Predictive Entropy Search for Multi-objective Bayesian Optimization

Daniel Hernández-Lobato DANIEL.HERNANDEZ@UAM.ES
Universidad Autónoma de Madrid, Francisco Tomás y Valiente 11, 28049, Madrid, Spain.

José Miguel Hernández-Lobato JMHL@SEAS.HARVARD.EDU
Harvard University, 33 Oxford street, Cambridge, MA 02138, USA.

Amar Shah AS793@CAM.AC.UK
Cambridge University, Trumpington Street, Cambridge CB2 1PZ, United Kingdom.

Ryan P. Adams RPA@SEAS.HARVARD.EDU
Harvard University and Twitter, 33 Oxford street Cambridge, MA 02138, USA.

Abstract

We present PESMO, a Bayesian method for identifying the Pareto set of multi-objective optimization problems, when the functions are expensive to evaluate. PESMO chooses the evaluation points to maximally reduce the entropy of the posterior distribution over the Pareto set. The PESMO acquisition function is decomposed as a sum of objective-specific acquisition functions, which makes it possible to use the algorithm in *decoupled* scenarios in which the objectives can be evaluated separately and perhaps with different costs. This decoupling capability is useful to identify difficult objectives that require more evaluations. PESMO also offers gains in efficiency, as its cost scales linearly with the number of objectives, in comparison to the exponential cost of other methods. We compare PESMO with other methods on synthetic and real-world problems. The results show that PESMO produces better recommendations with a smaller number of evaluations, and that a decoupled evaluation can lead to improvements in performance, particularly when the number of objectives is large.

1. Introduction

We address the problem of optimizing K real-valued functions $f_1(\mathbf{x}), \ldots, f_K(\mathbf{x})$ over some bounded domain $\mathcal{X} \subset \mathbb{R}^d$, where d is the dimensionality of the input space. This is a more general, challenging and realistic scenario than the one considered in traditional optimization problems where there is a single-objective function. For example, in a complex robotic system, we may be interested in minimizing the energy consumption while maximizing locomotion speed (Ariizumi et al., 2014). When selecting a financial portfolio, it may be desirable to maximize returns while minimizing various risks. In a mechanical design, one may wish to minimize manufacturing cost while maximizing durability. In each of these multi-objective examples, it is unlikely to be possible to optimize all of the objectives simultaneously as they may be conflicting: a fast-moving robot probably consumes more energy, high-return financial instruments typically carry greater risk, and cheaply manufactured goods are often more likely to break. Nevertheless, it is still possible to find a set of optimal points $\mathcal{X}^\star$ known as the *Pareto set* (Collette & Siarry, 2003). Rather than a single best point, this set represents a collection of solutions at which no objective can be improved without damaging one of the others.

In the context of minimization, we say that $\mathbf{x}$ *Pareto dominates* $\mathbf{x}'$ if $f_k(\mathbf{x}) \leq f_k(\mathbf{x}') \; \forall k$, with at least one of the inequalities being strict. The *Pareto set* $\mathcal{X}^\star$ is then the subset of non-dominated points in $\mathcal{X}$, i.e., the set such that $\forall \mathbf{x}^\star \in \mathcal{X}^\star, \; \forall \mathbf{x} \in \mathcal{X}, \; \exists \, k \in 1, \ldots, K$ for which $f_k(\mathbf{x}^\star) < f_k(\mathbf{x})$. The Pareto set is considered to be optimal because for each point in that set one cannot improve in one of the objectives without deteriorating some other objective. Given $\mathcal{X}^\star$, the user may choose a point from this set according to their preferences, *e.g.*, locomotion speed vs. energy consumption. The Pareto set is often not finite, and most strategies aim at finding a finite set with which to approximate $\mathcal{X}^\star$ well.

It frequently happens that there is a high cost to evaluat-

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

ing one or more of the functions $f_k(\cdot)$. For example, in the robotic example, the evaluation process may involve a time consuming experiment with the embodied robot. In this case, one wishes to minimize the number of evaluations required to obtain a useful approximation to the Pareto set $\mathcal{X}^\star$. Furthermore, it is often the case that there is no simple closed form for the objectives $f_k(\cdot)$, *i.e.*, they can be regarded as black boxes. One promising approach in this setting has been to use a probabilistic model such as a Gaussian process to approximate each function (Knowles, 2006; Emmerich, 2008; Ponweiser et al., 2008; Picheny, 2015). At each iteration, these strategies use the uncertainty captured by the probabilistic model to generate an acquisition (utility) function, the maximum of which provides an effective heuristic for identifying a promising location on which to evaluate the objectives. Unlike the actual objectives, the acquisition function is a function of the model and therefore relatively cheap to evaluate and maximize. This approach contrasts with model-free methods based on genetic algorithms or evolutionary strategies that are known to be effective for approximating the Pareto set, but demand a large number of function evaluations (Deb et al., 2002; Li, 2003; Zitzler & Thiele, 1999).

Despite these successes, there are notable limitations to current model-based approaches: 1) they often build the acquisition function by transforming the multi-objective problem into a single-objective problem using scalarization techniques (an approach that is expected to be suboptimal), 2) the acquisition function generally requires the evaluation of *all* of the objective functions at the same location in each iteration, and 3) the computational cost of evaluating the acquisition function typically grows exponentially with the number of objectives, which limits their applicability to optimization problems with just 2 or 3 objectives.

We describe here a strategy for multi-objective optimization that addresses these concerns. We extend previous single-objective strategies based on stepwise uncertainty reduction to the multi-objective case (Villemonteix et al., 2009; Hernández-Lobato et al., 2014; Henning & Schuler, 2012). In the single-objective case, these strategies choose the next evaluation location based on the reduction of the Shannon entropy of the posterior estimate of the minimizer $\mathbf{x}^\star$. The idea is that a smaller entropy implies that the minimizer $\mathbf{x}^\star$ is better identified; the heuristic then chooses candidate evaluations based on how much they are expected to improve the quality of this estimate. These information gain criteria have been shown to often provide better results than other alternatives based, *e.g.*, on the popular *expected improvement* (Hernández-Lobato et al., 2014; Henning & Schuler, 2012; Shah & Ghahramani, 2015).

The extension to the multi-objective case is obtained by considering the entropy of the posterior distribution over the Pareto set $\mathcal{X}^\star$. More precisely, we choose the next evaluation as the one that is expected to most reduce the entropy of our estimate of $\mathcal{X}^\star$. The proposed approach is called *predictive entropy search for multi-objective optimization* (PESMO). Several experiments involving real-world and synthetic optimization problems, show that PESMO can lead to better performance than related methods from the literature. Furthermore, in PESMO the acquisition function is expressed as a sum across the different objectives, allowing for *decoupled* scenarios in which we can choose to only evaluate a subset of objectives at any given location. In the robotics example, one might be able to decouple the problems by estimating energy consumption from a simulator even if the locomotion speed could only be evaluated via physical experimentation. Another example, inspired by Gelbart et al. (2014), might be the design of a low-calorie cookie: one wishes to maximize taste while minimizing calories, but calories are a simple function of the ingredients, while taste could require human trials. The results obtained show that PESMO can obtain better results with a smaller number of evaluations of the objective functions in such scenarios. Furthermore, we have observed that the decoupled evaluation provides significant improvements over a coupled evaluation when the number of objectives is large. Finally, unlike other methods (Ponweiser et al., 2008; Picheny, 2015), the computational cost of PESMO grows linearly with the number of objectives.

2. Multi-objective Bayesian Optimization via Predictive Entropy Search

In this section we describe the proposed approach for multi-objective optimization based on *predictive entropy search*. Given some previous evaluations of each objective function $f_k(\cdot)$, we seek to choose new evaluations that maximize the information gained about the Pareto set $\mathcal{X}^\star$. This approach requires a probabilistic model for the unknown objectives, and we therefore assume that each $f_k(\cdot)$ follows a Gaussian process (GP) prior (Rasmussen & Williams, 2006), with observation noise that is i.i.d. Gaussian with zero mean. GPs are often used in model-based approaches to multi-objective optimization because of their flexibility and ability to model uncertainty (Knowles, 2006; Emmerich, 2008; Ponweiser et al., 2008; Picheny, 2015). For simplicity, we initially consider a coupled setting in which we evaluate all objectives at the same location in any given iteration. Nevertheless, the approach described can be easily extended to the *decoupled* scenario.

Let $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$ be the data (function evaluations) collected up to step N, where $\mathbf{y}_n$ is a K-dimensional vector with the values resulting from the evaluation of all objectives at step n, and $\mathbf{x}_n$ is a vector in input space denoting the evaluation location. The next query $\mathbf{x}_{N+1}$ is the one

that maximizes the expected reduction in the entropy $H(\cdot)$ of the posterior distribution over the Pareto set $\mathcal{X}^\star$, i.e., $p(\mathcal{X}^\star|\mathcal{D})$. The acquisition function of PESMO is hence:

$$\alpha(\mathbf{x}) = H(\mathcal{X}^\star|\mathcal{D}) - \mathbb{E}_{\mathbf{y}}\left[H(\mathcal{X}^\star|\mathcal{D}\cup\{(\mathbf{x},\mathbf{y})\})\right], \quad (1)$$

where $\mathbf{y}$ is the output of all the GP models at $\mathbf{x}$ and the expectation is taken with respect to the posterior distribution for $\mathbf{y}$ given by these models, $p(\mathbf{y}|\mathcal{D},\mathbf{x}) = \prod_{k=1}^{K} p(y_k|\mathcal{D},\mathbf{x})$. The GPs are assumed to be independent *a priori*. This acquisition function is known as *entropy search* (Villemonteix et al., 2009; Henning & Schuler, 2012). Thus, at each iteration we set the location of the next evaluation to $\mathbf{x}_{N+1} = \arg\max_{\mathbf{x}\in\mathcal{X}} \alpha(\mathbf{x})$.

A practical difficulty, however, is that the exact evaluation of Eq. (1) is generally infeasible and the function must be approximated; we follow the approach described in (Hernández-Lobato et al., 2014; Houlsby et al., 2012). In particular, Eq. (1) is the mutual information between $\mathcal{X}^\star$ and $\mathbf{y}$ given $\mathcal{D}$. The mutual information is symmetric and hence we can exchange the roles of the variables $\mathcal{X}^\star$ and $\mathbf{y}$, leading to an expression that is equivalent to Eq. (1):

$$\alpha(\mathbf{x}) = H(\mathbf{y}|\mathcal{D},\mathbf{x}) - \mathbb{E}_{\mathcal{X}^\star}\left[H(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{X}^\star)\right], \quad (2)$$

where the expectation is now with respect to the posterior distribution for the Pareto set $\mathcal{X}^\star$ given the observed data, and $H(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{X}^\star)$ measures the entropy of $p(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{X}^\star)$, *i.e.*, the predictive distribution for the objectives at $\mathbf{x}$ given $\mathcal{D}$ and conditioned to $\mathcal{X}^\star$ being the Pareto set of the objective functions. This alternative formulation is known as *predictive entropy search* (Hernández-Lobato et al., 2014) and it significantly simplifies the evaluation of the acquisition function $\alpha(\cdot)$. In particular, we no longer have to evaluate or approximate the entropy of the Pareto set, $\mathcal{X}^\star$, which may be quite difficult. The new acquisition function obtained in Eq. (2) favors the evaluation in the regions of the input space for which $\mathcal{X}^\star$ is more informative about $\mathbf{y}$. These are precisely also the regions in which $\mathbf{y}$ is more informative about $\mathcal{X}^\star$.

The first term in the r.h.s. of Eq. (2) is straight-forward to evaluate; it is simply the entropy of the predictive distribution $p(\mathbf{y}|\mathcal{D},\mathbf{x})$, which is a factorizable K-dimensional Gaussian distribution. Thus, we have that

$$H(\mathbf{y}|\mathcal{D},\mathbf{x}) = \frac{K}{2}\log(2\pi e) + \sum_{k=1}^{K} 0.5\log(v_k^{\text{PD}}), \quad (3)$$

where v_k^{PD} is the predictive variance of $f_k(\cdot)$ at $\mathbf{x}$. The difficulty comes from the evaluation of the second term in the r.h.s. of Eq. (2), which is intractable and must be approximated; we follow Hernández-Lobato et al. (2014) and approximate the expectation using a Monte Carlo estimate of the Pareto set, $\mathcal{X}^\star$ given $\mathcal{D}$. This involves sampling several times the objective functions from their posterior distribution $p(f_1,\ldots,f_K|\mathcal{D})$. This step is done as in Hernández-Lobato et al. (2014) using random kernel features and linear models that accurately approximate the samples from $p(f_1,\ldots,f_K|\mathcal{D})$. In practice, we generate 10 samples from the posterior of each objective $f_k(\cdot)$.

Given the samples of the objectives, we must optimize them to obtain a sample from the Pareto set $\mathcal{X}^\star$. Note that unlike the true objectives, the sampled functions can be evaluated without significant cost. Thus, given these functions, we use a grid search with $d \times 1,000$ points to solve the corresponding multi-objective problem to find $\mathcal{X}^\star$, where d is the number of dimensions. Of course, in high dimensional problems such a grid search is expected to be sub-optimal; in that case, we use the NSGA-II evolutionary algorithm (Deb et al., 2002). The Pareto set is then approximated using a representative subset of 50 points. Given such a sample of $\mathcal{X}^\star$, the differential entropy of $p(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{X}^\star)$ is estimated using the expectation propagation algorithm (Minka, 2001), as described in the proceeding section.

2.1. Approximating the Conditional Predictive Distribution Using Expectation Propagation

To approximate the entropy of the conditional predictive distribution $p(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{X}^\star)$ we consider the distribution $p(\mathcal{X}^\star|f_1,\ldots,f_K)$. In particular, $\mathcal{X}^\star$ is the Pareto set of $f_1,\ldots,f_K$ iff $\forall\mathbf{x}^\star\in\mathcal{X}^\star, \forall\mathbf{x}'\in\mathcal{X}, \exists\, k\in 1,\ldots,K$ such that $f_k(\mathbf{x}^\star) \leq f_k(\mathbf{x}')$, assuming minimization. That is, each point within the Pareto set has to be better or equal to any other point in the domain of the functions in at least one of the objectives. Let $\mathbf{f}$ be the set $\{f_1,\ldots,f_K\}$. Informally, the conditions just described can be translated into the following un-normalized distribution for $\mathcal{X}^\star$:

$$\begin{aligned} p(\mathcal{X}^\star|\mathbf{f}) &\propto \prod_{\mathbf{x}^\star\in\mathcal{X}^\star}\prod_{\mathbf{x}'\in\mathcal{X}}\left[1 - \prod_{k=1}^{K}\Theta\left(f_k(\mathbf{x}') - f_k(\mathbf{x}^\star)\right)\right] \\ &= \prod_{\mathbf{x}^\star\in\mathcal{X}^\star}\prod_{\mathbf{x}'\in\mathcal{X}}\psi(\mathbf{x}',\mathbf{x}^\star), \end{aligned} \quad (4)$$

where $\psi(\mathbf{x}',\mathbf{x}^\star) = 1 - \prod_{k=1}^{K}\Theta\left(f_k(\mathbf{x}') - f_k(\mathbf{x}^\star)\right)$, $\Theta(\cdot)$ is the Heaviside step function, and we have used the convention that $\Theta(0) = 1$. Thus, the r.h.s. of Eq. (4) is non-zero only for a valid Pareto set. Next, we note that in the noiseless case $p(\mathbf{y}|\mathbf{x},\mathbf{f}) = \prod_{k=1}^{K}\delta(y_k - f_k(\mathbf{x}))$, where $\delta(\cdot)$ is the Dirac delta function; in the noisy case we simply replace the delta functions with Gaussians. We can hence write the unnormalized version of $p(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{X}^\star)$ as:

$$\begin{aligned} p(\mathbf{y}|\mathcal{D},\mathbf{x},\mathcal{X}^\star) &\propto \int p(\mathbf{y}|\mathbf{x},\mathbf{f})p(\mathcal{X}^\star|\mathbf{f})p(\mathbf{f}|\mathcal{D})d\mathbf{f} \\ &\propto \int\prod_{k=1}^{K}\delta(y_k - f_k(\mathbf{x}))\prod_{\mathbf{x}^\star\in\mathcal{X}^\star}\psi(\mathbf{x},\mathbf{x}^\star) \\ &\quad\times\prod_{\mathbf{x}'\in\mathcal{X}\setminus\{\mathbf{x}\}}\psi(\mathbf{x}',\mathbf{x}^\star)\,p(\mathbf{f}|\mathcal{D})\,d\mathbf{f}, \end{aligned} \quad (5)$$

where we have separated out the factors ψ that do not depend on $\mathbf{x}$, the point in which the acquisition function $\alpha(\cdot)$ is going to be evaluated. The approximation to the r.h.s. of Eq. (5) is obtained in two stages. First, we approximate $\mathcal{X}$ with the set $\tilde{\mathcal{X}} = \{\mathbf{x}_n\}_{n=1}^N \cup \mathcal{X}^\star \cup \{\mathbf{x}\}$, *i.e.*, the union of the input locations where the objective functions have been already evaluated, the current Pareto set and the candidate location $\mathbf{x}$ on which $\alpha(\cdot)$ should be evaluated. Then, we replace each non-Gaussian factor ψ with a corresponding approximate Gaussian factor $\tilde{\psi}$ whose parameters are found using expectation propagation (EP) (Minka, 2001). That is,

$$\psi(\mathbf{x}', \mathbf{x}^\star) = 1 - \prod_{k=1}^{K} \Theta\left(f_k(\mathbf{x}') - f_k(\mathbf{x}^\star)\right)$$

$$\approx \tilde{\psi}(\mathbf{x}', \mathbf{x}^\star) = \prod_{k=1}^{K} \tilde{\phi}_k(f_k(\mathbf{x}'), f_k(\mathbf{x}^\star)) \,, \quad (6)$$

where each approximate factor $\tilde{\phi}_k$ is an unnormalized two-dimensional Gaussian distribution. In particular, we set $\tilde{\phi}_k(f_k(\mathbf{x}'), f_k(\mathbf{x}^\star)) = \exp\left\{-\frac{1}{2}\boldsymbol{\upsilon}_k^{\mathrm{T}} \tilde{\mathbf{V}}_k \boldsymbol{\upsilon}_k + \tilde{\mathbf{m}}_k^{\mathrm{T}} \boldsymbol{\upsilon}_k\right\}$, where we have defined $\boldsymbol{\upsilon}_k = (f_k(\mathbf{x}'), f_k(\mathbf{x}^\star))^{\mathrm{T}}$, and $\tilde{\mathbf{V}}_k$ and $\tilde{\mathbf{m}}_k$ are parameters to be adjusted by EP, which refines each $\tilde{\psi}$ until convergence to enforce that it looks similar to the corresponding exact factor ψ (Minka, 2001). The approximate factors $\tilde{\psi}$ that do not depend on the candidate input $\mathbf{x}$ are reused multiple times to evaluate the acquisition function $\alpha(\cdot)$, and they only have to be computed once. The $|\mathcal{X}^\star|$ factors that depend on $\mathbf{x}$ must be obtained relatively quickly to guarantee that $\alpha(\cdot)$ is not very expensive to evaluate. Thus, in practice we only update those factors once using EP, i.e., they are not refined until convergence.

Once EP has been run, we approximate $p(\mathbf{y}|\mathcal{D}, \mathbf{x}, \mathcal{X}^\star)$ by the normalized Gaussian that results from replacing each exact factor ψ by the corresponding approximate $\tilde{\psi}$. Note that the Gaussian distribution is closed under the product operation, and because all non-Gaussian factors in Eq. (5) have been replaced by Gaussians, the result is a Gaussian distribution. That is, $p(\mathbf{y}|\mathcal{D}, \mathbf{x}, \mathcal{X}^\star) \approx \prod_{k=1}^K \mathcal{N}(f_k(\mathbf{x})|m_k^{\mathrm{CPD}}, v_k^{\mathrm{CPD}})$, where the parameters m_k^{CPD} and v_k^{CPD} can be obtained from each $\tilde{\psi}$ and $p(f_1, \ldots, f_K|\mathcal{D})$. If we combine this result with Eq. (3), we obtain an approximation to the acquisition function in Eq. (2) that is given by the difference in entropies before and after conditioning on the Pareto sets. That is,

$$\alpha(\mathbf{x}) \approx \sum_{k=1}^{K} \frac{\log v_k^{\mathrm{PD}}(\mathbf{x})}{2} - \frac{1}{S}\sum_{s=1}^{S} \frac{\log v_k^{\mathrm{CPD}}(\mathbf{x}|\mathcal{X}_{(s)}^\star)}{2} \,, \quad (7)$$

where S is the number of Monte Carlo samples, $\{\mathcal{X}_{(s)}^\star\}_{s=1}^S$ are the Pareto sets sampled to approximate the expectation in Eq. (2), and $v_k^{\mathrm{PD}}(\mathbf{x})$ and $v_k^{\mathrm{CPD}}(\mathbf{x}|\mathcal{X}_{(s)}^\star)$ are respectively the variances of the predictive distribution at $\mathbf{x}$, before and after conditioning to $\mathcal{X}_{(s)}^\star$. Last, in the case of noisy observations around each $f_k(\cdot)$, we just increase the predictive variances by adding the noise variance. The next evaluation is simply set to $\mathbf{x}_{N+1} = \arg\max_{\mathbf{x}\in\mathcal{X}} \alpha(\mathbf{x})$.

Note that Eq. (7) is the sum of K functions

$$\alpha_k(\mathbf{x}) = \frac{\log v_k^{\mathrm{PD}}(\mathbf{x})}{2} - \frac{1}{S}\sum_{s=1}^{S} \frac{\log v_k^{\mathrm{CPD}}(\mathbf{x}|\mathcal{X}_{(s)}^\star)}{2} \,, \quad (8)$$

that intuitively measure the contribution of each objective to the total acquisition. In a decoupled evaluation setting, each $\alpha_k(\cdot)$ can be individually maximized to identify the location $\mathbf{x}_k^{\mathrm{op}} = \arg\max_{\mathbf{x}\in\mathcal{X}} \alpha_k(\mathbf{x})$, on which it is expected to be most useful to evaluate each of the K objectives. The objective k with the largest individual acquisition $\alpha_k(\mathbf{x}_k^{\mathrm{op}})$ can then be chosen for evaluation in the next iteration. This approach is expected to reduce the entropy of the posterior over the Pareto set more quickly, *i.e.*, with a smaller number of evaluations of the objectives, and to lead to better results.

The total computational cost of evaluating the acquisition function $\alpha(\mathbf{x})$ includes the cost of running EP, which is $\mathcal{O}(Km^3)$, where $m = N + |\mathcal{X}_{(s)}^\star|$, N is the number of observations made and K is the number of objectives. This is done once per each sample $\mathcal{X}_{(s)}^\star$. After this, we can reuse the factors that are independent of the candidate location $\mathbf{x}$. The cost of computing the predictive variance at each $\mathbf{x}$ is hence $\mathcal{O}(K|\mathcal{X}_{(s)}^\star|^3)$. In our experiments, the size of the Pareto set sample $\mathcal{X}_{(s)}^\star$ is 50, which means that m is a few hundred at most. The supplementary material contains additional details about the EP approximation to Eq. (5).

3. Related Work

ParEGO is another method for multi-objective Bayesian optimization (Knowles, 2006). ParEGO transforms the multi-objective problem into a single-objective problem using a scalarization technique: at each iteration, a vector of K weights $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)^{\mathrm{T}}$, with $\theta_k \in [0, 1]$ and $\sum_{k=1}^K \theta_k = 1$, is sampled at random from a uniform distribution. Given $\boldsymbol{\theta}$, a single-objective function is built:

$$f_{\boldsymbol{\theta}}(\mathbf{x}) = \max_{k=1}^K(\theta_k f_k(\mathbf{x})) + \rho \sum_{k=1}^{K} \theta_k f_k(\mathbf{x}) \quad (9)$$

where ρ is set equal to 0.05. See (Nakayama et al., 2009, Sec. 1.3.3) for further details. After step N of the optimization process, and given $\boldsymbol{\theta}$, a new set of N observations of $f_{\boldsymbol{\theta}}(\cdot)$ are obtained by evaluating this function in the already observed points $\{\mathbf{x}_n\}_{n=1}^N$. Then, a GP model is fit to the new data and expected improvement (Mockus et al., 1978; Jones et al., 1998) is used find the location of the next evaluation $\mathbf{x}_{N+1}$. The cost of evaluating the acquisition function in ParEGO is $\mathcal{O}(N^3)$, where N is the number of observations made. This is the cost of fitting the GP to the new data (only done once). Thus, ParEGO is a simple and fast technique. Nevertheless, it is often outperformed by more advanced approaches (Ponweiser et al., 2008).

SMSego is another technique for multi-objective Bayesian optimization (Ponweiser et al., 2008). The first step in SMSego is to find a set of Pareto points $\tilde{\mathcal{X}}^\star$, *e.g.*, by optimizing the posterior means of the GPs, or by finding the non-dominated observations. Consider now an optimistic estimate of the objectives at input location $\mathbf{x}$ given by $m_k^{\text{PD}}(\mathbf{x}) - c \cdot v_k^{\text{PD}}(\mathbf{x})^{1/2}$, where c is some constant, and $m_k^{\text{PD}}(\mathbf{x})$ and $v_k^{\text{PD}}(\mathbf{x})$ are the posterior mean and variance of the kth objective at location $\mathbf{x}$, respectively. The acquisition value computed at a candidate location $\mathbf{x} \in \mathcal{X}$ by SMSego is given by the gain in hyper-volume obtained by the corresponding optimistic estimate, after an ϵ-correction has been made. The hyper-volume is simply the volume of points in functional space above the Pareto front (this is simply the function space values associated to the Pareto set), with respect to a given reference point (Zitzler & Thiele, 1999). Because the hyper-volume is maximized by the actual Pareto set, it is a natural measure of performance. Thus, SMSego does not reduce the problem to a single-objective. However, at each iteration it has to find a set of Pareto points and to fit a different GP to each one of the objectives. This gives a computational cost that is $\mathcal{O}(KN^3)$. Finally, evaluating the gain in hyper-volume at each candidate location $\mathbf{x}$ is also more expensive than the computation of expected improvement in ParEGO.

A similar method to SMSego is the Pareto active learning (PAL) algorithm (Zuluaga et al., 2013). At iteration N, PAL uses the GP prediction for each point $\mathbf{x} \in \mathcal{X}$ to maintain an uncertainty region $\mathcal{R}_N(\mathbf{x})$ about the objective values associated with $\mathbf{x}$. This region is defined as the intersection of $\mathcal{R}_{N-1}(\mathbf{x})$, *i.e.*, the uncertainty region in the previous iteration, and $\mathcal{Q}_c(\mathbf{x})$, defined as as the hyper-rectangle with lower-corner given by $m_k^{\text{PD}}(\mathbf{x}) - c \cdot v_k^{\text{PD}}(\mathbf{x})^{0.5}$, for $k = 1, \ldots, K$, and upper-corner given by $m_k^{\text{PD}}(\mathbf{x}) + c \cdot v_k^{\text{PD}}(\mathbf{x})^{1/2}$, for $k = 1, \ldots, K$, for some constant c. Given these regions, PAL classifies each point $\mathbf{x} \in \mathcal{X}$ as Pareto-optimal, non-Pareto-optimal or uncertain. A point is classified as Pareto-optimal if the worst value in $\mathcal{R}_N(\mathbf{x})$ is not dominated by the best value in $\mathcal{R}_N(\mathbf{x}')$, for any other $\mathbf{x}' \in \mathcal{X}$, with an ϵ tolerance. A point is classified as non-Pareto-optimal if the best value in $\mathcal{R}_N(\mathbf{x})$ is dominated by the worst value in $\mathcal{R}_N(\mathbf{x}')$ for any other $\mathbf{x}' \in \mathcal{X}$, with an ϵ tolerance. All other points remain uncertain. After the classification, PAL chooses the uncertain point $\mathbf{x}$ with the largest uncertainty region $\mathcal{R}_N(\mathbf{x})$. The total computational cost of PAL is hence similar to that of SMSego.

The expected hyper-volume improvement (EHI) (Emmerich, 2008) is a natural extension of expected improvement to the multi-objective setting (Mockus et al., 1978; Jones et al., 1998). Given the predictive distribution of the GPs at a candidate input location $\mathbf{x}$, the acquisition is the expected increment of the hyper-volume of a candidate Pareto set $\tilde{\mathcal{X}}^\star$. Thus, EHI also needs to find a Pareto set $\tilde{\mathcal{X}}^\star$. This set can be obtained as in SMSego. A difficulty is, however, that computing the expected increment of the hyper-volume is very expensive. For this, the output space is divided in a series of cells, and the probability of improvement is simply obtained as the probability that the observation made at $\mathbf{x}$ lies in a non-dominated cell. This involves a sum across all non-dominated cells, whose number grows exponentially with the number of objectives K. In particular, the total number of cells is $(|\tilde{\mathcal{X}}^\star|+1)^K$. Thus, although some methods have been suggested to speed-up its calculation, *e.g.*, (Hupkens et al., 2014; Feliot et al., 2015), EHI is only feasible for 2 or 3 objectives at most.

Sequential uncertainty reduction (SUR) is another method proposed for multi-objective Bayesian optimization (Picheny, 2015). The working principle of SUR is similar to that of EHI. However, SUR considers the probability of improving the hyper-volume in the whole domain of the objectives $\mathcal{X}$. Thus, SUR also needs to find a set of Pareto points $\tilde{\mathcal{X}}^\star$. These can be obtained as in SMSego. The acquisition computed by SUR is simply the expected decrease in the area under the probability of improving the hyper-volume, after evaluating the objectives at a new candidate location $\mathbf{x}$. The SUR acquisition is computed also by dividing the output space in a total of $(|\tilde{\mathcal{X}}^\star| + 1)^K$ cells, and the area under the probability of improvement is obtained using a Sobol sequence as the integration points. Although some grouping of the cells has been suggested (Picheny, 2015), SUR is an extremely expensive criterion that is only feasible for 2 or 3 objectives at most.

The proposed approach, PESMO, differs from the methods described in this section in that 1) it does not transform the multi-objective problem into a single-objective, 2) the acquisition function of PESMO can be decomposed as the sum of K individual acquisition functions, and this allows for decoupled evaluations, and 3) the computational cost of PESMO is linear in the total number of objectives K.

4. Experiments

We compare PESMO with the other strategies described in Section 3: ParEGO, SMSego, EHI and SUR. We do not compare results with PAL because it is expected to give similar results to those of SMSego, as both methods are based on a lower confidence bound. We have coded all these methods in the software for Bayesian optimization Spearmint (`https://github.com/HIPS/Spearmint`). We use a Matérn covariance function for the GPs and all hyper-parameters (noise, length-scales and amplitude) are approximately sampled from their posterior distribution (we generate 10 samples from this distribution). The acquisition function of each method is averaged over these samples. In ParEGO we consider a differ-

ent scalarization (*i.e.*, a different value of $\boldsymbol{\theta}$) for each sample of the hyper-parameters. In SMSego, EHI and SUR, for each hyper-parameter sample we consider a different Pareto set $\tilde{\mathcal{X}}^\star$, obtained by optimizing the posterior means of the GPs. The resulting Pareto set is extended by including all non-dominated observations. At iteration N, each method gives a recommendation in the form of a Pareto set obtained by optimizing the posterior means of the GPs. The acquisition function of each method is maximized using L-BFGS (a grid of size $1,000$ is used to find a good starting point). The gradients of the acquisition function are approximated by differences (except in ParEGO).

4.1. Accuracy of the PESMO Approximation

One question is whether the proposed approximations are sufficiently accurate for the effective identification of the Pareto set. We compare in a one-dimensional problem with 2 objectives the acquisition function computed by PESMO with a more accurate estimate obtained via expensive Monte Carlo sampling and a non-parametric estimator of the entropy (Singh et al., 2003). Figure 1 (top) shows at a given step the observed data and the posterior mean and the standard deviation of each objective. The figure on the bottom shows the acquisition function computed by PESMO and by the Monte Carlo method (Exact). Both functions look very similar, including the location of the global maximizer. This indicates that (7), obtained by expectation propagation, is potentially a good approximation of (2), the exact acquisition. The supplementary material has extra results showing that the individual acquisition functions computed by PESMO, *i.e.*, $\alpha_k(\cdot)$, for $k = 1, 2$, are also accurate.

4.2. Experiments with Synthetic Objectives

We compare PESMO with other approaches in a 3-dimensional problem with 2 objectives obtained by sampling the functions from the GP prior. We generate 100 of these problems and report the average performance when considering noiseless observations and when the observations are contaminated with Gaussian noise with standard deviation equal to 0.1. The performance metric is the hyper-volume indicator, which is maximized by the actual Pareto set (Zitzler & Thiele, 1999). At each iteration we report the logarithm of the relative difference between the hyper-volume of the actual Pareto set (obtained by optimizing the actual objectives) and the hyper-volume of the recommendation (obtained by optimizing the posterior means of the GPs). Figure 2 (left-column) shows, as a function of the evaluations made, the average performance of each method with error bars. PESMO obtains the best results, and when executed in a decoupled scenario, slight improvements are observed (only with noisy observations).

Table 1 shows the average time in seconds to determine the next evaluation in each method. The fastest method

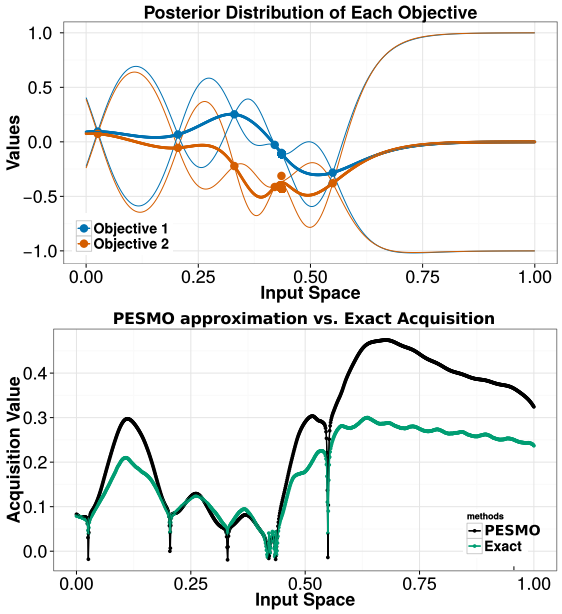

Figure 1. (top) Observations of each objective and posterior mean and standard deviations of each GP model. (bottom) Estimates of the acquisition function (2) by PESMO, and by a Monte Carlo method combined with a non-parametric estimator of the entropy (Exact), which is expected to be more accurate. Best seen in color.

is ParEGO followed by SMSego and PESMO. The decoupled version of PESMO, $\text{PESMO}_{\text{dec}}$, takes more time because it optimizes $\alpha_1(\cdot)$ and $\alpha_2(\cdot)$. The slowest methods are EHI and SUR. Most of their cost is in the last iterations, in which the Pareto set size, $|\tilde{\mathcal{X}}^\star|$, is large. The cost of evaluating the acquisition function in EHI and SUR is $\mathcal{O}((|\tilde{\mathcal{X}}^\star| + 1)^K)$, leading to expensive optimization via L-BFGS. In PESMO the cost of evaluating $\alpha(\cdot)$ is $\mathcal{O}(K|\mathcal{X}^\star_{(s)}|^3)$ because K linear systems are solved. These computations are faster because they are performed by the open-BLAS library, which is optimized for each processor. The acquisition function of EHI and SUR does not involve solving linear systems and these methods cannot use open-BLAS. Note that we also keep fixed $|\mathcal{X}^\star_{(s)}| = 50$ in PESMO.

Table 1. Avg. time in seconds doing calculations per iteration.

PESMO	$\text{PESMO}_{\text{dec}}$	ParEGO	SMSego	EHI	SUR
33±1.0	52±2.5	11±0.2	16±1.3	405±115	623±59

We have carried out additional synthetic experiments with 4 objectives on a 6-dimensional input space. In this case, EHI and SUR become infeasible, so we do not compare results with them. Again, we sample the objectives from the GP prior. Figure 2 (right-column) shows, as a function of the evaluations made, the average performance of each method. The best method is PESMO, and in this case, the decoupled evaluation performs significantly better. This improvement is because in the decoupled setting, PESMO identifies the most difficult objectives and evaluates them more times. In particular, because there are 4 objectives it is likely that some objectives are more difficult than oth-

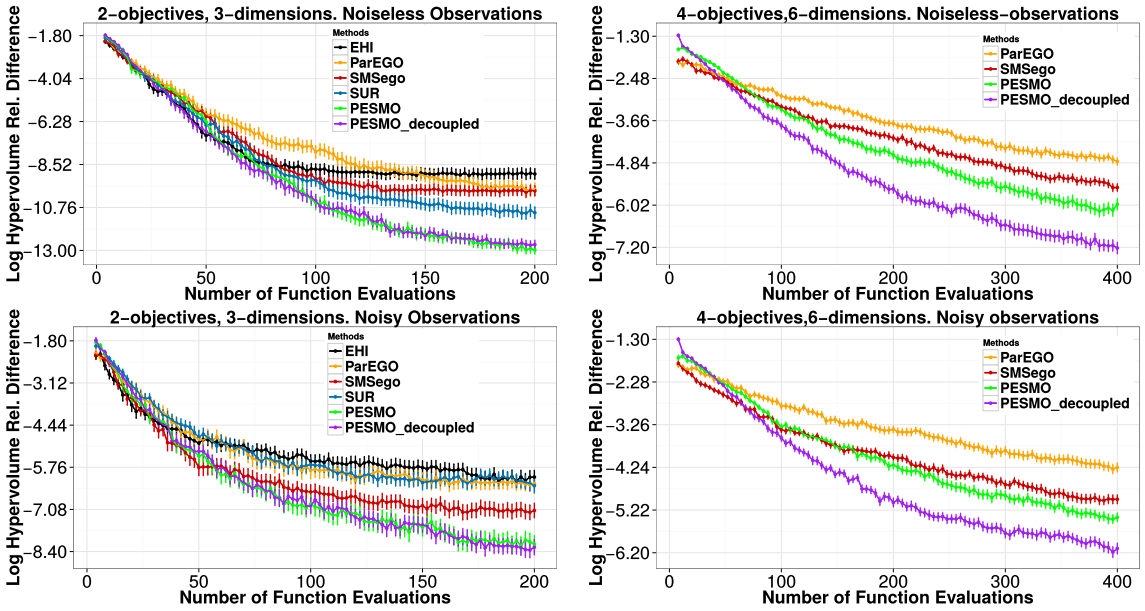

Figure 2. (left-column) Average log relative difference between the hyper-volume of the recommendation and the maximum hyper-volume for each number of evaluations made. We consider noiseless (top) and noisy observations (bottom). The problem considered has 2 objectives and 3 dimensions. (right-column) Similar results for a problem with 4 objectives and 6 dimensions. We do not compare results with EHI and SUR because they are infeasible due to their exponential cost with the number of objectives. Best seen in color.

ers just by chance. Figure 3 illustrates this behavior for a representative case in which the first two objectives are non-linear (difficult) and the last two objectives are linear (easy). We note that the decoupled version of PESMO evaluates the first two objectives almost three times more.

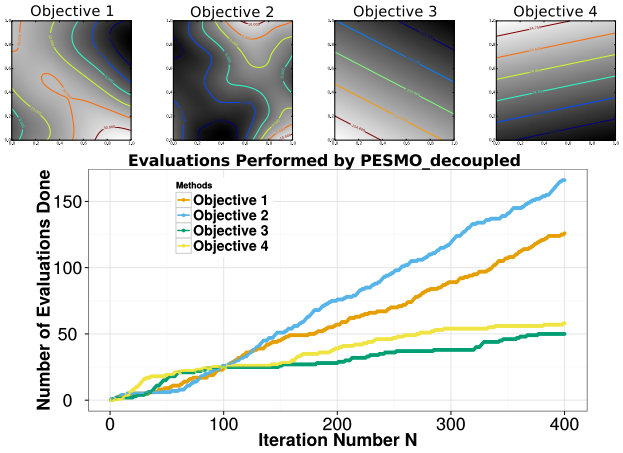

Figure 3. (top) Contour curves of 4 illustrative objectives on 6 dimensions obtained by changing the first two dimensions in input space while keeping the other 4 fixed to zero. The first 2 objectives are non-linear while the 2 last objectives are linear. (bottom) Number of evaluations of each objective done by $\text{PESMO}_{\text{decoupled}}$ as a function of the iterations performed N. Best seen in color.

4.3. Finding a Fast and Accurate Neural Network

We consider the MNIST dataset (LeCun et al., 1998) and evaluate each method on the task of finding a neural network with low prediction error and small prediction time. These are conflicting objectives because reducing the prediction error will involve larger networks which will take longer at test time. We consider feed-forward networks with ReLus at the hidden layers and a soft-max output layer. The networks are coded in the Keras library and they are trained using Adam (D. Kingma, 2014) with a mini-batch size of $4,000$ instances during 150 epochs. The adjustable parameters are: The number of hidden units per layer (between 50 and 300), the number of layers (between 1 and 3), the learning rate, the amount of dropout, and the level of ℓ_1 and ℓ_2 regularization. The prediction error is measured on a set of $10,000$ instances extracted from the training set. The rest of the training data, *i.e.*, $50,000$ instances, is used for training. We consider a logit transformation of the prediction error because the error rates are very small. The prediction time is measured as the average time required for doing $10,000$ predictions. We compute the logarithm of the ratio between the prediction time of the network and the prediction time of the fastest network, (*i.e.*, a single hidden layer and 50 units). When measuring the prediction time we do not train the network and consider random weights (the time objective is also set to ignore irrelevant parameters). Thus, the problem is suited for a decoupled evaluation because both objectives can be evaluated separately. We run each method for a total of 200 evaluations of the objectives and report results after 100 and 200 evaluations. Because there is no ground truth and the objectives are noisy, we re-evaluate 3 times the values associated with the recommendations made by each method (in the form of a Pareto set) and average the results.

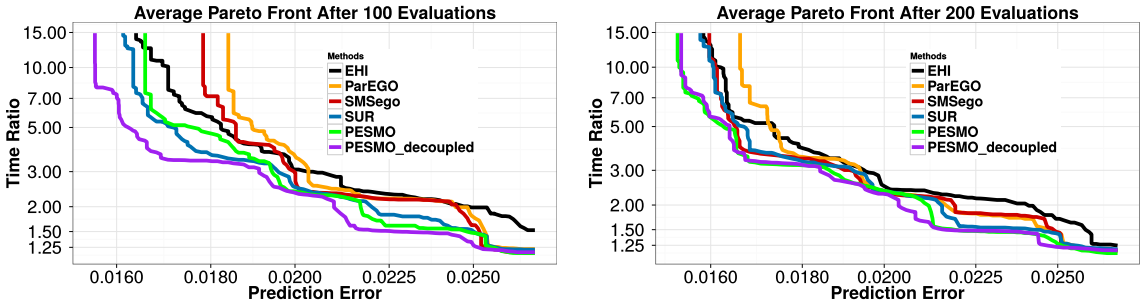

Figure 4. Avg. Pareto fronts obtained by each method after 100 (left) and 200 (right) evaluations of the objectives. Best seen in color.

Then, we compute the Pareto front (*i.e.*, the function space values of the Pareto set) and its hyper-volume. We repeat these experiments 50 times and report the average results.

Table 2 shows the hyper-volumes obtained in the experiments (the higher, the better). The best results, after 100 evaluations of the objectives, correspond to the decoupled version of PESMO, followed by SUR and by the coupled version. When 200 evaluations are done, the best method is PESMO in either setting, *i.e.*, coupled or decoupled. After PESMO, SUR gives the best results, followed by SMSego and EHI. ParEGO is the worst performing method in either setting. In summary, PESMO gives the best overall results, and its decoupled version performs much better than the other methods when the number of evaluations is small.

Table 2. Avg. hyper-volume after 100 and 200 evaluations.

# Eval.	PESMO	$\text{PESMO}_{\text{dec}}$	ParEGO	SMSego	EHI	SUR
100	66.2±.2	**67.6±.1**	62.9±1.2	65.0±.3	64.0±.9	66.6±.2
200	**67.8±.1**	**67.8±.1**	66.1±.2	67.1±.2	66.6±.2	67.2±.1

Figure 4 shows the average Pareto front obtained by each method after 100 and 200 evaluations of the objectives. The results displayed are consistent with the ones in Table 2. In particular, PESMO is able to find networks that are faster than the ones found by the other methods, for a similar prediction error on the validation set. This is especially the case of PESMO when executed in a decoupled setting, after doing only 100 evaluations of the objectives. We also note that PESMO finds the most accurate networks, with almost 1.5% of prediction error in the validation set.

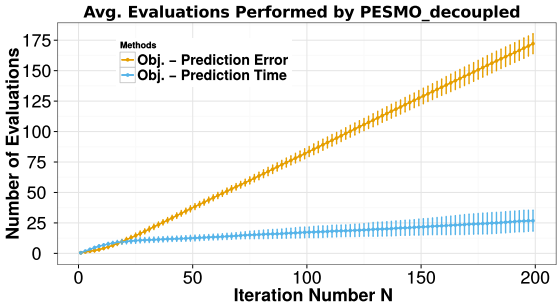

Figure 5. Number of evaluations of each objective done by $\text{PESMO}_{\text{decoupled}}$, as a function of the iteration number N, in the problem of finding good neural networks. Best seen in color.

The good results obtained by $\text{PESMO}_{\text{decoupled}}$ are explained by Figure 5, which shows the average number of evaluations of each objective. More precisely, the objective that measures the prediction time is evaluated just a few times. This makes sense because it depends on only two parameters, *i.e.*, the number of layers and the number of hidden units per layer. It is hence simpler than the prediction error. $\text{PESMO}_{\text{decoupled}}$ is able to detect this and focuses on the evaluation of the prediction error. Of course, evaluating the prediction error more times is more expensive, since it involves training the neural network more times. Nevertheless, this shows that $\text{PESMO}_{\text{decoupled}}$ is able to successfully discriminate between easy and difficult objective functions.

The supplementary material has extra experiments comparing each method on the task of finding an ensemble of decision trees of small size and good prediction accuracy. The results obtained are similar to the ones reported here.

5. Conclusions

We have described PESMO, a method for multi-objective Bayesian optimization. At each iteration, PESMO evaluates the objective functions at the input location that is most expected to reduce the entropy of posterior estimate of the Pareto set. Several synthetic experiments show that PESMO has better performance than other methods from the literature. That is, PESMO obtains better recommendations with a smaller number of evaluations, both in the case of noiseless and noisy observations. Furthermore, the acquisition function of PESMO can be understood as a sum of K individual acquisition functions, one per each of the K objectives. This allows for a *decoupled* evaluation scenario, in which the most promising objective is identified by maximizing the individual acquisition functions. When run in a decoupled evaluation setting, PESMO is able to identify the most difficult objectives and, by focusing on their evaluation, it provides better results. This behavior of PESMO has been illustrated on a multi-objective optimization problem that consists of finding an accurate and fast neural network. Finally, the computational cost of PESMO is small. In particular, it scales linearly with the number of objectives K. Other methods have an exponential cost with respect to K which makes them infeasible for more than 3 objectives.

Acknowledgments

Daniel Hernández-Lobato gratefully acknowledges the use of the facilities of Centro de Computación Científica (CCC) at Universidad Autónoma de Madrid. This author also acknowledges financial support from the Spanish Plan Nacional I+D+i, Grants TIN2013-42351-P and TIN2015-70308-REDT, and from Comunidad de Madrid, Grant S2013/ICE-2845 CASI-CAM-CM. José Miguel Hernández-Lobato acknowledges financial support from the Rafael del Pino Fundation. Amar Shah acknowledges support from the Qualcomm Innovation Fellowship program. Ryan P. Adams acknowledges support from the Alfred P. Sloan Foundation.

References

Ariizumi, R., Tesch, M., Choset, H., and Matsuno, F. Expensive multiobjective optimization for robotics with consideration of heteroscedastic noise. In *2014 IEEE International Conference on Intelligent Robots and Systems*, pp. 2230–2235, 2014.

Collette, Y. and Siarry, P. *Multiobjective Optimization: Principles and Case Studies*. Springer, 2003.

D. Kingma, J. Ba. Adam: A method for stochastic optimization. 2014. arXiv:1412.6980.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6: 182–197, 2002.

Emmerich, A. The computation of the expected improvement in dominated hypervolume of Pareto front approximations. Technical Report LIACS TR-4-2008, Leiden University, The Netherlands, 2008.

Feliot, P., Bect, J., and Vazquez, E. A Bayesian approach to constrained single- and multi-objective optimization. 2015. arXiv:1510.00503 [stat.CO].

Gelbart, M. A., Snoek, J., and Adams, R. P. Bayesian optimization with unknown constraints. In *Thirtieth Conference on Uncertainty in Artificial Intelligence*, 2014.

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

Hernández-Lobato, J. M., Hoffman, M. W., and Ghahramani, Z. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems 27*, pp. 918–926. 2014.

Houlsby, N., Hernández-lobato, J. M., Huszár, F., and Ghahramani, Z. Collaborative Gaussian processes for preference learning. In *Advances in Neural Information Processing Systems 25*, pp. 2096–2104. 2012.

Hupkens, I., Emmerich, M., and Deutz, A. Faster computation of expected hypervolume improvement. 2014. arXiv:1408.7114 [cs.DS].

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

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

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Li, X. A non-dominated sorting particle swarm optimizer for multiobjective optimization. In *Genetic and Evolutionary Computation GECCO 2003*, pp. 37–48. 2003.

Minka, T. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, MIT, 2001.

Mockus, J., Tiesis, V., and Zilinskas, A. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2(117-129):2, 1978.

Nakayama, H., Yun, Y., and M.Yoon. *Sequential Approximate Multiobjective Optimization Using Computational Intelligence*. Springer, 2009.

Picheny, V. Multiobjective optimization using Gaussian process emulators via stepwise uncertainty reduction. *Statistics and Computing*, 25:1265–1280, 2015.

Ponweiser, W., Wagner, T., Biermann, D., and Vincze, M. Multiobjective optimization on a limited budget of evaluations using model-assisted $\mathcal{S}$-metric selection. In *Parallel Problem Solving from Nature PPSN X*, pp. 784–794. 2008.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2006.

Shah, A. and Ghahramani, Z. Parallel predictive entropy search for batch global optimization of expensive objective functions. In *Advances in Neural Information Processing Systems 28*, pp. 3312–3320. 2015.

Singh, H., Misra, N., Hnizdo, V., Fedorowicz, A., and Demchuk, E. Nearest neighbor estimates of entropy. *American journal of mathematical and management sciences*, 23:301–321, 2003.

Villemonteix, J., Vazquez, E., and Walter, E. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44:509–534, 2009.

Zitzler, E. and Thiele, L. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, 3:257–271, 1999.

Zuluaga, M., Krause, A., Sergent, G., and Püschel, M. Active learning for multi-objective optimization. In *International Conference on Machine Learning*, pp. 462–470, 2013.