Bayesian Optimization in effective dimension using kernel based sensitivity indices

Adrien Spagnol$^{1,3}$ – Rodolphe Le Riche$^{2,1}$ – Sébastien Da Veiga$^3$

$^1$ Ecole Nationale Supérieure des Mines de Saint-Etienne, F-42023 Saint-Etienne, France
$^2$ CNRS LIMOS UMR 6158, France
$^3$ Safran Tech, Modelling & Simulation, Rue des Jeunes Bois, Châteaufort, 78114 Magny-Les-Hameaux, France
Summary

1. Motivation

2. Reminder on Bayesian Optimization

3. Online variable selection (using kernel based sensitivity indices)

4. Benchmark and results
Motivation: Design of complex systems

$x \in \mathbb{R}^D \subset X$
Design variables

Black-box function $f$

$Y = f(X)$
Objective function(s)
Constraints

Black-box optimization

✓ no closed-form for $f$, no access to derivatives
✓ $f$ is expensive – limited budget ($2 - 10D$)
✓ $f$ is high dimensional – $D > 50$

Examples at Safran

✓ design of turbine and compressor blades
✓ perforation pattern for combustion chambers
Bayesian Optimization

State-of the art sequential-based optimization method to optimize black-box functions

(Feliot, 2017)

(Chen, 2018)
Bayesian Optimization

State-of-the-art sequential-based optimization method to optimize black-box functions

Bayesian Inference

✓ Probabilistic model over $f$
✓ Given past observations $\mathcal{D} = \{X, y = f(X)\}$, we get a posterior distribution over $f$
✓ Typical choice: Gaussian Process (Rasmussen, 2004)
Bayesian Optimization

State-of the art sequential-based optimization method to optimize black-box functions

Bayesian Inference

✓ Probabilistic model over $f$
✓ Given past observations $\mathcal{D} = \{X, y = f(X)\}$, we get a posterior distribution over $f$
✓ Typical choice: Gaussian Process (Rasmussen, 2004)

Optimization loop

✓ Acquisition function (UCB (Srinivas, 2009), PI (Kushner, 1694), EI (Močkus, 1975)(Jones, 1998))

$$a_{EI}(x) = \mathbb{E}[\max\{f(x) - f^*, 0\}]$$
Bayesian Optimization

State-of the art sequential-based optimization method to optimize black-box functions

Bayesian Inference

- Probabilistic model over $f$
- Given past observations $\mathcal{D} = \{X, y = f(X)\}$, we get a posterior distribution over $f$
- Typical choice: Gaussian Process (Rasmussen, 2004)

Optimization loop

- Acquisition function (UCB (Srinivas, 2009), PI (Kushner, 1694), EI (Močkus, 1975)(Jones, 1998))

$$a^{EI}(x) = \mathbb{E}[\max\{f(x) - f^*, 0\}]$$
Bayesian Optimization

State-of-the-art sequential-based optimization method to optimize black-box functions

Bayesian Inference

✓ Probabilistic model over $f$
✓ Given past observations $\mathcal{D} = \{\mathbf{X}, y = f(\mathbf{X})\}$, we get a posterior distribution over $f$
✓ Typical choice: Gaussian Process (Rasmussen, 2004)

Optimization loop

✓ Acquisition function (UCB (Srinivas, 2009), PI (Kushner, 1694), EI (Močkus, 1975)(Jones, 1998))

$$a^{EI}(x) = \mathbb{E}[\max\{f(x) - f^*, 0\}]$$
Bayesian Optimization

State-of the art sequential-based optimization method to optimize black-box functions

Bayesian Inference

✓ Probabilistic model over $f$
✓ Given past observations $\mathcal{D} = \{\mathbf{X}, y = f(\mathbf{X})\}$, we get a posterior distribution over $f$
✓ Typical choice: Gaussian Process (Rasmussen, 2004)

Optimization loop

✓ Acquisition function (UCB (Srinivas, 2009), PI (Kushner, 1694), EI (Močkus, 1975)(Jones, 1998))
$$a^{EI}(x) = \mathbb{E}[\max\{f(x) - f^*, 0\}]$$
Bayesian Optimization

State-of the art sequential-based optimization method to optimize black-box functions

Bayesian Inference

✓ Probabilistic model over \( f \)
✓ Given past observations \( \mathcal{D} = \{ \mathbf{X}, \mathbf{y} = f(\mathbf{X}) \} \), we get a posterior distribution over \( f \)
✓ Typical choice: Gaussian Process (Rasmussen, 2004)

Optimization loop

✓ Acquisition function (UCB (Srinivas, 2009), PI (Kushner, 1694), EI (Močkus, 1975)(Jones, 1998))

\[
a^{EI}(x) = \mathbb{E}\left[ \max\{f(x) - f^*, 0\} \right]
\]
Bayesian Optimization

Limitations

✓ the acquisition function is mostly flat with several local minima, making the optimization complex in high dimension

✓ the estimation of the hyperparameters of the surrogate model is troublesome as the dimension grows

✓ a higher dimension implies a bigger search space $\mathcal{X}$, harder to cover (curse of dimensionality)
Bayesian Optimization for high dimensional problems

Two ways to overcome precedent limitations:

1. Simplify the structure of the surrogate models
   \[ f(x) = f_0 + f_1(x_1) + \cdots \]
   - ACOSSO (Storlie, 2011)
   - Additive models for optimization (ADD-GP-UCB, Kandasamy, 2015)

2. Assume a small effective dimension \( d_e \ll D \)
   - Offline sensitivity analysis (Shan, 2010) (Spagnol, 2018)
### Bayesian Optimization for high dimensional problems

Two ways to overcome precedent limitations:

1. **Simplify the structure of the surrogate models**
   
   \[ f(x) = f_0 + f_1(x_1) + \cdots \]

   - ACOSSO (Storlie, 2011)
   - Additive models for optimization (ADD-GP-UCB, Kandasamy, 2015)

2. **Assume a small effective dimension** \( d_e \ll D \)

   - Offline sensitivity analysis (Shan, 2010) (Spagnol, 2018)
   - REMBO (Wang, 2013)
   - Split-and-Doubt (Salem, 2018)
   - DropOut (Li, 2018)
DropOut (Li, 2018)

Instead of working on all variables, at each iteration

✓ randomly draw \( d < D \) variables
✓ optimize the acquisition function on those variables only

Several strategies to fix the other variables \( x^{D-d} \)

✓ **Rand**: randomly sample \( x^{D-d} \sim \mathcal{U}(x^{D-d}) \)
✓ **Copy**: use the observation value giving the best response so far
✓ **Mix**: mixture of methods above, one chosen with probability \( p \) and the other with probability \( 1 - p \)

---

**Algorithm 1** Dropout Algorithm for High-dimensional Bayesian Optimization

```
Input: \( D_1 = \{x_0, y_0\} \)
for \( t = 1, 2, \ldots \) do
    randomly select \( d \) dimensions
    \( x_t^d \leftarrow \arg \max_{x_t^d \in \mathcal{X}^d} a(x_t^d | D_t) \)
    \( x_t^{D-d} \leftarrow \) one of three fill-in strategies
    \( x_t \leftarrow x_t^d \cup x_t^{D-d} \)
    \( y_t \leftarrow \) Query \( y_t \) at \( x_t \)
end for
```
DropOut (Li, 2018)

Instead of working on all variables, at each iteration:

✓ randomly draw $d < D$ variables

✓ Copy: use the observation value giving the best response so far

✓ Mix: mixture of methods above, one chosen with probability $p$ and the other with probability $1 - p$

We will replace this step by an online variable selection using global sensitivity analysis.

Algorithm 1 Dropout Algorithm for High-dimensional Bayesian Optimization

<table>
<thead>
<tr>
<th>Input: $D_1 = {x_0, y_0}$ for $t = 1, 2, \cdots$ do</th>
</tr>
</thead>
<tbody>
<tr>
<td>randomly select $d$ dimensions</td>
</tr>
<tr>
<td>$x_t^d \leftarrow \arg\max_{x_t^d \in X^d} a(x_t^d</td>
</tr>
<tr>
<td>$x_t^{D-d} \leftarrow$ one of three fill-in strategies</td>
</tr>
<tr>
<td>$x_t \leftarrow x_t^d \cup x_t^{D-d}$</td>
</tr>
<tr>
<td>$y_t \leftarrow$ Query $y_t$ at $x_t$</td>
</tr>
<tr>
<td>end for</td>
</tr>
</tbody>
</table>

![Graph showing the comparison of different algorithms with D=20 and D=30]
Goal-oriented sensitivity analysis

In our case, we are interested in the relevance of a given input to reach a given level of performance $\mathcal{L}$. 
Goal-oriented sensitivity analysis

In our case, we are interested in the relevance of a given input to reach a given level of performance $\mathcal{L}$. 

low values of the objective function
Goal-oriented sensitivity analysis

In our case, we are interested in the relevance of a given input to reach a given level of performance $\mathcal{L}$.

Imagine we sample uniformly $X_1$ and $X_2$ at low values of the objective function.
Goal-oriented sensitivity analysis

In our case, we are interested in the relevance of a given input to reach a given level of performance $\mathcal{L}$.

Imagine we sample uniformly $X_1$ and $X_2$. Low values of the objective function...
Goal-oriented sensitivity analysis

In our case, we are interested in the relevance of a given input to reach a given level of performance $\mathcal{L}$.

Imagine we sample uniformly $X_1$ and $X_2$.
Goal-oriented sensitivity analysis

In our case, we are interested in the relevance of a given input to reach a given level of performance $\mathcal{L}$.

Measuring influence is done through comparison between $P_{X_1}$ and $P_{X_1 | X \in \mathcal{L}}$.

- Small influence of $X_1$
- Strong influence of $X_2$
Comparing probability distributions

- **Kolmogorov-Smirnov distance**
  → Regional Sensitivity Analysis (Hornberger, 1975) or Monte Carlo Filtering (Saltelli, 2004)
Comparing probability distributions

- **Kolmogorov-Smirnov distance**
  → Regional Sensitivity Analysis (Hornberger, 1975) or Monte Carlo Filtering (Saltelli, 2004)

- **Maximum mean discrepancy**
  → based on the representation of probability distributions in a Reproducing Kernel Hilbert Space (Gretton, 2012)
Comparing probability distributions

\[ \mu_{\mathbb{P}_{X_i}} \text{ is the representer of } \mathbb{P}_{X_i} \text{ in the RKHS } \mathcal{H}_k \]
Comparing probability distributions

\[ \mu_{\mathbb{P}_{X_i}} \text{ is the representer of } \mathbb{P}_{X_i} \text{ in the RKHS } \mathcal{H}_k \]

\[ \mathbb{M}_1^+ \]

\[ \mathcal{H}_k \]
Comparing probability distributions with MMD

Representer of a distribution $\mathbb{P}$ is the kernel mean embedding

$$\mu_\mathbb{P} = \int k(x, \cdot) d\mathbb{P}(x)$$

with $k$ a symmetric and positive definite kernel function. $\mu_\mathbb{P}$ characterizes the probability distribution in the RKHS $\mathcal{H}_k$ with kernel $k$. 
Comparing probability distributions with MMD

Representer of a distribution $\mathbb{P}$ is the kernel mean embedding

$$\mu_{\mathbb{P}} = \int k(x, \cdot) d\mathbb{P}(x)$$

with $k$ a symmetric and positive definite kernel function. $\mu_{\mathbb{P}}$ characterizes the probability distribution in the RKHS $\mathcal{H}_k$ with kernel $k$.

Distance between embeddings is the Maximum Mean Discrepancy (MMD) (Gretton, 2012)

$$\text{MMD}^2(\mathbb{P}_{X_i}, \mathbb{P}_{X_i|X \in \mathcal{L}}) = \left\| \mu_{\mathbb{P}_{X_i}} - \mu_{\mathbb{P}_{X_i|X \in \mathcal{L}}} \right\|_{\mathcal{H}_k}^2 = \int k(x, x') \left( p_{X_i|X \in \mathcal{L}}(x) - p_{X_i}(x) \right) \left( p_{X_i|X \in \mathcal{L}}(x') - p_{X_i}(x') \right) dx dx'$$

Under assumptions ($k$ characteristic), if $\mu_{\mathbb{P}_{X_i}} = \mu_{\mathbb{P}_{X_i|X \in \mathcal{L}}}$, then $\mathbb{P}_{X_i} = \mathbb{P}_{X_i|X \in \mathcal{L}}$ exactly.
Comparing probability distributions with MMD

Estimation with a sample from $X_i$

$$X_i = \begin{pmatrix} \vdots \end{pmatrix}$$
Comparing probability distributions with MMD

Estimation with a sample from $X_i$

$$X_i = \left( \begin{array}{c} \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \end{array} \right)$$

Which $X_i | X \in \mathcal{L}$

$$X_i | X \in \mathcal{L} = \left( \begin{array}{c} \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \end{array} \right) = \tilde{X}_i$$
Comparing probability distributions with MMD

Estimation with a sample from $X_i$

$$X_i = \begin{pmatrix} \vdots \end{pmatrix}$$

Which $X_i | X \in \mathcal{L}$

$$X_i | X \in \mathcal{L} = \begin{pmatrix} \vdots \end{pmatrix} = \tilde{X}_i$$

Unbiased estimator (Gretton, 2012)

$$\text{MMD}^2_u \left( \mathbb{P}_{X_i}, \mathbb{P}_{X_i | X \in \mathcal{L}} \right) = \frac{1}{N(N-1)} \sum_{p=1}^{N} \sum_{q \neq p}^{N} k(X_i^p, X_i^q) + \frac{1}{M(M-1)} \sum_{p=1}^{M} \sum_{q \neq p}^{M} k(\tilde{X}_i^p, \tilde{X}_i^q) - \frac{2}{NM} \sum_{p=1}^{N} \sum_{q=1}^{M} k(X_i^p, \tilde{X}_i^q)$$
Plugging MMD based indices in the DropOut

In our case, we define $\mathcal{L} = \{x \in \mathcal{X}, f(x) \leq q_{\alpha}\}$, with $q_{\alpha}$ the $\alpha\%$-quantile. One can easily extend this to constrained problems.

We define

$$s^{MMD}(X_i) = \text{MMD}^2_u(\mathbb{P}_{X_i}, \mathbb{P}_{X_i|X \in \mathcal{L}})$$
Plugging MMD based indices in the DropOut

In our case, we define \( \mathcal{L} = \{ x \in \mathcal{X}, f(x) \leq q_\alpha \} \), with \( q_\alpha \) the \( \alpha \%-\)quantile. One can easily extend this to constrained problems.

We define

\[
S^{\text{MMD}}(X_i) = \text{MMD}^2_u\left( \mathbb{P}_{X_i}, \mathbb{P}_{\mathcal{X} \mid X \in \mathcal{L}} \right)
\]

To ensure \( S^{\text{MMD}}(X_i) \in [0,1] \), we normalize our index

\[
S^{\text{MMD}}_N(X_i) = \frac{S^{\text{MMD}}(X_i)}{\sum_i S^{\text{MMD}}(X_i)}
\]

When plugged in the Dropout algorithm:

\[
\hat{s}^{\text{MMD}}(X_i) = \frac{\text{MMD}^2_u\left( \mathbb{P}_{X_i}, \mathbb{P}_{\mathcal{X} \mid X \in \hat{\mathcal{L}}} \right)}{\sum_i \hat{s}^{\text{MMD}}(X_i)}, \text{with } \hat{\mathcal{L}} = \{ x \in \mathcal{X}, \mu(x) \leq \hat{q}_\alpha \}
\]
Bayesian Optimization with sensitivity analysis step

<table>
<thead>
<tr>
<th>Algorithm 1</th>
<th>Bayesian optimization with Dropout guided by MMD based SA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> ( {X, y = f(X)} )</td>
<td></td>
</tr>
<tr>
<td><strong>while</strong> ( N &lt; N_{\text{max}} ) <strong>do</strong></td>
<td></td>
</tr>
<tr>
<td>Construction of the surrogate model ( \mu(x) ) and ( \sigma^2(x) )</td>
<td></td>
</tr>
<tr>
<td><strong>for</strong> ( i = 1, ..., D ) <strong>do</strong></td>
<td></td>
</tr>
<tr>
<td>Calculate ( \hat{S}^{\text{MMD}}(X_i) ) on the predictive mean ( \mu(x) )</td>
<td></td>
</tr>
<tr>
<td><strong>end for</strong></td>
<td></td>
</tr>
<tr>
<td>( d_{|N} \leftarrow \text{deterministic or probabilistic variable selection} )</td>
<td></td>
</tr>
<tr>
<td>( x_{|N} \leftarrow \text{fill-in strategies} )</td>
<td></td>
</tr>
<tr>
<td>( x_{|N} \leftarrow \text{arg max}<em>{x</em>{|N} \in x_{|N}} \alpha \left(x_{|N} \mid x_{|N} \right) )</td>
<td></td>
</tr>
<tr>
<td>( x_{N+1} \leftarrow x_{|N} \cup x_{|N} )</td>
<td></td>
</tr>
<tr>
<td>( y_{N+1} = f(x_{N+1}) )</td>
<td></td>
</tr>
<tr>
<td>( N \leftarrow N + 1 )</td>
<td></td>
</tr>
<tr>
<td><strong>end while</strong></td>
<td></td>
</tr>
</tbody>
</table>

- **Deterministic selection:** keep every variables with \( \hat{S}^{\text{MMD}}(X_i) \geq \tau \)
- **Probabilistic selection:** \( d \) draws with \( \mathbb{P}(x = d_i) = \hat{S}^{\text{MMD}}(X_i) \)

For the deterministic, we use the heuristic \( \tau = \frac{1}{D} \).

For the probabilistic, we use \( d = 5 \) as in (Li, 2018).
Various features for each function: many local minima, unimodality, nonlinear model, multiple global minima etc.

For most of the functions, \( D - d \) dummy variables are added to increase the dimensionality of the problem.

For each problem, the optimization is repeated 20 times using a different initial DoE.
Various features for each function: many local minima, unimodality, nonlinear model, multiple global minima etc.

For most of the functions, $D - d$ dummy variables are added to increase the dimensionality of the problem.

For each problem, the optimization is repeated 20 times using a different initial DoE. Using all optimization runs, we define easy, medium and hard minimization targets as 90, 50 and 10% quantiles of our results.
Performance of the optimizers is measured as the number of successful runs at each iteration

\[
\text{Prob. of Success} = \frac{\text{#runs reaching a level}}{N_{\text{runs}}}
\]

In the following results, we only present the optimizers using the \textit{mix} fill-in strategy as it gives the best results:

\begin{itemize}
  \item \textbf{Mix}: randomly sample \( x^{D-d} \sim \mathcal{U}(x^{D-d}) \), with probability \( p \) or use the observation value giving the best response so far, with probability \( 1 - p \)
\end{itemize}
Results on an optimization benchmark

Easy

- Prob + Mix
- Deter + Mix
- DropOut + Mix
- EGO
Results on an optimization benchmark

Dimension reduction allows visible gains

Easy

Medium

Hard

Prob. of success

Iteration #

Prob. of success

Iteration #

Prob. of success

Iteration #

Prob. + Mix

Deter + Mix

DropOut + Mix

EGO
Results on an optimization benchmark

**Easy**

**Medium**

**Hard**

Best method mix a deterministic selection with a probabilistic fill-in strategy
Conclusion and perspectives

✓ We proposed a method to improve Bayesian optimization through dimension reduction based on two strategies using kernel based sensitivity indices to efficiently select variables at each iteration:

✓ Both methods allow visible gains on a small benchmark of high dimensional functions

✓ Dummy variables in the problems are correctly detected as non-influent

✓ Improvements:

✓ Possible improvements are to design a third strategy which does not depend on a hard-coded parameter (based on an independence test for example)

✓ A last upgrade would be changing the way $\mathcal{L}$ is defined
Thank you for attention!

Do you have any questions?


Ce document et les informations qu’il contient sont la propriété de Safran. Ils ne doivent pas être copiés ni communiqués à un tiers sans l’autorisation préalable et écrite de Safran.
## Functions definition

<table>
<thead>
<tr>
<th>Name</th>
<th>d</th>
<th>D</th>
<th>Main features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branin</td>
<td>2</td>
<td>25</td>
<td>Multiple global minima</td>
</tr>
<tr>
<td>Ackley</td>
<td>6</td>
<td>20</td>
<td>Many local minima</td>
</tr>
<tr>
<td>Borehole</td>
<td>8</td>
<td>25</td>
<td>Simple problem</td>
</tr>
<tr>
<td>Rosenbrock</td>
<td>5</td>
<td>20</td>
<td>Unimodal, hard convergence</td>
</tr>
<tr>
<td>Stybtang</td>
<td>20</td>
<td>20</td>
<td>Multimodal problem</td>
</tr>
</tbody>
</table>
Variable selection

- Probabilistic
- Deterministic
Definition of the easy, medium and hard threshold, an example
The MMD between $\mathbb{P}_{X_i}$ and $\mathbb{P}_{X_i|X \in \mathcal{L}}$ is actually proportionate to a kernel based independence criterion, the Hilbert-Schmidt independence criterion.

Assuming $Z = 1_{X \in \mathcal{L}}$ with 1 the indicator function such that

$$Z = \begin{cases} 
1 & \text{if } X \in \mathcal{L} \\
0 & \text{else}
\end{cases}$$

if we use a categorical kernel adapted to discrete inputs, i.e. $l(z, z') = zz'$, then we can define

$$\text{HSIC}(X_i, Z) = \mathbb{P}(Z = 1)^2 \mathbb{E} \left( k(x, x') \left( p_{X_i|Z=z}(x) - p_{X_i}(x) \right) \left( p_{X_i|Z=z}(x') - p_{X_i}(x') \right) \right)$$

$$\text{HSIC}(X_i, Z) = \mathbb{P}(Z = 1)^2 \mathbb{E} \left( k(x, x') \left( p_{X_i|Z=z}(x) - p_{X_i}(x) \right) \left( p_{X_i|Z=z}(x') - p_{X_i}(x') \right) \right) = \mathbb{P}(Z = 1)^2 \text{MMD}^2(\mathbb{P}_{X_i}, \mathbb{P}_{X_i|X \in \mathcal{L}})$$