,

BOPrO

Bayesian Optimization with a Prior for the Optimum

Bartłomiej Sobieski

Faculty of Mathematics and Information Science
Warsaw University of Technology

January 17, 2022

Overview

Introduction

BOPrO:

- is a bayesian optimization variant that combines priors for the optimum with a probabilistic model of the observations made.
- allows users to inject priors that were previously difficult to inject into bayesian optimization.
- converges quickly even if the prior is not entirely accurate
- recovers from misleading priors.

Mathematical background

We're gonna usually refer to finding the best hyperparameter configuration of a machine learning model since it's closest to our intuition. Therefore we can assume that our goal is to minimize an unknown function

$$f : \mathcal{X} \to \mathbb{R}$$

that is expensive to evaluate over an input space X. Which is the same as finding

$$\mathbf{x}^* \in arg\,min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}).$$

Bayesian optimization

Bayesian optimization, which is our foundation here, approximates $\mathbf{x}^*$ with an optimal sequence of evaluations on $\mathbf{x}_1, \mathbf{x}_2, ... \in \mathcal{X}$ that maximizes some kind of a utility metric.

Each new $\mathbf{x}_{t+1}$ depends on previous function values $y_1, y_2, ..., y_t$ at $\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_t$ respectively.

Approximation is achieved by building a posterior on f based on the set of evaluated points. At each iteration a new point is selected and evaluated based on the posterior and the posterior is updated to include the new point $(\mathbf{x}_{t+1}, y_{t+1})$.

Bayesian optimization

Bayesian optimization requires having some kind of a 'guide' to dictate the points that it's going to explore. As we've already heard on the previous seminars, this 'guide' is called the acquisition function. It attributes an utility to each $\mathbf{x} \in \mathcal{X}$ by balancing:

- the predicted value of f
- uncertainty of the prediction

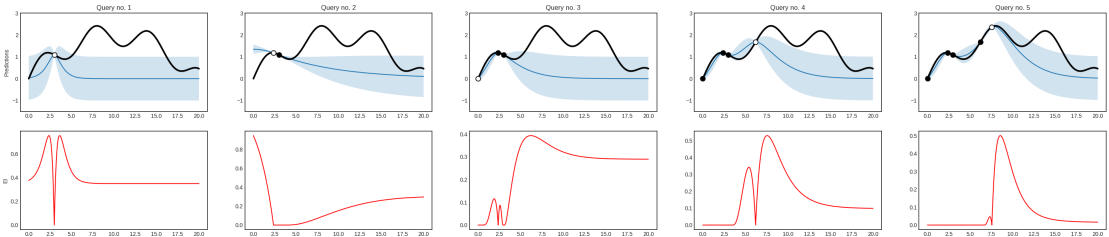
Bayesian optimization

The described method chooses *Expected Improvement (EI)* as its acquisition function which is given by:

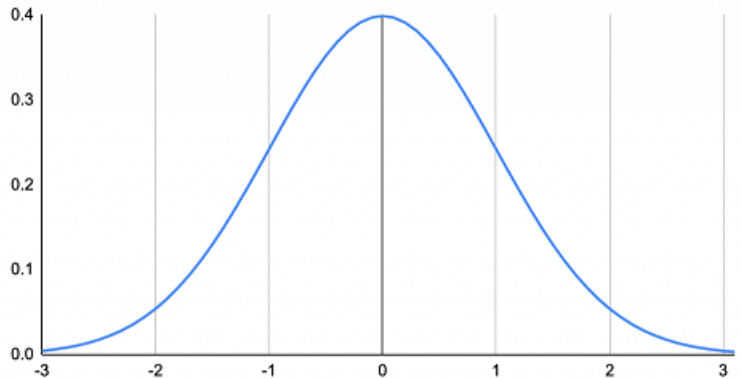
$$EI_{y_{inc}}(\mathbf{x}) := \int_{-\infty}^{\infty} max(y_{inc} - y, 0) p(y|\mathbf{x}) dy,$$

where y_{inc} is the incumbent function value (i.e. the best function value found so far) and $p(y|\mathbf{x})$ is given by a probabilistic model (e.g. a gaussian process but the presented method is generally independant of the model being used).

Expected Improvement visualized



BOPrO priors



BOPrO priors

Values	Prior
1	0.4
4	0.065
8	0.07
16	0.065
32	0.4

BOPrO priors

Algorithm 1 BOPrO Algorithm. $\mathcal{D}_t$ keeps track of all function evaluations so far: $(\boldsymbol{x}_i, y_i)_{i=1}^t$.

1: **Input:** Input space $\mathcal{X}$, user-defined prior distribution $P_g(\boldsymbol{x})$, quantile γ, initial model weight β, and BO budget B.
2: **Output:** Optimized point $\boldsymbol{x}_{inc}$.

BOPrO priors

BOPrO allows injecting prior knowledge with respect to promising areas into bayesian optimization. It's done by a prior distribution that informs where in the input space $\mathcal{X}$ we expect to find 'good' $f(\mathbf{x})$ values. The point $\mathbf{x}$ is obviously considered 'good' if it leads to low $f(\mathbf{x})$ values and potentially to a global minimum. Because of that we denote:

$$P_g(\mathbf{x}) \text{ - prior on good } (g) \text{ points}$$
$$P_b(\mathbf{x}) \text{ - prior on bad } (b) \text{ points}$$

Although we could allow for a user-defined probability distribution $P_b(\mathbf{x})$, it's better to keep the decision-making load on users low. Thus we will only require the definition of $P_g(\mathbf{x})$ and compute

$$P_b(\mathbf{x}) = 1 - P_g(\mathbf{x}).$$

BOPrO priors

In practice $\mathbf{x}$ has several dimensions and it might usually be difficult even for an expert to provide a joint probability distribution $P_g(\mathbf{x})$ for all of them. However, users can typically easily specify univariate or bivariate distributions for continuous dimensions or provide a list of probabilities for discrete dimensions. BOPrO allows the user to define complex multivariate distribution, but the expected standard use case is choosing to specify univariate distributions, implicitly assuming a prior that factors as

$$P_g(\mathbf{x}) = \prod_{i=1}^{D} P_g(x_i),$$

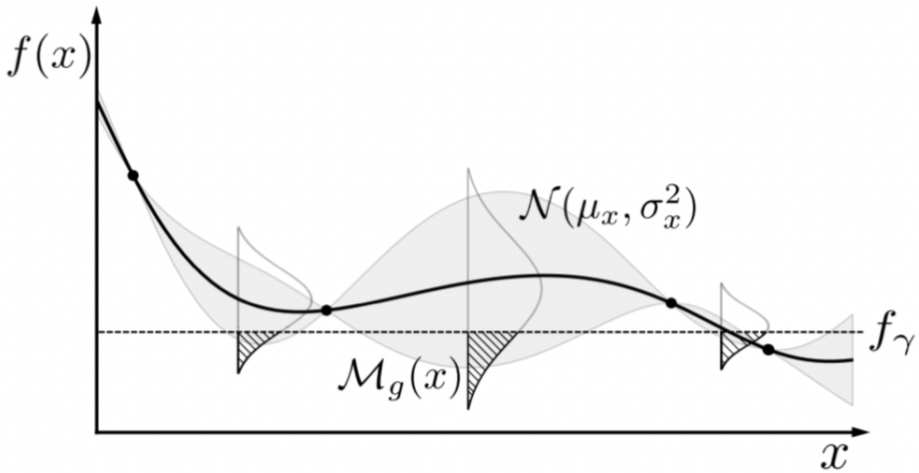
where D is the number of dimensions and x_i is i-th input dimension of $\mathbf{x}$. This factorized prior in fact leads to similiar performance as a multivariate prior.

BOPrO model

Algorithm 1 BOPrO Algorithm. $\mathcal{D}_t$ keeps track of all function evaluations so far: $(\boldsymbol{x}_i, y_i)_{i=1}^{t}$.

1: **Input:** Input space $\mathcal{X}$, user-defined prior distribution $P_g(\boldsymbol{x})$, quantile γ, initial model weight β, and BO budget B.
2: **Output:** Optimized point $\boldsymbol{x}_{inc}$.
3: $\mathcal{D}_1 \leftarrow initialize(\mathcal{X})$
4: **for** $t = 1$ **to** B **do**
5: $\quad \mathcal{M}_g(\boldsymbol{x}) \leftarrow fit\_model\_good(\mathcal{D}_t)$
6: $\quad \mathcal{M}_b(\boldsymbol{x}) \leftarrow fit\_model\_bad(\mathcal{D}_t)$

BOPrO model



BOPrO model

We will be defining hyperparameter configurations ($\mathbf{x}$'s) as 'good' if their observed y-value ($f(\mathbf{x})$) is below a certain quantile γ of the observed function values ($p(y < f_\gamma) = \gamma$).

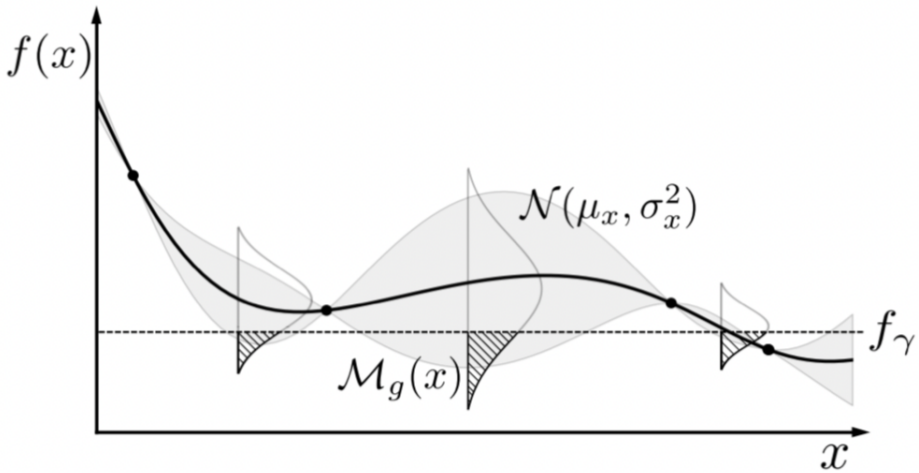
Therefore we want to define the probability $\mathcal{M}_g(\mathbf{x})$ of the fact that at $\mathbf{x}$ the function value lies below a certain quantile. Exploiting the fact that our standard probabilistic model $p(y|\mathbf{x})$ has a Gaussian form, we can define it as follows:

$$\mathcal{M}_g(\mathbf{x}) = p(f(\mathbf{x}) < f_\gamma | \mathbf{x}, \mathcal{D}_t) = \Phi\left(\frac{f_\gamma - \mu_\mathbf{x}}{\sigma_\mathbf{x}}\right),$$

where $\mathcal{D}_t = (\mathbf{x}_t, y_i)_{i=1}^t$ are the evaluated configurations, $\mu_\mathbf{x}$ and $\sigma_\mathbf{x}$ are the predictive mean and standard deviation of the probabilistic model at $\mathbf{x}$, and Φ is the standard normal distribution function. Analogously, we define $\mathcal{M}_b(\mathbf{x})$ as a probability of $\mathbf{x}$ being 'bad' by

$$\mathcal{M}_b(\mathbf{x}) = 1 - \mathcal{M}_g(\mathbf{x}).$$

BOPrO model visualized



BOPrO pseudo-posterior

Algorithm 1 BOPrO Algorithm. $\mathcal{D}_t$ keeps track of all function evaluations so far: $(\boldsymbol{x}_i, y_i)_{i=1}^t$.

Input: Input space $\mathcal{X}$, user-defined prior distribution $P_g(\boldsymbol{x})$, quantile γ, initial model weight β, and BO budget B.
Output: Optimized point $\boldsymbol{x}_{inc}$.
$\mathcal{D}_1 \leftarrow initialize(\mathcal{X})$
for $t = 1$ **to** B **do**
    $\mathcal{M}_g(\boldsymbol{x}) \leftarrow fit\_model\_good(\mathcal{D}_t)$
    $\mathcal{M}_b(\boldsymbol{x}) \leftarrow fit\_model\_bad(\mathcal{D}_t)$
    $g(\boldsymbol{x}) \leftarrow P_g(\boldsymbol{x}) \cdot \mathcal{M}_g(\boldsymbol{x})^{\frac{t}{\beta}}$
    $b(\boldsymbol{x}) \leftarrow P_b(\boldsymbol{x}) \cdot \mathcal{M}_b(\boldsymbol{x})^{\frac{t}{\beta}}$

BOPrO pseudo-posterior

Having the prior $P_g(\mathbf{x})$ and the model $\mathcal{M}_g(\mathbf{x})$ BOPrO constructs a pseudo-posterior on 'good' points. It represents the updated beliefs on where we can find 'good' points based on the prior and data that has been observed. It is computed as

$$g(\mathbf{x}) \propto P_g(\mathbf{x}) \mathcal{M}_g(\mathbf{x})^{\frac{t}{\beta}},$$

where t is the current optimization iteration, β is an optimization hyperparameter with both $P_g(\mathbf{x})$ and $\mathcal{M}_g(\mathbf{x})$ rescaled to $[0, 1]$ using min-max scaling. We refer to it as 'pseudo-posterior' because it's not normalized (it's not a standard posterior probability distribution), but it doesn't affect it's ability to determine the next $\mathbf{x}_t$ - we will soon find out why.

BOPrO pseudo-posterior

Analogously, we also want to represent our updated beliefs on where we can find 'bad' points so we compute:

$$b(\mathbf{x}) \propto P_b(\mathbf{x})\mathcal{M}_b(\mathbf{x})^{\frac{t}{\beta}}.$$

We are now ready to define $p(\mathbf{x}|y)$ which we will use in the acquisition function. We compute it as

$$p(\mathbf{x}|y) \propto \begin{cases} g(\mathbf{x}), y < f_\gamma \\ b(\mathbf{x}), y \geq f_\gamma. \end{cases}$$

Necessary formulations

Algorithm 1 BOPrO Algorithm. $\mathcal{D}_t$ keeps track of all function evaluations so far: $(\boldsymbol{x}_i, y_i)_{i=1}^{t}$.

Input: Input space $\mathcal{X}$, user-defined prior distribution $P_g(\boldsymbol{x})$, quantile γ, initial model weight β, and BO budget B.
Output: Optimized point $\boldsymbol{x}_{inc}$.
$\mathcal{D}_1 \leftarrow initialize(\mathcal{X})$
for $t = 1$ **to** B **do**
    $\mathcal{M}_g(\boldsymbol{x}) \leftarrow fit\_model\_good(\mathcal{D}_t)$
    $\mathcal{M}_b(\boldsymbol{x}) \leftarrow fit\_model\_bad(\mathcal{D}_t)$
    $g(\boldsymbol{x}) \leftarrow P_g(\boldsymbol{x}) \cdot \mathcal{M}_g(\boldsymbol{x})^{\frac{t}{\beta}}$
    $b(\boldsymbol{x}) \leftarrow P_b(\boldsymbol{x}) \cdot \mathcal{M}_b(\boldsymbol{x})^{\frac{t}{\beta}}$
    $\boldsymbol{x}_t \in \arg\max_{\boldsymbol{x} \in \mathcal{X}} EI_{f_\gamma}(\boldsymbol{x})$
    $y_t \leftarrow f(\boldsymbol{x}_t)$
    $\mathcal{D}_{t+1} \leftarrow \mathcal{D}_t \cup (\boldsymbol{x}_t, y_t)$

Necessary formulations

BOPrO uses all of the previously defined terms based on two important propositions:

$$EI_{f_\gamma}(\mathbf{x}) := \int_{-\infty}^{\infty} max(f_\gamma - y, 0)p(y|\mathbf{x})dy = \int_{-\infty}^{f_\gamma} (f_\gamma - y)\frac{p(\mathbf{x}|y)p(y)}{p(\mathbf{x})}dy \propto \left(\gamma + \frac{b(\mathbf{x})}{g(\mathbf{x})}(1-\gamma)\right)^{-1}$$

This statement shows us that in order to find $\mathbf{x}$ that maximizes the acquisition function we just need to find the one that minimizes the ratio $\frac{b(\mathbf{x})}{g(\mathbf{x})}$.

Necessary formulations

$$\lim_{t\to\infty}\left(arg\,max_{\mathbf{x}\in\mathcal{X}}EI_{f_\gamma}(\mathbf{x})\right) = \lim_{t\to\infty}\left(arg\,max_{\mathbf{x}\in\mathcal{X}}\mathcal{M}_g(\mathbf{x})\right)$$

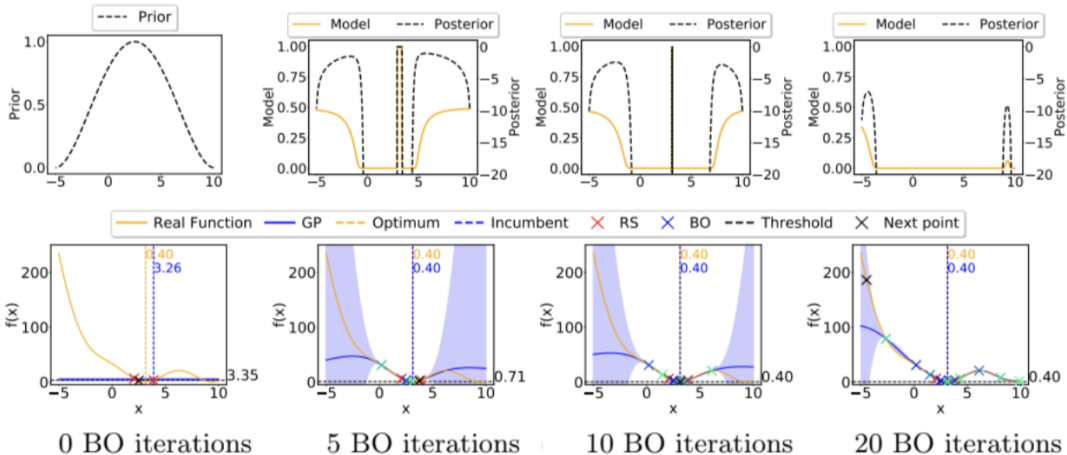
This statement however says that for higher number of iterations BOPrO converges to standard bayesian optimization with Probability of Improvement as its acquisition function. In other words after some time it washes out the given prior and only trusts the model informed by the data.

The algorithm

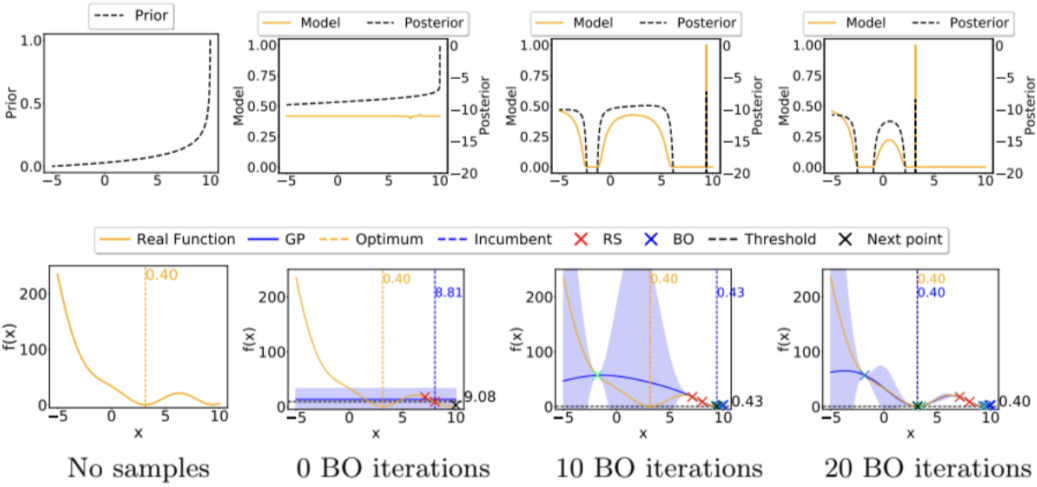
Algorithm 1 BOPrO Algorithm. $\mathcal{D}_t$ keeps track of all function evaluations so far: $(\boldsymbol{x}_i, y_i)_{i=1}^{t}$.

Input: Input space $\mathcal{X}$, user-defined prior distribution $P_g(\boldsymbol{x})$, quantile γ, initial model weight β, and BO budget B.
Output: Optimized point $\boldsymbol{x}_{inc}$.
$\mathcal{D}_1 \leftarrow initialize(\mathcal{X})$
for $t = 1$ **to** B **do**
    $\mathcal{M}_g(\boldsymbol{x}) \leftarrow fit\_model\_good(\mathcal{D}_t)$
    $\mathcal{M}_b(\boldsymbol{x}) \leftarrow fit\_model\_bad(\mathcal{D}_t)$
    $g(\boldsymbol{x}) \leftarrow P_g(\boldsymbol{x}) \cdot \mathcal{M}_g(\boldsymbol{x})^{\frac{t}{\beta}}$
    $b(\boldsymbol{x}) \leftarrow P_b(\boldsymbol{x}) \cdot \mathcal{M}_b(\boldsymbol{x})^{\frac{t}{\beta}}$
    $\boldsymbol{x}_t \in \arg\max_{\boldsymbol{x} \in \mathcal{X}} EI_{f_\gamma}(\boldsymbol{x})$
    $y_t \leftarrow f(\boldsymbol{x}_t)$
    $\mathcal{D}_{t+1} \leftarrow \mathcal{D}_t \cup (\boldsymbol{x}_t, y_t)$
end for
$\boldsymbol{x}_{inc} \leftarrow compute\_best(\mathcal{D}_{t+1})$
return $\boldsymbol{x}_{inc}$

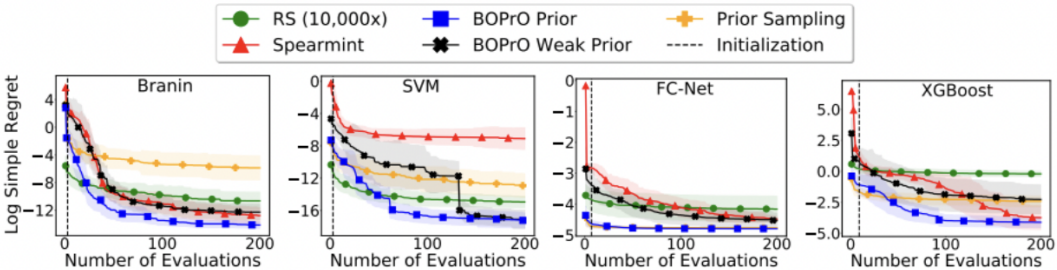
Application example



Forgetting the prior



Comparison with strong baselines



Summary

'So far, BO failed to leverage the experience of domain experts, not only causing inefficiency but also driving users away from adopting BO because they could not exploit their years of knowledge in optimizing their black-box functions. BOPrO addresses this issue and we therefore expect it to facilitate the adoption of BO. We showed that BOPrO is 6.67$\times$ more sample efficient than strong BO baselines, and 10,000$\times$ faster than random search, on a common suite of benchmarks. We also showed that BOPrO converges faster and robustly recovers from misleading priors. In future work, we will study how our approach can be used to leverage prior knowledge from meta-learning. Bringing these two worlds together will likely boost the performance of BO even further.'