A Tutorial on Bayesian Optimization

Peter I. Frazier

July 10, 2018

Abstract

Bayesian optimization is an approach to optimizing objective functions that take a long time (minutes or hours) to evaluate. It is best-suited for optimization over continuous domains of less than 20 dimensions, and tolerates stochastic noise in function evaluations. It builds a surrogate for the objective and quantifies the uncertainty in that surrogate using a Bayesian machine learning technique, Gaussian process regression, and then uses an acquisition function defined from this surrogate to decide where to sample. In this tutorial, we describe how Bayesian optimization works, including Gaussian process regression and three common acquisition functions: expected improvement, entropy search, and knowledge gradient. We then discuss more advanced techniques, including running multiple function evaluations in parallel, multi-fidelity and multi-information source optimization, expensive-to-evaluate constraints, random environmental conditions, multi-task Bayesian optimization, and the inclusion of derivative information. We conclude with a discussion of Bayesian optimization software and future research directions in the field. Within our tutorial material we provide a generalization of expected improvement to noisy evaluations, beyond the noise-free setting where it is more commonly applied. This generalization is justified by a formal decision-theoretic argument, standing in contrast to previous ad hoc modifications.

1 Introduction

Bayesian optimization (BayesOpt) is a class of machine-learning-based optimization methods focused on solving the problem

$$\max_{x \in A} f(x), \tag{1}$$

where the feasible set and objective function typically have the following properties:

- The input x is in $\mathbb{R}^d$ for a value of d that is not too large. Typically $d \leq 20$ in most successful applications of BayesOpt.
- The feasible set A is a simple set, in which it is easy to assess membership. Typically A is a hyper-rectangle $\{x \in \mathbb{R}^d : a_i \leq x_i \leq b_i\}$ or the d-dimensional simplex $\{x \in \mathbb{R}^d : \sum_i x_i = 1\}$. Later (Section 5) we will relax this assumption.
- The objective function f is continuous. This will typically be required to model f using Gaussian process regression.
- f is "expensive to evaluate" in the sense that the number of evaluations that may be performed is limited, typically to a few hundred. This limitation typically arises because each evaluation takes a substantial amount of time (typically hours), but may also occur because each evaluation bears a monetary cost (e.g., from purchasing cloud computing power, or buying laboratory materials), or an opportunity cost (e.g., if evaluating f requires asking a human subject questions who will tolerate only a limited number).
- f lacks known special structure like concavity or linearity that would make it easy to optimize using techniques that leverage such structure to improve efficiency. We summarize this by saying f is a "black box."
- When we evaluate f, we observe only $f(x)$ and no first- or second-order derivatives. This prevents the application of first- and second-order methods like gradient descent, Newton's method, or quasi-Newton methods. We refer to problems with this property as "derivative-free".
- Through most of the article, we will assume $f(x)$ is observed without noise. Later (Section 5) we will allow $f(x)$ to be obscured by stochastic noise. In almost all work on Bayesian optimization, noise is assumed independent across evaluations and Gaussian with constant variance.

- Our focus is on finding a *global* rather than local optimum.

We summarize these problem characteristics by saying that BayesOpt is designed for black-box derivative-free global optimization.

The ability to optimize expensive black-box derivative-free functions makes BayesOpt extremely versatile. Recently it has become extremely popular for tuning hyperparameters in machine learning algorithms, especially deep neural networks (Snoek et al., 2012). Over a longer period, since the 1960s, BayesOpt has been used extensively for designing engineering systems (Močkus, 1989; Jones et al., 1998; Forrester et al., 2008). BayesOpt has also been used to choose laboratory experiments in materials and drug design (Negoescu et al., 2011; Frazier and Wang, 2016; Packwood, 2017), in calibration of environmental models (Shoemaker et al., 2007), and in reinforcement learning (Brochu et al., 2009; Lizotte, 2008; Lizotte et al., 2007).

BayesOpt originated with the work of Kushner (Kushner, 1964), Zilinskas (Žilinskas, 1975; Močkus et al., 1978), and Močkus (Močkus, 1975; Močkus, 1989), but received substantially more attention after that work was popularized by Jones et al. (1998) and their work on the Efficient Global Optimization (EGO) algorithm. Following Jones et al. (1998), innovations developed in that same literature include multi-fidelity optimization (Huang et al., 2006; Sóbester et al., 2004), multi-objective optimization (Keane, 2006; Knowles, 2006; Močkus and Močkus, 1991), and a study of convergence rates (Calvin, 1997; Calvin and Žilinskas, 2000; Calvin and Žilinskas, 2005; Calvin and Žilinskas, 1999). The observation made by Snoek et al. (2012) that BayesOpt is useful for training deep neural networks sparked a surge of interest within machine learning, with complementary innovations from that literature including multi-task optimization (Swersky et al., 2013; Toscano-Palmerin and Frazier, 2018), multi-fidelity optimization specifically aimed at training deep neural networks (Klein et al., 2016), and parallel methods (Ginsbourger et al., 2007, 2010; Wang et al., 2016a; Wu and Frazier, 2016). Gaussian process regression, its close cousin kriging, and BayesOpt have also been studied recently in the simulation literature (Kleijnen et al., 2008; Salemi et al., 2014; Mehdad and Kleijnen, 2018) for modeling and optimizing systems simulated using discrete event simulation.

There are other techniques outside of BayesOpt that can be used to optimize expensive derivative-free black-box functions. While we do not review methods from this literature here in detail, many of them have a similar flavor to BayesOpt methods: they maintain a surrogate that models the objective function, which they use to choose where to evaluate (Booker et al., 1999; Regis and Shoemaker, 2007b,a, 2005). This more general class of methods is often called "surrogate methods." Bayesian optimization distinguishes itself from other surrogate methods by using surrogates developed using Bayesian statistics, and in deciding where to evaluate the objective using a Bayesian interpretation of these surrogates.

We first introduce the typical form that Bayesian optimization algorithms take in Section 2. This form involves two primary components: a method for statistical inference, typically Gaussian process (GP) regression; and an acquisition function for deciding where to sample, which is often expected improvement. We describe these two components in detail in Sections 3 and 4.1. We then describe three alternate acquisition functions: knowledge-gradient (Section 4.2), entropy search, and predictive entropy search (Section 4.3). These alternate acquisition functions are particularly useful in problems falling outside the strict set of assumptions above, which we call "exotic" Bayesian optimization problems and we discuss in Section 5. These exotic Bayesian optimization problems include those with parallel evaluations, constraints, multi-fidelity evaluations, multiple information sources, random environmental conditions, multi-task objectives, and derivative observations. We then discuss Bayesian optimization and Gaussian process regression software in Section 6 and conclude with a discussion of future research directions in Section 7.

Other tutorials and surveys on Bayesian optimization include Shahriari et al. (2016); Brochu et al. (2009); Sasena (2002); Frazier and Wang (2016). This tutorial differs from these others in its coverage of non-standard or "exotic" Bayesian optimization problems. It also differs in its substantial emphasis on acquisition functions, with less emphasis on GP regression. Finally, it includes what we believe is a novel analysis of expected improvement for noisy measurements, and argues that the acquisition function previously proposed by Scott et al. (2011) is the most natural way to apply the expected improvement acquisition function when measurements are noisy.

2 Overview of BayesOpt

BayesOpt consists of two main components: a Bayesian statistical model for modeling the objective function, and an acquisition function for deciding where to sample next. After evaluating the objective according to an initial space-filling experimental design, often consisting of points chosen uniformly at

random, they are used iteratively to allocate the remainder of a budget of N function evaluations, as shown in Algorithm 1.

Algorithm 1 Basic pseudo-code for Bayesian optimization

Place a Gaussian process prior on f
Observe f at n_0 points according to an initial space-filling experimental design. Set $n = n_0$.
while $n \leq N$ **do**
  Update the posterior probability distribution on f using all available data
  Let x_n be a maximizer of the acquisition function over x, where the acquisition function is computed using the current posterior distribution.
  Observe $y_n = f(x_n)$.
  Increment n
end while
Return a solution: either the point evaluated with the largest $f(x)$, or the point with the largest posterior mean.

The statistical model, which is invariably a Gaussian process, provides a Bayesian posterior probability distribution that describes potential values for $f(x)$ at a candidate point x. Each time we observe f at a new point, this posterior distribution is updated. We discuss Bayesian statistical modeling using Gaussian processes in detail in Section 3. The acquisition function measures the value that would be generated by evaluation of the objective function at a new point x, based on the current posterior distribution over f. We discuss expected improvement, the most commonly used acquisition function, in Section 4.1, and then discuss other acquisition functions in Section 4.2 and 4.3.

One iteration of BayesOpt from Algorithm 1 using GP regression and expected improvement is illustrated in Figure 1. The top panel shows noise-free observations of the objective function with blue circles at three points. It also shows the output of GP regression. We will see below in Section 3 that GP regression produces a posterior probability distribution on each $f(x)$ that is normally distributed with mean $\mu_n(x)$ and variance $\sigma_n^2(x)$. This is pictured in the figure with $\mu_n(x)$ as the solid red line, and a 95% Bayesian credible interval for $f(x)$, $\mu_n(x) \pm 1.96 \times \sigma_n(x)$, as dashed red lines. The mean can be interpreted as a point estimate of $f(x)$. The credible interval acts like a confidence interval in frequentist statistics, and contains $f(x)$ with probability 95% according to the posterior distribution. The mean interpolates the previously evaluated points. The credible interval has 0 width at these points, and grows wider as we move away from them.

The bottom panel shows the expected improvement acquisition function that corresponds to this posterior. Observe that it takes value 0 at points that have previously been evaluated. This is reasonable when evaluations of the objective are noise-free because evaluating these points provides no useful information toward solving (1). Also observe that it tends to be larger for points with larger credible intervals, because observing a point where we are more uncertain about the objective tends to be more useful in finding good approximate global optima. Also observe it tends to be larger for points with larger posterior means, because such points tend to be near good approximate global optima.

We now discuss the components of BayesOpt in detail, first discussing GP regression in Section 3, then discuss acquisition functions in Section 4, starting with expected improvement in Section 4.1. We then discuss more sophisticated acquisition functions (knowledge gradient, entropy search, and predictive entropy search) in Sections 4.2 and 4.3. Finally, we discuss extensions of the basic problem described in Section 1 in Section 5, discussing problems with measurement noise, parallel function evaluations, constraints, multi-fidelity observations, and others.

3 Gaussian Process (GP) Regression

GP regression is a Bayesian statistical approach for modeling functions. We offer a brief introduction here. A more complete treatment may be found in Rasmussen and Williams (2006).

We first describe GP regression, focusing on f's values at a finite collection of points $x_1, \ldots, x_k \in \mathbb{R}^d$. It is convenient to collect the function's values at these points together into a vector $[f(x_1), \ldots, f(x_k)]$. Whenever we have a quantity that is unknown in Bayesian statistics, like this vector, we suppose that it was drawn at random by nature from some prior probability distribution. GP regression takes this prior distribution to be multivariate normal, with a particular mean vector and covariance matrix.

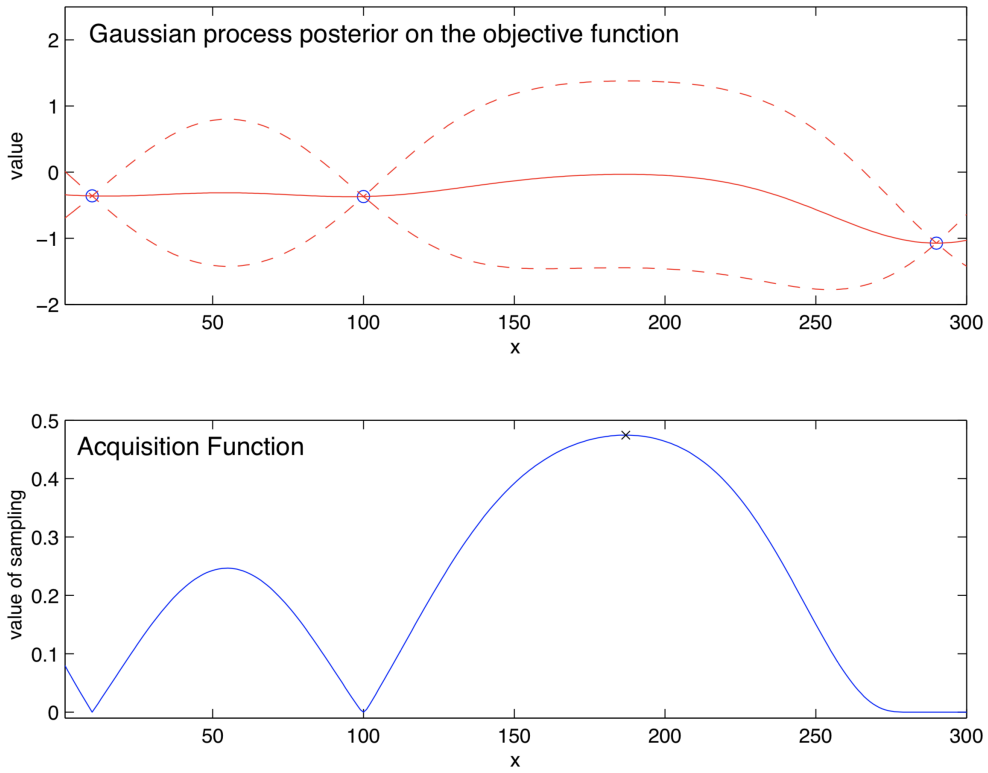

Figure 1: Illustration of BayesOpt, maximizing an objective function f with a 1-dimensional continuous input. The top panel shows: noise-free observations of the objective function f at 3 points, in blue; an estimate of $f(x)$ (solid red line); and Bayesian credible intervals (similar to confidence intervals) for $f(x)$ (dashed red line). These estimates and credible intervals are obtained using GP regression. The bottom panel shows the acquisition function. Bayesian optimization chooses to sample next at the point that maximizes the acquisition function, indicated here with an "x."

We construct the mean vector by evaluating a *mean function* μ_0 at each x_i. We construct the covariance matrix by evaluating a *covariance function* or *kernel* Σ_0 at each pair of points x_i, x_j. The kernel is chosen so that points x_i, x_j that are closer in the input space have a large positive correlation, encoding the belief that they should have more similar function values than points that are far apart. The kernel should also have the property that the resulting covariance matrix is positive semi-definite, regardless of the collection of points chosen. Example mean functions and kernels are discussed below in Section 3.1.

The resulting prior distribution on $[f(x_1), \ldots, f(x_k)]$ is,

$$f(x_{1:k}) \sim \text{Normal}\left(\mu_0(x_{1:k}), \Sigma_0(x_{1:k}, x_{1:k})\right), \tag{2}$$

where we use compact notation for functions applied to collections of input points: $x_{1:k}$ indicates the sequence $x_1, \ldots, x_k$, $f(x_{1:k}) = [f(x_1), \ldots, f(x_k)]$, $\mu_0(x_{1:k}) = [\mu_0(x_1), \ldots, \mu_0(x_k)]$, and $\Sigma_0(x_{1:k}, x_{1:k}) = [\Sigma_0(x_1, x_1), \ldots, \Sigma_0(x_1, x_k); \ldots; \Sigma_0(x_k, x_1), \ldots, \Sigma_0(x_k, x_k)]$.

Suppose we observe $f(x_{1:n})$ without noise for some n and we wish to infer the value of $f(x)$ at some new point x. To do so, we let $k = n + 1$ and $x_k = x$, so that the prior over $[f(x_{1:n}), f(x)]$ is given by (2). We may then compute the conditional distribution of $f(x)$ given these observations using Bayes' rule (see details in Chapter 2.1 of Rasmussen and Williams (2006)),

$$\begin{aligned} f(x)|f(x_{1:n}) &\sim \text{Normal}(\mu_n(x), \sigma_n^2(x)) \\ \mu_n(x) &= \Sigma_0(x, x_{1:n})\Sigma_0(x_{1:n}, x_{1:n})^{-1}\left(f(x_{1:n}) - \mu_0(x_{1:n})\right) + \mu_0(x) \\ \sigma_n^2(x) &= \Sigma_0(x, x) - \Sigma_0(x, x_{1:n})\Sigma_0(x_{1:n}, x_{1:n})^{-1}\Sigma_0(x_{1:n}, x). \end{aligned} \tag{3}$$

This conditional distribution is called the *posterior probability distribution* in the nomenclature of

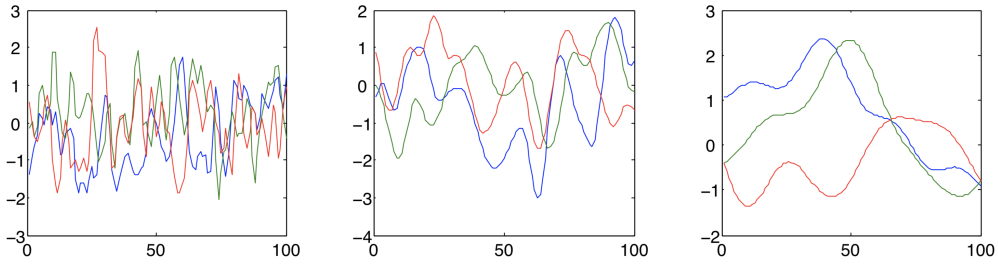

Figure 2: Random functions f drawn from a Gaussian process prior with a power exponential kernel. Each plot corresponds to a different value for the parameter α_1, with α_1 decreasing from left to right. Varying this parameter creates different beliefs about how quickly $f(x)$ changes with x.

Bayesian statistics. The posterior mean $\mu_n(x)$ is a weighted average between the prior $\mu_0(x)$ and an estimate based on the data $f(x_{1:n})$, with a weight that depends on the kernel. The posterior variance $\sigma_n^2(x)$ is equal to the prior covariance $\Sigma_0(x, x)$ less a term that corresponds to the variance removed by observing $f(x_{1:n})$.

Rather than computing posterior means and variances directly using (3) and matrix inversion, it is typically faster and more numerically stable to use a Cholesky decomposition and then solve a linear system of equations. This more sophisticated technique is discussed as Algorithm 2.1 in Section 2.2 of Rasmussen and Williams (2006). Additionally, to improve the numerical stability of this approach or direct computation using (3), it is often useful to add a small positive number like 10^{-6} to each element of the diagonal of $\Sigma_0(x_{1:n}, x_{1:n})$, especially when $x_{1:n}$ contains two or more points that are close together. This prevents eigenvalues of $\Sigma_0(x_{1:n}, x_{1:n})$ from being too close to 0, and only changes the predictions that would be made by an infinite-precision computation by a small amount.

Although we have modeled f at only a finite number of points, the same approach can be used when modeling f over a continuous domain A. Formally a *Gaussian process* with mean function μ_0 and kernel Σ_0 is a probability distribution over the function f with the property that, for any given collection of points $x_{1:k}$, the marginal probability distribution on $f(x_{1:k})$ is given by (2). Moreover, the arguments that justified (3) still hold when our prior probability distribution on f is a Gaussian process.

In addition to calculating the conditional distribution of $f(x)$ given $f(x_{1:n})$, it is also possible to calculate the conditional distribution of f at more than one unevaluated point. The resulting distribution is multivariate normal, with a mean vector and covariance kernel that depend on the location of the unevaluated points, the locations of the measured points $x_{1:n}$, and their measured values $f(x_{1:n})$. The functions that give entries in this mean vector and covariance matrix have the form required for a mean function and kernel described above, and the conditional distribution of f given $f(x_{1:n})$ is a Gaussian process with this mean function and covariance kernel.

3.1 Choosing a Mean Function and Kernel

We now discuss the choice of kernel. Kernels typically have the property that points closer in the input space are more strongly correlated, i.e., that if $||x - x'|| < ||x - x''||$ for some norm $||\cdot||$, then $\Sigma_0(x, x') > \Sigma_0(x, x'')$. Additionally, kernels are required to be positive semi-definite functions. Here we describe two example kernels and how they are used.

One commonly used and simple kernel is the *power exponential* or *Gaussian* kernel,

$$\Sigma_0(x, x') = \alpha_0 \exp\left(-||x - x'||^2\right),$$

where $||x - x'||^2 = \sum_{i=1}^{d} \alpha_i (x_i - x_i')^2$, and $\alpha_{0:d}$ are parameters of the kernel. Figure 2 shows random functions with a 1-dimensional input drawn from a Gaussian process prior with a power exponential kernel with different values of α_1. Varying this parameter creates different beliefs about how quickly $f(x)$ changes with x.

Another commonly used kernel is the *Màtern* kernel,

$$\Sigma_0(x, x') = \alpha_0 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}||x - x'||\right)^{\nu} K_\nu(\sqrt{2\nu}||x - x'||)$$

where K_ν is the modified Bessel function, and we have a parameter ν in addition to the parameters $\alpha_{0:d}$. We discuss choosing these parameters below in Section 3.2.

Perhaps the most common choice for the mean function is a constant value, $\mu_0(x) = \mu$. When f is believed to have a trend or some application-specific parametric structure, we may also take the mean function to be

$$\mu_0(x) = \mu + \sum_{i=1}^{p} \beta_i \Psi_i(x), \tag{4}$$

where each Ψ_i is a parametric function, and often a low-order polynomial in x.

3.2 Choosing Hyperparameters

The mean function and kernel contain parameters. We typically call these parameters of the prior *hyperparameters*. We indicate them via a vector η. For example, if we use a Màtern kernel and a constant mean function, $\eta = (\alpha_{0:d}, \nu, \mu)$.

To choose the hyperparameters, three approaches are typically considered. The first is to find the *maximum likelihood estimate* (MLE). In this approach, when given observations $f(x_{1:n})$, we calculate the likelihood of these observations under the prior, $P(f(x_{1:n})|\eta)$, where we modify our notation to indicate its dependence on η. This likelihood is a multivariate normal density. Then, in maximum likelihood estimation, we set η to the value that maximizes this likelihood,

$$\hat{\eta} = \operatorname*{argmax}_{\eta} P(f(x_{1:n})|\eta)$$

The second approach amends this first approach by imagining that the hyperparameters η were themselves chosen from a prior, $P(\eta)$. We then estimate η by the *maximum a posteriori* (MAP) estimate (Gelman et al., 2014), which is the value of η that maximizes the posterior,

$$\hat{\eta} = \operatorname*{argmax}_{\eta} P(\eta|f(x_{1:n})) = \operatorname*{argmax}_{\eta} P(f(x_{1:n})|\eta)P(\eta)$$

In moving from the first expression to the second we have used Bayes' rule and then dropped a normalization constant $\int P(f(x_{1:n})|\eta')P(\eta')\,d\eta'$ that does not depend on the quantity η being optimized.

The MLE is a special case of the MAP if we take the prior on the hyperparameters $P(\eta)$ to be the (possibly degenerate) probability distribution that has constant density over the domain of η. The MAP is useful if the MLE sometimes estimates unreasonable hyperparameter values, for example, corresponding to functions that vary too quickly or too slowly (see Figure 2). By choosing a prior that puts more weight on hyperparameter values that are reasonable for a particular problem, MAP estimates can better correspond to the application. Common choices for the prior include the uniform distribution (for preventing estimates from falling outside of some pre-specified range), the normal distribution (for suggesting that the estimates fall near some nominal value without setting a hard cutoff), and the log-normal and truncated normal distributions (for providing a similar suggestion for positive parameters).

The third approach is called the *fully Bayesian* approach. In this approach, we wish to compute the posterior distribution on $f(x)$ marginalizing over all possible values of the hyperparameters,

$$P(f(x) = y|f(x_{1:n})) = \int P(f(x) = y|f(x_{1:n}), \eta)P(\eta|f(x_{1:n}))\,d\eta \tag{5}$$

This integral is typically intractable, but we can approximate it through sampling:

$$P(f(x) = y|f(x_{1:n})) \approx \frac{1}{J}\sum_{j=1}^{J} P(f(x) = y|f(x_{1:n}), \eta = \hat{\eta}_j) \tag{6}$$

where $(\hat{\eta}_j : j = 1, \ldots, J)$ are sampled from $P(\eta|f(x_{1:n}))$ via an MCMC method, e.g., slice sampling (Neal, 2003). MAP estimation can be seen as an approximation to fully Bayesian inference: if we approximate the posterior $P(\eta|f(x_{1:n}))$ by a point mass at the η that maximizes the posterior density, then inference with the MAP recovers (5).

4 Acquisition Functions

Having surveyed Gaussian processes, we return to Algorithm 1 and discuss the acquisition function used in that loop. We focus on the setting described in Section 1 with noise-free evaluations, which we call the "standard" problem, and then discuss noisy evaluations, parallel evaluations, derivative observations, and other "exotic" extensions in Section 5.

The most commonly used acquisition function is expected improvement, and we discuss it first, in Section 4.1. Expected improvement performs well and is easy to use. We then discuss the knowledge gradient (Section 4.2), entropy search and predictive entropy search (Section 4.3) acquisition functions. These alternate acquisition functions are most useful in exotic problems where an assumption made by expected improvement, that the primary benefit of sampling occurs through an improvement *at the point sampled*, is no longer true.

4.1 Expected Improvement

The expected improvement acquisition function is derived by a thought experiment. Suppose we are using Algorithm 1 to solve (1), in which x_n indicates the point sampled at iteration n and y_n indicates the observed value. Assume that we may only return a solution that we have evaluated as our final solution to (1). Also suppose for the moment that we have no evaluations left to make, and must return a solution based on those we have already performed. Since we observe f without noise, the optimal choice is the previously evaluated point with the largest observed value. Let $f_n^* = \max_{m \le n} f(x_m)$ be the value of this point, where n is the number of times we have evaluated f thus far.

Now suppose in fact we have one additional evaluation to perform, and we can perform it anywhere. If we evaluate at x, we will observe $f(x)$. After this new evaluation, the value of the best point we have observed will either be $f(x)$ (if $f(x) \ge f_n^*$) or f_n^* (if $f(x) \le f_n^*$). The improvement in the value of the best observed point is then $f(x) - f_n^*$ if this quantity is positive, and 0 otherwise. We can write this improvement more compactly as $[f(x) - f_n^*]^+$, where $a^+ = \max(a, 0)$ indicates the positive part.

While we would like to choose x so that this improvement is large, $f(x)$ is unknown until after the evaluation. What we can do, however, is to take the expected value of this improvement and choose x to maximize it. We define the *expected improvement* as,

$$\mathrm{EI}_n(x) := E_n\left[[f(x) - f_n^*]^+\right] \tag{7}$$

Here, $E_n[\cdot] = E[\cdot|x_{1:n}, y_{1:n}]$ indicates the expectation taken under the posterior distribution given evaluations of f at $x_1, \ldots x_n$. This posterior distribution is given by (3): $f(x)$ given $x_{1:n}, y_{1:n}$ is normally distributed with mean $\mu_n(x)$ and variance $\sigma_n^2(x)$.

The expected improvement can be evaluated in closed form using integration by parts, as described in Jones et al. (1998) or Clark (1961). The resulting expression is

$$\mathrm{EI}_n(x) = [\Delta_n(x)]^+ + \sigma_n(x)\varphi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right) - |\Delta_n(x)|\Phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right), \tag{8}$$

where $\Delta_n(x) := \mu_n(x) - f_n^*$ is the expected difference in quality between the proposed point x and the previous best.

The expected improvement algorithm then evaluates at the point with the largest expected improvement,

$$x_{n+1} = \operatorname{argmax} \mathrm{EI}_n(x), \tag{9}$$

breaking ties arbitrarily. This algorithm was first proposed by Močkus (Močkus, 1975) but was popularized by Jones et al. (1998). The latter article also used the name "Efficient Global Optimization" or EGO.

Implementations use a variety of approaches for solving (9). Unlike the objective f in our original optimization problem (1), $\mathrm{EI}_n(x)$ is inexpensive to evaluate and allows easy evaluation of first- and second-order derivatives. Implementations of the expected improvement algorithm can then use a continuous first- or second-order optimization method to solve (9). For example, one technique that has worked well for the author is to calculate first derivatives and use the quasi-Newton method L-BFGS-B (Liu and Nocedal, 1989).

Figure 3 shows the contours of $\mathrm{EI}_n(x)$ in terms of $\Delta_n(x)$ and the posterior standard deviation $\sigma_n(x)$. $\mathrm{EI}_n(x)$ is increasing in both $\Delta_n(x)$ and $\sigma_n(x)$. Curves of $\Delta_n(x)$ versus $\sigma_n(x)$ with equal EI show how EI balances between evaluating at points with high expected quality (high $\Delta_n(x)$)) versus high uncertainty (high $\sigma_n(x)$). In the context of optimization, evaluating at points with high expected quality relative

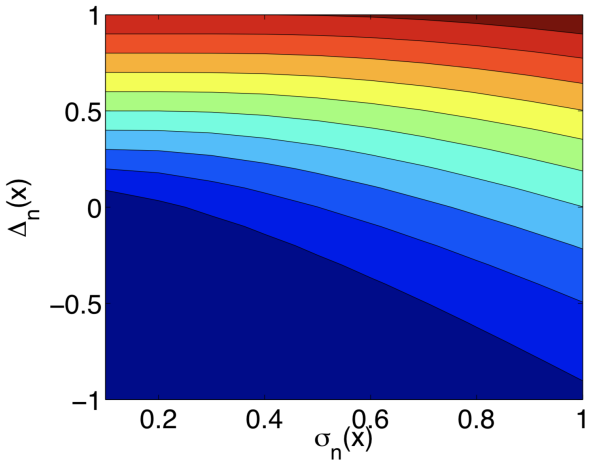

Figure 3: Contour plot of EI(x), the expected improvement (8), in terms of $\Delta_n(x)$ (the expected difference in quality between the proposed point and the best previously evaluated point) and the posterior standard deviation $\sigma_n(x)$. Blue indicates smaller values and red higher ones. The expected improvement is increasing in both quantities, and curves of $\Delta_n(x)$ versus $\sigma_n(x)$ with equal EI define an implicit tradeoff between evaluating at points with high expected quality (high $\Delta_n(x)$ versus high uncertainty (high $\sigma_n(x)$).

to the previous best point is valuable because good approximate global optima are likely to reside at such points. On the other hand, evaluating at points with high uncertainty is valuable because it teaches about the objective in locations where we have little knowledge and which tend to be far away from where we have previously measured. A point that is substantially better than one we have seen previously may very well reside there.

Figure 1 shows EI(x) in the bottom panel. We see this tradeoff, with the largest expected improvement occurring where the posterior standard deviation is high (far away from previously evaluated points), and where the posterior mean is also high. The smallest expected improvement is 0, at points where we have previously evaluated. The posterior standard deviation is 0 at this point, and the posterior mean is necessarily no larger than the best previously evaluated point. The expected improvement algorithm would evaluate next at the point indicated with an x, where EI is maximized.

Choosing where to evaluate based on a tradeoff between high expected performance and high uncertainty appears in other domains, including multi-armed bandits Mahajan and Teneketzis (2008) and reinforcement learning Sutton and Barto (1998), and is often called the "exploration vs. exploitation tradeoff" (Kaelbling et al., 1996).

4.2 Knowledge Gradient

The knowledge-gradient acquisition function is derived by revisiting the assumption made in EI's derivation that we are only willing to return a previously evaluated point as our final solution. That assumption is reasonable when evaluations are noise-free and we are highly risk averse, but if the decision-maker is willing to tolerate some risk then she might be willing to report a final solution that has some uncertainty attached to it. Moreover, if evaluations have noise (discussed below in Section 5) then the final solution reported will necessarily have uncertain value because we can hardly evaluate it an infinite number of times.

We replace this assumption by allowing the decision-maker to return any solution she likes, even if it has not been previously evaluated. We also assume risk-neutrality (Berger, 2013), i.e., we value a random outcome X according to its expected value. The solution that we would choose if we were to stop sampling after n samples would be the one with the largest $\mu_n(x)$ value. This solution (call it $\widehat{x^*}$, since it approximates the global optimum x^*) would have value $f(\widehat{x^*})$. $f(\widehat{x^*})$ is random under the posterior, and has conditional expected value $\mu_n(\widehat{x^*}) = \max_{x'} \mu_n(x') =: \mu_n^*$.

On the other hand, if we were to take one more sample at x, we would obtain a new posterior distribution with posterior mean $\mu_{n+1}(\cdot)$. This posterior mean would be computed via (3), but including the additional observation x_{n+1}, y_{n+1}. If we were to report a final solution after this sample, its expected

value under the new posterior distribution would be $\mu^*_{n+1} := \max_{x'} \mu_{n+1}(x')$. Thus, the increase in conditional expected solution value due to sampling is $\mu^*_{n+1} - \mu^*_n$.

While this quantity is unknown before we sample at x_{n+1}, we can compute its expected value given the observations at $x_1, \ldots, x_n$ that we have. We call this quantity the *knowledge gradient (KG)* for measuring at x,

$$\mathrm{KG}_n(x) := E_n \left[\mu^*_{n+1} - \mu^*_n | x_{n+1} = x \right]. \tag{10}$$

Using the knowledge gradient as our acquisition function then leads us to sample at the point with largest $\mathrm{KG}_n(x)$, $\mathrm{argmax}_x \mathrm{KG}_n(x)$.

This algorithm was first proposed in Frazier et al. (2009) for GP regression over discrete A, building on earlier work (Frazier et al., 2008) that proposed the same algorithm for Bayesian ranking and selection (Chick and Inoue, 2001) with an independent prior. (Bayesian ranking and selection is similar to Bayesian optimization, except that A is discrete and finite, observations are necessarily noisy, and the prior is typically independent across x.)

The conceptually simplest way to compute the KG acquisition function is via simulation, as shown in Algorithm 2. Within a loop, this algorithm simulates one possible value for the observation y_{n+1} that may result from taking evaluation $n+1$ at a designated x. Then it computes what the maximum of the new posterior mean μ^*_{n+1} would be if that value for y_{n+1} were the one that actually resulted from the measurement. It then subtracts μ^*_n to obtain the corresponding increase in solution quality. This comprises one loop of the algorithm. It iterates this loop many (J) times and averages the differences $\mu^*_{n+1} - \mu^*_n$ obtained from different simulated values for y_{n+1} to estimate the KG acquisition function $\mathrm{KG}_n(x)$. As J grows large, this estimate converges to $\mathrm{KG}_n(x)$.

In principle, this algorithm can be used to evaluate $\mathrm{KG}_n(x)$ within a derivative-free simulation-based optimization method to optimize the KG acquisition function. However, optimizing noisy simulation-based functions without access to derivatives is challenging. Frazier et al. (2009) proposed discretizing A and calculating (10) exactly using properties of the normal distribution. This works well for low-dimensional problems but becomes computationally burdensome in higher dimensions.

Algorithm 2 Simulation-based computation of the knowledge-gradient factor $\mathrm{KG}_n(x)$.

Let $\mu^*_n = \max_{x'} \mu_n(x')$.
(When calculating μ^*_n and μ^*_{n+1} below, use a nonlinear optimization method like L-BFGS.)
for $j = 1$ to J: **do**
  Generate $y_{n+1} \sim \mathrm{Normal}(\mu_n(x), \sigma^2_n(x))$. (Equivalently, $Z \sim \mathrm{Normal}(0,1)$ and $y_{n+1} = \mu_n(x) + \sigma_n(x)Z$.)
  Set $\mu_{n+1}(x'; x, y_{n+1})$ to the posterior mean at x' via (3) with (x, y_{n+1}) as the last observation.
  $\mu^*_{n+1} = \max_{x'} \mu_{n+1}(x'; x, y_{n+1})$.
  $\Delta^{(j)} = \mu^*_{n+1} - \mu^*_n$.
end for
Estimate $\mathrm{KG}_n(x)$ by $\frac{1}{J} \sum_{j=1}^{J} \Delta^{(j)}$.

Overcoming this challenge with dimensionality, Wu and Frazier (2016) proposed[1] a substantially more efficient and scalable approach, based on multi-start stochastic gradient ascent. Stochastic gradient ascent (Robbins and Monro, 1951; Blum, 1954) is an algorithm for finding local optima of functions used such unbiased gradient estimates widely used in machine learning (Bottou, 2012). Multistart stochastic gradient ascent (Martí et al., 2016) runs multiple instances of stochastic gradient ascent from different starting points and selects the best local optimum found as an approximate global optimum.

We summarize this approach for maximizing the KG acquisition function in Algorithm 3. The algorithm iterates over starting points, indexed by r, and for each maintains a sequence of iterates $x^{(r)}_t$, indexed by t, that converges to a local optimum of the KG acquisition function. The inner loop over t relies on a *stochastic gradient* G, which is a random variable whose expected value is equal to the gradient of the KG acquisition function with respect to where we sample, evaluated at the current iterate $x^{(r)}_{t-1}$. We obtain the next iterate by taking a step in the direction of the stochastic gradient G. The size of this step is determined by the magnitude of G and a decreasing step size α_t. Once stochastic gradient ascent has run for T iterations for each start, Algorithm 3 uses simulation (Algorithm 2) to evaluate the KG acquisition function for the final point obtained from each starting point, and selects the best one.

[1] This approach was proposed in the context of BayesOpt with parallel function evaluations, but can also be used in the setting considered here in which we perform one evaluation at a time.

Algorithm 3 Efficient method for finding x with the largest $\mathrm{KG}_n(x)$, based on multistart stochastic gradient ascent. Takes as input a number of starts R, a number of iterations T for each pass of stochastic gradient ascent, a parameter a used to define a stepsize sequence, and a number of replications J. Suggested input parameters: $R = 10$, $T = 10^2$, $a = 4$, $J = 10^3$.

for $r = 1$ to R **do**
  Choose $x_0^{(r)}$ uniformly at random from A.
  for $t = 1$ to T **do**
    Let G be the stochastic gradient estimate of $\nabla \mathrm{KG}_n(x_{t-1}^{(r)})$ from Algorithm 4.
    Let $\alpha_t = a/(a+t)$.
    $x_t^{(r)} = x_{t-1}^{(r)} + \alpha_t G$.
  end for
  Estimate $\mathrm{KG}_n(x_T^{(r)})$ using Algorithm 2 and J replications.
end for
Return the $x_T^{(r)}$ with the largest estimated value of $\mathrm{KG}_n(x_T^{(r)})$.

This stochastic gradient G used by the inner loop of Algorithm 3 is calculated via Algorithm 4. This algorithm is based on the idea that we can exchange gradient and expectation (under sufficient regularity conditions) to write,

$$\nabla \mathrm{KG}_n(x) = \nabla E_n\left[\mu_{n+1}^* - \mu_n^* | x_{n+1} = x\right]. = E_n\left[\nabla \mu_{n+1}^* | x_{n+1} = x\right],$$

where we have noted that μ_n^* does not depend on x. This approach is called *infinitesimal perturbation analysis* (Ho et al., 1983). Thus, to construct a stochastic gradient it is sufficient to sample $\nabla \mu_{n+1}^*$. In other words, imagine first sampling Z in the inner loop in Algorithm 2, and then holding Z fixed while calculating the gradient of μ_{n+1}^* with respect to x. To calculate this gradient, see that μ_{n+1}^* is a maximum over x' of $\mu_{n+1}(x'; x, y_{n+1}) = \mu_{n+1}(x'; x, \mu_n(x) + \sigma_n(x)Z)$. This is a maximum over collection of functions of x. The envelope theorem (Milgrom and Segal, 2002) tells us (under sufficient regularity conditions) that the gradient with respect to x of a maximum of a collection of functions of x is given simply by first finding the maximum in this collection, and then differentiating this single function with respect to x. In our setting, we apply this by letting $\widehat{x^*}$ be the x' maximizing $\mu_{n+1}(x'; x, \mu_n(x) + \sigma_n(x)Z)$, and then calculating the gradient of $\mu_{n+1}(\widehat{x^*}; x, \mu_n(x) + \sigma_n(x)Z)$ with respect to x while holding $\widehat{x^*}$ fixed. In other words,

$$\nabla \max_{x'} \mu_{n+1}(x'; x, \mu_n(x) + \sigma_n(x)Z) = \nabla \mu_{n+1}(\widehat{x^*}; x, \mu_n(x) + \sigma_n(x)Z),$$

where we remind the reader that ∇ refers to taking the gradient with respect to x, here and throughout. This is summarized in Algorithm 4.

Algorithm 4 Simulation of unbiased stochastic gradients G with $E[G] = \nabla \mathrm{KG}_n(x)$. This stochastic gradient can then be used within stochastic gradient ascent to optimize the KG acquisition function.

for $j = 1$ to J **do**
  Generate $Z \sim \mathrm{Normal}(0, 1)$
  $y_{n+1} = \mu_n(x) + \sigma_n(x)Z$.
  Let $\mu_{n+1}(x'; x, y_{n+1}) = \mu_{n+1}(x'; x, \mu_n(x) + \sigma_n(x)Z)$ be the posterior mean at x' computed via (3) with (x, y_{n+1}) as the last observation.
  Solve $\max_{x'} \mu_{n+1}(x'; x, y_{n+1})$, e.g., using L-BFGS. Let $\widehat{x^*}$ be the maximizing x'.
  Let $G^{(j)}$ be the gradient of $\mu_{n+1}(\widehat{x^*}; x, \mu_n(x) + \sigma_n(x)Z)$ with respect to x, holding $\widehat{x^*}$ fixed.
end for
Estimate $\nabla \mathrm{KG}_n(x)$ by $G = \frac{1}{J}\sum_{j=1}^J G^{(j)}$.

Unlike expected improvement, which only considers the posterior at the point sampled, the KG acquisition considers the posterior over f's full domain, and how the sample will change that posterior. KG places a positive value on measurements that cause the maximum of the posterior mean to improve, even if the value at the sampled point is not better than the previous best point. This provides a small performance benefit in the standard BayesOpt problem with noise-free evaluations (Frazier et al., 2009),

and provides substantial performance improvements in problems with noise, multi-fidelity observations, derivative observations, the need to integrate over environmental conditions, and other more exotic problem features (Section 5). In these alternate problems, the value of sampling comes not through an improvement in the best solution *at the sampled point*, but through an improvement in the maximum of the posterior mean across feasible solutions. For example, a derivative observation may show that the function is increasing in a particular direction in the vicinity of the sampled point. This may cause the maximum of the posterior mean to be substantially larger than the previous maximum, even if the function value at the sampled point is worse than the best previously sampled point. When such phenomenon are first-order, KG tends to significantly outperform EI (Wu et al., 2017; Poloczek et al., 2017; Wu and Frazier, 2016; Toscano-Palmerin and Frazier, 2018).

4.3 Entropy Search and Predictive Entropy Search

The entropy search (ES) (Hennig and Schuler, 2012) acquisition function values the information we have about the location of the global maximum according to its differential entropy. ES seeks the point to evaluate that causes the largest decrease in differential entropy. (Recall from, e.g., Cover and Thomas (2012), that the differential entropy of a continuous probability distribution $p(x)$ is $\int p(x)\log(p(x))\,dx$, and that smaller differential entropy indicates less uncertainty.) Predictive entropy search (PES) (Hernández-Lobato et al., 2014) seeks the same point, but uses a reformulation of the entropy reduction objective based on mutual information. Exact calculations of PES and ES would give equivalent acquisition functions, but exact calculation is not typically possible, and so the difference in computational techniques used to approximate the PES and ES acquisition functions creates practical differences in the sampling decisions that result from the two approaches. We first discuss ES and then PES.

Let x^* be the global optimum of f. The posterior distribution on f at time n induces a probability distribution for x^*. Indeed, if the domain A were finite, then we could represent f over its domain by a vector $(f(x) : x \in A)$, and x^* would correspond to the largest element in this vector. The distribution of this vector under the time-n posterior distribution would be multivariate normal, and this multivariate normal distribution would imply the distribution of x^*. When A is continuous, the same ideas apply, where x^* is a random variable whose distribution is implied by the Gaussian process posterior on f.

With this understanding, we represent the entropy of the time-n posterior distribution on x^* with the notation $H(P_n(x^*))$. Similarly, $H(P_n(x^*|x, f(x)))$ represents the entropy of what the time-$n+1$ posterior distribution on x^* will be if we observe at x and see $f(x)$. This quantity depends on the value of $f(x)$ observed. Then, the entropy reduction due to sampling x can be written,

$$\mathrm{ES}_n(x) = H(P_n(x^*)) - E_{f(x)}\left[H(P_n(x^*|f(x)))\right]. \tag{11}$$

In the second term, the subscript in the outer expectation indicates that we take the expectation over $f(x)$. Equivalently, this can be written $\int \varphi(y;\mu_n(x),\sigma_n^2(x))H(P_n(x^*|f(x)=y))\,dy$ where $\varphi(y;\mu_n(x),\sigma_n^2(x))$ is the normal density with mean $\mu_n(x)$ and variance $\sigma_n^2(x)$.

Like KG, ES and PES below are influenced by how the measurement changes the posterior over the whole domain, and not just on whether it improves over an incumbent solution at the point sampled. This is useful when deciding where to sample in exotic problems, and it is here that ES and PES can provide substantial value relative to EI.

While ES can be computed and optimized approximately (Hennig and Schuler, 2012), doing so is challenging because (a) the entropy of the maximizer of a Gaussian process is not available in closed form; (b) we must calculate this entropy for a large number of y to approximate the expectation in (11); and (c) we must then optimize this hard-to-evaluate function. Unlike KG, there is no known method for computing stochastic gradients that would simplify this optimization.

PES offers an alternate approach for computing (11). This approach notes that the reduction in the entropy of x^* due to measuring $f(x)$ is equal to the mutual information between $f(x)$ and x^*, which is in turn equal to the reduction in the entropy of $f(x)$ due to measuring x^*. This equivalence gives the expression

$$\mathrm{PES}_n(x) = \mathrm{ES}_n(x) = H(P_n(f(x))) - E_{x^*}\left[H(P_n(f(x)|x^*))\right] \tag{12}$$

Here, the subscript in the expectation in the second term indicates that the expectation is taken over x^*.

Unlike ES, the first term in the PES acquisition function, $H(P_n(f(x)))$, can be computed in closed form. The second term must still be approximated: Hernández-Lobato et al. (2014) provides a method for sampling x^* from the posterior distribution, and a method for approximating $H(P_n(f(x)|x^*))$ using expectation propagation (Minka, 2001). This evaluation method may then be optimized by a method for derivative-free optimization via simulation.

4.4 Multi-Step Optimal Acquisition Functions

We can view the act of solving problem (1) as a sequential decision-making problem (Ginsbourger and Riche, 2010; Frazier, 2012), in which we sequentially choose x_n, and observe $y_n = f(x_n)$, with the choice of x_n depending on all past observations. At the end of these observations, we then receive a reward that might be equal to the value of the best point observed, $\max_{m \leq N} f(x_m)$, as it was in the analysis of EI, or could be equal to the value $f(\widehat{x^*})$ of the objective at some new point $\widehat{x^*}$ chosen based on these observations as in the analysis of KG, or it could be the entropy of the posterior distribution on x^* as in ES or PES.

By construction, the EI, KG, ES, and PES acquisition functions are optimal when $N = n + 1$, in the sense of maximizing the expected reward under the posterior. However, they are no longer obviously optimal when $N > n + 1$. In principle, it is possible to compute a *multi-step optimal* acquisition function that would maximize expected reward for general N via stochastic dynamic programming (Dynkin and Yushkevich, 1979), but the so-called curse of dimensionality (Powell, 2007) makes it extremely challenging to compute this multi-step optimal acquisition function in practice.

Nevertheless, the literature has recently begun to deploy approximate methods for computing this solution, with attempts including Lam et al. (2016); Ginsbourger and Riche (2010); González et al. (2016). These methods do not yet seem to be in a state where they can be deployed broadly for practical problems, because the error and extra cost introduced in solving stochastic dynamic programming problems approximately often overwhelms the benefit that considering multiple steps provides. Nevertheless, given concurrent advances in reinforcement learning and approximate dynamic programming, this represents a promising and exciting direction for Bayesian optimization.

In addition, there are other problem settings closely related to the one most commonly considered by Bayesian optimization where it is possible to compute multi-step optimal algorithms. For example, Cashore et al. (2016) and Xie and Frazier (2013) use problem structure to efficiently compute multi-step optimal algorithms for certain classes of Bayesian feasibility determination problems, where we wish to sample efficiently to determine whether $f(x)$ is above or below a threshold for each x. Similarly, Waeber et al. (2013), building on Jedynak et al. (2012), computes the multi-step optimal algorithm for a one-dimensional stochastic root-finding problem with an entropy objective. While these optimal multi-step methods are only directly applicable to very specific settings, they offer an opportunity to study the improvement possible more generally by going from one-step optimal to multi-step optimal. Surprisingly, in these settings, existing acquisition functions perform almost as well as the multi-step optimal algorithm. For example, experiments conducted in Cashore et al. (2016) show the KG acquisition function is within 98% of optimal in the problems computed there, and Waeber et al. (2013) shows that the entropy search acquisition function is multi-step optimal in the setting considered there. Generalizing from these results, it could be that the one-step acquisition functions are close enough to optimal that further improvement is not practically meaningful, or it could be that multi-step optimal algorithms will provide substantially better performance in yet-to-be-identified practically important settings.

5 Exotic Bayesian Optimization

Above we described methodology for solving the "standard" Bayesian optimization problem described in Section 1. This problem assumed a feasible set in which membership is easy to evaluate, such as a hyperrectangle or simplex; a lack of derivative information; and noise-free evaluations.

While there are quite a few applied problems that meet all of the assumptions of the standard problem, there are even more where one or more of these assumptions are broken. We call these "exotic" problems. Here, we describe some prominent examples and give references for more detailed reading. (Although we discuss noisy evaluations in this section on exotic problems, they are substantially less exotic than the others considered, and are often considered to be part of the standard problem.)

Noisy Evaluations GP regression can be extended naturally to observations with independent normally distributed noise of known variance (Rasmussen and Williams, 2006). This adds a diagonal term with entries equal to the variance of the noise to the covariance matrices in (3). In practice, this variance is not known, and so the most common approach is to assume that the noise is of common variance and to include this variance as a hyperparameter. It is also possible to perform inference assuming that the variance changes with the domain, by modeling the log of the variance with a second Gaussian process (Kersting et al., 2007).

The KG, ES, and PES acquisition functions apply directly in the setting with noise and they retain

their one-step optimality properties. One simply uses the posterior mean of the Gaussian process that includes noise.

Direct use of the EI acquisition function presents conceptual challenges, however, since the "improvement" that results from a function value is no longer easily defined, and $f(x)$ in (7) is no longer observed. Authors have employed a variety of heuristic approaches, substituting different normal distributions for the distribution of $f(x)$ in (7), and typically using the maximum of the posterior mean at the previously evaluated points in place of f_n^*. Popular substitutes for the distribution of $f(x)$ include the distribution of $\mu_{n+1}(x)$, the distribution of y_{n+1}, and continuing to use the distribution of $f(x)$ even though it is not observed. Because of these approximations, KG can outperform EI substantially in problems with substantial noise (Wu and Frazier, 2016; Frazier et al., 2009).

As an alternative approach to applying EI when measurements are noisy, Scott et al. (2011) considers noisy evaluations under the restriction made in the derivation of EI: that the reported solution needs to be a previously reported point. It then finds the one-step optimal place to sample under this assumption. Its analysis is similar to that used to derive the KG policy, except that we restrict $\widehat{x_*}$ to those points that have been evaluated.

Indeed, if we were to report a final solution after n measurements, it would be the point among $x_{1:n}$ with the largest value of $\mu_n(x)$, and it would have conditional expected value $\mu_n^{**} = \max_{i=1,\ldots,n} \mu_n(x_i)$. If we were to take one more sample at $x_{n+1} = x$, it would have conditional expected value under the new posterior of $\mu_{n+1}^{**} = \max_{i=1,\ldots,n+1} \mu_{n+1}(x_i)$. Taking the expected value of the difference, the value of sampling at x is

$$E_n \left[\mu_{n+1}^{**} - \mu_n^{**} | x_{n+1} = x\right]. \tag{13}$$

Unlike the case with noise-free evaluations, this sample may cause $\mu_{n+1}(x_i)$ to differ from $\mu_n(x_i)$ for $i \leq n$, necessitating a more complex calculation than in the noise-free setting (but a simpler calculation than for the KG policy). A procedure for calculating this quantity and its derivative is given in Scott et al. (2011). While we can view this acquisition function as an approximation to the KG acquisition function as Scott et al. (2011) does (they call it the KGCP acquisition function), we argue here that it is the most natural generalization of EI's assumptions to the case with noisy measurements.

Parallel Evaluations Performing evaluations in parallel using multiple computing resources allow obtaining multiple function evaluations in the time that would ordinarily be required to obtain just one with sequential evaluations. For this reason, parallel function evaluations is a conceptually appealing way to solve optimization problems in less time. EI, KG, ES, and PES can all be extended in a straightforward way to allow parallel function evaluations. For example, EI becomes

$$\mathrm{EI}_n(x^{(1:q)}) = E_n \left[\left[\max_{i=1,\ldots,q} f(x^{(i)}) - f_n^* \right]^+ \right], \tag{14}$$

where $x^{(1:q)} = (x^{(1)}, \ldots, x^{(q)})$ is a collection of points at which we are proposing to evaluate (Ginsbourger et al., 2007). Parallel EI (also called *multipoints EI* by Ginsbourger et al. (2007)) then proposes to evaluate the set of points that jointly maximize this criteria. This approach can also be used asynchronously, where we hold fixed those $x^{(i)}$ currently being evaluated and we allocate our idle computational resources by optimizing over their corresponding $x^{(j)}$.

Parallel EI (14) and other parallel acquisition functions are more challenging to optimize than their original sequential versions from Section 4. One innovation is the Constant Liar approximation to the parallel EI acquisition function (Ginsbourger et al., 2010), which chooses $x^{(i)}$ sequentially by assuming that $f(x^{(j)})$ for $j < i$ have been already observed, and have values equal to a constant (usually the expected value of $f(x^{(j)})$) under the posterior. This substantially speeds up computation. Expanding on this, Wang et al. (2016a) showed that infinitesimal perturbation analysis can produce random stochastic gradients that are unbiased estimates of $\nabla \mathrm{EI}_n(x^{(1:q)})$, which can then be used in multistart stochastic gradient ascent to optimize (14). This method has been used to implement the parallel EI procedure for as many as $q = 128$ parallel evaluations. Computational methods for parallel KG were developed by Wu and Frazier (2016), and are implemented in the Cornell MOE software package discussed in Section 6. That article follows the stochastic gradient ascent approach described above in Section 4.2, which generalizes well to the parallel setting.

Constraints In the problem posed in Section 1, we assumed that the feasible set was a simple one in which it was easy to assess membership. The literature has also considered the more general problem,

$$\max_x f(x)$$
$$\text{subject to } g_i(x) \geq 0, \quad i = 1, \ldots, I,$$

where the g_i are as expensive to evaluate as f. EI generalizes naturally to this setting when f and g_i can be evaluated without noise: improvement results when the evaluated x is feasible ($g_i(x) \geq 0$ for all x) and $f(x)$ is better than the best previously evaluated feasible point. This was proposed in Section 4 of Schonlau et al. (1998) and studied independently by Gardner et al. (2014). PES has also been studied for this setting (Hernández-Lobato et al., 2015).

Multi-Fidelity and Multi-Information Source Evaluations In multi-fidelity optimization, rather than a single objective f, we have a collection of information sources $f(x, s)$ indexed by s. Here, s controls the "fidelity", with lower s giving higher fidelity, and $f(x, 0)$ corresponding to the original objective. Increasing the fidelity (decreasing s) gives a more accurate estimate of $f(x, 0)$, but at a higher cost $c(s)$. For example, x might describe the design of an engineering system, and s the size of a mesh used in solving a partial differential equation that models the system. Or, s might describe the time horizon used in a steady-state simulation. Authors have also recently considered optimization of neural networks, where s indexes the number of iterations or amount of data used in training a machine learning algorithm (Swersky et al., 2014; Klein et al., 2016). Accuracy is modeled by supposing that $f(x, s)$ is equal to $f(x, 0)$ and is observed with noise whose variance $\lambda(s)$ increases with s, or by supposing that $f(x, s)$ provides deterministic evaluations with $f(x, s+1) - f(x, s)$ modeled by a mean 0 Gaussian process that varies with x. Both settings can be modeled via a Gaussian process on f, including both x and s as part of the modeled domain.

The overarching goal is to solve $\max_x f(x, 0)$ by observing $f(x, s)$ at a sequence of points and fidelities (x_n, s_n) with total cost $\sum_{n=1}^{N} c(s_n)$ less than some budget B. Work on multi-fidelity optimization includes Huang et al. (2006); Sóbester et al. (2004); Forrester et al. (2007); McLeod et al. (2017); Kandasamy et al. (2016).

In the more general problem of multi-information source optimization, we relax the assumption that the $f(\cdot, s)$ are ordered by s in terms of accuracy and cost. Instead, we simply have a function f taking a design input x and an information source input s, with $f(x, 0)$ being the objective, and $f(x, s)$ for $s \neq 0$ being observable with different biases relative to the objective, different amounts of noise, and different costs.

For example, x might represent the design of an aircraft's wing, $f(x, 0)$ the predicted performance of the wing under an accurate but slow simulator, and $f(x, s)$ for $s = 1, 2$ representing predicted performance under two inexpensive approximate simulators making different assumptions. It may be that $f(x, 1)$ is accurate for some regions of the search space and substantially biased in others, with $f(x, 2)$ being accurate in other regions. In this setting, the relative accuracy of $f(x, 1)$ vs. $f(x, 2)$ depends on x. Work on multi-information source optimization includes Lam et al. (2015); Poloczek et al. (2017).

EI is difficult to apply directly in these problems because evaluating $f(x, s)$ for $s \neq 0$ never provides an improvement in the best objective function value seen, $\max\{f(x_n, 0) : s_n = 0\}$. Thus, a direct translation of EI to this setting causes $EI = 0$ for $s \neq 0$, leading to measurement of only the highest fidelity. For this reason, the EI-based method from Lam et al. (2015) uses EI to select x_n assuming that $f(x, 0)$ will be observed (even if it will not), and uses a separate procedure to select s. KG, ES, and PES can be applied directly to these problems, as in Poloczek et al. (2017).

Random Environmental Conditions and Multi-Task Bayesian Optimization Closely related to multi-information source optimization is the pair of problems

$$\max_x \int f(x, w)p(w)\, dw,$$
$$\max_x \sum_w f(x, w)p(w),$$

where f is expensive to evaluate. These problems appear in the literature with a variety of names: optimization with random environmental conditions (Chang et al., 2001) in statistics, multi-task Bayesian optimization (Swersky et al., 2013) in machine learning, along with optimization of integrated response functions (Williams et al., 2000) and optimization with expensive integrands (Toscano-Palmerin and Frazier, 2018).

Rather than taking the objective $\int f(x,w)p(w)\,dw$ as our unit of evaluation, a natural approach is to evaluate $f(x,w)$ at a small number of w at an x of interest. This gives partial information about the objective at x. Based on this information, one can explore a different x, or resolve the current x with more precision. Moreover, by leveraging observations at w for a nearby x, one may already have substantial information about a particular $f(x,w)$ reducing the need to evaluate it. Methods that act on this intuition can substantially outperform methods that simply evaluate the full objective in each evaluation via numerical quadrature or a full sum (Toscano-Palmerin and Frazier, 2018).

This pair of problems arise in the design of engineering systems and biomedical problems, such as joint replacements (Chang et al., 2001) and cardiovascular bypass grafts (Xie et al., 2012), where $f(x,w)$ is the performance of design x under environmental condition w as evaluated by some expensive-to-evaluate computational model, $p(w)$ is some simple function (e.g., the normal density) describing the frequency with which condition w occurs, and our goal is to optimize average performance. It also arises in machine learning, in optimizing cross-validation performance. Here, we divide our data into chunks or "folds" indexed by w, and $f(x,w)$ is the test performance on fold w of a machine learning model trained without data from this fold.

Methods in this area include KG for noise-free (Xie et al., 2012) and general problems (Toscano-Palmerin and Frazier, 2018), PES (Swersky et al., 2013), and modifications of EI (Groot et al., 2010; Williams et al., 2000). As in multi-information source optimization, the unmodified EI acquisition function is inappropriate here because observing $f(x,w)$ does not provide an observation of the objective (unless all $w' \neq w$ have already been observed at that x) nor a strictly positive improvement. Thus, Groot et al. (2010) and Williams et al. (2000) use EI to choose x as if it we did observe the objective, and then use a separate strategy for choosing w.

Derivative Observations Finally, we discuss optimization with derivatives. Observations of $\nabla f(x)$, optionally with normally distributed noise, may be incorporated directly into GP regression (Rasmussen and Williams, 2006, Sect 9.4). Lizotte (2008) proposed using gradient information in this way in Bayesian optimization, together with the EI acquisition function, showing an improvement over BFGS (Liu and Nocedal, 1989). EI is unchanged by a proposed observation of $\nabla f(x)$ in addition to $f(x)$ as compared to its value when observing $f(x)$ alone. (Though, if previous derivative observations have contributed to the time n posterior, then that time-n posterior will differ from what it would be if we had observed only $f(x)$.) Thus, EI does not take advantage of the availability of derivative information to, for example, evaluate at points far away from previously evaluated ones where derivative information would be particularly useful. A KG method alleviating this problem was proposed by Wu et al. (2017). In other related work in this area, Osborne et al. (2009) proposed using gradient information to improve conditioning of the covariance matrix in GP regression, and Ahmed et al. (2016) proposed a method for choosing a single directional derivative to retain when observing gradients to improve the computational tractability of GP inference.

6 Software

There are a variety of codes for Bayesian optimization and Gaussian process regression. Several of these Gaussian process regression and Bayesian optimization packages are developed together, with the Bayesian optimization package making use of the Gaussian process regression package. Other packages are standalone, providing only either Gaussian process regression support or Bayesian optimization support. We list here several of the most prominent packages, along with URLs that are current as of June 2018.

- DiceKriging and DiceOptim are packages for Gaussian process regression and Bayesian optimization respectively, written in R. They are described in detail in Roustant et al. (2012) and are available from CRAN via `https://cran.r-project.org/web/packages/DiceOptim/index.html`.
- GPyOpt (`https://github.com/SheffieldML/GPyOpt`) is a python Bayesian optimization library built on top of the Gaussian process regression library GPy (`https://sheffieldml.github.io/GPy/`) both written and maintained by the machine learning group at Sheffield University.
- Metrics Optimization Engine (MOE, `https://github.com/Yelp/MOE`) is a Bayesian optimization library in C++ with a python wrapper that supports GPU-based computations for improved speed. It was developed at Yelp by the founders of the Bayesian optimization startup, SigOpt (`http://sigopt.com`). Cornell MOE (`https://github.com/wujian16/Cornell-MOE`) is built on MOE with changes that make it easier to install, and support for parallel and derivative-enabled knowledge-gradient algorithms.

- Spearmint (https://github.com/HIPS/Spearmint), with an older version under a different license available at https://github.com/JasperSnoek/spearmint, is a python Bayesian optimization library. Spearmint was written by the founders of the Bayesian optimization startup Whetlab, which was acquired by Twitter in 2015 Perez (2015).
- DACE (Design and Analysis of Computer Experiments) is a Gaussian process regression library written in MATLAB, available at http://www2.imm.dtu.dk/projects/dace/. Although it was last updated in 2002, it remains widely used.
- GPFlow (https://github.com/GPflow/GPflow) and GPyTorch (https://github.com/cornellius-gp/gpytorch) are python Gaussian process regression library built on top of Tensorflow (https://www.tensorflow.org/) and PyTorch (https://pytorch.org/) respectively.
- laGP (https://cran.r-project.org/web/packages/laGP/index.html) is an R package for Gaussian process regression and Bayesian optimization with support for inequality constraints.

7 Conclusion and Research Directions

We have introduced Bayesian optimization, first discussing GP regression, then the expected improvement, knowledge gradient, entropy search, and predictive entropy search acquisition functions. We then discussed a variety of exotic Bayesian optimization problems: those with noisy measurements; parallel evaluations; constraints; multiple fidelities and multiple information sources; random environmental conditions and multi-task BayesOpt; and derivative observations.

Many research directions present themselves in this exciting field. First, there is substantial room for developing a deeper theoretical understanding of Bayesian optimization. As described in Section 4.4, settings where we can compute multi-step optimal algorithms are extremely limited. Moreover, while the acquisition functions we currently use in practice seem to perform almost as well as optimal multi-step algorithms when we can compute them, we do not currently have finite-time bounds that explain their near-optimal empirical performance, nor do we know whether multi-step optimal algorithms can provide substantial practical benefit in yet-to-be-understood settings. Even in the asymptotic regime, relatively little is known about rates of convergence for Bayesian optimization algorithms: while Bull (2011) establishes a rate of convergence for expected improvement when it is combined with periodic uniform sampling, it is unknown whether removing uniform sampling results in the same or different rate.

Second, there is room to build Bayesian optimization methods that leverage novel statistical approaches. Gaussian processes (or variants thereof such as Snoek et al. (2014) and Kersting et al. (2007)) are used in most work on Bayesian optimization, but it seems likely that classes of problems exist where the objective could be better modeled through other approaches. It is both of interest to develop new statistical models that are broadly useful, and to develop models that are specifically designed for applications of interest.

Third, developing Bayesian optimization methods that work well in high dimensions is of great practical and theoretical interest. Directions for research include developing statistical methods that identify and leverage structure present in high-dimensional objectives arising in practice, which has been pursued by recent work including Wang et al. (2013, 2016b); Kandasamy et al. (2015). See also Shan and Wang (2010). It is also possible that new acquisition functions may provide substantial value in high dimensional problems.

Fourth, it is of interest to develop methods that leverage exotic problem structure unconsidered by today's methods, in the spirit of the Section 5. It may be particularly fruitful to combine such methodological development with applying Bayesian optimization to important real-world problems, as using methods in the real world tends to reveal unanticipated difficulties and spur creativity.

Fifth, substantial impact in a variety of fields seems possible through application of Bayesian optimization. One set of application areas where Bayesian optimization seems particularly well-positioned to offer impact is in chemistry, chemical engineering, materials design, and drug discovery, where practitioners undertake design efforts involving repeated physical experiments consuming years of effort and substantial monetary expense. While there is some early work in these areas (Ueno et al., 2016; Frazier and Wang, 2016; Negoescu et al., 2011; Seko et al., 2015; Ju et al., 2017) the number of researchers working in these fields aware of the power and applicability of Bayesian optimization is still relatively small.

Acknowledgments

While writing this tutorial, the author was supported by the Air Force Office of Scientific Research and the National Science Foundation (AFOSR FA9550-15-1-0038 and NSF CMMI-1254298, CMMI-1536895 DMR-1719875, DMR-1120296). The author would also like to thank Roman Garnett, whose suggestions helped shape the discussion of expected improvement with noise, and several anonymous reviewers.

References

Ahmed, M. O., Shahriari, B., and Schmidt, M. (2016). Do we need "harmless" Bayesian optimization and "first-order" Bayesian optimization. In *Neural Information Processing Systems 2016 Workshop on Bayesian Optimization*.

Berger, J. O. (2013). *Statistical Decision Theory and Bayesian Analysis*. Springer Science & Business Media.

Blum, J. R. (1954). Multidimensional stochastic approximation methods. *The Annals of Mathematical Statistics*, pages 737–744.

Booker, A., Dennis, J., Frank, P., Serafini, D., Torczon, V., and Trosset, M. (1999). A rigorous framework for optimization of expensive functions by surrogates. *Structural and Multidisciplinary Optimization*, 17(1):1–13.

Bottou, L. (2012). Stochastic gradient descent tricks. In Montavon, G., Orr, G. B., and Müller, K. R., editors, *Neural Networks: Tricks of the Trade*, pages 421–436. Springer.

Brochu, E., Cora, M., and de Freitas, N. (2009). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical Report TR-2009-023, Department of Computer Science, University of British Columbia. arXiv:1012.2599.

Bull, A. D. (2011). Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12(Oct):2879–2904.

Calvin, J. (1997). Average performance of a class of adaptive algorithms for global optimization. *The Annals of Applied Probability*, 7(3):711–730.

Calvin, J. and Žilinskas, A. (2005). One-dimensional global optimization for observations with noise. *Computers & Mathematics with Applications*, 50(1-2):157–169.

Calvin, J. and Žilinskas, A. (1999). On the convergence of the P-algorithm for one-dimensional global optimization of smooth functions. *Journal of Optimization Theory and Applications*, 102(3):479–495.

Calvin, J. and Žilinskas, A. (2000). One-dimensional P-algorithm with convergence rate O(n-3+δ) for smooth functions. *Journal of Optimization Theory and Applications*, 106(2):297–307.

Cashore, J. M., Kumarga, L., and Frazier, P. I. (2016). Multi-step Bayesian optimization for one-dimensional feasibility determination. *arXiv preprint arXiv:1607.03195*.

Chang, P. B., Williams, B. J., Bhalla, K. S. B., Belknap, T. W., Santner, T. J., Notz, W. I., and Bartel, D. L. (2001). Design and analysis of robust total joint replacements: finite element model experiments with environmental variables. *Journal of Biomechanical Engineering*, 123(3):239–246.

Chick, S. E. and Inoue, K. (2001). New two-stage and sequential procedures for selecting the best simulated system. *Operations Research*, 49(5):732–743.

Clark, C. E. (1961). The greatest of a finite set of random variables. *Operations Research*, 9(2):145–162.

Cover, T. M. and Thomas, J. A. (2012). *Elements of Information Theory*. John Wiley & Sons.

Dynkin, E. and Yushkevich, A. (1979). *Controlled Markov Processes*. Springer, New York.

Forrester, A., Sóbester, A., and Keane, A. (2008). *Engineering Design via Surrogate Modelling: A Practical Guide*. Wiley, West Sussex, UK.

Forrester, A. I., Sóbester, A., and Keane, A. J. (2007). Multi-fidelity optimization via surrogate modelling. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 463, pages 3251–3269. The Royal Society.

Frazier, P., Powell, W., and Dayanik, S. (2009). The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing*, 21(4):599–613.

Frazier, P. I. (2012). Tutorial: Optimization via simulation with bayesian statistics and dynamic programming. In Laroque, C., Himmelspach, J., Pasupathy, R., Rose, O., and Uhrmacher, A. M., editors, *Proceedings of the 2012 Winter Simulation Conference Proceedings*, pages 79–94, Piscataway, New Jersey. Institute of Electrical and Electronics Engineers, Inc.

Frazier, P. I., Powell, W. B., and Dayanik, S. (2008). A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439.

Frazier, P. I. and Wang, J. (2016). Bayesian optimization for materials design. In Lookman, T., Alexander, F. J., and Rajan, K., editors, *Information Science for Materials Discovery and Design*, pages 45–75. Springer.

Gardner, J. R., Kusner, M. J., Xu, Z. E., Weinberger, K. Q., and Cunningham, J. P. (2014). Bayesian optimization with inequality constraints. In *ICML*, pages 937–945.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2014). *Bayesian Data Analysis*, volume 2. CRC Press Boca Raton, FL.

Ginsbourger, D., Le Riche, R., and Carraro, L. (2007). A multi-points criterion for deterministic parallel global optimization based on kriging. In *International Conference on Nonconvex Programming, NCP07*, Rouen, France.

Ginsbourger, D., Le Riche, R., and Carraro, L. (2010). Kriging is well-suited to parallelize optimization. In Tenne, Y. and Goh, C. K., editors, *Computational Intelligence in Expensive Optimization Problems*, volume 2, pages 131–162. Springer.

Ginsbourger, D. and Riche, R. (2010). Towards Gaussian process-based optimization with finite time horizon. In Giovagnoli, A., Atkinson, A., Torsney, B., and May, C., editors, *mODa 9–Advances in Model-Oriented Design and Analysis*, pages 89–96. Springer.

González, J., Osborne, M., and Lawrence, N. (2016). GLASSES: Relieving the myopia of bayesian optimisation. In *Artificial Intelligence and Statistics*, pages 790–799.

Groot, P., Birlutiu, A., and Heskes, T. (2010). Bayesian monte carlo for the global optimization of expensive functions. In *ECAI*, pages 249–254.

Hennig, P. and Schuler, C. J. (2012). Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13:1809–1837.

Hernández-Lobato, J. M., Gelbart, M. A., Hoffman, M. W., Adams, R. P., and Ghahramani, Z. (2015). Predictive entropy search for bayesian optimization with unknown constraints. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, pages 1699–1707. JMLR. org.

Hernández-Lobato, J. M., Hoffman, M. W., and Ghahramani, Z. (2014). Predictive entropy search for efficient global optimization of black-box functions. In *Advances in neural information processing systems*, pages 918–926.

Ho, Y.-C., Cao, X., and Cassandras, C. (1983). Infinitesimal and finite perturbation analysis for queueing networks. *Automatica*, 19(4):439–445.

Huang, D., Allen, T., Notz, W., and Miller, R. (2006). Sequential kriging optimization using multiple-fidelity evaluations. *Structural and Multidisciplinary Optimization*, 32(5):369–382.

Jedynak, B., Frazier, P. I., and Sznitman, R. (2012). Twenty questions with noise: Bayes optimal policies for entropy loss. *Journal of Applied Probability*, 49(1):114–136.

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492.

Ju, S., Shiga, T., Feng, L., Hou, Z., Tsuda, K., and Shiomi, J. (2017). Designing nanostructures for phonon transport via Bayesian optimization. *Physical Review X*, 7.

Kaelbling, L. P., Littman, M. L., and Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285.

Kandasamy, K., Dasarathy, G., Oliva, J. B., Schneider, J., and Póczos, B. (2016). Gaussian process bandit optimisation with multi-fidelity evaluations. In *Advances in Neural Information Processing Systems*, pages 992–1000.

Kandasamy, K., Schneider, J., and Póczos, B. (2015). High dimensional bayesian optimisation and bandits via additive models. In *International Conference on Machine Learning*, pages 295–304.

Keane, A. (2006). Statistical improvement criteria for use in multiobjective design optimization. *AIAA Journal*, 44(4):879–891.

Kersting, K., Plagemann, C., Pfaff, P., and Burgard, W. (2007). Most likely heteroscedastic gaussian process regression. In *Proceedings of the 24th International Conference on Machine learning*, pages 393–400. ACM.

Kleijnen, J. P. et al. (2008). *Design and Analysis of Simulation Experiments*, volume 20. Springer.

Klein, A., Falkner, S., Bartels, S., Hennig, P., and Hutter, F. (2016). Fast Bayesian optimization of machine learning hyperparameters on large datasets. *arXiv preprint arXiv:1605.07079*.

Knowles, J. (2006). ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66.

Kushner, H. J. (1964). A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86(1):97–106.

Lam, R., Allaire, D. L., and Willcox, K. E. (2015). Multifidelity optimization using statistical surrogate modeling for non-hierarchical information sources. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 0143.

Lam, R., Willcox, K., and Wolpert, D. H. (2016). Bayesian optimization with a finite budget: An approximate dynamic programming approach. In *Advances in Neural Information Processing Systems*, pages 883–891.

Liu, D. C. and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528.

Lizotte, D. (2008). *Practical Bayesian Optimization*. PhD thesis, University of Alberta.

Lizotte, D., Wang, T., Bowling, M., and Schuurmans, D. (2007). Automatic gait optimization with Gaussian process regression. In *Proceedings of IJCAI*, pages 944–949.

Mahajan, A. and Teneketzis, D. (2008). Multi-armed bandit problems. In Hero, A., Castañón, D., Cochran, D., and Kastella, K., editors, *Foundations and Applications of Sensor Management*, pages 121–151. Springer.

Martí, R., Lozano, J. A., Mendiburu, A., and Hernando, L. (2016). Multi-start methods. *Handbook of Heuristics*, pages 1–21.

McLeod, M., Osborne, M. A., and Roberts, S. J. (2017). Practical bayesian optimization for variable cost objectives. *arXiv preprint arXiv:1703.04335*.

Mehdad, E. and Kleijnen, J. P. (2018). Efficient global optimisation for black-box simulation via sequential intrinsic kriging. *Journal of the Operational Research Society*, 69:1–13.

Milgrom, P. and Segal, I. (2002). Envelope theorems for arbitrary choice sets. *Econometrica*, 70(2):583–601.

Minka, T. P. (2001). *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology.

Močkus, J. (1975). On Bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404. Springer.

Močkus, J. (1989). *Bayesian Approach to Global Optimization: Theory and Applications*. Kluwer Academic Publishers.

Močkus, J. and Močkus, L. (1991). Bayesian approach to global optimization and application to multiobjective and constrained problems. *Journal of Optimization Theory and Applications*, 70(1):157–172.

Močkus, J., Tiesis, V., and Žilinskas, A. (1978). The application of Bayesian methods for seeking the extremum. In Dixon, L. and Szego, G., editors, *Towards Global Optimisation*, volume 2, pages 117–129. Elsevier Science Ltd., North Holland, Amsterdam.

Neal, R. M. (2003). Slice sampling. *Annals of Statistics*, 31(3):705–741.

Negoescu, D. M., Frazier, P. I., and Powell, W. B. (2011). The knowledge gradient algorithm for sequencing experiments in drug discovery. *INFORMS Journal on Computing*, 23(1):46–363.

Osborne, M. A., Garnett, R., and Roberts, S. J. (2009). Gaussian processes for global optimization. In *3rd International Conference on Learning and Intelligent Optimization (LION3)*, pages 1–15. Citeseer.

Packwood, D. (2017). *Bayesian Optimization for Materials Science*, volume 3. Springer.

Perez, S. (2015). Twitter acquires machine learning startup whetlab. *TechCrunch*. Accessed July 3, 2018.

Poloczek, M., Wang, J., and Frazier, P. (2017). Multi-information source optimization. In *Advances in Neural Information Processing Systems*, pages 4291–4301.

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, New York.

Rasmussen, C. and Williams, C. (2006). *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA.

Regis, R. and Shoemaker, C. (2005). Constrained global optimization of expensive black box functions using radial basis functions. *Journal of Global Optimization*, 31(1):153–171.

Regis, R. and Shoemaker, C. (2007a). Improved strategies for radial basis function methods for global optimization. *Journal of Global Optimization*, 37(1):113–135.

Regis, R. and Shoemaker, C. (2007b). Parallel radial basis function methods for the global optimization of expensive functions. *European Journal of Operational Research*, 182(2):514–535.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407.

Roustant, O., Ginsbourger, D., and Deville, Y. (2012). Dicekriging, diceoptim: Two r packages for the analysis of computer experiments by kriging-based metamodeling and optimization. *Journal of Statistical Software, Articles*, 51(1):1–55.

Salemi, P., Nelson, B. L., and Staum, J. (2014). Discrete optimization via simulation using Gaussian Markov random fields. In *Proceedings of the 2014 Winter Simulation Conference*, pages 3809–3820. IEEE Press.

Sasena, M. (2002). *Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations*. PhD thesis, University of Michigan.

Schonlau, M., Welch, W. J., and Jones, D. R. (1998). Global versus local search in constrained optimization of computer models. *Lecture Notes — Monograph Series*, 34:11–25.

Scott, W., Frazier, P. I., and Powell, W. B. (2011). The correlated knowledge gradient for simulation optimization of continuous parameters using Gaussian process regression. *SIAM Journal on Optimization*, 21(3):996–1026.

Seko, A., Togo, A., Hayashi, H., Tsuda, K., Chaput, L., and Tanaka, I. (2015). Prediction of low-thermal-conductivity compounds with first-principles anharmonic lattice-dynamics calculations and Bayesian optimization. *Physical Review Letters*, 115.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and de Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175.

Shan, S. and Wang, G. G. (2010). Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. *Structural and Multidisciplinary Optimization*, 41(2):219–241.

Shoemaker, C., Regis, R., and Fleming, R. (2007). Watershed calibration using multistart local optimization and evolutionary optimization with radial basis function approximation/calage au niveau du bassin versant à l'aide d'une optimisation locale à démarrage multiple et d'une optimisation évolutive avec approximation à fonctions de base radiale. *Hydrological Sciences Journal/Journal des Sciences Hydrologiques*, 52(3):450–465.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, pages 2951–2959.

Snoek, J., Swersky, K., Zemel, R., and Adams, R. (2014). Input warping for Bayesian optimization of non-stationary functions. In *International Conference on Machine Learning*, pages 1674–1682.

Sóbester, A., Leary, S., and Keane, A. (2004). A parallel updating scheme for approximating and optimizing high fidelity computer simulations. *Structural and Multidisciplinary Optimization*, 27(5):371–383.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT press Cambridge.

Swersky, K., Snoek, J., and Adams, R. P. (2013). Multi-task Bayesian optimization. In *Advances in Neural Information Processing Systems*, pages 2004–2012.

Swersky, K., Snoek, J., and Adams, R. P. (2014). Freeze-thaw bayesian optimization. *arXiv preprint arXiv:1406.3896*.

Toscano-Palmerin, S. and Frazier, P. I. (2018). Bayesian optimization with expensive integrands. *arXiv preprint arXiv:1803.08661*.

Ueno, T., Rhone, T. D., Hou, Z., Mizoguchi, T., and Tsuda, K. (2016). COMBO: An efficient Bayesian optimization library for materials science. *Materials Discovery*, 4:18–21.

Žilinskas, A. (1975). Single-step Bayesian search method for an extremum of functions of a single variable. *Cybernetics and Systems Analysis*, 11(1):160–166.

Waeber, R., Frazier, P. I., and Henderson, S. G. (2013). Bisection search with noisy responses. *SIAM Journal on Control and Optimization*, 51(3):2261–2279.

Wang, J., Clark, S. C., Liu, E., and Frazier, P. I. (2016a). Parallel Bayesian global optimization of expensive functions. *arXiv preprint arXiv:1602.05149*.

Wang, Z., Hutter, F., Zoghi, M., Matheson, D., and de Feitas, N. (2016b). Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, 55:361–387.

Wang, Z., Zoghi, M., Hutter, F., Matheson, D., De Freitas, N., et al. (2013). Bayesian optimization in high dimensions via random embeddings. In *IJCAI*, pages 1778–1784.

Williams, B. J., Santner, T. J., and Notz, W. I. (2000). Sequential design of computer experiments to minimize integrated response functions. *Statistica Sinica*, 10(4):1133–1152.

Wu, J. and Frazier, P. (2016). The parallel knowledge gradient method for batch Bayesian optimization. In *Advances in Neural Information Processing Systems*, pages 3126–3134.

Wu, J., Poloczek, M., Wilson, A. G., and Frazier, P. (2017). Bayesian optimization with gradients. In *Advances in Neural Information Processing Systems*, pages 5273–5284.

Xie, J. and Frazier, P. I. (2013). Sequential Bayes-optimal policies for multiple comparisons with a known standard. *Operations Research*, 61(5):1174–1189.

Xie, J., Frazier, P. I., Sankaran, S., Marsden, A., and Elmohamed, S. (2012). Optimization of computationally expensive simulations with Gaussian processes and parameter uncertainty: Application to cardiovascular surgery. In *50th Annual Allerton Conference on Communication, Control, and Computing*, pages 406–413. IEEE.