Information Theory and Inference

Bayesian Optimization

•••

Festa Filippo Mojsovska Marija Orlando Filippo Peri Andrea

Outilines

- Introduction on Bayesian Optimization
- Gaussian Process (GP)
- Bayesian Optimization (BO)
- Maximisation algorithms
- Bayesian Optimization on SVM and NN
- Dealing with integer and categorical variable
- GP parameters optimization: Frequentist vs Bayesian approach
- BAMSOO: a comparative approach
- Conclusion

Introduction

- **Optimization problems** are **common** in scientific and industrial fields, such as artificial intelligence, data mining, bioinformatics, software engineering, economics, etc.

- Bayesian optimization has emerged at the forefront of expensive **black-box optimization** thanks to its ability to incorporate prior beliefs to help guide the sampling of new data. It allows to achieve a good **balance** between **exploration and exploitation** in the search. It is data efficient unlike popular algorithms that require a large number of function evaluations.

Gaussian Process

A Gaussian Process is a probability distribution over functions:

$$f : \chi^{(d)} \to \mathbb{R}$$

It is defined by the property that any finite set of N points $\{x_i\}$ induces a multivariate gaussian distribution on $\mathbb{R}^N$.

In the Bayesian optimization framework we assume the objective function is sampled from a Gaussian Process:

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

which is univocally specified by the mean and covariance functions:

$$m : X \mapsto \mathbb{R} \qquad k : X \times X \mapsto \mathbb{R}$$

Gaussian Process

Kernels: it is a positive-defined function fundamental to drive the BO algorithm in unexplored regions

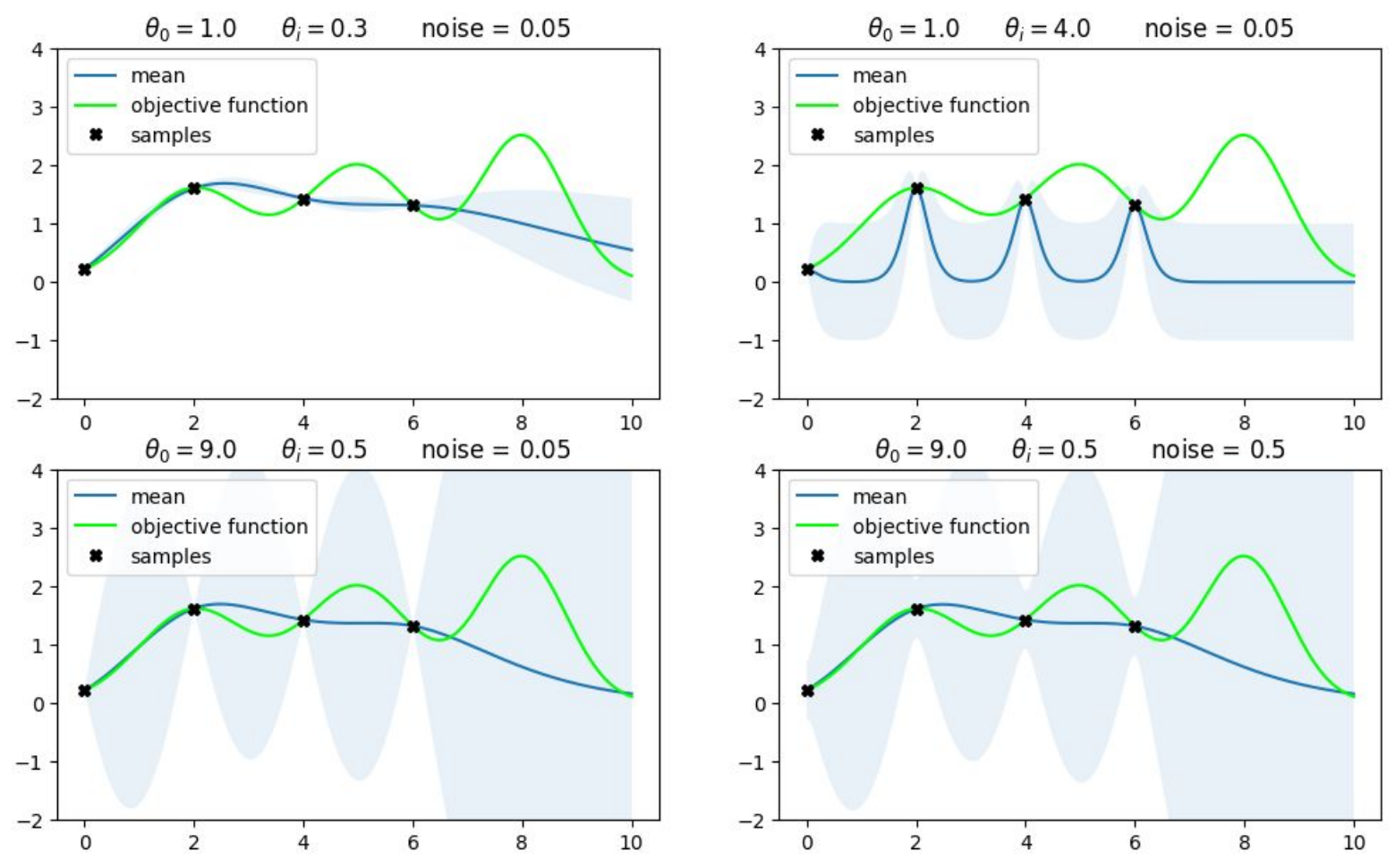
Matern 5 kernel :

$$k_{\text{MATÉRN5}}\left(\mathbf{x}, \mathbf{x}'\right) = \theta_0^2 \exp(-\sqrt{5}r)\left(1 + \sqrt{5}r + \frac{5}{3}r^2\right)$$

$$r^2 = \left(\mathbf{x} - \mathbf{x}'\right)^T \mathbf{\Lambda} \left(\mathbf{x} - \mathbf{x}'\right) \qquad \mathbf{\Lambda} = diag(\theta_1^2, \theta_2^2, ..., \theta_d^2)$$

[3] J. Snoek, H. Larochelle and R. P. Adams, “Practical Bayesian Optimization of Machine Learning Algorithms” *Proc. Adv. Neural Inf. Process. Syst., 2012, pp. 2951–2959.*

GP parameters analysis



BO: Acquisition Function

Probability of Improvement (PI):

$$\mathrm{PI}(\mathrm{x}) = \Phi(Z(x)) \qquad Z(x) = \frac{\mu(x) - \mu_{max} - \epsilon}{\sigma(x)}$$

Expected improvement (EI):

$$\mathrm{EI}(\mathrm{x}) = \begin{cases} (\mu(x) - \mu_{max} - \epsilon)\Phi(Z(x)) + \sigma(x)\phi(Z(x)) & \text{if} \quad \sigma(x) > 0 \\ 0 & \text{if} \quad \sigma(x) = 0 \end{cases}$$

Gaussian Process Upper Confidence Bound (GP-UCB):

$$\mathrm{UCB}(\mathrm{x}) = \mu(x) + \tau\sigma(x)$$

[2] E. Brochu, V. M. Cora and N. de Freitas, "A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning" *arXiv:1012.2599v1 (2010)*

Bayesian Optimization algorithm

Purpose: find the maximum of an objective function f(x) that is expensive or time-consuming to evaluate.

$$x^* = \underset{x}{\operatorname{argmax}} f(x)$$

Algorithm:

- **GP update**

$$\mu_n(\mathbf{x}) = \mathbf{k}(\mathbf{x})^T \left(\mathbf{K} + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{y}$$

$$\sigma_n^2(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}(\mathbf{x})^T \mathbf{K}^{-1} \mathbf{k}(\mathbf{x})$$

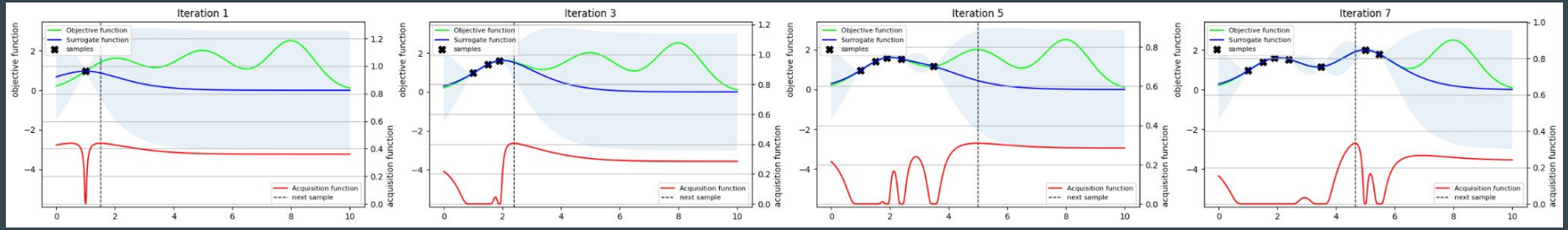
where:

$$\mathbf{k}(x) = (k(x, x_1), k(x, x_2), ..., k(x, x_d)) \qquad \mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_d) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_d, \mathbf{x}_1) & \cdots & k(\mathbf{x}_d, \mathbf{x}_d) \end{bmatrix} + \sigma_{noise}^2 I$$

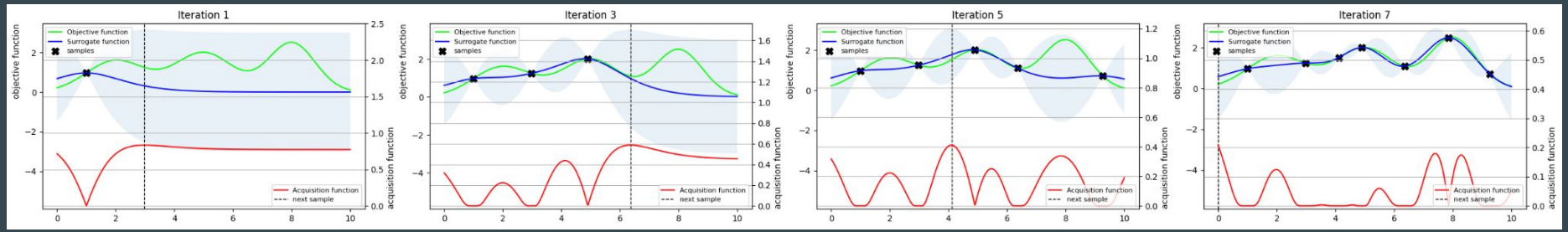
- **Maximisation of the acquisition function**
- **Evaluation of the objective function**

[1] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the Human Out of the Loop: A Review of Bayesian Optimization” *Proceedings of the IEEE | Vol. 104, No. 1, January 2016*

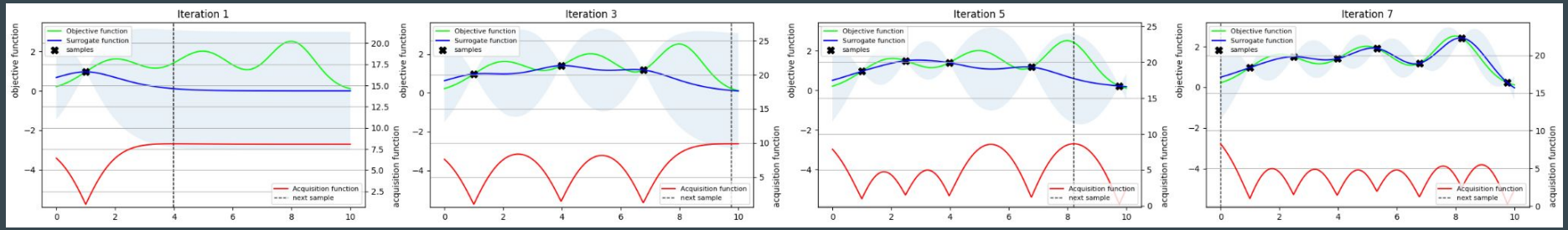
Probability of Improvement



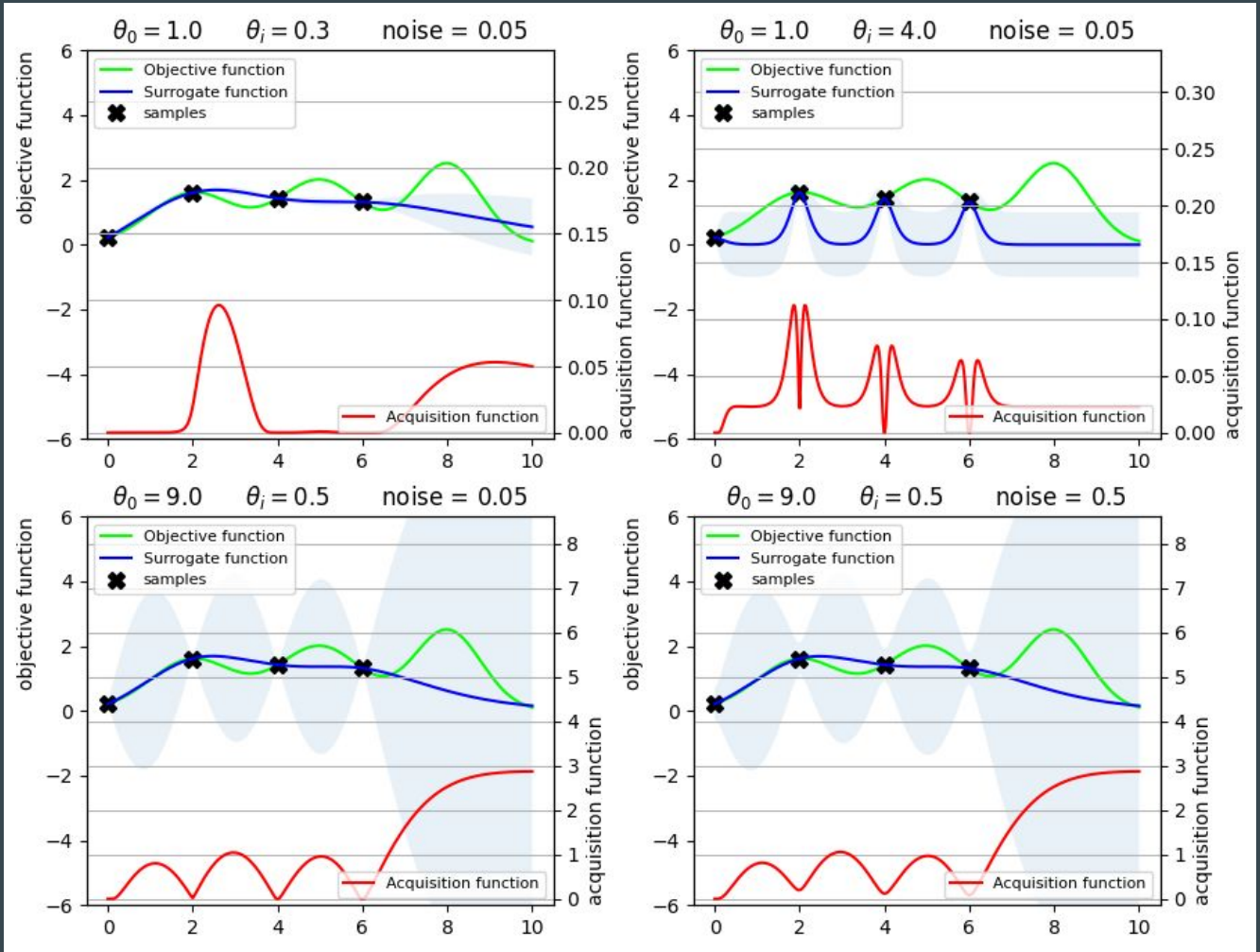
Expected Improvement



Gaussian Process Upper Confidence Bound

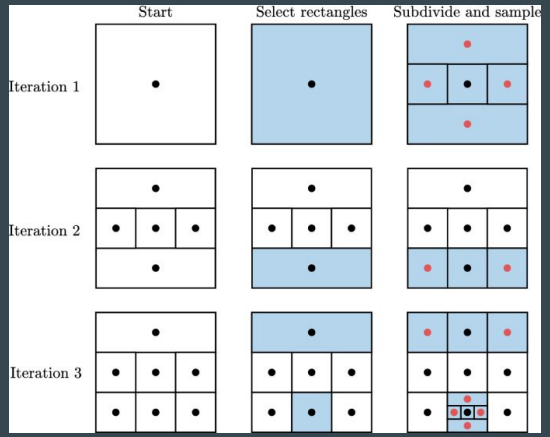


GP parameters analysis

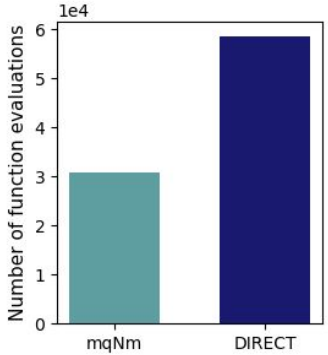


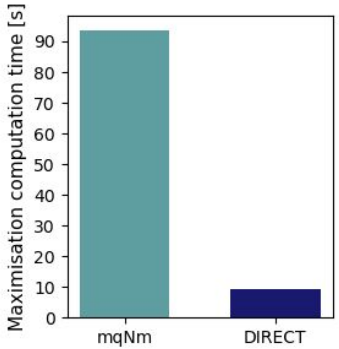
BO: Maximization algorithm

Direct optimization strategy

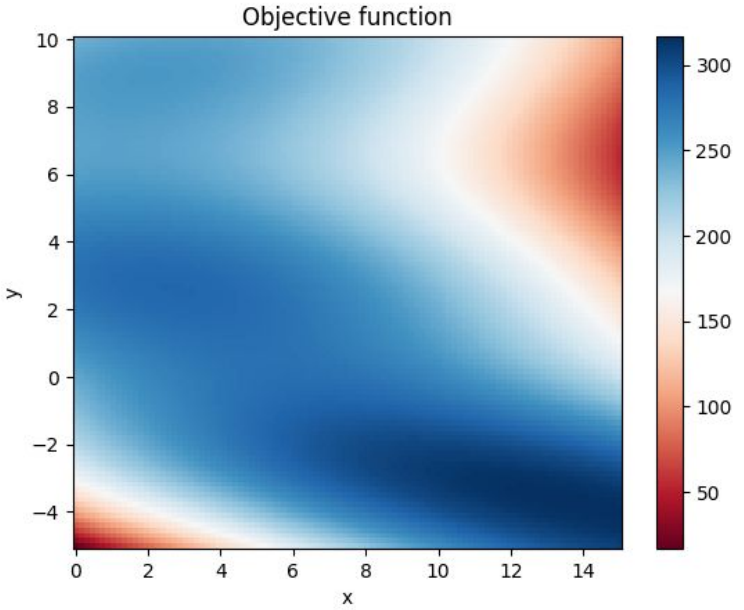


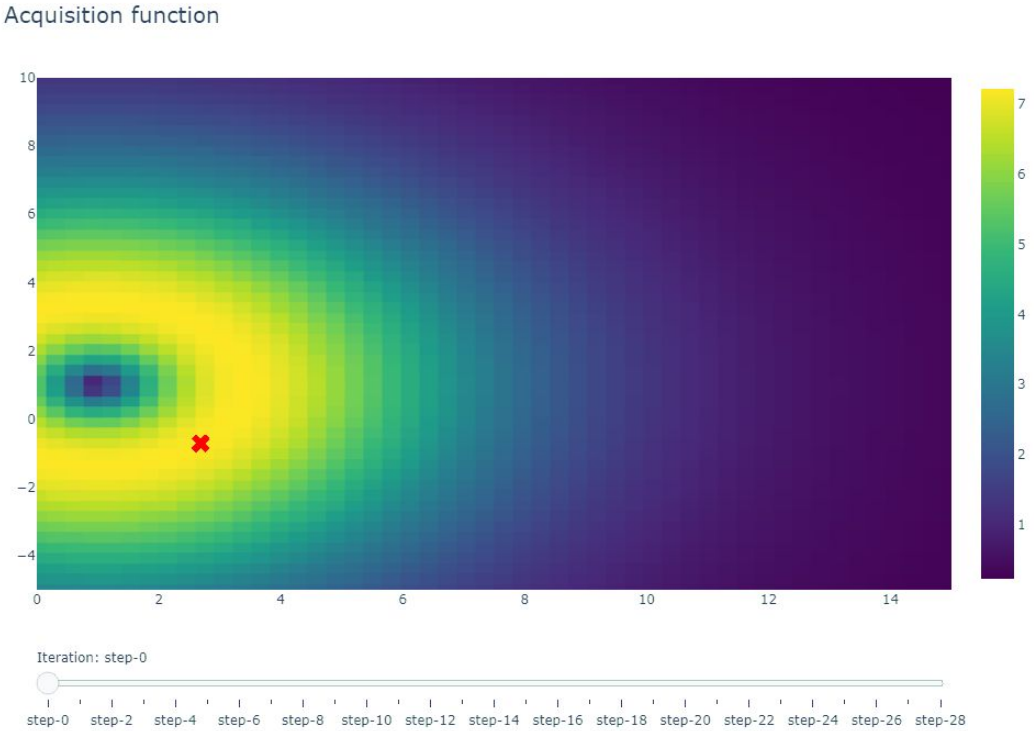
- **Quasi Newton methods** are **local** optimization methods based on **Newton's method.** They use the information of the **first derivatives** of the function to compute an approximation of the Hessian matrix to update the search direction..
- **DIRECT algorithm** is a **global** optimization method that does **not** require **derivatives** of the objective function. It is based on dividing the search domain into **hyperrectangles** and sampling the function at their centers.





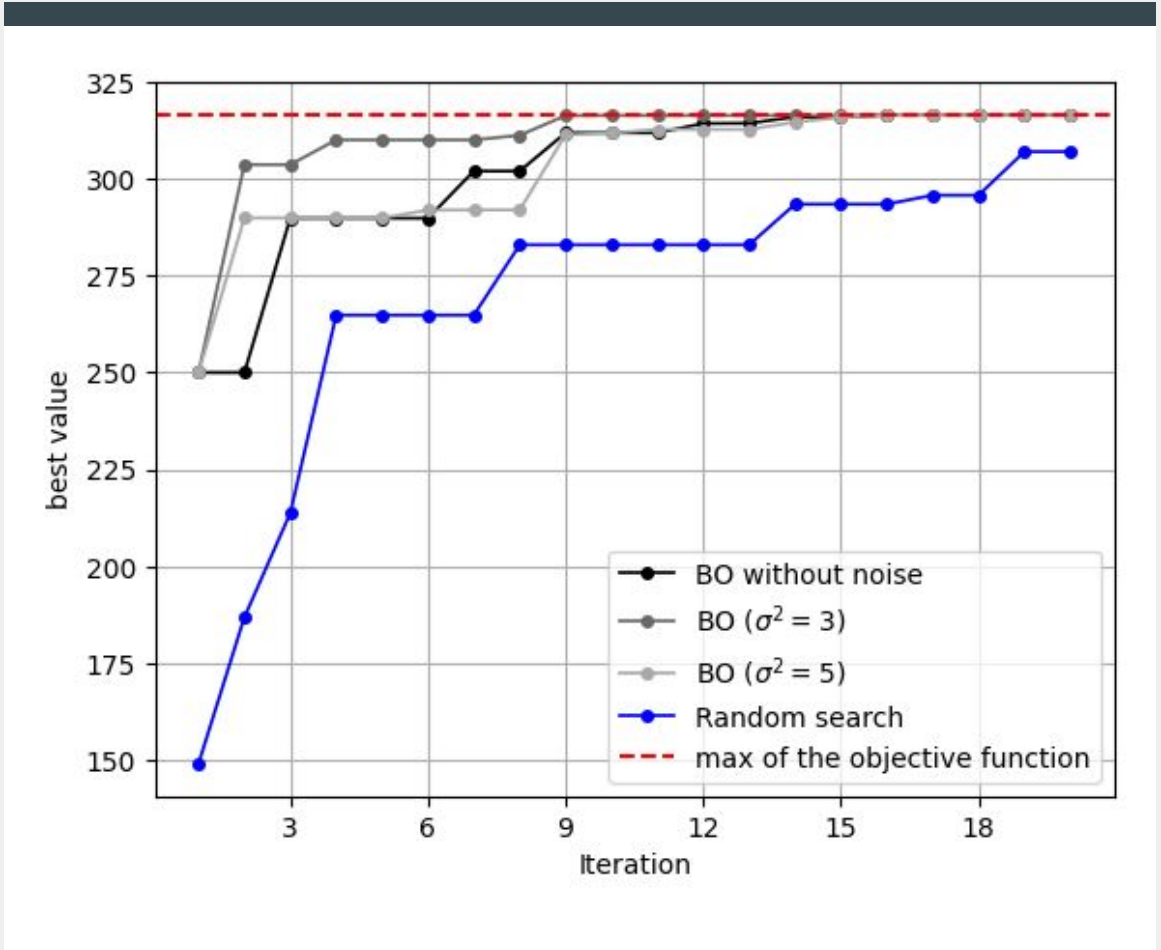
Bayesian optimization on Branin function (2D parameter space)





Comparison of BO and Random Search

Objective: Branin function



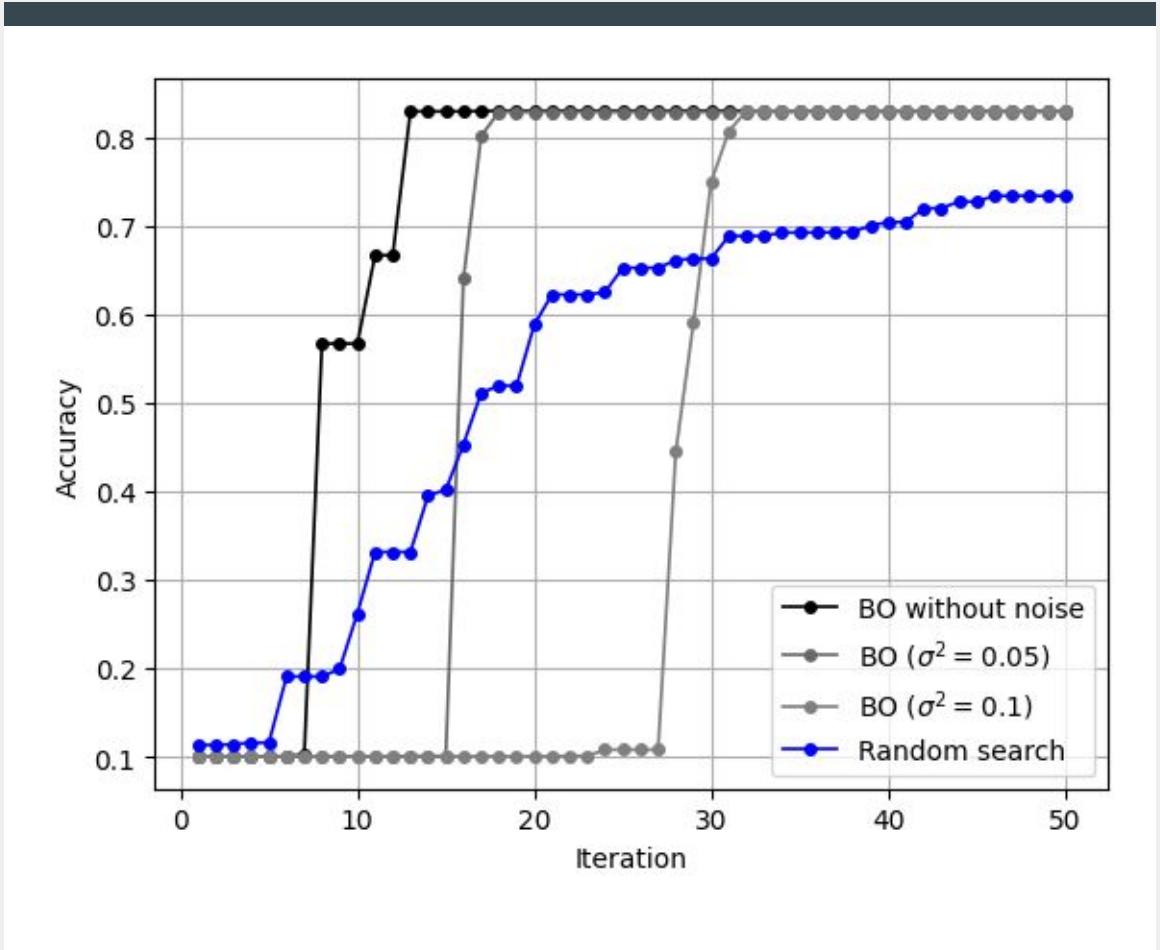
Comparison of BO and Random Search

Objective: SVM

Dataset: MNIST

Optimized hyperparameters:

- **gamma** ($1/\sigma$ of the Gaussian Kernel): [0.001, 1]
- **C** (regularization parameter) : [0.1, 10]



Integer and Categorical-value Variables

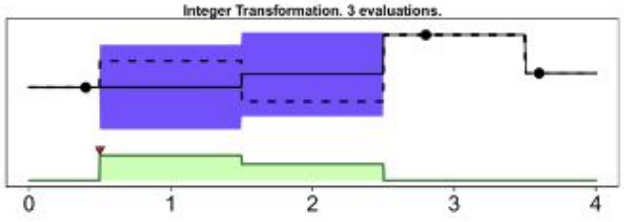
- When some of the input variables are **categorical or integer, they are not directly compatible with the Gaussian process (GP)** because the covariant function of GP assumes that input are real-valued.
- To solve this problem we use a transformation **T(x)** right before computing the covariance function:

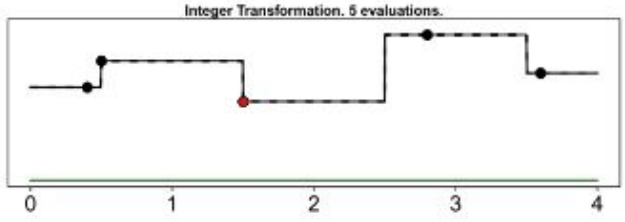
$$k'(\mathbf{x}_i, \mathbf{x}_j) = k(T(\mathbf{x}_i), T(\mathbf{x}_j))$$

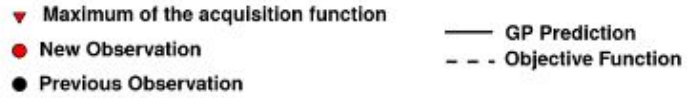
- **Categorical variables**: the input variables that correspond to the same categorical variable are assigned the value 0, unless they take the largest value among its corresponding group of extra variables. If they take the largest value, they are set equal to 1 (**one hot encoding approximation**).
- **Integer-valued variables**: the input variables that correspond to an integer-valued variable are simply rounded to the closest integer value.

[4] E. C. Garrido-Merchán, D. Hernández-Lobato, "Dealing with categorical and integer-valued variables in Bayesian Optimization with Gaussian processes" *ttps://doi.org/10.1016/j.neucom.2019.11.004*

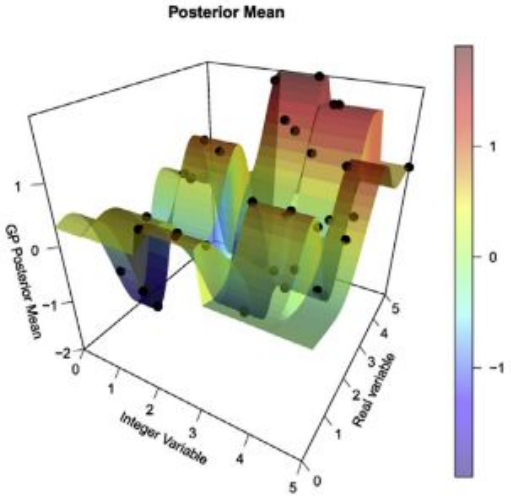
Integer and Cathegorical: SVM, ML

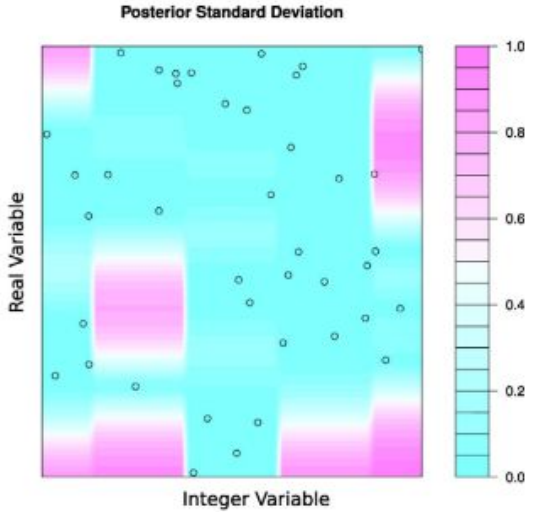






Proposed Approach





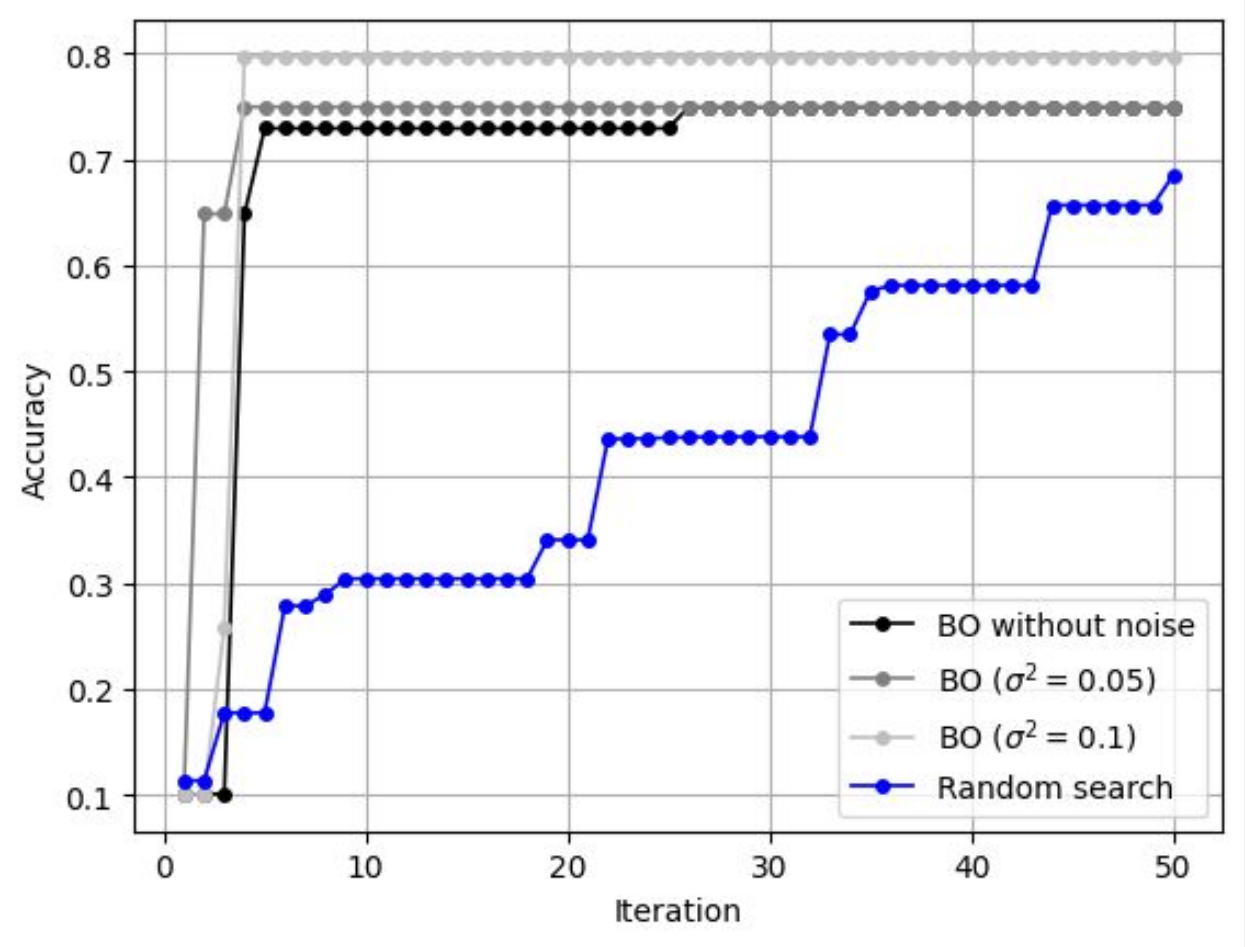
SVM and NN with categorical and integer

- **Support Vector Machine:**
  - Dataset: MNIST
  - **Hyperparameters**:
    - **gamma** (1/sigma of the Gaussia Kernel): [0.001, 1]
    - **C** (regularization parameter) : [0.1, 10]
    - **kernel**: ‘rbf’, ‘poly’
    - **degree**: degree of the polinomial

- **Neural Network:**
  - Multi-layer Perceptron classifier
  - Dataset: MNIST
  - Hyperparameters:
    - **learning rate**: [0.0002, 2]
    - **batch size**: [1, 500]
    - **solver**: ‘sgd’, ‘adam’, ‘lbfgs’
    - **number of neurons**: [5, 50]
    - **alpha**: [0.00001, 0.1]

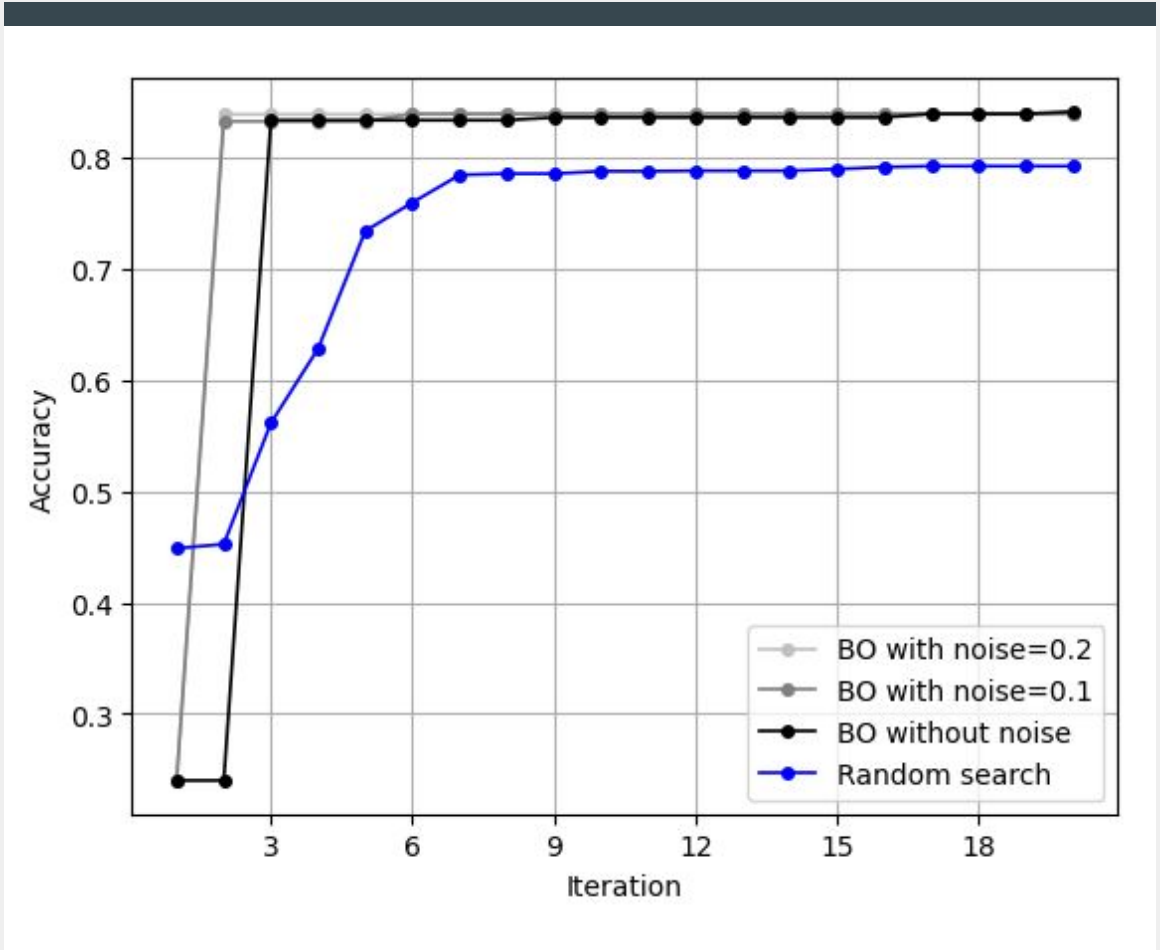
Comparison of BO and Random Search

Objective: SVM (5D hyperparameter space)



Comparison of BO and Random Search

Objective: NN (7D hyperparameter space)



How to deal with GP parameters optimization?

Gaussian Processes depends on some (GP) parameters:

$$GP(m(x), k(x, x')|\theta_0, \{\theta_i\}, \sigma_{noise})$$

The optimization of these parameters is easily carried out by the user when dealing with a low dimensional objective function ($D \leq 2$), but it soon becomes intractable as the dimensionality increase.

There are two ways to approach this problem:

- Maximise the posterior $p_{\Theta|x}(\theta|x)$ at every iteration of the Bayesian Optimization, and such parameters to describe the GP
- Obtain a sample $\{\Theta_n\}$ distributed according to $p_{\Theta|x}(\theta|x)$ from a Markov Chain and use it to marginalize the acquisition function over the GP parameters

GP parameters posterior distribution

From the mathematical expression of the gaussian process we can derive an analytical formula of the likelihood of the data given the GP parameters:

$$\log p\left(\mathbf{y} \mid \{x_i\}, \theta\right) = -\frac{1}{2}\mathbf{y}^T \left(\mathbf{K}_{\{x_i\},\theta} + \sigma^2\mathbf{I}\right)^{-1}\mathbf{y} - \frac{1}{2}\log\left|\mathbf{K}_{\{\mathbf{x_i}\},\theta} + \sigma^2\mathbf{I}\right| - \frac{n}{2}\log(2\pi)$$

We can define the posterior distribution of the parameters as:

$$p(\theta|y, \{x_i\}) = \frac{1}{Z} \cdot p(y|\{x_i\}, \theta) \cdot p_{prior}(\theta)$$

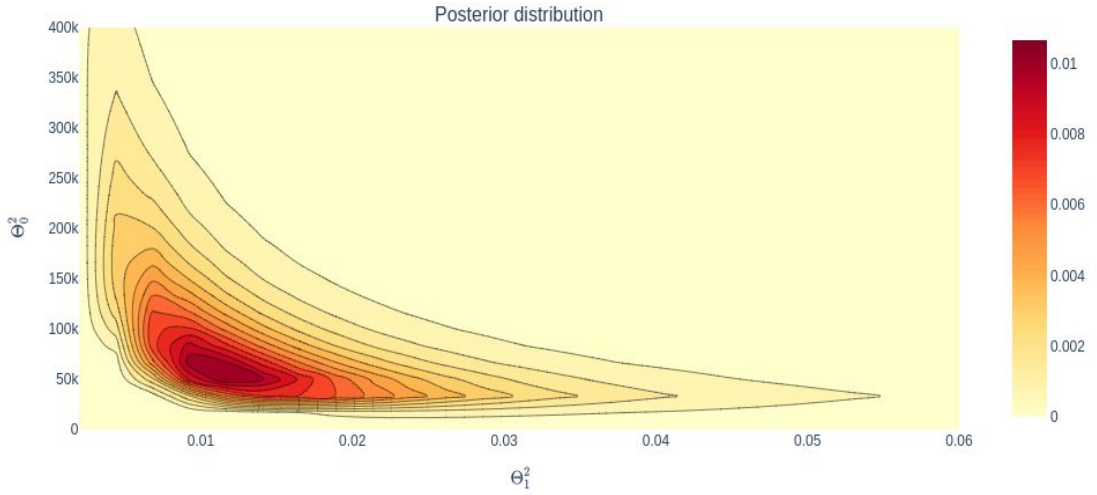
where we used a uniform (non-informative) prior.

Maximising the posterior distribution allows to follow a frequentist approach to GP parameters optimization.

[1] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the Human Out of the Loop: A Review of Bayesian Optimization” *Proceedings of the IEEE | Vol. 104, No. 1, January 2016*

Posterior distribution: an example

The following plot shows the posterior obtained from a run of the BO maximising the Branin function.
The GP parameters space is restricted to show a bidimensional plot:



Bayesian approach: the integrated acquisition function

The integrated acquisition function allows to use a fully-bayesian strategy in the maximisation of the objective function.

Instead of defining an acquisition function depending on the GP parameters Θ, we marginalize over them and so define the integrated acquisition function:

$$\hat{a}(\mathbf{x}; \{\mathbf{x}_n, \mathbf{y}_n\}) = \int \hat{a}(\mathbf{x}; \{\mathbf{x}_n, \mathbf{y}_n\}, \theta) p(\theta | \{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^{N}) \mathrm{d}\theta$$

We designed a Metropolis-Hastings (MH) algorithm having the posterior distribution of the GP parameters as its limiting distribution. In this way, we obtain an estimate of the integral exploiting the Monte Carlo method:

$$\mathbb{E}_{\theta | \{\mathbf{x}_n\}_{n=1,\dots,N}} [\hat{a}(\mathbf{x}; \{\mathbf{x}_n, \mathbf{y}_n\}, \theta)] \sim \frac{1}{M} \sum_{i=1}^{M} \hat{a}(\mathbf{x}; \{\mathbf{x}_n, \mathbf{y}_n\}, \theta_i)$$

[1] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the Human Out of the Loop: A Review of Bayesian Optimization” *Proceedings of the IEEE | Vol. 104, No. 1, January 2016*

Metropolis-Hastings algorithm

In our framework we deal with non-negative GP parameters.
We tested two different approaches:

1. Use a truncated Gaussian as proposal distribution, which results in the following acceptance ratio for the MH:

$$A(x', x) = min\left\{1, \frac{\pi(x') \cdot \Phi(x)}{\pi(x) \cdot \Phi(x')}\right\}$$

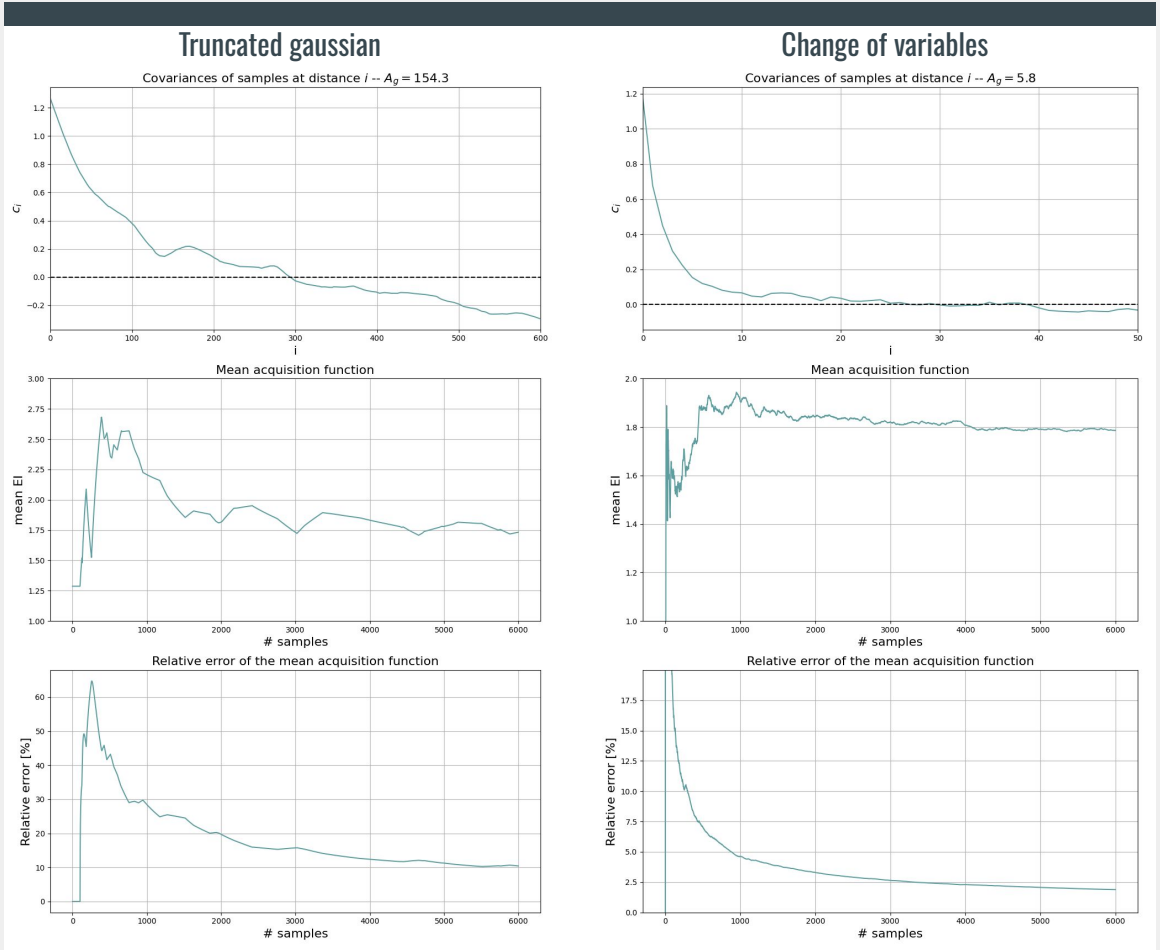
2. Perform a change of variable into a logarithmic space. We employed a custom proposal distribution function, sampled with a rejection sampling algorithm:
   - Proposal distribution: $q(x', x) = \frac{1}{x} log\mathcal{N}(\log(x), \sigma)$
   - Acceptance ratio: $A(x', x) = min\left\{1, \frac{\pi(x') \cdot \Pi_i x_i'^2}{\pi(x) \cdot \Pi_i x_i^2}\right\}$

After conducting several tests, we have chosen the second approach due to its faster convergence and lower autocorrelation.

Optimization of Metropolis-Hastings algorithm for BO applications

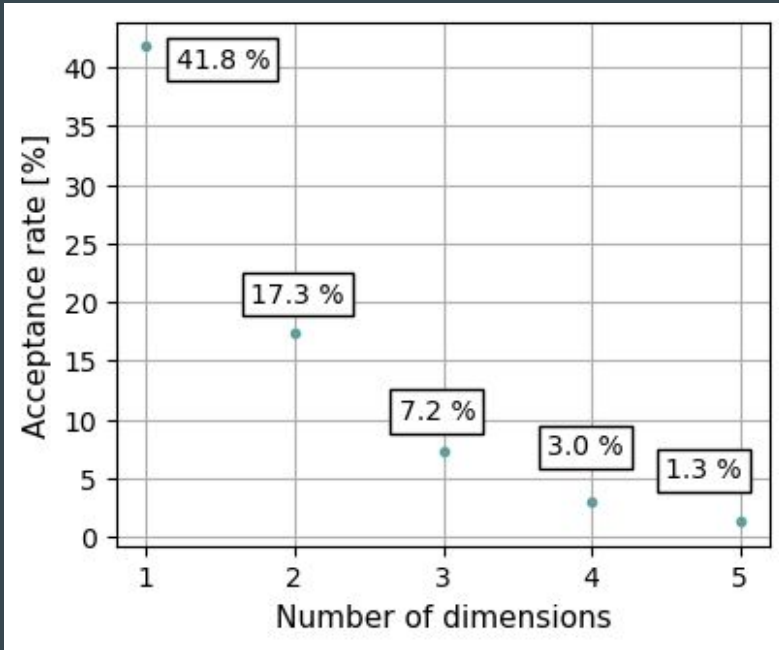
To obtain a fast convergence and a small autocorrelation, we need to tune differents parameters of the sampling algorithm, such as:

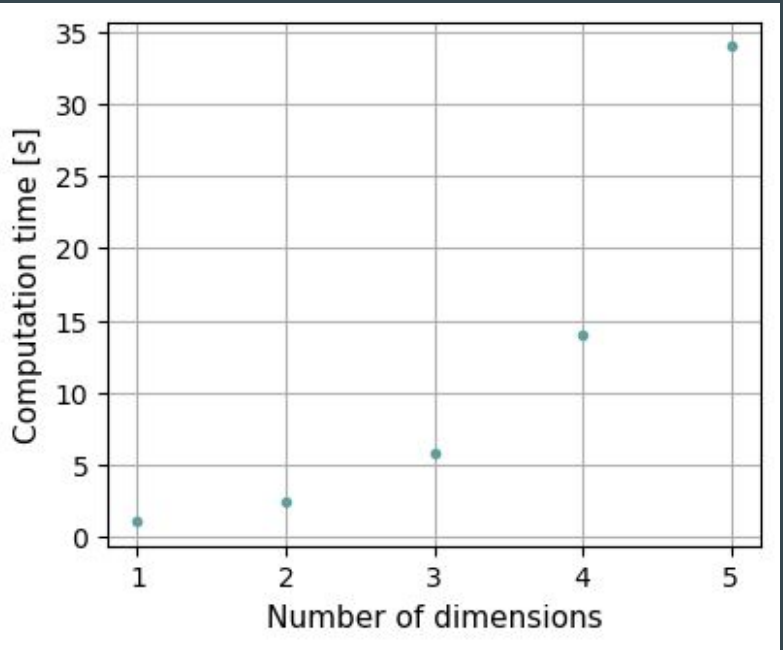
- proposal distributions and related parameters
- number of samples
- thinning



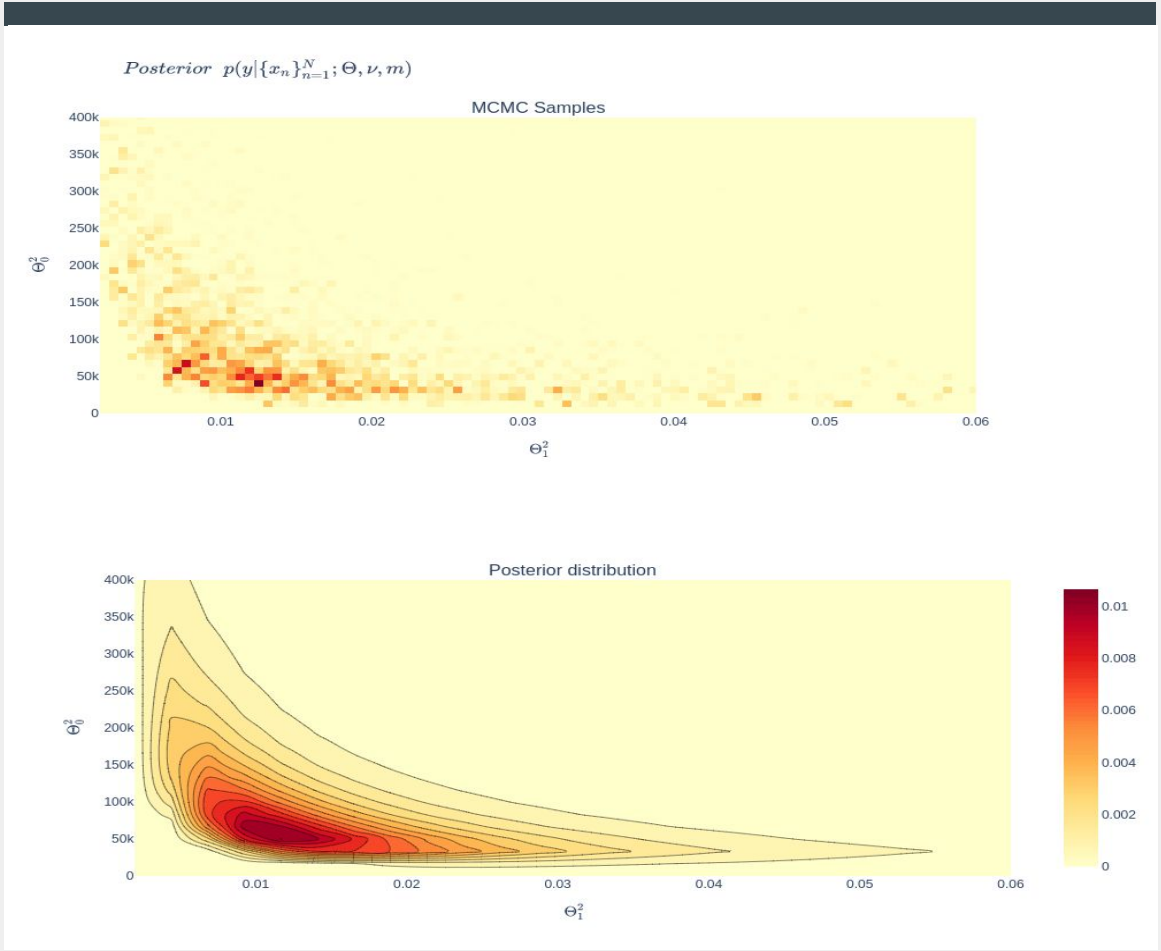
Rejection Sampling

Rejection sampling has been exploited to sample the proposal distribution for each Metropolis-Hastings step.

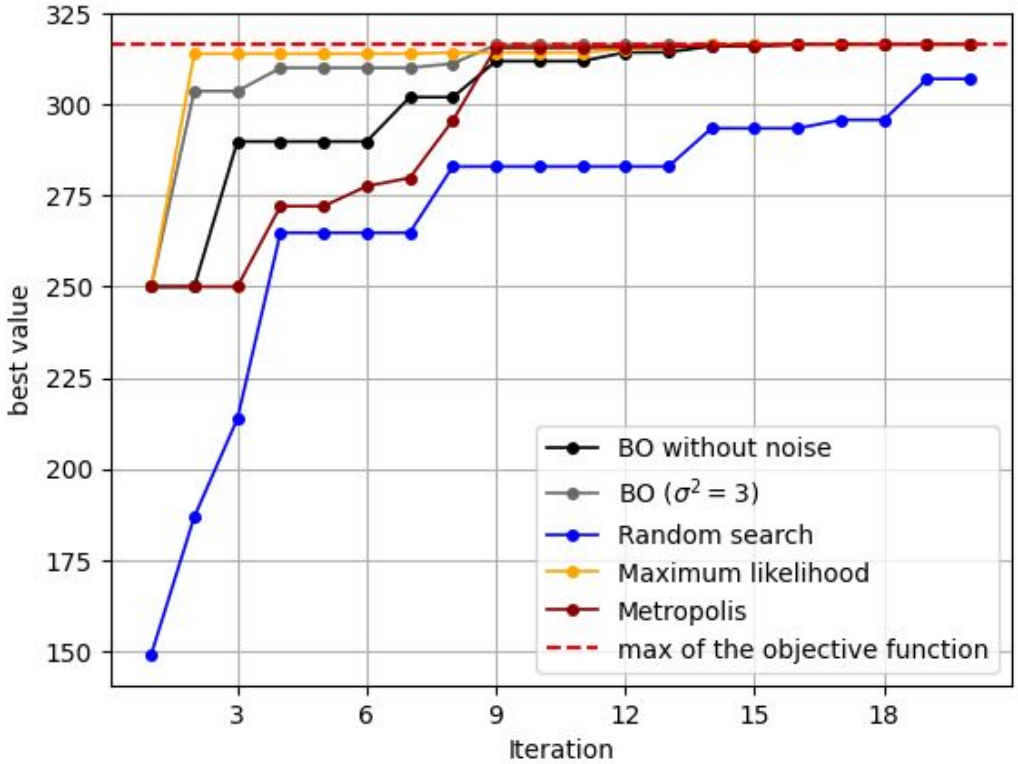




2D visualization of the performed sampling



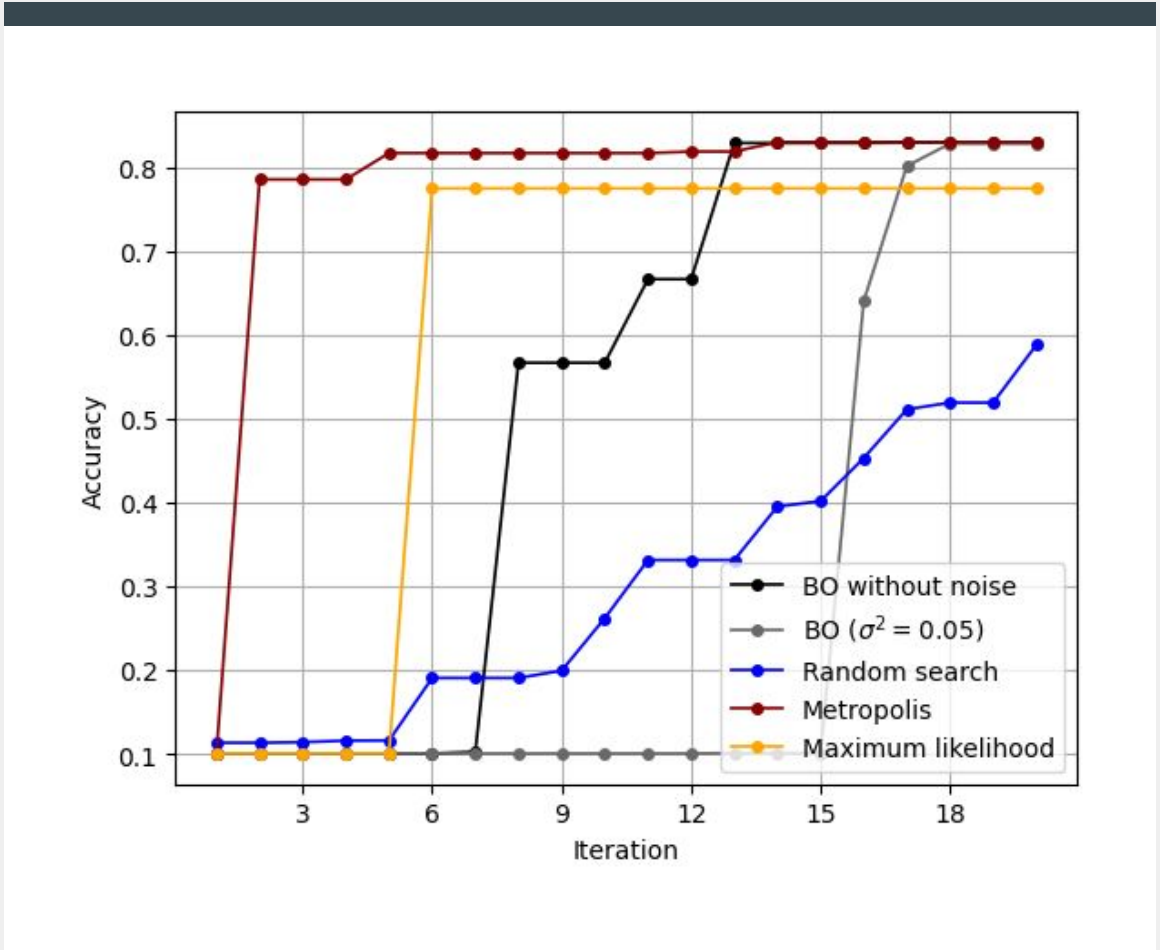
Comparison of different BO methods: Branin



Comparison of different BO methods: SVM

Optimized hyperparameters:

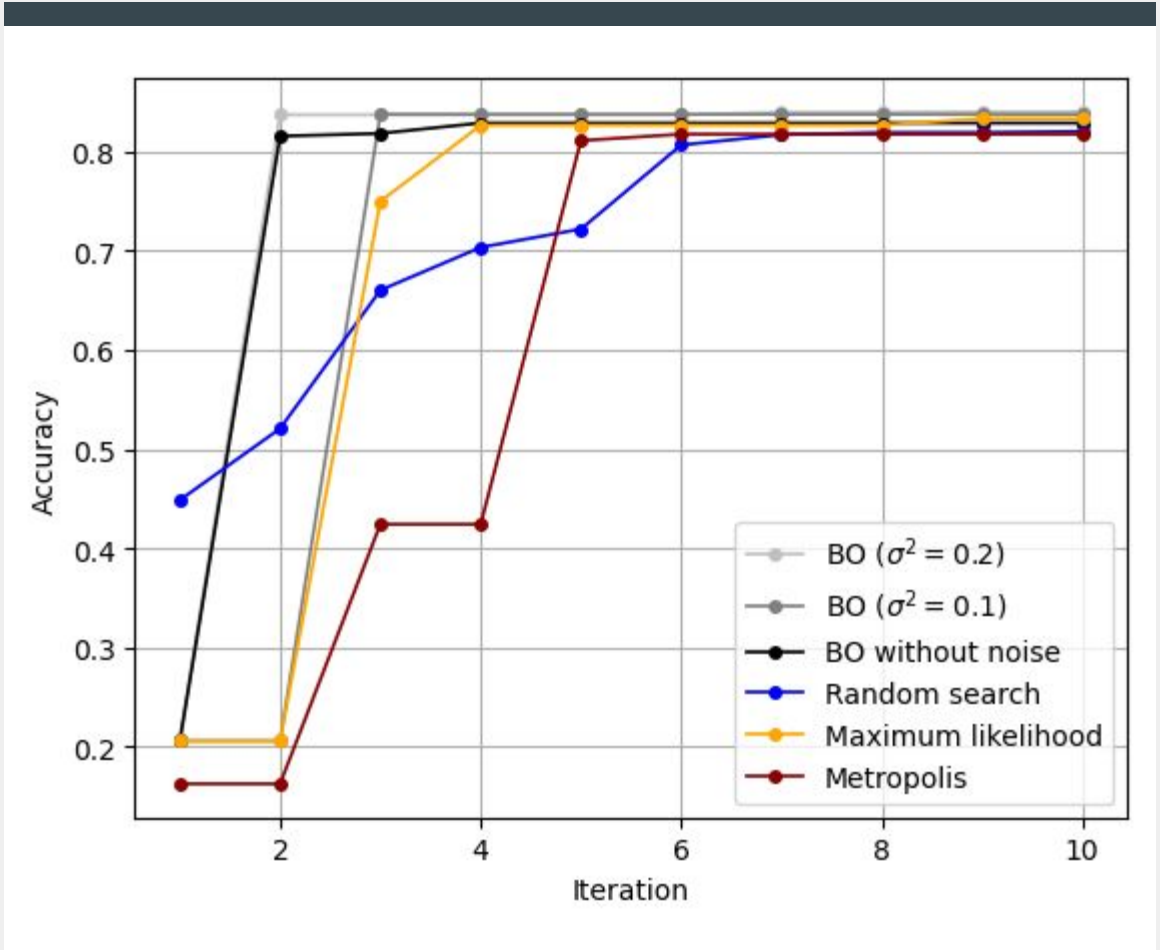
- **gamma** (1/sigma of the Gaussian Kernel): [0.001, 1]
- **C** (regularization parameter) : [0.1, 10]



Comparison of different BO methods: NN

Optimized hyperparameters:

- learning rate
  $[2 \cdot 10^{-4}, 2]$
- batch size
  $[1, 500]$



BaMSOO: Bayesian Multi-Scale Optimistic Optimization

One of the shortcomings of BO is that it requires auxiliary optimization of an acquisition function at each iteration.

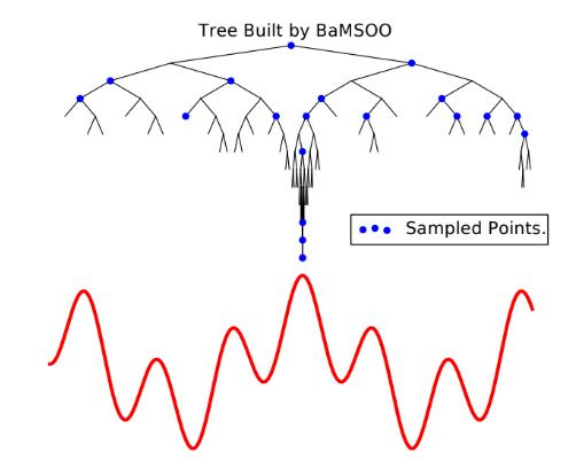
- running an auxiliary optimizer at each iteration of the BO algorithm can be unnecessarily costly
- theoretical convergence guarantees are only valid on the assumption that the optimum of the acquisition function can be found exactly

BaMSOO *(Wang et al.(2014))*:
Combines elements of BO and tree based simultaneous optimistic optimization (SOO) to eliminate the need for auxiliary optimization of acquisition functions

- Uses **optimistic optimization** to effectively **balance exploration** and **exploitation** .
- The **multi-scale** allows it to model the objective function at different scales: capture both the **global** and **local** structure of the objective function.

[5] Z. Wang, B. Shakibi, L. Jin and N. de Freitas, “Bayesian Multi-Scale Optimistic Optimization” *AI and Statistics, pp. 1005–1014, 2014.*

BaMSOO : the algorithm

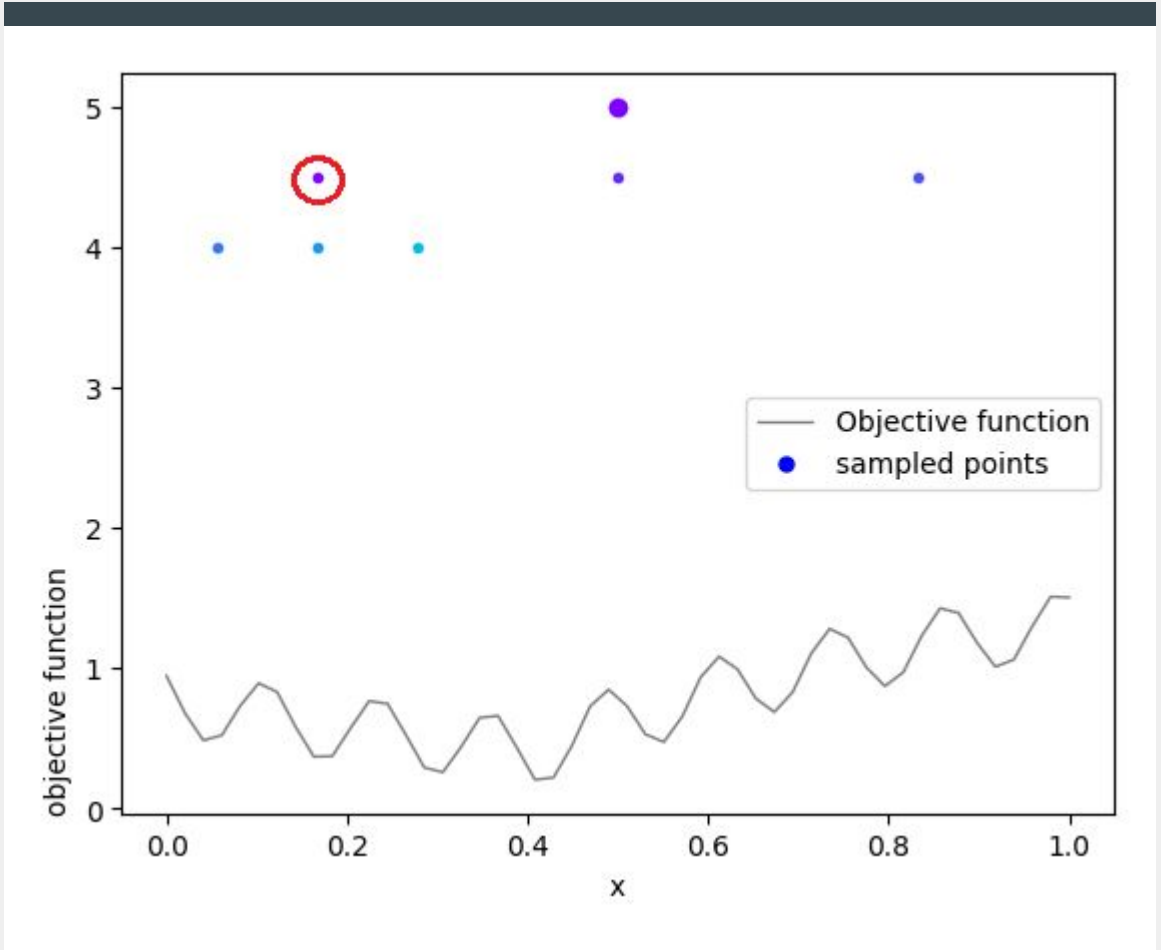


Algorithm 1 BaMSOO pseudocode

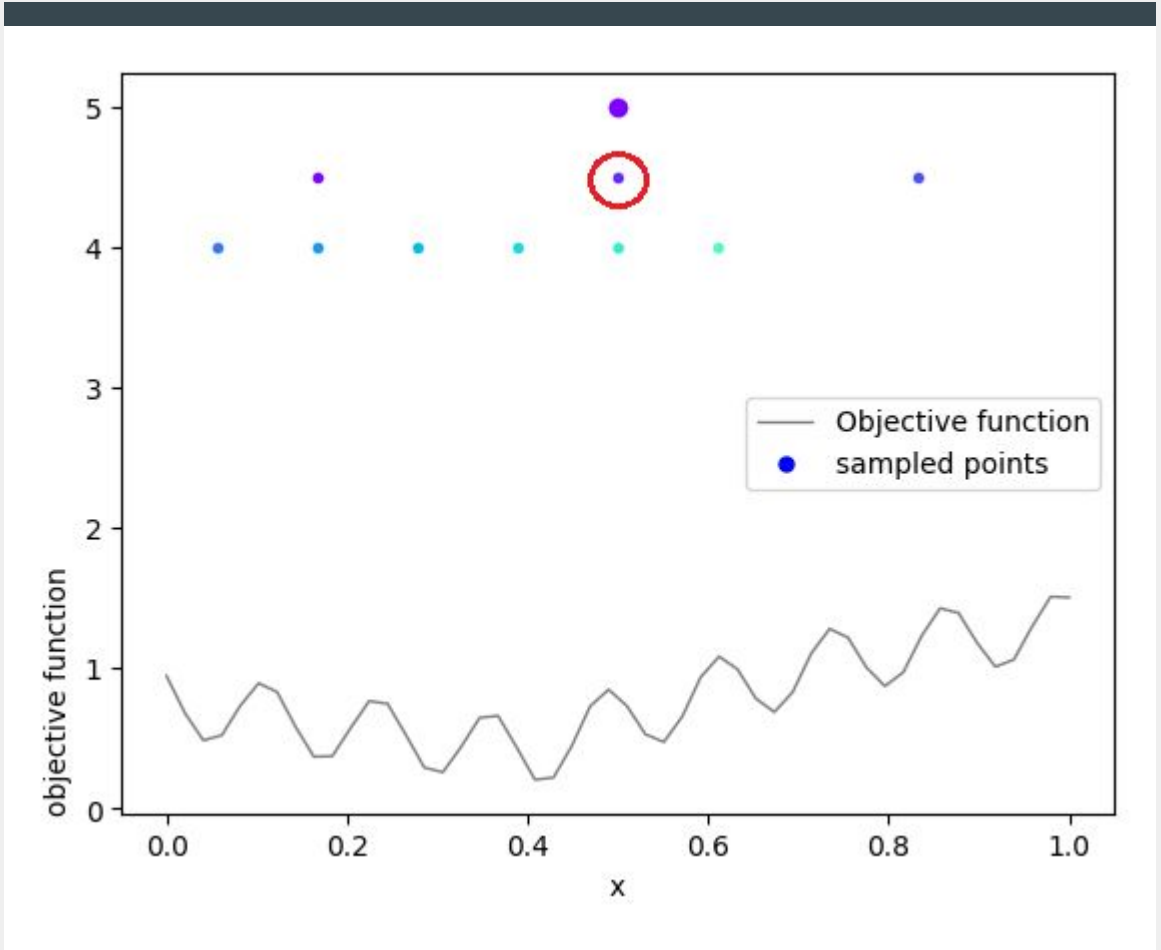
Set $g_{0,0} = f(\mathrm{x}_{0,0})$
Set $f^+ = g_{0,0}$
Initialize the tree $\mathcal{T}_1 = \{0, 0\}$
Set $t = 1$, $n = 1, N = 1$, and $\mathcal{D}_t = \{(\mathrm{x}_{0,0}, g(\mathrm{x}_{0,0}))\}$
while true **do**
  Set $\nu_{\min} = \infty$.
  for $h = 0$ to $\min\{\mathrm{depth}(\mathcal{T}_n), h_{\max}(n)\}$ **do**
    Select $(h, j) = \arg\min_{j \in \{j \mid (h,j) \in L_n\}} g(\mathrm{x}_{h,j})$
    if $g(x_{h,j}) < \nu_{\min}$ **then**
      for $i = 0$ to $k - 1$ **do**
        Set $N = N + 1$
        if $\mathcal{L}_N(\mathrm{x}_{h+1,kj+i} | \mathcal{D}_t) \leq f^+$ **then**
          Set $g(\mathrm{x}_{h+1,kj+i}) = f(\mathrm{x}_{h+1,kj+i})$
          Set $t = t + 1$
          $\mathcal{D}_t = \{\mathcal{D}_{t-1}, (\mathrm{x}_{h+1,kj+i}, g(\mathrm{x}_{h+1,kj+i}))\}$
        else
          Set $g(\mathrm{x}_{h+1,kj+i}) = \mathcal{U}_N(\mathrm{x}_{h+1,kj+i} | \mathcal{D}_t)$
        end if
        if $g(\mathrm{x}_{h+1,kj+i}) < f^+$ **then**
          Set $f^+ = g(\mathrm{x}_{h+1,kj+i})$
        end if
      end for
      Add the children of (h, j) to $\mathcal{T}_n$
      Set $\nu_{\min} = g(\mathrm{x}_{h,j})$
      Set $n = n + 1$
    end if
  end for
end while

$$\mathcal{U}_N(\mathbf{x} \mid \mathcal{D}_t) = \mu(\mathbf{x} \mid \mathcal{D}_t) + B_N \sigma(\mathbf{x} \mid \mathcal{D}_t)$$
$$\mathcal{L}_N(\mathbf{x} \mid \mathcal{D}_t) = \mu(\mathbf{x} \mid \mathcal{D}_t) - B_N \sigma(\mathbf{x} \mid \mathcal{D}_t)$$
$$B_N = \sqrt{2 \log\left(\pi^2 N^2 / 6\eta\right)} \quad \eta \in (0, 1)$$

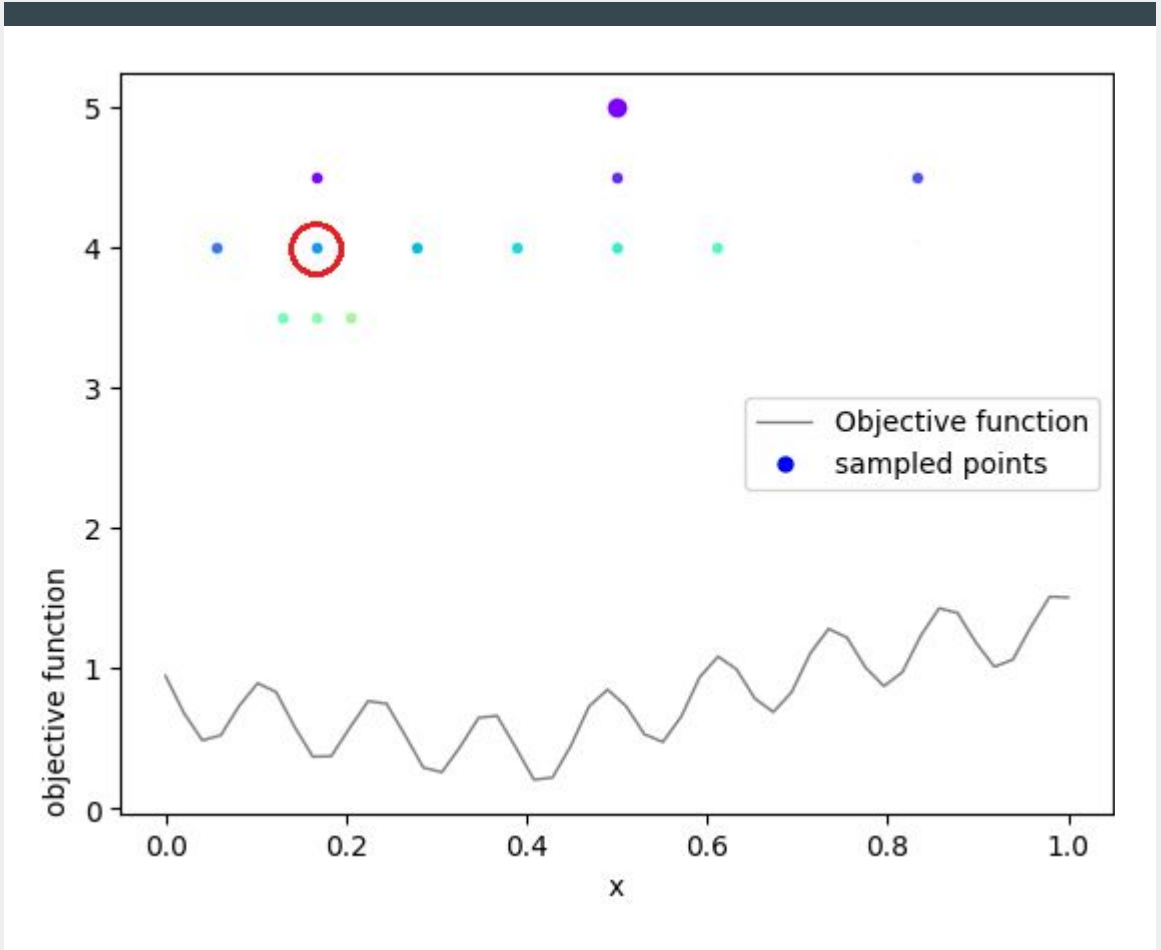
BaMSOO : 1D


Objective function
sampled points
objective function
x
0
1
2
3
4
5
0.0
0.2
0.4
0.6
0.8
1.0

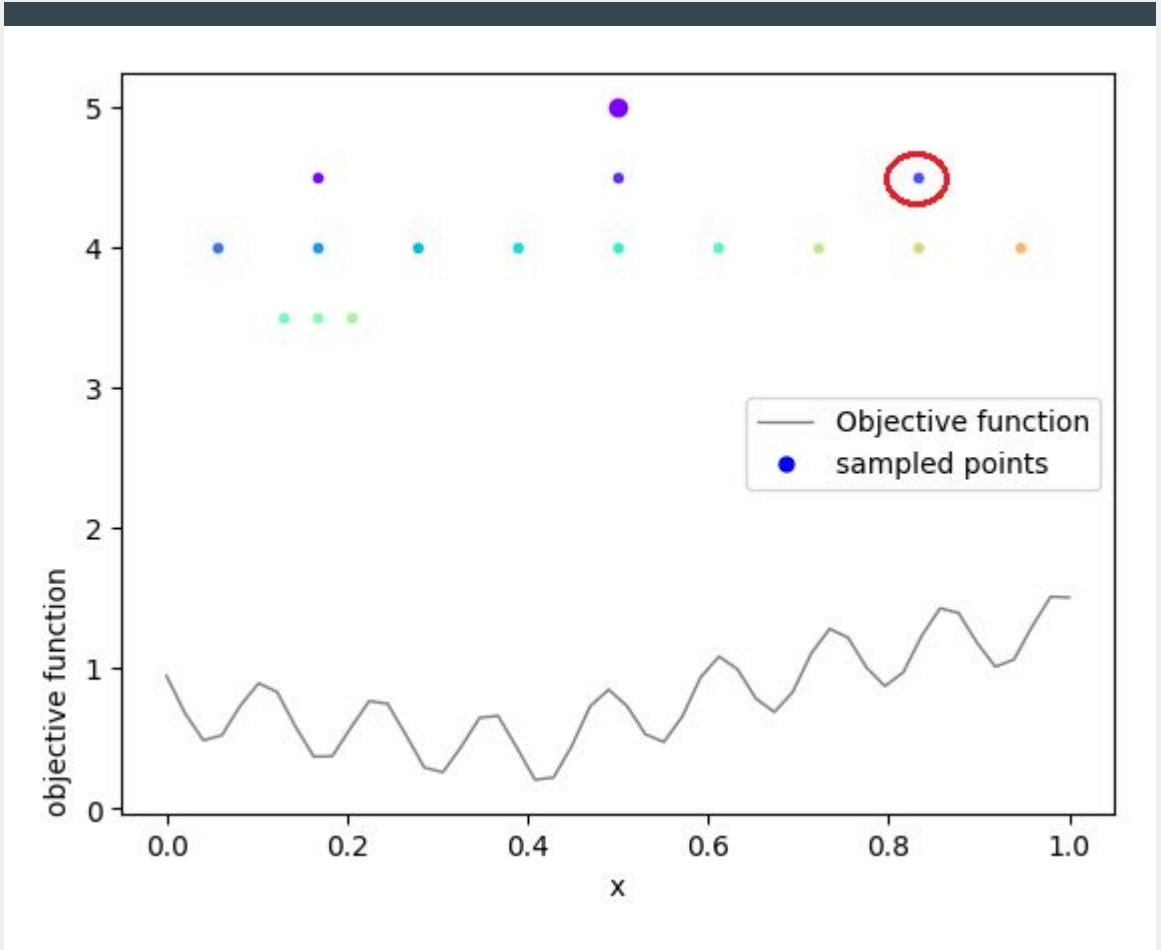
BaMSOO : 1D


5
4
3
2
1
0
objective function
0.0
0.2
0.4
0.6
0.8
1.0
x
Objective function
sampled points

BaMSOO : 1D

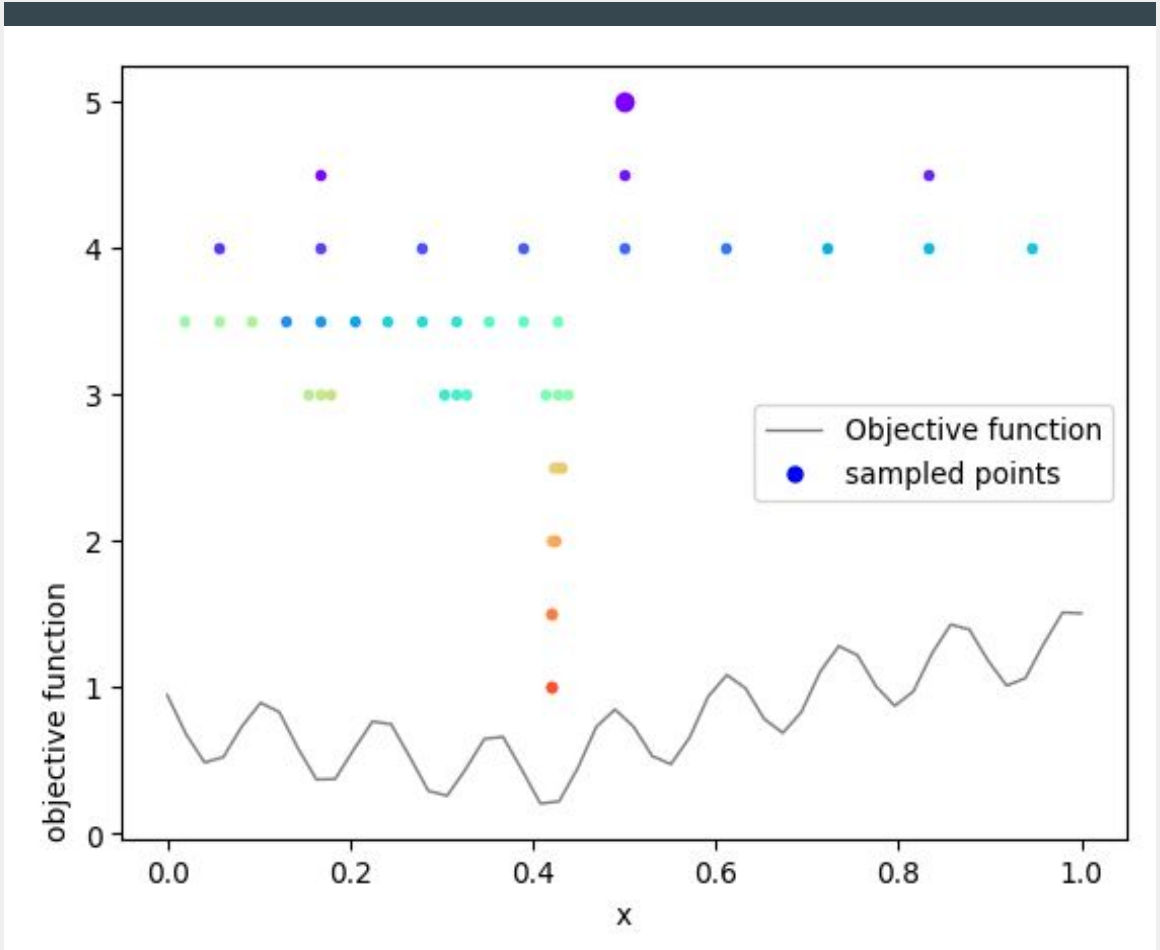

Objective function
sampled points
objective function
x
0
1
2
3
4
5
0.0
0.2
0.4
0.6
0.8
1.0

BaMSOO : 1D

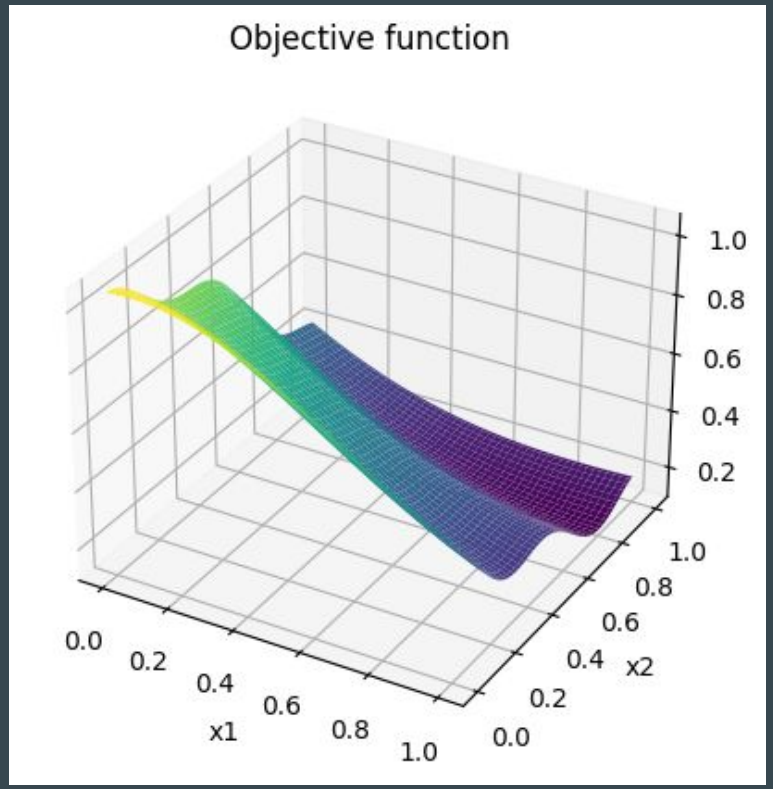

Objective function
sampled points
objective function
x
0
1
2
3
4
5
0.0
0.2
0.4
0.6
0.8
1.0

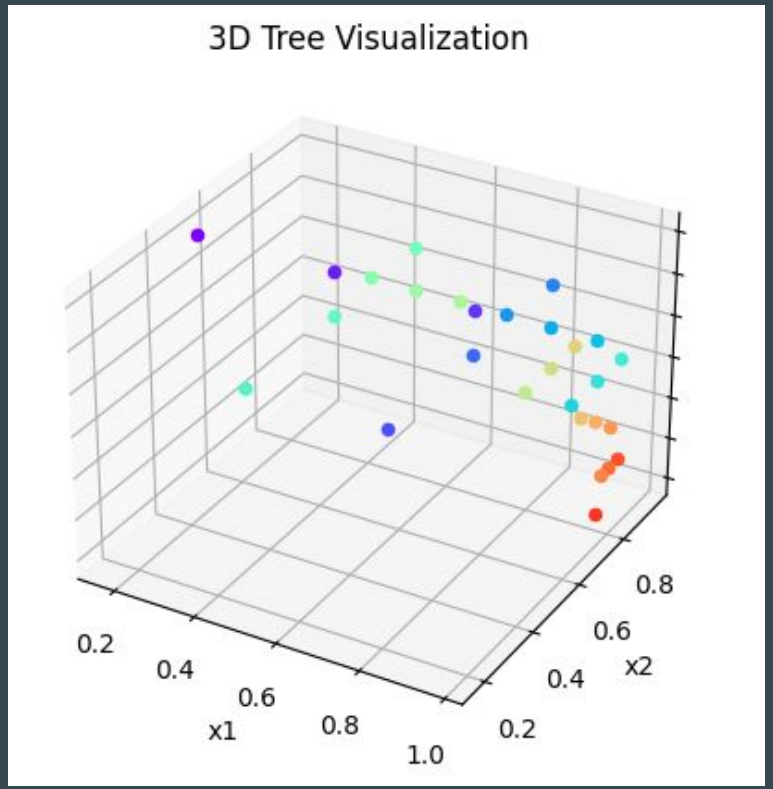
BaMSOO : 1D

```
Number of nodes in tree:  46
Number of function evaluations:  20
Max depth reached:  8
Min depth reached:  3
```



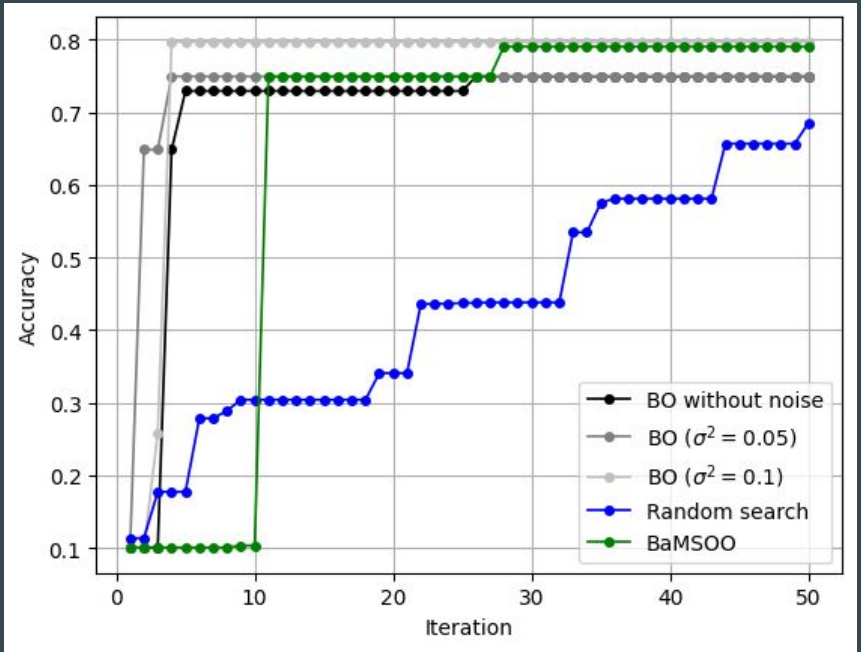
BaMSOO : 2D


Objective function
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
1.0
x1
0.0
0.2
0.4
0.6
0.8
1.0
x2

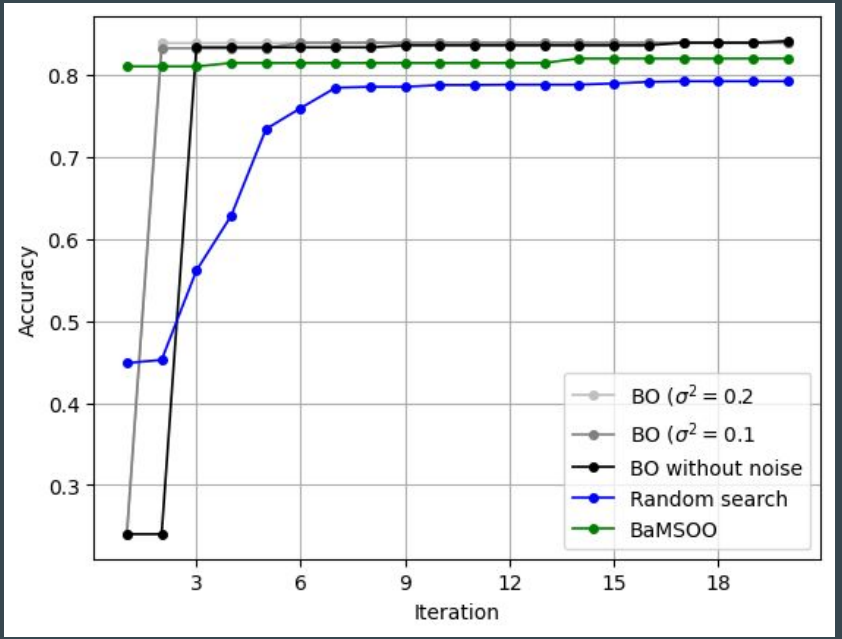

3D Tree Visualization
0.2
0.4
0.6
0.8
1.0
x1
0.2
0.4
0.6
0.8
x2

BaMSOO : SVM and NN

SVM



NN



Execution time

	SVM (2d)	SVM (5d)	NN (2d)	NN (5d)	NN (7d)
BaMSOO	89s	83s	37s	46s	124s
BO	143s	295s	108s	205s	360s

BaMSOO: Pros and Cons

- **PROS**:
  - The **optimistic optimization** allows it to **balance exploration** and **exploitation** effectively.
  - The **multi-scale** allows it to model the objective function at different scales: capture both the **global** and **local** structure of the objective function.
  - **Large-scale** optimization problems: handle high-dimensional search spaces effectively.

- **CONS**:
  - Only handles **deterministic observations**
  - **More computational resources**: it uses a multi-scale approach that requires more computation than other approaches.
  - *Merrill et al.(2021)* demonstrate that fast partitioning-based methods tend to 'flat-line' on difficult problems even when given significantly more observations, or at least equal runtime, than competing expensive algorithms.

[1] E. Merrill, A. Fern, X. Fern and N. Dolatnia, "An Empirical Study of Bayesian Optimization: Acquisition Versus Partition" *Journal of Machine Learning Research 22 (2021) 1-25*

Conclusion

Throughout this project, we delved into certain aspects of Bayesian optimization (from basic understanding of the technique to test its validity on simple ML algorithms), yet it is essential to acknowledge that our exploration merely skimmed the surface of this vast field.

Recent years have witnessed a massive growth of studies on the development of new Bayesian optimization algorithms and their applications [7]. Among the latest advancements a particular focus has been on addressing challenges like distributed data handling, data privacy, and fairness within optimization frameworks.
Despite these exciting developments, numerous open questions remain to be explored.

[7] X. Wang, Y. Jin, S. Schmitt and M. Olhofer, “Recent Advances in Bayesian Optimization”, arXiv:2206.03301v2 (2022)

Thank you for your attention

REFERENCES

[1] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, “Taking the Human Out of the Loop: A Review of Bayesian Optimization” *Proceedings of the IEEE | Vol. 104, No. 1, January 2016*

[2] E. Brochu, V. M. Cora and N. de Freitas, “A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning” *arXiv:1012.2599v1 (2010)*

[3] J. Snoek, H. Larochelle and R. P. Adams, “Practical Bayesian Optimization of Machine Learning Algorithms” *Proc. Adv. Neural Inf. Process. Syst., 2012, pp. 2951–2959.*

[4] E. C. Garrido-Merchán, D. Hernández-Lobato, “Dealing with categorical and integer-valued variables in Bayesian Optimization with Gaussian processes” *ttps://doi.org/10.1016/j.neucom.2019.11.004*

[5] Z. Wang, B. Shakibi, L. Jin and N. de Freitas, “Bayesian Multi-Scale Optimistic Optimization” *AI and Statistics, pp. 1005–1014, 2014.*

[6] E. Merrill, A. Fern, X. Fern and N. Dolatnia, “An Empirical Study of Bayesian Optimization: Acquisition Versus Partition” *Journal of Machine Learning Research 22 (2021) 1-25*

[7] X. Wang, Y. Jin, S. Schmitt and M. Olhofer, “Recent Advances in Bayesian Optimization”, *arXiv:2206.03301v2 (2022)*