SIAM J. SCI. COMPUT.
Vol. 42, No. 4, pp. A2134–A2157

OPTIMAL LEARNING WITH LOCAL NONLINEAR PARAMETRIC MODELS OVER CONTINUOUS DESIGNS*

XINYU HE†, KRISTOFER G. REYES‡, AND WARREN B. POWELL§

Abstract. We consider the problem of optimizing an unknown function over a multidimensional continuous domain, where function evaluation is noisy and expensive. We assume that a globally accurate model of the function is not available, but there exist some parametric models that can well approximate the function in local regions. In this paper, we propose an algorithm in the optimal learning framework that learns the shape of the function and finds the optimal design with a limited number of measurements. We construct belief functions using a radial basis function–based local approximation technique, and use the knowledge gradient policy to decide where to measure, aiming at maximizing the value of information from each measurement. Experiments on both synthetic test problems and a real materials science application show the strong performance of our algorithm.

Key words. optimal learning, knowledge gradient, local approximation, nonlinear parametric models

AMS subject classifications. 62L05, 62F07, 62F15, 65D15

DOI. 10.1137/19M1245608

1. Introduction.

We consider the following problem: we have a smooth function over a multidimensional continuous domain. The function itself is unknown to us, and there does not exist a model to describe it globally. However, within any local region, the function can be well approximated by a parametric function, say $g(x;\theta)$, which may be nonlinear in the parameter vector θ. Our goal is to find the global maximum of the true function. We can take a series of measurements to collect information at any point in the domain, but the measurements are noisy and expensive, as might arise in simulations, laboratory, or field experiments. Therefore, we only have a budget of N measurements, and after exhausting the budget, we need to give our best estimate of the optimal x (where x is also called an *alternative* or a *design*).

There has been a substantial literature on the problem of optimizing unknown functions through sequential noisy measurements. The author of [61] provides a thorough review of stochastic search methods. Some of the most popular policies include gradient-based methods [45, 56, 24], random search methods [1, 2, 36], metaheuristics such as tabu search, genetic algorithms, and simulated annealing algorithms [51, 7, 9], and metamodel methods such as trust regions [18, 16], response surface methodology [41, 55], and surrogate models [8, 54]. However, these methods usually assume that function evaluations are inexpensive and thus require a large number of measurements, while our focus is when the measurements are expensive.

The problem of optimizing expensive and noisy functions, especially when the

*Submitted to the journal's Methods and Algorithms for Scientific Computing section February 25, 2019; accepted for publication (in revised form) April 9, 2020; published electronically July 13, 2020.
https://doi.org/10.1137/19M1245608
Funding: This work was supported by the Air Force Office of Scientific Research under contract FA9550-12-1-0200.

†Department of Electrical Engineering, Princeton University, Princeton, NJ 08544 (xinyuhe@alumni.princeton.edu).
‡Department of Materials Design and Innovation, SUNY University at Buffalo, Buffalo, NY 14260 (kreyes3@buffalo.edu).
§Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08544 (powell@princeton.edu).

function is defined on a finite number of alternatives, is often referred to as the ranking and selection (R&S) problem (see, e.g, [42, 43, 35]). Some heuristics for the R&S problem include epsilon-greedy [62, 60], interval estimation [40], and upper confidence bounding methods [4]. More sophisticated methods are often established in two approaches: the frequentist (see, e.g, [31, 28, 23, 25, 63, 3]), which is based entirely on the observed data, and the Bayesian (see, e.g, [12, 32, 30, 57]), which starts with a prior distribution of belief and updates the belief model as data is collected.

We focus on a class of Bayesian policies for maximizing the value of information known as the knowledge gradient (KG), which is well suited to problems where the cost of running experiments is high. By construction, KG is a myopically optimal policy, but has also been shown to be asymptotically optimal, which means it is able to identify the optimal alternative if the budget $N \to \infty$. KG has been developed for the three major classes of belief models: look-up table models, parametric models, and nonparametric models. For look-up table models, the authors of [21] propose KG for independent alternatives, and the authors of [22] establish KG for correlated alternatives. For parametric belief models, the authors of [49] focus on the case when the function is linear in the unknown parameters, while the authors of [13, 34] study the case when the parametric function is nonlinear in unknown parameters. Some work on nonparametric models includes [47], which proposes a hierarchical aggregation technique for KG, and [5], which estimates functions using kernel regression. All these methods are designed for discrete alternatives. For problems with continuous alternatives, the authors of [59] use Gaussian process regression based on a look-up table belief model, but it is limited to only low dimensions; the authors of [33] propose an algorithm when the true function can be modeled by a differentiable parametric function, which works well in problems with even 10 or 20 dimensions, but requires a globally accurate parametric model for the true function.

For problems that do not have a globally accurate parametric belief model, an estimate of the global function is usually built using local approximation methods. There has been a rich literature on local approximation techniques (see [37] for a comprehensive review). The locally linear model proposed in [19, 20] builds linear models around each observation but cannot provide a global mathematical formula for the regression function. Another class of algorithms builds local models around local regions instead of each data point. The authors of [29] use the Dirichlet process mixture model to identify local regions of the input space and then fit linear models around each region. Moreover, radial basis functions (RBFs) [10, 11] have received considerable attention due to their simplicity and generality. The authors of [39] present normalized RBFs, which perform well for limited training data. The authors of [37] propose the Dirichlet cloud radial basis function (DC-RBF) model, which provides a compact representation over the entire domain and a fast method to update the approximation. Additionally, the authors of [14] combine the DC-RBF model with the KG model for linear functions [49] and derive the KG policy for a local approximation model that fits linear functions around local regions. This is the first optimal learning work based on local approximation models. However, it can only handle the case when the local approximation is a linear function and the alternative space (i.e., the domain) is discrete.

In this paper, we address the problem when the alternative space is continuous and multidimensional, but a globally accurate model is not available, while any local region can be well described by a parametric function (which can be either linear or nonlinear). This paper makes the following contributions:

(1) We propose a method to build a belief model consisting of a variety of global

approximations and show how the model gets updated dynamically as we run experiments.

(2) We further incorporate the optimal learning methods and develop the KG policy for our belief model over a continuous alternative space.

(3) We evaluate our algorithm on a variety of problems, ranging from a series of test functions to a real-world application on carbon nanotube growth, all showing strong performance.

This paper is organized as follows. We review some related work on the KG policy in section 2 and a local approximation method known as the DC-RBF model [37] in section 3. In section 4, we introduce our method of constructing a belief model of the true function and how we use the KG policy to select an alternative to measure. We show some empirical results on several different synthetic problems and a real application in section 5 and conclude in section 6. We also provide some heuristics and suggestions for using the algorithm in Appendix A.

2. Review of knowledge gradient. We start by introducing the ranking and selection (R&S) problem (see, e.g, [42, 43, 35]). In an R&S problem, there are a finite number of alternatives $\mathcal{X} = \{x_1, x_2, \ldots, x_M\}$, each associated with a true utility f_x^*. The utilities are presumably unknown to us, and our goal is to identify the $x \in \mathcal{X}$ with the largest utility through a budget of N sequential noisy measurements. In this paper, we assume the noise is Gaussian with mean 0 and variance σ^2.

2.1. The basic knowledge gradient policy. The knowledge gradient policy solves the offline R&S problem by maximizing the value of information from each experiment. At time n (i.e, after n measurements), suppose we decide to measure an alternative x^n according to some policy. Let S^n denote our state of knowledge at time n, and $V^n(S^n)$ denote the value of being in (knowledge) state S^n, which is the value if we stop learning. Moreover, let $\bar{f}_x^n$ be our estimate of f_x^* at time n. In R&S, $V^n(S^n) = \max_{x\in\mathcal{X}} \bar{f}_x^n$, which represents the current highest utility based on our estimation. Conditioned on S^n, the expected improvement of the value of information by measuring x, which we call the *knowledge gradient* (KG) score, is given by

$$\begin{aligned}\nu^{KG,n}(x) &= \mathbb{E}\left[V^{n+1}(S^{n+1}(x)) - V^n(S^n)|S^n = s, x^n = x\right] \\ &= \mathbb{E}^n\left[\max_{x'\in\mathcal{X}} \bar{f}_{x'}^{n+1}(x)|x^n = x\right] - \max_{x'\in\mathcal{X}} \bar{f}_{x'}^n, \end{aligned} \tag{2.1}$$

where $\mathbb{E}^n$ is the expectation over the random outcome of measuring x, conditioned on S^n. For each experiment, the KG policy chooses the alternative x with the largest KG score.

While the KG score in (2.1) may be easy to compute when the utility function has some simple form such as a linear utility [49], it is generally computationally intractable when the utility is nonlinear in the parameters. Below we briefly review some previous work handling nonlinear parametric belief models, introduced in [13, 34, 33]. Note that all of these methods require a known globally accurate model, while our work in this paper breaks this limitation.

2.2. KG for nonlinear parametric models. Consider the case where the unknown true utility f_x^* can be described by a parametric function $f(x;\theta)$, in which $x \in \mathcal{X}$ is the alternative, and $\theta \in \Theta$ is a vector of unknown parameters. The expression of $f(x;\theta)$ is known, but the true parameter θ^* is unknown, and $f(x;\theta)$ can be nonlinear in θ. We call this the *nonlinear parametric belief model.*

The authors of [13] propose an algorithm that can handle the cases when both $\mathcal{X}$ and Θ are discrete, assuming the true parameter θ^* is in the discrete set of Θ. Specifically, each $\theta \in \Theta$, also called a *candidate*, is associated with a probability. Our estimate of $f^*(x)$ is given by $\bar{f}_n(x) = \sum_{\theta\in\Theta} f(x;\theta)p^n(\theta)$. The KG score is calculated accordingly by (2.1). We measure the alternative with the largest KG score at each step and update the posterior probabilities $p^n(\theta)$ while the candidate set Θ stays fixed.

The requirement for Θ and $\mathcal{X}$ to both be discrete limits the method in [13] to a small θ space and a small x space. The authors of [34, 33] extend the algorithm to a larger θ space and a larger x space, respectively, both of which can be multidimensional. Specifically, the authors of [34] design a resampling algorithm in the θ space, using the mean squared error (MSE) between $f(x;\theta)$ and the measurements so far as the metric. Rather than keeping a fixed candidate set, it maintains a small set of θ's, while dynamically replacing the less probable θ's with newly sampled more probable ones as we take measurements.

Furthermore, the authors of [33] focus on a continuous multidimensional $\mathcal{X}$ and introduce an approximation of the KG score defined in (2.1) (note that (2.1) is computationally intractable if $\mathcal{X}$ is continuous for a general $f(x;\theta)$). It then describes a method to calculate the approximated KG score as well as its derivative at any $x \in \mathcal{X}$ efficiently (assuming $f(x;\theta)$ is differentiable in x). Then it uses gradient ascent to find the x with the largest approximated KG score, which will be the alternative we measure next. We will refer to this algorithm as *continuous KG* below. The authors of [34, 33] provide theoretical proof on the asymptotical optimality of the methods, saying that as we take an infinite number of measurements, we will find the optimal x and the true θ^* almost surely; empirical results on problems that are multidimensional in x and θ also show the strong performance of these methods.

However, the biggest restriction of these methods is that they all assume a globally accurate model $f(x;\theta)$ is known to us. In many real applications, such a globally accurate model either may not exist or may be difficult to obtain. Instead, many times we only have a locally accurate model that can well describe any local neighborhood. We solve this problem in this paper, with the help of a local approximation technique, known as DC-RBF.

3. Review of a local approximation technique: DC-RBF. The Dirichlet cloud radial basis function (DC-RBF) model [37] is a function approximation technique that uses locally linear models to construct a global representation the underlying function. Suppose we have measured a sequence of data points. DC-RBF groups these data into several clusters based on their distance in x space, and fits a linear model for each cluster, which forms a local approximation of the neighborhood. It then employs normalized radial basis functions (RBFs) to generate the weights of the local approximations and uses the weighted sum as a global approximation.

DC-RBF clusters the measured data in a recursive way. It has a predefined fixed parameter, known as the *radius* and denoted as D_T, which controls the size of the local regions. Initially, the first measured alternative, x^0, forms the only cluster itself; at time n, either the newly measured x^n is added to the nearest cluster if the *distance* is smaller than D_T, or it forms a new cluster itself. The *distance* from x to a cluster is defined as the distance from x to the cluster's centroid, which is the arithmetic mean of all data points in the cluster.

For each cluster, a linear model is fitted based on the data in it. Assume there are C^n clusters at time n and the linear model for cluster j is given by $g_j(x;\theta)$; then

a global approximation of the underlying true function is given by

$$\bar{f}^n(x) = \sum_{j=1}^{C^n} w_j(x) \cdot g(x; \theta_j) \quad \forall x \in \mathcal{X}, \tag{3.1}$$

where $w_j(x)$ is defined by the normalized RBF functions [39, 37] based on the distance from x to cluster j, which represents the weights of $g(x; \theta_j)$ and is close to one near the centroid of cluster j but goes to zero as x moves farther from the centroid.

Despite the impressive theoretical and empirical results in [37], DC-RBF is imperfect in a few aspects. First, it is usually hard to properly pick a fixed value for the radius D_T, especially before we have made enough observations; and second, the clustering result is often defective since it is sensitive to the order of the data. In our algorithm, we adopt the basic idea of DC-RBF, but we then introduce the following changes:

1. rather than using a fixed D_T, we predefine a range for D_T and dynamically sample a set of radii within the range, exploiting the measurement history;
2. we refine the clustering result using k-means to make it less sensitive to the order of measurements;
3. we no longer restrict $g(x; \theta)$ to linear functions;
4. more importantly, we combine it with our optimal learning framework to solve the problem mentioned above.

4. Algorithm. We consider the problem

$$\underset{x \in \mathcal{X}}{\operatorname{argmax}} f^*(x),$$

where $f^*(x)$ is the underlying true function which is unknown to us, and $\mathcal{X}$ is a compact set of alternatives. Our goal is to find the optimal alternative x^* after a budget of N sequential measurements, which are noisy and expensive. We assume that for any local region in $\mathcal{X}$, $f^*(x)$ can be well approximated by a parametric function $g(x; \theta)$, which is differentiable in x and can be nonlinear in θ. In other words, $\forall \epsilon > 0, \forall x \in \mathcal{X}$, there exists θ^* such that $|f^*(x) - g(x; \theta^*)| < \epsilon$ for any $x \in \mathcal{B}(x_0, r)$.

At time n, suppose we decide to measure an alternative x^n according to some policy, and the measurement result is $\hat{y}^{n+1}$, where the superscripts imply that $\hat{y}^{n+1}$ is unknown until after the $(n+1)$th measurement. Therefore, our measurements up to time n are $\{(x^0, \hat{y}^1), \ldots, (x^{n-1}, \hat{y}^n)\}$. In this paper, we assume the measurement noise W^n is Gaussian with 0 mean and variance σ^2, i.e, $\hat{y}^{n+1} = f^*(x^n) + W^{n+1}$, where $W^{n+1} \sim \mathcal{N}(0, \sigma^2)$, i.i.d. We focus on Gaussian noise in this paper because of its wide usage in various areas (see, e.g, [17, 26, 6, 46]), especially in most Bayesian work [15] as well as R&S problems [27, 52]. However, note that our approach of constructing belief models and selecting alternatives using KG is also applicable to other noise settings (with some changes to the computations).

4.1. Sketch of the basic ideas. We start with an overview of the key elements of our algorithm:

1. We dynamically sample a set of radii and correspondingly construct a variety of clustering results, as apposed to having a single radius and a single clustering result as in DC-RBF.
2. For each local region in each clustering result, we fit a local, low-order approximation $g(x; \theta)$ with the help of the resampling logic in [34].

3. Then for each clustering result, we use the normalized RBF [39, 37] to stitch the local approximations (i.e, the $g(x;\theta)$'s) together as a global approximation of the true function. Hence, we have a family of candidate global approximations, each corresponding to a different radius.
4. We keep a probability for each candidate global approximation and update the probabilities as we run experiments. This also allows us to represent the uncertainty of our estimate.
5. If the probability of a candidate global approximation is too low, we may update its radius, the clustering result, and the fitted local approximations to create a new candidate.
6. We apply the logic of the continuous KG [33] to our belief model to select an x to measure.

In the following, we discuss items 1–3 in section 4.2, item 4 in section 4.3 and section 4.4, item 5 in section 4.5, and item 6 in section 4.6.

4.2. Construction of candidate functions. In this part, we introduce the general idea of constructing our belief model, while some implementation details are introduced in the following sections. We illustrate the construction of the belief model with the help of Figure 1. As shown in the figure, we keep L different radii $\{D^n_{T,i}\}$ (column 1 in Figure 1), each leading to a different clustering result where the size of the cluster is controlled by the radius (column 2). For each cluster, we fit $g(x;\theta)$ according to the data points in that cluster (column 3). Then we combine the local $g(x;\theta)$'s to to get $f^n_i(x)$, which is a global approximation $f^n_i(x)$ parameterized by the radius $D^n_{T,i}$ (column 4), and finally we calculate the weighted sum of $f^n_i(x)$'s as $\bar{f}^n(x)$, which is our estimate of $f^*(x)$ at time n.

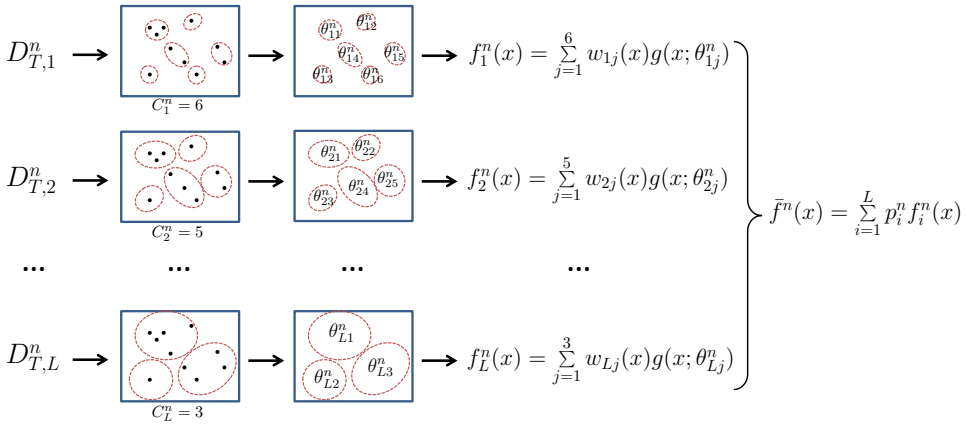

FIG. 1. *Illustration of construction of candidate functions and our estimate. From left to right:* (1) *L different radii (assuming $D^n_{T,1} < D^n_{T,2} < \cdots < D^n_{T,L}$) controlling the size of local regions;* (2) *clustering results (each dot is a measurement, and each circle is a cluster; C^n_i is the count of clusters; note that the sizes of the circles are different: the sizes in the first row are the smallest since $D^n_{T,1}$ is the smallest);* (3) *fitting $g(x;\theta)$ for each cluster using measurements in that circle;* (4) *combining $g(x;\theta)$'s to get candidate functions $\{f^n_i(x)\}$;* (5) *$\bar{f}^n(x)$: estimate of the true function.*

Specifically, we predefine a range, $[D_{T,min}, D_{T,max}]$, for the radius, which represents our belief of the reasonable size that can be covered by the local model. At time 0, we start with L radii, evenly distributed in $[D_{T,min}, D_{T,max}]$. The size of this set (namely L) is fixed, while the values of the radii may get updated as we take

measurements (section 4.5.2). Suppose at time n the radii are $D^n_{T,1}, D^n_{T,2}, \ldots, D^n_{T,L}$. As shown in Figure 1, each $D^n_{T,i}$ results in a clustering result i containing C^n_i clusters (the clustering method is introduced in section 4.4 and section 4.5.3). For each cluster j in clustering result i, we solve a statistical fitting problem (section 4.5.1), to get a local parameter, θ^n_{ij}, such that $g(x;\theta^n_{ij})$ approximates the measurement results in the local region around cluster j. For each radius $D^n_{T,i}$, we combine the $g(x;\theta^n_{ij})$'s to generate a candidate function $f^n_i(x)$, defined as

$$f^n_i(x) = \sum_{j=1}^{C^n_i} w_{ij}(x) g(x;\theta^n_{ij}), \tag{4.1}$$

where $w_{ij}(x)$ is given by the normalized RBF [37] as $w_{ij}(x) = \frac{\phi(||x-c_{ij}||_{W_{ij}})}{\sum_{k=1}^{C^n_i}\phi(||x-c_{ik}||_{W_{ik}})}$, where $\phi(\cdot)$ is the RBF function, c_{ij} is the centroid of cluster j in clustering result i, and W_{ij} is the regularized empirical covariance matrix of the data in cluster j.

Note that $g(x;\theta)$ may not necessarily be linear. Moreover, since $w_{ij}(x)$ and $g(x;\theta^n_{ij})$ are both differentiable in x, so is $f^n_i(x)$, which makes it possible to apply continuous KG since it requires $f^n_i(x)$ to be differentiable in x [33]. In practice, it is usually sufficiently accurate to use numerical methods to calculate the gradient of $f^n_i(x)$.

For each candidate function $f^n_i(x)$, we maintain a probability p^n_i and compute the expectation of the candidate functions as the estimate of the true function $f^*(x)$, i.e.,

$$f^*(x) \approx \bar{f}^n(x) = \sum_{i=1}^{L} p^n_i f^n_i(x). \tag{4.2}$$

At time n, our best estimate of the optimal alternative $\hat{x}^n$ is given by $\hat{x}^n = \operatorname{argmax}_{x\in\mathcal{X}} \bar{f}^n(x)$.

4.3. Posterior probabilities of candidate function. At time 0, we assign equal prior probabilities to the L candidate functions, namely $p^0_i = \frac{1}{L}\ \forall i$. We update their probabilities after each measurement according to Bayes's rule.

In previous variations of KG [13, 34, 33], due to the assumption that our parametric model is globally accurate, the probabilities are calculated only considering the measurement noise σ. However, since in our settings the candidate functions $f^n_i(x)$ can hardly be the same as the true function, we should use a larger σ to compensate for the model error. For instance, consider the case when there is no measurement noise: with a globally accurate parametric model, we might be able to identify the true function with only a few measurements; but this is no longer a trivial problem in our case since the noise-free measurements can only allow us to find the truth of a local region.

Now let $\tilde{\sigma}^2 > \sigma^2$ be the adjusted variance. In practice, $\tilde{\sigma}$ should be set according to our estimate of the model discrepancy. Some typical values might be $\tilde{\sigma} = 1.5\sigma$ or $\tilde{\sigma} = \sigma + 0.1(f_{max} - f_{min})$, where $(f_{max} - f_{min})$ is the estimated range of the true function. Recall that our experiments have noise $W^n \sim \mathcal{N}(0,\sigma^2)$, and at time n, the measurement results are $\{(x^0,\hat{y}^1),\ldots,(x^{n-1},\hat{y}^n)\}$. Hence, the likelihood of f^n_i is given by

$$\mathcal{L}^n_i = \prod_{j=1}^{n} \exp\left(-\frac{[\hat{y}^j - f^n_i(x^{j-1})]^2}{2\tilde{\sigma}^2}\right). \tag{4.3}$$

By Bayes's theorem, we know the posterior probability $p_i^n \propto p_i^0 \cdot \mathcal{L}_i^n$, which is given by

$$p_i^n = \frac{\mathcal{L}_i^n}{\sum_{j=1}^{L} \mathcal{L}_j^n}. \tag{4.4}$$

4.4. Minor update of candidate functions. After each measurement, we get a new result $(x^n, \hat{y}^{n+1})$. We first update each clustering result by adding x^n in a recursive way, as in DC-RBF. Specifically, for each clustering result i $(1 \le i \le L)$, we calculate the smallest distance from x^n to all clusters. If this minimum distance is smaller than $D_{T,i}^n$, x^n is added to the nearest cluster; otherwise, x^n forms a new cluster itself. We also update the corresponding centroid and empirical covariance matrix (i.e., c_{ij} and W_{ij} in (4.1)) accordingly.

If x^n is added to an existing cluster for all clustering results in the approach above, we call such an update a *minor update*. In a minor update, only one c_{ij} and one W_{ij} in each clustering result get updated; therefore, the weights $w_{ij}(x)$ in (4.1) are moderately changed, and hence the candidate functions are only slightly different from before updating. In other words, in Figure 1, the radii are unchanged (column 1), the clustering results are almost unchanged (with only x^n to an existing clustering) (column 2), and the set of θ_{ij}'s stays the same (column 3), while only $w_{ij}(x)$ is refreshed (column 4). We then also update the posterior probabilities $\{p_i^{n+1}\}$ according to (4.4) once a minor update is done.

However, if x^n creates new clusters in some clustering results, or if some p_i^{n+1}'s after a minor update are too small, we turn to a *major update*, in which we update the first three steps in Figure 1 for corresponding candidates, namely (1) the radii, (2) the clustering results, and (3) the set of θ_{ij}'s.

After each measurement, we always try a minor update first rather than do a major update directly, which makes the candidate functions more stable.

4.5. Major update of candidate functions. At certain times, we may find that the probabilities of some candidate functions are too small or that new clusters have been created during recursive clustering. In such cases, we do a *major update* for the corresponding candidate functions to better adapt to the data by three possible operations: (1) resampling the radii ($D_{T,i}$'s), (2) reclustering, and (3) refitting local θ_{ij}'s, corresponding to the first three columns in Figure 1.

We define three conditions that can trigger a major update by any of them: (1) More than $L/2$ candidate functions have probabilities smaller than some predefined threshold ϵ_p. (2) Any clustering result has created new clusters. (3) We have not done major updates for n^r (say $n^r = 5$) time steps, or in other words, we have used roughly the same candidates for n^r times. In (1), at least all candidates with $p_i^n < \epsilon_p$ would get updated; in (2), at least all candidates with new clusters would get updated; in (3), we first set the $L/4$ smallest probabilities as 0, and thus at least $L/4$ of candidates would get updated. We define the termination condition for a major update to be less than $L/4$ candidates having $p_i^n < \epsilon_p$.

Every time a major update is triggered, for all candidate functions with low probabilities, we first update the local function $g(x; \theta_{ij})$ for each cluster, keeping the $D_{T,i}$'s and clustering results unchanged. We then recalculate the probabilities according to (4.4). If the termination condition is satisfied, the major update finishes; otherwise, we next update the radii and clustering results, followed by refitting the $g(x; \theta_{ij})$'s. We keep updating all three elements until the termination condition is met. In practice, to make sure the update terminates after some time, we count the

number of rounds each function fails to reduce the fitting error compared with before this round. If the failure count hits some threshold, we exclude the function from the termination condition (refer to Algorithm 4.1, particularly *failure_count*, for more details).

We assume f_i^n is one of the candidate functions that needs a major update and discuss the three operations of a major update from section 4.5.1 to section 4.5.3 below, in the order that they are conducted, namely (1) refitting θ_{ij}'s for local regions, (2) resampling $D_{T,i}$, and (3) constructing new clusterings. We include a flowchart of the major update in the end.

4.5.1. Refitting local functions. When a major update is needed for f_i^n, we first try refitting $\{\theta_{ij}^n,\, j = 1, \ldots, C_i^n\}$ while fixing the radius $D_{T,i}^n$ as well as the clustering result. For each cluster, we search for the θ that minimizes the sum of squared error (SSE) from the measurement history, which is equivalent to maximizing the likelihood under Gaussian noise. We use a method similar to the resampling method in [34]. We keep a large pool of K θ's, assuming K is large enough to well represent the discretized θ space. Suppose cluster j contains k_j data points: $\{(x_j^{(m)}, \hat{y}_j^{(m)}),\, m = 1, \ldots, k_j\}$. For each θ in the large pool, we calculate the SSE between $g(x;\theta)$ and the k_j measurements, given by

$$SSE(\theta) = \sum_{m=1}^{k_j} \left[g(x_j^{(m)};\theta) - \hat{y}_j^{(m)} \right]^2. \tag{4.5}$$

We resample θ from a sublevel set of the $SSE(\theta)$. Specifically, let the S θ's with the smallest $SSE(\theta)$ form the *small pool*, denoted as $\mathbb{S}$, where $L < S \ll K$. The new θ_{ij} is then sampled from a small pool. The probability of $\theta \in \mathbb{S}$ being drawn is given by its normalized likelihood, namely

$$p(\theta) = \frac{\exp\left(-\frac{\sum_{m=1}^{k_j} \left[g(x_j^{(m)};\theta) - \hat{y}_j^{(m)} \right]^2}{2\tilde{\sigma}^2} \right)}{\sum_{\theta' \in \mathbb{S}} \exp\left(-\frac{\sum_{m=1}^{k_j} \left[g(x_j^{(m)};\theta') - \hat{y}_j^{(m)} \right]^2}{2\tilde{\sigma}^2} \right)} = \frac{\exp\left(-\frac{SSE(\theta)}{2\tilde{\sigma}^2} \right)}{\sum_{\theta' \in \mathbb{S}} \exp\left(-\frac{SSE(\theta')}{2\tilde{\sigma}^2} \right)}. \tag{4.6}$$

Note that we use the adjusted variance $\tilde{\sigma}^2$ in this formula. As illustrated in [34], the method of sampling from the small pool keeps a balance of exploration and exploitation in the θ space.

There are two heuristics in refitting local functions: (1) we can include some data in nearby clusters when fitting θ_{ij}, in order to make the candidate function f_i^n smoother, and (2) if we have enough data points in a certain cluster, we can directly solve a fitting problem that minimizes the SSE to obtain the optimal θ_{ij} instead of sampling it from a large pool. See Appendix A.2 for more details on both points.

After updating $\{\theta_{ij}^n,\, j = 1, \ldots, C_i^n\}$, we check whether the SSE of the new candidate function has decreased from before this update and only replace the candidate if so. Additionally, every time a round of candidate function updating is finished, we should recalculate the probabilities of all candidates using (4.4) and check whether the new probabilities satisfy the termination condition. If not, which means $p_i^n < \epsilon_p$ for some candidates, we turn to updating the radii and clustering results for these least likely candidates.

4.5.2. Resampling a radius. As discussed above, if the procedure of updating $\{\theta_{ij}^n,\, j = 1, \ldots, C_i^n\}$ cannot make $p_i^n > \epsilon_p$, we need to update the radius $D_{T,i}^n$. Note that there is a tradeoff in the value of D_T^n: a smaller radius enables $g(x; \theta_{ij}^n)$ to match the measurement results better, but it also results in more clusters, each containing fewer data, which makes the fitting problem undetermined. In our algorithm, we look at the probability distribution of the current L radii (where the *probability* is the posterior probability) and resample a $D_{T,i}^n$ in $[D_{T,min}, D_{T,max}]$ according to a mixture of an approximated Beta distribution and a uniform distribution.

Specifically, we first rescale all $D_{T,l}^n$'s from $[D_{T,min}, D_{T,max}]$ to $[0, 1]$ by letting $D'_T = \frac{D_T^n - D_{T,min}}{D_{T,max} - D_{T,min}}$. We then approximate the probability distribution of $\{D_{T,l}^n\}$ by a Beta distribution, which provides a variety of distribution curves on a closed interval with a small number of parameters. We assume there exists a constant c such that for any $l = 1, 2, \ldots, L$,

$$p_l^n = c \cdot (D'_{T,l})^{\alpha-1}(1 - D'_{T,l})^{\beta-1}.$$

Therefore, we can approximate α, β, and c by solving the following optimization problem of minimizing the SSE:

$$\min_{\alpha>0,\beta>0,c>0} \sum_{l=1}^{L} \left[c \cdot (D'_{T,l})^{\alpha-1}(1 - D'_{T,l})^{\beta-1} - p_l^n\right]^2.$$

Note that for given α and β, the optimal c can be uniquely determined by solving an ordinary least squares problem, with a closed-form equation given by

$$c = \frac{\sum_{l=1}^{L} (D'_{T,l})^{\alpha-1}(1 - D'_{T,l})^{\beta-1} p_l^n}{\sum_{l=1}^{L} \left[(D'_{T,l})^{\alpha-1}(1 - D'_{T,l})^{\beta-1}\right]^2}, \tag{4.7}$$

and so the optimization problem can be simplified to

$$\min_{\alpha>0,\beta>0} \sum_{l=1}^{L} \left[c \cdot (D'_{T,l})^{\alpha-1}(1 - D'_{T,l})^{\beta-1} - p_l^n\right]^2, \tag{4.8}$$

where c is given in (4.7). Equation (4.8) can be solved by grid search.

Figure 2 shows some results of the fitted Beta distribution. As shown in Figure 2, the probability densities of the left- and rightmost parts are sometimes too low. We thus sample $D'_{T,i}$ from a mixture of the Beta distribution and uniform distribution, each with probability 0.5, and then transform it back to get a new $D_{T,i}^n$ by $D_{T,i}^n = D_{T,min} + D'_{T,i} \cdot (D_{T,max} - D_{T,min})$.

4.5.3. Reconstructing clustering results. Once a new $D_{T,i}^n$ is sampled, we also need to update the corresponding clustering result. We first permute the data. Let $(x^{(0)}, x^{(1)}, \ldots, x^{(n-1)})$ be a random permutation of the previously measured alternatives. We add $(x^{(0)}, x^{(1)}, \ldots, x^{(n-1)})$ one by one according to the DC-RBF clustering procedure, as if they arrive in such an order, resulting in C_i^n clusters with centroids $\{c_{ij},\, j = 1, \ldots, C_i^n\}$. We then run k-means to refine the clustering result, with the centroids as the initial centers and C^n as the target number of clusters.

Recall that the original DC-RBF clustering procedure heavily depends on the order of the data, which usually generates suboptimal clustering results, and therefore, we use permutation and k-means as refinements. Figure 3 shows two examples

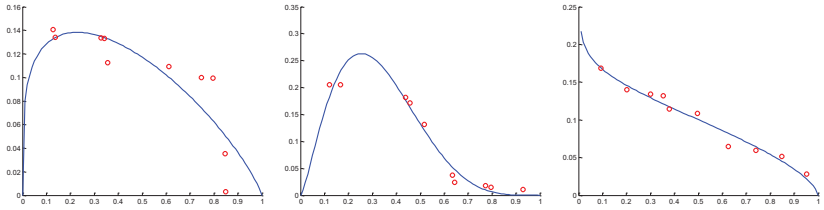

FIG. 2. *Three examples of fitted Beta distribution. The dots show the probability mass function of the L radii, where the x-axis is the value of a radius (after normalization) and the y-axis is the probability. Curves: fitted Beta distribution curve given the L dots.*

comparing DC-RBF clustering and our refined clustering. The left image in each example runs DC-RBF clustering, while the right one runs k-means on top of that, with different colors indicating different clusters. We can see the right images show better clustering results.

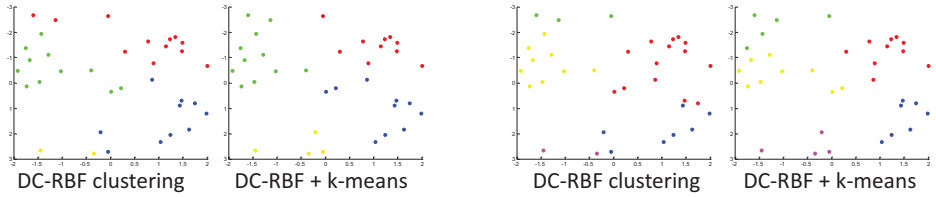

FIG. 3. *Two examples of DC-RBF clustering vs. DC-RBF+k-means clustering in a two-dimensional $\mathcal{X}$. The dots are where we have measured. Each color indicates a cluster. The first and third images run DC-RBF with different permutations of the measurements. The second and fourth images run k-means based on the clustering results shown in the first and third images, respectively. We can see that k-means has noticeable improvement on naive DC-RBF.*

4.5.4. Flowchart of major update of candidate function. We summarize the procedure of a major update of the candidate functions, as shown in Algorithm 4.1.

4.6. Incorporating continuous KG. We discuss how to incorporate the continuous KG algorithm [33] such that we can handle a multidimensional continuous $\mathcal{X}$. Note that $\mathbb{E}^n\left[\max_{x'\in\mathcal{X}} \bar{f}^{n+1}(x')\right]$ in (2.1) is generally computationally intractable if $\mathcal{X}$ is continuous. As in [33], we approximate $\nu^{KG,n}$ by considering the improvement of information on a discrete set. Let $\mathcal{X}_s$ be a uniformly sampled discrete subset of $\mathcal{X}$. Then $\forall x \in \mathcal{X}$,

$$\nu^{KG,n}(x) \approx \nu^{KG-s,n}(x) = \mathbb{E}^n\left[\max_{x'\in\mathcal{X}_s} \bar{f}^{n+1}(x')\right] - \max_{x'\in\mathcal{X}_s} \bar{f}^n(x'). \tag{4.9}$$

The expectation is over both the model uncertainty and the measurement noise. Hence, we further write it as

Algorithm 4.1. Major update of candidate functions.

Input: Current candidates: $\{f_i^n\}$; Probabilities: $\{p_i^n\}$; Measurement history: $\{(x^0, \hat{y}^1), \ldots, (x^{n-1}, \hat{y}^n)\}$.

Output: Updated candidates: $\{f_i^n\}$; Updated probabilities: $\{p_i^n\}$.

if $|\{i : p_i^n < \epsilon_p\}| \geq L/2$, or some candidates have new clusters, or major update hasn't happened for n^r iterations **then**
  $\mathbb{L}_{rm-old} = \emptyset$; $failure\_count = \vec{0}$.
  Set probabilities of the $L/4$ least probable candidates as 0.
  while $|\{i : p_i^n < \epsilon_p$ and $failure\_count\ (i) < 100\}| < L/4$ **do**
    $\mathbb{L}_{rm} = \{i : p_i^n < \epsilon_p\} \bigcup \{i :$ clustering result i has new clusters$\}$.
    $j = 1$.
    while $j \leq |\mathbb{L}_{rm}|$ **do**
      $i = \mathbb{L}_{rm}(j)$.
      if i is in $\mathbb{L}_{rm-old}$ **then**
        Fit a Beta distribution and sample a new $D_{T,i}^n$.
        Permute $\{x^0, \ldots, x^{n-1}\}$ and run DC-RBF clustering.
        Use k-means to refine the clustering result.
      end if
      for each cluster **do**
        Resample θ from the large pool (or fit θ if overdetermined).
      end for
      if the new candidate has smaller SSE **then**
        Replace the candidate f_i^n with the newly updated one; $j = j + 1$.
      else
        $failure\_count(i) = failure\_count(i) + 1$.
        if $failure\_count(i) \geq 100$ **then**
          $j = j + 1$.
        end if
      end if
    end while
    $\mathbb{L}_{rm-old} = \mathbb{L}_{rm}$.
  end while
  Recalculate p_i^n for $i = 1, \ldots, L$ by (4.4).
end if
return $\{f_i^n(x)\}$, $\{p_i^n\}$, $i = 1, \ldots, L$.

$$\nu^{KG-s,n}(x) = \mathbb{E}_j \left\{ \mathbb{E}_\omega^n \left[\max_{x' \in \mathcal{X}_s} \sum_{i=1}^{L} f_i^n(x') p_i^{n+1}(x) \middle| f_j^n = f^* \right] \right\} - \max_{x' \in \mathcal{X}_s} \sum_{i=1}^{L} f_i^n(x') p_i^n$$

$$= \sum_{j=1}^{L} \left[\int_\omega \max_{x' \in \mathcal{X}_s} \sum_{i=1}^{L} f_i^n(x') p_{i|j}^{n+1}(x, \omega) g(\omega) d\omega \right] p_j^n - \max_{x' \in \mathcal{X}_s} \sum_{i=1}^{L} f_i^n(x') p_i^n, \tag{4.10}$$

where $g(\omega) = \frac{1}{\sqrt{2\pi}\tilde{\sigma}} \exp\left(-\frac{\omega^2}{2\tilde{\sigma}^2}\right)$ is the probability density of measurement noise W^{n+1} (compensated with model discrepancy), and $p_{i|j}^{n+1}(x, \omega)$ is the posterior probability of f_i^n given $f^*(x) = f_j^n(x)$ and measurement noise $W^{n+1} = \omega$. We now show how to calculate $p_{i|j}^{n+1}(x, \omega)$.

We start by considering how to calculate p_i^{n+1} from p_i^n. Recall that after a minor

update, we have $f_i^{n+1}(x) \approx f_i^n(x)\ \forall x \in \mathcal{X}$. Then the likelihood of f_i^{n+1} is given by

$$\mathcal{L}_i^{n+1} = \prod_{j=1}^{n+1} \exp\left(-\frac{[\hat{y}^j - f_i^{n+1}(x^{j-1})]^2}{2\tilde{\sigma}^2}\right) \approx \prod_{j=1}^{n+1} \exp\left(-\frac{[\hat{y}^j - f_i^n(x^{j-1})]^2}{2\tilde{\sigma}^2}\right)$$
$$= \mathcal{L}_i^n \exp\left(-\frac{[\hat{y}^{n+1} - f_i^n(x^n)]^2}{2\tilde{\sigma}^2}\right).$$

Note that we use the adjusted variance $\tilde{\sigma}^2$ here. Therefore, p_i^{n+1} can be approximately calculated by

$$p_i^{n+1} = \frac{\mathcal{L}_i^{n+1}}{\sum_{j=1}^L \mathcal{L}_j^{n+1}} \approx \frac{\mathcal{L}_i^n \exp\left(-\frac{[\hat{y}^{n+1} - f_i^n(x^n)]^2}{2\tilde{\sigma}^2}\right)}{\sum_{j=1}^L \mathcal{L}_j^n \exp\left(-\frac{[\hat{y}^{n+1} - f_j^n(x^n)]^2}{2\tilde{\sigma}^2}\right)}$$
$$= \frac{p_i^n \exp\left(-\frac{[\hat{y}^{n+1} - f_i^n(x^n)]^2}{2\tilde{\sigma}^2}\right)}{\sum_{j=1}^L p_j^n \exp\left(-\frac{[\hat{y}^{n+1} - f_j^n(x^n)]^2}{2\tilde{\sigma}^2}\right)}. \tag{4.11}$$

Back to $p_{i|j}^{n+1}(x,\omega)$, we have $\hat{y}^{n+1} = f_j^n(x) + \omega$ in (4.11) according to the definition of $p_{i|j}^{n+1}(x,\omega)$, so

$$p_{i|j}^{n+1}(x,\omega) = \frac{p_i^n \exp\left(-\frac{[f_j^n(x) + \omega - f_i^n(x)]^2}{2\tilde{\sigma}^2}\right)}{\sum_{l=1}^L p_l^n \exp\left[-\frac{[f_j^n(x) + \omega - f_l^n(x)]^2}{2\tilde{\sigma}^2}\right]}.$$

Now we can calculate $\nu^{KG-s,n}(x)$ for any $x \in \mathcal{X}$. In order to find the alternative $x = \arg\max_{x \in \mathcal{X}} \nu^{KG-s,n}(x)$ which will be what we measure for the next experiment, we can calculate the gradient of $\nu^{KG-s,n}(x)$ using the algorithm proposed in [33] and then run gradient ascent with multistarts in $\mathcal{X}$.

4.7. Overall flowchart. Algorithm 4.2 summarizes our overall algorithm, which uses local approximation to generate some candidate functions, uses a certain policy to select alternatives to measure, updates the candidate functions and their probabilities, and estimates the optimal alternative $\hat{x}^N$ from the approximated true function. Besides continuous KG, we provide another three policies on selecting the next alternative to measure for empirical study in section 5, including the following:

1. KG: namely the vanilla KG on a discrete set of alternatives, where we uniformly sample a discrete set $\mathcal{X}_s$ from $\mathcal{X}$, and measure $\text{argmax}_{x \in \mathcal{X}_s} \nu^{KG-s,n}(x)$, where the argmax can be obtained by explicitly computing $\nu^{KG-s,n}(x)\ \forall x \in \mathcal{X}_s$ by (4.10);
2. Pure Exploration: selecting an x from $\mathcal{X}$ randomly from a uniform distribution;
3. Max-Var: selecting the x whose function value has the largest variance from a discrete set $\mathcal{X}_s$.

The flowchart for selecting an alternative using different policies is summarized in Algorithm 4.3.

5. Empirical results. We present empirical results on a variety of problems in this section. In all these problems, we show that our approximation framework coupled with the KG policy, especially the continuous KG, is very effective in estimating the optimal alternative as well as approximating the overall function. We

Algorithm 4.2. Overall flowchart.

Input: Budget for measurements: N; Alternative space: $\mathcal{X}$; The *policy* to select x^n.
Output: The estimated optimal alternative: $\hat{x}^N$.

Initialize $f_i^0(x)$ with random θ_i's. Set $p_1^0 = \cdots = p_L^0 = \frac{1}{L}$.
for $n = 0$ to $N-1$ **do**
  $x^n = select\_x(policy, \mathcal{X}, \{f_i^n\}, \{p_i^n\})$.
  Take a measurement at x^n, and the output is $\hat{y}^{n+1}$.
  major_update_flag$= 0$.
  for $i = 1, 2, \ldots, L$ **do**
    Add x^n to clustering result i in a recursive way.
    if x^n creates a new cluster in clustering result i **then**
      major_update_flag$= 1$.
    end if
  end for
  if *major_update_flag*$= 0$ **then**
    Recalculate probabilities $\{p_i^n\}$.
  end if
  if *major_update_flag*$= 1$ **or** $|\{i : p_i^n < \epsilon_p\}| > L/2$ **then**
    Run a major update of candidate functions, as in Algorithm 4.1.
  end if
end for
return $\hat{x}^N = \arg\max_{x \in \mathcal{X}} p_i^N f_i^N(x)$.

Algorithm 4.3. $select\_x(policy, \mathcal{X}, \{f_i^n\}, \{p_i^n\})$.

Input: Policy: for discrete x: KG, Max-Var, and for continuous x: Continuous KG, Pure Exploration; The alternative space: $\mathcal{X}$; the candidate functions: $\{f_i^n(x),\ i = 1, \ldots, L\}$; the probabilities: $\{p_i^n,\ i = 1, \ldots, L\}$.
Output: The alternative to measure next: x^n.

switch (*policy*)
case KG**:**
  $x^n = \text{argmax}_{x \in \mathcal{X}_s} \nu^{KG-s,n}(x)$.
case Continuous KG**:**
  $x^n = \text{argmax}_{x \in \mathcal{X}} \nu^{KG-s,n}(x)$.
case Pure Exploration**:**
  $x^n = uniform\_rand(\mathcal{X})$.
case Max-Var**:**
  $x^n = \text{argmax}_{x \in \mathcal{X}_s} \sum_{j=1}^{L} p_j^n [f_j^n(x) - \bar{f}^n(x)]^2$.
end switch
return x^n.

first demonstrate a two-dimensional problem in section 5.1, in which we illustrate the evolution of our estimated function $\bar{f}^n$ to show how it approaches the truth $f^*(x)$ as we take more measurements. We then study a multidimensional unimodal problem in section 5.2, which we have found to be particularly challenging since it produces a family of asymmetric functions that are hard to approximate. We then evaluate a multidimensional multimodal problem, which requires a decent approximation of multiple local regions. Finally, we present results on a real-world application that studies the carbon nanotube growth.

In short, we demonstrate results from the following aspects:

1. the visualization of how our estimation of the true function progresses;
2. the effectiveness of our belief model in approximating the unknown function;
3. the performance of different policies to choose the next alternative x^n, specifically how continuous KG performs compared with other competing policies defined in section 4.7;
4. the empirical rate of convergence on a variety of problems under noise-free and noisy settings.

Specifically, note that previous optimal learning approaches are either limited by the dimensionality of the problem (see, e.g., [21, 22, 59, 14]) or the availability of a globally accurate model (see, e.g., [49, 13, 34, 33]). Therefore, these methods cannot be effectively applied to our problem settings, where our focus is on multidimensional problems with only locally accurate models. As such, we use our local approximation approach as the common belief model, and show that based on this belief model a few alternative-selecting policies (defined in section 4.7) all show promising performance, while continuous KG is usually among the most effective.

We define two metrics to measure the performance:

1. *Opportunity cost (OC):* To evaluate the quality of the estimated optimal alternative $\hat{x}^n$, we define OC as the difference between the true function values at the true optimal x^* and at our estimated optimal $\hat{x}^n$, i.e.,

$$OC(n) = f^*(x^*) - f^*(\hat{x}^n) = \max_{x\in\mathcal{X}} f^*(x) - f^*\left(\operatorname*{argmax}_{x\in\mathcal{X}} \bar{f}^n(x)\right).$$

   We normalize OC by the range of $f^*(x)$, i.e.,

$$OC\%(n) = \frac{OC(n)}{\max_{x\in\mathcal{X}} f^*(x) - \min_{x\in\mathcal{X}} f^*(x)}.$$

2. *Mean squared error (MSE):* To evaluate our function approximation, we calculate the normalized MSE between $f^*(x)$ and $\bar{f}^n(x)$ on a sampled set $\mathcal{X}_s$, defined as

$$MSE(n) = \frac{\sqrt{\frac{1}{|\mathcal{X}_s|}\sum_{x\in\mathcal{X}_s} |f^*(x) - \bar{f}^n(x)|^2}}{\max_{x\in\mathcal{X}} f^*(x) - \min_{x\in\mathcal{X}} f^*(x)}. \tag{5.1}$$

Also note that below we run the experiments under noise-free and noisy settings, where the noise level is defined as the ratio of the standard deviation of the measurement noise (i.e., σ) to the range of the true function, namely $\frac{\sigma}{\max_{x\in\mathcal{X}} f^*(x) - \min_{x\in\mathcal{X}} f^*(x)}$. When we say the noise is 20%, we mean this ratio is 0.2.

5.1. Visualization of function approximation. We start with a two-dimensional problem such that we can visualize the true function $f^*(x)$ versus our estimated function $\bar{f}^N(x)$ at certain times. The true function is defined as

$$f^*(x) = 100x_1 \cdot x_2^3 \cdot \exp\left(-x_1^2 - 0.5x_2^2\right). \tag{5.2}$$

The shape of $f^*(x)$ is shown in the left column of Figure 4, which has two local maxima and two local minima. We use the following quadratic function as the local approximation model:

$$g(x;\theta) = \theta_1 x_1^2 + \theta_2 x_2^2 + \theta_3 x_1 x_2 + \theta_4 x_1 + \theta_5 x_2 + \theta_6.$$

Note that for any local region, the quadratic function is not a perfect fit of the true function, but our goal is to capture the general shape of the true function and find the true x^*. Figure 4 shows the evolution of the approximation $\bar{f}^n$ at certain time points under the KG policy with 20% noise. As we can see, after about 20 measurements, we are able to give a decent estimate of x^*; after 35 measurements, we are able to recognize that there exist two local maximums and two local minimums; and after 50 measurements, our estimate gets very close to the truth.

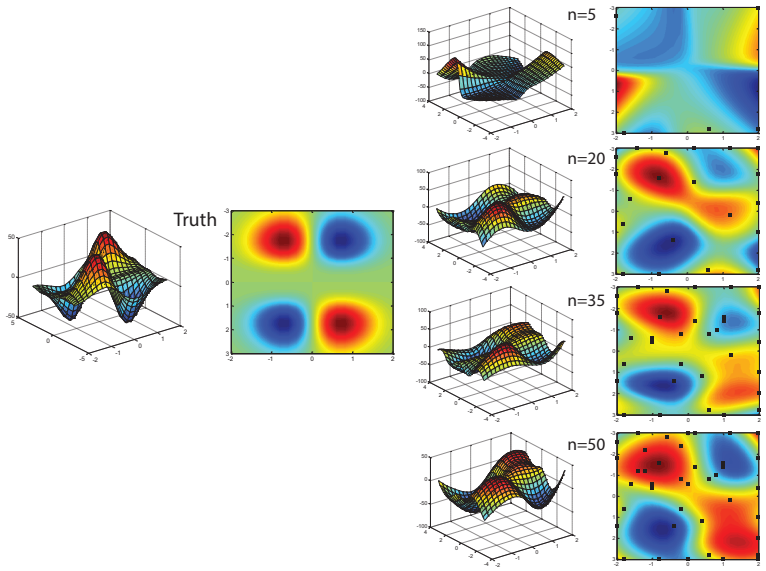

FIG. 4. *Visualization of our estimate $\bar{f}^n$ vs. the truth $f^*(x)$ (showing three dimensions and the heat map). The black dots in the right column show where we have measured.*

5.2. Multidimensional unimodal problem. In this section, we focus on a multidimensional benchmark function, known as the *newsvendor problem*, defined as

$$f^*(x) = \sum_{i=1}^{k} \eta_{1,i} \mathbb{E}\left[\min\left(x_i, \left(D - \sum_{j=1}^{i-1} x_j \right)^+ \right) \right] - \sum_{i=1}^{k} \eta_{2,i} x_i.$$

The newsvendor function produces a wide range of unimodal asymmetric functions, exhibiting shapes that cannot be globally approximated by linear or quadratic functions.

In our experiment, we set $k = 4$ and thus have a four-dimensional function as the truth. We try two different local models: (1) the linear function, i.e., $g_{lin}(x;\theta) = \sum_{i=1}^{k} \theta_i x_i + \theta_{k+1}$, and (2) the quadratic function without cross terms, i.e., $g_{quad}(x;\theta) = \sum_{i=1}^{k} \theta_i x_i^2 + \sum_{i=1}^{k} \theta_{k+i} x_i + \theta_{2k+1}$.

Figure 5 shows how OC and MSE decrease as we take more measurements on the four-dimensional problem, using the linear function as the local approximation. The noise level is 20%. As we can see, the KG and continuous KG policies have steady OC decrease over time as we take measurements and they outperform both Max-Var and Pure Exploration. For MSE, KG and Max-Var are among the best, with continuous KG being slightly worse.

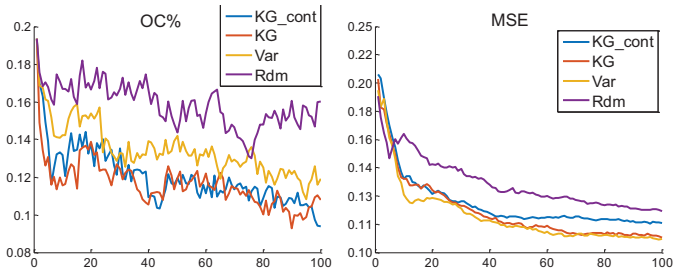

FIG. 5. *OC% and MSE of a four-dimensional newsvendor problem, approximated by linear functions. x-axis: number of measurements; noise level:* 20%.

Figure 6 shows how OC and MSE change over time on the same four-dimensional problem, using quadratic functions as the local model, with no noise and 20% of noise, respectively. Note that with the parametric functions being accurate only in small local regions, it is not trivial to solve the problem with zero noise. In the noise-free setting (left two images in Figure 6), we can see that the OC and MSE both decrease rapidly, especially with continuous KG. However, with 20% of noise (right two images), OC and MSE hardly drop even after 100 measurements, highlighting the challenge brought by the noise when there is no globally accurate model.

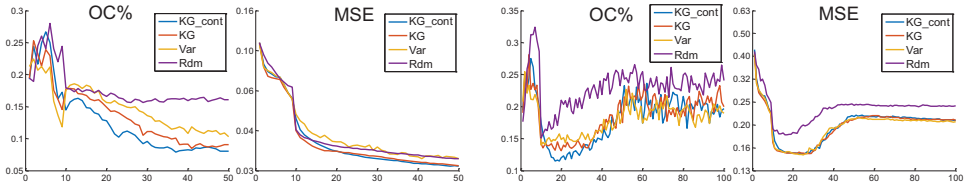

FIG. 6. *Results of the four-dimensional newsvendor problem, approximated by quadratic functions. x-axis: number of measurements. From left to right:* (1) *OC% under no noise;* (2) *MSE under no noise;* (3) *OC% under* 20% *noise;* (4) *MSE under* 20% *noise.*

Note that from Figures 5 and 6, we see that using the linear local models achieves better results than using quadratic models under the same noise setting. This is not surprising since the newsvendor function is closer to being linear than quadratic in most of the regions. This result highlights the challenge of overfitting using a local model with too many parameters.

Additionally, Figure 7 shows OC and MSE against time on a *six*-dimensional newsvendor problem, approximated by linear functions under 20% noise. Compared with the four-dimensional case in Figure 5, we can see continuous KG has more advantage in high-dimensional settings, as designed.

5.3. Multidimensional multimodal problem. In this section, we move to a multidimensional multimodal function, which requires a decent approximation in a number of local regions to approximate. We use the extended version of the two-dimensional case in section 5.1, defined as

(5.3)
$$f^*(x) = 100x_1 \cdot x_2^3 \cdot \exp\left(-x_1^2 - 0.5x_2^2\right) + 50x_3 \cdot x_4^3 \cdot \exp\left(-0.5x_3^2 - x_4^2 + 0.5x_4\right).$$

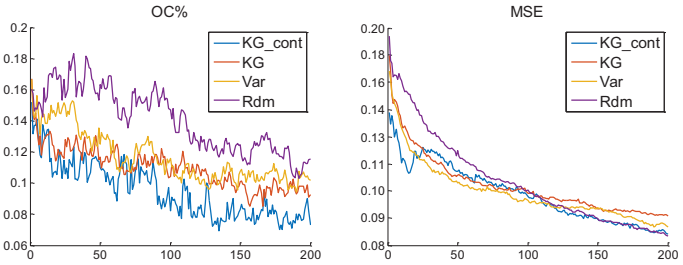

FIG. 7. *Results of the six-dimensional newsvendor problem, approximated by linear functions. x-axis: number of measurements; noise:* 20%.

Since the two-dimensional function in section 5.1 has (at least) four local regions, the four-dimensional $f^*(x)$ has at least 16 local regions. We use the quadratic function without cross terms as the local model, i.e., $g(x;\theta) = \sum_{i=1}^{4} \theta_i x_i^2 + \sum_{i=1}^{4} \theta_{4+i} x_i + \theta_9$.

Figure 8 shows the OC under 0 noise and 20% noise, respectively. As we can see, in both cases, continuous KG outperforms the other three policies and its OC steadily decreases over time, which proves the effectiveness of our algorithm with the continuous KG policy in finding the optimal alternative.

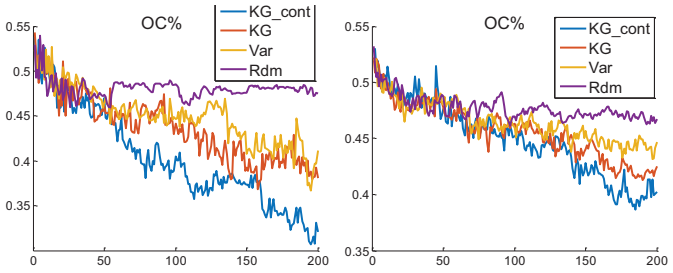

FIG. 8. *OC% of four-dimensional test function. x-axis: number of measurements; left: no noise; right:* 20% *noise.*

5.4. An application: Carbon nanotube growth. We now consider an application in optimizing carbon nanotube (CNT) growth using data obtained from the Air Force Research Laboratory's ARES project [50]. ARES is an autonomous research system that autonomously synthesizes CNTs, characterizes growth via an in situ Raman spectrometer, and plans the next experiment based on experimental results. Growth of CNTs is performed by chemical vapor deposition in which feedstock gasses C_2H_4, H_2, and CO_2 are introduced at specified partial concentrations into a growth chamber that contains catalyst nanoparticles [44]. A specific temperature, ranging between $600K$ and $1200K$, is maintained during CNT growth. Through a system of various gas and surface chemical reactions, CNTs nucleate and grow on the catalyst.

The myriad kinetic processes involved include gas phase and surface reactions [53], catalyst nanoparticle dynamics such as dewetting, ripening [48], and poisoning [58, 53], and phase change phenomena [38]. The exact nature of such processes is poorly understood, and as such, capturing a complete global picture of the dynamics, as a function of temperature and feedstock gas concentrations, is a difficult task. As

a result, we need to use simpler models to approximate the growth of CNTs in local regions of the design space.

In this problem, our goal is to maximize the concentration of CNT after a synthesis process of 300 seconds. The concentrations of C_2H_4, H_2, and CO_2, together with the temperature, constitute a four-dimensional design space. For the purpose of evaluating our algorithm, we define the true function $f^*(x)$ by fitting from the ARES data (see the supplementary materials, linked from the main article webpage, for more detail on the formulation of $f^*(x)$), where f^* is the CNT concentration at the end of a *noise-free* synthesis given a design vector x. We approximate $f^*(x)$ using a linear function $g(x;\theta)$ as the local model, namely $g(x;\theta) = \sum_{i=1}^{4} \theta_i x_i + \theta_5$, which is a natural and general model to locally approximate many unknown smooth functions.

We run simulations under 0% and 20% of noise, respectively. The results of OC% and MSE of both cases are shown in Figure 9. As we can see, using the local linear models returns very promising results, especially when we use the continuous KG policy, in which the OC% decreases to a relatively low level after 200 experiments.

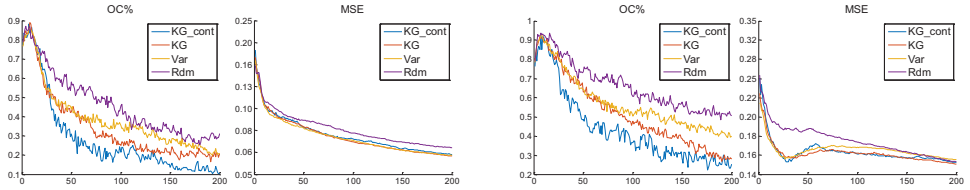

FIG. 9. *OC% and MSE using the linear function $g_{lin}(x;\theta_j)$ locally, with* 0% *(left) and* 20% *(right) noise. x-axis: number of measurements.*

6. Conclusion. Optimal learning for nonlinear parametric belief models, especially in a multidimensional continuous alternative space, has been a challenging problem. Previous methods usually assume we have a parametric function $f(x;\theta)$ that can describe the true function globally, where the expression of f is known but θ is unknown. In this category, the authors of [13] start from a discrete set of x's and a small set of θ's and develop the knowledge gradient with discrete priors (KGDP) model. The authors of [34] propose a resampling algorithm in the θ space to adaptively learn more probable θ's and thus extend KGDP to a much larger θ space. Based on this, the authors of [33] present a gradient-based algorithm that works in a multidimensional continuous x space. These methods show strong performance, even for 10- or 20-dimensional x and θ spaces [33]. However, they all require that the true function be well described by a parametric function $f(x;\theta)$ globally. In many cases, the true function can be too complicated to model exactly, while a simpler model that can well describe a local region might exist.

In this paper, we focus on the optimal learning problem in which a global parametric belief model is unavailable, but the true function at any local region can be well described by some nonlinear parametric model. We use DC-RBF [37] and k-means to cluster data and identify local regions, and we apply normalized RBF to aggregate local functions as a possible global approximation of the true function. We use a family of global approximations as our candidate functions, each corresponding to a different clustering result and consisting of different locally fitted models, and then we adopt the KG policy for discrete priors proposed and studied in [13, 34, 33] to select the alternatives to measure out of a multidimensional continuous alternative space.

Experiments on some synthetic test functions and a real materials science application show the strong performance of our algorithm.

There are two points worth noting. First, our algorithm is still a nonparametric algorithm. Hence, as with other nonparametric algorithms, it may suffer from the curse of dimensionality. Specifically, if there are too many local regions, especially in a high-dimensional alternative space, and if the measurement budget is not large enough, our estimate of the true function, $\bar{f}^n$, might be very different from the ground-truth f^*. This can be seen in the OC results of the first several measurements in Figure 9. Second, increasing the complexity of the local model does not necessarily help with the overall performance, as shown in Figures 5 and 6. Actually, our method may favor a simpler local model, especially when each local region does not contain sufficient data points. Overall, our algorithm provides a powerful tool for optimal learning problems with multidimensional continuous alternative spaces, and it works most effectively when there are not too many local regions such that the local models can be well approximated with a limited budget.

Appendix A. Some extra guidance on using the algorithm. We list some detailed heuristics and suggestions in using the algorithm, which would be overdetailed had we introduced them in the main text.

A.1. Suggestions on the local model.

A.1.1. How to pick a local model. In many real-world applications, the experts may already have some domain knowledge on what the functions may look like, which many times would provide a simplified model that can be locally accurate. The application in section 5.4 is an example in which scientists have studied and constructed locally accurate models. The existing local model is also what we started with, but it had too many unknown parameters (i.e., dimensionality of θ is too large) and did not yield satisfactory results in our setting, where a limited number of measurements are available (see the supplementary materials for more details). Then we turned to the linear model and it performed amazingly well, which has helped us realize many properties of the CNT growth.

In short, our suggestion is that if there is some existing model based on domain knowledge, it is worth giving it a try unless the number of unknown variables is large; otherwise, we recommend using low-order models such as linear or quadratic models.

A.1.2. How to pick the range for the radii. Despite the fact that our approach is already an improvement compared with specifying a fixed radius in DC-RBF, it could still be tricky to decide the range $[D_{T,min}, D_{T,max}]$ for the radii. To start with, we should note that it is better to set $D_{T,max}$ to larger values. If a radius is too large, the candidate function it generates will not fit the measurements well and thus end up with a low probability, which can be naturally fixed by resampling the radius; if a radius is too small, the probability of its candidate function will be high, but the local models would be overfitted.

For $D_{T,min}$, our advice is that we should set it according to the size of $\mathcal{X}$ and our budget N. Note that we can get an estimate on the number of local regions by comparing the size of $\mathcal{X}$ with that of the local regions, which is controlled by the radius, and this ratio should be on the same order of magnitude as the budget. Moreover, if there is domain knowledge on the model, we can take it into account, too.

A.1.3. Scale the x space. If possible, it would be helpful to scale the x space such that the region a local model covers is roughly a hypersphere. This is because

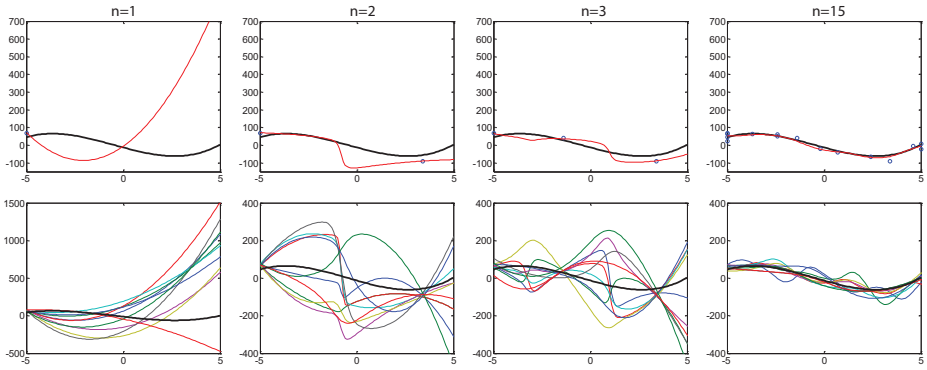

FIG. 10. *Evolution of $\bar{f}$ (red curves in the first row) and candidate functions (thin nonblack curves in the second row) under KG policy in a one-dimensional problem under 20% noise (the noise level is defined in section 5); i.e., the red line in the first row is the weighted sum of the thin curves in the line below. The thick black curve is the truth, given by $f^*(x) = (x+0.5)^3 - 30x - 13$, approximated by quadratic functions, i.e., $g(x;\theta) = \theta_1 x^2 + \theta_2 x + \theta_3$. The blue dots are where we take measurements. Note that images in the first row have the same scale, while the last three images in the second row have the same scale.*

both DC-RBF clustering and k-means use the distance as the metric, and therefore the clusters would resemble a hypersphere. By scaling properly, we can better utilize the capability of a local model and therefore learn the optimal solution more quickly. However, note that this is not a requirement; the only downside of not doing so is that we would only potentially need more local regions and more measurements.

A.2. Heuristics in refitting local functions. There are two heuristics in implementing the logic of refitting θ_{ij}. First, to make the candidate function f_i^n smooth, we can include some nearby data points of other clusters when calculating $SSE(\theta)$ but assign smaller weights to these data. For example, for cluster j, let $\mathcal{X}'_j = \left\{x \in \{x^0, \ldots, x^n\} : x \notin \text{ cluster } j, \text{and } ||x - c_{ij}|| \leq 1.5D_T\right\}$, and let $\hat{y}'(x)$ be the measurement result associated with any $x \in \mathcal{X}'_j$. Let $SSE(\theta) = \sum_{i=1}^{k_j} \left[g(x_j^{(i)};\theta) - \hat{y}_j^{(i)}\right]^2 + \frac{1}{10}\sum_{x\in\mathcal{X}'_j}\left[g(x;\theta) - \hat{y}'(x)\right]^2$, and then calculate $p(\theta)$ according to (4.6). This heuristic is also useful when cluster j contains too few data points.

Another heuristic is that if there are enough data points in a cluster such that the fitting problem is overdetermined, and if solving the fitting problem is efficient (for example, if $g(x;\theta)$ is linear in θ, then θ can be found using linear regression), then we can calculate θ_{ij} directly instead of sampling it from the large pool. However, calculating θ directly in an underdetermined case is not encouraged, especially if the fitting algorithm always returns some θ out of a small range. This is because our candidate functions are supposed to have various shapes and capture the distribution of possible truths. Intuitively, if a local region contains only a few data but the candidate functions $f_i^n(x)$ containing this region all use the same $g(x;\theta_j^n)$, we may overlook the uncertainty in this region and thus may not measure any point in this region. One example is given in Figure 10, which shows results of a one-dimensional problem under the KG policy proposed in section 4.6. When we only have one measurement as in "$n = 1$" in Figure 10, the fitting results using our resampling logic include a variety of curves, and the uncertainty on the right-hand side is the largest. Hence, the KG policy then measures an alternative on the right-hand side, as in "$n = 2$." Next, KG

measures some x in the middle as in "$n = 3$." However, if we fit L curves using only x^0 in "$n = 1$" rather than resample from the small pool, the fitted L curves might be too close to reflect our uncertainty on the right-hand side.

REFERENCES

[1] S. Andradóttir, *A global search method for discrete stochastic optimization*, SIAM J. Optim., 6 (1996), pp. 513–530, https://doi.org/10.1137/0806027.

[2] S. Andradóttir and A. A. Prudius, *Adaptive random search for continuous simulation optimization*, Naval Res. Logist., 57 (2010), pp. 583–604, https://doi.org/10.1002/nav.20422.

[3] J.-Y. Audibert and S. Bubeck, *Best arm identification in multi-armed bandits*, in Proceedings of the 23rd Conference on Learning Theory (COLT 2010).

[4] P. Auer, N. Cesa-Bianchi, and P. Fischer, *Finite-time analysis of the multiarmed bandit problem*, Mach. Learn., 47 (2002), pp. 235–256, https://doi.org/10.1023/A:1013689704352.

[5] E. Barut and W. B. Powell, *Optimal learning for sequential sampling with non-parametric beliefs*, J. Global Optim., 58 (2014), pp. 517–543, https://doi.org/10.1007/s10898-013-0050-5.

[6] V. E. Bening and V. Y. Korolev, *Generalized Poisson Models and Their Applications in Insurance and Finance*, Walter de Gruyter, Berlin, 2002.

[7] L. Bianchi, M. Dorigo, L. M. Gambardella, and W. J. Gutjahr, *A survey on metaheuristics for stochastic combinatorial optimization*, Nat. Comput., 8 (2009), pp. 239–287, https://doi.org/10.1007/s11047-008-9098-4.

[8] A. J. Booker, J. E. Dennis, Jr., P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset, *A rigorous framework for optimization of expensive functions by surrogates*, Struct. Multidiscip. Optim., 17 (1999), pp. 1–13, https://doi.org/10.1007/BF01197708.

[9] J. Branke, S. Meisel, and C. Schmidt, *Simulated annealing in the presence of noise*, J. Heuristics, 14 (2007), pp. 627–654, https://doi.org/10.1007/s10732-007-9058-7.

[10] D. S. Broomhead and D. Lowe, *Multivariable functional interpolation and adaptive networks*, Complex Systems, 2 (1988), pp. 321–355.

[11] M. D. Buhmann, *Radial basis functions*, Acta Numer., 9 (2000), pp. 1–38, https://doi.org/10.1017/S0962492900000015.

[12] C. H. Chen, J. Lin, E. Yücesan, and S. E. Chick, *Simulation budget allocation for further enhancing the efficiency of ordinal optimization*, Discrete Event Dyn. Syst., 10 (2000), pp. 251–270, https://doi.org/10.1023/A:1008349927281.

[13] S. Chen, K.-R. G. Reyes, M. K. Gupta, M. C. McAlpine, and W. B. Powell, *Optimal learning in experimental design using the knowledge gradient policy with application to characterizing nanoemulsion stability*, SIAM/ASA J. Uncertain. Quantif., 3 (2015), pp. 320–345, https://doi.org/10.1137/140971129.

[14] B. Cheng, A. Jamshidi, and W. B. Powell, *Optimal learning with a local parametric belief model*, J. Global Optim., 63 (2015), pp. 401–425, https://doi.org/10.1007/s10898-015-0299-y.

[15] S. E. Chick, J. Branke, and C. Schmidt, *Sequential sampling to myopically maximize the expected value of information*, INFORMS J. Comput., 22 (2010), pp. 71–80, https://doi.org/10.1287/ijoc.1090.0327.

[16] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Geometry of interpolation sets in derivative free optimization*, Math. Program., 111 (2008), pp. 141–172, https://doi.org/10.1007/s10107-006-0073-5.

[17] M. Denny and S. Gaines, *Chance in Biology: Using Probability to Explore Nature*, http://www.scopus.com/inward/record.url?eid=2-s2.0-84890580418&partnerID=tZOtx3y1, 2011.

[18] C. Elster and A. Neumaier, *A trust region method for the optimization of noisy functions*, Computing, 58 (1997), pp. 31–46.

[19] J. Fan, *Design-adaptive nonparametric regression*, J. Amer. Statist. Assoc., 87 (1992), pp. 998–1004, https://doi.org/10.2307/2290637.

[20] J. Fan and I. Gijbels, *Local Polynomial Modelling and Its Applications*, Chapman & Hall, London, 1996.

[21] P. I. Frazier, W. B. Powell, and S. Dayanik, *A knowledge-gradient policy for sequential information collection*, SIAM J. Control Optim., 47 (2008), pp. 2410–2439, https://doi.org/10.1137/070693424.

[22] P. I. Frazier, W. B. Powell, and S. Dayanik, *The knowledge-gradient policy for correlated normal beliefs*, INFORMS J. Comput., 21 (2009), pp. 599–613, https://doi.org/10.1287/

ijoc.1080.0314.
[23] M. C. Fu, *Feature article: Optimization for simulation: Theory vs. practice*, INFORMS J. Comput., 14 (2002), pp. 192–215, https://doi.org/10.1287/ijoc.14.3.192.113.
[24] M. C. Fu, *Stochastic gradient estimation*, in Handbook of Simulation Optimization, Internat. Ser. Oper. Res. Management Sci. 216, Springer, New York, 2015, pp. 105–147, https://doi.org/10.1007/978-1-4939-1384-8_5.
[25] M. C. Fu, J. Q. Hu, C. H. Chen, and X. Xiong, *Simulation allocation for determining the best design in the presence of correlated sampling*, INFORMS J. Comput., 19 (2007), pp. 101–111, https://doi.org/10.1287/ijoc.1050.0141.
[26] A. Garcia, *Probability, Statistics, and Random Processes for Electrical Engineering*, Prentice-Hall, Upper Saddle River, NJ, 2008.
[27] D. Goldsman, *A practical guide to ranking and selection methods*, in The Operations Research Revolution, Institute for Operations Research and the Management Sciences, Catonsville, MD, pp. 89–110.
[28] D. Goldsman, S.-H. Kim, W. Marshall, and B. Nelson, *Ranking and selection for steady-state simulation: Procedures and perspectives*, INFORMS J. Comput., 14 (2002), pp. 2–19, https://doi.org/10.1109/WSC.2000.899762.
[29] L. A. Hannah, D. M. Blei, and W. B. Powell, *Dirichlet process mixtures of generalized linear models*, J. Mach. Learn. Res., 12 (2011), pp. 1923–1953.
[30] J. M. Harrison, N. B. Keskin, and A. Zeevi, *Bayesian dynamic pricing policies: Learning and earning under a binary prior distribution*, Manag. Sci., 58 (2012), pp. 570–586, https://doi.org/10.1287/mnsc.1110.1426.
[31] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., Springer, New York, 2009, https://doi.org/10.1007/b94608.
[32] D. He, S. E. Chick, and C.-H. Chen, *Opportunity cost and OCBA selection procedures in ordinal optimization for a fixed number of alternative systems*, IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., 37 (2007), pp. 951–961, https://doi.org/10.1109/TSMCC.2007.900656.
[33] X. He, Y. Hu, and W. B. Powell, *Optimal learning for nonlinear parametric belief models over multidimensional continuous spaces*, SIAM J. Optim., 28 (2018), pp. 2945–2974, https://doi.org/10.1137/17M1129301.
[34] X. He and W. Powell, *Optimal learning for stochastic optimization with nonlinear parametric belief models*, SIAM J. Optim., 28 (2018), pp. 2327–2359, https://doi.org/10.1137/16M1073042.
[35] L. Hong and B. L. Nelson, *A brief introduction to optimization via simulation*, in Proceedings of the Winter Simulation Conference, 2009, pp. 75–85.
[36] L. J. Hong and B. L. Nelson, *Discrete optimization via simulation using COMPASS*, Oper. Res., 54 (2006), pp. 115–129, https://doi.org/10.1287/opre.1050.0237.
[37] A. A. Jamshidi and W. B. Powell, *A recursive local polynomial approximation method using Dirichlet clouds and radial basis functions*, SIAM J. Sci. Comput., 38 (2016), pp. B619–B644, https://doi.org/10.1137/15M1008592.
[38] A. P. J. Jansen, R. Agrawal, and L. Spanu, *Thermodynamics and kinetics of carbon deposits on cobalt: A combined density functional theory and kinetic Monte Carlo study*, Phys. Chem. Chem. Phys., 18 (2016), pp. 28515–28523, https://doi.org/10.1039/C6CP04719J.
[39] R. D. Jones, Y. C. Lee, C. W. Barnes, G. W. Flake, K. Lee, P. S. Lewis, and S. Qian, *Function approximation and time series prediction with neural networks*, in Proceedings of the IJCNN International Joint Conference on Neural Networks, Vol. 1, 1990, pp. 649–665, https://doi.org/10.1109/IJCNN.1990.137644.
[40] L. P. Kaelbling, *Learning in Embedded Systems*, MIT Press, Cambridge, MA, 1993.
[41] A. I. Khuri and S. Mukhopadhyay, *Response surface methodology*, WIREs Comp. Stats, 2 (2010), pp. 128–149, https://doi.org/10.1002/wics.73.
[42] S.-H. Kim and B. L. Nelson, *Selecting the best system*, in Handbooks in Operations Research and Management Science, Vol. 13, Elsevier, Amsterdam, 2006, pp. 501–534, https://doi.org/10.1016/S0927-0507(06)13017-0.
[43] S.-H. Kim and B. L. Nelson, *Recent advances in ranking and selection*, in Proceedings of the Winter Simulation Conference, 2007, pp. 162–172, https://doi.org/10.1109/WSC.2007.4419598.
[44] M. Kumar and Y. Ando, *Chemical vapor deposition of carbon nanotubes: A review on growth mechanism and mass production*, J. Nanosci. Nanotechnol., 10 (2010), pp. 3739–3758.
[45] H. Kushner and D. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, Springer-Verlag, New York, Berlin, 1978.

[46] A. Lyon, *Why are normal distributions normal?*, British J. Philos. Sci., 65 (2014), pp. 621–649.
[47] M. R. Mes, W. B. Powell, and P. I. Frazier, *Hierarchical knowledge gradient for sequential sampling*, J. Mach. Learn. Res., 12 (2011), pp. 2931–2974.
[48] M. Moseler, F. Cervantes-Sodi, S. Hofmann, G. Csányi, and A. C. Ferrari, *Dynamic catalyst restructuring during carbon nanotube growth*, ACS Nano, 4 (2010), pp. 7587–7595, https://doi.org/10.1021/nn102118y.
[49] D. M. Negoescu, P. I. Frazier, and W. B. Powell, *The knowledge-gradient algorithm for sequencing experiments in drug discovery*, INFORMS J. Comput., 23 (2011), pp. 346–363, https://doi.org/10.1287/ijoc.1100.0417.
[50] P. Nikolaev, D. Hooper, F. Webber, R. Rao, K. Decker, M. Krein, J. Poleski, R. Barto, and B. Maruyama, *Autonomy in materials research: A case study in carbon nanotube growth*, Npj Computational Materials, 2 (2016), 16031, https://doi.org/10.1038/npjcompumats.2016.31.
[51] S. Ólafsson, *Metaheuristics*, in Handbooks in Operations Research and Management Science, Vol. 13, Elsevier, Amsterdam, 2006, pp. 633–654, https://doi.org/10.1016/S0927-0507(06)13021-2.
[52] W. B. Powell and I. O. Ryzhov, *Optimal Learning*, Wiley Ser. Probab. Stat. 841, John Wiley & Sons, Hoboken, NJ, 2012.
[53] A. Puretzky, D. Geohegan, S. Jesse, I. Ivanov, and G. Eres, *In situ measurements and modeling of carbon nanotube array growth kinetics during chemical vapor deposition*, Appl. Phys. A, 81 (2005), pp. 223–240, https://doi.org/10.1007/s00339-005-3256-7.
[54] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. Kevin Tucker, *Surrogate-based analysis and optimization*, Prog. Aerosp. Sci., 41 (2005), pp. 1–28, https://doi.org/10.1016/j.paerosci.2005.02.001.
[55] R. G. Regis and C. A. Shoemaker, *A stochastic radial basis function method for the global optimization of expensive functions*, INFORMS J. Comput., 19 (2007), pp. 497–509, https://doi.org/10.1287/ijoc.1060.0182.
[56] R. Y. Rubinstein and A. Shapiro, *Optimization of static simulation models by the score function method*, Math. Comput. Simulation, 32 (1990), pp. 373–392, https://doi.org/10.1016/0378-4754(90)90142-6.
[57] D. Russo and B. Van Roy, *Learning to optimize via posterior sampling*, Math. Oper. Res., 39 (2014), pp. 1221–1243, https://doi.org/10.1287/moor.2014.0650.
[58] C. Schünemann, F. Schäffel, A. Bachmatiuk, U. Queitsch, M. Sparing, B. Rellinghaus, K. Lafdi, L. Schultz, B. Büchner, and M. H. Rümmeli, *Catalyst poisoning by amorphous carbon during carbon nanotube growth: Fact or fiction?*, ACS Nano, 5 (2011), pp. 8928–8934, https://doi.org/10.1021/nn2031066.
[59] W. Scott, P. Frazier, and W. B. Powell, *The correlated knowledge gradient for simulation optimization of continuous parameters using Gaussian process regression*, SIAM J. Optim., 21 (2011), pp. 996–1026, https://doi.org/10.1137/100801275.
[60] S. Singh, T. Jaakkola, M. L. Littman, C. Szepes, and A. S. Hu, *Convergence results for single-step on-policy reinforcement-learning algorithms*, Mach. Learn., 39 (2000), pp. 287–308, https://doi.org/10.1023/A:1007678930559.
[61] J. Spall, *Introduction to Stochastic Search and Optimization*, 1st ed., Wiley, New York, 2003.
[62] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.
[63] J. Villemonteix, E. Vazquez, and E. Walter, *An informational approach to the global optimization of expensive-to-evaluate functions*, J. Global Optim., 44 (2009), pp. 509–534, https://doi.org/10.1007/s10898-008-9354-2.