2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)
August 23-27, 2021. Lyon, France

Using Gaussian Processes to Automate Probabilistic Branch & Bound for Global Optimization

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

2021 IEEE 17th International Conference on Automation Science and Engineering (CASE) | 978-1-6654-1873-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/CASE49439.2021.9551592

Abstract— **Manufacturing, aerospace, energy and several other industries have witnessed a steep growth of increasingly complex, information rich, devices and systems of devices requiring simulation-based approaches. In fact, most modern systems have such complex behavior that their performance can only be evaluated through, usually computationally expensive, simulations. In such settings, it is of paramount importance to provide solutions with quality guarantees. In this manuscript, we focus on algorithms capable of identifying a level set of solutions in proximity of the global optimum, and specifically on the Probabilistic Branch and Bound (PBnB) method. We propose a new way to automate branching decisions by coupling this method with Gaussian process (GP) estimation. The result is PBnB-GP, where, at each iteration a collection of GPs is used to decide how to branch the input space. PBnB-GP not only returns an estimate of the regions with near-optimal reward (using the power of PBnB), but also a "collection of Gaussian processes" that can produce point estimations for any location in the input space, thus harnessing the power of model-based black-box optimization. We present PBnB-GP for the first time together with preliminary numerical results.**

I. INTRODUCTION AND BACKGROUND

With the increasing complexity of devices within the manufacturing, aerospace, energy and other industries, complex simulations have become key for the design, evaluation, certification and control of devices and systems in these areas. In this work, we are particularly interested in the identification of close-to-optimal conditions, i.e., we focus on optimization tasks. More specifically, we are interested in identifying regions where the inputs or design variables achieve close-to-optimal behavior, rather than returning a single solution to the user.

Oftentimes, such simulators are computationally demanding, and it is important for optimization methods that make use of those, to be able to perform as few evaluations as possible. In this arena, we have seen a proliferation of black-box optimization methods. Two main families of black-box methods have been used in the literature: (i) direct search methods; (ii) model based methods. For the algorithms in class (i), methods differ in the way they balance the exploration/exploitation trade-off, but only "local" information is used, i.e., only the current point informs the decision and no model of the response is built. Examples are the Greedy Randomized Adaptive Search Procedure (GRASP) [1], and Improving Hit-and-run (IHR) [2]. A potential challenge arising from this family of algorithms is their sample efficiency. An alternative is a meta-model based search, which, different from direct methods, uses previous samples to build a meta-model (surrogate) of the objective function (reward) at locations that have not been previously evaluated. The Efficient Global Optimization (EGO) [3], for deterministic black-box settings, is a reference algorithm in this category.

In this work, we look into the specific problem of identifying the level set of solutions in proximity of the global minimum, and to do so we couple Gaussian processes with Probabilistic Branch and Bound (PBnB), that uses a directed random search to approximate a target level set associated with a target quantile of the global solutions [4], [5], [6]. An advantage to PBnB is that it partitions the space iteratively, and performs more function evaluations in the promising regions as it refines its approximation of the target level set. In addition, there is a probabilistic bound on the error of the approximation.

A. Relevant State of the Art

Bayesian optimization represents a practical response to several optimization problems requiring the simulation of complex systems, which usually lead to rewards that are multi-modal and non-differentiable [7], [8], [9]. Bayesian optimization is a general terms that is now associated to a large number of algorithms. The basic ides is to use an *a priori* random process to represent the reward and iteratively update the posterior of such random process using the (conditional) distribution of the objective function as a means to sequentially sample locations in the input space (the interested reader can refer to [10] for more details). While Gaussian processes are prevalent in the implementation of Bayesian optimization, several approaches have been proposed using different models, such as radial basis functions which have shown an important level of success [11]. In the context of Bayesian optimization in high dimensions, additive forms have been investigated [12], [13], as well as embeddings, both with the purpose of tackling the scalability of model-based approaches [14], [15].

Within the statistics literature, partitioning and Gaussian processes have received increasing attention, particularly, within the context of machine learning in large data sets with goals such as classification and level set estimation. We generally differ in two main aspects from this literature: (i) rather than estimating a level set for a given reference

*This work was supported by NSF CMMI#, NSF CISE #

[1]Giulia Pedrielli is with School of Computing Informatics and Decision Systems Engineering, Arizona State University, 699 S Mill Ave, Tempe, AZ 85258, USA giulia.pedrielli@asu.edu

[2]Hao Huang is with College of Engineering, Yuan Ze University, TAIWAN haohuang@saturn.yzu.edu.tw

[3]Zelda B. Zabinsky is with the Department of Industrial and Systems Engineering, University of Washington, Seattle, WA 98195-2650, USA zelda@uw.edu

function value, we estimate the reference function value that achieves a target $1-\alpha$ quantile of the function; (ii) rather than specifying the sampling and branching mechanisms as inputs to the statistical method, we sample and branch adaptively and intelligently to efficiently identify the target region.An example of this related literature in level set estimation is [16], where the authors use partitions, and a unique Gaussian process to decide, based on the predicted model variance, whether to add sampling budget and/or branch the subregions. Conformal regression [17] has also been proposed as a statistical approach (in contrast to Gaussian processes). The issue of level set identification using conformal regression is that predictions can only be made at a region level and no point estimates are returned by the method [18]. Another approach in the statistical literature are "treed" Gaussian processes, a learning method that solves challenges such as nonstationarity, and the large size of the data set based on the idea of Bayesian partition models [19], [20], [21]. As an example [21] couples stationary Gaussian processes with partitioning. An extension to this method has been proposed that partitions the subregions using nonstationary Bayesian Gaussian processes [22]. Both methods fix the branching scheme and only allow a single cut on each iteration. In our proposed method, we allow variable cuts and decide which dimension to cut based on the results of the Gaussian process.

B. Contribution

In this paper, we combine Probabilistic Branch and Bound with Gaussian processes, and use the statistical power to strategically branch the input space and use fewer computationally-expensive function evaluations while preserving the theoretical features of PBnB. The new approach uses the subregions identified by PBnB and learns a different model in each of them. The models are necessary to estimate, for each branching decision, the so-called *probability of classification*, which quantifies the ability at a specific iteration to *prune* (the subregion is unlikely to contain the global minimum) or *maintain* (the subregion is likely to contain the global minimum). The branching with the associated highest probability of classification is pursued.

II. PROPOSED FRAMEWORK

We consider an optimization problem with an unknown reward function $f(x)$:

$$\min_{x} \; f(x) \qquad (1)$$

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

where $S \subset \Re^d$ is the feasible region of inputs or design variables, and $f : S \to \Re$. We also consider a collection of Gaussian process models, $G_i : S_{i,k} \to \Re$, where $S_{i,k}$ is subregion i generated on the k^{th} iteration of PBnB. On iteration k, subregions $S_{i,k}$ for all i are mutually exclusive subsets of S. It is important to highlight that we can quickly generate observations from $G_i(x)$, i.e., the computational cost of generating predictions from a Gaussian process is orders of magnitude lower than evaluating $f(x)$, a costly simulation.

We are interested in determining the set of the best δ-quantile solutions, which can be defined as a level set bounded by a quantile $y(\delta,S)$, for $0<\delta<1$,

$$y(\delta,S) = \arg\min_{f}\{P(f(X)\le y | x\in S)\ge \delta\}, \qquad (2)$$

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

$$L(\delta,S) = \{x\in S : f(x)\le y(\delta,S)\}, \text{ for } 0<\delta<1. \qquad (3)$$

We note that for quantile level δ, $\delta = \frac{\nu(L(\delta,S))}{\nu(S)}$, where $\nu(\cdot)$ is the d-dimensional volume (i.e., Lebesgue measure) of a set. Similarly, we define $y_{G_i}(\delta,S)$ and $L_{G_i}(\delta,S)$ as quantile and target set associated with the Gaussian prediction model $G_i(x)$, respectively.

The goal of the algorithm introduced in this paper is to approximate the target level set $L(\delta,S)$ using relatively few $f(x)$ function evaluations and allow many more evaluations of the Gaussian processes.

A. Algorithm Overview

We propose the *Gaussian process-driven Probabilistic Branch and Bound* (PBnB-GP), to efficiently branch the input space and approximate $L(\delta,S)$. Given a fixed overall sampling budget, we conjecture efficient partitioning helps to reduce the prediction error associated with a Gaussian prediction model in highly promising regions, by eliminating poor performing regions from the prediction.

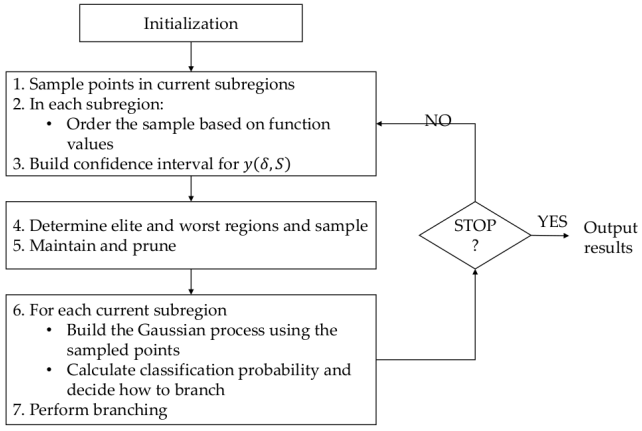


Fig. 1. Proof of Concept for PBnB-GP.

Figure 1 shows the basic architecture of PBnB-GP with its main components: (i) Initialization; (ii) Sampling and function quantile estimation of $y(\delta,S)$ using available data and a frequentist argument; (iii) Subregions update (maintain or prune), where sampling is directed to a subset of regions in order to statistically determine whether a subregion should be eliminated from future consideration, either the region is deemed part of the target level set (maintained), or the region is determined to be outside the target level set in which case it is pruned; (iv) Update of the Gaussian process

G_i in each active subregion (not pruned or maintained) $S_{i,k}$ at each iteration k of the algorithm, resulting in a collection of statistical learning models $\{G_i\}$; (v) For each active subregion $S_{i,k}$ at each iteration k, determine the best dimension to branch based on the Gaussian process $G_i(x)$.

PBnB-GP uses the Gaussian processes predicted for each active subregion to establish the location of the cutting hyperplanes (we consider orthogonal cutting planes in this work). This is different from common partitioning approaches that uniformly cut along each dimension in the active subregions. We argue that, if the metamodel captures the function behavior, the resulting partitions will increase the algorithm ability to separate promising subregions from poor performing areas, and enable faster pruning and maintaining decisions. The approach has deep roots in kernel learning, where mixture, additive, and graph-based models can be thought of as relevant alternatives to PBnB-GP [23], [24], [25], [26], [27], [28], [29]. However, their challenge is how to combine optimization and prediction needs, when learning the large number of model parameters needed to capture separable components of the model and their relation [30], [31], [32]. We argue that the proposed approach will rival state-of-the-art structured kernels. In fact, PBnB-GP uses simpler models coupled with partitioning to learn promising subregions.

The numerical results in Section III illustrate the performance of PBnB-GP when used to identify a target level set.

B. Gaussian Processes for Subregion Reward Estimation

A Gaussian process (GP) is a statistical learning model used to build predictions for non-linear, possibly non-convex smooth functions [30]. The basic idea is to interpret the true, unknown function $f(\mathbf{x})$ as a realization from a stochastic process, the Gaussian process. If we can measure the function without noise, then the Gaussian process will interpolate the true function values at the evaluated points, and, conditional on the sampled locations $\mathbf{x}_1,\ldots,\mathbf{x}_n$, produce the conditional probability density. In particular, we are interested in the process $Y(\mathbf{x}) = \mu + Z(\mathbf{x})$, where μ is the constant process mean, and $Z(\mathbf{x}) \sim GP(0,\tau^2 R)$, with τ^2 being the constant process variance and R the correlation matrix. Under the Gaussian correlation assumption, $R_{ij} = \prod_{l=1}^{d} \exp\left(-\theta_l \left(x_{il} - x_{jl}\right)^2\right)$, for $i,j,=1,\ldots,n$. The d-dimensional vector of hyperparameters θ controls the smoothing intensity of the predictor in the different dimensions. The parameters μ and τ^2 are estimated through maximum likelihood [30]: $\hat{\mu} = \frac{1_n^T \mathrm{R}^{-1} f(X_n)}{1_n^T \mathrm{R}^{-1} 1_n}$, $\hat{\tau}^2 = \frac{(f(\mathbf{X}_n) - 1_n\hat{\mu}_g)^T \mathrm{R}^{-1}((f(\mathbf{X}_n) - 1_n\hat{\mu}_g)}{n}$. The best linear unbiased predictor form is [30]:

$$\hat{f}(\mathbf{x}) = \hat{\mu} + \mathbf{r}^T \mathrm{R}^{-1}(f(\mathbf{X}_n) - \mathbf{1}_n\hat{\mu}) \tag{4}$$

where $\mathbf{X}_n$ is a set of n sampled locations, and $f(\mathbf{X}_n)$ is the n-dimensional vector having as elements the function value at the sampled locations. The model variance associated to the predictor is:

$$s^2(\mathbf{x}) = \tau^2\left(1 - \mathbf{r}^T \mathrm{R}^{-1}\mathbf{r} + \frac{(1 - \mathbf{1}_n^T \mathrm{R}^{-1}\mathbf{r})^2}{\mathbf{1}_n^T \mathrm{R}^{-1}\mathbf{1}_n}\right) \tag{5}$$

where $\mathbf{r}$ is the n-dimensional vector having as elements the Gaussian correlation between location $\mathbf{x} \in \mathbb{X}$ and the n elements of $\mathbf{X}_n$, i.e., $\mathbf{r}_i(\mathbf{x}) = \prod_{l=1}^{d} \exp\left(-\theta_l (x_l - x_{il})^2\right), i = 1,\ldots,n$.

C. Probabilistic Branch and Bound

Probabilistic Branch and Bound is a partitioning-based random search global optimization algorithm that approximates a target level set with statistical confidence. For example, a user may desire to capture a *set* of the best 10% solutions, instead of a single estimate of a global optimum [33], [4], [5], [34], [6].

PBnB iteratively maintains, prunes or branches subregions of a bounded solution space based on an updated confidence interval of a target quantile see [6]. PBnB *maintains* a subregion when there is statistical confidence that it is contained in the target level set, and similarly, *prunes* a subregion when there is statistical confidence that it does not intersect the target level set. Order statistics are used to determine the quality of each subregion for pruning, maintaining, and branching decisions. Finite time probability bounds are derived on the maximum volume of incorrectly maintained or incorrectly pruned regions, as stated below.

Let Σ_k^P be the union of all of the subregions that have been pruned on the k-th iteration, and let Σ_k^M be union of all of the subregions that have been maintained on the k-th iteration. A main result provides a probability bound on the volume of incorrectly pruned regions, and incorrectly maintained regions, as stated below.

Theorem 1 (cf. [6]): The pruned subregions Σ_k^P on the k-th iteration of PBnB contain at most $\varepsilon > 0$ volume of the target δ-quantile level set $L(\delta,S)$ with probability at least $(1-\alpha)^4$, and similarly, the maintained subregions Σ_k^M on the k-th iteration of PBnB contain at most $\varepsilon > 0$ volume outside the target δ-quantile level set $L(\delta,S)$ with probability at least $(1-\alpha)^4$,

$$P\left(v(L(\delta,S) \setminus \Sigma_k^P\right) \leq \varepsilon) \quad \geq \quad (1-\alpha)^4 \tag{6}$$

$$P\left(v(\Sigma_k^M \setminus L(\delta,S)\right) \leq \varepsilon) \quad \geq \quad (1-\alpha)^4. \tag{7}$$

Here, ε is the volume that we are willing to tolerate making an error, for example, we may be willing to tolerate a region that is 2% of the volume of S that is pruned or maintained incorrectly.

The branching decision in PBnB is straight-forward; the longest dimension of a subregion is branched into B equal sized subregions, where B is a user-defined parameter. The active subregions on any iteration are all of the same size. We are convinced that Gaussian processes can provide valuable information to branch more effectively.

D. Partitioning to Maximize the Classified Volume

The main idea of the algorithm is to make use of a number of Gaussian process predictors available at the k^{th} iteration of the algorithm in order to make a determination of the best partitioning scheme within a finite set. Such a determination is not made based on evaluation using simulation, but simply generating observations from the Gaussian process (which is largely cheaper than simulating the original function).

At each iteration k, we have a set of statistically undecided, active subregions $\tilde{\Sigma}_k = S_{1,k} \cup \ldots \cup S_{i,k} \cup \ldots \cup S_{r_k,k}$, each associated with the Gaussian process G_i, $i = 1, \ldots, r_k$. Given a specified number of cuts B, we decide the cutting dimension d, from $d = 1, \ldots, D$, that maximizes the number of "classified" regions according to the Gaussian process prediction. We define the likelihood that a subregion is likely-to-be-pruned, as

$$p_i^p(d) = \inf_{x \in S_{i,k}(d)} P\left(G_i(x) > y(\delta, S) + \delta^u\right), \forall i \tag{8}$$

and is likely-to-be-maintained, as

$$p_i^m(d) = \inf_{x \in S_{i,k}(d)} P\left(G_i(x) < y(\delta, S) - \delta^\ell\right), \forall i \tag{9}$$

where δ^ℓ and δ^u are arbitrary non negative real values, and each subregion $S_{i,k}(d)$ is a function of the cutting dimension d selected, hence the notation $S_{i,k}(d)$. In this work, we generate points from the Gaussian processes in order to have a frequentist estimation of the "classified volume".

From the definitions, it is apparent that we want to achieve a high probability of classifying a region. The worst case is when the two likelihoods are equal, i.e.,

$$p_i^p(d) = p_i^m(d) = 1/2$$

Hence, for each subregion, the classification probability is

$$p_i^c(d) = |p_i^p(d) - p_i^m(d)| \tag{10}$$

This probability is non-negative and it is at its minimum when the two likelihoods are identical. Given a cutting dimension d, we have that the probability of classification for $\tilde{\Sigma}_k$ is

$$\bar{p}^c(d) = \frac{1}{|\tilde{\Sigma}_k(d)|} \sum_{i=1}^{|\tilde{\Sigma}_k(d)|} [p_i^c(d)] \tag{11}$$

and the dimension is chosen such that:

$$d_k^* \in \arg\max_{d=1,\ldots,D} \bar{p}^c(d). \tag{12}$$

It is important to notice that, the sample average $\frac{1}{|\tilde{\Sigma}_k|}\sum_{i=1}^{|\tilde{\Sigma}_k|}[p_i^c]$ can be generalized using subregion volume-dependent weights:

$$\bar{p}^c(d) = \sum_{i=1}^{|\tilde{\Sigma}_k|} \frac{v\left(S_{i,k}\right)}{v\left(\tilde{\Sigma}_k\right)} [p_i^c(d)]. \tag{13}$$

This allows us to apply our classification-based criteria to branching algorithms that allow subregions of different sizes.

E. PBnB-GP

In this section, we present the main steps for PBnB-GP.

Step 1. *Sample points in the current subregions*:
Generate additional sample points in each subregion $S_{i,k}$ for all $i = 1, \ldots, I$, and $S_{i,k} \in \Sigma_k$ so there is at least one point in each subregion for output evaluations. Update the set of sampled points. While the number of samples is important for cross validating the statistical models, the idea is that the number of simulation runs is much lower than the number of predictions from the Gaussian processes obtained up to iteration k.

Using the expensive function evaluations, build, for each subregion, the Gaussian process $G_i(x;\theta_i)$, where θ_i is the vector of hyperparameters for the model.

Step 2. *Test classification capability of the set of partitioning schemes*:
Given a set of cutting dimensions $d \in \{1, \ldots, D\}$, we produce the associated subregions within the current set of subregions Σ_k into B new subregions, denoted $S_{1,k}(d), \ldots, S_{B,k}(d)$, where the new subregions are mutually exclusive. Update Σ_k with the newly branched subregions.

For each subregion calculate the classification probability, as in (10). Calculate the summary classification probability associated with the cutting dimension d, as in (11). Choose the best cutting direction:

$$d_k^* \in \arg\max_d \bar{p}^c(d)$$

Generate iteration $k+1$ partitions.

Step 3. *Make pruning decision*:
For each subregion $S_{i,k}$ in Σ_k, $i = 1, \ldots, I$, uniformly and independently sample $N_k = \left\lceil \frac{\ln \alpha_k}{\ln\left(1 - \frac{\varepsilon}{v(S)}\right)} \right\rceil$ points $\tilde{x}_{S_{i,k},n}$ for $n = 1, \ldots, N_k$. Within each subregion $S_{i,k}$, order these sampled points by their values f_i and denote them $\tilde{x}_{S_{i,k},(1)}, \ldots, \tilde{x}_{S_{i,k},(N_k)}$, where

$$\hat{y}_i(\tilde{x}_{S_{i,k},(1)}) \le \hat{y}_i(\tilde{x}_{S_{i,k},(2)}) \le \cdots \le \hat{y}_i(\tilde{x}_{S_{i,k},(N_k)}).$$

For each subregion $S_{i,k}$ perform the maintaining and pruning test,

$$\hat{y}_i\left(\tilde{x}_{S_{i,k},(N_k)}\right) - z_{\frac{\alpha_k}{2}} s_i(\tilde{x}_{S_{i,k},(N_k)}) < \hat{y}_k\left(\tilde{x}_{\Sigma_k,(r)}\right) + z_{\frac{\alpha_k}{2}} s_k(x_{\Sigma_k,(r)}) \tag{14}$$

$$\hat{y}_i\left(\tilde{x}_{S_{i,k},(1)}\right) + z_{\frac{\alpha_k}{2}} s_i(\tilde{x}_{S_{i,k},(1)}) > \hat{y}_k\left(\tilde{x}_{\Sigma_k,(s)}\right) - z_{\frac{\alpha_k}{2}} s_k(x_{\Sigma_k,(s)}) \tag{15}$$

where r and s satisfies,

$$\max r : \sum_{i=0}^{r-1} \binom{N_k}{i} (\delta_k)^i (1-\delta_k)^{N_k - i} \le \frac{\alpha_k}{2}. \tag{16}$$

$$\min s : \sum_{i=0}^{s-1} \binom{N_k}{i} (\delta_k)^i (1-\delta_k)^{N_k - i} \ge 1 - \frac{\alpha_k}{2}. \tag{17}$$

The test in (15) compares the best value in a subregion with the s-best value overall, accounting for an error term with confidence α_k.

If (15) is satisfied, prune $S_{i,k}$. If (15) is not satisfied, keep $S_{i,k}$. Update Σ_{k+1} with the subregions in Σ_k that have not been pruned.

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

$$\delta_{k+1} = \frac{\delta_k v(\Sigma_k)}{v(\Sigma_k) - v(\Sigma_k \setminus \Sigma_{k+1})}. \tag{18}$$

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.

III. NUMERICAL RESULTS

Stochastic algorithms similar to PBnB-GP have been applied in the context of verification of cyber-physical systems [35], [36], and it has been highlighted how approaches that provide guarantees on the lower bound of the objective function are necessary in the field. In this study, rather than looking at the engineering application, where the shape of the level set is typically unknown, we examine the performance of PBnB-GP on three test functions and compare different criteria to explore branching directions. In fact, our experimental objective is to verify the correctness and efficiency of the proposed automation. In order to graphically illustrate the results, all test functions are set to be two-dimensional. The test functions are as follows.

- Three-hump camel function $(-5 \le x_1, x_2 \le 5)$

$$g_0(x) = 2x_1^2 - 1.05x_1^4 + \frac{x_1^6}{6} + x_1x_2 + x_2^2$$

  The global optimum is located at $x_* = (0,0)$ with $g_0(x_*) = 0$.
- Rosenbrock's function $(-2 \le x_i \le 2, i = 1,\dots,n,\ n = 2)$

$$g_1(x) = \sum_{i=1}^{n-1} [(1-x_i)^2 + 100(x_{i+1} - x_i^2)^2].$$

  The global optimum is located at $x_* = (1,1)$ with $g_1(x_*) = 0$.
- Shifted sinusoidal function $(0 \le x_i \le 180, i = 1,\dots,n,\ n = 2)$

$$g_2(x) = -2.5\prod_{i=1}^{n} \sin\left(\frac{\pi(x_i+60)}{180}\right) - \prod_{i=1}^{n} \sin\left(\frac{\pi(x_i+60)}{36}\right).$$

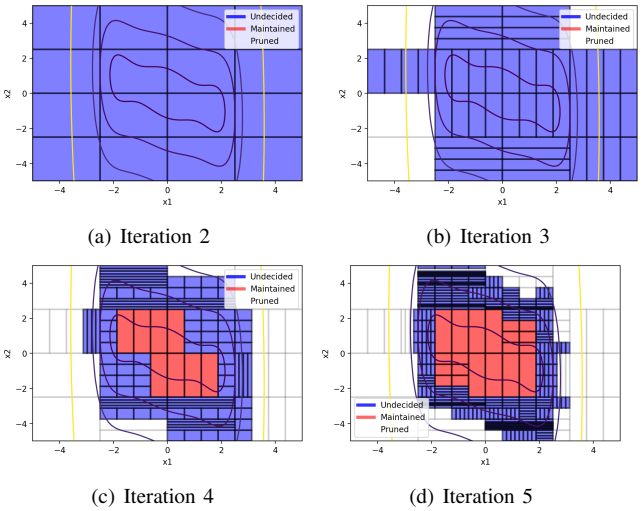
  The global optimum is located at $x_* = (30,30)$ with $g_2(x_*) = -3.5$.

For PBnB-GP, we set the confidence level at $\alpha = 0.1$ and the target level set as $\delta = 0.3, 0.1, 0.1$ for the three test functions, respectively. The branching parameter B is set to four.

For the Three-hump camel function, PBnB-GP approaches the 0.3-quantile level set iteratively in Figure 2, showing the second to fifth iteration. The curves in Figure 2 show the contours of 0.1, 0.3, 0.5, and 0.7 quantiles. Hence, the second inner curve is the target level set we wish to approximate. As shown in the legend, the white area represents the pruned subregions, the red area is the maintained subregions, and the blue region is the currently undecided.

Figure 2(a) and 2(b) demonstrate that PBnB-GP quickly selects branch directions and prunes a decent amount of solution space, where more samples are concentrated to maintain the level set in Figure 2(c) and 2(d). PBnB-GP quickly branches x_1 and identifies its border.

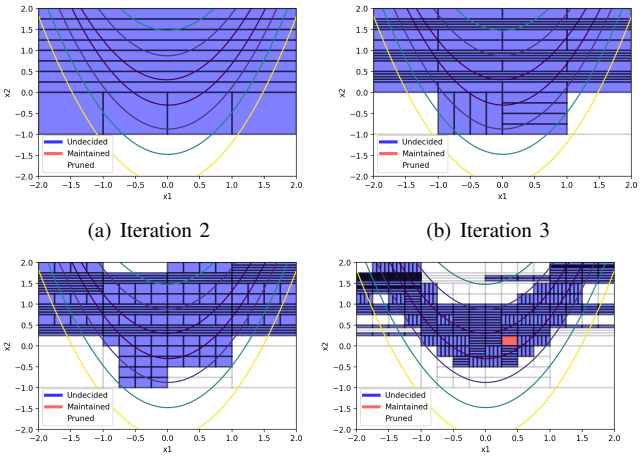
Table I shows the detail performances of PBnB-GP on the Three-hump camel, Rosenbrock's, and Shifted sinusoidal functions, and compares the two criteria of branching directions, using (10) for absolute value (abbreviated Abs in Table I), and using the maximum $p_i^c(d) = \max\{p_i^p(d), p_i^m(d)\}$ (abbreviated as Max in Table I). For the volume pruned and



(a) Iteration 2 (b) Iteration 3

(c) Iteration 4 (d) Iteration 5

Fig. 2. Iterations of solution space from PBnB-GP on Three-hump camel function



(a) Iteration 2 (b) Iteration 3

(c) Iteration 4 (d) Iteration 5

Fig. 3. Iterations of solution space from PBnB-GP on Rosenbrock's function

maintained, the max criterion seems to slightly out-perform the abs on three-hump camel function, where the shifted sinusoidal function shows the reverse. For Rosenbrock's function, they perform similarly, where the domain space of one run is shown in Figure 3. The pruned and maintained volume also leads to the same comparison of the sample size. Since the Three-hump camel function has a more extreme shape on different dimensions, the abs and max criterion prune more subregions than PBnB, as shown in the bottom of Table I.

IV. CONCLUSIONS

In this paper, we present for the first time the PBnB-GP algorithm with the objective to automate and intelligently choose the branching dimension at each iteration. In order to make this branching decision, we use a collection of Gaussian processes (one for each of the subregions generated by the partitioning at each iteration). Specifically, each Gaussian process is used to estimate the probability that a subregion

TABLE I

COMPARISON OF PBNB-GP PERFORMANCE ON FUNCTIONS

Metrics	Three-hump camel		Rosenbrock		Shifted sinusoidal	
Abs	Mean	SE	Mean	SE	Mean	SE
Vol P	57.73	2.67	10.57	0.48	25,515	813
Vol M	14.69	3.71	0.05	0.05	1,038	349
Best f	0.0013	0.001	0.004	0.004	-3.49	0.001
#Sample	32,227	5,707	27,475	2,548	19,949	3,023
Max	Mean	SE	Mean	SE	Mean	SE
Vol P	54.69	3.22	10.6	0.260	25,768	1,239
Vol M	17.42	1.87	0.063	0.108	734	315
Best f	0.0014	0.001	0.007	0.011	-3.49	0.001
#Sample	35,021	5,624	27,753	1,641	16,586	2,785
PBnB	Mean	SE	Mean	SE	Mean	SE
Vol P	51.59	6.99	11.93	0.128	25,717	1,218
Vol M	20	6.10	0.075	0.103	1,102	310
Best f	0.0089	0.112	0.116	0.115	-3.39	0.139
#Sample	27,122	3,454	26,002	795	19,765	1,716

will be classified as pruned or maintained. A subregion is classified if either the maximum prediction is above the current function level estimation or below. Results show how the PBnB-GP algorithm preserves the performance of the original PBnB while being able to specialize the cutting decision for the different subregions.

Current and future work focus on several aspects: (i) allowing the algorithm to stop branching a region based on the classification probability; (ii) choosing the number of cuts together with cutting dimension; and (iii) using Gaussian processes to decide the sampling effort in each subregion.

ACKNOWLEDGMENT

The research in this paper has been partially supported by the grant NSF #2000792, and NSF #1829238.

REFERENCES

[1] T. A. Feo and M. G. Resende, "Greedy randomized adaptive search procedures," *Journal of global optimization*, vol. 6, no. 2, pp. 109–133, 1995.

[2] Z. B. Zabinsky, R. L. Smith, J. F. McDonald, H. E. Romeijn, and D. E. Kaufman, "Improving hit-and-run for global optimization," *Journal of Global Optimization*, vol. 3, no. 2, pp. 171–192, Jun 1993. [Online]. Available: https://doi.org/10.1007/BF01096737

[3] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[4] W. Wang, "Adaptive random search for noisy and global optimization," *Ph.D. thesis, University of Washington*, 2011.

[5] Z. B. Zabinsky, W. Wang, Y. Prasetio, A. Ghate, and J. W. Yen, "Adaptive probabilistic branch and bound for level set approximation," in *Proceedings of the 2011 Winter Simulation Conference*, S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, Eds. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc., 2011, pp. 46–57.

[6] Z. B. Zabinsky and H. Huang, "A partition-based optimization approach for level set approximation: Probabilistic branch and bound," in *Women in Industrial and Systems Engineering: Key Advances and Perspectives on Emerging Topics*, A. E. Smith, Ed. Springer Nature, 2020, pp. 113–155.

[7] J. Mockus, *Bayesian Approach to Global Optimization*. Kluwer Academic Publishers, Dordrecht, 1989.

[8] ——, "Application of Bayesian approach to numerical methods of global and stochastic optimization," *Journal of Global Optimization*, no. 4, pp. 347–365, 1994.

[9] A. Törn and A. Zilinskas, *Global Optimization*. Springer-Verlag, Berlin, 1989.

[10] P. Frazier, "A tutorial on Bayesian optimization," *arXiv:1807.02811v1 [stat.ML] 8 Jul 2018*, 2018.

[11] S. M. Wild, R. G. Regis, and C. A. Shoemaker, "Orbit: Optimization by radial basis function interpolation in trust-regions," *SIAM Journal on Scientific Computing*, vol. 30, no. 6, pp. 3197–3219, 2008.

[12] J. Gardner, C. Guo, K. Weinberger, R. Garnett, and R. Grosse, "Discovering and exploiting additive structure for Bayesian optimization," in *Artificial Intelligence and Statistics*, 2017, pp. 1311–1319.

[13] K. Kandasamy, J. Schneider, and B. Póczos, "High dimensional Bayesian optimisation and bandits via additive models," in *International Conference on Machine Learning*, 2015, pp. 295–304.

[14] C. Li, S. Gupta, S. Rana, V. Nguyen, S. Venkatesh, and A. Shilton, "High dimensional Bayesian optimization using dropout," *arXiv:1802.05400*, 2018.

[15] Z. Wang, M. Zoghi, F. Hutter, D. Matheson, and N. De Freitas, "Bayesian optimization in high dimensions via random embeddings," in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[16] S. Shekhar and T. Javidi, "Multiscale gaussian process level set estimation," in *Proceedings of Machine Learning Research*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and M. Sugiyama, Eds., vol. 89. PMLR, 16–18 Apr 2019, pp. 3283–3291.

[17] C. Fan, X. Qin, and J. Deshmukh, "Parameter searching and partition with probabilistic coverage guarantees," 04 2020.

[18] J. Lei, M. G'Sell, A. Rinaldo, R. J. Tibshirani, and L. Wasserman, "Distribution-free predictive inference for regression," *Journal of the American Statistical Association*, vol. 113, no. 523, pp. 1094–1111, 2018. [Online]. Available: https://doi.org/10.1080/01621459.2017.1307116

[19] H. Chipman, E. George, and R. McCulloch, "Bayesian treed models," *Machine Learning*, vol. 48, pp. 299–320, 07 2002.

[20] D. Denison, N. Adams, C. Holmes, and D. Hand, "Bayesian partition modelling," *Computational Statistics Data Analysis*, vol. 38, no. 4, pp. 475–485, 2002, nonlinear Methods and Data Mining.

[21] R. B. Gramacy and H. K. H. Lee, "Bayesian treed gaussian process models with an application to computer modeling," Tech. Rep., 2005.

[22] W. Liang and H. Lee, "Bayesian nonstationary gaussian process models via treed process convolutions," *Advances in Data Analysis and Classification*, vol. 13, 09 2018.

[23] V. Tresp, "Mixtures of gaussian processes," in *Advances in neural information processing systems*, 2001, pp. 654–660.

[24] M. I. Jordan, *Learning in graphical models*. Springer Science & Business Media, 1998, vol. 89.

[25] A. Damianou and N. Lawrence, "Deep gaussian processes," in *Artificial Intelligence and Statistics*, 2013, pp. 207–215.

[26] M. Seeger, "Gaussian processes for machine learning," *International journal of neural systems*, vol. 14, no. 02, pp. 69–106, 2004.

[27] E. B. Sudderth, M. J. Wainwright, and A. S. Willsky, "Embedded trees: Estimation of gaussian processes on graphs with cycles," *IEEE Transactions on Signal Processing*, vol. 52, no. 11, pp. 3136–3150, 2004.

[28] Z. Wang, B. Zhou, and S. Jegelka, "Optimization as estimation with gaussian processes in bandit settings," in *Artificial Intelligence and Statistics*, 2016, pp. 1022–1031.

[29] F. Bachoc, "Asymptotic analysis of the role of spatial sampling for covariance parameter estimation of gaussian processes," *Journal of Multivariate Analysis*, vol. 125, pp. 1–35, 2014.

[30] T. J. Santner, B. J. Williams, and W. I. Notz, *The design and analysis of computer experiments*. Springer Science & Business Media, 2013.

[31] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced lectures on machine learning*. Springer, 2004, pp. 63–71.

[32] J. P. Kleijnen, "Regression and kriging metamodels with their experimental designs in simulation: review," 2015.

[33] Y. Prasetio, "Simulation-based optimization for complex stochastic systems," *Ph.D. thesis, University of Washington*, 2011.

[34] H. Huang, "Discrete-event simulation and optimization to improve the performance of a healthcare system," *Ph.D. thesis, University of Washington*, 2016.

[35] L. Mathesen, S. Yaghoubi, G. Pedrielli, and G. Fainekos, "Falsification of cyber-physical systems with robustness uncertainty quantification through stochastic optimization with adaptive restart," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2019, pp. 991–997.

[36] G. Ernst, P. Arcaini, I. Bennani, A. Donze, G. Fainekos, G. Frehse, L. Mathesen, C. Menghi, G. Pedrinelli, M. Pouzet, *et al.*, "Arch-comp 2020 category report: Falsification," *EPiC Series in Computing*, 2020.