Constrained Bayesian Optimization with Max-Value Entropy Search: Supplemental Material

Valerio Perrone, Iaroslav Shcherbatyi, Rodolphe Jenatton,* Cédric Archambeau, Matthias Seeger
Amazon
Berlin, Germany
{vperrone, siarosla, cedrica, matthis}@amazon.com

1 Derivations

Consider the problem of Bayesian Optimization (BO) with unknown constraints. There are two real-valued functions $y(\boldsymbol{x})$, $c(\boldsymbol{x})$ over a common space $\mathcal{X}$. The first models the criterion to be minimized, the second parameterizes the constraint. An evaluation produces z_y, z_c according to likelihood functions. First,

$$z_y \sim N(z_y|y(\boldsymbol{x}), \alpha_y^{-1}).$$

For z_c, we consider two different options. First, we may observe $c(\boldsymbol{x})$ directly, up to Gaussian noise:

$$z_c \sim N(z_c|c(\boldsymbol{x}), \alpha_c^{-1}).$$

Second, we may observe a binary target only:

$$z_c \sim \sigma(z_c c(\boldsymbol{x})), \quad z_c \in \{\pm 1\}.$$

Here, $z_c = -1$ means the evaluation at $\boldsymbol{x}$ is feasible, and $z_c = +1$ means it is infeasible. We use the logistic parameterization, involving

$$\sigma(t) = \frac{1}{1+e^{-t}},$$

but any other likelihood could be used instead. The constrained optimization problem we would like to solve is

$$y_* = \min_{\boldsymbol{x}\in\mathcal{X}} \{y(\boldsymbol{x}) \parallel c(\boldsymbol{x}) \leq \delta\}. \tag{1}$$

Here, δ is a confidence parameter. In the case of binary feedback, $z_c \in \{\pm 1\}$, we can also write

$$y_* = \min_{\boldsymbol{x}\in\mathcal{X}} \{y(\boldsymbol{x}) \parallel P(z_c = +1|\boldsymbol{x}) = \sigma(c(\boldsymbol{x})) \leq \sigma(\delta)\},$$

where the confidence parameter is $\sigma(\delta) \in (0, 1)$. Importantly, both $y(\cdot)$ and $c(\cdot)$ are unknown up front and have to be learned from noisy samples z_y, z_c.

We assume that some data $\mathcal{D}$ has already been acquired, based on which independent *Gaussian* posterior processes are obtained for $y(\boldsymbol{x})$ and $c(\boldsymbol{x})$. The marginals of these are denoted by $N(y|\mu_y, \sigma_y^2)$ and $N(c|\mu_c, \sigma_c^2)$, where we drop the indexing by $\boldsymbol{x}$. In the sequel, we drop both the conditioning on $\boldsymbol{x}$ and on $\mathcal{D}$ from the notation. For example, we write $P(y, c)$ instead of $P(y, c|\mathcal{D}, \boldsymbol{x})$:

$$P(y, c) = P(y)P(c) = N(y|\mu_y, \sigma_y^2)N(c|\mu_c, \sigma_c^2).$$

The MES acquisition function [1] without constraints is given by:

$$\mathcal{I}(y; y_*) = \mathrm{H}[P(y)] - \mathrm{E}\left[\mathrm{H}[P(y|y_*)]\right],$$

*Work done while affiliated with Amazon; now at Google Brain, Berlin, rjenatton@google.com

3rd Workshop on Meta-Learning at NeurIPS 2019, Vancouver, Canada.

where the expectation is over $P(y_*|\mathcal{D})$, and $y_* = \min_{\boldsymbol{x}\in\mathcal{X}} y(\boldsymbol{x})$. Here, $P(y|y_*) \propto P(y)\mathrm{I}_{\{y\geq y_*\}}$ is a truncated Gaussian. It should be noted that this is a simplifying assumption. In PES [2], the related distribution $P(y|\boldsymbol{x}_*)$ is approximated, where $\boldsymbol{x}_*$ is the argmin. Several local constraints on $y(\cdot)$ at $\boldsymbol{x}_*$ are taken into account, such as $\nabla_{\boldsymbol{x}_*} y = \mathbf{0}$. This is not done in MES, which simplifies derivations dramatically. Second, the expectation over y_* is approximated by Monte Carlo sampling.

1.1 Real-valued Constraint Feedback

In this section, we assume that the constraint function $c(\cdot)$ can be observed directly, so that we obtain real-valued feedback from both $y(\cdot)$ and $c(\cdot)$. Our generalization of MES to the constrained case uses

$$A_1(\boldsymbol{x}) = \mathcal{I}((y,c); y_*) = \mathrm{H}[P(y,c)] - \mathrm{E}\left[\mathrm{H}[P(y,c|y_*)]\right], \tag{2}$$

where the expectation is over $P(y_*|\mathcal{D})$, and y_* is the constrained minimum (1). There are two points to be worked out:

- Expression $\mathrm{H}[P(y,c)] - \mathrm{H}[P(y,c|y_*)]$ for fixed y_*
- Efficient approximate sampler from $P(y_*|\mathcal{D})$, where y_* is given by (1), given that $y(\cdot)$ and $c(\cdot)$ are sampled from their respective posterior distributions (assumed to be independent).

In fact, the formulation so far ignores that we observe z_y, z_c at $\boldsymbol{x}$, not $y(\boldsymbol{x})$, $c(\boldsymbol{x})$. Even though this is ignored in the original MES paper, a better acquisition function would therefore be

$$A_2(\boldsymbol{x}) = \mathcal{I}((z_y,z_c); y_*) = \mathrm{H}[P(z_y,z_c)] - \mathrm{E}\left[\mathrm{H}[P(z_y,z_c|y_*)]\right]. \tag{3}$$

We start with the entropy difference in (2), where noise models are ignored, and come back to the noisy case (3) below. We will define $P(y,c|y_*)$ in the same "local" way as in MES, avoiding all complications as considered in PES. What do we learn by conditioning on y_*? If $c \leq \delta$, then $y \geq y_*$. Otherwise ($c > \delta$), our belief in y remains the same. Therefore:

$$P(y,c|y_*) = Z^{-1}P(y,c)\mathrm{I}_{\{c>\delta \vee y\geq y_*\}} = Z^{-1}P(y,c)(1 - \mathrm{I}_{\{c\leq\delta\}}\mathrm{I}_{\{y\leq y_*\}}).$$

Here, we replaced $y < y_*$ by $y \leq y_*$, which makes no difference for a distribution with a density. In the remainder of this section, $\mathrm{E}[\cdot]$ is always over $P(y,c)$, unless otherwise indicated. Denote

$$\kappa(y,c) := 1 - \mathrm{I}_{\{c\leq\delta\}}\mathrm{I}_{\{y\leq y_*\}} \quad \Rightarrow \quad P(y,c|y_*) = Z^{-1}P(y,c)\kappa(y,c).$$

We need some notation:

$$\gamma_c := \frac{\delta - \mu_c}{\sigma_c}, \quad \gamma_y := \frac{y_* - \mu_y}{\sigma_y}, \quad Z_c = \mathrm{E}[\mathrm{I}_{\{c\leq\delta\}}] = \Phi(\gamma_c), \quad Z_y = \mathrm{E}[\mathrm{I}_{\{y\leq y_*\}}] = \Phi(\gamma_y).$$

Here, $\Phi(t) = \mathrm{E}[\mathrm{I}_{\{n\leq t\}}]$, $n \sim N(0,1)$, is the cumulative distribution function for a standard normal variate. The normalization constant is

$$Z = \mathrm{E}[\kappa(y,c)] = 1 - Z_c Z_y.$$

Also,

$$\mathrm{H}[P(y,c|y_*)] = Z^{-1}\mathrm{E}\left[\kappa(y,c)(\log Z - \log P(y,c))\right] = \log Z + Z^{-1}\mathrm{E}\left[\kappa(y,c)(-\log P(y,c))\right].$$

Note that the $-\log\kappa(y,c)$ drops out, because $1\log 1 = 0\log 0 = 0$. If we parameterize $c = \mu_c + \sigma_c n_c$, $y = \mu_y + \sigma_y n_y$, where n_c, n_y are independent $N(0,1)$ variates, we have that

$$-\log P(y,c) = \frac{1}{2}\left(n_c^2 + n_y^2 + \log(2\pi\sigma_c^2) + \log(2\pi\sigma_y^2)\right).$$

Plugging this in:

$$\mathrm{H}[P(y,c|y_*)] = \log Z + \frac{1}{2}\left(\log(2\pi\sigma_c^2) + \log(2\pi\sigma_y^2)\right) + \frac{1}{2Z}\mathrm{E}\left[(1 - \mathrm{I}_{\{n_c\leq\gamma_c\}}\mathrm{I}_{\{n_y\leq\gamma_y\}})(n_c^2 + n_y^2)\right].$$

At this point, we need the simple identity:

$$\mathrm{E}[\mathrm{I}_{\{n\leq\gamma\}}n^2] = \mathrm{E}[\mathrm{I}_{\{n\leq\gamma\}}] - \gamma N(\gamma) = \Phi(\gamma) - \gamma N(\gamma), \quad N(x) := N(x|0,1).$$

Concentrating on the final expectation term:

$$\begin{aligned}(2Z)^{-1}\mathrm{E}[\dots] &= Z^{-1} - (2Z)^{-1}\mathrm{E}\left[\mathrm{I}_{\{n_c\le\gamma_c\}}\mathrm{I}_{\{n_y\le\gamma_y\}}(n_c^2+n_y^2)\right] \\ &= Z^{-1} - (2Z)^{-1}\left(Z_y(Z_c-\gamma_c N(\gamma_c)) + Z_c(Z_y-\gamma_y N(\gamma_y))\right) \\ &= Z^{-1}\left(Z + \frac{1}{2}\left(Z_y\gamma_c N(\gamma_c) + Z_c\gamma_y N(\gamma_y)\right)\right).\end{aligned}$$

The hazard function of the standard normal is defined as

$$h(x) := \frac{N(x)}{\Phi(-x)}.$$

Noting that $\mathrm{H}[P(y,c)] = \mathrm{H}[P(y)] + \mathrm{H}[P(c)]$ and $\mathrm{H}[P(y)] = (1+\log(2\pi\sigma_y^2))/2$, some algebra gives

$$\mathrm{H}[P(y,c|y_*)] = \mathrm{H}[P(y,c)] + \log Z + \frac{\gamma_c h(-\gamma_c) + \gamma_y h(-\gamma_y)}{2(\exp(-\log Z_c - \log Z_y) - 1)}.$$

Here, we used

$$\frac{Z_y Z_c}{Z} = \frac{Z_y Z_c}{1 - Z_y Z_c} = \frac{1}{\exp(-\log Z_c - \log Z_y) - 1}.$$

All in all:

$$\mathrm{H}[P(y,c)] - \mathrm{H}[P(y,c|y_*)] = -\log Z - \frac{\gamma_c h(-\gamma_c) + \gamma_y h(-\gamma_y)}{2(\exp(-\log Z_c - \log Z_y) - 1)}.$$

Note that $\log Z_c$, $\log Z_y$ are negative. The only case when this expression becomes problematic is if both $\log Z_y$ and $\log Z_c$ tend to zero. This happens only if both y is much smaller than y_* and c is much smaller than δ. If y_* is sampled from $P(y_*|\mathcal{D})$, this is very unlikely to be the case. We need numerically robust code for computing $\log\Phi(x)$ and $h(x)$.

1.2 Entropy Difference for Noisy Targets

As noted above, we would ideally compute the entropy difference for the noisy targets z_y, z_c instead of the latents y, c, so use the acquisition function (3) instead of (2). How would this look like for the case where both z_y and z_c are real-valued with Gaussian likelihood? Define

$$\Psi(z_y, z_c) = \int P(z_y, y)P(z_c, c)\kappa(y, c)\, dydc = P(z_y)P(z_c)\left(1 - \tilde{Z}_y(z_y)\tilde{Z}_c(z_c)\right),$$

where $\tilde{Z}_y(z_y)$ is defined as Z_y, but with $P(y)$ being replaced by the posterior $P(y|z_y)$. Then:

$$P(z_y, z_c|y_*) = Z^{-1}\Psi(z_y, z_c), \quad Z = 1 - Z_y Z_c.$$

To our knowledge, there is no simple closed-form expression for $\mathrm{H}[P(z_y, z_c|y_*)]$. The problem is that $\Psi(z_y, z_c)$ is not the product of a Gaussian with an indicator, and in particular $\log\Psi(z_y, z_c)$ is a complex function.

Here is a simple idea which may work better than just ignoring the noise and using (2). Complications arise because the expectations over $P(y|z_y)$ and $P(c|z_c)$ in $\Psi(z_y, z_c)$ do not result in a term which is the product of Gaussians and indicators. We can mitigate this problem by approximating $P(y|z_y)$ with $\delta(y - \mathrm{E}[y|z_y])$. Doing so results in

$$\Psi(z_y, z_c) = P(z_y)P(z_c)(1 - \mathrm{I}_{\{\mathrm{E}[y|z_y]\le y_*\}}\mathrm{I}_{\{\mathrm{E}[c|z_c]\le\delta\}}).$$

Here, $P(z_y) = N(\mu_y, \sigma_y^2 + \alpha_y^{-1})$, $P(z_c) = N(\mu_c, \sigma_c^2 + \alpha_c^{-1})$. Since $\mathrm{E}[y|z_y]$ is an affine function of z_y, this can be brought into the same form as is used in the noise-free case, but y is replaced by z_y, y_* by a different value, and $P(y)$ by $P(z_y)$. Namely,

$$\mathrm{E}[y|z_y] = \mu_y + \frac{\sigma_y^2}{\sigma_y^2 + \alpha_y^{-1}}(z_y - \mu_y) = \mu_y + \rho_y^2(z_y - \mu_y), \quad \rho_y^2 = \frac{\sigma_y^2\alpha_y}{1 + \sigma_y^2\alpha_y},$$

so that

$$\mathrm{E}[y|z_y] \le y_* \quad \Leftrightarrow \quad z_y \le \tilde{y}_* := \mu_y + \rho_y^{-2}(y_* - \mu_y).$$

We can now simply use the derivation from above. In fact,

$$\tilde{\gamma}_y = \frac{\tilde{y}_* - \mu_y}{(\sigma_y^2 + \alpha_y^{-1})^{1/2}} = \frac{y_* - \mu_y}{\sigma_y\rho_y}, \quad \tilde{\gamma}_c = \frac{\delta - \mu_c}{\sigma_c\rho_c}, \quad \rho_y = \frac{\sigma_y\alpha_y^{1/2}}{\sqrt{1 + (\sigma_y\alpha_y^{1/2})^2}}$$

just have to be used instead of γ_y, γ_c.

1.3 Binary Constraint Feedback

For binary response $z_c \in \{\pm 1\}$, we have to take into account that much less information is obtained by sampling the constraint at $\boldsymbol{x}$. Here, a sensible approach is to ignore the noise on y, but not ignore the likelihood $c \to z_c$. In other words, we can try to approximate

$$A_3(\boldsymbol{x}) = \mathcal{I}((y, z_c); y_*) = \mathrm{H}[P(y, z_c)] - \mathrm{E}\left[\mathrm{H}[P(y, z_c|y_*)]\right]. \tag{4}$$

In this case, we use some approximate inference method for

$$Q(z_c)Q(c|z_c) \approx P(z_c|c)P(c), \quad z_c \in \{\pm 1\},$$

where $Q(c|z_c)$ are Gaussians. In our current code, we use Laplace's approximation, where mode finding is approximated by a single Newton step. Also, $Q(z_c)$ is using the highly accurate approximation given in [3, Sect. 4.5.2]. Now:

$$\Psi(y, z_c) := \int Q(z_c)Q(c|z_c)P(y)\kappa(y, c)\, dc = P(y)Q(z_c)\tilde{\kappa}(y, z_c),$$
$$\tilde{\kappa}(y, z_c) := \left(1 - \mathrm{I}_{\{y \le y_*\}}F(z_c)\right), \; F(z_c) = \mathrm{E}_{Q(c|z_c)}[\mathrm{I}_{\{c \le \delta\}}],$$

and

$$P(y, z_c|y_*) \approx Z^{-1}\Psi(y, z_c), \quad Z = 1 - Z_y\tilde{Z}_c, \quad \tilde{Z}_c = \mathrm{E}_Q[F(z_c)].$$

Importantly, $\tilde{\kappa}(y, z_c)$ is piece-wise constant, while not an indicator function anymore. Note that $\tilde{Z}_c \neq Z_c$ in general, due to the approximation we use, but it should be close.

In the following, $\mathrm{E}[\cdot]$ is over $P(y)Q(z_c)$, $\mathrm{E}_P[\cdot]$ is over $P(y)$, and $\mathrm{E}_Q[\cdot]$ is over $Q(z_c)$. First,

$$\mathrm{H}[P(y, z_c|y_*)] = \log Z + Z^{-1}\mathrm{E}\left[\tilde{\kappa}(y, z_c)\left(-\log P(y) - \log Q(z_c) - \log \tilde{\kappa}(y, z_c)\right)\right]$$
$$= \log Z + \frac{1}{2}\log(2\pi\sigma_y^2) + \mathrm{E}_Q[G(z_c)],$$
$$G(z_c) := Z^{-1}\mathrm{E}_P\left[\tilde{\kappa}(y, z_c)\left(n_y^2/2 - \log Q(z_c) - \log \tilde{\kappa}(y, z_c)\right)\right].$$

We split this in three parts, using the derivation of the noise-free case above. First:

$$G_1(z_c) = Z^{-1}\mathrm{E}_P\left[\tilde{\kappa}(y, z_c)n_y^2/2\right] = \frac{1}{2Z}\left(1 - F(z_c)(Z_y - \gamma_y N(\gamma_y))\right).$$

Next:

$$G_2(z_c) = Z^{-1}\mathrm{E}_P\left[\tilde{\kappa}(y, z_c)(-\log Q(z_c))\right] = Z^{-1}(1 - Z_y F(z_c))(-\log Q(z_c)).$$

Finally, note that if $y \ge y_*$, then $\log \tilde{\kappa}(y, z_c) = \log 1 = 0$, so we can replace $\tilde{\kappa}(y, z_c)$ by $\mathrm{I}_{\{y \le y_*\}}(1 - F(z_c))$, therefore:

$$G_3(z_c) = Z^{-1}\mathrm{E}_P\left[\tilde{\kappa}(y, z_c)(-\log \tilde{\kappa}(y, z_c))\right] = Z^{-1}Z_y(1 - F(z_c))(-\log(1 - F(z_c))).$$

Next, the expectation over $Q(z_c)$. First,

$$G_1(z_c) = \frac{1}{2Z}\left(1 - F(z_c)Z_y + F(z_c)\gamma_y N(\gamma_y)\right),$$

so that

$$\mathrm{E}_Q[G_1(z_c)] = \frac{1}{2} + \frac{Z_y\tilde{Z}_c}{2Z}\gamma_y h(-\gamma_y).$$

Next, using $1 - Z_y F(z_c) = Z - Z_y(F(z_c) - \tilde{Z}_c)$:

$$\mathrm{E}_Q[G_2(z_c)] = Z^{-1}\mathrm{E}_Q\left[(1 - Z_y F(z_c))(-\log Q(z_c))\right]$$
$$= \mathrm{H}[Q(z_c)] - \frac{Z_y\tilde{Z}_c}{Z}\tilde{Z}_c^{-1}\mathrm{E}_Q\left[(F(z_c) - \tilde{Z}_c)(-\log Q(z_c))\right].$$

Finally,

$$\mathrm{E}_Q[G_3(z_c)] = \frac{Z_y\tilde{Z}_c}{Z}\tilde{Z}_c^{-1}\mathrm{E}_Q\left[(1 - F(z_c))(-\log(1 - F(z_c)))\right].$$

Altogether, we obtain

$$\mathrm{H}[P(y)] + \mathrm{H}[Q(z_c)] - \mathrm{H}[P(y, z_c|y_*)] = -\log Z$$
$$- B\left(\gamma_y h(-\gamma_y)/2 + \tilde{Z}_c^{-1}\mathrm{E}_Q\left[(1 - F(z_c))(-\log(1 - F(z_c))) + (F(z_c) - \tilde{Z}_c)\log Q(z_c)\right]\right),$$
$$B = \frac{Z_y \tilde{Z}_c}{Z} = \frac{1}{\exp(-\log Z_y - \log \tilde{Z}_c) - 1}.$$

We would compute $\log Z_y$, $h(-\gamma_y)$, $\log F(z_c)$, $\log(1 - F(z_c))$, then $\log \tilde{Z}_c$ by `logsumexp`. In fact, if

$$\gamma_c(z_c) = \frac{\delta - \mathrm{E}_Q[c|z_c]}{\sqrt{\mathrm{Var}_Q[c|z_c]}},$$

then

$$\log F(z_c) = \log \Phi(\gamma_c(z_c)), \quad \log(1 - F(z_c)) = \log \Phi(-\gamma_c(z_c)).$$

The term $\tilde{Z}_c^{-1}\mathrm{E}_Q[\dots]$ is computed by folding the normalization into the argument inside $\mathrm{E}_Q[\dots]$, which is computed as

$$\left(e^{\log F(z_c) - \log \tilde{Z}_c} - 1\right)\log Q(z_c) - e^{\log(1 - F(z_c)) - \log \tilde{Z}_c}\log(1 - F(z_c)).$$

We then multiply with $Q(z_c)$ and sum over $z_c = -1, +1$.

1.4 Sampling from $P(y_\star|\mathcal{D})$

In the constrained case, we aim to sample from $P(y_\star|\mathcal{D})$, where $y_\star = \min_{\boldsymbol{x}\in\mathcal{X}}\{y(\boldsymbol{x}) \parallel c(\boldsymbol{x}) \le \delta\}$. Here, $y(\cdot)$ and $c(\cdot)$ are posterior GPs conditioned on the current data $\mathcal{D}$. At least for commonly used infinite-dimensional kernels, it is intractable to draw exact sample functions from these GPs, let alone to solve the conditional optimization problem for $y_\star$.

In [4], a finite-dimensional random kitchen sink (RKS) approximation is used to draw approximate sample paths, and the constrained problem is solved for these. Since the RKS basis functions are nonlinear in $\boldsymbol{x}$, so are objective and constraint function, and solving for $y_\star$ requires complex machinery. A simpler approach is used in [1]. They target the cumulative distribution function (CDF) of $y_\star$, which can be written as expectation over $y(\cdot)$ and $c(\cdot)$ of an infinite product. This is approximated by restricting the product over a finite set $\hat{\mathcal{X}}$, and by assuming *independence* of all $y(\boldsymbol{x})$ and $c(\boldsymbol{x})$ for $\boldsymbol{x} \in \hat{\mathcal{X}}$. While this gives rise to a tractable approximation of the CDF, we found this approximation to be problematic in our experiments. As noted in [1], $y_\star$ drawn under these assumptions are underbiased. In fact, due to the independence assumption, this bias gets *worse* the larger $\hat{\mathcal{X}}$ is: $y_\star$ diverges as $|\hat{\mathcal{X}}| \to \infty$.

In our experiments, we follow [1] by restricting our attention to a finite set $\hat{\mathcal{X}}$ (we use a Sobol sequence [5]), but then draw *joint* samples of $y(\hat{\mathcal{X}})$ and $c(\hat{\mathcal{X}})$ respectively, based on which $y_\star$ (restricted to $\hat{\mathcal{X}}$) is trivial to compute. While joint sampling scales cubically in the size of $\hat{\mathcal{X}}$, sampling takes less than a second for $|\hat{\mathcal{X}}| = 2000$, the size we used in our experiments.

More precisely, the posterior for $y(\cdot)$ conditioned on data $\boldsymbol{z}_y = [z_{yi}] \in \mathbb{R}^n$ is defined in terms of the Cholesky factor $\boldsymbol{L}$ and the vector $\boldsymbol{p}$, where

$$\boldsymbol{L}\boldsymbol{L}^T = \boldsymbol{K} + \alpha_y^{-1}\boldsymbol{I}, \quad \boldsymbol{p} = \boldsymbol{L}^{-1}\boldsymbol{z}_y,$$

where $\boldsymbol{K} = k_y(\boldsymbol{X}, \boldsymbol{X}) \in \mathbb{R}^{n\times n}$ is the kernel matrix on the training set ($\boldsymbol{X} = [\boldsymbol{x}_i] \in \mathbb{R}^{n\times p}$), and α_y is the noise precision. The posterior distribution of $y(\hat{\mathcal{X}})$ is a Gaussian with mean and covariance

$$\hat{\boldsymbol{\mu}} = \boldsymbol{M}\boldsymbol{p}, \quad \boldsymbol{M} = \boldsymbol{K}_{*,\cdot}\boldsymbol{L}^{-T}, \quad \hat{\boldsymbol{\Sigma}} = \boldsymbol{K}_{*,*} - \boldsymbol{M}\boldsymbol{M}^T,$$

where $\boldsymbol{K}_{*,\cdot} = k_y(\hat{\mathcal{X}}, \boldsymbol{X}) \in \mathbb{R}^{m\times n}$, $m = |\hat{\mathcal{X}}|$, and $\boldsymbol{K}_{*,*} = k_y(\hat{\mathcal{X}}, \hat{\mathcal{X}}) \in \mathbb{R}^{m\times m}$. Samples of $y(\hat{\mathcal{X}})$ are drawn as

$$\hat{\boldsymbol{Y}} = \hat{\boldsymbol{L}}\boldsymbol{N} + \hat{\boldsymbol{\mu}}\boldsymbol{1}_k^T, \quad \hat{\boldsymbol{L}}\hat{\boldsymbol{L}}^T = \hat{\boldsymbol{\Sigma}}, \quad \boldsymbol{N} = [\nu_{rs}] \in \mathbb{R}^{m\times k},\ \nu_{rs} \sim N(0, 1).$$

Due to the Cholesky factorization, joint sampling scales cubically in m. On the other hand, sampling takes less than one second for sizes smaller than 2000.

1.5 Scoring Constraint or Criterion Evaluation

In some situations, we may be able to evaluate criterion and constraints independent of each other. For example, one may be much cheaper to evaluate than the other. To this end, we would like to score the value of sampling $y(\boldsymbol{x})$ or $c(\boldsymbol{x})$ at $\boldsymbol{x}$.

To this end, we just marginalize the joint distributions worked out above. First, consider the case where z_y, z_c are real-valued, and we would like to score the value of sampling z_y (the case of sampling z_c is symmetric then). Note that this is the noisy case, where σ_y is replaced by $\sigma_y\rho_y$, γ_y by $\tilde{\gamma}_y$, etc. We have that

$$\Psi(z_y) = P(z_y)(1 - \mathrm{I}_{\{\mathrm{E}[y|z_y]\le y_*\}} Z_c), \quad Z = 1 - \tilde{Z}_y Z_c.$$

Then, $P(z_y|y_*) = Z^{-1}\Psi(z_y)$. Note that $Z_c = \Phi(\gamma_c)$ without the noise. In fact, the marginal does not depend on the noise $c \to z_c$, so the same expression is obtained in the case $z_c \in \{\pm 1\}$. In the following, we use that

$$\log(1 - \mathrm{I}_{\{\mathrm{E}[y|z_y]\le y_*\}} Z_c) = \mathrm{I}_{\{\mathrm{E}[y|z_y]\le y_*\}} \log(1 - Z_c).$$

Then:

$$\begin{aligned}\mathrm{H}[P(z_y|y_*)] = \log Z + \frac{1}{2}\log(2\pi\mathrm{Var}[z_y]) + Z^{-1}\mathrm{E}\Big[&(1 - Z_c\mathrm{I}_{\{n_y\le\tilde{\gamma}_y\}})n_y^2/2 \\ &+ \mathrm{I}_{\{n_y\le\tilde{\gamma}_y\}}(1 - Z_c)(-\log(1 - Z_c))\Big].\end{aligned}$$

Some algebra gives

$$\mathrm{H}[P(z_y)] - \mathrm{H}[P(z_y|y_*)] = -\log Z - \frac{\tilde{\gamma}_y h(-\tilde{\gamma}_y)/2 - Z_c^{-1}(1 - Z_c)\log(1 - Z_c)}{\exp(-\log\tilde{Z}_y - \log Z_c) - 1}, \quad Z = 1 - \tilde{Z}_y Z_c.$$

By symmetry, if $z_c \in \mathbb{R}$ with Gaussian noise:

$$\mathrm{H}[P(z_c)] - \mathrm{H}[P(z_c|y_*)] = -\log Z - \frac{\tilde{\gamma}_c h(-\tilde{\gamma}_c)/2 - Z_y^{-1}(1 - Z_y)\log(1 - Z_y)}{\exp(-\log\tilde{Z}_c - \log Z_y) - 1}, \quad Z = 1 - Z_y\tilde{Z}_c.$$

Finally, consider $z_c \in \{\pm 1\}$. Here,

$$\tilde{\kappa}(z_c) = 1 - Z_y F(z_c), \quad Z = 1 - Z_y\tilde{Z}_c, \quad P(z_c|y_*) = Z^{-1}Q(z_c)\tilde{\kappa}(z_c).$$

Then:

$$\mathrm{H}[P(z_c|y_*)] = \log Z + Z^{-1}\mathrm{E}_Q\left[\tilde{\kappa}(z_c)\left(-\log Q(z_c) - \log\tilde{\kappa}(z_c)\right)\right].$$

Some algebra gives

$$\mathrm{H}[Q(z_c)] - \mathrm{H}[P(z_c|y_*)] = -\log Z + Z^{-1}\mathrm{E}_Q\left[\tilde{\kappa}(z_c)\log\tilde{\kappa}(z_c) - Z_y(F(z_c) - \tilde{Z}_c)\log Q(z_c)\right],$$

where $Z = 1 - Z_y\tilde{Z}_c$. Using the notation from above, this can also be written as

$$\begin{aligned}&\mathrm{H}[Q(z_c)] - \mathrm{H}[P(z_c|y_*)] = -\log Z - Z^{-1}\mathrm{E}_Q[\tilde{\kappa}(z_c)(-\log\tilde{\kappa}(z_c))] \\ &- B\tilde{Z}_c^{-1}\mathrm{E}_Q[(F(z_c) - \tilde{Z}_c)\log Q(z_c)], \quad B = \frac{Z_y\tilde{Z}_c}{Z} = \frac{1}{\exp(-\log Z_y - \log\tilde{Z}_c) - 1}.\end{aligned}$$

1.6 Observe $y(\boldsymbol{x})$ only in Feasible Region

In this section, we deal with binary feedback $z_c \in \{-1, +1\}$. For some important applications, feedback z_y on $y(\boldsymbol{x})$ is obtained only if $z_c = -1$ (feasible). For example, BO may be used to tune parameters of deep neural networks. A function evaluation z_y of a test set metric may fail, because training crashed due to out of memory errors ($z_c = +1$). Note that y_* itself does not depend on values of $y(\boldsymbol{x})$ in the infeasible region.

It seems hard to properly define the entropy difference (conditioned on y_*) in this case. One idea is to simply use the entropy difference from Section 1.3. Even though this assumes noise-free feedback for y, the value conveys information about y_* only if $\boldsymbol{x}$ is feasible. Another idea is to consider the mixture of $Q(z_c = -1)$ times the entropy difference from Section 1.3 plus $Q(z_c = +1)$ times the entropy difference from Section 1.5. At least for $Q(z_c)$ away from $1/2$, this could be a more reasonable score. Note that the part

$$-\log Z - B\tilde{Z}_c^{-1}\mathrm{E}_Q[(F(z_c) - \tilde{Z}_c)\log Q(z_c)]$$

appears in both entropy difference expressions.

Model	Dataset	Constraint	Threshold	Feasible points	d
XGBoost	mg	model size	50000 bytes	72%	7
Decision tree	mpg	model size	3500 bytes	48%	4
Random forest	pyrim	model size	5000 bytes	26%	4
Random forest	cpusmall	model size	27000 bytes	80%	4
MLP	pyrim	model size	27000 bytes	79%	11
kNN + rnd. projection	australian	model size	28000 bytes	29%	5
MLP	heart	error on neg.	13.3%	30%	12
MLP	higgs	error on neg.	60%	38%	12
Factorization machine	heart	error on neg.	17%	39%	7
MLP	diabetes	error on neg.	80%	74%	12

Table 1: Constrained HPO problems considered in our experiments. Here d is a number of dimensions of a blackbox function.

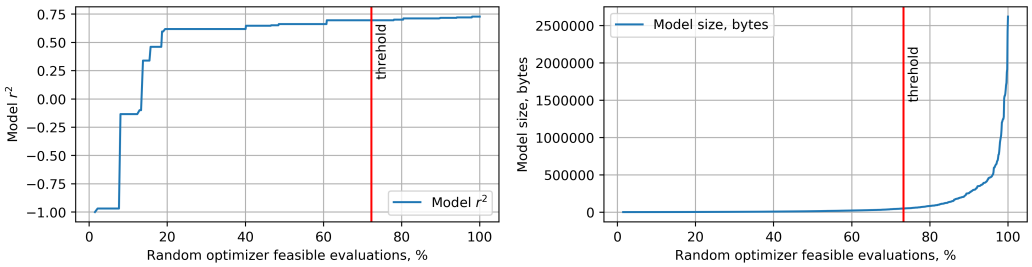

Figure 1: A tradeoff between model performance (r^2) and threshold value for `xgboost` model, on `mg` dataset. Smaller threshold results in a smaller number of weak learners, thus degrading the performance of the model.

2 Experiments

In this section, we provide additional details about our experiments.

2.1 Real-world Hyperparameter Tuning Problems

We created a wide range of constrained HPO problems, spanning different `scikitlearn` algorithms [6], `libsvm` datasets [7], and constraint modalities. Our results are for ten problems drawn from this range. The first six problems are about optimizing an accuracy metric[2], subject to a constraint on model size, a setup motivated by applications in IOT or on mobile devices. The remaining four problems require to minimize the error on positives, subject to a limit on the error on negatives[3], as is relevant for example in applications in medical domains. A summary of algorithms, datasets, and fraction of feasible configurations is given in Table 1. When sampling a problem, and then a hyperparameter configuration at random, we hit a feasible point with probability 51.5%. Also note that for all these problems, the overall global minimum point is unfeasible.

All our example functions require a threshold, either on the size of trained model, or on the error on negatives. For a given HPO problem, the threshold is chosen as follows. First, we sample 2000 random values of the criterion function, without constraint. This allows us to access an effect of a particular threshold on the value of objective, and on a fraction of points which are unfeasible. We select a threshold at random, such that a total fraction of unfeasible points is within 20% to 80% interval. An example visualization for `xgboost` is given in Figure 1.

[2] AUC for binary classification, coefficient of determination for regression.

[3] Here, one hyperparameter to tune is the fraction of the positive class in the data (both training and validation), which is adjusted by resampling with replacement.

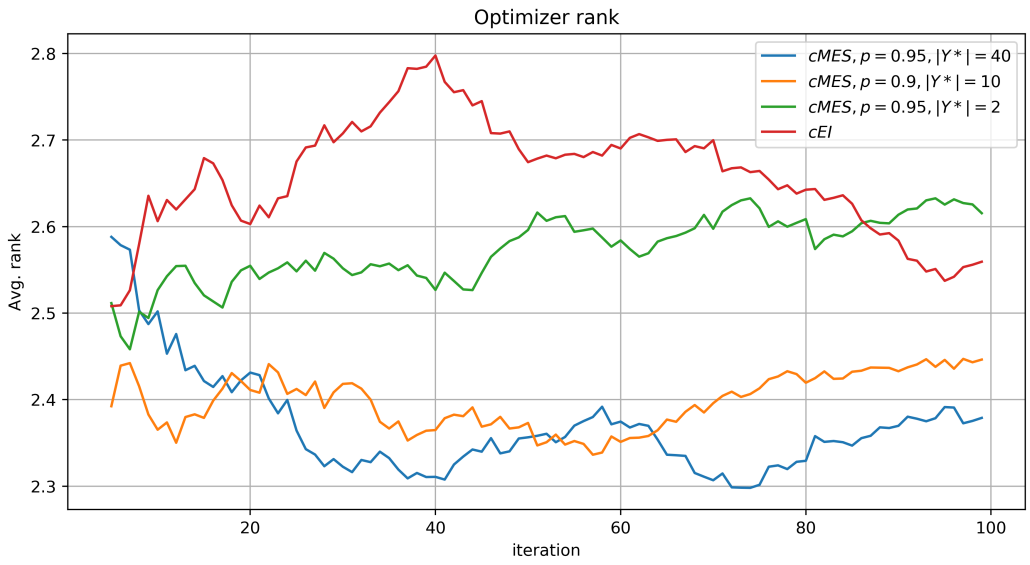

Figure 2: Average ranking of $cMES$ and cEI, where objective is not available when a constraint is violated.

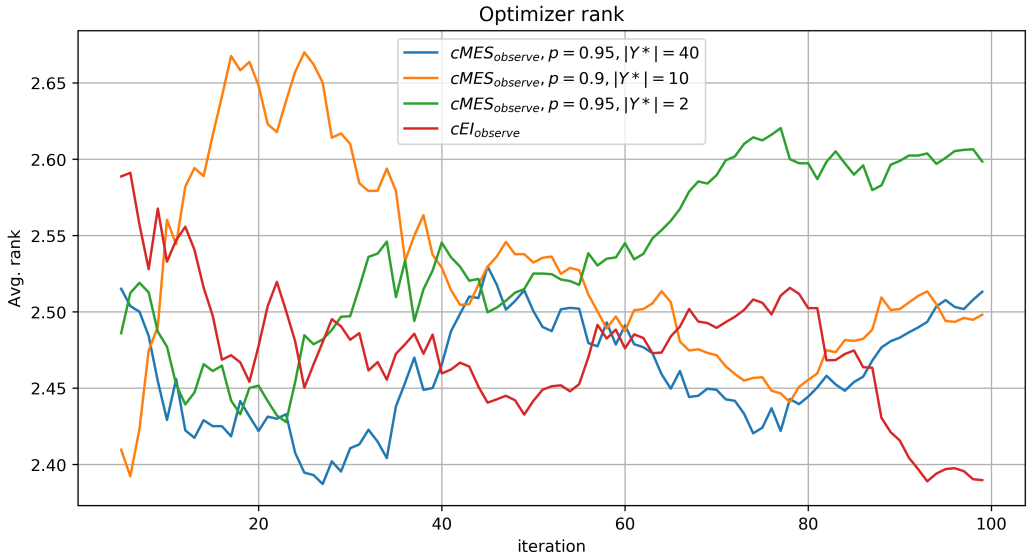

Figure 3: Average ranking of $cMES$ and cEI; objective is available when a constraint is violated.

2.2 Effect of number of y^* samples on optimization performance

Let Y^* be a set of all sampled minima, and let $|Y^*|$ be its size. In our experiments, using more than 10 samples of y^* does not lead to improvement of the algorithm performance. Results are summarized in Figure 2 and 3 and Table 2, 3.

Optimizers	Unfeasible evaluations, percent	Average ranking
$cMES, p = 0.5, \|Y*\| = 40$	43.08	5.55
$cMES, p = 0.9, \|Y*\| = 40$	47.14	5.5
$cMES, p = 0.95, \|Y*\| = 40$	49.12	5.22
$cMES, p = 0.1, \|Y*\| = 10$	39.25	5.58
$cMES, p = 0.5, \|Y*\| = 10$	41.41	5.48
$cMES, p = 0.9, \|Y*\| = 10$	46.75	**5.09**
$cMES, p = 0.5, \|Y*\| = 2$	43.29	5.59
$cMES, p = 0.9, \|Y*\| = 2$	48.92	5.6
$cMES, p = 0.95, \|Y*\| = 2$	51.07	5.49
cEI	**24.26**	5.89

Table 2: Comparison of various number of $|Y^*|$ on ten `scikitlearn` problems. Here, $p = \sigma(\delta)$ for cMES. AP denotes adaptive percentile. For methods with subscript $observe$, the objective $y(\cdot)$ is observed at unfeasible points. No objective is available when a constraint is violated.

Optimizers	Unfeasible evaluations, percent	Average ranking
$cMES_{observe}, p = 0.5, \|Y*\| = 40$	46.31	5.34
$cMES_{observe}, p = 0.9, \|Y*\| = 40$	55.5	5.73
$cMES_{observe}, p = 0.95, \|Y*\| = 40$	54.95	5.33
$cMES_{observe}, p = 0.1, \|Y*\| = 10$	46.73	5.87
$cMES_{observe}, p = 0.5, \|Y*\| = 10$	48.93	5.7
$cMES_{observe}, p = 0.9, \|Y*\| = 10$	53.23	5.4
$cMES_{observe}, p = 0.5, \|Y*\| = 2$	47.81	**5.2**
$cMES_{observe}, p = 0.9, \|Y*\| = 2$	55.45	5.68
$cMES_{observe}, p = 0.95, \|Y*\| = 2$	57.81	5.37
$cEI_{observe}$	**35.54**	5.4

Table 3: Comparison of various number of $|Y^*|$ on ten `scikitlearn` problems. Here, $p = \sigma(\delta)$ for cMES. AP denotes adaptive percentile. For methods with subscript $observe$, the objective $y(\cdot)$ is observed at unfeasible points. Objective is available when a constraint is violated.

References

[1] Z. Wang and S. Jegelka. Max-value entropy search for efficient Bayesian optimization. In D. Precup and Y. W. Teh, editors, *International Conference on Machine Learning 34*. JMLR.org, 2017.

[2] D. Hernandez-Lobato, J. Hernandez-Lobato, A. Shah, and R. Adams. Predictive entropy search for multi-objective Bayesian optimization. In M. Balcan and K. Weinberger, editors, *International Conference on Machine Learning 33*. JMLR.org, 2016.

[3] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 1st edition, 2006.

[4] José Miguel Hernández-Lobato, Michael A. Gelbart, Matthew W. Hoffman, Ryan P. Adams, and Zoubin Ghahramani. Predictive entropy search for Bayesian optimization with unknown constraints. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1699–1707, 2015.

[5] Ilya M Sobol. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967.

[6] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research (JMLR)*, 12:2825–2830, 2011.

[7] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.