High-Dimensional Bayesian Optimization via Tree-Structured Additive Models

Eric Han,[1] Ishank Arora,[2] Jonathan Scarlett[1,3]

[1]School of Computing, National University of Singapore (NUS)
[2]Indian Institute of Technology (BHU) Varanasi
[3]Department of Mathematics & Institute of Data Science, NUS
eric_han@nus.edu.sg, ishank.arora.cse14@iitbhu.ac.in, scarlett@comp.nus.edu.sg

35th AAAI Conference on Artificial Intelligence (Feb 2-9, 2021)

Motivating Example

Classifier SGDClassifier from sklearn has the following parameters and more:

- loss
- penalty
- alpha
- l1_ratio
- fit_intercept
- max_iter
- epsilon $\cdots$

Just varying 2 parameters (7×3), we get a huge variation in performance for MNIST:

	#1	#2	#3	$\cdots$	#21
alpha	100	10	100		1
penalty	l1	l1	none		l2
test accuracy	0.099	0.100	0.119		0.925

How can we find the best parameters in such a large space?

Global Optimization - Bayesian Optimization

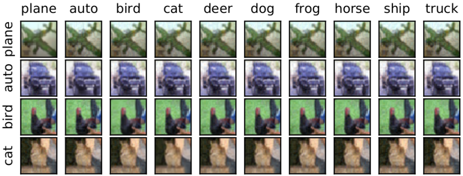
Find the global maximizer $x_{\max}$ in $\mathcal{X}$:

$$x_{\max} = \arg\max_{x \in \mathcal{X}} f(x)$$

BO most suitable for black-box function $f(x)$ with the following properties:

1. is explicitly unknown
2. may be perturbed (i.e. noise) when evaluated
3. is expensive when evaluated

Applications



- **Black-box Adversarial Attack**: Attack Neural Network[1]
- **Model Selection & Parameter Tuning**: Auto-Sklearn
- **Robotics**: Control Problems
- **Finance**: Optimizing portfolio
- **Medicine**: Pharmaceutical Product Development

[1] Diagram taken from Ru et al. (2020)

Bayesian Optimization

Mockus (1994) formulated BO as **a sequential decision process**:

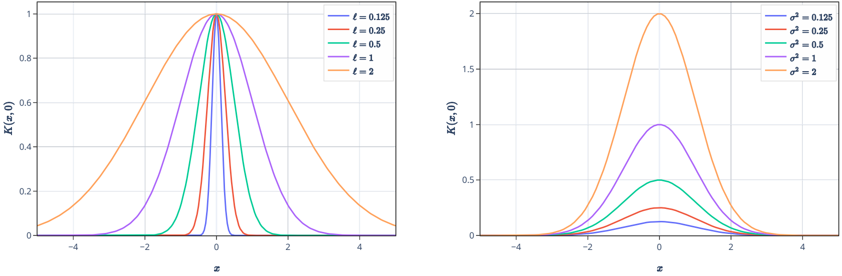
1. Define a prior over the space of possible functions $f(x)$
2. Given some observations, get a posterior over $f(x)$
3. Decide next best location x to evaluate using acquisition function
4. Evaluate $f(x)$ and add to observations

Two key ingredients:

- **Suitable Surrogate model**: prior and posterior
- **Acquisition function**: balance exploration vs exploitation

Suitable Surrogate model - Gaussian Process

RBF Kernel: $K(x, x') = \sigma^2 \exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$, ℓ: length-scale, σ: scale



Kernels describes the covariance of the GP random variables (smoothness)

HDBO - Key Challenges

Curse of dimensionality - needing exponentially many observations

Two significant opposing challenges:

1. **Structural Assumptions**: Identify low-dimensional structure to facilitate efficient the possibility of learning with relatively few samples.
2. **Computational Challenge**: Acquisition functions should be computationally efficient over higher dimensions.

Two key approaches from insights:

1. **Low Effective Dimensionality**: Only few dimensions significantly affect f
2. **Additive Structure**: Small subsets of variables interact with each other

Approach 2 - Additive Structure

Kandasamy, Schneider, and Póczos (2015) formulated $f : \mathcal{X} \to \mathbb{R}$ as additive components:

$$f(x) = \sum_{G \in \mathcal{G}} f^G\left(x^G\right), \qquad \mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_{N_d}$$

- G denotes the set of variables, $G \subseteq \{1, \cdots, N_d\}$
- $f^G : \mathcal{X}^G \to \mathbb{R}$ is a low dimensional function defined on G
- $|\mathcal{G}|$ is the number of low dimensional functions
- Assumed non-overlapping: f^G are pairwise independent

Rolland et al. (2018) generalizes Kandasamy, Schneider, and Póczos (2015) by allowing the groups to be overlapping.

Additive HDBO on Tree Structures - Contributions

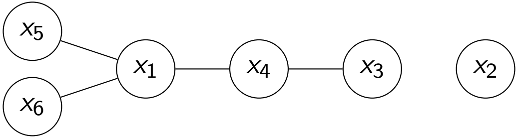
The trend in the study of additive models has been to increase model expressiveness.

Simpler function class, reduces computation and
allows suitable function to be found with fewer samples.

1. Trade-off expressiveness for scalability - constraint dependency graph to **trees**
2. Extended message passsing with a zooming technique to **continuous** domains
3. **Hybrid** method, exploiting tree structures
   3.1 Grows tree via Gibbs sampling
   3.2 Edge mutation
4. Demonstrate the effectiveness of our approach in a wide range of experiments

Additive HDBO on Tree Structures

$$h(x) = h^A(x_1, x_6) + h^B(x_1, x_5) + h^C(x_1, x_4) + h^D(x_3, x_4) + h^E(x_2)$$



UCB acquisition functions are broken into its subsequent components.

$$\phi_t(x) = \sum_{G \in \mathcal{G}} \phi_t^G\left(x^G\right), \qquad \phi_t^G = \mu_{t-1}^G + \beta_t^{1/2} \sigma_{t-1}^G$$

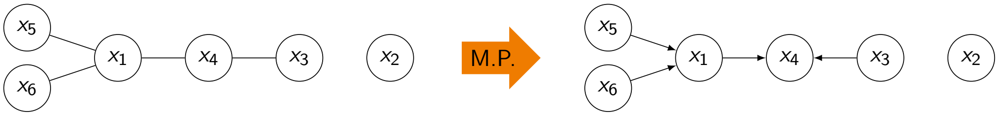
Additive HDBO on Tree Structures

Algorithm 1: TREE-GP-UCB

```
Initialize 𝒟_0 ← {(x_t, y_t)}_{x_t ∈ X_init}
for t = N_init + 1, ..., N_iter do
    if t mod C = 0 then
        Learn 𝒢 ← TREE-LEARNING (Alg. 3)
    Update μ_t^G, σ_t^G : ∀G ∈ 𝒢 (3)
    Optimize x_t ← arg max_{x∈𝒳} φ_t(x) (Alg. 2)
    Observe y_t ← f(x_t) + ε
    Augment 𝒟_t ← 𝒟_{t−1} ∪ {(x_t, y_t)}
return arg max_{(x,y)∈𝒟} y
```

Optimize Acquisiton Functions - Message Passing (Discrete)

Optimization problem is broken down over junction trees, but for tree-structure:



Due to tree structure, computation is reduced from **exponential** in the size of the maximum clique to **quadratic of the domain**.

Optimize Acquisiton Functions - Message Passing (Continuous)

Algorithm 2: MSG-PASSING-CONTINUOUS

Initialize $(\mathbf{a}, \mathbf{b})$ with the bounds of $\mathcal{X}$

for $l = 1, \ldots, L$ **do**

  for $d = 1, \ldots, D$ **do**

    Discretize $\mathcal{X}_d \leftarrow [[a_d, b_d]]_R$

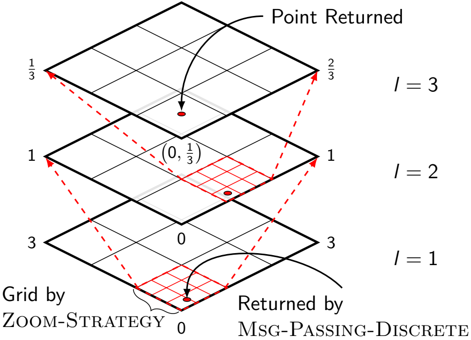
    // $|\mathcal{X}_d| = R$

  $\mathcal{X} \leftarrow \times_{d=1}^{D} \mathcal{X}_d$

  $(x, y) \leftarrow$ MSG-PASSING-DISCRETE $(\mathcal{X})$

  $\boxed{\text{Select } (\mathbf{a}, \mathbf{b}) \leftarrow \text{ZOOM-STRATEGY}\,(x)}$

return (x, y)



Learn Dependency Structure

Algorithm 3: TREE-LEARNING

```
Z_set ← {Z^current}
Z^(k) ← Z^current
while k < S do
    if NUMBER-OF-EDGES(Z^(k)) < D − 1 then
        Update (Z_set, k) via GIBBS-SAMPLING (Alg. 4)
    else
        Update (Z_set, k) via MUTATION (Alg. 5)
return Z ∈ Z_set with the highest likelihood score
```

1. $\mathcal{Z} \leftarrow \{Z^{\text{current}}\}$
2. $Z^{(k)} \leftarrow Z^{\text{current}}$
3. **while** $k < S$ **do**
4. **if** $\boxed{\textsc{Number-Of-Edges}\left(Z^{(k)}\right) < D - 1}$ **then**
5. Update $(\mathcal{Z}, k)$ via $\textsc{Gibbs-Sampling}$ (Alg. 4)
6. **else**
7. Update $(\mathcal{Z}, k)$ via $\textsc{Mutation}$ (Alg. 5)
8. **return** $Z \in \mathcal{Z}$ with the highest likelihood score

Exploiting Tree-Structure: ▶ Tree Sturcture: Gibbs-Sampling ▶ Tree: Mutation

Learn Dependency Structure - Gibbs-Sampling

Algorithm 4: GIBBS-SAMPLING at k-th iteration

```
Initalize UF data structure
for j = 1, ..., D do
    for i = 1, ..., j − 1 do
        Z^(k+1) ← Z^(k)
        if cycle not formed by Z_ij^(k+1) = 1 then
            Sample Z_ij^(new) from posterior
            Z^(k+1) ← Z_ij^(new)
            Update UF via union operation
            Add 𝒵 ← 𝒵 ∪ {Z^(k+1)}
        k ← k + 1
```

Initalize UF data structure

for $j = 1, \ldots, D$ **do**

- **for** $i = 1, \ldots, j-1$ **do**
  - $Z^{(k+1)} \leftarrow Z^{(k)}$
  - **if** *cycle not formed by* $Z_{ij}^{(k+1)} = 1$ **then**
    - Sample $Z_{ij}^{(\text{new})}$ from posterior
    - $Z^{(k+1)} \leftarrow Z_{ij}^{(\text{new})}$
    - Update UF via union operation
    - Add $\mathcal{Z} \leftarrow \mathcal{Z} \cup \left\{ Z^{(k+1)} \right\}$
  - $k \leftarrow k + 1$

Learn Dependency Structure - Mutation

Algorithm 5: MUTATION at k-th iteration

$Z^{(k+1)} \leftarrow Z^{(k)}$

$\boxed{i, j \leftarrow \text{Sample random edge for which } Z_{ij}^{(k+1)} = 1}$

Remove edge: $Z_{ij}^{(k+1)} = 0$

$\boxed{i', j' \leftarrow \text{Sample nodes from the disconnected sub-trees}}$

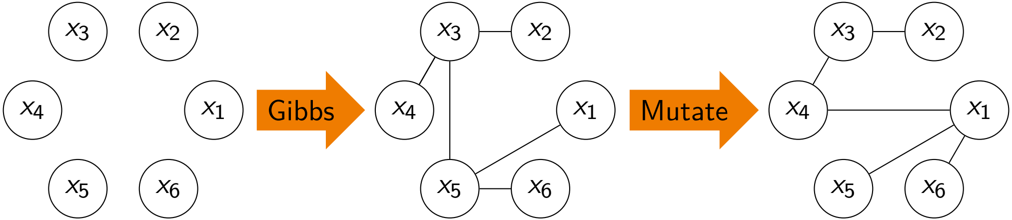
Sample $Z_{i'j'}^{(\text{new})}$ using posterior

$Z^{(k+1)} \leftarrow Z_{i'j'}^{(\text{new})}$

Augment the dataset: $\mathcal{Z} \leftarrow \mathcal{Z} \cup \left\{ Z^{(k+1)} \right\}$

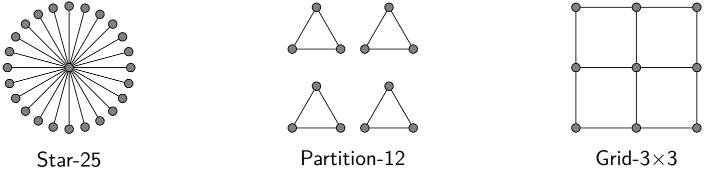
$k \leftarrow k + 1$

Learn Dependency Structure - Example



Experimental Setup

Identical parameters used across all methods, run 25 times in each experiment.



Experiments: ▶ Additive GP Fn ▶ Non-GP Fn
Compared: ▶ Graph Overlap ▶ Graph No-Overlap ▶ LineBO ▶ REMBO
Kernel used: ▶ RBF-ARD ▶ Matern-5/2
Software/Hardware: Conda env over compute cluster

Metrics

- **Optimization Performance** as best regret - closeness to the true optimal.

$$R_t = f_{\max} - f_t^*$$

- **Graph Learning Performance** measures closeness of estimate G is from G_{opt}

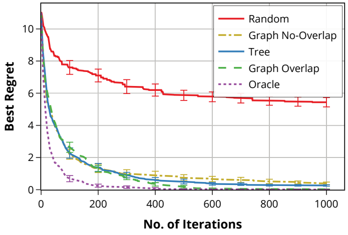
$$\mathrm{F_1score}(G) = 2\frac{\mathrm{Precision}(G) \times \mathrm{Recall}(G)}{\mathrm{Precision}(G) + \mathrm{Recall}(G)}$$

$$\mathrm{Precision}(G) = \frac{|\mathrm{Edges}(G) \cap \mathrm{Edges}(G_{\text{opt}})|}{|\mathrm{Edges}(G)|}, \qquad \mathrm{Recall}(G) = \frac{|\mathrm{Edges}(G) \cap \mathrm{Edges}(G_{\text{opt}})|}{|\mathrm{Edges}(G_{\text{opt}})|}$$

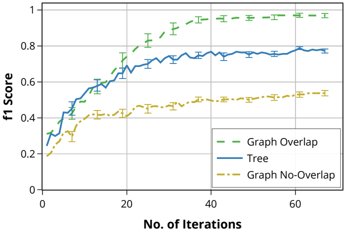
- **Cost Efficiency** counts the number of individual acquisition function evaluations

Additive GP Functions - Not Realizable

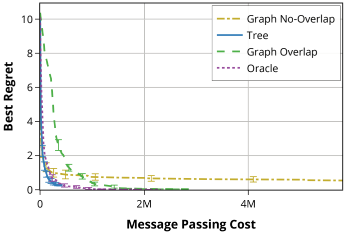
Grid-3×3 (Continuous) - Tree and Graph No-Overlap are not realizable.



Performance

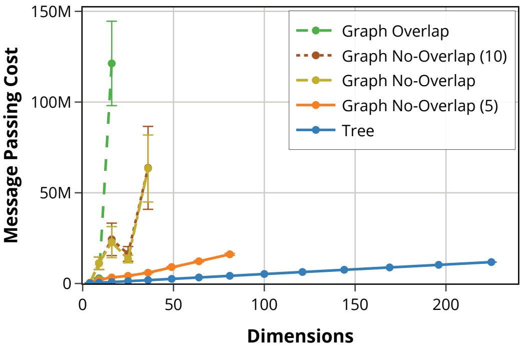


F_1score



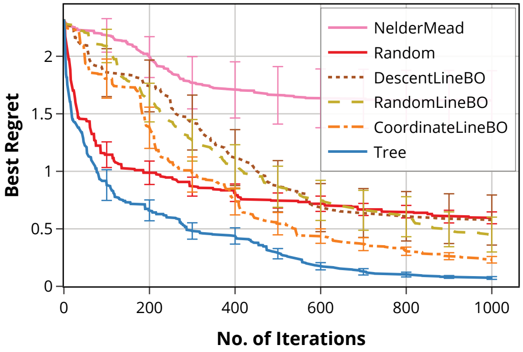
Cost

Additive GP Functions - Scalability

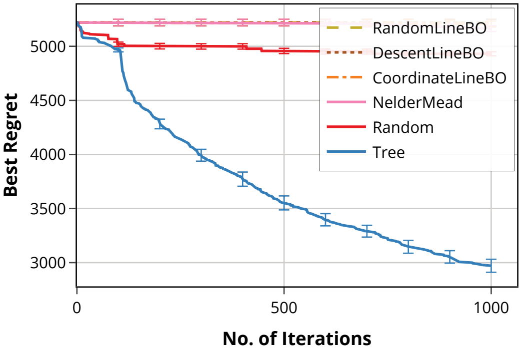


Scalability of Tree over dimensions

Experiments with Non-GP Functions - Synthetic



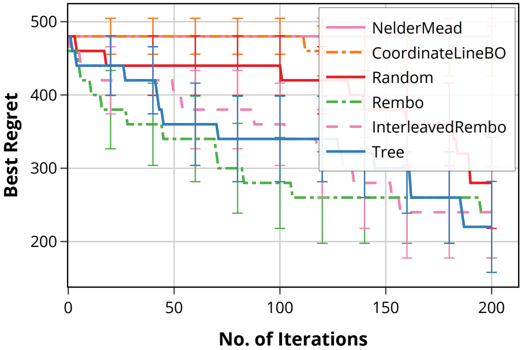
Hartmann6+14Aux Performance



Stybtang250 Performance

Experiments with Non-GP Functions - Real

Additional experiments in the Appendix.



Lpsolve-misc05inf Performance

Conclusion

Tree is competitive on both synthetic and real datasets.

- **Constraint to tree-structures**: Trade-off expressivity for computational efficiency and ease of model learning by reducing model complexity
- **Hybrid structure learning**: Exploit tree structure
  - Gibbs sampling: fast cycle checking
  - Edge mutation: mutation
- **Zooming-based Message Passing**: Extend generalized additive models to continuous domains.

References

Chen, Y.; Huang, A.; Wang, Z.; Antonoglou, I.; Schrittwieser, J.; Silver, D.; and de Freitas, N. 2018. Bayesian optimization in alphago. *arXiv preprint arXiv:1812.06855*.

Kandasamy, K.; Schneider, J.; and Póczos, B. 2015. High dimensional Bayesian optimisation and bandits via additive models. In *Int. Conf. Mach. Learn. (ICML)*, 295–304.

Mockus, J. 1994. Application of Bayesian approach to numerical methods of global and stochastic optimization. *Journal of Global Optimization* 4(4):347–365.

Rolland, P.; Scarlett, J.; Bogunovic, I.; and Cevher, V. 2018. High-dimensional Bayesian optimization via additive models with overlapping groups. In *Int. Conf. Art. Intel. Stats. (AISTATS)*, 298–307.

Ru, B.; Cobb, A.; Blaas, A.; and Gal, Y. 2020. BayesOpt Adversarial Attack. In *Proc. of the International Conference on Learning Representations*.

AlphaGo

"Bayesian optimization provided an automatic solution to tune the game playing hyper-parameters of AlphaGo."[1]

High-Dimensional Bayesian Optimization remains difficult, our work aim to

- lower the computational resources
- facilitate faster model learning
- reducing the model complexity
- retaining the sample-efficiency of additive methods

High-Dimensional Bayesian Optimization via Tree-Structured Additive Models (863)

Eric Han, Ishank Arora, Jonathan Scarlett

AAAI 2021

[1]Quote from: Chen et al. (2018)