Gaussian Process Based Random Search for Continuous Optimization via Simulation

Haihui Shen（沈海辉）

Sino-US Global Logistics Institute
Shanghai Jiao Tong University

Joint work with Xiuxian Wang (SJTU), Jeff Hong (Fudan), and Zhibin Jiang (SJTU)

@ 第十八届管理科学与工程学会协同创新与管理分会
安徽 合肥

2023年7月19–21日

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY
董浩云智能制造与服务管理研究院
CY TUNG Institute of Intelligent Manufacturing and Service Management
（中美物流研究院）
(Sino-US Global Logistics Institute)

Contents

Optimization via Simulation (OvS)

- Consider

$$\max_{\boldsymbol{x}\in\mathcal{X}} \; g(\boldsymbol{x}).$$

  - The form of $g(\boldsymbol{x})$ is unknown to us;
  - $g(\boldsymbol{x})$ can only be evaluated via noisy simulation observation $G(\boldsymbol{x};\omega)$ such that $g(\boldsymbol{x}) = \mathbb{E}[G(\boldsymbol{x};\omega)]$;
  - ω represents the randomness of simulation experiments;
  - It is a black-box optimization with random noises.

Optimization via Simulation (OvS)

- Consider

$$\max_{\boldsymbol{x}\in\mathcal{X}} \; g(\boldsymbol{x}).$$

  - The form of $g(\boldsymbol{x})$ is unknown to us;
  - $g(\boldsymbol{x})$ can only be evaluated via noisy simulation observation $G(\boldsymbol{x};\omega)$ such that $g(\boldsymbol{x}) = \mathbb{E}[G(\boldsymbol{x};\omega)]$;
  - ω represents the randomness of simulation experiments;
  - It is a black-box optimization with random noises.

- When $\boldsymbol{x}$ takes continuous or discrete values in $\mathcal{X}$, the problem is called continuous OvS (COvS) or discrete OvS (DOvS).

Optimization via Simulation (OvS)

- Consider

$$\max_{\boldsymbol{x}\in\mathcal{X}} \; g(\boldsymbol{x}).$$

  - The form of $g(\boldsymbol{x})$ is unknown to us;
  - $g(\boldsymbol{x})$ can only be evaluated via noisy simulation observation $G(\boldsymbol{x};\omega)$ such that $g(\boldsymbol{x}) = \mathbb{E}[G(\boldsymbol{x};\omega)]$;
  - ω represents the randomness of simulation experiments;
  - It is a black-box optimization with random noises.

- When $\boldsymbol{x}$ takes continuous or discrete values in $\mathcal{X}$, the problem is called continuous OvS (COvS) or discrete OvS (DOvS).

Optimization via Simulation (OvS)

- Consider

$$\max_{\boldsymbol{x}\in\mathcal{X}} \; g(\boldsymbol{x}).$$

  - The form of $g(\boldsymbol{x})$ is unknown to us;
  - $g(\boldsymbol{x})$ can only be evaluated via noisy simulation observation $G(\boldsymbol{x};\omega)$ such that $g(\boldsymbol{x}) = \mathbb{E}[G(\boldsymbol{x};\omega)]$;
  - ω represents the randomness of simulation experiments;
  - It is a black-box optimization with random noises.

- When $\boldsymbol{x}$ takes continuous or discrete values in $\mathcal{X}$, the problem is called continuous OvS (COvS) or discrete OvS (DOvS).
- Examples of COvS:
  - Traffic signal optimization to optimize the expected throughput of a transportation hub;
  - Parameter tuning in machine learning.

Random Search

- Random search is an important category of algorithms to solve OvS problems.

Random Search

- Random search is an important category of algorithms to solve OvS problems.
- The key of a random search algorithm is to handle three "E":
  - Exploration: Search globally in the entire domain;
  - Exploitation: Search locally near the current optimum;
  - Estimation: Estimate objective function values based on noisy simulation observations.

Random Search

- Random search is an important category of algorithms to solve OvS problems.
- The key of a random search algorithm is to handle three "E":
  - Exploration: Search globally in the entire domain;
  - Exploitation: Search locally near the current optimum;
  - Estimation: Estimate objective function values based on noisy simulation observations.
- The first two E's are tackled by the sampling distribution.

Random Search

- Random search is an important category of algorithms to solve OvS problems.
- The key of a random search algorithm is to handle three "E":
  - Exploration: Search globally in the entire domain;
  - Exploitation: Search locally near the current optimum;
  - Estimation: Estimate objective function values based on noisy simulation observations.
- The first two E's are tackled by the sampling distribution.
- Estimation can be conducted using
  - the multi-observation approach;
  - the single-observation approach.

Random Search

- Random search is an important category of algorithms to solve OvS problems.
- The key of a random search algorithm is to handle three "E":
  - Exploration: Search globally in the entire domain;
  - Exploitation: Search locally near the current optimum;
  - Estimation: Estimate objective function values based on noisy simulation observations.
- The first two E's are tackled by the sampling distribution.
- Estimation can be conducted using
  - the multi-observation approach;
    - repeatedly sample the same solution
    - convergence due to the Strong Law of Large Numbers
  - the single-observation approach.

Random Search

- Random search is an important category of algorithms to solve OvS problems.
- The key of a random search algorithm is to handle three "E":
  - Exploration: Search globally in the entire domain;
  - Exploitation: Search locally near the current optimum;
  - Estimation: Estimate objective function values based on noisy simulation observations.
- The first two E's are tackled by the sampling distribution.
- Estimation can be conducted using
  - the multi-observation approach;
    - repeatedly sample the same solution
    - convergence due to the Strong Law of Large Numbers
  - the single-observation approach.
    - sample each solution only once
    - k-nearest neighbor / shrinking-ball mechanism

Random Search

- Random search is an important category of algorithms to solve OvS problems.
- The key of a random search algorithm is to handle three "E":
  - Exploration: Search globally in the entire domain;
  - Exploitation: Search locally near the current optimum;
  - Estimation: Estimate objective function values based on noisy simulation observations.
- The first two E's are tackled by the sampling distribution.
- Estimation can be conducted using
  - the multi-observation approach;
    - repeatedly sample the same solution
    - convergence due to the Strong Law of Large Numbers
  - the single-observation approach. (preferable for COvS)
    - sample each solution only once
    - k-nearest neighbor / shrinking-ball mechanism

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

Gaussian Process Regression

- It takes a Bayesian viewpoint.
- Suppose the unknown $g(\boldsymbol{x})$ is a (random) sample path of a Gaussian process $f_{\mathcal{GP}}$ on $\mathcal{X}$, with
  - *mean function* $\mu_0 : \mathcal{X} \to \mathbb{R}$, defined by

$$\mu_0(\boldsymbol{x}) = \mathbb{E}[f_{\mathcal{GP}}(\boldsymbol{x})];$$

  - *covariance function* $k_0 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, defined by

$$k_0(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f_{\mathcal{GP}}(\boldsymbol{x}) - \mu_0(\boldsymbol{x}))(f_{\mathcal{GP}}(\boldsymbol{x}') - \mu_0(\boldsymbol{x}'))].$$

Gaussian Process Regression

- It takes a Bayesian viewpoint.
- Suppose the unknown $g(\boldsymbol{x})$ is a (random) sample path of a Gaussian process $f_{\mathcal{GP}}$ on $\mathcal{X}$, with
  - *mean function* $\mu_0 : \mathcal{X} \to \mathbb{R}$, defined by

$$\mu_0(\boldsymbol{x}) = \mathbb{E}[f_{\mathcal{GP}}(\boldsymbol{x})];$$

  - *covariance function* $k_0 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, defined by

$$k_0(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f_{\mathcal{GP}}(\boldsymbol{x}) - \mu_0(\boldsymbol{x}))(f_{\mathcal{GP}}(\boldsymbol{x}') - \mu_0(\boldsymbol{x}'))].$$

- For any $\boldsymbol{x}$, $g(\boldsymbol{x}) \sim \mathcal{N}(\mu_0(\boldsymbol{x}), k_0(\boldsymbol{x}, \boldsymbol{x}))$ (prior distribution).

Gaussian Process Regression

- It takes a Bayesian viewpoint.
- Suppose the unknown $g(\boldsymbol{x})$ is a (random) sample path of a Gaussian process $f_{\mathcal{GP}}$ on $\mathcal{X}$, with
  - *mean function* $\mu_0 : \mathcal{X} \to \mathbb{R}$, defined by
  $$\mu_0(\boldsymbol{x}) = \mathbb{E}[f_{\mathcal{GP}}(\boldsymbol{x})];$$
  - *covariance function* $k_0 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, defined by
  $$k_0(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f_{\mathcal{GP}}(\boldsymbol{x}) - \mu_0(\boldsymbol{x}))(f_{\mathcal{GP}}(\boldsymbol{x}') - \mu_0(\boldsymbol{x}'))].$$
- For any $\boldsymbol{x}$, $g(\boldsymbol{x}) \sim \mathcal{N}(\mu_0(\boldsymbol{x}), k_0(\boldsymbol{x}, \boldsymbol{x}))$ (prior distribution).
- After running simulation at $\boldsymbol{X}^n = \{\boldsymbol{x}_i\}_{i=1}^n$ with observations $\boldsymbol{G}^n = (G(\boldsymbol{x}_1), \ldots, G(\boldsymbol{x}_n))^\intercal$, how to predict $g(\boldsymbol{x})$?

Gaussian Process Regression

- Assume $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$.

Gaussian Process Regression

- Assume $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$.
- For any $\boldsymbol{x}$,

$$g(\boldsymbol{x})|\{\boldsymbol{X}^n, \boldsymbol{G}^n\} \sim \mathcal{N}(\mu_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x})),$$

$$\begin{aligned}\mu_n(\boldsymbol{x}) &:= \mu_0(\boldsymbol{x}) + k_0(\boldsymbol{x}, \boldsymbol{X}^n)[k_0(\boldsymbol{X}^n, \boldsymbol{X}^n) + \boldsymbol{\Sigma}^n]^{-1}[\boldsymbol{G}^n - \mu_0(\boldsymbol{X}^n)],\\ k_n(\boldsymbol{x}, \boldsymbol{x}) &:= k_0(\boldsymbol{x}, \boldsymbol{x}) - k_0(\boldsymbol{x}, \boldsymbol{X}^n)[k_0(\boldsymbol{X}^n, \boldsymbol{X}^n) + \boldsymbol{\Sigma}^n]^{-1} k_0(\boldsymbol{X}^n, \boldsymbol{x}),\end{aligned}$$

Gaussian Process Regression

- Assume $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$.
- For any $\boldsymbol{x}$,

$$g(\boldsymbol{x})|\{\boldsymbol{X}^n, \boldsymbol{G}^n\} \sim \mathcal{N}(\mu_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x})),$$

$$\mu_n(\boldsymbol{x}) := \mu_0(\boldsymbol{x}) + k_0(\boldsymbol{x}, \boldsymbol{X}^n)[k_0(\boldsymbol{X}^n, \boldsymbol{X}^n) + \boldsymbol{\Sigma}^n]^{-1}[\boldsymbol{G}^n - \mu_0(\boldsymbol{X}^n)],$$
$$k_n(\boldsymbol{x}, \boldsymbol{x}) := k_0(\boldsymbol{x}, \boldsymbol{x}) - k_0(\boldsymbol{x}, \boldsymbol{X}^n)[k_0(\boldsymbol{X}^n, \boldsymbol{X}^n) + \boldsymbol{\Sigma}^n]^{-1} k_0(\boldsymbol{X}^n, \boldsymbol{x}),$$

where

- $\boldsymbol{\Sigma}^n = \mathrm{diag}(\lambda^2(\boldsymbol{x}_i), \ldots, \lambda^2(\boldsymbol{x}_n))$;
- $k_0(\boldsymbol{X}^n, \boldsymbol{X}^n) = [k_0(\boldsymbol{x}_i - \boldsymbol{x}_j)]_{1 \le i,\, j \le n} \in \mathbb{R}^{n \times n}$;
- $k_0(\boldsymbol{x}, \boldsymbol{X}^n) = (k_0(\boldsymbol{x} - \boldsymbol{x}_1), \ldots, k_0(\boldsymbol{x} - \boldsymbol{x}_n)) \in \mathbb{R}^{1 \times n}$;
- $k_0(\boldsymbol{X}^n, \boldsymbol{x}) = k_0(\boldsymbol{x}, \boldsymbol{X}^n)^\mathsf{T}$.

Gaussian Process Regression

- Assume $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$.
- For any $\boldsymbol{x}$,

$$g(\boldsymbol{x})|\{\boldsymbol{X}^n, \boldsymbol{G}^n\} \sim \mathcal{N}(\mu_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x})),$$

$$\mu_n(\boldsymbol{x}) := \mu_0(\boldsymbol{x}) + k_0(\boldsymbol{x}, \boldsymbol{X}^n)[k_0(\boldsymbol{X}^n, \boldsymbol{X}^n) + \boldsymbol{\Sigma}^n]^{-1}[\boldsymbol{G}^n - \mu_0(\boldsymbol{X}^n)],$$
$$k_n(\boldsymbol{x}, \boldsymbol{x}) := k_0(\boldsymbol{x}, \boldsymbol{x}) - k_0(\boldsymbol{x}, \boldsymbol{X}^n)[k_0(\boldsymbol{X}^n, \boldsymbol{X}^n) + \boldsymbol{\Sigma}^n]^{-1} k_0(\boldsymbol{X}^n, \boldsymbol{x}),$$

where

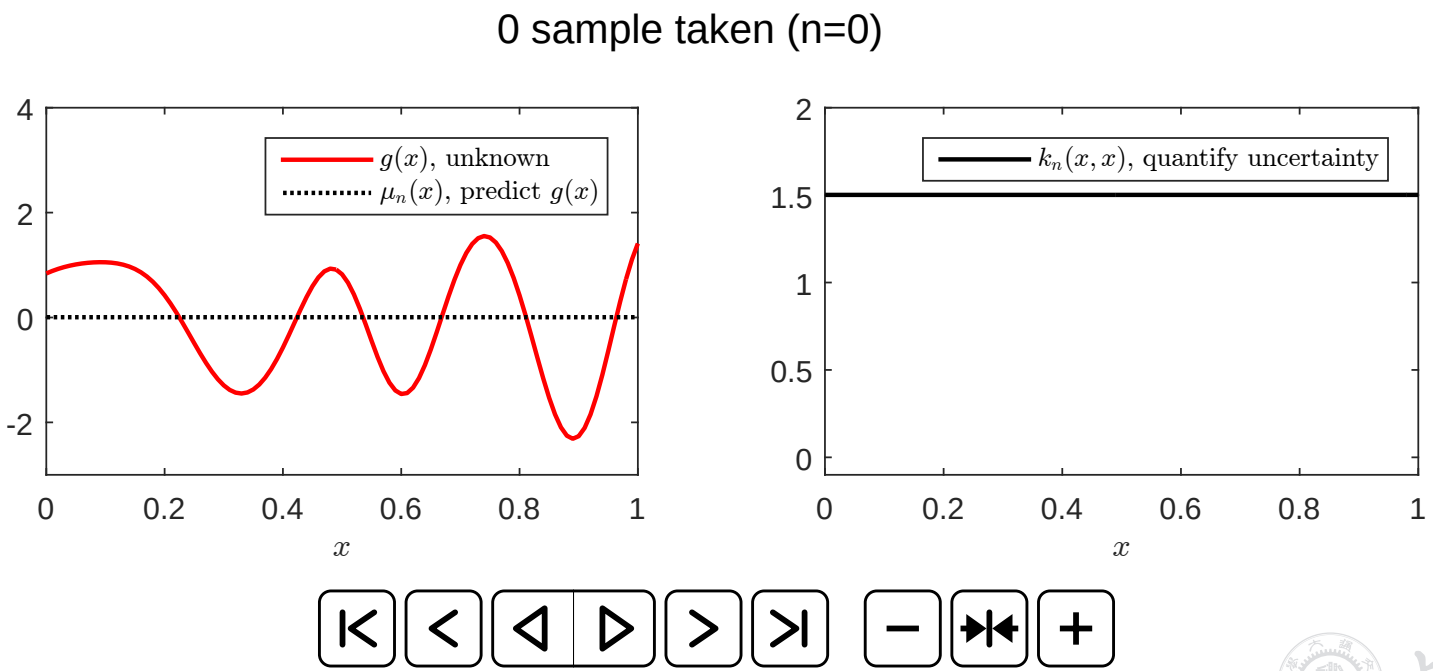
  - $\boldsymbol{\Sigma}^n = \mathrm{diag}(\lambda^2(\boldsymbol{x}_i), \ldots, \lambda^2(\boldsymbol{x}_n))$;
  - $k_0(\boldsymbol{X}^n, \boldsymbol{X}^n) = [k_0(\boldsymbol{x}_i - \boldsymbol{x}_j)]_{1 \le i,\, j \le n} \in \mathbb{R}^{n \times n}$;
  - $k_0(\boldsymbol{x}, \boldsymbol{X}^n) = (k_0(\boldsymbol{x} - \boldsymbol{x}_1), \ldots, k_0(\boldsymbol{x} - \boldsymbol{x}_n)) \in \mathbb{R}^{1 \times n}$;
  - $k_0(\boldsymbol{X}^n, \boldsymbol{x}) = k_0(\boldsymbol{x}, \boldsymbol{X}^n)^\mathsf{T}$.

- Usually, use $\mu_n(\boldsymbol{x})$ to predict $g(\boldsymbol{x})|\{\boldsymbol{X}^n, \boldsymbol{G}^n\}$, and use $k_n(\boldsymbol{x}, \boldsymbol{x})$ to quantify the uncertainty.

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

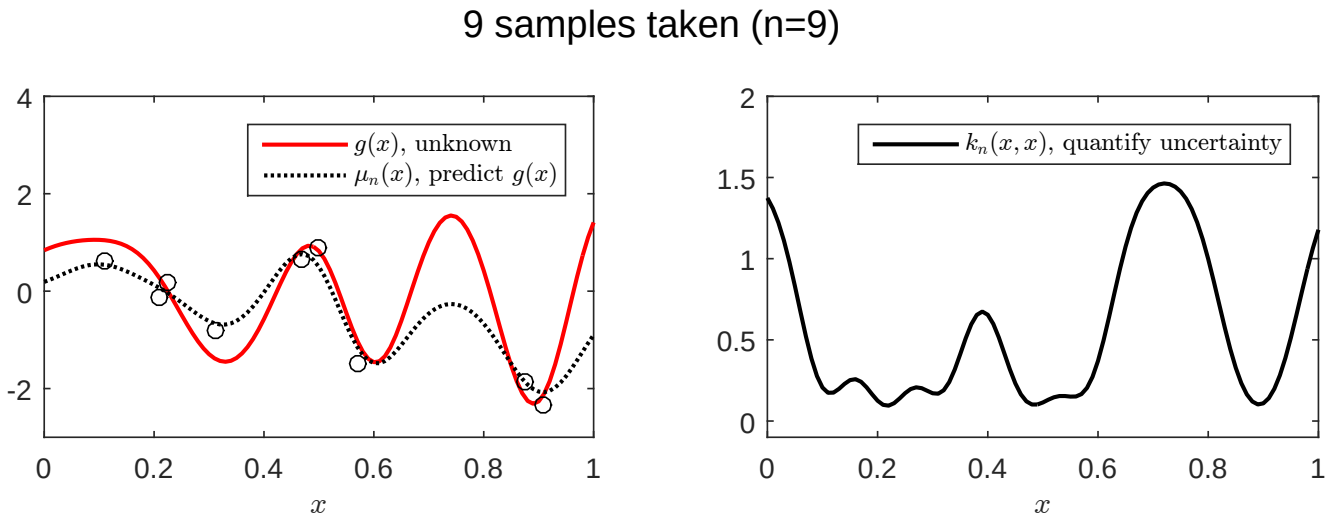
Gaussian Process Regression

- An illustration:
  - $d = 1$, $\mathcal{X} = [0, 1]$;
  - $\mu_0(x) \equiv 0$, $k_0(x, x') = 1.5 \times e^{-100(x-x')^2}$;
  - $G(x)|g(x) \sim \mathcal{N}(g(x), 0.5^2)$;
  - $\{x_i\}_{i=1}^n$ is generated from uniform$[0, 1]$.



Exploration, Exploitation, Estimation

- Recall the Gaussian process regression



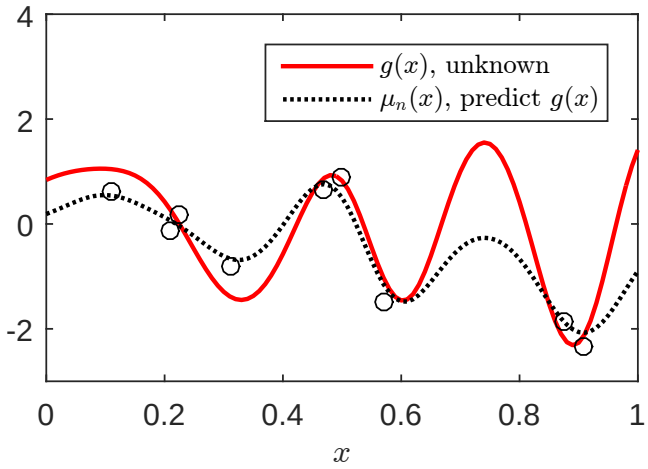
- It provides a natural way to handle the three "E".

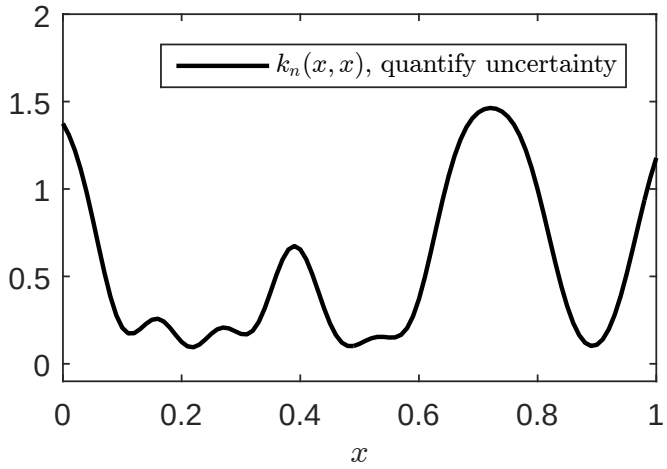
Sampling Distribution

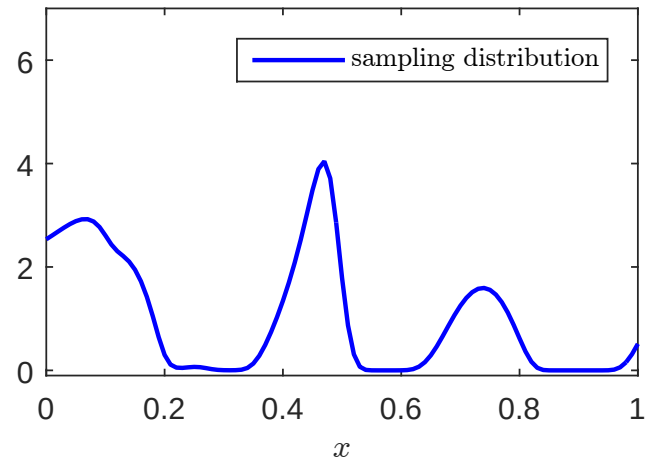
- Construct $f_n(\boldsymbol{x}) = \frac{\mathbb{P}\{Z(\boldsymbol{x})>c\}}{\int_{\mathcal{X}} \mathbb{P}\{Z(\boldsymbol{z})>c\}\mathrm{d}\boldsymbol{z}}$, $\boldsymbol{x} \in \mathcal{X}$, where
  - $c = \max_{\boldsymbol{x}\in\mathcal{X}} \mu_n(\boldsymbol{x})$;
  - $Z(\boldsymbol{x}) \sim \mathcal{N}(\mu_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x}))$.

Sampling Distribution

9 samples taken (n=9)







Sampling Distribution

- Construct $f_n(\boldsymbol{x}) = \frac{\mathbb{P}\{Z(\boldsymbol{x})>c\}}{\int_{\mathcal{X}} \mathbb{P}\{Z(\boldsymbol{z})>c\}\mathrm{d}\boldsymbol{z}}$, $\boldsymbol{x} \in \mathcal{X}$, where
  - $c = \max_{\boldsymbol{x}\in\mathcal{X}} \mu_n(\boldsymbol{x})$;
  - $Z(\boldsymbol{x}) \sim \mathcal{N}(\mu_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x}))$.
- It is a straightforward extension of the proposed sampling distribution in Sun et al. (2014, OR)[†].

[†] Sun L, Hong LJ, Hu Z (2014) Balancing exploitation and exploration in discrete optimization via simulation through a Gaussian process-based search. *Operations Research* 62(6):1416–1438.

Sampling Distribution

- Construct $f_n(\boldsymbol{x}) = \frac{\mathbb{P}\{Z(\boldsymbol{x})>c\}}{\int_{\mathcal{X}} \mathbb{P}\{Z(\boldsymbol{z})>c\}\mathrm{d}\boldsymbol{z}}$, $\boldsymbol{x} \in \mathcal{X}$, where
  - $c = \max_{\boldsymbol{x}\in\mathcal{X}} \mu_n(\boldsymbol{x})$;
  - $Z(\boldsymbol{x}) \sim \mathcal{N}(\mu_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x}))$.
- It is a straightforward extension of the proposed sampling distribution in Sun et al. (2014, OR)[†].
- However,
  - Sun et al. (2014, OR) considers DOvS problems.
  - The Gaussian process regression is only used for constructing sampling distribution, so a fast approximation (instead of the original form) is adopted.
  - Estimation is achieved with the multi-observation approach.

[†] Sun L, Hong LJ, Hu Z (2014) Balancing exploitation and exploration in discrete optimization via simulation through a Gaussian process-based search. *Operations Research* 62(6):1416–1438.

The Algorithm

Step 0 (Initialization). Impose a Gaussian process with μ_0 and k_0. Specify a $r > 0$. Set $s = 0$, $n = 0$, $\boldsymbol{X}^0 = \emptyset$ and $\boldsymbol{G}^0 = \emptyset$, and sampling distribution $f_0(\boldsymbol{x})$.

Step 1 (Sampling). Set $s = s + 1$. Sample $\boldsymbol{x}_{r(s-1)+1}, \ldots, \boldsymbol{x}_{rs}$ independently from $f_n(\boldsymbol{x})$, and obtain simulation observations $G(\boldsymbol{x}_{r(s-1)+1}), \ldots, G(\boldsymbol{x}_{rs})$.

Step 2 (Calculation). Set $n = rs$. Let $\boldsymbol{X}^n = \boldsymbol{X}^{r(s-1)} \cup \{\boldsymbol{x}_{r(s-1)+1}, \ldots, \boldsymbol{x}_{rs}\}$ and $\boldsymbol{G}^n = \left([\boldsymbol{G}^{r(s-1)}]^\intercal, G(\boldsymbol{x}_{r(s-1)+1}), \ldots, G(\boldsymbol{x}_{rs})\right)^\intercal$. Calculate $\mu_n(\boldsymbol{x})$ and $k_n(\boldsymbol{x}, \boldsymbol{x})$. Let $\boldsymbol{x}_n^* = \operatorname{argmax}_{\boldsymbol{x} \in \mathcal{X}} \mu_n(\boldsymbol{x})$. Construct sampling distribution $f_n(\boldsymbol{x})$.

Step 3 (Stopping). If the stopping condition is not met, go to Step 1; otherwise, stop and output $\boldsymbol{x}_n^*$ and $\mu_n(\boldsymbol{x}_n^*)$.

For A General Framework

- The convergence analysis is for a general framework of Gaussian process based random search algorithms for COvS problems.
- More specifically, it only requires that the constructed sampling distribution $f_n(\boldsymbol{x})$ is lower bounded.
  - Not necessary to utilize the information contained in $\mu_n(\boldsymbol{x})$ and $k_n(\boldsymbol{x}, \boldsymbol{x})$;
  - Can be either static or dynamic.

Global Convergence

- Assumptions:
  - A1. [feasible region] $\mathcal{X}$ is a compact set in $\mathbb{R}^d$, and $\mathrm{cl}(\mathrm{int}(\mathcal{X})) = \mathcal{X}$.
  - A2. [simulation noise] $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$, and $\lambda^2(\boldsymbol{x})$ is bounded on $\mathcal{X}$.
  - A3. [$g(\boldsymbol{x})$ & GP] $g(\boldsymbol{x})$ is a sample path of a Gaussian process, whose $\mu_0(\boldsymbol{x})$ is continuous and $k_0(\boldsymbol{x}, \boldsymbol{x}')$ satisfies certain regularity conditions.
  - A4. [sampling distribution] $f_n(\boldsymbol{x}) \geq \alpha$ for all n and $\boldsymbol{x} \in \mathcal{X}$.

Global Convergence

- Assumptions:
  - A1. [feasible region] $\mathcal{X}$ is a compact set in $\mathbb{R}^d$, and $\mathrm{cl}(\mathrm{int}(\mathcal{X})) = \mathcal{X}$.
  - A2. [simulation noise] $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$, and $\lambda^2(\boldsymbol{x})$ is bounded on $\mathcal{X}$.
  - A3. [$g(\boldsymbol{x})$ & GP] $g(\boldsymbol{x})$ is a sample path of a Gaussian process, whose $\mu_0(\boldsymbol{x})$ is continuous and $k_0(\boldsymbol{x}, \boldsymbol{x}')$ satisfies certain regularity conditions.
  - A4. [sampling distribution] $f_n(\boldsymbol{x}) \geq \alpha$ for all n and $\boldsymbol{x} \in \mathcal{X}$.
- Extra notations:
  - $g^* = \max_{\boldsymbol{x} \in \mathcal{X}} g(\boldsymbol{x})$;
  - $\mathcal{X}^* = \mathrm{argmax}_{\boldsymbol{x} \in \mathcal{X}} g(\boldsymbol{x})$;
  - $d(\boldsymbol{x}, \mathcal{A}) = \inf_{\boldsymbol{x}' \in \mathcal{A}} \|\boldsymbol{x} - \boldsymbol{x}'\|$.

Global Convergence

- Assumptions:
  - A1. [feasible region] $\mathcal{X}$ is a compact set in $\mathbb{R}^d$, and $\mathrm{cl}(\mathrm{int}(\mathcal{X})) = \mathcal{X}$.
  - A2. [simulation noise] $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$, and $\lambda^2(\boldsymbol{x})$ is bounded on $\mathcal{X}$.
  - A3. [$g(\boldsymbol{x})$ & GP] $g(\boldsymbol{x})$ is a sample path of a Gaussian process, whose $\mu_0(\boldsymbol{x})$ is continuous and $k_0(\boldsymbol{x}, \boldsymbol{x}')$ satisfies certain regularity conditions.
  - A4. [sampling distribution] $f_n(\boldsymbol{x}) \geq \alpha$ for all n and $\boldsymbol{x} \in \mathcal{X}$.
- Extra notations:
  - $g^* = \max_{\boldsymbol{x} \in \mathcal{X}} g(\boldsymbol{x})$;
  - $\mathcal{X}^* = \mathrm{argmax}_{\boldsymbol{x} \in \mathcal{X}} g(\boldsymbol{x})$;
  - $d(\boldsymbol{x}, \mathcal{A}) = \inf_{\boldsymbol{x}' \in \mathcal{A}} \|\boldsymbol{x} - \boldsymbol{x}'\|$.
- Main result:

Theorem 1. $\mu_n(\boldsymbol{x}_n^*) \to g^*$ and $d(\boldsymbol{x}_n^*, \mathcal{X}^*) \to 0$ almost surely as $n \to \infty$.

Rate of Convergence

- Assumptions:
  - A1'. [feasible region] $\mathcal{X} \subset \mathbb{R}^d$ is a bounded convex set with nonempty interior.
  - A2. [simulation noise] $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$, and $\lambda^2(\boldsymbol{x})$ is bounded on $\mathcal{X}$.
  - A3'. [$g(\boldsymbol{x})$ & GP] $g(\boldsymbol{x})$ is a sample path of a Gaussian process $f_{\mathcal{GP}}$, and the first-order derivatives of $f_{\mathcal{GP}}$ are stationary Gaussian processes with almost-sure continuous sample paths.
  - A4. [sampling distribution] $f_n(\boldsymbol{x}) \geq \alpha$ for all n and $\boldsymbol{x} \in \mathcal{X}$.

Rate of Convergence

- Assumptions:

  A1'. [feasible region] $\mathcal{X} \subset \mathbb{R}^d$ is a bounded convex set with nonempty interior.

  A2. [simulation noise] $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \lambda^2(\boldsymbol{x}))$, and $\lambda^2(\boldsymbol{x})$ is bounded on $\mathcal{X}$.

  A3'. [$g(\boldsymbol{x})$ & GP] $g(\boldsymbol{x})$ is a sample path of a Gaussian process $f_{\mathcal{GP}}$, and the first-order derivatives of $f_{\mathcal{GP}}$ are stationary Gaussian processes with almost-sure continuous sample paths.

  A4. [sampling distribution] $f_n(\boldsymbol{x}) \geq \alpha$ for all n and $\boldsymbol{x} \in \mathcal{X}$.

- Main result:

Theorem 2. There exists a constant $C_0 > 0$ such that, as $n \to \infty$,

$$\mathbb{P}\left\{|\mu_n(\boldsymbol{x}_n^*) - g^*| > \left(\frac{16C_0 \log n}{n^{\kappa(n)}}\right)^{1/2}\right\} \to 0, \text{ where } \kappa(n) = \frac{2}{d+2} - \frac{b \log\log n}{\log n}.$$

That is, the rate of convergence is $\tilde{O}_p(n^{-1/(d+2)})$.

Estimation of $\lambda^2(\boldsymbol{x})$ and GP Parameters

- If the simulation noises are homoscedastic, i.e., $\lambda^2(\boldsymbol{x}) \equiv \lambda^2$:
  - Estimate λ^2, together with GP parameters, using MLE method.
  - Only estimate these parameters once, to alleviate the computational burden.

Estimation of $\lambda^2(\boldsymbol{x})$ and GP Parameters

- If the simulation noises are homoscedastic, i.e., $\lambda^2(\boldsymbol{x}) \equiv \lambda^2$:
  - Estimate λ^2, together with GP parameters, using MLE method.
  - Only estimate these parameters once, to alleviate the computational burden.
- If the simulation noises are heteroscedastic:
  - To deal with the situation that only single observation is available on each sampled point, a kernel-based sample variance estimator is adopted to estimate $\{\lambda^2(\boldsymbol{x}_i)\}_{i=1}^n$.
  - GP parameters are estimated using MLE method after the variances are estimated.
  - GP parameters are estimated only once, while $\{\lambda^2(\boldsymbol{x}_i)\}_{i=1}^n$ are updated repeatedly when new observations are obtained.

Sampling from the Sampling Distribution

- Recall the sampling distribution $f_n(\boldsymbol{x}) = \frac{\mathbb{P}\{Z(\boldsymbol{x})>c\}}{\int_{\mathcal{X}} \mathbb{P}\{Z(\boldsymbol{z})>c\}\mathrm{d}\boldsymbol{z}}$.

Sampling from the Sampling Distribution

- Recall the sampling distribution $f_n(\boldsymbol{x}) = \frac{\mathbb{P}\{Z(\boldsymbol{x})>c\}}{\int_{\mathcal{X}} \mathbb{P}\{Z(\boldsymbol{z})>c\}\mathrm{d}\boldsymbol{z}}$.
- Acceptance-Rejection Sampling Scheme (exact):
  S1. Generate $\boldsymbol{y}$ from $\mathrm{uniform}(\mathcal{X})$ and u from $\mathrm{uniform}[0,1]$.
  S2. If $u \leq 2\,\mathbb{P}\{Z(\boldsymbol{y}) > c\}$, return $\boldsymbol{y}$; otherwise, go to S1.

Sampling from the Sampling Distribution

- Recall the sampling distribution $f_n(\boldsymbol{x}) = \frac{\mathbb{P}\{Z(\boldsymbol{x})>c\}}{\int_{\mathcal{X}} \mathbb{P}\{Z(\boldsymbol{z})>c\}\mathrm{d}\boldsymbol{z}}$.
- Acceptance-Rejection Sampling Scheme (exact):
  - S1. Generate $\boldsymbol{y}$ from $\mathrm{uniform}(\mathcal{X})$ and u from $\mathrm{uniform}[0,1]$.
  - S2. If $u \leq 2\,\mathbb{P}\{Z(\boldsymbol{y}) > c\}$, return $\boldsymbol{y}$; otherwise, go to S1.
- Markov Chain Coordinate Sampling Scheme (approximate):
  - S0. Specify iteration number T. Let $t = 0$, $\boldsymbol{y} = \boldsymbol{y}_0$.
  - S1. Let $t = t + 1$. Generate j uniformly from $\{1, \ldots, d\}$. Let $l(\boldsymbol{y}, j)$ be the line that passes through $\boldsymbol{y}$ and parallel to the y_j coordinate axis. Sample a point on $l(\boldsymbol{y}, j) \cap \mathcal{X}$ uniformly, whose j-th coordinate is denoted as b. Set $\boldsymbol{z} = \boldsymbol{y}$ and $z_j = b$.
  - S2. Generate u from $\mathrm{uniform}[0,1]$. If $u \leq \frac{\mathbb{P}\{Z(\boldsymbol{z})>c\}}{\mathbb{P}\{Z(\boldsymbol{y})>c\}}$, set $\boldsymbol{y} = \boldsymbol{z}$.
  - S2. If $t = T$, return $\boldsymbol{y}$; otherwise, go to S1.

Solving $\boldsymbol{x}_n^* = \operatorname{argmax}_{\boldsymbol{x}\in\mathcal{X}} \mu_n(\boldsymbol{x})$ in Each Iteration

- It requires not only for outputting the current solution in each iteration, but also for constructing sampling distribution in each iteration (key reason).

Solving $\boldsymbol{x}_n^* = \operatorname{argmax}_{\boldsymbol{x}\in\mathcal{X}} \mu_n(\boldsymbol{x})$ in Each Iteration

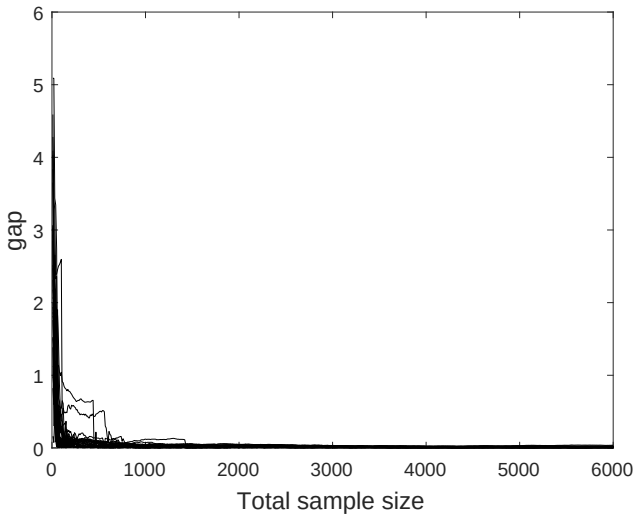
- It requires not only for outputting the current solution in each iteration, but also for constructing sampling distribution in each iteration (key reason).
- Numerical methods:
  - When the dimension is low, simply evaluate $\mu_n(\boldsymbol{x})$ on a dense grid within $\mathcal{X}$.
  - When the dimension is high, compute $\hat{\boldsymbol{x}}_n^\dagger = \operatorname{argmax}_{\boldsymbol{x}\in\boldsymbol{X}^n} \mu_n(\boldsymbol{x})$, and use some nonlinear optimization solvers with $\hat{\boldsymbol{x}}_n^\dagger$ as initial solution.

Solving $\boldsymbol{x}_n^* = \operatorname{argmax}_{\boldsymbol{x}\in\mathcal{X}} \mu_n(\boldsymbol{x})$ in Each Iteration

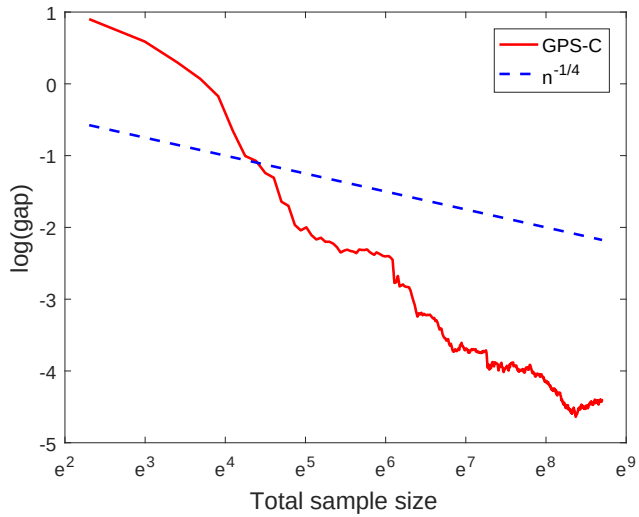
- It requires not only for outputting the current solution in each iteration, but also for constructing sampling distribution in each iteration (key reason).
- Numerical methods:
  - When the dimension is low, simply evaluate $\mu_n(\boldsymbol{x})$ on a dense grid within $\mathcal{X}$.
  - When the dimension is high, compute $\hat{\boldsymbol{x}}_n^\dagger = \operatorname{argmax}_{\boldsymbol{x}\in\boldsymbol{X}^n} \mu_n(\boldsymbol{x})$, and use some nonlinear optimization solvers with $\hat{\boldsymbol{x}}_n^\dagger$ as initial solution.
- Revise the original algorithm:
  - Simply use $\hat{\boldsymbol{x}}_n^\dagger$ instead of $\boldsymbol{x}_n^*$.
  - Under Assumption A3', the aforementioned global convergence and rate of convergence still hold.

$g(\boldsymbol{x})$ Generated from GP with Known Parameters

- Setting:
  - $d = 2$, $\mathcal{X} = [0, 1]^2$;
  - $\mu_0(\boldsymbol{x}) \equiv 1$, $k_0(\boldsymbol{x}, \boldsymbol{x}') = 4 \times e^{-80\|\boldsymbol{x}-\boldsymbol{x}'\|^2}$;
  - $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), 0.5^2)$.



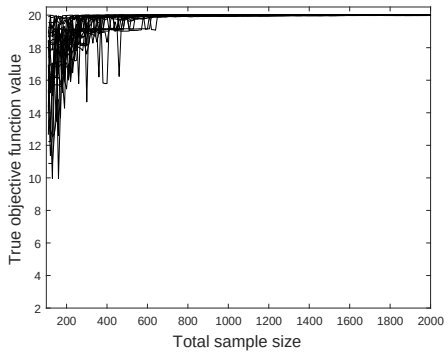
(a) The optimality gap



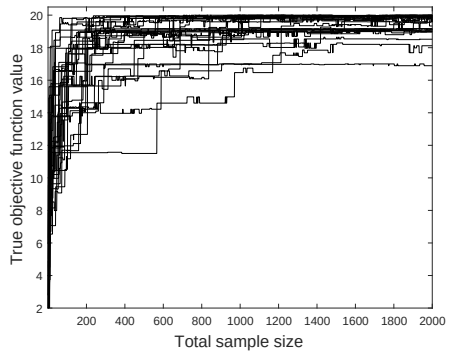
(b) The rate of convergence

Given and Deterministic $g(\boldsymbol{x})$

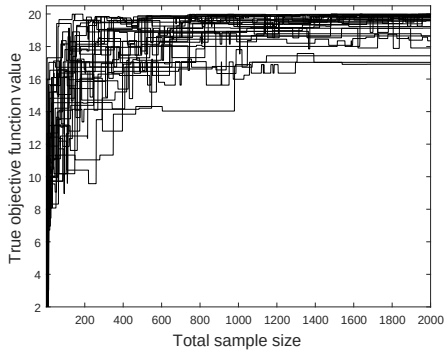
- Setting:
  - $d = 2$, $\mathcal{X} = [0, 100]^2$;
  - $g(\boldsymbol{x}) = 10 \cdot \frac{\sin^6(0.05\pi x_1)}{2^{2((x_1-90)/50)^2}} + 10 \cdot \frac{\sin^6(0.05\pi x_2)}{2^{2((x_2-90)/50)^2}}$;
  - $G(\boldsymbol{x})|g(\boldsymbol{x}) \sim \mathcal{N}(g(\boldsymbol{x}), \frac{1}{4}g(\boldsymbol{x}))$ (variances treated as unknown).



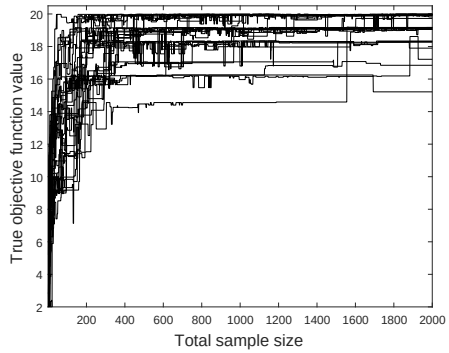
(a) GPS-C algorithm



(b) ASR algorithm



(c) IHR-SO algorithm



(d) AP-SO algorithm

Concluding Remarks

- We propose a framework of Gaussian process based random search algorithms for COvS problems.
  - It uses Gaussian process regression for estimation (single-observation approach);
  - It allows flexible sampling distribution to balance exploration and exploitation (a good choice is to utilize the Gaussian process regression again);
- For general sampling distributions, the global convergence and rate of convergence are established.
  - By exploring the properties of Gaussian process regression;
  - Some intermediate results and techniques have potential to be applied in other applications of Gaussian process regression.
- Some implementation issues are addressed.

Thank you for your attention!

Haihui Shen（沈海辉）
shenhaihui@sjtu.edu.cn

July 20, 2023