Bayesian Optimization and Its Applications

Jungtaek Kim

jtkim@postech.ac.kr

POSTECH
Pohang 37673, Republic of Korea
https://jungtaek.github.io

January 31, 2022

Table of Contents

Bayesian Optimization

Mathematical Optimization

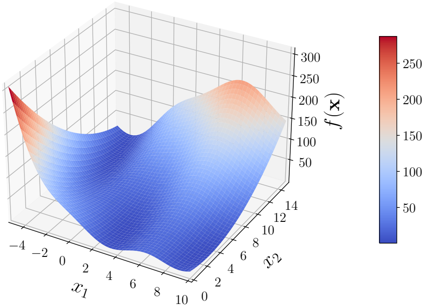


Figure 1: Branin function.

- Given an objective $f : \mathcal{A} \to \mathbb{R}$ where $\mathcal{A}$ is some set, it seeks minimum or maximum of the target function:

$$\mathbf{x}^* = \arg\min f(\mathbf{x}), \quad (1)$$

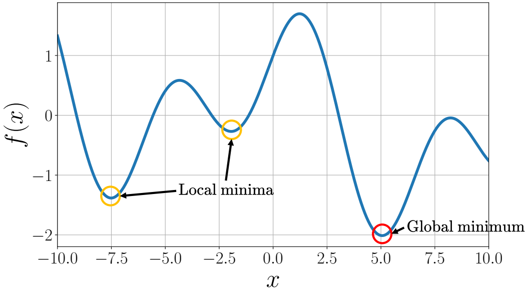
or

$$\mathbf{x}^* = \arg\max f(\mathbf{x}). \quad (2)$$

Mathematical Optimization

- To optimize an objective, we can select one of such strategies:
  - random searches;
  - gradient-based approaches;
  - convex programming;
  - evolutionary algorithms;
  - simulated annealing.
- Each strategy has the advantage in the corresponding conditions of optimization problem.
- However, under certain circumstances, Bayesian optimization is the most effective method to solve some class of mathematical optimization problems.

Global Optimization



- Global optimization solves a problem to find a global minimizer $\mathbf{x}^\star$:

$$\mathbf{x}^\star = \arg\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}), \tag{3}$$

where $\mathcal{X} \subset \mathbb{R}^d$ is a compact search space.

Black-Box Optimization

Definition 1 (Black-box function)

If an objective f, defined in (3), satisfies the following statements, we call it as a black-box function:

(i) a function f is unknown, but evaluations of f are available;

(ii) a gradient ∇f and Hessian matrix $\nabla^2 f$ are also unknown;

(iii) the condition that f is Lipschitz continuous is known;

(iv) moreover, differentiability and continuity of f are unknown,

on a compact search space $\mathcal{X}$.

Black-Box Optimization

- According to recent work [Hansen et al., 2010, Turner et al., 2020], we can apply some classes of possible candidates:
  - random search [Bergstra and Bengio, 2012];
  - evolutionary strategies [Hansen, 2006, 2016];
  - Lipschitzian optimization method without the Lipschitz constant [Jones et al., 1993, Jones and Martins, 2021];
  - Bayesian optimization [Kushner, 1964, Močkus, 1975];
  - sequential model-based optimization with tree-based surrogates [Hutter et al., 2011].
- Unfortunately, there is no rule of thumb for choosing the best approach to solving a certain objective without directly conducting the method on the optimization problem.

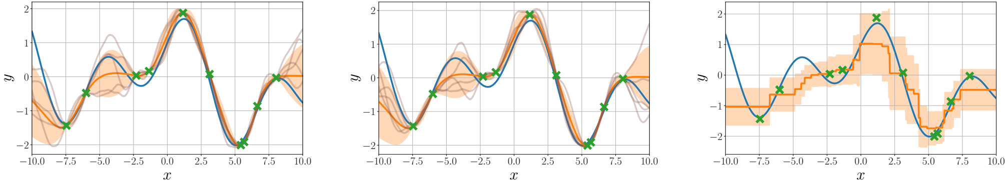
Bayesian Optimization

- Bayesian optimization is a promising method to find a global optimizer of black-box objective function.
- Evaluation of the objective is only available.
- Since we do not know a target function, it optimizes an acquisition function, instead of the target function.
- An acquisition function is defined with factors for exploiting available information up to current iteration and exploring an unexplored region.

Surrogate Models

- A surrogate model estimates a true objective function, where historical evaluations are given.
- To balance a trade-off between exploration and exploitation, it predicts a function estimate and its uncertainty estimate over any query $\mathbf{x} \in \mathcal{X}$.
- Gaussian process regression [Rasmussen and Williams, 2006], Student-t process regression [Martinez-Cantin et al., 2018], random forest regression [Hutter et al., 2011], and Bayesian neural network [Springenberg et al., 2016] have been used.

Surrogate Models



(a) Gaussian process (b) Student-t process (c) Random forest

Figure 2: Examples of surrogate models.

Gaussian Process

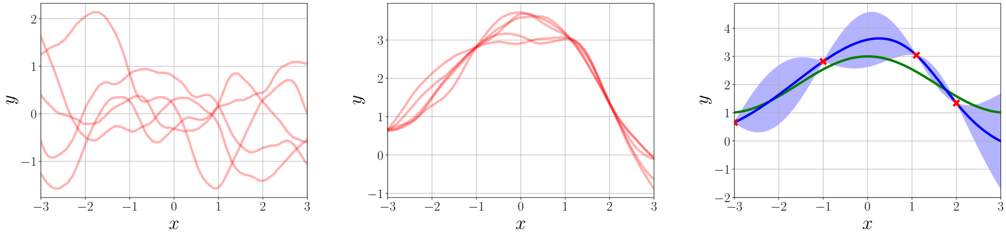
- A collection of random variables, any finite number of which have a joint Gaussian distribution [Rasmussen and Williams, 2006].
- Generally, a Gaussian process is defined as

$$f \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')), \tag{4}$$

where

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})], \tag{5}$$
$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]. \tag{6}$$

Gaussian Process Regression



(a) From prior function dist. (b) From posterior function dist. (c) Predictive dist.

Figure 3: Gaussian process regression for a function $\cos(x) + 2$ with an observation noise.

Gaussian Process Regression

- One of popular covariance functions, the exponentiated quadratic covariance function in one dimension is defined as

$$k\left(x, x'\right) = s^2 \exp\left(-\frac{1}{2l^2}\left(x - x'\right)^2\right) + \sigma_n^2 \delta_{xx'}, \tag{7}$$

where s is a signal scale, l is a length scale and σ_n^2 is a noise variance [Rasmussen and Williams, 2006].

- Posterior mean function $\mu(\mathbf{x}^*; \mathbf{X}, \mathbf{y})$ and variance function $\sigma^2(\mathbf{x}^*; \mathbf{X}, \mathbf{y})$:

$$\mu(\mathbf{x}^*; \mathbf{X}, \mathbf{y}) = \mathbf{k}(\mathbf{x}^*, \mathbf{X})(\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \boldsymbol{I})^{-1}\mathbf{y}, \tag{8}$$

$$\sigma^2(\mathbf{x}^*; \mathbf{X}, \mathbf{y}) = k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}(\mathbf{x}^*, \mathbf{X})(\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \boldsymbol{I})^{-1}\mathbf{k}(\mathbf{X}, \mathbf{x}^*), \tag{9}$$

where $\mathbf{X} \in \mathbb{R}^{n \times d}$ and $\mathbf{y} \in \mathbb{R}^n$.

Gaussian Process Regression

- If non-zero mean prior is given, posterior mean and variance functions:

$$\mu(\mathbf{x}^*; \mathbf{X}, \mathbf{y}) = \mathbf{k}(\mathbf{x}^*, \mathbf{X})(\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \boldsymbol{I})^{-1}(\mathbf{y} - \boldsymbol{\mu}_p(\mathbf{X})) + \mu_p(\mathbf{x}^*), \tag{10}$$

$$\sigma^2(\mathbf{x}^*; \mathbf{X}, \mathbf{y}) = k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}(\mathbf{x}^*, \mathbf{X})(\mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \boldsymbol{I})^{-1}\mathbf{k}(\mathbf{X}, \mathbf{x}^*), \tag{11}$$

where μ_p is a prior mean function, and $\boldsymbol{\mu}_p(\mathbf{X}) = [\mu_p(\mathbf{x}_1), \ldots, \mu_p(\mathbf{x}_n)]$.

Student-t Process Regression

- If non-zero mean prior is given, posterior mean and variance functions:

$$\mu(\mathbf{x}^*; \mathbf{X}, \mathbf{y}) = \mathbf{k}(\mathbf{x}^*, \mathbf{X})\tilde{\mathbf{K}}^{-1}\tilde{\mathbf{y}} + \mu_p(\mathbf{x}^*), \quad (12)$$

$$\sigma^2(\mathbf{x}^*; \mathbf{X}, \mathbf{y}) = \frac{\nu + \tilde{\mathbf{y}}^\top \tilde{\mathbf{K}}^{-1}\tilde{\mathbf{y}} - 2}{\nu + n - 2} \left(k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}(\mathbf{x}^*, \mathbf{X})\tilde{\mathbf{K}}^{-1}\mathbf{k}(\mathbf{X}, \mathbf{x}^*) \right), \quad (13)$$

  where μ_p is a prior mean function, $\boldsymbol{\mu}_p(\mathbf{X}) = [\mu_p(\mathbf{x}_1), \ldots, \mu_p(\mathbf{x}_n)]$, $\tilde{\mathbf{y}} = \mathbf{y} - \boldsymbol{\mu}_p(\mathbf{X})$, and $\tilde{\mathbf{K}} = \mathbf{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \boldsymbol{I}$.

- The parameter ν for the posterior distribution is set to $\nu + n$.

Random Forest Regression

- Posterior mean and variance functions:

$$\begin{aligned} \mu\left(\mathbf{x}^*; \{\mathcal{T}_b\}_{b=1}^B, \mathbf{X}, \mathbf{y}\right) &= \frac{1}{B} \sum_{b=1}^{B} \mu_b(\mathbf{x}^*) \\ &= \frac{1}{B} \sum_{b=1}^{B} \sum_{\tau \in \boldsymbol{\tau}_{b,l}} \mu_\tau 1_{\mathbf{x}^* \in \tau}, \end{aligned} \tag{14}$$

$$\begin{aligned} \sigma^2\left(\mathbf{x}^*; \{\mathcal{T}_b\}_{b=1}^B, \mathbf{X}, \mathbf{y}\right) &= \frac{1}{B} \sum_{b=1}^{B} \left(\sigma_b^2(\mathbf{x}^*) + \mu_b^2(\mathbf{x}^*)\right) - \mu\left(\mathbf{x}^*; \{\mathcal{T}_b\}_{b=1}^B, \mathbf{X}, \mathbf{y}\right)^2 \\ &= \frac{1}{B} \sum_{b=1}^{B} \left(\left(\sum_{\tau \in \boldsymbol{\tau}_{b,l}} \sigma_\tau 1_{\mathbf{x}^* \in \tau}\right)^2 + \left(\sum_{\tau \in \boldsymbol{\tau}_{b,l}} \mu_\tau 1_{\mathbf{x}^* \in \tau}\right)^2\right) \\ &\quad - \left(\frac{1}{B} \sum_{b=1}^{B} \mu_b(\mathbf{x}^*)\right)^2. \end{aligned} \tag{15}$$

Acquisition Functions

- An acquisition function acquires the next sample to evaluate by a black-box function f.
- As a popular choice of acquisition functions, the following acquisition functions:
  - probability of improvement (PI) [Kushner, 1964];
  - expected improvement (EI) [Močkus et al., 1978];
  - Gaussian process upper confidence bound (GP-UCB) [Srinivas et al., 2010],

  have been suggested.

Acquisition Functions

- Diverse acquisition functions:
  - knowledge gradient [Frazier et al., 2009];
  - entropy search [Hennig and Schuler, 2012];
  - predictive entropy search [Hernández-Lobato et al., 2014];
  - clustering-guided Gaussian process upper confidence bound (CG-GPUCB) [Kim and Choi, 2018b];
  - portfolio allocation of various acquisition functions [Hoffman et al., 2011];
  - alternatives of expected improvement by tree-structured Parzen estimator [Bergstra et al., 2011] and class-probability estimation [Tiao et al., 2021],

  have been also proposed.

[Kim and Choi, 2018] **J. Kim** and S. Choi. Clustering-guided GP-UCB for Bayesian optimization. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2461–2465, Calgary, Alberta, Canada, 2018b.

Popular Acquisition Functions (Minimization Case)

- Suppose that

$$(\mathbf{x}^\dagger, y^\dagger) = \operatorname*{arg\,min}_{(\mathbf{x}, y) \in \mathcal{D}_{t-1}} y, \tag{16}$$

$$\mu(\mathbf{x}; \mathbf{X}, \mathbf{y}) = \mu(\mathbf{x}; \mathcal{D}_{t-1}), \tag{17}$$

$$\sigma(\mathbf{x}; \mathbf{X}, \mathbf{y}) = \sigma(\mathbf{x}; \mathcal{D}_{t-1}). \tag{18}$$

- PI criterion [Kushner, 1964] is defined as

$$a_{\mathrm{PI}}(\mathbf{x} \mid \mathcal{D}_{t-1}) = \begin{cases} \Phi\left(\frac{y^\dagger - \mu(\mathbf{x}; \mathcal{D}_{t-1})}{\sigma(\mathbf{x}; \mathcal{D}_{t-1})}\right) & \text{if } \sigma^2(\mathbf{x}; \mathcal{D}_{t-1}) > 0, \\ 0 & \text{otherwise}, \end{cases} \tag{19}$$

where Φ is a cumulative distribution function of the standard normal distribution.

Popular Acquisition Functions (Minimization Case)

- EI criterion [Močkus et al., 1978] is defined as

$$a_{\mathrm{EI}}(\mathbf{x} \mid \mathcal{D}_{t-1}) = \begin{cases} \sigma(\mathbf{x}; \mathcal{D}_{t-1}) \left(z(\mathbf{x}) \Phi \left(z(\mathbf{x}) \right) + \phi \left(z(\mathbf{x}) \right) \right) & \text{if } \sigma^2(\mathbf{x}; \mathcal{D}_{t-1}) > 0, \\ 0 & \text{otherwise}, \end{cases} \tag{20}$$

where $z(\mathbf{x}) = \frac{y^\dagger - \mu(\mathbf{x}; \mathcal{D}_{t-1})}{\sigma(\mathbf{x}; \mathcal{D}_{t-1})}$, Φ is a cumulative distribution function of the standard normal distribution, and ϕ is a probability density function of the standard normal distribution.

- GP-UCB criterion [Srinivas et al., 2010] is defined as

$$a_{\mathrm{UCB}}(\mathbf{x} \mid \mathcal{D}_{t-1}) = -\mu(\mathbf{x}; \mathcal{D}_{t-1}) + \beta_t \sigma(\mathbf{x}; \mathcal{D}_{t-1}), \tag{21}$$

where β_t is a trade-off hyperparameter at iteration t.

Acquisition Function Optimization

- We should find a global optimizer of acquisition function.
- But, in practice, either local optimizer or multi-started local optimizer can be a good option as a substitute of global optimizer.
- Analyses on these selections are provided in [Kim and Choi, 2020].

[Kim and Choi, 2020] J. Kim and S. Choi. On local optimizers of acquisition functions in Bayesian optimization. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, pages 675–690, Virtual, 2020.

On Local Optimizers of Acquisition Functions in Bayesian Optimization

Theorem 2 (Instantaneous regret difference between global and local optimizers)

Given $\delta_l \in [0, 1)$ *and* $\epsilon_l, \epsilon_1, \epsilon_2 > 0$*, the regret difference for a local optimizer* $\mathbf{x}_{t,l}$ *at iteration* t*,* $|r_{t,g} - r_{t,l}|$ *is less than* ϵ_l *with a probability at least* $1 - \delta_l$*:*

$$\mathbb{P}\big(|r_{t,g} - r_{t,l}| < \epsilon_l \big) \geq 1 - \delta_l, \tag{22}$$

where $\delta_l = \frac{\gamma}{\epsilon_1}(1 - \beta_g) + \frac{M}{\epsilon_2}$*,* $\epsilon_l = \epsilon_1 \epsilon_2$*,* $\gamma = \max_{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}} \|\mathbf{x}_i - \mathbf{x}_j\|_2$ *is the size of* $\mathcal{X}$*,* β_g *is the probability that a local optimizer of the acquisition function collapses with its global optimizer, and* M *is the Lipschitz constant.*

On Local Optimizers of Acquisition Functions in Bayesian Optimization

Theorem 3 (Instantaneous regret difference between global and multi-started local optimizers)

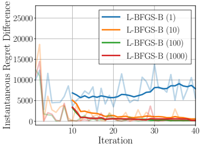
Given $\delta_m \in [0, 1)$ and $\epsilon_m, \epsilon_2, \epsilon_3 > 0$, a regret difference for a multi-started local optimizer $\mathbf{x}_{t,m}$, determined by starting from N initial points at iteration t, is less than ϵ_m with a probability at least $1 - \delta_m$:

$$\mathbb{P}\big(|r_{t,g} - r_{t,m}| < \epsilon_m \big) \geq 1 - \delta_m, \tag{23}$$

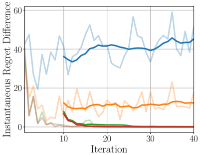
where $\delta_m = \frac{\gamma}{\epsilon_3} (1 - \beta_g)^N + \frac{M}{\epsilon_2}$, $\epsilon_m = \epsilon_2 \epsilon_3$, $\gamma = \max_{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}} \|\mathbf{x}_i - \mathbf{x}_j\|_2$ is the size of $\mathcal{X}$, β_g is the probability that a local optimizer of the acquisition function collapses with its global optimizer, and M is the Lipschitz constant.

- By following our intuition, this bound is tighter than the bound provided in Theorem 2.

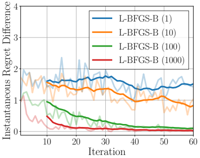
On Local Optimizers of Acquisition Functions in Bayesian Optimization



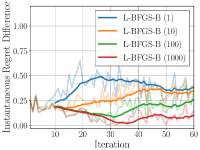
(a) Beale



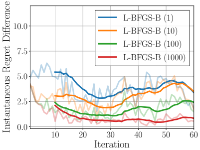
(b) Branin



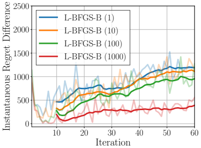
(c) Cosines (8 dim.)



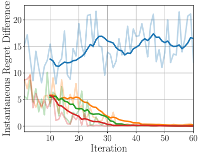
(d) Hartmann6D



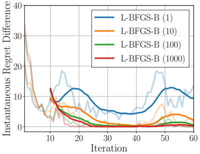
(e) Holdertable



(f) Rosenbrock



(g) Six-Hump Camel



(h) Sphere

Figure 4: Empirical results on Theorems 2 and 3.

On Local Optimizers of Acquisition Functions in Bayesian Optimization

Table 1: Time (sec.) consumed in optimizing acquisition functions.

	Beale	**Branin**	**Cosines (8 dim.)**	**Hart-mann6D**	**Holder-table**	**Rosen-brock**	**Six-Hump Camel**	**Sphere**
DIRECT	3.434	2.987	2.508	0.728	2.935	13.928	4.639	10.707
L-BFGS-B (1)	0.010	0.004	0.023	0.026	0.017	0.005	0.010	0.030
L-BFGS-B (10)	0.096	0.036	0.224	0.253	0.177	0.050	0.100	0.311
L-BFGS-B (100)	0.977	0.363	2.224	2.533	1.760	0.504	0.969	3.048
L-BFGS-B (1000)	9.720	3.633	22.306	25.305	17.629	5.049	9.682	30.764

- Multi-started local optimizer provides a more efficient approach than global optimizer, in terms of computational complexities.

Overall Procedure of Bayesian Optimization

Algorithm 1 Overall Procedure of Bayesian Optimization

Input: A domain of interest $\mathcal{X} \subset \mathbb{R}^d$, an initial set of data $\mathcal{D}_0$, an evaluation budget T, and a true unknown objective f.

Output: The best optimizer found until T, $\mathbf{x}_{\text{best}}$.

for $t = 1, \ldots, T$ **do**

  Construct a surrogate model $\hat{f}(\mathbf{x}; \mathcal{D}_{t-1})$.

  Choose the next point to evaluate by maximizing an acquisition function, defined with $\hat{f}$: $\mathbf{x}_t = \arg\max_{\mathbf{x} \in \mathcal{X}} a(\mathbf{x} \mid \mathcal{D}_{t-1})$.

  Evaluate $\mathbf{x}_t$ by f: $y_t = f(\mathbf{x}_t) + \epsilon_t$, where ϵ_t is observation noise.

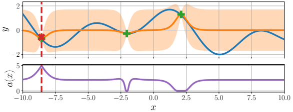
  Append $(\mathbf{x}_t, y_t)$ to $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\mathbf{x}_t, y_t)\}$.

end for

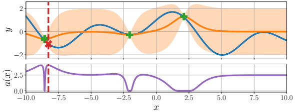
Determine the best optimizer found until T: $\mathbf{x}_{\text{best}} = \arg\min_{(\mathbf{x}, y) \in \mathcal{D}_T} y$.

return $\mathbf{x}_{\text{best}}$

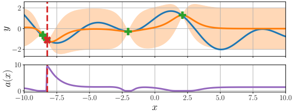
Bayesian Optimization Results with PI



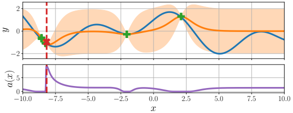
(a) Iteration 1



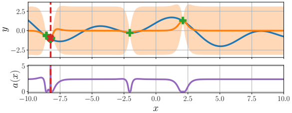
(b) Iteration 2



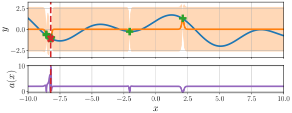
(c) Iteration 3



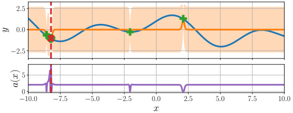
(d) Iteration 4



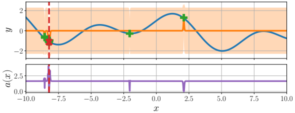
(e) Iteration 5



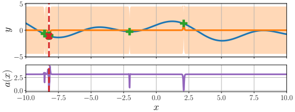
(f) Iteration 6



(g) Iteration 7



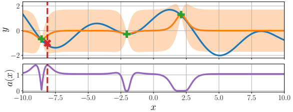
(h) Iteration 8



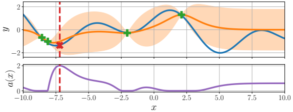
(i) Iteration 9

Figure 5: Bayesian optimization results with PI criterion.

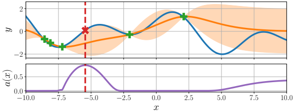
Bayesian Optimization Results with EI



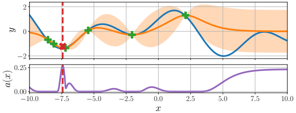
(a) Iteration 1



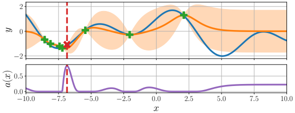
(b) Iteration 2



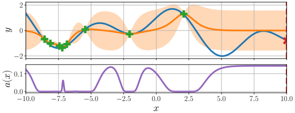
(c) Iteration 3



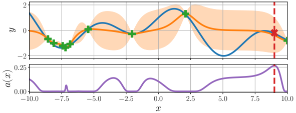
(d) Iteration 4



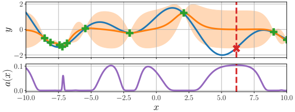
(e) Iteration 5



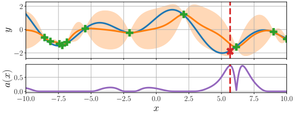
(f) Iteration 6



(g) Iteration 7



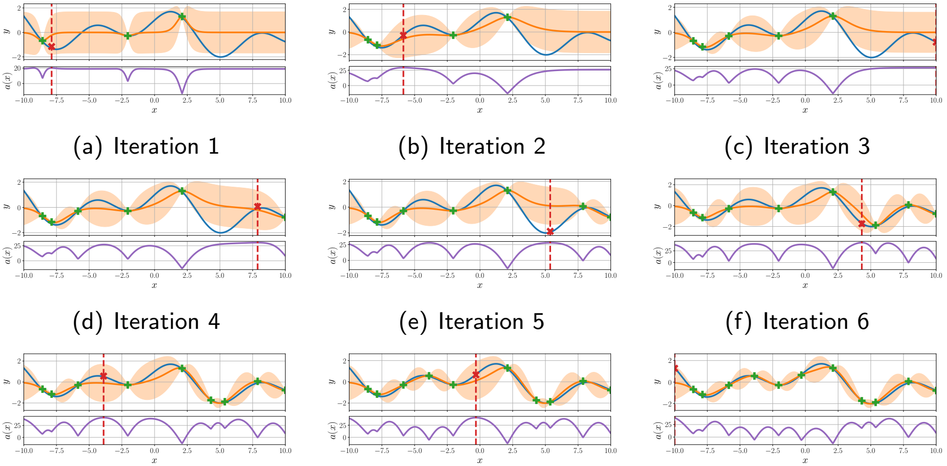
(h) Iteration 8



(i) Iteration 9

Figure 6: Bayesian optimization results with EI criterion.

Bayesian Optimization Results with GP-UCB



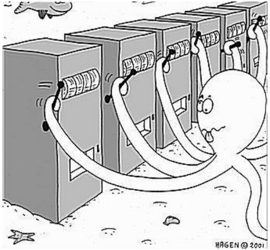
(a) Iteration 1 (b) Iteration 2 (c) Iteration 3

(d) Iteration 4 (e) Iteration 5 (f) Iteration 6

(g) Iteration 7 (h) Iteration 8 (i) Iteration 9

Figure 7: Bayesian optimization results with GP-UCB criterion.

Relationship to Multi-Armed Bandit Problem



- Each machine returns a reward $\hat{r}_a \sim p_{\boldsymbol{\theta}_a}(r_a)$ where $a \in \{1, \ldots, K\}$.
- It minimizes a cumulative regret $T\mu^* - \sum_{t=1}^{T} \hat{r}_{a_t}$ where $\mu^* = \max_{a \in \{1,\ldots,K\}} \mu_a$.
- Bayesian optimization can be considered as infinite bandits with dependent arms.

Relationship to Thompson Sampling

- Thompson sampling is usually applied in multi-armed bandit problems.
- For the case of a beta-Bernoulli bandit, Thompson sampling is defined as follows.

Algorithm 2 Thompson Sampling for a Beta-Bernoulli Bandit

```
for t = 1, 2, ..., T do
    for k = 1, ..., K do
        Sample θ̂_k ~ beta(α_k, β_k).
    end for
    x_t ← argmax_k θ̂_k.
    Apply x_t and observe r_t.
    (α_{x_t}, β_{x_t}) ← (α_{x_t} + r_t, β_{x_t} + 1 − r_t).
end for
```

- After sampling the possibilities, it chooses a maximizer of those sampled values.

BayesO [Kim and Choi, 2017]

- Current version: 0.5.2
- Supported Python version: 3.6, 3.7, 3.8, 3.9
- Web page: `https://bayeso.org`
- GitHub repository: `https://github.com/jungtaekkim/bayeso`
- Documentation: `https://bayeso.readthedocs.io`
- License: MIT license

[Kim and Choi, 2017] J. Kim and S. Choi. BayesO: A Bayesian optimization framework in Python. `https://bayeso.org`, 2017.

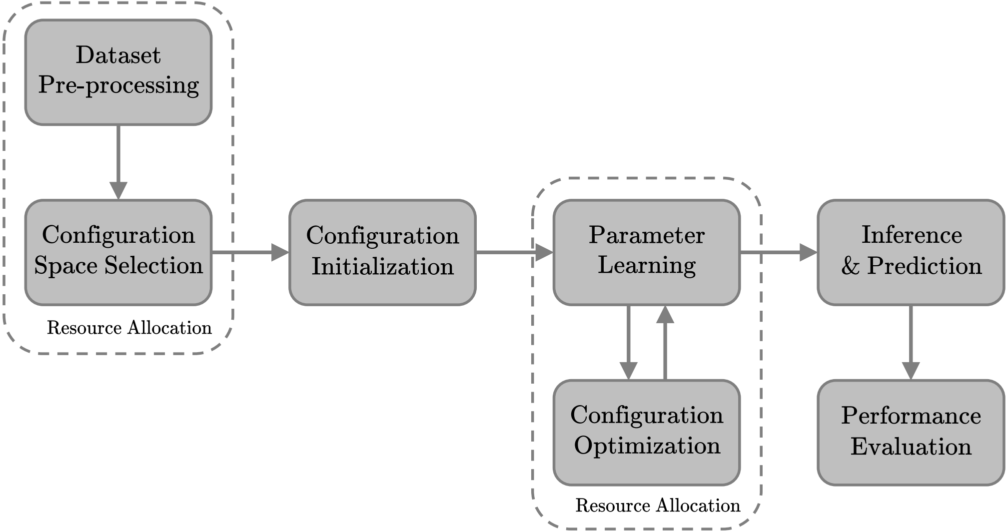
Applications of Bayesian Optimization

Automated Machine Learning

- Automated machine learning is a framework to automatically find an optimal machine learning model without human intervention [Guyon et al., 2015, Hutter et al., 2019].

- Using training and validation datasets, $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{valid}}$, the automated machine learning system finds the optimal algorithm $\mathbf{A}^*$ and the optimal hyperparameters $\boldsymbol{\lambda}^*$:
$$\mathbf{A}^*, \boldsymbol{\lambda}^* = \text{AutoML}(\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{valid}}, \mathcal{A}, \Lambda), \tag{24}$$
where $\mathcal{A}$ is a search space for algorithm selection and Λ is a search space for hyperparameter optimization.

Automated Machine Learning



Automated Machine Learning

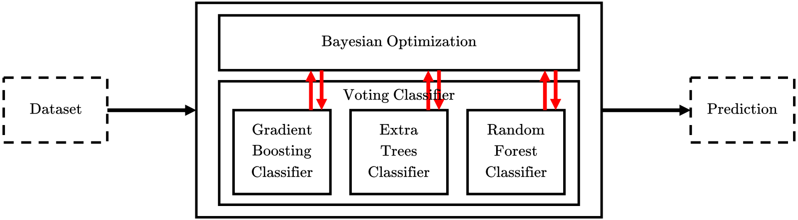


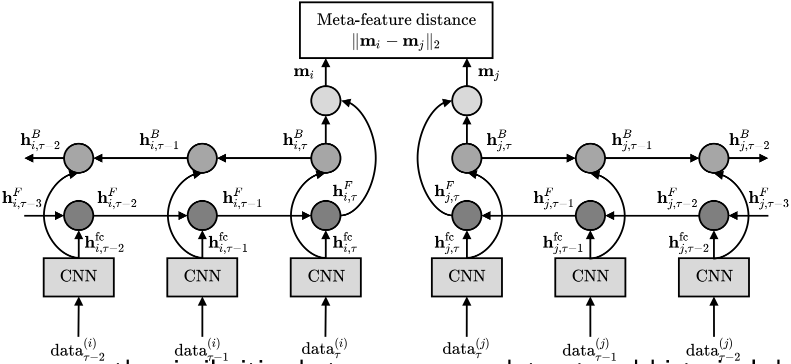
Figure 8: Our automated machine learning system for AutoML Challenge 2018.

- Approaches that take the 3rd place in AutoML5 phase of AutoML Challenge [Kim et al., 2016] and the 2nd place in AutoML Challenge 2018 [Kim and Choi, 2018a] have been presented.

[Kim et al., 2016] **J. Kim**, J. Jeong, and S. Choi. AutoML Challenge: AutoML framework using random space partitioning optimizer. *ICML Workshop on Automatic Machine Learning (AutoML)*, New York, New York, USA, 2016.

[Kim and Choi, 2018a] **J. Kim** and S. Choi. Automated machine learning for soft voting in an ensemble of tree-based classifiers. *ICML Workshop on Automatic Machine Learning (AutoML)*, Stockholm, Sweden, 2018a.

Learning to Transfer Initializations for Bayesian Hyperparameter Optimization [Kim et al., 2017]



- It can measure the similarities between unseen dataset and historical datasets by learning to warm-start Bayesian hyperparameter optimization.

[Kim et al., 2017] **J. Kim**, S. Kim, and S. Choi. Learning to transfer initializations for Bayesian hyperparameter optimization. *NeurIPS Workshop on Bayesian Optimization (BayesOpt)*, Long Beach, California, USA, 2017.

Combinatorial 3D Shape Generation via Sequential Assembly

- 3D shape generation via sequential assembly mimics a human assembly process, by allocating a budget of primitives given [Kim et al., 2020].
- We solve a sequential problem with Bayesian optimization-based framework of combinatorial 3D shape generation, composed of a set of geometric primitives.
- To determine the position of the next primitive, two evaluation functions regarding occupiability and stability are defined.
- Occupiability encourages us to follow a target shape and stability helps to create a physically-stable combination.
- A new combinatorial 3D shape dataset that consists of 14 classes and 406 instances is also introduced in this work.

[Kim et al., 2020] **J. Kim**, H. Chung, J. Lee, M. Cho, and J. Park. Combinatorial 3D shape generation via sequential assembly. *NeurIPS Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng)*, Virtual, 2020.

Experimental Results

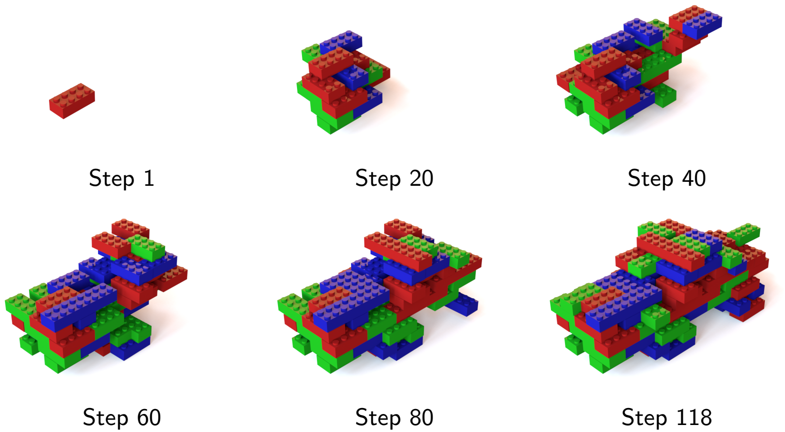
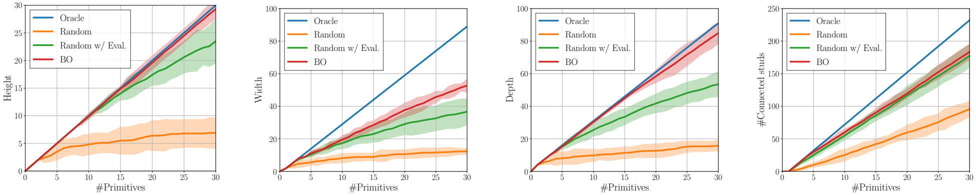


Figure 9: Generated assembling sequence that creates a *car* shape with 118 unit primitives.

Experimental Results

- We apply our framework in optimizing specific explicit functions.



(a) Height (b) Width (c) Depth (d) #Conn. studs

Figure 10: Quantitative results on maximizing explicit evaluation functions.

Combinatorial 3D Shape Dataset

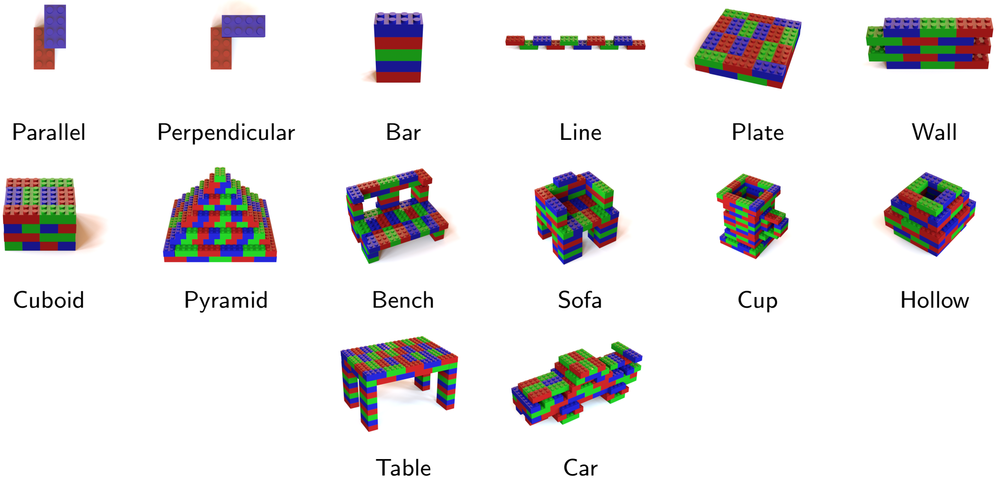


Figure 11: Selected examples from our dataset.

Related Work on Combinatorial and Sequential Assembly

- By following the problem formulation of combinatorial 3D construction and sequential assembly, Thompson et al. [2020] suggest a deep generative model for graphs to construct a 3D object with LEGO bricks.
- Unlike [Kim et al., 2020, Thompson et al., 2020], Lee et al. [2020] solve a 2D jigsaw puzzle with randomly-partitioned fragments via an approach to assembling the fragments sequentially.
- Chung et al. [2021] propose a deep reinforcement learning-based method to assemble 2×4 LEGO bricks, where the incomplete information of a target object, i.e., 2D images, is given to construct the target object.

[Kim et al., 2020] **J. Kim**, H. Chung, J. Lee, M. Cho, and J. Park. Combinatorial 3D shape generation via sequential assembly. *NeurIPS Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng)*, Virtual, 2020.

[Lee et al., 2020] J. Lee*, **J. Kim***, H. Chung, J. Park, and M. Cho. Fragment relation networks for geometric shape assembly. *NeurIPS Workshop on Learning Meets Combinatorial Algorithms (LMCA)*, Virtual, 2020.

[Chung et al., 2021] H. Chung*, **J. Kim***, B. Knyazev, J. Lee, G. W. Taylor, J. Park, and M. Cho. Brick-by-Brick: Combinatorial construction with deep reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, Virtual, 2021.

Takeaway

- Bayesian optimization is a powerful method to optimize a black-box function.
- Instead of methods based on heuristic or prior knowledge, it provides a structured approach to finding an optimal solution.
- Bayesian optimization is expanding into various real-world applications.

Thank you!

References I

J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.

J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 24, pages 2546–2554, Granada, Spain, 2011.

H. Chung, J. Kim, B. Knyazev, J. Lee, G. W. Taylor, J. Park, and M. Cho. Brick-by-Brick: Combinatorial construction with deep reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, Virtual, 2021.

P. I. Frazier, W. B. Powell, and S. Dayanik. The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing*, 21(4): 599–613, 2009.

I. Guyon, K. Bennett, G. Cawley, H. J. Escalante, S. Escalera, T. K. Ho, N. Macià, B. Ray, M. Saeed, A. Statnikov, and E. Viegas. Design of the 2015 ChaLearn AutoML Challenge. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Killarney, Ireland, 2015.

N. Hansen. The CMA evolution strategy: a comparing review. *Towards a new evolutionary computation*, pages 75–102, 2006.

N. Hansen. The CMA evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

N. Hansen, A. Auger, R. Ros, S. Finck, and P. Pošík. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In *Proceedings of the Annual Conference on Genetic and Evolutionary Computation (GECCO)*, pages 1689–1696, Portland, Oregon, USA, 2010.

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

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

M. Hoffman, E. Brochu, and N. de Freitas. Portfolio allocation for Bayesian optimization. In *Proceedings of the Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 327–336, Barcelona, Spain, 2011.

F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Proceedings of the International Conference on Learning and Intelligent Optimization (LION)*, pages 507–523, Rome, Italy, 2011.

F. Hutter, L. Kotthoff, and J. Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

D. R. Jones and J. R. R. A. Martins. The DIRECT algorithm: 25 years later. *Journal of Global Optimization*, 79(3):521–566, 2021.

D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.

References II

J. Kim and S. Choi. BayesO: A Bayesian optimization framework in Python. `https://bayeso.org`, 2017.

J. Kim and S. Choi. Automated machine learning for soft voting in an ensemble of tree-based classifiers. In *International Conference on Machine Learning Workshop on Automatic Machine Learning (AutoML)*, Stockholm, Sweden, 2018a.

J. Kim and S. Choi. Clustering-guided GP-UCB for Bayesian optimization. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2461–2465, Calgary, Alberta, Canada, 2018b.

J. Kim and S. Choi. On local optimizers of acquisition functions in Bayesian optimization. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, pages 675–690, Virtual, 2020.

J. Kim, J. Jeong, and S. Choi. AutoML Challenge: AutoML framework using random space partitioning optimizer. In *International Conference on Machine Learning Workshop on Automatic Machine Learning (AutoML)*, New York, New York, USA, 2016.

J. Kim, S. Kim, and S. Choi. Learning to transfer initializations for Bayesian hyperparameter optimization. In *Neural Information Processing Systems Workshop on Bayesian Optimization (BayesOpt)*, Long Beach, California, USA, 2017.

J. Kim, H. Chung, J. Lee, M. Cho, and J. Park. Combinatorial 3D shape generation via sequential assembly. In *Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng)*, Virtual, 2020.

H. J. Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86(1):97–106, 1964.

J. Lee, J. Kim, H. Chung, J. Park, and M. Cho. Fragment relation networks for geometric shape assembly. In *Neural Information Processing Systems Workshop on Learning Meets Combinatorial Algorithms (LMCA)*, Virtual, 2020.

R. Martinez-Cantin, K. Tee, and M. McCourt. Practical Bayesian optimization in the presence of outliers. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1722–1731, Lanzarote, Canary Islands, Spain, 2018.

J. Močkus. On Bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404, Novosibirsk, Russia, 1975.

J. Močkus, V. Tiesis, and A. Žilinskas. The application of Bayesian methods for seeking the extremum. *Towards Global Optimization*, 2:117–129, 1978.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

References III

J. T. Springenberg, A. Klein, S. Falkner, and F. Hutter. Bayesian optimization with robust Bayesian neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, pages 4134–4142, Barcelona, Spain, 2016.

N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1015–1022, Haifa, Israel, 2010.

R. Thompson, G. Elahe, T. DeVries, and G. W. Taylor. Building LEGO using deep generative models of graphs. In *Neural Information Processing Systems Workshop on Machine Learning for Engineering Modeling, Simulation, and Design (ML4Eng)*, Virtual, 2020.

L. C. Tiao, A. Klein, M. Seeger, E. V. Bonilla, C. Archambeau, and F. Ramos. BORE: Bayesian optimization by density-ratio estimation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 10289–10300, Virtual, 2021.

R. Turner, D. Eriksson, M. McCourt, J. Kiili, E. Laaksonen, Z. Xu, and I. Guyon. Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. In *Proceedings of the NeurIPS 2020 Competition and Demonstration Track*, pages 3–26, Virtual, 2020.