Bayesian Optimization of Composite Functions

Raúl Astudillo
Cornell University

Joint work with Peter I. Frazier

INFORMS Annual Meeting, September 21st, 2019
What to expect

Bayesian Optimization of Composite Functions

Raúl Astudillo  ra598@cornell.edu
How it works: an illustration

Suppose

- $x$ is a parameter of a simulator
- $h(x)$ is simulator’s prediction under $x$
- $y$ is our observed data

We want to solve

$$\min_x (h(x) - y)^2.$$
Standard BayesOpt
Figure: Evaluations of $(h(x) - y)^2$
Figure: GP posterior on \((h(x) - y)^2\)
Figure: GP posterior on $(h(x) - y)^2$
Our approach
(a) Evaluations of $h(x) - y$

(b) Evaluations of $(h(x) - y)^2$
(a) GP posterior on $h(x) - y$

(b) Implied posterior on $(h(x) - y)^2$
(a) GP posterior on $h(x) - y$

(b) Implied posterior on $(h(x) - y)^2$
Problem setup

We consider problems of the form

$$\max_{x \in \mathcal{X}} f(x),$$

where

$$f(x) = g(h(x))$$

and

- $h : \mathcal{X} \subset \mathbb{R}^d \rightarrow \mathbb{R}^m$ is a time-consuming-to-evaluate black-box,
- $g : \mathbb{R}^m \rightarrow \mathbb{R}$ and its gradient are known in closed form and fast-to-evaluate.
Composite functions arise naturally in practice.
Hyperparameter tuning of classification algorithms

\[ g(h(x)) = - \sum_{j=1}^{m} h_j(x), \]

where \( h_j(x) \) is the classification error on the \( j \)-th class under hyperparameters \( x \).
Calibration of an oil reservoir simulator

\[ g(h(x)) = - \sum_{j=1}^{m} (h_j(x) - y_j)^2, \]

where \( h(x) \) is the output of the simulator under parameters \( x \) and \( y \) is a vector of observed data.
Optimization of posteriors with expensive likelihoods

\[
\log p(x \mid y) = \log L(y \mid x) + \log \pi(x).
\]

Very often, \( L(y \mid x) \propto g(y \mid h(x)) \), where \( g \) is known in closed form and \( h(x) \) is a vector of parameters governing properties of the data’s distribution.

E.g., for a Gaussian likelihood,

\[
g(y \mid h(x)) \propto -\|h(x) - y\|_2^2.
\]
Related work
BayesOpt for sums of functions

Constrained BayesOpt

BayesOpt for sums of squared errors

The most widely used acquisition function in standard BayesOpt is:

\[ \text{EI}_n(x) = \mathbb{E}_n \left[ \{ f(x) - f^*_n \}^+ \right], \]

where

- \( f^*_n \) is the best observed value so far,
- \( \mathbb{E}_n \) is the conditional expectation under the posterior after \( n \) evaluations.
The most widely used acquisition function in standard BayesOpt is:

$$EI_n(x) = \mathbb{E}_n \left[ \{ f(x) - f_n^* \}^+ \right].$$

When $f(x)$ is Gaussian, EI and its derivative have a closed form which make it easy to optimize.
Our contribution

1. A statistical approach for modeling $f$ that greatly improves over the standard BayesOpt approach.

2. An efficient way to optimize the expected improvement under this new statistical model.
Our approach

• Model $h$ using a multi-output Gaussian process instead of $f$ directly.

• This implies a (non-Gaussian) posterior on $f(x) = g(h(x))$.

• To decide where to sample next: compute and optimize the expected improvement under this new posterior.
Expected Improvement for Composite Functions

Our acquisition function is Expected Improvement for Composite Functions (EI-CF):

$$EI-CF_n(x) = \mathbb{E}_n \left[ \{ g(h(x)) - f^*_n \}^+ \right],$$

where $h$ is a GP, making $h(x)$ Gaussian.
Bayesian Optimization of Composite Functions

Raúl Astudillo ra598@cornell.edu
Challenge: maximizing EI-CF is hard

Expected Improvement for Composite Functions (EI-CF):

$$\text{EI-CF}_n(x) = \mathbb{E}_n \left[ \{ g(h(x)) - f^*_n \}^+ \right].$$

Challenge:

• When $h$ is a GP and $g$ is nonlinear, $f(x) = g(h(x))$ is not Gaussian.

• EI-CF does not have a closed form, making it hard to optimize.
Calculating EI-CF

To estimate $\text{EI-CF}_n(x)$, repeat the following $L$ times:

1. Sample $h(x)$ from the Gaussian process posterior
2. Calculate the improvement $\{g(h(x)) - f^*_n\}^+$

Then average the results.
A naive approach to maximize EI-CF

- **Naive optimization method:** Maximize EI-CF directly, e.g., using a genetic algorithm

- **Problem:** this will be really slow because we don’t have gradients and the evaluations are noisy
1. Reparametrization trick
2. Evaluate using Monte Carlo
3. Optimize using a novel gradient estimator
Reparametrization trick

\[ h(x) \overset{d}{=} \mu_n(x) + C_n(x)Z, \]

where

- \( \mu_n \) and \( K_n \) are the posterior mean and covariance functions of \( h \)
- \( C_n(x) \) is the Cholesky factor of \( K_n(x, x) \)
- \( Z \) is a \( m \)-variate standard normal random vector
Reparametrization trick

\[ h(x) \overset{d}{=} \mu_n(x) + C_n(x)Z, \]

where

- \( \mu_n \) and \( K_n \) are the posterior mean and covariance functions of \( h \)
- \( C_n(x) \) is the Cholesky factor of \( K_n(x, x) \)
- \( Z \) is a \( m \)-variate standard normal random vector

Thus,

\[
\text{EI-CF}_n(x) = \mathbb{E} \left[ \{ g(\mu_n(x) + C_n(z)Z) - f^*_n \}^+ \right].
\]
Evaluate using Monte Carlo

\[
\text{EI-CF}_n(x) \approx \frac{1}{L} \sum_{\ell=1}^{L} \left\{ g(\mu_n(x) + C_n(x)Z^{(\ell)}) - f_n^* \right\}^+,
\]

where \( Z^{(1)}, \ldots, Z^{(L)} \sim \mathcal{N}(0, I_m) \).
Gradient of EI-CF

Lemma.
Under mild regularity conditions, $\text{EI-CF}_n$ is differentiable almost everywhere and its gradient, when it exists, is given by

$$\nabla \text{EI-CF}_n(x) = \mathbb{E}_n [\gamma_n(x, Z)],$$

where

$$\gamma_n(x, Z) = \begin{cases} 0, & \text{if } g(\mu_n(x) + C_n(x)Z) \leq f_n^*; \\ \nabla g(\mu_n(x) + C_n(x)Z), & \text{otherwise}. \end{cases}$$
To get a stochastic gradient, i.e., an unbiased estimate of $\nabla_x \text{EI-CF}_n(x)$:

1. Sample a standard normal random vector $Z$
2. Return $\gamma_n(x, Z)$
Our improved method for maximizing EI-CF

To get a stochastic gradient, i.e., an unbiased estimate of $\nabla_x \text{EI-CF}_n(x)$:

1. Sample a standard normal random vector $Z$
2. Return $\gamma_n(x, Z)$

We use these stochastic gradients within multi-start stochastic gradient ascent to efficiently maximize $\text{EI-CF}_n$. 
When outputs of $h$ are modeled independently, the complexity of exact posterior inference is $O(mn^2)$ (with a precomputation of complexity $O(mn^3)$).

Recent advances on fast approximate GP prediction alleviate this computational burden.
Theorem.

If \( g \) is continuous and under additional suitable regularity conditions, EI-CF is asymptotically consistent, i.e., it finds the true global optimum as the number of evaluations goes to infinity.
Numerical experiments
GP-generated test problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>$\mathcal{X}$</th>
<th>$g$</th>
<th>$m$</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>$[0, 1]^4$</td>
<td>$g(h(x)) = -\sum_{j=1}^{5} (h_j(x) - y_j^*)^2$</td>
<td>5</td>
</tr>
<tr>
<td>b</td>
<td>$[0, 1]^3$</td>
<td>$g(h(x)) = -\sum_{j=1}^{4} \exp(h_j(x))$</td>
<td>4</td>
</tr>
</tbody>
</table>

(a)

(b)
Langermann test problem

\[ f(x) = g(h(x)) \] where

\[ h_j(x) = \sum_{i=1}^{d} (x_i - A_{ij}), \quad j = 1, \ldots, 5, \]

and

\[ g(h(x)) = -\sum_{j=1}^{5} c_j \exp(-h_j(x)/\pi) \cos(\pi h_j(x)). \]
5d Rosenbrock test problem

\[ f(x) = - \sum_{j=1}^{d-1} 100(x_{j+1} - x_j^2)^2 + (x_j - 1)^2 \]

We set \( d = 5 \) and adapt this problem to our framework by taking

\[ h_j(x) = x_{j+1} - x_j^2, \quad j = 1, \ldots, 4, \]

\[ h_{j+4}(x) = x_j - 1, \quad j = 1, \ldots, 4, \]

and

\[ g(h(x)) = - \sum_{j=1}^{4} 100h_j(x)^2 + h_{j+4}(x)^2. \]
5d Rosenbrock test problem

Bayesian Optimization of Composite Functions

Raúl Astudillo ra598@cornell.edu
Models a chemical accident that has caused a pollutant to spill at two locations

Given 12 measurements at different geospatial locations, invert the 4 parameters of this simulator

We solve

$$
\min_{x \in \mathcal{X}} \sum_{j=1}^{12} (s(\theta_j; x^*) - s(\theta_j; x))^2
$$
Environmental model test problem (Bliznyuk et al., 2008)

Bayesian Optimization of Composite Functions

Raúl Astudillo ra598@cornell.edu
Conclusion and future work

- Exploiting composite objective functions can substantially improve the performance of BayesOpt
- Develop efficient implementations of other acquisition functions in this setting
- Some of them would allow noisy and decoupled evaluations
Thanks for your attention

- Check out our code: https://github.com/RaulAstudillo06/BOCF