PBnB to reduce the number of high-fidelity function evaluations and focus them on subregions of interest.

2.2 Algorithm Details and Behavior

The main idea for the algorithm is to make a few high-fidelity function evaluations and more low-fidelity function evaluations, construct the statistical models, and use them to focus the location of more high-fidelity observations to eventually obtain subregions that are likely to contain the target set $L_H(\delta, S)$. We iteratively partition subregions of S, take additional high- and low-fidelity function evaluations, refine the statistical models, and subdivide the subregions maintaining statistical confidence that they contain the target set.

In the algorithm, we test whether $Y_{L|H}$ generates a good predictor $\hat{y}_L$ on a subregion of S. The hypothesis is that the model in (11) generates an accurate prediction of the low-fidelity response. Such a test is relevant to the search since

the model in (11) is built using also the high-fidelity observations. The rationale is that, if $Y_{L|H}$ generates a good predictor for f_L, then $Y_{H|L}$ will also give good performance in predicting f_H, the function we are interested in optimizing. However, testing over the model $Y_{H|L}$ would require many high-fidelity function evaluations, which is undesirable due to the high evaluation cost.

When we have a good predictor over a subregion, we can use the predictor to decide if the subregion is likely or unlikely to contain the δ-quantile target set. If it is likely to contain the target set, we refine the partition of the subregion to focus the use of the expensive high-fidelity function evaluations. If the subregion is determined to be unlikely to contain the target set, we conclude that there is enough statistical evidence (at 1-α) to refrain from making more high-fidelity evaluations in that subregion. We use the framework of PBnB, integrated with the statistical models, to make decisions whether to evaluate using a high or low precision model and to guide the sampling locations.

Overview of Multi-fidelity Algorithm:

Step 0. *Initialize*: Set confidence level $\alpha, 0 < \alpha < 1$, target quantile $\delta, 0 < \delta < 1$, tolerated volume $\epsilon, \epsilon > 0$, and partitioning number B integer valued. Initialize the set of current subregions $\Sigma_1 = \{\sigma_1, \ldots, \sigma_B\}$, with $\cup_{j=1}^{B} \sigma_j = S$ and $\cap_{j=1}^{B} \sigma_j = \emptyset$, the number of subregions $J = B$, and the iteration counter $k = 1$.

Step 1. *Evaluate functions and build statistical models*:
Generate additional sample points in each subregion σ_j for all $j = 1, \ldots, J$, and $\sigma_j \in \Sigma_k$ so there is at least one point in each subregion for high-fidelity evaluations, for low-fidelity evaluations, and for both high- and low-fidelity function evaluations. Update the sets $\mathbb{X}_{H_k}$, $\mathbb{X}_{L_k}$, and $\mathbb{X}_{B_k}$ for those points with high-, low-, and both high- and low-fidelity function evaluations. While the number of samples is important for cross validating the statistical models, the idea is that the number of high-fidelity evaluations is much less than the number of low-fidelity evaluations, $|\mathbb{X}_H| \approx |\mathbb{X}_B| << |\mathbb{X}_L|$, where $|\cdot|$ represents the cardinality of a set. In the numerical results, we let $|\mathbb{X}_L| \geq 10Bd$ as is common when constructing Gaussian processes. However, this may be too large for computationally expensive high-fidelity function evaluations, so we just ensure that $|\mathbb{X}_H \cap \sigma_j| \geq 1$ and $|\mathbb{X}_B \cap \sigma_j| \geq 1$ for $j = 1, \ldots, J$.

Update $\bar{f}_H(x_i) = f_H(x_i)$ for $x_i \in \mathbb{X}_{H_k}$, $\bar{f}_L(x_i) = f_L(x_i)$ for $x_i \in \mathbb{X}_{L_k}$, and $\bar{f}_B(x_i) = f_H(x_i) - f_L(x_i)$ for $x_i \in \mathbb{X}_{B_k}$.

Given the function evaluations, build cross-validated models using the available observations for the two fidelities and the bias as well as the conditional densities: $Y_H, Y_L, Y_B, Y_{L|H}$, and $Y_{H|L}$.

Step 2. *Test predictive capability of low-fidelity model*:
As in PBnB, partition each subregion in the current set of subregions Σ_k into B new subregions, denoted $\sigma_1, \ldots, \sigma_J$, where $J = |\Sigma_k|B$ and $\cap_{j=1}^{J} \sigma_j = \emptyset$. Update Σ_k with the newly branched subregions.

Only using the low-fidelity samples, build J low-fidelity Gaussian process models, denoted $Y_{L,j}$, for each newly created subregion σ_j, $j = 1, \ldots, J$. Note that there may be no points in $\mathbb{X}_{L_k}$ that are also in a new σ_j, in which case,

more points may be generated and evaluated with f_L. Also, if there are no points in $\mathbb{X}_{H_k}$ or $\mathbb{X}_{B_k}$ that are also in σ_j, use an extrapolation procedure to build the subregion-specific models $Y_{H,j}$ and $Y_{B,j}$.

For each subregion σ_j, without evaluating the high-fidelity function at any new points, use $Y_{L,j}$, $Y_{H,j}$ and $Y_{B,j}$ to build subregion-specific models $Y_{L|H,j}$ and $Y_{H|L,j}$ for $j = 1, \ldots, J$.

For each subregion σ_j, test the hypothesis that the Gaussian process $Y_{L|H,j}$ generates an accurate prediction of the low-fidelity response $Y_{L,j}$. We propose the following test to derive a low-fidelity certificate:

$$\mathrm{H}_0 : Y_{L|H,j}(x) \sim \mathcal{N}\left(\hat{y}_{L|H,j}(x), s^2_{L|H,j}(x)\right) \quad \text{and} \quad Q(x) = \frac{f_L(x) - \hat{y}_{L|H,j}(x)}{s_{L|H,j}(x)}$$

for $x \in \sigma_j$ and $j = 1, \ldots, J$. We compute the test statistic Q on a grid of points in σ_j to estimate the quantile and compare it to a standard normal value $z_{\alpha/2}$, assuming the normality assumption in the null hypothesis H_0.

If the test fails in a subregion, then, with $1 - \alpha$ confidence, we reject the null hypothesis and conclude that the predictor is providing poor performance in that subregion. In this case, we need to make more high-fidelity evaluations in the subregions that fail, so the algorithm goes to Step 1.

If the test does not fail, then proceed to Step 3 with $Y_{H|L,j}$ for each subregion σ_j, $j = 1, \ldots, J$.

Step 3. *Use $Y_{H|L,j}$ with additional samples to make pruning decision*:

For each subregion σ_j in Σ_k, $j = 1, \ldots, J$, uniformly and independently sample $N_k = \left\lceil \frac{\ln \alpha_k}{\ln(1 - \frac{\epsilon}{v(S)})} \right\rceil$ points $\tilde{x}_{\sigma_j,n}$ for $n = 1, \ldots, N_k$, and evaluate them with $\hat{y}_{H|L,j}$. Notice that this is relatively fast to compute and does not require any high-fidelity function evaluations. Within each subregion σ_j, order these sampled points by their predicted values $\hat{y}_{H|L,j}$ and denote them $\tilde{x}_{\sigma_j,(1)}, \ldots, \tilde{x}_{\sigma_j,(N_k)}$, where

$$\hat{y}_{H|L,j}(\tilde{x}_{\sigma_j,(1)}) \le \hat{y}_{H|L,j}(\tilde{x}_{\sigma_j,(2)}) \le \cdots \le \hat{y}_{H|L,j}(\tilde{x}_{\sigma_j,(N_k)}).$$

Also order *all* of the points $\tilde{x}_{\sigma_j,n}$ for $j = 1, \ldots, J$ and $n = 1, \ldots, N_k$ by their predicted values. Since the predicted value function is subregion specific, for notational purposes, let $\hat{y}^k_{H|L}(\tilde{x}) = \hat{y}_{H|L,j}(\tilde{x})$ for $\tilde{x} \in \sigma_j$. Then, we can use this function $\hat{y}^k_{H|L}$ to order the points $\tilde{x}_{\sigma_j,n}$ and denote them $\tilde{x}_{\Sigma_k,(1)}, \ldots, \tilde{x}_{\Sigma_k,(JN_k)}$, where

$$\hat{y}^k_{H|L}(\tilde{x}_{\Sigma_k,(1)}) \le \hat{y}^k_{H|L}(\tilde{x}_{\Sigma_k,(2)}) \le \cdots \le \hat{y}^k_{H|L}(\tilde{x}_{\Sigma_k,(JN_k)}).$$

Perform the following comparison, the pruning test, for each subregion σ_j,

$$\hat{y}_{H|L,j}\left(\tilde{x}_{\sigma_j,(1)}\right) - z_{\frac{\alpha_k}{2}} s_{H|L,j}(\tilde{x}_{\sigma_j,(1)}) > \hat{y}^k_{H|L}\left(\tilde{x}_{\Sigma_k,(s)}\right) + z_{\frac{\alpha_k}{2}} s^k_{H|L}(x_{\Sigma_k,(s)}) \tag{12}$$

where s satisfies,

$$\min s : \sum_{i=0}^{s-1} \binom{N_k}{i} (\delta_k)^i (1 - \delta_k)^{N_k - i} \ge 1 - \frac{\alpha_k}{2}. \tag{13}$$

The test in (12) compares the best value in a subregion with the s-best value overall, accounting for an error term with confidence α_k.

If (12) is satisfied, prune σ_j. If (12) is not satisfied, keep σ_j. Update Σ_{k+1} with the subregions in Σ_k that have not been pruned.

Update $\alpha_{k+1} = \alpha_k / B$ and

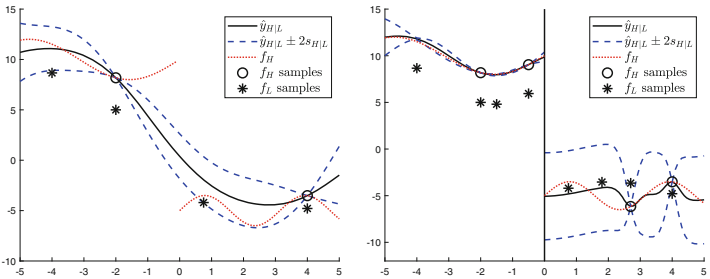
$$\delta_{k+1} = \frac{\delta_k v(\Sigma_k)}{v(\Sigma_k) - v(\Sigma_k \setminus \Sigma_{k+1})}. \tag{14}$$

Step 4. *Continue?* Check a stopping criterion and either stop and return the current set of subregions in Σ_{k+1} to provide an approximation to the target level set, or increment the iteration counter $k \leftarrow k+1$ and go back to Step 1.

Illustrative Example. We showcase the proposed algorithm with a simple example where the true (unknown) function, with a discontinuity at $x = 0$, and the corresponding low fidelity are:

$$f_H = \begin{cases} 2 \cdot \sin(x) + 10 & \text{if } -5 \le x < 0 \\ 1.5 \cdot \sin(2x) - 5 & \text{if } 0 \le x \le 5. \end{cases}, \quad f_L = \begin{cases} 2.2 \cdot \sin(x) + 7 & \text{if } -5 \le x < 0 \\ 1.5 \cdot \sin(0.75x) - 5 & \text{if } 0 \le x \le 5. \end{cases}$$

Figure 2(a) shows the quality of the predictor $\hat{y}_{H|L}(x)$, when 4 points are evaluated in low-fidelity and only 2 in high-fidelity. In Fig. 2(b), the interval is partitioned into two subregions, and 4 more points are evaluated in low-fidelity, while only 2 more in high-fidelity. The partition is very effective in this example, and Step 3 is able to confidently prune the left half.



(a) First iteration with 4 low-fidelity and 2 high-fidelity evaluations.

(b) Partition and add 4 low-fidelity and 2 high-fidelity evaluations.

Fig. 2. Illustration of a one-dimensional function $f_H(x)$ with a predictor function $\hat{y}_{H|L}(x)$ and predictive error $s^2_{H|L}(x)$, across two iterations of the algorithm with increased simulation budget. Note that only high fidelity values are interpolated.

3 Algorithm Analysis

The analysis of the multi-fidelity algorithm provides a probabilistic guarantee that the approximation of the target set does not incorrectly prune a large volume. The main result in Theorem 1 says that, on any kth iteration, the volume of accumulated region that was pruned incorrectly is less than ϵ with probability $(1-\alpha)^2$.

Lemma 1. *For any iteration $k \geq 1$, suppose all previous pruning is correct, that is, the comparison in (12) is satisfied for all of the pruned subregions, $x \in S \setminus \Sigma_k$. Then, the δ_k updated according to (14) can be used to determine the quantile over the entire set S, i.e.,*

$$y^k_{H|L}(\delta, S) = y^k_{H|L}(\delta_k, \Sigma_k) \tag{15}$$

where the quantile notation, as in (2), is used with function $\hat{y}^k_{H|L}$.

Proof. Lemma 1 is a special case of Theorem 1 in Huang and Zabinsky [5] with no maintained subregions.

Lemma 2. *Suppose σ_p has been pruned on the kth iteration. Also, suppose $y^k_{H|L}(\delta, S) \leq \hat{y}^k_{H|L}\left(x_{\Sigma_k,(s)}\right)$. Then, the volume of the incorrectly pruned region, i.e., $v(L^k_{H|L}(\delta, S) \cap \sigma_p)$, is less than or equal to ϵ_k with probability at least $1-\alpha_k$:*

$$P\left(v(L^k_{H|L}(\delta, S) \cap \sigma_p) \leq \epsilon_k | y^k_{H|L}(\delta, S) \leq \hat{y}^k_{H|L}\left(x_{\Sigma_k,(s)}\right)\right) \geq (1-\alpha_k)^2, \epsilon_k = \frac{\epsilon}{B^k}$$

Proof. Lemma 2 is a special case of Theorem 2 in Huang and Zabinsky [5] applied to function $\hat{y}^k_{H|L}$.

Theorem 1. *The pruned subregions on the kth iteration contain at most ϵ volume of the target δ level set $L^k_{H|L}(\delta, S)$ with probability $(1-\alpha)^2$,*

$$P\left(v(L^k_{H|L}(\delta, S) \setminus \Sigma_k\right) \leq \epsilon) \geq (1-\alpha)^2. \tag{16}$$

Proof. Suppose there are $d_m < B^m$ subregions pruned at iteration m and $\epsilon_m = \frac{\epsilon}{B^m}$, then we have $d_m \epsilon_m < \epsilon$. Considering a subregion σ^m_p that was pruned at iteration m, $m = 1, \ldots, k$, then the incorrect pruned volume results

$$\begin{aligned} P_v &= P\left(v(L^m_{H|L}(\delta, S) \setminus \cup_{p=1}^{d_m} \sigma^m_p) \leq d_m \epsilon_m | y^m_{H|L}(\delta, S) \leq \hat{y}^m_{H|L}\left(x_{\Sigma_m,(s)}\right)\right) \\ &= \prod_{p=1}^{d_m} P\left(v(L^m_{H|L}(\delta, S) \setminus \sigma^m_p) \leq \epsilon_m | y^m_{H|L}(\delta, S) \leq \hat{y}^m_{H|L}\left(x_{\Sigma_m,(s)}\right)\right) \qquad (17) \\ &\geq (1-\alpha_m)^{2d_m} \qquad (18) \end{aligned}$$

where (17) holds since each subregion is pruned independently, and (18) holds due to Lemma 2. Assuming that the pruning decision in iteration k is made independently from the decisions on prior iterations, the following holds,

$$
\begin{aligned}
P(v(L^k_{H|L}(\delta,S)\setminus\Sigma_k)\le\epsilon) &\ge P\left(v(L^k_{H|L}(\delta,S)\setminus\cup_{m=1}^k\cup_{p=1}^{d_m}\sigma_p^m)\le d_m\epsilon_m\right)\\
&=\prod_{m=1}^k P\left(v(L^k_{H|L}(\delta,S)\setminus\cup_p\sigma_p^m)\le d_m\epsilon_m|y_{H|L}(\delta,S)\le\hat{y}_{H|L}\left(x_{\Sigma_m,(s)}\right)\right)\\
&\qquad\cdot P\left(y_{H|L}(\delta,S)\le\hat{y}_{H|L}\left(x_{\Sigma_m,(s)}\right)\right)\\
&\ge\prod_{m=1}^k(1-\alpha_m)^{2d_m}P\left(y_{H|L}(\delta_m,\Sigma_m)\le\hat{y}_{H|L}\left(x_{\Sigma_m,(s)}\right)\right) \qquad (19)\\
&\ge\prod_{m=1}^k(1-\alpha_m)^{2d_m}(1-\alpha_m) \qquad (20)\\
&\ge\prod_{m=1}^k(1-\alpha_m)^{2B}=\prod_{m=1}^k(1-\alpha_m)^B(1-\alpha_m)^B \qquad (21)\\
&\ge(1-\alpha)(1-\alpha)=(1-\alpha)^2 \qquad (22)
\end{aligned}
$$

where (19) holds based on Lemma 1 and (20) applies the interval quantile estimated from [1] with s calculated as in (13). The inequality in (21) holds since $d_m < B$. Finally, the inequality in (22) is obtained by repeatedly applying Bernoulli's inequality $(1-\frac{\alpha_k}{B})^B \ge (1-B\frac{\alpha_k}{B})$. □

4 Conclusions

Our multi-fidelity algorithm iteratively constructs a predictor function that is updated and specialized over subregions of the entire feasible region. Since the predictor is constructed to pass a test indicating that it is of good accuracy, we can use it to guide the placement of high-fidelity function evaluations. Leveraging our statistical model, Theorem 1 provides a finite time probabilistic guarantee for the quality of the resulting approximate target level set.

References

1. Conover, W.J.: Practical Nonparametric Statistics. Wiley, Hoboken (1980)
2. Frazier, P.: A tutorial on Bayesian optimization. arXiv:1807.02811v1 [stat.ML], 8 July 2018 (2018)
3. Gardner, J., Guo, C., Weinberger, K., Garnett, R., Grosse, R.: Discovering and exploiting additive structure for Bayesian optimization. In: Artificial Intelligence and Statistics, pp. 1311–1319 (2017)
4. Hoag, E., Doppa, J.R.: Bayesian optimization meets search based optimization: a hybrid approach for multi-fidelity optimization. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
5. Huang, H., Zabinsky, Z.B.: Adaptive probabilistic branch and bound with confidence intervals for level set approximation. In: Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World, pp. 980–991. IEEE Press (2013)
6. Inanlouganji, A., Pedrielli, G., Fainekos, G., Pokutta, S.: Continuous simulation optimization with model mismatch using Gaussian process regression. In: 2018 Winter Simulation Conference (WSC), pp. 2131–2142. IEEE (2018)

7. Kandasamy, K., Dasarathy, G., Schneider, J., Póczos, B.: Multi-fidelity Bayesian optimisation with continuous approximations. In: Proceedings of the 34th International Conference on Machine Learning, vol. 70. pp. 1799–1808. JMLR.org (2017)
8. Kandasamy, K., Schneider, J., Póczos, B.: High dimensional Bayesian optimisation and bandits via additive models. In: International Conference on Machine Learning, pp. 295–304 (2015)
9. Li, C., Gupta, S., Rana, S., Nguyen, V., Venkatesh, S., Shilton, A.: High dimensional Bayesian optimization using dropout. arXiv:1802.05400 (2018)
10. Linz, D.D., Huang, H., Zabinsky, Z.B.: Multi-fidelity simulation optimization with level set approximation using probabilistic branch and bound. In: 2017 Winter Simulation Conference (WSC), pp. 2057–2068. IEEE (2017)
11. March, A., Willcox, K.: Provably convergent multifidelity optimization algorithm not requiring high-fidelity derivatives. AIAA J. **50**(5), 1079–1089 (2012)
12. Mockus, J.: Bayesian Approach to Global Optimization. Kluwer Academic Publishers, Dordrecht (1989)
13. Mockus, J.: Application of Bayesian approach to numerical methods of global and stochastic optimization. J. Global Optim. **4**, 347–365 (1994)
14. Poloczek, M., Wang, J., Frazier, P.: Multi-information source optimization. In: Advances in Neural Information Processing Systems, pp. 4288–4298 (2017)
15. Santner, T.J., Williams, B.J., Notz, W., Williams, B.J.: The Design and Analysis of Computer Experiments, vol. 1. Springer, New York (2003). https://doi.org/10.1007/978-1-4757-3799-8
16. Takeno, S., et al.: Multi-fidelity Bayesian optimization with max-value entropy search. arXiv preprint arXiv:1901.08275 (2019)
17. Törn, A., Zilinskas, A.: Global Optimization. Springer, Berlin (1989). https://doi.org/10.1007/3-540-50871-6
18. Wang, Z., Zoghi, M., Hutter, F., Matheson, D., De Freitas, N.: Bayesian optimization in high dimensions via random embeddings. In: Twenty-Third International Joint Conference on Artificial Intelligence (2013)
19. Wild, S.M., Regis, R.G., Shoemaker, C.A.: ORBIT: optimization by radial basis function interpolation in trust-regions. SIAM J. Sci. Comput. **30**(6), 3197–3219 (2008)
20. Wu, J., Toscano-Palmerin, S., Frazier, P.I., Wilson, A.G.: Practical multi-fidelity Bayesian optimization for hyperparameter tuning. arXiv:1903.04703 (2019)
21. Xu, J., Zhang, S., Huang, E., Chen, C.H., Lee, L.H., Celik, N.: An ordinal transformation framework for multi-fidelity simulation optimization. In: 2014 IEEE International Conference on Automation Science and Engineering (CASE), pp. 385–390. IEEE (2014)
22. Xu, J., Zhang, S., Huang, E., Chen, C.H., Lee, L.H., Celik, N.: MO2TOS: multi-fidelity optimization with ordinal transformation and optimal sampling. Asia-Pac. J. Oper. Res. **33**(03), 1650017 (2016)
23. Zabinsky, Z.B., Huang, H.: A partition-based optimization approach for level set approximation: probabilistic branch and bound. In: Smith, A.E. (ed.) Women in Industrial and Systems Engineering. WES, pp. 113–155. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-11866-2_6