AISTATS 2022

On Uncertainty Estimation by Tree-based Surrogate Models in Sequential Model-based Optimization

Jungtaek Kim (POSTECH) Seungjin Choi (BARO AI)

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Introduction

- GP regression is a popular choice as a surrogate model, because of its capability of calculating uncertainties analytically.
- On the other hand, an ensemble of randomized trees is another option and has practical merits over GPs due to its scalability and easiness of handling continuous/discrete mixed variables.
- We revisit various ensembles of randomized trees to investigate their behavior in the perspective of prediction uncertainty estimation.
- Then, we propose our own tree-based model, referred to as *BwO forest*.

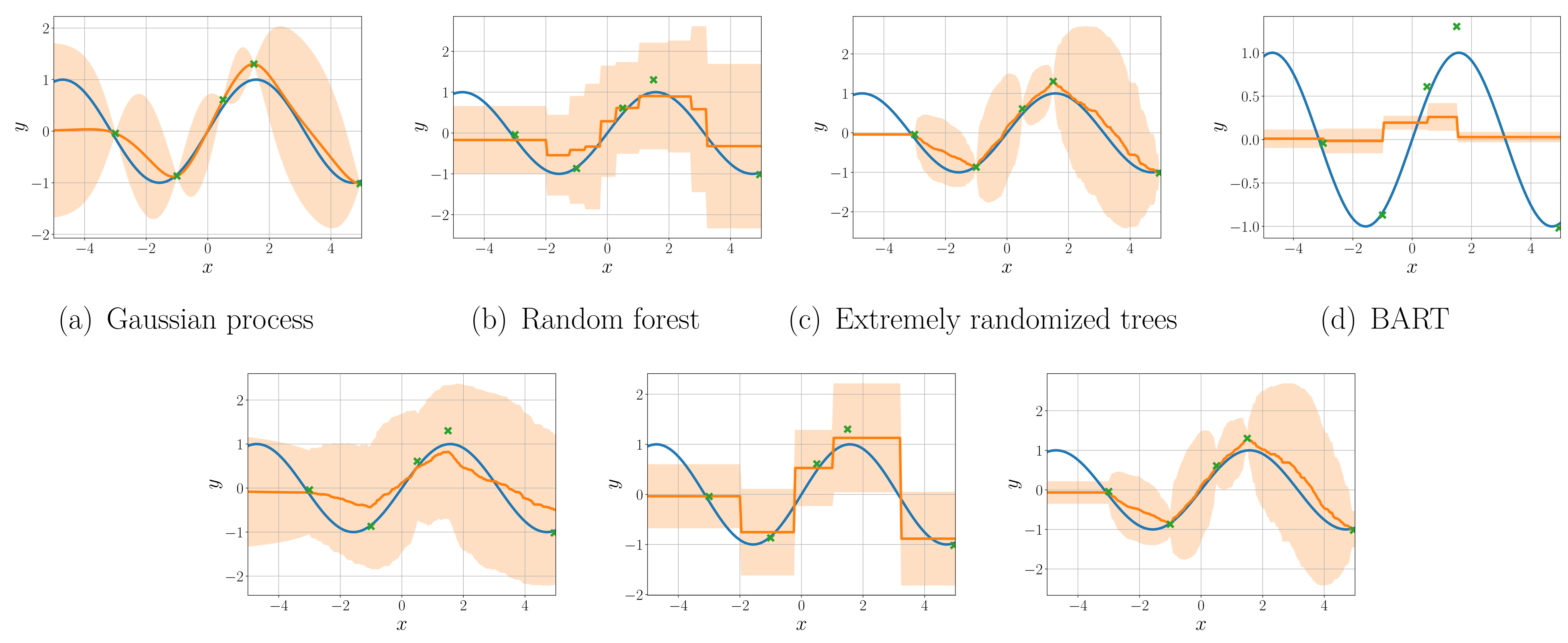
Uncertainty Estimation by Tree-based Models

- **Sum-of-Trees Model**: Posterior mean and variance functions are

$$\mu\big(\mathbf{x}; \{\mathcal{T}_b\}_{b=1}^B, \mathbf{X}, \mathbf{y}\big) = \frac{1}{B} \sum_{b=1}^{B} \sum_{\tau \in \boldsymbol{\tau}_{b,l}} \mu_\tau 1_{\mathbf{x} \in \tau},$$

$$\sigma^2\big(\mathbf{x}; \{\mathcal{T}_b\}_{b=1}^B, \mathbf{X}, \mathbf{y}\big) = \frac{1}{B} \sum_{b=1}^{B} \left(\Big(\sum_{\tau \in \boldsymbol{\tau}_{b,l}} \sigma_\tau 1_{\mathbf{x} \in \tau} \Big)^2 + \Big(\sum_{\tau \in \boldsymbol{\tau}_{b,l}} \mu_\tau 1_{\mathbf{x} \in \tau} \Big)^2 \right) - \mu\big(\mathbf{x}; \{\mathcal{T}_b\}_{b=1}^B, \mathbf{X}, \mathbf{y}\big)^2.$$

- **Gradient Boosting Models**: It updates parameters $\boldsymbol{\theta}$ using their gradients in terms of the objective of parametric dist., e.g., likelihood function.



(a) Gaussian process (b) Random forest (c) Extremely randomized trees (d) BART (e) Mondrian forest (f) NGBoost (g) BwO forest

Figure 1: Results with GP regression and tree-based models such as random forest, extremely randomized trees, BART, Mondrian forest, NGBoost, and BwO forest (ours).

Tree Construction

- The uncertainty of an ensemble is derived from the randomness of individual trees:

(i) **bagging**: it samples a bootstrap sample with replacement and then aggregates base estimators;

(ii) **random feature selection**: this technique randomly selects a feature from a set of dimensions;

(iii) **random split location**: it randomly selects a split location between lower and upper bounds of the selected dimension;

(iv) **random tree sampling**: this strategy randomly samples a tree under the assumption on a prior distribution.

- Random forest employs (i) and (ii), BART employs (i), (ii), and (iv), and Mondrian forest employs (i) and (iii).

Elaborating Uncertainty Estimation

- The expectation and variance of an indicator for the existence of $\mathbf{x}_i$ in a bootstrap sample $\mathbf{B}$ are expressed as

$$\mathbb{E}[1_{\mathbf{x}_i \in \mathbf{B}}] = 1 - \left(1 - \frac{1}{N}\right)^M,$$

$$\mathrm{Var}[1_{\mathbf{x}_i \in \mathbf{B}}] = \left(1 - \frac{1}{N}\right)^M - \left(1 - \frac{1}{N}\right)^{2M},$$

where N is the size of $\mathbf{X}$ and M is the size of a bootstrap sample.

- The distribution of unique original elements in a bootstrap sample $\mathbf{B}$ is:

$$\mathbb{E}[|\mathrm{unique}(\mathbf{B})|] = N - \frac{(N-1)^M}{N^{M-1}},$$

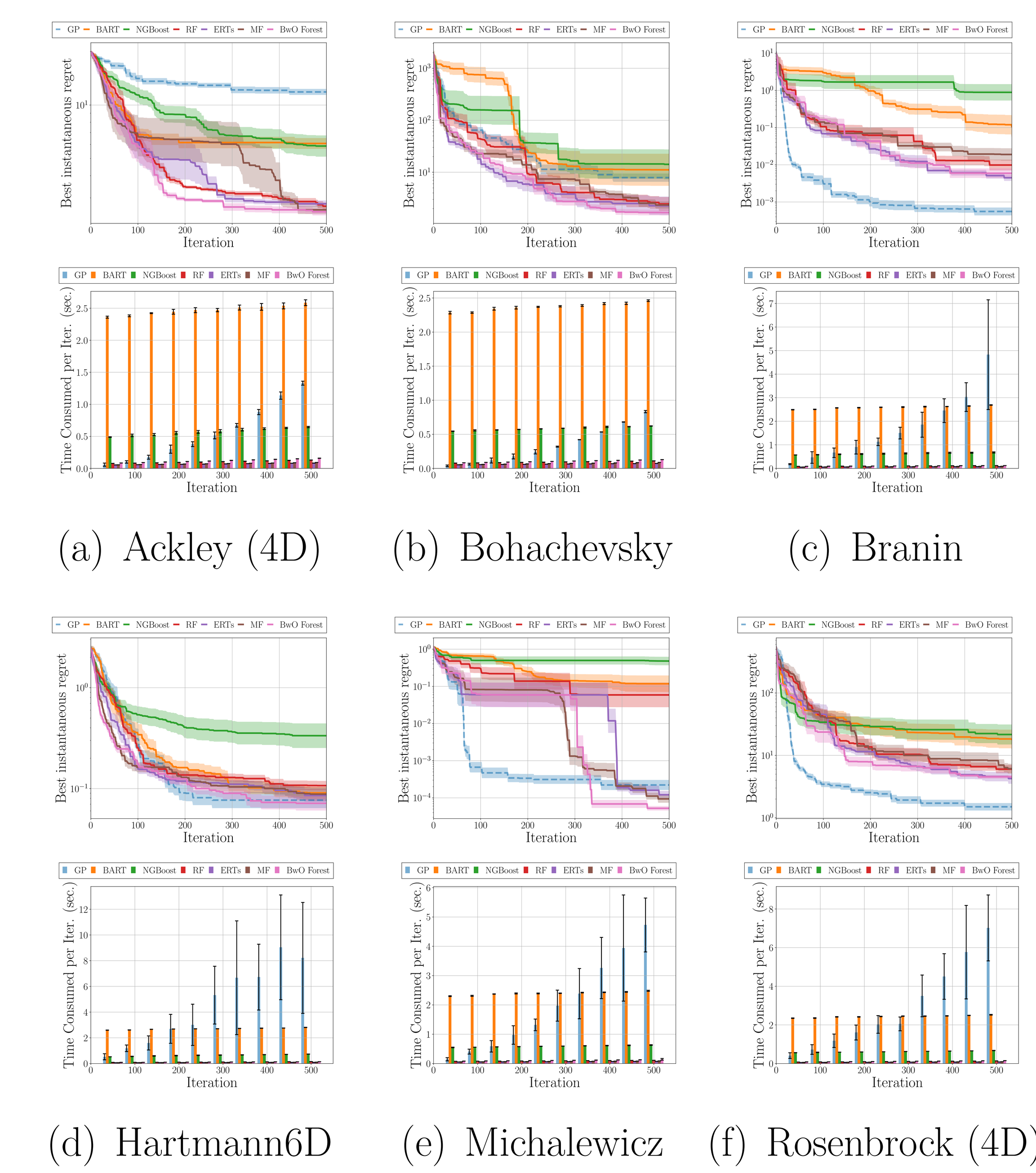
$$\mathrm{Var}[|\mathrm{unique}(\mathbf{B})|] = (N-1)\frac{(N-2)^M}{N^{M-1}} + \frac{(N-1)^M}{N^{M-1}} - \frac{(N-1)^{2M}}{N^{2M-2}},$$

where $\mathrm{unique}(\mathbf{B})$ filters duplicates.

Algorithm 1 Training BwO Forest

Require: Ensemble size B, training data and evaluations $\mathbf{X} \in \mathbb{R}^{N \times d}$ and $\mathbf{y} \in \mathbb{R}^N$, size of bootstrap sample $M = \alpha N$ for an oversampling rate $\alpha > 1$.
Ensure: Set of decision trees $\{\mathcal{T}_b\}_{b=1}^B$

```
Initialize a set of decision trees.
for b = 1, ..., B do
  Sample a bootstrap sample B_b ∈ R^{M×d} from X.
  Set a root node τ_r that contains all the elements in B_b, and τ_r is set as the current split node.
  Train a decision tree using random feature selection and random split location.
end for
return A set of decision trees {T_b}_{b=1}^B
```



(a) Ackley (4D) (b) Bohachevsky (c) Branin (d) Hartmann6D (e) Michalewicz (f) Rosenbrock (4D)

Figure 2: Experimental results on various benchmark functions.

Available on arXiv/GitHub

arXiv GitHub