MEM6804 Modeling and Simulation for Logistics & Supply Chain 物流与供应链建模与仿真

Theory | Analysis

Lecture 10: Output Analysis III: Optimization

SHEN Haihui 沈海辉

Sino-US Global Logistics Institute
Shanghai Jiao Tong University

shenhaihui.github.io/teaching/mem6804f
shenhaihui@sjtu.edu.cn

Spring 2021 (full-time)

上海交通大学 SHANGHAI JIAO TONG UNIVERSITY
董浩云航运与物流研究院 CY TUNG Institute of Maritime and Logistics
中美物流研究院（工程系统管理研究院） Sino-US Global Logistics Institute (Institute of Industrial & System Engineering)

Contents

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

1. Introduction
   - Definition
   - Types
2. White-box OvS Problem
   - Sample Average Approximation
3. Black-box COvS Problem
   - Gradient Descent
   - Stochastic Approximation
4. Black-box DOvS Problem
   - Simulated Annealing
   - COMPASS
5. Usage in Softwares

- **Optimization via Simulation (OvS)**, or, simply called Simulation Optimization (SO):

$$\min_{\boldsymbol{x}\in\mathcal{X}} \; g(\boldsymbol{x}) := \mathbb{E}[G(\boldsymbol{x},\xi)],$$

where $\mathcal{X} \subset \mathbb{R}^d$ is the feasible set, and $g : \mathcal{X} \to \mathbb{R}$ is a deterministic function whose values can only be evaluated with noisy observations.

- **Optimization via Simulation (OvS)**, or, simply called Simulation Optimization (SO):

$$\min_{\boldsymbol{x}\in\mathcal{X}} \ g(\boldsymbol{x}) := \mathbb{E}[G(\boldsymbol{x},\xi)],$$

  where $\mathcal{X} \subset \mathbb{R}^d$ is the feasible set, and $g : \mathcal{X} \to \mathbb{R}$ is a deterministic function whose values can only be evaluated with noisy observations.

- Given $\boldsymbol{x}$, $G(\boldsymbol{x},\xi)$ is a random variable (the randomness is from ξ), and the distribution of $G(\boldsymbol{x},\xi)$ is unknown.

- **Optimization via Simulation (OvS)**, or, simply called Simulation Optimization (SO):

$$\min_{\boldsymbol{x}\in\mathcal{X}} \; g(\boldsymbol{x}) := \mathbb{E}[G(\boldsymbol{x},\xi)],$$

  where $\mathcal{X}\subset\mathbb{R}^d$ is the feasible set, and $g:\mathcal{X}\to\mathbb{R}$ is a deterministic function whose values can only be evaluated with noisy observations.

- Given $\boldsymbol{x}$, $G(\boldsymbol{x},\xi)$ is a random variable (the randomness is from ξ), and the distribution of $G(\boldsymbol{x},\xi)$ is unknown.

- Given $\boldsymbol{x}$, realizations of $G(\boldsymbol{x},\xi)$ can be observed by running simulation, or more generally, taking samples.

- OvS Problem can be classified into two types according to whether the explicit form of $G(\boldsymbol{x}, \xi)$ is available.

- OvS Problem can be classified into two types according to whether the explicit form of $G(\boldsymbol{x}, \xi)$ is available.
- **White-box**: The explicit form of $G(\boldsymbol{x}, \xi)$ is available.
  - Example: $G(x, \xi) = \sin\big((x - \xi)^2\big)$, where the distribution of ξ is unknown.

- OvS Problem can be classified into two types according to whether the explicit form of $G(\boldsymbol{x}, \xi)$ is available.
- **White-box**: The explicit form of $G(\boldsymbol{x}, \xi)$ is available.
  - Example: $G(x, \xi) = \sin\big((x - \xi)^2\big)$, where the distribution of ξ is unknown.
- **Black-box**: The explicit form of $G(\boldsymbol{x}, \xi)$ is not available and it is embedded in a simulation model.
  - Example: Let $G(\boldsymbol{x}, \xi)$ be the waiting time of a customer in a complex queueing network, where $\boldsymbol{x}$ represents the configuration parameters.

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

- OvS Problem can be classified into three types according to the feasible set $\mathcal{X}$.

- OvS Problem can be classified into three types according to the feasible set $\mathcal{X}$.
- **Ranking and selection (R&S)**: $\mathcal{X}$ is a set of relatively small number of (discrete) solutions.

- OvS Problem can be classified into three types according to the feasible set $\mathcal{X}$.
- **Ranking and selection (R&S)**: $\mathcal{X}$ is a set of relatively small number of (discrete) solutions.
- **Discrete OvS (DOvS)**: $\mathcal{X}$ is a discrete set, with huge or even countably infinite number of solutions.
  - One can also view R&S problem as a special type of DOvS problem.

- OvS Problem can be classified into three types according to the feasible set $\mathcal{X}$.
- **Ranking and selection (R&S)**: $\mathcal{X}$ is a set of relatively small number of (discrete) solutions.
- **Discrete OvS (DOvS)**: $\mathcal{X}$ is a discrete set, with huge or even countably infinite number of solutions.
    - One can also view R&S problem as a special type of DOvS problem.
- **Continuous OvS (COvS)**: $\mathcal{X}$ is a continuous set, hence there exits uncountably infinite number of solutions.

White-box OvS Problem

- For white-box OvS problems, we can use the sample average approximation.

White-box OvS Problem

- For white-box OvS problems, we can use the sample average approximation.
- Of course, those algorithms designed for black-box OvS problems can also be applied to white-box OvS problems.

- Suppose that we have an iid sample $\{\xi_1, \ldots, \xi_n\}$ of ξ.
- To solve $\min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}) := \mathbb{E}[G(\boldsymbol{x}, \xi)]$, we try to solve

$$\min_{\boldsymbol{x}\in\mathcal{X}} \ \widehat{g}_n(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^{n} G(\boldsymbol{x}, \xi_i),$$

with any suitable deterministic optimization algorithm (after $\{\xi_1, \ldots, \xi_n\}$ is realized).

- Suppose that we have an iid sample $\{\xi_1, \ldots, \xi_n\}$ of ξ.
- To solve $\min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}) \coloneqq \mathbb{E}[G(\boldsymbol{x}, \xi)]$, we try to solve

$$\min_{\boldsymbol{x}\in\mathcal{X}} \ \widehat{g}_n(\boldsymbol{x}) \coloneqq \frac{1}{n}\sum_{i=1}^{n} G(\boldsymbol{x}, \xi_i),$$

  with any suitable deterministic optimization algorithm (after $\{\xi_1, \ldots, \xi_n\}$ is realized).
- This method is called Sample Average Approximation (SAA); see Kim et al. (2015) for a review.

- Suppose that we have an iid sample $\{\xi_1, \ldots, \xi_n\}$ of ξ.
- To solve $\min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}) := \mathbb{E}[G(\boldsymbol{x}, \xi)]$, we try to solve

$$\min_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^{n} G(\boldsymbol{x}, \xi_i),$$

  with any suitable deterministic optimization algorithm (after $\{\xi_1, \ldots, \xi_n\}$ is realized).
- This method is called Sample Average Approximation (SAA); see Kim et al. (2015) for a review.
- Clearly, for finite n, $\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})$ is a random variable (before $\{\xi_1, \ldots, \xi_n\}$ is realized), and it is not strictly equal to $\min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x})$.

- Indeed, one can prove that

$$\mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \le \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}).$$

- Indeed, one can prove that

$$\mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \le \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}).$$

Proof.

- Indeed, one can prove that

$$\mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \leq \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}).$$

Proof. For any $\boldsymbol{y} \in \mathcal{X}$,

$$\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x}) \leq \widehat{g}_n(\boldsymbol{y}) \Longrightarrow \mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \leq \mathbb{E}[\widehat{g}_n(\boldsymbol{y})] = g(\boldsymbol{y}).$$

- Indeed, one can prove that

$$\mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \leq \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}).$$

Proof. For any $\boldsymbol{y} \in \mathcal{X}$,

$$\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x}) \leq \widehat{g}_n(\boldsymbol{y}) \Longrightarrow \mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \leq \mathbb{E}[\widehat{g}_n(\boldsymbol{y})] = g(\boldsymbol{y}).$$

Minimizing the right-hand side over all $\boldsymbol{y} \in \mathcal{X}$ completes the proof. ■

- Indeed, one can prove that

$$\mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \leq \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}).$$

Proof. For any $\boldsymbol{y} \in \mathcal{X}$,

$$\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x}) \leq \widehat{g}_n(\boldsymbol{y}) \Longrightarrow \mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \leq \mathbb{E}[\widehat{g}_n(\boldsymbol{y})] = g(\boldsymbol{y}).$$

Minimizing the right-hand side over all $\boldsymbol{y} \in \mathcal{X}$ completes the proof. ■

- Moreover, it can also be shown that

$$\mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x})\right] \leq \mathbb{E}\left[\inf_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_{n+1}(\boldsymbol{x})\right] \leq \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}).$$

(Prove it as an exercise)

- What can we say if we continuously increase sample size n?

- What can we say if we continuously increase sample size n?
- It will be **reassuring** if we know that the obtained solution will be closer and closer to the true solution, as we increase sample size n.

- What can we say if we continuously increase sample size n?
- It will be **reassuring** if we know that the obtained solution will be closer and closer to the true solution, as we increase sample size n.
- Formally, we are seeking for a **convergence** guarantee for SAA method.

- For set $\mathcal{A} \subset \mathbb{R}^d$, the distance from $\boldsymbol{x} \in \mathbb{R}^d$ to $\mathcal{A}$ is defined as

$$\operatorname{dist}(\boldsymbol{x}, \mathcal{A}) := \inf_{\boldsymbol{y} \in \mathcal{A}} \|\boldsymbol{x} - \boldsymbol{y}\|,$$

where $\|\cdot\|$ denotes the Euclidean distance.

- For set $\mathcal{A} \subset \mathbb{R}^d$, the distance from $\boldsymbol{x} \in \mathbb{R}^d$ to $\mathcal{A}$ is defined as

$$\operatorname{dist}(\boldsymbol{x}, \mathcal{A}) := \inf_{\boldsymbol{y} \in \mathcal{A}} \|\boldsymbol{x} - \boldsymbol{y}\|,$$

  where $\|\cdot\|$ denotes the Euclidean distance.

- For sets $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^d$, the deviation from $\mathcal{A}$ to $\mathcal{B}$ is defined as

$$D(\mathcal{A}, \mathcal{B}) := \sup_{\boldsymbol{x} \in \mathcal{A}} \operatorname{dist}(\boldsymbol{x}, \mathcal{B}).$$

- For set $\mathcal{A} \subset \mathbb{R}^d$, the distance from $\boldsymbol{x} \in \mathbb{R}^d$ to $\mathcal{A}$ is defined as

$$\operatorname{dist}(\boldsymbol{x}, \mathcal{A}) := \inf_{\boldsymbol{y} \in \mathcal{A}} \|\boldsymbol{x} - \boldsymbol{y}\|,$$

  where $\|\cdot\|$ denotes the Euclidean distance.

- For sets $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^d$, the deviation from $\mathcal{A}$ to $\mathcal{B}$ is defined as

$$D(\mathcal{A}, \mathcal{B}) := \sup_{\boldsymbol{x} \in \mathcal{A}} \operatorname{dist}(\boldsymbol{x}, \mathcal{B}).$$

- Let

$$\mathcal{S} := \operatorname*{argmin}_{\boldsymbol{x} \in \mathcal{X}} g(\boldsymbol{x}),$$

$$\widehat{\mathcal{S}}_n := \operatorname*{argmin}_{\boldsymbol{x} \in \mathcal{X}} \ \widehat{g}_n(\boldsymbol{x}).$$

Convergence of SAA (Theorem 5.3 of Shapiro et al. (2009))

Suppose that

1. $\mathcal{X}$ is a compact set;
2. $g(x)$ is finite valued and continuous on $\mathcal{X}$;
3. $\mathbb{P}\{\widehat{g}_n(\boldsymbol{x}) \to g(\boldsymbol{x})$ uniformly in $\boldsymbol{x} \in \mathcal{X}\} = 1$;
4. $\mathbb{P}\{\widehat{\mathcal{S}}_n$ is nonempty for n large enough$\} = 1$;

Then, as $n \to \infty$,

$$\min_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x}) \xrightarrow{a.s.} \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}), \text{ and } D(\widehat{\mathcal{S}}_n, \mathcal{S}) \xrightarrow{a.s.} 0.$$

Convergence of SAA (Theorem 5.3 of Shapiro et al. (2009))

Suppose that

1. $\mathcal{X}$ is a compact set;
2. $g(x)$ is finite valued and continuous on $\mathcal{X}$;
3. $\mathbb{P}\{\widehat{g}_n(\boldsymbol{x}) \to g(\boldsymbol{x})$ uniformly in $\boldsymbol{x} \in \mathcal{X}\} = 1$;
4. $\mathbb{P}\{\widehat{\mathcal{S}}_n$ is nonempty for n large enough$\} = 1$;

Then, as $n \to \infty$,

$$\min_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x}) \xrightarrow{a.s.} \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x}), \text{ and } D(\widehat{\mathcal{S}}_n, \mathcal{S}) \xrightarrow{a.s.} 0.$$

Besides, if $\mathcal{S} = \{\boldsymbol{x}^*\}$ is a singleton, then for any $\widehat{\boldsymbol{x}}_n \in \widehat{\mathcal{S}}_n$,

$$\widehat{\boldsymbol{x}}_n \xrightarrow{a.s.} \boldsymbol{x}^*, \text{ as } n \to \infty.$$

- **How fast** does the SAA solution converge to the true solution?

- **How fast** does the SAA solution converge to the true solution?
- Formally, it's known as the **rate of convergence**.

- **How fast** does the SAA solution converge to the true solution?
- Formally, it's known as the **rate of convergence**.
- Under certain regularity conditions, one may show that

$$\left|\min_{\boldsymbol{x}\in\mathcal{X}} \widehat{g}_n(\boldsymbol{x}) - \min_{\boldsymbol{x}\in\mathcal{X}} g(\boldsymbol{x})\right| = O_p(n^{-1/2}),$$

and given $\mathcal{S} = \{\boldsymbol{x}^*\}$ is a singleton,

$$\|\widehat{\boldsymbol{x}}_n - \boldsymbol{x}^*\| = O_p(n^{-1/2}).$$

Black-box COvS Problem

- Main types of algorithms for black-box COvS problems:
  - random search; see Andradóttir (2015) for a review;
  - stochastic approximation; see Chau and Fu (2015) for a review;
  - surrogate-based methods; see Hong and Zhang (2021) for a review.

Black-box COvS Problem

- Main types of algorithms for black-box COvS problems:
    - random search; see Andradóttir (2015) for a review;
    - stochastic approximation; see Chau and Fu (2015) for a review;
    - surrogate-based methods; see Hong and Zhang (2021) for a review.
- Stochastic Approximation (SA) was proposed by Robbins and Monro (1951) and Kiefer and Wolfowitz (1952).
- SA can be viewed as a stochastic version of the gradient descent (or called steepest descent) algorithm, so it is also called stochastic gradient descent.

- Gradient descent is a first-order iterative optimization algorithm for finding a local minimum of a differentiable (deterministic) function:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \gamma \nabla g(\boldsymbol{x}_k),$$

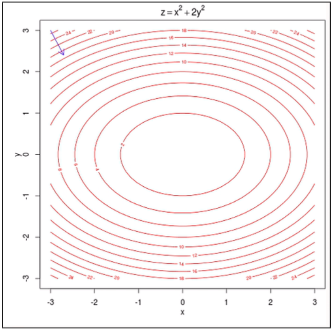
where $\nabla g(\boldsymbol{x})$ is the gradient and $\gamma > 0$ is the step size.

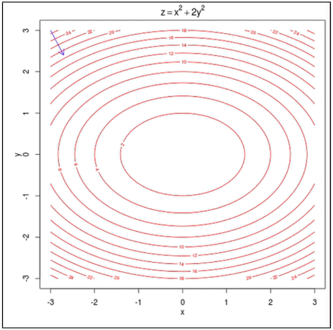
- Gradient descent is a first-order iterative optimization algorithm for finding a local minimum of a differentiable (deterministic) function:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \gamma \nabla g(\boldsymbol{x}_k),$$

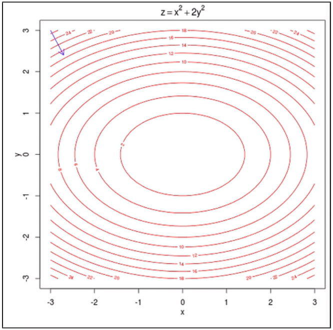
  where $\nabla g(\boldsymbol{x})$ is the gradient and $\gamma > 0$ is the step size.

- If the minimization problem is constrained, say the feasible set $\mathcal{X} \subset \mathbb{R}^d$ is convex and compact, one can easily add a projection $\Pi_{\mathcal{X}}(\boldsymbol{x})$ mapping $\boldsymbol{x} \notin \mathcal{X}$ back into $\mathcal{X}$.


$z = x^2 + 2y^2$
x
y



- The value of the step size γ is allowed to change at every iteration, and with proper choice, convergence to a local minimizer (say, $\boldsymbol{x}^*$) can be guaranteed, i.e., $\boldsymbol{x}_k \to \boldsymbol{x}^*$.



- The value of the step size γ is allowed to change at every iteration, and with proper choice, convergence to a local minimizer (say, $\boldsymbol{x}^*$) can be guaranteed, i.e., $\boldsymbol{x}_k \to \boldsymbol{x}^*$.
- Under certain regularity conditions, one can show that $|g(\boldsymbol{x}_k) - g(\boldsymbol{x}^*)| = O(k^{-1})$ for unconstraied problem with constant γ.

- SA as a stochastic version of the gradient ascent:

$$\boldsymbol{X}_{k+1} = \Pi_{\mathcal{X}} \left(\boldsymbol{X}_k - a_k \widehat{\nabla} g(\boldsymbol{X}_k) \right),$$

where $\Pi_{\mathcal{X}}$ is the projection, $\{a_k\}_{k\geq 1}$ is a deterministic positive sequence for step size, and $\widehat{\nabla} g(\boldsymbol{x})$ is an estimmator of the gradient $\nabla g(\boldsymbol{x})$.

- SA as a stochastic version of the gradient ascent:

$$\boldsymbol{X}_{k+1} = \Pi_{\mathcal{X}} \left(\boldsymbol{X}_k - a_k \widehat{\nabla} g(\boldsymbol{X}_k) \right),$$

  where $\Pi_{\mathcal{X}}$ is the projection, $\{a_k\}_{k\geq 1}$ is a deterministic positive sequence for step size, and $\widehat{\nabla} g(\boldsymbol{x})$ is an estimmator of the gradient $\nabla g(\boldsymbol{x})$.

- In some simulation experiments, unbiased $\widehat{\nabla} g(\boldsymbol{x})$ is available,[†] then it is the Robbins-Monro (RM) type SA (Robbins and Monro 1951).

[†] When we observe $G(\boldsymbol{x}, \xi)$, we will also observe $\widehat{\nabla} g(\boldsymbol{x}, \xi)$ at the same time such that $\mathbb{E}[\widehat{\nabla} g(\boldsymbol{x}, \xi)] = \nabla g(\boldsymbol{x})$.

- SA as a stochastic version of the gradient ascent:

$$\boldsymbol{X}_{k+1} = \Pi_{\mathcal{X}}\left(\boldsymbol{X}_k - a_k \widehat{\nabla} g(\boldsymbol{X}_k)\right),$$

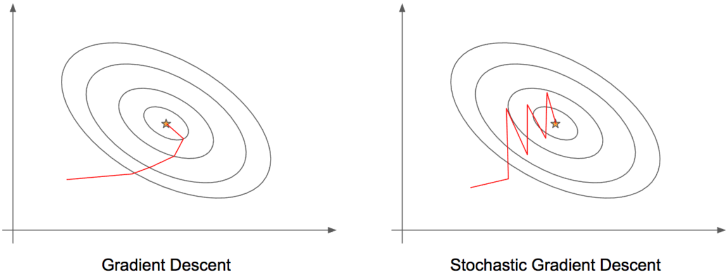
  where $\Pi_{\mathcal{X}}$ is the projection, $\{a_k\}_{k\geq 1}$ is a deterministic positive sequence for step size, and $\widehat{\nabla} g(\boldsymbol{x})$ is an estimmator of the gradient $\nabla g(\boldsymbol{x})$.

- In some simulation experiments, unbiased $\widehat{\nabla} g(\boldsymbol{x})$ is available,† then it is the Robbins-Monro (RM) type SA (Robbins and Monro 1951).

- Otherwise, $\widehat{\nabla} g(\boldsymbol{x})$ needs to be constructed with certain indirect method (thus biased), then it is the Kiefer-Wolfowitz (KW) type SA Kiefer and Wolfowitz (1952).

† When we observe $G(\boldsymbol{x}, \xi)$, we will also observe $\widehat{\nabla} g(\boldsymbol{x}, \xi)$ at the same time such that $\mathbb{E}[\widehat{\nabla} g(\boldsymbol{x}, \xi)] = \nabla g(\boldsymbol{x})$.

- Gradient descent vs SA (i.e., stochastic gradient desecent):



- Construct $\widehat{\nabla} g(\boldsymbol{X}_k)$ via symmetric (or central) finite difference:

$$\widehat{\nabla} g\left(\boldsymbol{X}_k\right) := \left(g_1\left(\boldsymbol{X}_k\right), \ldots, g_d\left(\boldsymbol{X}_k\right)\right)^{\mathsf{T}},$$

where

$$g_i\left(\boldsymbol{X}_k\right) := \frac{G(\boldsymbol{X}_k + c_k \boldsymbol{e}_i) - G(\boldsymbol{X}_k - c_k \boldsymbol{e}_i)}{2c_k},$$

$\boldsymbol{e}_i$ denotes a $d \times 1$ vector whose ith element is one and other elements are all zeros, $i = 1, \ldots, d$, and $\{c_k\}_{k \geq 1}$ is a deterministic positive sequence.

- Construct $\widehat{\nabla} g(\boldsymbol{X}_k)$ via symmetric (or central) finite difference:

$$\widehat{\nabla} g\left(\boldsymbol{X}_k\right) := \left(g_1\left(\boldsymbol{X}_k\right), \ldots, g_d\left(\boldsymbol{X}_k\right)\right)^{\intercal},$$

where

$$g_i\left(\boldsymbol{X}_k\right) := \frac{G(\boldsymbol{X}_k + c_k \boldsymbol{e}_i) - G(\boldsymbol{X}_k - c_k \boldsymbol{e}_i)}{2c_k},$$

$\boldsymbol{e}_i$ denotes a $d \times 1$ vector whose ith element is one and other elements are all zeros, $i = 1, \ldots, d$, and $\{c_k\}_{k \geq 1}$ is a deterministic positive sequence.

- It requires $2d$ **aditional** simulation runs (samples) to compute $\widehat{\nabla} g(\boldsymbol{X}_k)$.

- Let $\mathcal{M}$ denote the set of local optimal solutions:

$$\mathcal{M} := \left\{ \boldsymbol{x} \in \mathcal{X} : \; g(\boldsymbol{x}) \leq \min_{\boldsymbol{y} \in \mathcal{B}(\boldsymbol{x})} g(\boldsymbol{y}) \right\},$$

where $\mathcal{B}(\boldsymbol{x}) \subset \mathcal{X}$ denotes a neighborhood of $\boldsymbol{x} \in \mathcal{X}$.

- Let $\mathcal{M}$ denote the set of local optimal solutions:

$$\mathcal{M} := \left\{ \boldsymbol{x} \in \mathcal{X} : \ g(\boldsymbol{x}) \leq \min_{\boldsymbol{y} \in \mathcal{B}(\boldsymbol{x})} g(\boldsymbol{y}) \right\},$$

where $\mathcal{B}(\boldsymbol{x}) \subset \mathcal{X}$ denotes a neighborhood of $\boldsymbol{x} \in \mathcal{X}$.

Local Convergence of SA (Theorem 3 of Blum (1954))

Suppose that

1. $g(x)$ satisfies certain regularity conditions;
2. $\mathrm{Var}(G(\boldsymbol{x}, \xi)) \leq \sigma^2 < \infty$;
3. $\lim_{k\to\infty} c_k = 0$, $\sum_{k=1}^{\infty} a_k = \infty$, $\sum_{k=1}^{\infty} a_k c_k < \infty$, and $\sum_{k=1}^{\infty} a_k^2 c_k^{-2} < \infty$.

Then, for KW type SA with symmetric difference gradient estimator, $\mathrm{dist}(\boldsymbol{X}_k, \mathcal{M}) \xrightarrow{a.s.} 0$ as $k \to \infty$.

- Uunder certain conditions, for $\boldsymbol{x}^* \in \mathcal{M}$ such that $\boldsymbol{X}_k \xrightarrow{a.s.} \boldsymbol{x}^*$, RM type SA can reach $O_p(k^{-1/2})$ rate of convergence, i.e.,

$$\|\boldsymbol{X}_k - \boldsymbol{x}^*\| = O_p(k^{-1/2}),$$

while KW type SA can reach $O_p(k^{-1/3})$ rate of convergence.

- Uunder certain conditions, for $\boldsymbol{x}^* \in \mathcal{M}$ such that $\boldsymbol{X}_k \xrightarrow{a.s.} \boldsymbol{x}^*$, RM type SA can reach $O_p(k^{-1/2})$ rate of convergence, i.e.,

$$\|\boldsymbol{X}_k - \boldsymbol{x}^*\| = O_p(k^{-1/2}),$$

  while KW type SA can reach $O_p(k^{-1/3})$ rate of convergence.

- Note that the above order is in terms of the iteration number k, rather than the number of simulation runs (sample size).

- Uunder certain conditions, for $\boldsymbol{x}^* \in \mathcal{M}$ such that $\boldsymbol{X}_k \xrightarrow{a.s.} \boldsymbol{x}^*$, RM type SA can reach $O_p(k^{-1/2})$ rate of convergence, i.e.,

$$\|\boldsymbol{X}_k - \boldsymbol{x}^*\| = O_p(k^{-1/2}),$$

  while KW type SA can reach $O_p(k^{-1/3})$ rate of convergence.

- Note that the above order is in terms of the iteration number k, rather than the number of simulation runs (sample size).

- If in terms of the sample size n, the rate of convergence of KW type SA is $O_p((n/d)^{-1/3})$, which depends on the dimensionality d.

- Simultaneous perturbation stochastic approximation (SPSA):

$$\widehat{\nabla} g\left(\boldsymbol{X}_k\right) := \left(g_1\left(\boldsymbol{X}_k\right), \ldots, g_d\left(\boldsymbol{X}_k\right)\right)^\intercal,$$

where

$$g_i\left(\boldsymbol{X}_k\right) := \frac{G(\boldsymbol{X}_k + c_k \boldsymbol{B}_k) - G(\boldsymbol{X}_k - c_k \boldsymbol{B}_k)}{2 c_k B_{k,\,i}},$$

$\boldsymbol{B}_k := (B_{k,\,1}, \ldots, B_{k,\,d})^\intercal$, and $B_{k,\,i} = 1$ or -1 with probability $1/2$.

- Simultaneous perturbation stochastic approximation (SPSA):

$$\widehat{\nabla} g\left(\boldsymbol{X}_k\right) := \left(g_1\left(\boldsymbol{X}_k\right), \ldots, g_d\left(\boldsymbol{X}_k\right)\right)^\intercal,$$

where

$$g_i\left(\boldsymbol{X}_k\right) := \frac{G(\boldsymbol{X}_k + c_k \boldsymbol{B}_k) - G(\boldsymbol{X}_k - c_k \boldsymbol{B}_k)}{2 c_k B_{k,\,i}},$$

$\boldsymbol{B}_k := (B_{k,\,1}, \ldots, B_{k,\,d})^\intercal$, and $B_{k,\,i} = 1$ or -1 with probability $1/2$.

- It requires only 2 **aditional** simulation runs (samples) to compute $\widehat{\nabla} g(\boldsymbol{X}_k)$, no matter what d is.

- Simultaneous perturbation stochastic approximation (SPSA):

$$\widehat{\nabla} g\left(\boldsymbol{X}_k\right) := \left(g_1\left(\boldsymbol{X}_k\right), \ldots, g_d\left(\boldsymbol{X}_k\right)\right)^\intercal,$$

  where

$$g_i\left(\boldsymbol{X}_k\right) := \frac{G(\boldsymbol{X}_k + c_k \boldsymbol{B}_k) - G(\boldsymbol{X}_k - c_k \boldsymbol{B}_k)}{2 c_k B_{k,\,i}},$$

  $\boldsymbol{B}_k := (B_{k,\,1}, \ldots, B_{k,\,d})^\intercal$, and $B_{k,\,i} = 1$ or -1 with probability $1/2$.

- It requires only 2 **aditional** simulation runs (samples) to compute $\widehat{\nabla} g(\boldsymbol{X}_k)$, no matter what d is.

- SPSA can reach $O_p(n^{-1/3})$ rate of convergence in terms of the sample size n.

Black-box DOvS Problem

- Many black-box DOvS algorithms are based on random search; see Hong et al. (2015) for a review.

Black-box DOvS Problem

- Many black-box DOvS algorithms are based on random search; see Hong et al. (2015) for a review.
- The framework of random search:
  - Initialization:
  - At Iteration k:
    - Sampling:
    - Evaluation:
    - Updating:

SHANGHAI JIAO TONG UNIVERSITY

Black-box DOvS Problem

- Many black-box DOvS algorithms are based on random search; see Hong et al. (2015) for a review.
- The framework of random search:
  - Initialization: Arbitrarily choose $\boldsymbol{x}_0^* \in \mathcal{X}$; set the information set (that keeps visited solutions and their corresponding observations) $\mathcal{F}_0$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling:
    - Evaluation:
    - Updating:

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Black-box DOvS Problem

- Many black-box DOvS algorithms are based on random search; see Hong et al. (2015) for a review.
- The framework of random search:
  - Initialization: Arbitrarily choose $\boldsymbol{x}_0^* \in \mathcal{X}$; set the information set (that keeps visited solutions and their corresponding observations) $\mathcal{F}_0$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Choose the estimation set $\mathcal{E} \subset \mathcal{X}$ (that contains solutions at which simulation will be run); some or all of the solutions in $\mathcal{E}$ are randomly sampled from $\mathcal{X}$ with distribution determined by information $\mathcal{F}_k$.
    - Evaluation:
    - Updating:

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Black-box DOvS Problem

- Many black-box DOvS algorithms are based on random search; see Hong et al. (2015) for a review.
- The framework of random search:
  - Initialization: Arbitrarily choose $\boldsymbol{x}_0^* \in \mathcal{X}$; set the information set (that keeps visited solutions and their corresponding observations) $\mathcal{F}_0$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Choose the estimation set $\mathcal{E} \subset \mathcal{X}$ (that contains solutions at which simulation will be run); some or all of the solutions in $\mathcal{E}$ are randomly sampled from $\mathcal{X}$ with distribution determined by information $\mathcal{F}_k$.
    - Evaluation: For each $\boldsymbol{x} \in \mathcal{E}$, spend simulation effort according to certain rule determined by $\mathcal{F}_k$ and $\mathcal{E}$.
    - Updating:

Black-box DOvS Problem

- Many black-box DOvS algorithms are based on random search; see Hong et al. (2015) for a review.
- The framework of random search:
  - Initialization: Arbitrarily choose $\boldsymbol{x}_0^* \in \mathcal{X}$; set the information set (that keeps visited solutions and their corresponding observations) $\mathcal{F}_0$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Choose the estimation set $\mathcal{E} \subset \mathcal{X}$ (that contains solutions at which simulation will be run); some or all of the solutions in $\mathcal{E}$ are randomly sampled from $\mathcal{X}$ with distribution determined by information $\mathcal{F}_k$.
    - Evaluation: For each $\boldsymbol{x} \in \mathcal{E}$, spend simulation effort according to certain rule determined by $\mathcal{F}_k$ and $\mathcal{E}$.
    - Updating: Update $\mathcal{F}_{k+1}$; choose some $\boldsymbol{x}_{k+1}^*$ as the current best solution based on certain estimator; set $k \leftarrow k + 1$.

- The simulated annealing algorithm dates back to the pioneering work by Metropolis et al. (1953).
  - It studied how in the physical annealing process, particles of a solid arrange themselves into thermal equibibrium at a given temperature.

- The simulated annealing algorithm dates back to the pioneering work by Metropolis et al. (1953).
  - It studied how in the physical annealing process, particles of a solid arrange themselves into thermal equibibrium at a given temperature.
- A large body of literature has developed the simulated annealing algorithm to solve deterministic global optimization problems over **finite** set; important works include Kirkpatrick et al. (1983), Mitra et al. (1986), Hajek (1988), etc.

- The simulated annealing algorithm dates back to the pioneering work by Metropolis et al. (1953).
  - It studied how in the physical annealing process, particles of a solid arrange themselves into thermal equibibrium at a given temperature.
- A large body of literature has developed the simulated annealing algorithm to solve deterministic global optimization problems over **finite** set; important works include Kirkpatrick et al. (1983), Mitra et al. (1986), Hajek (1988), etc.
- Later, the simulated annealing was extended to solve black-box DOvS problems over **finite** set; important works include Bulgak and Sander (1988), Gelfand and Mitter (1989), Alrefaei and Andradóttir (1999), etc.

- Let $\mathcal{B}(\boldsymbol{x}) \subset \mathcal{X}$ denote a neighborhood† of $\boldsymbol{x} \in \mathcal{X}$.

†The neighborhood structer can be quite different in discrete optimization compared to continuous optimization!

- Let $\mathcal{B}(\boldsymbol{x}) \subset \mathcal{X}$ denote a neighborhood[†] of $\boldsymbol{x} \in \mathcal{X}$.
- $\mathcal{B}(\boldsymbol{x})$ is carefully desined such that, for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, $\boldsymbol{y}$ is reachable from $\boldsymbol{x}$.
  - That is, there exists a finite sequence $\boldsymbol{x} = \boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell = \boldsymbol{y}$ such that $\boldsymbol{x}_{i+1} \in \mathcal{B}(\boldsymbol{x}_i)$, $i = 0, 1, \ldots, \ell - 1$.

[†] The neighborhood structer can be quite different in discrete optimization compared to continuous optimization!

- Let $\mathcal{B}(\boldsymbol{x}) \subset \mathcal{X}$ denote a neighborhood[†] of $\boldsymbol{x} \in \mathcal{X}$.
- $\mathcal{B}(\boldsymbol{x})$ is carefully desined such that, for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, $\boldsymbol{y}$ is reachable from $\boldsymbol{x}$.
  - That is, there exists a finite sequence $\boldsymbol{x} = \boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell = \boldsymbol{y}$ such that $\boldsymbol{x}_{i+1} \in \mathcal{B}(\boldsymbol{x}_i)$, $i = 0, 1, \ldots, \ell - 1$.
- Define transition probability $R(\boldsymbol{x}, \boldsymbol{y})$, where $R : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ and $R(\boldsymbol{x}, \boldsymbol{y}) > 0 \iff y \in \mathcal{B}(\boldsymbol{x})$.

[†] The neighborhood structer can be quite different in discrete optimization compared to continuous optimization!

- Let $\mathcal{B}(\boldsymbol{x}) \subset \mathcal{X}$ denote a neighborhood† of $\boldsymbol{x} \in \mathcal{X}$.
- $\mathcal{B}(\boldsymbol{x})$ is carefully desined such that, for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}$, $\boldsymbol{y}$ is reachable from $\boldsymbol{x}$.
  - That is, there exists a finite sequence $\boldsymbol{x} = \boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\ell = \boldsymbol{y}$ such that $\boldsymbol{x}_{i+1} \in \mathcal{B}(\boldsymbol{x}_i)$, $i = 0, 1, \ldots, \ell - 1$.
- Define transition probability $R(\boldsymbol{x}, \boldsymbol{y})$, where $R : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ and $R(\boldsymbol{x}, \boldsymbol{y}) > 0 \iff y \in \mathcal{B}(\boldsymbol{x})$.
- Let $\{t_k\}_{k \geq 1}$ be a positive sequence of numbers, which is konwn as the temperature.

† The neighborhood structer can be quite different in discrete optimization compared to continuous optimization!

- Simulated annealing algorithm for deterministic optimization:
  - Initialization:
  - At Iteration k:
    - Sampling:
    - Evaluation: No need in the deterministic optimization.
    - Updating:

- Simulated annealing algorithm for deterministic optimization:
  - Initialization: Arbitrarily choose $\boldsymbol{X}_0 \in \mathcal{X}$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling:
    - Evaluation: No need in the deterministic optimization.
    - Updating:

- Simulated annealing algorithm for deterministic optimization:
  - Initialization: Arbitrarily choose $\boldsymbol{X}_0 \in \mathcal{X}$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Sample a candidate solution $\boldsymbol{Y}_{k+1} \in \mathcal{B}(\boldsymbol{X}_k)$ according to distribution $R(\boldsymbol{X}_k, \cdot)$, i.e.,

$$\mathbb{P}(\boldsymbol{Y}_{k+1} = \boldsymbol{y} | \boldsymbol{X}_k = \boldsymbol{x}) = R(\boldsymbol{x}, \boldsymbol{y}).$$

    - Evaluation: No need in the deterministic optimization.
    - Updating:

- Simulated annealing algorithm for deterministic optimization:
  - Initialization: Arbitrarily choose $\boldsymbol{X}_0 \in \mathcal{X}$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Sample a candidate solution $\boldsymbol{Y}_{k+1} \in \mathcal{B}(\boldsymbol{X}_k)$ according to distribution $R(\boldsymbol{X}_k, \cdot)$, i.e.,

$$\mathbb{P}(\boldsymbol{Y}_{k+1} = \boldsymbol{y} | \boldsymbol{X}_k = \boldsymbol{x}) = R(\boldsymbol{x}, \boldsymbol{y}).$$

    - Evaluation: No need in the deterministic optimization.
    - Updating: Let

$$\boldsymbol{X}_{k+1} := \begin{cases} \boldsymbol{Y}_{k+1}, & \text{with probability } \exp\left\{\frac{-[g(\boldsymbol{Y}_{k+1}) - g(\boldsymbol{X}_k)]^+}{t_{k+1}}\right\}, \\ \boldsymbol{X}_k, & \text{otherwise;} \end{cases}$$

set $k \leftarrow k + 1$.

- To ensuer the simulated annealing algorithm for deterministic optimization is globally convergent, i.e.,

$$\mathrm{dist}(\boldsymbol{X}_k, \mathcal{S}) \xrightarrow{a.s.} 0, \text{ as } k \to \infty \ ,$$

Hajek (1988, Theorem 1) gives a sufficient condition.

- To ensuer the simulated annealing algorithm for deterministic optimization is globally convergent, i.e.,

$$\operatorname{dist}(\boldsymbol{X}_k, \mathcal{S}) \xrightarrow{a.s.} 0, \text{ as } k \to \infty ,$$

Hajek (1988, Theorem 1) gives a sufficient condition.

1. $R(\boldsymbol{x}, \boldsymbol{y})$ satisfies weak reversibility; a sufficient example is that

$$R(\boldsymbol{x}, \boldsymbol{y}) := \begin{cases} \frac{1}{|\mathcal{B}(\boldsymbol{x})|}, & \text{if } \boldsymbol{y} \in \mathcal{B}(\boldsymbol{x}), \\ 0, & \text{otherwise}, \end{cases}$$

with symmetric neighborhood, i.e., $\boldsymbol{y} \in \mathcal{B}(\boldsymbol{x}) \iff \boldsymbol{x} \in \mathcal{B}(\boldsymbol{y})$.

- To ensuer the simulated annealing algorithm for deterministic optimization is globally convergent, i.e.,

$$\operatorname{dist}(\boldsymbol{X}_k, \mathcal{S}) \xrightarrow{a.s.} 0, \text{ as } k \to \infty \ ,$$

Hajek (1988, Theorem 1) gives a sufficient condition.

1. $R(\boldsymbol{x}, \boldsymbol{y})$ satisfies weak reversibility; a sufficient example is that

$$R(\boldsymbol{x}, \boldsymbol{y}) := \begin{cases} \frac{1}{|\mathcal{B}(\boldsymbol{x})|}, & \text{if } \boldsymbol{y} \in \mathcal{B}(\boldsymbol{x}), \\ 0, & \text{otherwise}, \end{cases}$$

   with symmetric neighborhood, i.e., $\boldsymbol{y} \in \mathcal{B}(\boldsymbol{x}) \iff \boldsymbol{x} \in \mathcal{B}(\boldsymbol{y})$.

2. $\{t_k\}_{k\geq 1}$ takes the form

$$t_k = \frac{c}{\ln(k+1)},$$

   where c is sufficiently large.†

† $c \geq d^*$, where d^* is the maximum depth (Hajek (1988, p313)) of the local but not global optimal solutions.

- Simulated annealing algorithm for black-box DOvS (Gelfand and Mitter 1989):
  - Initialization: Arbitrarily choose $\boldsymbol{X}_0 \in \mathcal{X}$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Sample a candidate solution $\boldsymbol{Y}_{k+1} \in \mathcal{B}(\boldsymbol{X}_k)$ according to distribution $R(\boldsymbol{X}_k, \cdot)$, i.e.,

$$\mathbb{P}(\boldsymbol{Y}_{k+1} = \boldsymbol{y} | \boldsymbol{X}_k = \boldsymbol{x}) = R(\boldsymbol{x}, \boldsymbol{y}).$$

    - Evaluation: Let $\widehat{g}(\boldsymbol{Y}_{k+1}) := \frac{1}{n_{k+1}} \sum_{i=1}^{n_{k+1}} G(\boldsymbol{Y}_{k+1}, \xi_i)$, $\widehat{g}(\boldsymbol{X}_k) := \frac{1}{n_{k+1}} \sum_{i=1}^{n_{k+1}} G(\boldsymbol{X}_k, \xi_i)$.[†]
    - Updating: Let

$$\boldsymbol{X}_{k+1} := \begin{cases} \boldsymbol{Y}_{k+1}, & \text{with probability } \exp\left\{\frac{-[\widehat{g}(\boldsymbol{Y}_{k+1}) - \widehat{g}(\boldsymbol{X}_k)]^+}{t_{k+1}}\right\}, \\ \boldsymbol{X}_k, & \text{otherwise;} \end{cases}$$

    set $k \leftarrow k + 1$.

[†] ξ_i in $G(\boldsymbol{Y}_{k+1}, \xi_i)$ and ξ_i in $G(\boldsymbol{X}_k, \xi_i)$ denote different randomness; written in this way just for notation simplity.

- Gelfand and Mitter (1989) show that if

$$\widehat{g}(\boldsymbol{Y}_{k+1})\big|\boldsymbol{Y}_{k+1}=\boldsymbol{y}\sim\mathcal{N}(g(\boldsymbol{y}),\sigma_{k+1}^2),$$

  such that $\sigma_k=o(t_k)$, then the simulated annealing algorithm used for DOvS has the same global convergence as its counterpart used for deterministic optimization.

- Gelfand and Mitter (1989) show that if

$$\widehat{g}(\boldsymbol{Y}_{k+1})\big|\boldsymbol{Y}_{k+1} = \boldsymbol{y} \sim \mathcal{N}(g(\boldsymbol{y}), \sigma_{k+1}^2),$$

  such that $\sigma_k = o(t_k)$, then the simulated annealing algorithm used for DOvS has the same global convergence as its counterpart used for deterministic optimization.

- A sufficient condition is that:
  - $G(\boldsymbol{x}, \xi) \sim \mathcal{N}(g(\boldsymbol{x}), \sigma^2(\boldsymbol{x}))$ with $\sigma^2(\boldsymbol{x}) \le \sigma^2 < \infty$ for all $\boldsymbol{x} \in \mathcal{X}$.
  - $\{n_k\}_{k\ge1}$ satisfies $\lim_{k\to\infty} \frac{1}{t_k\sqrt{n_k}} = 0$, i.e., $n_k := t_k^{-\alpha}$ with $\alpha > 2$.

- Gelfand and Mitter (1989) show that if

$$\widehat{g}(\boldsymbol{Y}_{k+1})\big|\boldsymbol{Y}_{k+1}=\boldsymbol{y}\sim\mathcal{N}(g(\boldsymbol{y}),\sigma_{k+1}^2),$$

such that $\sigma_k = o(t_k)$, then the simulated annealing algorithm used for DOvS has the same global convergence as its counterpart used for deterministic optimization.

- A sufficient condition is that:
  - $G(\boldsymbol{x},\xi)\sim\mathcal{N}(g(\boldsymbol{x}),\sigma^2(\boldsymbol{x}))$ with $\sigma^2(\boldsymbol{x})\le\sigma^2<\infty$ for all $\boldsymbol{x}\in\mathcal{X}$.
  - $\{n_k\}_{k\ge1}$ satisfies $\lim_{k\to\infty}\frac{1}{t_k\sqrt{n_k}}=0$, i.e., $n_k:=t_k^{-\alpha}$ with $\alpha>2$.

- Alrefaei and Andradóttir (1999) propose a modified simulated annealing algorithm for DOvS, which is also globally convergent:
  - temperature t_k is constant;
  - the current best solution is chosed in a different way.

- Convergent Optimization via Most-Promising-Area Stochastic Search (COMPASS) is a **locally convergent** algorithm for black-box algorithm proposed by Hong and Nelson (2006).

- Convergent Optimization via Most-Promising-Area Stochastic Search (COMPASS) is a **locally convergent** algorithm for black-box algorithm proposed by Hong and Nelson (2006).
- It can be used when the discrete feasible set is finite (i.e., fully constrained) or infinite (i.e., partially constrained or unconstrained).

- COMPASS for DOvS Hong and Nelson (2006):
  - Initialization:
  - At Iteration k:
    - Sampling:
    - Evaluation:
    - Updating:

- COMPASS for DOvS Hong and Nelson (2006):
  - Initialization: Arbitrarily choose $\boldsymbol{x}_0 \in \mathcal{X}$; set $\boldsymbol{x}_0^* = \boldsymbol{x}_0$ and $\mathcal{V}_0 = \{\boldsymbol{x}_0\}$; take observations according to a simulation allocation rule (SAR) from $\boldsymbol{x}_0$; let $\mathcal{P}_0 = \mathcal{X}$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling:
    - Evaluation:
    - Updating:

- COMPASS for DOvS Hong and Nelson (2006):
  - Initialization: Arbitrarily choose $\boldsymbol{x}_0 \in \mathcal{X}$; set $\boldsymbol{x}_0^* = \boldsymbol{x}_0$ and $\mathcal{V}_0 = \{\boldsymbol{x}_0\}$; take observations according to a simulation allocation rule (SAR) from $\boldsymbol{x}_0$; let $\mathcal{P}_0 = \mathcal{X}$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Sample m solutions uniformly and independently from $\mathcal{P}_k$, denoted as $\{\boldsymbol{x}_{k1}, \ldots, \boldsymbol{x}_{km}\}$; let $\mathcal{V}_{k+1} := \mathcal{V}_k \cup \{\boldsymbol{x}_{k1}, \ldots, \boldsymbol{x}_{km}\}$ be the estimation set.
    - Evaluation:
    - Updating:

- COMPASS for DOvS Hong and Nelson (2006):
  - Initialization: Arbitrarily choose $\boldsymbol{x}_0 \in \mathcal{X}$; set $\boldsymbol{x}_0^* = \boldsymbol{x}_0$ and $\mathcal{V}_0 = \{\boldsymbol{x}_0\}$; take observations according to a simulation allocation rule (SAR) from $\boldsymbol{x}_0$; let $\mathcal{P}_0 = \mathcal{X}$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Sample m solutions uniformly and independently from $\mathcal{P}_k$, denoted as $\{\boldsymbol{x}_{k1}, \ldots, \boldsymbol{x}_{km}\}$; let $\mathcal{V}_{k+1} := \mathcal{V}_k \cup \{\boldsymbol{x}_{k1}, \ldots, \boldsymbol{x}_{km}\}$ be the estimation set.
    - Evaluation: For each $\boldsymbol{x} \in \mathcal{V}_{k+1}$, take *additional* observations according to the SAR.
    - Updating:

- COMPASS for DOvS Hong and Nelson (2006):
  - Initialization: Arbitrarily choose $\boldsymbol{x}_0 \in \mathcal{X}$; set $\boldsymbol{x}_0^* = \boldsymbol{x}_0$ and $\mathcal{V}_0 = \{\boldsymbol{x}_0\}$; take observations according to a simulation allocation rule (SAR) from $\boldsymbol{x}_0$; let $\mathcal{P}_0 = \mathcal{X}$; set iteration index $k = 0$.
  - At Iteration k:
    - Sampling: Sample m solutions uniformly and independently from $\mathcal{P}_k$, denoted as $\{\boldsymbol{x}_{k1}, \ldots, \boldsymbol{x}_{km}\}$; let $\mathcal{V}_{k+1} := \mathcal{V}_k \cup \{\boldsymbol{x}_{k1}, \ldots, \boldsymbol{x}_{km}\}$ be the estimation set.
    - Evaluation: For each $\boldsymbol{x} \in \mathcal{V}_{k+1}$, take *additional* observations according to the SAR.
    - Updating: Update $\mathcal{P}_{k+1}$; choose the solution in $\mathcal{V}_{k+1}$ with smallest estimated funtion value as $\boldsymbol{x}_{k+1}^*$; set $k \leftarrow k + 1$.

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

- The way to construct $\mathcal{P}_k$ – the most promising area:

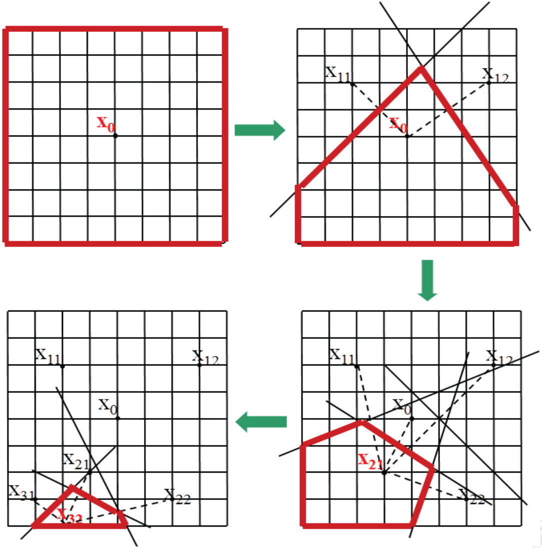


Image from Jeff Hong

Usage in Softwares

- In many commercial simulation softwares, like Arena, AnyLogic, Simio and FlexSim, OptQuest is integrated for simulation optimization.

Usage in Softwares

- In many commercial simulation softwares, like Arena, AnyLogic, Simio and FlexSim, OptQuest is integrated for simulation optimization.
- OptQuest is based on a combination of methods, including linear/integer programming, heuristics and metaheuristics.
  - It is robust when used to solve practical OvS problems;
  - but it has no provable convergence for OvS problems.

Usage in Softwares

- In many commercial simulation softwares, like Arena, AnyLogic, Simio and FlexSim, OptQuest is integrated for simulation optimization.
- OptQuest is based on a combination of methods, including linear/integer programming, heuristics and metaheuristics.
  - It is robust when used to solve practical OvS problems;
  - but it has no provable convergence for OvS problems.
- None of those OvS algirhtms have been integrated into the commercial simulation softwares yet.

Usage in Softwares

- In many commercial simulation softwares, like Arena, AnyLogic, Simio and FlexSim, OptQuest is integrated for simulation optimization.
- OptQuest is based on a combination of methods, including linear/integer programming, heuristics and metaheuristics.
  - It is robust when used to solve practical OvS problems;
  - but it has no provable convergence for OvS problems.
- None of those OvS algirhtms have been integrated into the commercial simulation softwares yet.
- So, for reaseachers in the field of OvS, there is still a long way to go...