
Predictive Entropy Search for Bayesian Optimization with Unknown Constraints

José Miguel Hernández-Lobato¹

Harvard University, Cambridge, MA 02138 USA

JMH@SEAS.HARVARD.EDU

Michael A. Gelbart¹

Harvard University, Cambridge, MA 02138 USA

MGELBART@SEAS.HARVARD.EDU

Matthew W. Hoffman

University of Cambridge, Cambridge, CB2 1PZ, UK

MWH30@CAM.AC.UK

Ryan P. Adams

Harvard University, Cambridge, MA 02138 USA

RPA@SEAS.HARVARD.EDU

Zoubin Ghahramani

University of Cambridge, Cambridge, CB2 1PZ, UK

ZOUBIN@ENG.CAM.AC.UK

Abstract

Unknown constraints arise in many types of expensive black-box optimization problems. Several methods have been proposed recently for performing Bayesian optimization with constraints, based on the expected improvement (EI) heuristic. However, EI can lead to pathologies when used with constraints. For example, in the case of decoupled constraints—i.e., when one can independently evaluate the objective or the constraints—EI can encounter a pathology that prevents exploration. Additionally, computing EI requires a current best solution, which may not exist if none of the data collected so far satisfy the constraints. By contrast, information-based approaches do not suffer from these failure modes. In this paper, we present a new information-based method called Predictive Entropy Search with Constraints (PESC). We analyze the performance of PESC and show that it compares favorably to EI-based approaches on synthetic and benchmark problems, as well as several real-world examples. We demonstrate that PESC is an effective algorithm that provides a promising direction towards a unified solution for constrained Bayesian optimization.

1. Introduction

We are interested in finding the global minimum $\mathbf{x}_*$ of an objective function $f(\mathbf{x})$ over some bounded domain, typically $\mathcal{X} \subset \mathbb{R}^d$, subject to the non-negativity of a series of constraint functions $c_1, \dots, c_K$. This can be formalized as

$$\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \quad \text{s.t.} \quad c_1(\mathbf{x}) \geq 0, \dots, c_K(\mathbf{x}) \geq 0. \quad (1)$$

However, f and $c_1, \dots, c_K$ are unknown and can only be evaluated pointwise via expensive queries to black-boxes that provide noise-corrupted evaluations of f and $c_1, \dots, c_K$. We assume that f and each of the constraints c_k are defined over the entire space $\mathcal{X}$. We seek to find a solution to (1) with as few queries as possible. *Bayesian optimization* (Mockus et al., 1978) methods approach this type of problem by building a Bayesian model of the unknown objective function and/or constraints, using this model to compute an *acquisition function* that represents how useful each input $\mathbf{x}$ is thought to be as a next evaluation, and then maximizing this acquisition function to select a *suggestion* for function evaluation.

In this work we extend Predictive Entropy Search (PES) (Hernández-Lobato et al., 2014) to solve (1), an approach that we call Predictive Entropy Search with Constraints (PESC). PESC is an acquisition function that approximates the expected information gain about the value of the constrained minimizer $\mathbf{x}_*$. As we will show below, PESC is effective in practice and can be applied to a much wider variety of constrained problems than existing methods.

¹Authors contributed equally.

2. Related Work and Challenges

Most previous approaches to Bayesian optimization with unknown constraints are variants of expected improvement (EI) (Mockus et al., 1978; Jones et al., 1998). EI measures the expected amount by which observing at $\mathbf{x}$ leads to improvement over the current best value or *incumbent* η :

$$\alpha_{\text{EI}}(\mathbf{x}|\eta, \mathcal{D}) = \int \max(0, f(\mathbf{x}) - \eta) p(f(\mathbf{x})|\mathcal{D}) df(\mathbf{x}), \quad (2)$$

where $\mathcal{D}$ is the collected data.

2.1. Expected improvement with constraints

One way to use EI with constraints works by discounting EI by the posterior probability of a constraint violation. The resulting acquisition function, which we call expected improvement with constraints (EIC), is given by

$$\alpha_{\text{EIC}}(\mathbf{x}) = \alpha_{\text{EI}}(\mathbf{x}|\eta, \mathcal{D}^f) \prod_{k=1}^K p(c_k(\mathbf{x}) \geq 0|\mathcal{D}^k), \quad (3)$$

where $\mathcal{D}^f$ is the set of objective function observations and $\mathcal{D}^k$ is the set of observations for constraint k . Initially proposed by Schonlau et al. (1998), EIC has recently been independently developed in Snoek (2013); Gelbart et al. (2014); Gardner et al. (2014). In the constrained case, η is the smallest value of the posterior mean of f such that all the constraints are satisfied at the corresponding location.

2.2. Augmented Lagrangian

Gramacy et al. (2014) propose a combination of the expected improvement heuristic and the augmented Lagrangian (AL) optimization framework for constrained blackbox optimization. AL methods are a class of algorithms for constrained nonlinear optimization that work by iteratively optimizing the unconstrained AL:

$$L_A(\mathbf{x}|\lambda, p) = f(\mathbf{x}) + \sum_{k=1}^K \left[\frac{1}{2p} \min(0, c_k(\mathbf{x}))^2 - \lambda_k c_k(\mathbf{x}) \right]$$

where $p > 0$ is a penalty parameter and $\lambda \geq 0$ is an approximate Lagrange multiplier, both of which are updated at each iteration.

The method proposed by Gramacy et al. (2014) uses Bayesian optimization with EI to solve the unconstrained *inner* loop of the augmented Lagrangian formulation. AL is limited by requiring noiseless constraints so that p and λ can be updated at each iteration. In section 4.3 we show that PESC and EIC perform better than AL on the synthetic benchmark problem considered in Gramacy et al. (2014), even when the AL method has access to the true objective function and PESC and EIC do not.

2.3. Integrated expected conditional improvement

Gramacy & Lee (2011) propose an acquisition function based on the integrated expected conditional improvement (IECI), which is given by

$$\alpha_{\text{IECI}}(\mathbf{x}) = \int [\alpha_{\text{EI}}(\mathbf{x}') - \alpha_{\text{EI}}(\mathbf{x}'|\mathbf{x})] h(\mathbf{x}') d\mathbf{x}', \quad (4)$$

where $\alpha_{\text{EI}}(\mathbf{x}')$ is the expected improvement at $\mathbf{x}'$ and $\alpha_{\text{EI}}(\mathbf{x}'|\mathbf{x})$ is the expected improvement at $\mathbf{x}'$ when the objective has been evaluated at $\mathbf{x}$, but without knowing the value obtained. The IECI at $\mathbf{x}$ is the expected reduction in improvement at $\mathbf{x}'$ under the density $h(\mathbf{x}')$ caused by observing the objective at that location, where $h(\mathbf{x}')$ is the probability of all the constraints being satisfied at $\mathbf{x}'$. Gelbart et al. (2014) compare IECI with EIC for optimizing the hyper-parameters of a topic model with constraints on the entropy of the per-topic word distribution and show that EIC outperforms IECI for this problem.

2.4. Expected volume reduction

Picheny (2014) proposes to sequentially explore the location that yields that largest the expected volume reduction (EVR) of the feasible region below the best feasible objective value η found so far. This quantity is given by integrating the product of the probability of improvement and the probability of feasibility. That is,

$$\alpha_{\text{EVR}}(\mathbf{x}) = - \int p[f(\mathbf{x}') \leq \min(\eta, f(\mathbf{x}))] h(\mathbf{x}') d\mathbf{x}', \quad (5)$$

where, as in IECI, $h(\mathbf{x}')$ is the probability that the constraints are satisfied at $\mathbf{x}'$. This step-wise uncertainty reduction approach is similar to PESC in that both methods work by reducing a specific type of uncertainty measure (entropy for PESC and expected volume for EVR).

2.5. Challenges

EI-based methods for constrained optimization have several issues. First, when no point in the search space is feasible under the above definition, η does not exist and the EI cannot be computed. This issue affects EIC, IECI, and EVR. To address this issue, Gelbart et al. (2014) modify EIC to ignore the factor $\text{EI}(\mathbf{x}|\eta, \mathcal{D}^f)$ in (3) and only consider the posterior probability of the constraints being satisfied when η is not defined. The resulting acquisition function focuses only on searching for a feasible location and ignores learning about the objective f .

Furthermore, Gelbart et al. (2014) identify a pathology with EIC when one is able to separately evaluate the objective or the constraints, i.e., the *decoupled* case. The best solution $\mathbf{x}_*$ must satisfy a conjunction of low objective value and high (non-negative) constraint values. By only evaluating the objective or a single constraint, this conjunction

cannot be satisfied by a single observation under a myopic search policy. Thus, the new observed $\mathbf{x}$ cannot become the new incumbent as a result of a decoupled observation and the expected improvement is zero. Therefore standard EIC fails in the decoupled setting. Gelbart et al. (2014) circumvent this pathology by treating decoupling as a special case and using a two-stage acquisition function: first, $\mathbf{x}$ is chosen with EIC, and then, given $\mathbf{x}$, the task (whether to evaluate the objective or one of the constraints) is chosen with the method in Villemonteix et al. (2009). This approach does not take full advantage of the available information in the way a joint selection of $\mathbf{x}$ and the task would. Like EIC, the methods AL, IECI, and EVR are also not easily extended to the decoupled setting.

In addition to this difficulties, EVR and IECI are limited by having to compute the integrals in (4) and (5) over the entire domain, which is done numerically over a grid on $\mathbf{x}'$ (Gramacy & Lee, 2011; Picheny, 2014). The resulting acquisition function must then be globally optimized, which also requires a grid on $\mathbf{x}$. This nesting of grid operations limits the application of this method to small d .

Our new method, PESC, does not suffer from these pathologies. First, the PESC acquisition function does not depend on the current best feasible solution, so it can operate coherently even when there is not yet a feasible solution. Second, PESC naturally separates the contribution of each task (objective or constraint) in its acquisition function. As a result, no pathology arises in the decoupled case and, thus, no *ad hoc* modifications to the acquisition function are required. Third, likewise EVR and IECI, PESC also involves computing a difficult integral (over the posterior on $\mathbf{x}_*$). However, this can be done efficiently using the sampling approach described in Hernández-Lobato et al. (2014). Furthermore, in addition to its increased generality, our experiments show that PESC performs favorably when compared to EIC and AL even in the basic setting of joint evaluations to which these methods are most suited.

3. Predictive entropy search with constraints

We seek to maximize information about the location $\mathbf{x}_*$, the constrained global minimum, whose posterior distribution is $p(\mathbf{x}_*|\mathcal{D}^0, \dots, \mathcal{D}^K)$. We assume that f and $c_1, \dots, c_K$ follow independent Gaussian process (GP) priors (see, e.g., Rasmussen & Williams, 2006) and that observation noise is i.i.d. Gaussian with zero mean. GPs are widely-used probabilistic models for Bayesian nonparametric regression which provide a flexible framework for working with unknown response surfaces.

In the coupled setting we will let $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n \leq N}$ denote all the observations up to step N , where $\mathbf{y}_n$ is a vector collecting the objective and constraint observations at

step n . The next query $\mathbf{x}_{N+1}$ can then be defined as that which maximizes the expected reduction in the differential entropy $H[\cdot]$ of the posterior on $\mathbf{x}_*$. We can write the PESC acquisition function as

$$\alpha(\mathbf{x}) = H[\mathbf{x}_*|\mathcal{D}] - \mathbb{E}_{\mathbf{y}} \{H[\mathbf{x}_*|\mathcal{D} \cup (\mathbf{x}, \mathbf{y})]\} \quad (6)$$

where the expectation is taken with respect to the posterior distribution on the noisy evaluations of f and $c_1, \dots, c_K$ at $\mathbf{x}$, that is, $p(\mathbf{y}|\mathcal{D}, \mathbf{x})$.

The exact computation of the above expression is infeasible in practice. Instead, we follow Houlisby et al. (2012); Hernández-Lobato et al. (2014) and take advantage of the symmetry of mutual information, rewriting this acquisition function as the mutual information between $\mathbf{y}$ and $\mathbf{x}_*$ given the collected data $\mathcal{D}$. That is,

$$\alpha(\mathbf{x}) = H[\mathbf{y}|\mathcal{D}, \mathbf{x}] - \mathbb{E}_{\mathbf{x}_*} \{H[\mathbf{y}|\mathcal{D}, \mathbf{x}, \mathbf{x}_*]\} \quad (7)$$

where the expectation is now with respect to the posterior $p(\mathbf{x}_*|\mathcal{D})$ and where $p(\mathbf{y}|\mathcal{D}, \mathbf{x}, \mathbf{x}_*)$ is the posterior predictive distribution for objective and constraint values given past data and the location of the global solution to the constrained optimization problem $\mathbf{x}_*$. We call $p(\mathbf{y}|\mathcal{D}, \mathbf{x}, \mathbf{x}_*)$ the *conditioned predictive distribution* (CPD).

The first term on the right-hand side of (7) is straightforward to compute: it is the entropy of a product of independent Gaussians, which is given by

$$H(\mathbf{y}|\mathcal{D}, \mathbf{x}) = \log v_f + \sum_{k=1}^K \log v_k + \frac{K+1}{2} \log(2\pi e), \quad (8)$$

where v_f and v_k are the predictive variances of the objective and constraints, respectively. However, the second term in the right-hand side of (7) has to be approximated. For this, we first approximate the expectation by averaging over samples of $\mathbf{x}_*$ approximately drawn from $p(\mathbf{x}_*|\mathcal{D})$. To sample $\mathbf{x}_*$, we first approximately draw f and $c_1, \dots, c_K$ from their GP posteriors using a finite parameterization of these functions. Then we solve a constrained optimization problem using the sampled functions to yield a sample of $\mathbf{x}_*$. This optimization approach is an extension of the approach described in more detail by Hernández-Lobato et al. (2014), extended to the constrained setting. For each value of $\mathbf{x}_*$ generated by this procedure, we approximate the CPD $p(\mathbf{y}|\mathcal{D}, \mathbf{x}, \mathbf{x}_*)$ as described in the next section.

3.1. Approximating the CPD

Let $\mathbf{z} = [f(\mathbf{x}), c_1(\mathbf{x}), \dots, c_K(\mathbf{x})]^T$ denote the concatenated vector of the noise-free objective and constraint values at $\mathbf{x}$. We can approximate the CPD by first approximating the posterior predictive distribution of $\mathbf{z}$ conditioned on $\mathcal{D}$, $\mathbf{x}$, and $\mathbf{x}_*$, which we call the *noise free CPD*

(NFCPD), and then convolving that approximation with additive Gaussian noise of variance $\sigma_0^2, \dots, \sigma_K^2$.

We first consider the distribution $p(\mathbf{x}_* | f, c_1, \dots, c_K)$. The variable $\mathbf{x}_*$ is in fact a deterministic function of the latent functions $f, c_1, \dots, c_K$: in particular, $\mathbf{x}_*$ is the global minimizer if and only if (i) all constraints are satisfied at $\mathbf{x}_*$ and (ii) $f(\mathbf{x}_*)$ is the smallest feasible value in the domain. We can informally translate these deterministic conditions into a conditional probability:

$$p(\mathbf{x}_* | f, c_1, \dots, c_K) = \left[\prod_{k=1}^K \Theta[c_k(\mathbf{x}_*)] \right] \prod_{\mathbf{x}' \in \mathcal{X}} \Psi(\mathbf{x}'), \quad (9)$$

where $\Psi(\mathbf{x}')$ is defined as

$$\left(\prod_{k=1}^K \Theta[c_k(\mathbf{x}')] \right) \Theta[f(\mathbf{x}') - f(\mathbf{x}_*)] + \left(1 - \prod_{k=1}^K \Theta[c_k(\mathbf{x}')] \right)$$

and the symbol Θ denotes the Heaviside step function with the convention that $\Theta(0) = 1$. The first product in (9) encodes condition (i) and the infinite product over $\Psi(\mathbf{x}')$ encodes condition (ii). Note that $\Psi(\mathbf{x}')$ also depends on $\mathbf{x}_*$ and $f, c_1, \dots, c_K$; we use the notation $\Psi(\mathbf{x}')$ for brevity.

Because $\mathbf{z}$ is simply a vector containing the values of $f, c_1, \dots, c_K$ at $\mathbf{x}$, $\mathbf{z}$ is also a deterministic function of $f, c_1, \dots, c_K$ and we can write $p(\mathbf{z} | f, c_1, \dots, c_K, \mathbf{x})$ using Dirac delta functions to pick out the values at $\mathbf{x}$:

$$p(\mathbf{z} | f, c_1, \dots, c_K, \mathbf{x}) = \delta[z_0 - f(\mathbf{x})] \prod_{k=1}^K \delta[z_k - c_k(\mathbf{x})]. \quad (10)$$

We can now write the NFCPD by i) noting that $\mathbf{z}$ is independent of $\mathbf{x}^*$ given $f, c_1, \dots, c_K$, ii) multiplying the product of (9) and (10) by $p(f, c_1, \dots, c_K | \mathcal{D})$ and iii) integrating out the latent functions $f, c_1, \dots, c_K$:

$$p(\mathbf{z} | \mathcal{D}, \mathbf{x}, \mathbf{x}_*) \propto \int \delta[z_0 - f(\mathbf{x})] \left[\prod_{k=1}^K \delta[z_k - c_k(\mathbf{x})] \right] \left[\prod_{k=1}^K \Theta[c_k(\mathbf{x}_*)] \right] \left[\prod_{\mathbf{x}' \neq \mathbf{x}} \Psi(\mathbf{x}') \right] \Psi(\mathbf{x}) p(f, c_1, \dots, c_K | \mathcal{D}) df dc_1 \dots dc_K, \quad (11)$$

where $p(f, c_1, \dots, c_K | \mathcal{D})$ is an infinite-dimensional Gaussian given by the GP posterior on $f, c_1, \dots, c_K$, and we have separated $\Psi(\mathbf{x})$ out from the infinite product over $\mathbf{x}'$.

We find a Gaussian approximation to (11) in several steps. The general approach is to separately approximate the factors that do and do not depend on $\mathbf{x}$, so that the computations associated with the latter factors can be reused rather than recomputed for each $\mathbf{x}$. In (11), the only factors that depend on $\mathbf{x}$ are the deltas in the first line, and $\Psi(\mathbf{x})$.

Let $\mathbf{f}$ denote the $(N+1)$ -dimensional vector containing objective function evaluations at $\mathbf{x}_*$ and $\mathbf{x}_1, \dots, \mathbf{x}_N$, and define constraint vectors $\mathbf{c}_1, \dots, \mathbf{c}_K$ similarly. Then, we approximate (11) by conditioning only on $\mathbf{f}$ and $\mathbf{c}_1, \dots, \mathbf{c}_K$,

rather than the full $f, c_1, \dots, c_K$. We first approximate the factors in (11) that do not depend on $\mathbf{x}$ as

$$q_1(\mathbf{f}, \mathbf{c}_1, \dots, \mathbf{c}_K) = \left[\prod_{k=1}^K \Theta[c_{k0}] \right] \left[\prod_{n=1}^N \Psi(\mathbf{x}_n) \right] p(\mathbf{f}, \mathbf{c}_1, \dots, \mathbf{c}_K | \mathcal{D}) \quad (12)$$

where $p(\mathbf{f}, \mathbf{c}_1, \dots, \mathbf{c}_K | \mathcal{D})$ is the GP predictive distribution for objective and constraint values. Because (12) is not tractable, we approximate the normalized version of q_1 with a product of Gaussians using expectation propagation (EP) (Minka, 2001). In particular, we obtain

$$Z_1^{-1} q_1(\mathbf{f}, \mathbf{c}_1, \dots, \mathbf{c}_K) \approx q_2(\mathbf{f}, \mathbf{c}_1, \dots, \mathbf{c}_K) = \mathcal{N}(\mathbf{f} | \mathbf{m}_0, \mathbf{V}_0) \prod_{k=1}^K \mathcal{N}(\mathbf{c}_k | \mathbf{m}_k, \mathbf{V}_k), \quad (13)$$

where Z_1 is the normalization constant of q_1 and $(\mathbf{m}_k, \mathbf{V}_k)$ for $k = 0, \dots, K$ are the mean and covariance terms determined by EP. See the supplementary material for details on the EP approximation. Roughly speaking, EP approximates each true (but intractable) factor in (12) with a Gaussian factor whose parameters are iteratively refined. The product of all these Gaussian factors produces a tractable Gaussian approximation to (12).

We now approximate the portion of (11) that does depend on $\mathbf{x}$, namely the first line and the factor $\Psi(\mathbf{x})$, by replacing the deltas with $p(\mathbf{z} | \mathbf{f}, c_1, \dots, c_K)$, the $K+1$ dimensional, Gaussian conditional distribution given by the GP priors on $f, c_1, \dots, c_K$. Our full approximation to (11) is then

$$p(\mathbf{z} | \mathcal{D}, \mathbf{x}, \mathbf{x}_*) \approx Z_2^{-1} \int p(\mathbf{z} | \mathbf{f}, \mathbf{c}_1, \dots, \mathbf{c}_K) \Psi(\mathbf{x}) q_2(\mathbf{f}, \mathbf{c}_1, \dots, \mathbf{c}_K) d\mathbf{f} d\mathbf{c}_1 \dots d\mathbf{c}_K, \quad (14)$$

where Z_2 is a normalization constant. From here, we analytically marginalize out all integration variables except $f_0 = f(\mathbf{x}_*)$; see the supplementary material for the full details. This calculation, and those that follow, must be repeated for every $\mathbf{x}$; however, the EP approximation in (13) can be reused over all $\mathbf{x}$. After performing the integration, we arrive at

$$p(\mathbf{z} | \mathcal{D}, \mathbf{x}, \mathbf{x}_*) \approx \frac{1}{Z_3} \int \Psi(\mathbf{x}) \mathcal{N}([z_0, f_0] | \mathbf{m}'_0, \mathbf{V}'_0) \prod_{k=1}^K \mathcal{N}(z_k | m'_k, v'_k) df_0, \quad (15)$$

where $z_0 = f(\mathbf{x})$. Details on how to compute the means $m'_1, \dots, m'_K$ and variances $v'_1, \dots, v'_K$, as well as the 2-dimensional mean vector $\mathbf{m}'_0$ and the 2×2 covariance matrix $\mathbf{V}'_0$ can be found in the supplementary material.

We perform one final approximation to (15). We approximate this distribution with a product of independent Gaussians that have the same marginal means and variances as (15). This corresponds to a single iteration of EP; see the supplementary material for details.

3.2. The PESC acquisition function

By approximating the NFCPD with a product of independent Gaussians, we can approximate the entropy in the CPD by performing the following operations. First, we add the noise variances to the marginal variances of our final approximation of the NFCPD and second, we compute the entropy with (8). The PESC acquisition function, which approximates (6), is then

$$\alpha_{\text{PESC}}(\mathbf{x}) = \left\{ \log v_f^{\text{PD}}(\mathbf{x}) + \sum_{k=1}^K \log v_k^{\text{PD}}(\mathbf{x}) \right\} - \frac{1}{M} \sum_{m=1}^M \left\{ \log v_f^{\text{CPD}}(\mathbf{x} | \mathbf{x}_*^{(m)}) + \sum_{k=1}^K \log v_k^{\text{CPD}}(\mathbf{x} | \mathbf{x}_*^{(m)}) \right\}, \quad (16)$$

where M is the number of samples drawn from $p(\mathbf{x}_* | \mathcal{D})$, $\mathbf{x}_*^{(m)}$ is the m -th of these samples, $v_f^{\text{PD}}(\mathbf{x})$ and $v_k^{\text{PD}}(\mathbf{x})$ are the predictive variances for the noisy evaluations of f and c_k at $\mathbf{x}$, respectively, and $v_f^{\text{CPD}}(\mathbf{x} | \mathbf{x}_*^{(m)})$ and $v_k^{\text{CPD}}(\mathbf{x} | \mathbf{x}_*^{(m)})$ are the approximated marginal variances of the CPD for the noisy evaluations of f and c_k at $\mathbf{x}$ given that $\mathbf{x}_* = \mathbf{x}_*^{(m)}$. Marginalization of (16) over the GP hyper-parameters can be done efficiently as in [Hernández-Lobato et al. \(2014\)](#).

The PESC acquisition function is additive in the expected amount of information that is obtained from the evaluation of each task (objective or constraint) at any particular location $\mathbf{x}$. For example, the expected information gain obtained from the evaluation of f at $\mathbf{x}$ is given by the term $\frac{1}{M} \sum_{m=1}^M \left[\log v_f^{\text{PD}}(\mathbf{x}) - \log v_f^{\text{CPD}}(\mathbf{x} | \mathbf{x}_*^{(m)}) \right]$ in (16). The other K terms in (16) measure the corresponding contribution from evaluating each of the constraints. This allows PESC to easily address the decoupled scenario when one can independently evaluate the different functions at different locations. In other words, Equation (16) is a sum of individual acquisition functions, one for each function that we can evaluate. Existing methods for Bayesian optimization with unknown constraints (described in Section 2) do not possess this desirable property. Finally, the complexity of PESC is of order $O(MKN^3)$ per iteration in the coupled setting. As with unconstrained PES, this is dominated by the cost of a matrix inversion in the EP step.

4. Experiments

We evaluate the performance of PESC through experiments with i) synthetic functions sampled from the GP prior distribution, ii) analytic benchmark problems previously used in the literature on Bayesian optimization with unknown constraints and iii) real-world constrained optimization problems.

For case i) above, the synthetic functions sampled from the GP prior are generated following the same experimental set

up as in [Hennig & Schuler \(2012\)](#) and [Hernández-Lobato et al. \(2014\)](#). The search space is the unit hypercube of dimension d , and the ground truth objective f is a sample from a zero-mean GP with a squared exponential covariance function of unit amplitude and length scale $\ell = 0.1$ in each dimension. We represent the function f by first sampling from the GP prior on a grid of 1000 points generated using a Halton sequence (see [Leobacher & Pillichshammer, 2014](#)) and then defining f as the resulting GP posterior mean. We use a single constraint function c_1 whose ground truth is sampled in the same way as f . The evaluations for f and c_1 are contaminated with i.i.d. Gaussian noise with variance $\sigma_f^2 = \sigma_1^2 = 0.01$.

4.1. Accuracy of the PESC approximation

We first analyze the accuracy of the approximation to (7) generated by PESC. We compare the PESC approximation with a ground truth for (7) obtained by rejection sampling (RS). The RS method works by discretizing the search space using a uniform grid. The expectation with respect to $p(\mathbf{x}_* | \mathcal{D}_n)$ in (7) is then approximated by Monte Carlo. To achieve this, f and $c_1, \dots, c_K$ are sampled on the grid and the grid cell with positive $c_1, \dots, c_K$ (feasibility) and the lowest value of f (optimality) is selected. For each sample of $\mathbf{x}_*$ generated by this procedure, $H[p(\mathbf{y} | \mathcal{D}_n, \mathbf{x}, \mathbf{x}_*)]$ is approximated by rejection sampling: we select those samples of f and $c_1, \dots, c_K$ whose corresponding feasible optimal solution is the sampled $\mathbf{x}_*$ and reject the other samples. We then assume that the selected samples for f and $c_1, \dots, c_K$ are independent and have Gaussian marginal distributions. Under this assumption, $H[p(\mathbf{y} | \mathcal{D}_n, \mathbf{x}, \mathbf{x}_*)]$ can be approximated using the formula for the entropy of independent Gaussian random variables, with the variance parameters in this formula being equal to the empirical marginal variances of the selected samples of f and $c_1, \dots, c_K$ at $\mathbf{x}$ plus the corresponding noise variances σ_f^2 and $\sigma_1^2, \dots, \sigma_K^2$.

The left plot in Figure 1 shows the posterior distribution for f and c_1 given 5 evaluations sampled from the GP prior with $d = 1$. The posterior is computed using the optimal GP hyperparameters. The corresponding approximations to (7) generated by PESC and RS are shown in the middle plot of Figure 1. Both PESC and RS use a total of 50 samples from $p(\mathbf{x}_* | \mathcal{D}_n)$ when approximating the expectation in (7). The PESC approximation is very accurate, and importantly its maximum value is very close to the maximum value of the RS approximation.

One disadvantage of the RS method is its high cost, which scales with the size of the grid used. This grid has to be large to guarantee good performance, especially when d is large. An alternative is to use a small dynamic grid that changes as data is collected. Such a grid can be obtained

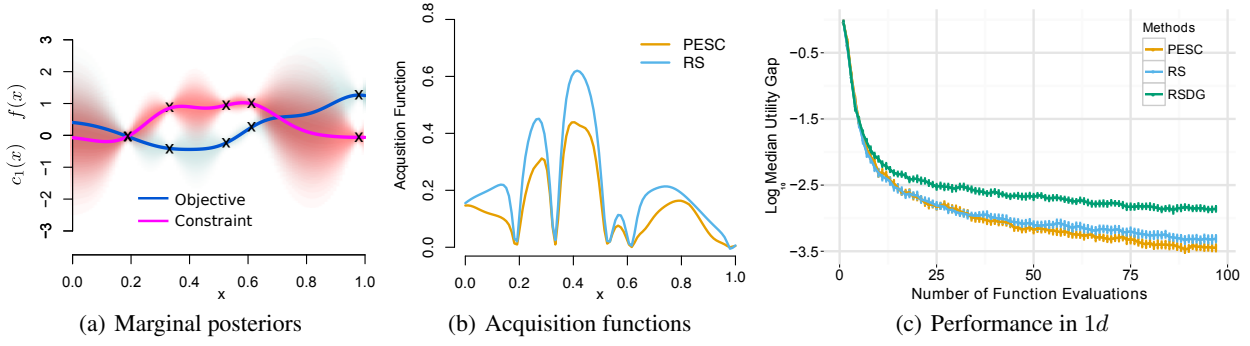


Figure 1. Assessing the accuracy of the PESC approximation. (a) Marginal posterior predictive distributions for the objective and constraint given some collected data denoted by $\times$'s. (b) PESC and RS acquisition functions given the data in (a). (c) Median utility gap for PESC, RS and RSDG in the experiments with synthetic functions sampled from the GP prior with $d = 1$.

by sampling from $p(\mathbf{x}_*|\mathcal{D}_n)$ using the same approach as in PESC. The samples obtained would then form the dynamic grid. The resulting method is called Rejection Sampling with a Dynamic Grid (RSDG).

We compare the performance of PESC, RS and RSDG in experiments with synthetic data corresponding to 500 pairs of f and c_1 sampled from the GP prior with $d = 1$. At each iteration, RSDG draws the same number of samples of $\mathbf{x}_*$ as PESC. We assume that the GP hyperparameter values are known to each method. Recommendations are made by finding the location with lowest posterior mean for f such that c_1 is non-negative with probability at least $1 - \delta_1$, where $\delta_1 = 0.05$. For reporting purposes, we set the utility $u(\mathbf{x})$ of a recommendation $\mathbf{x}$ to be $f(\mathbf{x})$ if $\mathbf{x}$ satisfies the constraint, and otherwise a penalty value of the worst (largest) objective function value achievable in the search space. For each recommendation at $\mathbf{x}$, we compute the utility gap $|u(\mathbf{x}) - u(\mathbf{x}_*)|$, where $\mathbf{x}_*$ is the true solution of the optimization problem. Each method is initialized with the same three random points drawn with Latin hypercube sampling.

The right plot in Figure 1 shows the median of the utility gap for each method across the 500 realizations of f and c_1 . The x -axis in this plot is the number of joint function evaluations for f and c_1 . We report the median because the empirical distribution of the utility gap is heavy-tailed and in this case the median is more representative of the location of the bulk of the data than the mean. The heavy tails arise because we are measuring performance across 500 different optimization problems with very different degrees of difficulty. In this and all following experiments, standard errors on the reported plot are computed using the bootstrap. The plot shows that PESC and RS are better than RSDG. Furthermore, PESC is very similar to RS, with PESC even performing slightly better at the end of the data collection process since PESC is not limited by a finite

grid as RS is. These results show that PESC yields a very accurate approximation of the information gain. Furthermore, although RSDG performs worse than PESC, RSDG is faster because the rejection sampling operation (with a small grid) is less expensive than the EP algorithm. Thus, RSDG is an attractive alternative to PESC when the available computing time is very limited.

4.2. Synthetic functions in 2 and 8 input dimensions

We also compare the performance of PESC and RSDG with that of EIC (Section 2.1) using the same experimental protocol as in the previous section, but with dimensionalities $d = 2$ and $d = 8$. We do not compare with RS here because its use of grids does not scale to higher dimensions. Figure 4.1 shows the utility gap for each method across 500 different samples of f and c_1 from the GP prior with $d = 2$ (a) and $d = 8$ (b). Overall, PESC is the best method, followed by RSDG and EIC. RSDG performs similarly to PESC when $d = 2$, but is significantly worse when $d = 8$. This shows that, when d is high, grid based approaches (e.g. RSDG) are at a disadvantage with respect to methods that do not require a grid (e.g. PESC).

4.3. A toy problem

We compare PESC with EIC and AL (Section 2.2) in the toy problem described in Gramacy et al. (2014). We seek to minimize the function $f(x) = x_1 + x_2$, subject to the constraint functions $c_1(\mathbf{x}) \geq 0$ and $c_2(\mathbf{x}) \geq 0$, given by

$$c_1(\mathbf{x}) = 0.5 \sin(2\pi(x_1^2 - 2x_2)) + x_1 + 2x_2 - 1.5, \quad (17)$$

$$c_2(\mathbf{x}) = -x_1^2 - x_2^2 + 1.5, \quad (18)$$

where $\mathbf{x}$ is confined to the unit square. The evaluations for f , c_1 and c_2 are noise-free. We compare PESC and EIC with $\delta_1 = \delta_2 = 0.025$ and a squared exponential GP kernel. PESC uses 10 samples from $p(\mathbf{x}_*|\mathcal{D}_n)$ when approx-

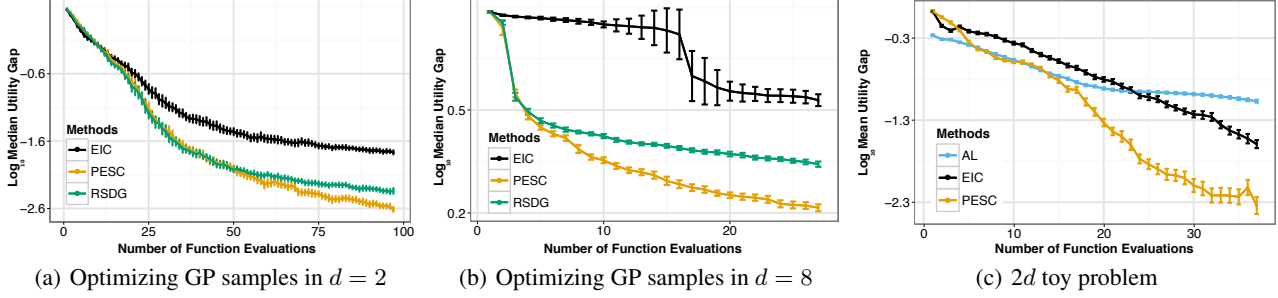


Figure 2. Assessing PESC on synthetic problems. (a,b) Compare PESC to EIC and RSDG on optimizing samples from the GP in dimension 2 and 8 respectively, and (c) compares PESC to AL and EIC.

imating the expectation in (7). We use the AL implementation provided by Gramacy et al. (2014) in the R package *laGP* which is based on the squared exponential kernel and assumes the objective f is known. Thus, in order for this implementation to be used, AL has an advantage over other methods in that it has access to the true objective function. In all three methods, the GP hyperparameters are estimated by maximum likelihood.

Figure 2(c) shows the mean utility gap for each method across 500 independent realizations. Each realization corresponds to a different initialization of the methods with three data points selected with Latin hypercube sampling. Here, we report the mean because we are now measuring performance across realizations of the same optimization problem and the heavy-tailed effect described in Section 4.1 is less severe. The results show that PESC is significantly better than EIC and AL for this problem. EIC is superior to AL, which performs slightly better at the beginning, presumably because it has access to the ground truth objective f .

4.4. Finding a fast neural network

In this experiment, we tune the hyperparameters of a three-hidden-layer neural network subject to the constraint that the prediction time must not exceed 2 ms on a GeForce GTX 580 GPU (also used for training). The search space consists of 12 parameters: 2 learning rate parameters (initial and decay rate), 2 momentum parameters (initial and final), 2 dropout parameters (input layer and other layers), 2 other regularization parameters (weight decay and max weight norm), the number of hidden units in each of the 3 hidden layers, the activation function (RELU or sigmoid). The network is trained using the *deepnet* package¹, and the prediction time is computed as the average time of 1000 predictions, each for a batch of size 128. The network is trained on the MNIST digit classification task with

¹<https://github.com/nitishsrivastava/deepnet>

momentum-based stochastic gradient descent for 5000 iterations. The objective is reported as the classification error rate on the validation set. As above, we treat constraint violations as the worst possible value (in this case a classification error of 1.0).

Figure 3(a) shows the results of 50 iterations of Bayesian optimization. In this experiment and the next, the y -axis represents observed objective values, $\delta_1 = 0.05$, a Matérn $5/2$ GP covariance kernel is used, and GP hyperparameters are integrated out using slice sampling (Neal, 2000) as in Snoek et al. (2012). Curves are the mean over 5 independent experiments. We find that PESC performs significantly better than EIC. However, when the noise level is high, reporting the best objective observation is an overly optimistic metric (due to “lucky” evaluations); on the other hand, ground-truth is not available. Therefore, to validate our results further, we used the recommendations made at the final iteration of Bayesian optimization for each method (EIC and PESC) and evaluated the function with these recommended parameters. We repeated the evaluation 10 times for each of the 5 repeated experiments to compute a ground-truth score averaged of 50 function evaluations. This procedure yields a score of $7.0 \pm 0.6\%$ for PESC and $49 \pm 4\%$ for EIC (as in the figure, constraint violations are treated as a classification error of 100%). This result is consistent with Figure 3(a) in that PESC performs significantly better than EIC, but also demonstrates that, due to noise, Figure 3(a) is overly optimistic. While we may believe this optimism to affect both methods equally, the ground-truth measurement provides a more reliable result and a much clearer understanding of the classification error attained by Bayesian optimization.

4.5. Tuning Markov chain Monte Carlo

Hybrid Monte Carlo, also known as Hamiltonian Monte Carlo (HMC), is a popular Markov Chain Monte Carlo (MCMC) technique that uses gradient information in a numerical integration to select the next sample. However,

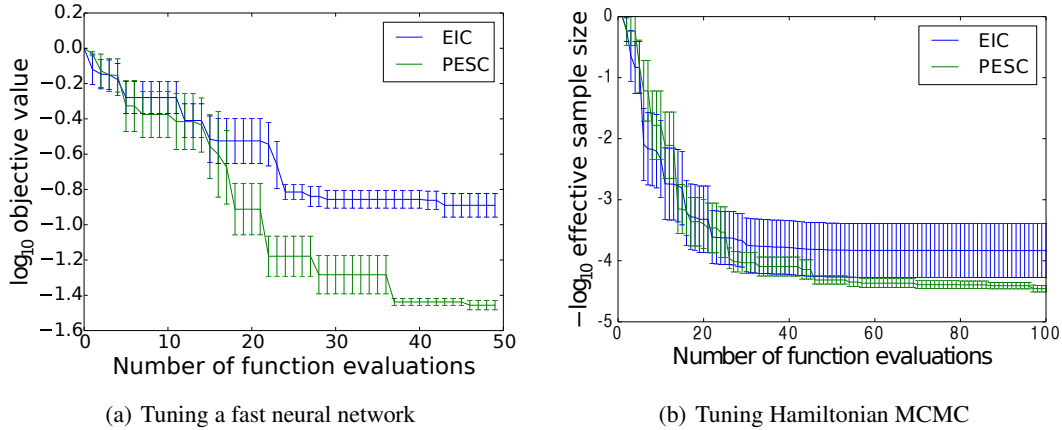


Figure 3. Comparing PESC and EIC for (a) minimizing classification error of a 3-hidden-layer neural network constrained to make predictions in under 2 ms, and (b) tuning Hamiltonian Monte Carlo to maximize the number of effective samples within 5 minutes of compute time.

using numerical integration gives rise to new parameters like the integration step size and the number of integration steps. Following the experimental set up in Gelbart et al. (2014), we optimize the number of effective samples produced by an HMC sampler limited to 5 minutes of computation time, subject to passing of the Geweke (Geweke, 1992) and Gelman-Rubin (Gelman & Rubin, 1992) convergence diagnostics, as well as the constraint that the numerical integration should not diverge. We tune 4 parameters of an HMC sampler: the integration step size, number of integration steps, fraction of the allotted 5 minutes spent in burn-in, and an HMC mass parameter (see Neal, 2011). We use the *coda* R package (Plummer et al., 2006) to compute the effective sample size and the Geweke convergence diagnostic, and the *PyMC* python package (Patil et al., 2010) to compute the Gelman-Rubin diagnostic over two independent traces. Following Gelbart et al. (2014), we impose the constraints that the absolute value of the Geweke test score be at most 2.0 and the Gelman-Rubin score be at most 1.2, and sample from the posterior distribution of a logistic regression problem using the UCI German credit data set (Frank & Asuncion, 2010).

Figure 3(b) evaluates EIC and PESC on this task, averaged over 10 independent experiments. As above, we perform a ground-truth assessment of the final recommendations. The average effective sample size is 3300 ± 1200 for PESC and 2300 ± 900 for EIC. From these results we draw a similar conclusion to that of Figure 3(b); namely, that PESC outperforms EIC but only by a small margin, and furthermore that the experiment is very noisy.

5. Discussion

In this paper, we addressed global optimization with unknown constraints. Motivated by the weaknesses of existing methods, we presented PESC, a method based on the theoretically appealing expected information gain heuristic. We showed that the approximations in PESC are quite accurate, and that PESC performs about equally well to a ground truth method based on rejection sampling. In sections 4.2 to 4.5, we showed that PESC outperforms current methods such as EIC and AL over a variety of problems. Furthermore, PESC is easily applied to problems with decoupled constraints, without additional computational cost or the pathologies discussed in Gelbart et al. (2014).

One disadvantage of PESC is that it is relatively difficult to implement: in particular, the EP approximation often leads to numerical instabilities. Therefore, we have integrated our implementation, which carefully addresses these numerical issues, into the open-source Bayesian optimization package *Spear* at <https://github.com/HIPS/Spear>. We have demonstrated that PESC is a flexible and powerful method and we hope the existence of such a method will bring constrained Bayesian optimization into the standard toolbox of Bayesian optimization practitioners.

Acknowledgements

José Miguel Hernández-Lobato acknowledges support from the Rafael del Pino Foundation. Zoubin Ghahramani acknowledges support from Google Focused Research Award and EPSRC grant EP/I036575/1. Matthew W. Hoffman acknowledges support from EPSRC grant EP/J012300/1.

References

- Frank, Andrew and Asuncion, Arthur. UCI machine learning repository, 2010.
- Gardner, Jacob R., Kusner, Matt J., Xu, Zhixiang (Edie), Weinberger, Kilian Q., and Cunningham, John P. Bayesian optimization with inequality constraints. In *ICML*, 2014.
- Gelbart, Michael A., Snoek, Jasper, and Adams, Ryan P. Bayesian optimization with unknown constraints. In *UAI*, 2014.
- Gelman, Andrew and Rubin, Donald R. A single series from the Gibbs sampler provides a false sense of security. In *Bayesian Statistics*, pp. 625–32. 1992.
- Geweke, John. Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments. In *Bayesian Statistics*, pp. 169–193, 1992.
- Gramacy, Robert B. and Lee, Herbert K. H. Optimization under unknown constraints. *Bayesian Statistics*, 9, 2011.
- Gramacy, Robert B., Gray, Genetha A., Digabel, Sebastien Le, Lee, Herbert K. H., Ranjan, Pritam, Wells, Garth, and Wild, Stefan M. Modeling an augmented Lagrangian for improved blackbox constrained optimization, 2014. arXiv:1403.4890v2 [stat.CO].
- Hennig, Philipp and Schuler, Christian J. Entropy search for information-efficient global optimization. *JMLR*, 13, 2012.
- Hernández-Lobato, J. M, Hoffman, M. W., and Ghahramani, Z. Predictive entropy search for efficient global optimization of black-box functions. In *NIPS*. 2014.
- Houlsby, N., Hernández-Lobato, J. M, Huszar, F., and Ghahramani, Z. Collaborative Gaussian processes for preference learning. In *NIPS*. 2012.
- Jones, Donald R, Schonlau, Matthias, and Welch, William J. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13 (4):455–492, 1998.
- Leobacher, Gunther and Pillichshammer, Friedrich. *Introduction to quasi-Monte Carlo integration and applications*. Springer, 2014.
- Minka, Thomas P. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.
- Mockus, Jonas, Tiesis, Vytautas, and Zilinskas, Antanas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2, 1978.
- Neal, Radford. Slice sampling. *Annals of Statistics*, 31: 705–767, 2000.
- Neal, Radford. MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, 2011.
- Patil, Anand, Huard, David, and Fonnesbeck, Christopher. PyMC: Bayesian stochastic modelling in Python. *Journal of Statistical Software*, 2010.
- Picheny, Victor. A stepwise uncertainty reduction approach to constrained global optimization. In *AISTATS*, 2014.
- Plummer, Martyn, Best, Nicky, Cowles, Kate, and Vines, Karen. CODA: Convergence diagnosis and output analysis for MCMC. *R News*, 6(1):7–11, 2006.
- Rasmussen, C. and Williams, C. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- Schonlau, Matthias, Welch, William J, and Jones, Donald R. Global versus local search in constrained optimization of computer models. *Lecture Notes-Monograph Series*, pp. 11–25, 1998.
- Snoek, Jasper. *Bayesian Optimization and Semiparametric Models with Applications to Assistive Technology*. PhD thesis, University of Toronto, Toronto, Canada, 2013.
- Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. Practical Bayesian optimization of machine learning algorithms. In *NIPS*, 2012.
- Villemonteix, Julien, Vázquez, Emmanuel, and Walter, Eric. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44(4):509–534, 2009.