Modeling Extreme Events from Computer Simulations
Part IV

Emmanuel Vazquez

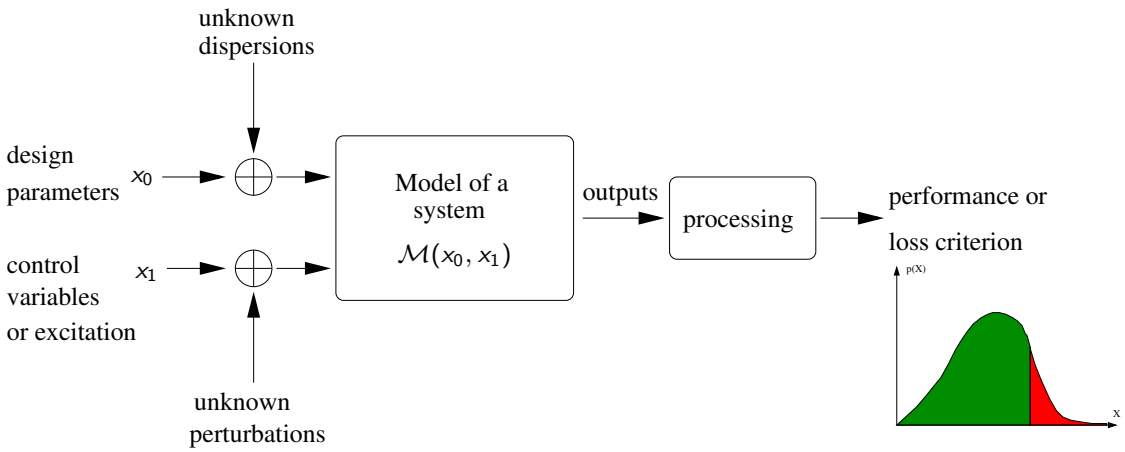
SUPELEC, Gif-sur-Yvette, France

Summer School CEA-EDF-INRIA, 2011

Outline

1. Uses of computer models in engineering: a reminder

Computer models in engineering



Model implemented under the form of a computer program (e.g., a finite element model). A single run of the program may be time- and resource-consuming.

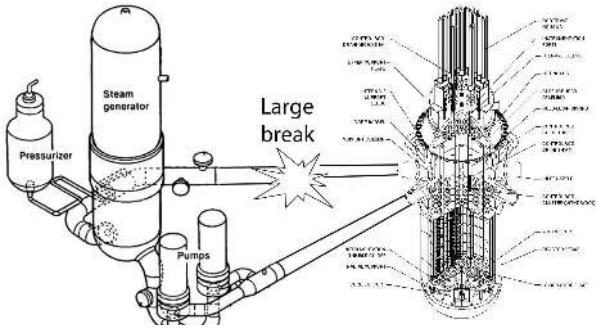
Computer models in engineering

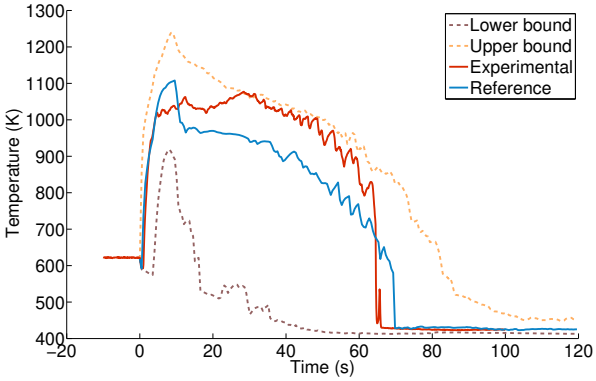
- $\mathbb{X} \subseteq \mathbb{R}^d$: input domain of the system
- $f : \mathbb{X} \to \mathbb{R}$: a performance or cost function (function of the outputs of the system)
- Main classes of problems
  1. Optimization of the performances of a system, cost minimization...
  $$x^\star = \underset{x \in \mathbb{X}}{\operatorname{argmax}}\, f(x)$$
  2. In presence of uncertain factors: minimize a probability of failure, i.e.,
  $$\begin{aligned} \mathbb{X} &= \mathbb{X}_0 \times \mathbb{X}_1 \\ x_0^\star &= \underset{x_0 \in \mathbb{X}_0}{\operatorname{argmin}}\, \alpha(x_0) \\ \alpha(x_0) &:= \mathrm{P}_{\mathbb{X}_1}\{x_1 \in \mathbb{X}_1 : f(x_0, x_1) > u\} \end{aligned}$$
  where $\mathrm{P}_{\mathbb{X}_1}$ is some probability distribution on $(\mathbb{X}_1, \mathcal{B}(\mathbb{X}_1))$
  3. Performance assessment: estimation of a quantile
  $$q_\alpha(x_0) = \inf\{u \in \mathbb{R};\ \mathrm{P}_{\mathbb{X}_1}\{x_1 \in \mathbb{X}_1 : f(x_0, x_1) \leq u\} \geq \alpha\}$$

(This is a simplified view. Most real problems have several performance functions, and mix different objectives.)

Computer models in engineering

- Computer simulations to assess the probability of undesirable events in a nuclear reactor



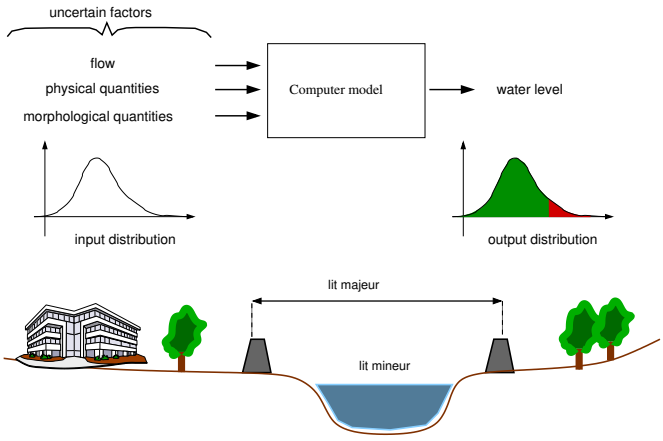


(Courtesy of CEA)

- A serious accident: loss of coolant in a pressurized water nuclear reactor
- Under these conditions, temperature of fuel rods can be described by ~ 50 dimensioning factors, which are not known accurately
- Peak temperature can be estimated using complex and time-consuming simulations
- $f : \mathbb{X} \to \mathbb{R}$ peak temp. as a function of the factors
- Objective: estimate a probability of exceeding a critical value
$$\alpha = \mathrm{P}_{\mathbb{X}}\{f \geq u\}$$
or a quantile
$$q_\alpha = \inf\{u \in \mathbb{R};\ \mathrm{P}_{\mathbb{X}}\{f \leq u\} \geq \alpha\}$$
or a worst-case
$$M = \sup_{x \in \mathbb{X}} f(x)$$

Computer models in engineering

- Computer simulations to assess the probability of river flooding



(Courtesy of EDF)

- Risk of water flooding in an inhabitable or industrial area assessed by modeling the water-surface profile of a river as a function of factors, such as the river discharge and the features of the riverbed
- Because a single evaluation of such a model for known discharge and riverbed features is potentially time-consuming, risk of flooding must be assessed with a small budget of simulations
- $f : \mathbb{X} \to \mathbb{R}$ the water level as a function of the factors
- The objective is to estimate a quantile for the water level

$$q_{1-\alpha} = \inf\{u \in \mathbb{R};\ \mathrm{P}_{\mathbb{X}}\{f \leq u\} \geq 1-\alpha\}$$

for a given α that is close to zero.

2. How to construct a good estimation procedure?

Estimation from computer experiments

- Let $f : \mathbb{X} \to \mathbb{R}$ be a continuous function.
  (f corresponds to a computer program whose output is not a closed-form expression of the inputs.)
- Our objective: to obtain an approximation of

$$m(f) = \min_{x \in \mathbb{X}} f(x) = f(x^\star)$$

or

$$\alpha^u(f) = \mathrm{P}_{\mathbb{X}}\{f > u\} = \int_{\mathbb{X}} \mathbb{1}_{f>u} \mathrm{dP}_{\mathbb{X}}$$

or

$$q_{1-\alpha}(f) = \inf\{u \in \mathbb{R};\ \mathrm{P}_{\mathbb{X}}\{f \leq u\} \geq 1 - \alpha\}$$

- The approximation of $m(f)$, $\alpha^u(f)$, etc. has to be built from a set of computer experiments (where an experiment simply consists in choosing an $x \in \mathbb{X}$, and computing the value of f at x).
- The result of a pointwise evaluation of f carries information about f and quantities depending on f (in particular, $m(f)$, $\alpha^u(f)$, or $q_{1-\alpha}(f)$)
- Expensive computer experiments: the number of evaluations is limited $\to$ $m(f)$, $\alpha^u(f)$, etc. must be estimated using a fixed number, say N, of evaluations of f.

The case of optimization

- Formally, an optimization algorithm corresponds to a pair $(\underline{X}_N, \widehat{m}_N)$,

$$\begin{aligned} \underline{X}_N &: f \mapsto \underline{X}_N(f) = (X_1(f), X_2(f), \ldots, X_N(f)) \in \mathbb{X}^N, \\ \widehat{m}_N &: f \mapsto \widehat{m}_N(f) \in \mathbb{R}_+, \end{aligned}$$

  with the following properties:

  a) There exists $x_1 \in \mathbb{X}$ such that $X_1(f) = x_1$

  b) Let $Z_n(f) = f(X_n(f))$, $1 \leq n \leq N$ For all $1 \leq n < N$, $X_{n+1}(f)$ depends measurably[1] on $\mathcal{I}_n(f)$, where $\mathcal{I}_n = ((X_1, Z_1), \ldots, (X_n, Z_n))$

  c) $\widehat{m}_N(f)$ depends measurably on $\mathcal{I}_N(f)$

- $\underline{X}_N$ is called a strategy, or policy, or design of experiments
- $\widehat{m}_N(f)$ is an estimator of $m(f)$

[1] i.e., there is a measurable map $\varphi_n : (\mathbb{X} \times \mathbb{R})^n \to \mathbb{X}$ such that $X_n = \varphi_n \circ \mathcal{I}_n$

The case of optimization

❑ Formally, an optimization algorithm corresponds to a pair $(\underline{X}_N, \widehat{m}_N)$,

$$\begin{array}{rcl} \underline{X}_N & : & f \mapsto \underline{X}_N(f) = (X_1(f), X_2(f), \ldots, X_N(f)) \in \mathbb{X}^N \,, \\ \widehat{m}_N & : & f \mapsto \widehat{m}_N(f) \in \mathbb{R}_+ \,, \end{array}$$

with the following properties:

a) There exists $x_1 \in \mathbb{X}$ such that $X_1(f) = x_1$

b) Let $Z_n(f) = f(X_n(f))$, $1 \leq n \leq N$ For all $1 \leq n < N$, $X_{n+1}(f)$ depends measurably[1] on $\mathcal{I}_n(f)$, where $\mathcal{I}_n = ((X_1, Z_1), \ldots, (X_n, Z_n))$

c) $\widehat{m}_N(f)$ depends measurably on $\mathcal{I}_N(f)$

- $\underline{X}_N$ is called a strategy, or policy, or design of experiments
- $\widehat{m}_N(f)$ is an estimator of $m(f)$

[1] i.e., there is a measurable map $\varphi_n : (\mathbb{X} \times \mathbb{R})^n \to \mathbb{X}$ such that $X_n = \varphi_n \circ \mathcal{I}_n$

How to construct a good estimation procedure?

The case of optimization

❑ Formally, an optimization algorithm corresponds to a pair $(\underline{X}_N, \widehat{m}_N)$,

$$\begin{array}{rcl} \underline{X}_N & : & f \mapsto \underline{X}_N(f) = (X_1(f), X_2(f), \ldots, X_N(f)) \in \mathbb{X}^N \,, \\ \widehat{m}_N & : & f \mapsto \widehat{m}_N(f) \in \mathbb{R}_+ \,, \end{array}$$

with the following properties:

a) There exists $x_1 \in \mathbb{X}$ such that $X_1(f) = x_1$

b) Let $Z_n(f) = f(X_n(f))$, $1 \leq n \leq N$ For all $1 \leq n < N$, $X_{n+1}(f)$ depends measurably[1] on $\mathcal{I}_n(f)$, where $\mathcal{I}_n = ((X_1, Z_1), \ldots, (X_n, Z_n))$

c) $\widehat{m}_N(f)$ depends measurably on $\mathcal{I}_N(f)$

- $\underline{X}_N$ is called a strategy, or policy, or design of experiments
- $\widehat{m}_N(f)$ is an estimator of $m(f)$

[1] i.e., there is a measurable map $\varphi_n : (\mathbb{X} \times \mathbb{R})^n \to \mathbb{X}$ such that $X_n = \varphi_n \circ \mathcal{I}_n$

E. Vazquez Extreme events modeling IV Summer School CEA-EDF-INRIA, 2011 10 / 141

The case of optimization

- The algorithm $(\underline{X}_N, \widehat{m}_N)$ describes a sequence of decisions, made from an increasing amount of information:
  - $X_1(f) = x_1$ is chosen prior to any evaluation
  - for each $n = 1, \ldots, N-1$, the algorithm uses information $\mathcal{I}_n(f)$ to choose the next evaluation point $X_{n+1}(f)$
  - the estimator $\widehat{m}_N(f)$ of $m(f)$ is the terminal decision
- In the framework of optimization, we generally consider $\widehat{m}_N = \min_{1 \leq n \leq N} Z_n$.

Sequential estimation of a probability of failure

- Likewise, an algorithm to estimate a probability of failure corresponds to a pair $(\underline{X}_N, \widehat{\alpha}_N)$,

$$\begin{aligned} \underline{X}_N &: f \mapsto \underline{X}_N(f) = (X_1(f), X_2(f), \ldots, X_N(f)) \in \mathbb{X}^N, \\ \widehat{\alpha}_N &: f \mapsto \widehat{\alpha}_N(f) \in \mathbb{R}_+, \end{aligned}$$

  with the following properties:

  a) There exists $x_1 \in \mathbb{X}$ such that $X_1(f) = x_1$

  b) Let $Z_n(f) = f(X_n(f))$, $1 \leq n \leq N$. For all $1 \leq n < N$, $X_{n+1}(f)$ depends measurably on $\mathcal{I}_n(f)$, where $\mathcal{I}_n = ((X_1, Z_1), \ldots, (X_n, Z_n))$

  c) $\widehat{\alpha}_N(f)$ depends measurably on $\mathcal{I}_N(f)$

  - Again, $\underline{X}_N$ is called a strategy
  - $\widehat{\alpha}_N(f)$ is an estimator of $\alpha(f)$

- To simplify our presentation, we deal first with the problem of optimization
- The case of the estimation of a probability of failure, and that of the estimation of a quantile will be detailed later

3. Optimization of an expensive-to-evaluate function

How to predict the worst case from time- and resource-consuming computer experiments?

- In the context of rare events estimation and risk analysis, it is often desirable to assess the worst-case performance of a system, that is, to determine

$$M = \sup_{x \in \mathbb{X}} f(x)$$

or

$$m = \inf_{x \in \mathbb{X}} f(x)$$

  - ➟ f may be non-convex
  - ➟ this is a global optimization problem

- How to define a good strategy $\underline{X}_N$ for the optimization problem?
- In a context of risk analysis, we want a strategy that will provide a robust estimation of the global optimum

3.1 Why local optimization methods may not be satisfactory for risk analysis

An illustrative example

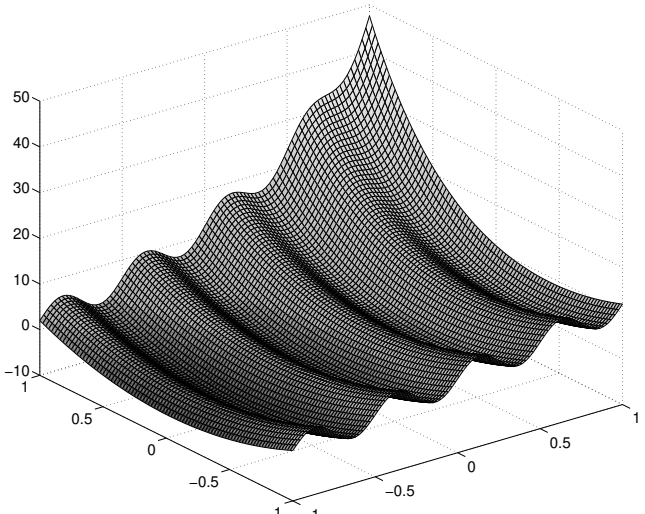
- Consider

$$\begin{array}{rccl} f: & \mathbb{R}^2 & \to & \mathbb{R} \\ & x & \mapsto & f(x) = \exp\left(1.8\left(x_{[1]} + x_{[2]}\right)\right) + 5x_{[1]} + 6{x_{[2]}}^2 + 3\sin\left(4\pi x_{[1]}\right) \end{array}$$

- Objective: find an approximation of

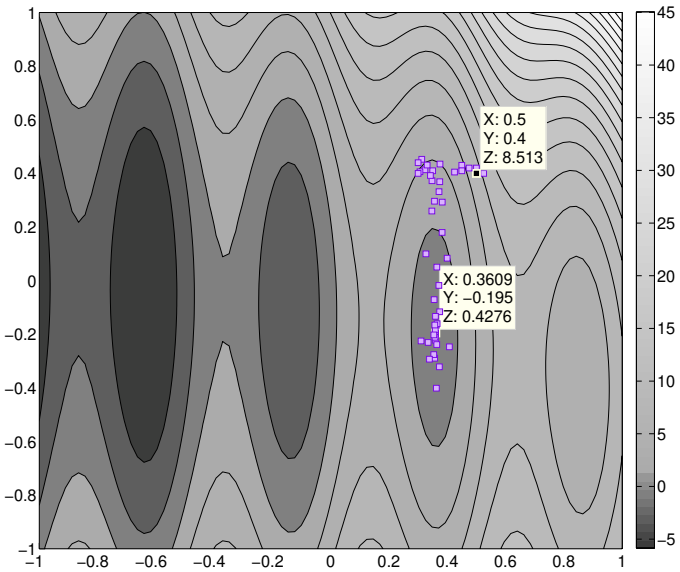
$$x^\star = \underset{x\in[-1,1]^2}{\operatorname{argmin}}\ f(x)\,.$$

with a budget of $N = 60$ experiments



An illustrative example (continued)

Evaluations points using a Nelder-Mead algorithm (fminsearch function of Matlab)



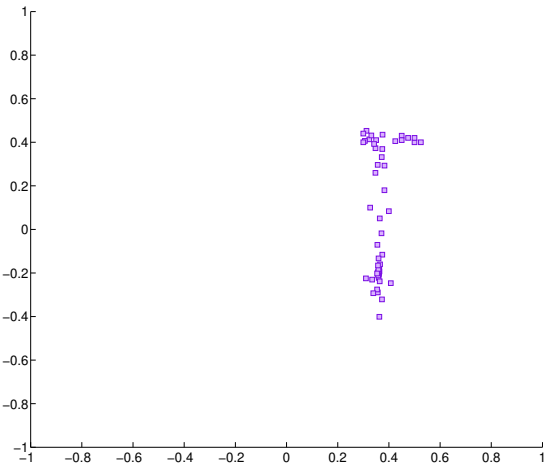
→ the algorithm converges to a local minimum (≈ 0.427)

This comes as no surprise (local search algorithm). But above all...

- after having spent the budget of (possibly expensive) evaluations, the behavior of the function is only known in a small region of the search domain
- the global behavior of the function is unknown
- potentially interesting regions have not been explored

This comes as no surprise (local search algorithm). But above all...

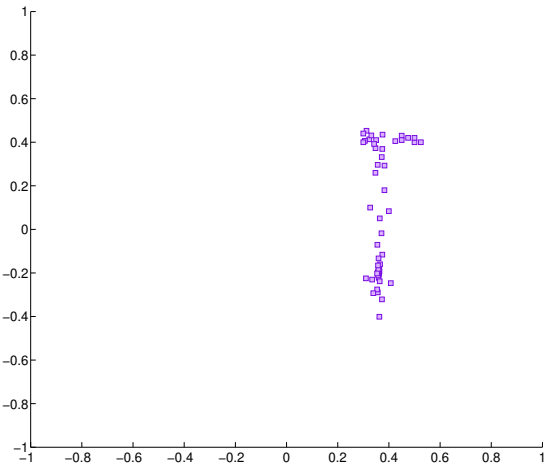
- after having spent the budget of (possibly expensive) evaluations, the behavior of the function is only known in a small region of the search domain



- the global behavior of the function is unknown
- potentially interesting regions have not been explored

This comes as no surprise (local search algorithm). But above all...

- after having spent the budget of (possibly expensive) evaluations, the behavior of the function is only known in a small region of the search domain

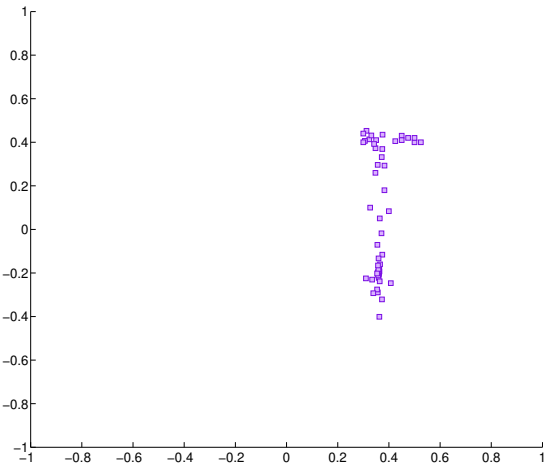


- the global behavior of the function is unknown
- potentially interesting regions have not been explored

Optimization of an expensive-to-evaluate function The problem with local optimization methods

This comes as no surprise (local search algorithm). But above all...

- after having spent the budget of (possibly expensive) evaluations, the behavior of the function is only known in a small region of the search domain



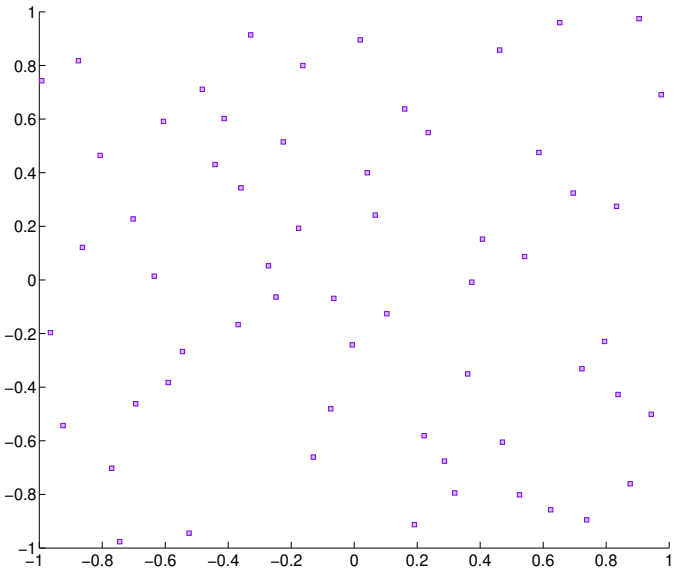
- the global behavior of the function is unknown
- potentially interesting regions have not been explored

In a context of expensive-to-evaluate functions and a small budget of evaluations, a "safer" strategy would consists in sampling f uniformly on the search domain

$\rightarrow$ minimum of evaluation results is ≈ -5.823 (global minimum is ≈ -5.845)

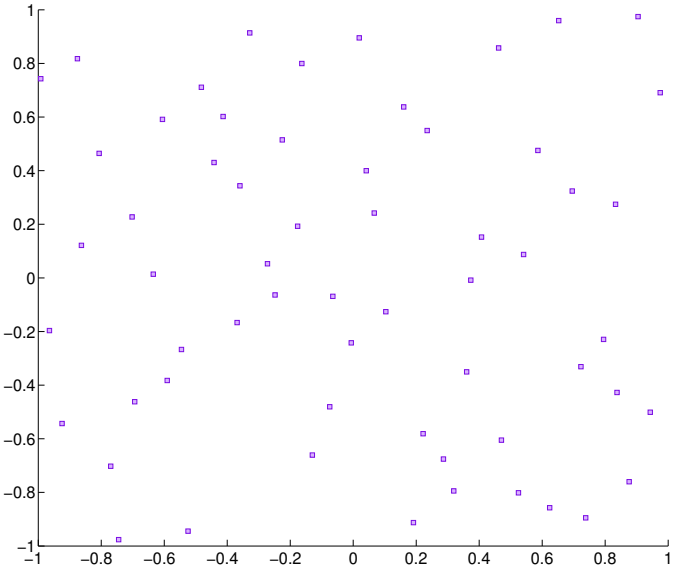
Optimization of an expensive-to-evaluate function The problem with local optimization methods

In a context of expensive-to-evaluate functions and a small budget of evaluations, a "safer" strategy would consists in sampling f uniformly on the search domain



$\rightarrow$ minimum of evaluation results is ≈ -5.823 (global minimum is ≈ -5.845)

In a context of expensive-to-evaluate functions and a small budget of evaluations, a "safer" strategy would consists in sampling f uniformly on the search domain



$\rightarrow$ minimum of evaluation results is ≈ -5.823 (global minimum is ≈ -5.845)

Summing up

- In a context of risk analysis and a limited budget of evaluations, it seems safer to balance between local search and exploration of the search domain
- An exploration/exploitation trade-off has to be achieved
- How to define a robust strategy?

Summing up

- In a context of risk analysis and a limited budget of evaluations, it seems safer to balance between local search and exploration of the search domain
- An exploration/exploitation trade-off has to be achieved
- How to define a robust strategy?

Optimization of an expensive-to-evaluate function The problem with local optimization methods

Summing up

- In a context of risk analysis and a limited budget of evaluations, it seems safer to balance between local search and exploration of the search domain
- An exploration/exploitation trade-off has to be achieved
- How to define a robust strategy?

The worst-case approach

- Let $\mathcal{A}_N$ be the class of all strategies $\underline{X}_N$ that query sequentially N evaluations of f.
- Define the error of approximation of a strategy $\underline{X}_N \in \mathcal{A}_N$ on f as

$$\varepsilon(\underline{X}_N, f) = \widehat{m}_N(f) - m(f)$$

- Assume that f belongs to a class of functions $\mathcal{F} \rightarrow$ prior information

➜ A first idea to define a notion of a good strategy is to consider robustness with respect to a worst case

- Define the minimax risk

$$r_{\text{minimax}}(\mathcal{F}) = \inf_{\underline{X}_N \in \mathcal{A}_N} \sup_{f \in \mathcal{F}} \varepsilon(\underline{X}_N, f)$$

- A strategy ${\underline{X}_N}^\star$ that attains $r_{\text{minimax}}(\mathcal{F})$ is called an optimal minimax strategy
- ${\underline{X}_N}^\star$ has the best worst-case performance on $\mathcal{F}$

Example of a minimax strategy: case of Lipschitz functions

Definition

A function $f : \mathbb{X} \rightarrow \mathbb{R}$ is called Lipschitz continuous if there exists a real constant $K \geq 0$ such that, for all x_1 and x_2 in $\mathbb{X}$,

$$|f(x_1) - f(x_2)| \leq K\|x_1 - x_2\|.$$

Any such K is referred to as a Lipschitz constant for the function f.

Example of a minimax strategy: case of Lipschitz functions

- Let $\mathcal{F}$ be the class of all Lipschitz continuous functions $\mathbb{X} \to \mathbb{R}$, with Lipschitz constant K
- Assume that $f \in \mathcal{F}$
- For any strategy $\underline{X}_N$, define the fill distance as

$$h_N = \sup_{x \in \mathbb{X}} \min_{i=1,\ldots,N} |x - X_i|$$

- For any $\underline{X}_N \in \mathcal{A}_N$ and any $f \in \mathcal{F}$,

$$\varepsilon(\underline{X}_N, f) = f(X_1) \wedge \cdots \wedge f(X_N) - f(x^\star) \leq f(X_{i^\star}) - f(x^\star) \leq K h_N \,,$$

  where $X_{i^\star}$ is the nearest point to $x^\star$
- Thus, for any $\underline{X}_N \in \mathcal{A}_N$, $\sup_{f \in \mathcal{F}} \varepsilon(\underline{X}_N, f) \leq K h_n$
- For any $\underline{X}_N$, there exists a function $f \in \mathcal{F}$ such that

$$\varepsilon(\underline{X}_N, f) = K h_N$$

  Thus,

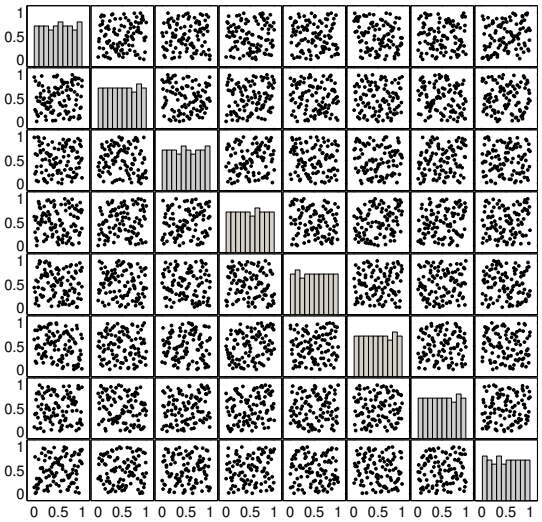
$$\sup_{f \in \mathcal{F}} \varepsilon(\underline{X}_N, f) = K h_N$$

Example of a minimax strategy: case of Lipschitz functions

- Consequence: a minimax strategy minimizes h_N
  $\to$ sample points have to be uniformly distributed over the search domain
- In dimension one:
  - for any $\underline{X}_N$, $h_N \geq \frac{|X|}{(N+1)}$
  - the optimal strategy is the uniform sampling: $r_{\text{minimax}}(\mathcal{F}) = K \frac{|X|}{(N+1)}$
- How to deal dimension $d > 1$?
  - using a uniform grid is not optimal (not mentioning the fact that the budget of evaluations must be at least $N = 2^d$)
  - sampling randomly with a uniform distribution over $\mathbb{X}$ provides no guarantee that h_n will be small
  - optimizing the design of experiments to yield a small h_n is interesting but may numerically expensive
  - Minimax Latin Hypercube Sampling is an easy procedure that will generally provide good suboptimal designs

Latin Hypercube Sampling [McKay, Conover and Beckman (1979)]

- Assume $\mathbb{X} = [0,1]^d$. To obtain an LHS of size N:
  - Along each dimension of $\mathbb{X}$, split the interval $[0,1]$ into N intervals of equal length $\rightarrow N \times d$ cells
  - Choose n cells in such a way that there is exactly one cell per interval of each dimension
  - In each of the n selected cells, sample one point
- This sampling scheme does not require more samples for more dimensions
- A simple LHS provides not guarantee that h_N will be small
- Key idea: generate several LHS (cheap procedure) and select the design that achieves the smallest h_N
- Maximin LHS design is implemented in the Statistical Toolbox of Matlab



Example of a maximin Latin hypercube sampling of size $n = 100$ in dimension $d = 8$

Latin Hypercube Sampling for optimization

Pros

- Straightforward implementation
- Global search and near optimal minimax strategy
- The behavior of the function is well captured over the search domain

Cons

- The minimax approach is a pessimistic approach
- No local search

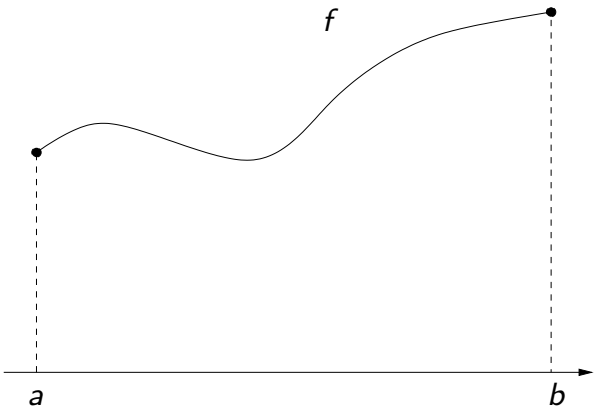
➜ In practice, we would like to achieve a balance between exploration of the search domain and local search in promising regions (good performance on worst cases and good convergence rate)

3.4 Sequential exploration/exploitation strategies for Lipschitz continuous functions

A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

- Assume:
  - $\mathbb{X} = [a, b]$, with $-\infty < a < b < +\infty$
  - f is Lipschitz continuous, with Lipschitz constant K
- For any two points $a \leq x_i < x_j \leq b$, and $\forall x \in [x_i, x_j]$, the following lower-bounds hold

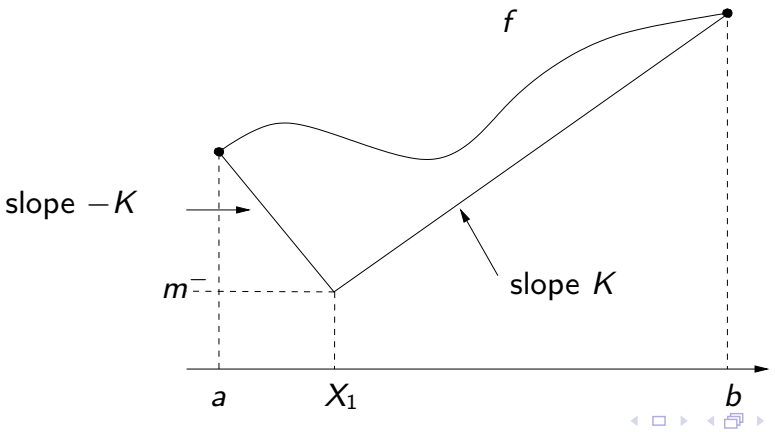
$$f(x) \geq f(x_i) - K(x - x_i)$$
$$f(x) \geq f(x_j) + K(x - x_j).$$



A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

- Assume:
  - $\mathbb{X} = [a, b]$, with $-\infty < a < b < +\infty$
  - f is Lipschitz continuous, with Lipschitz constant K
- For any two points $a \leq x_i < x_j \leq b$, and $\forall x \in [x_i, x_j]$, the following lower-bounds hold

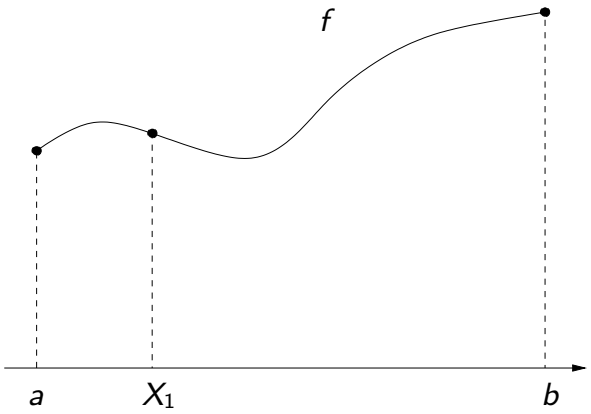
$$f(x) \geq f(x_i) - K(x - x_i)$$
$$f(x) \geq f(x_j) + K(x - x_j).$$



A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

- Assume:
  - $\mathbb{X} = [a, b]$, with $-\infty < a < b < +\infty$
  - f is Lipschitz continuous, with Lipschitz constant K
- For any two points $a \leq x_i < x_j \leq b$, and $\forall x \in [x_i, x_j]$, the following lower-bounds hold

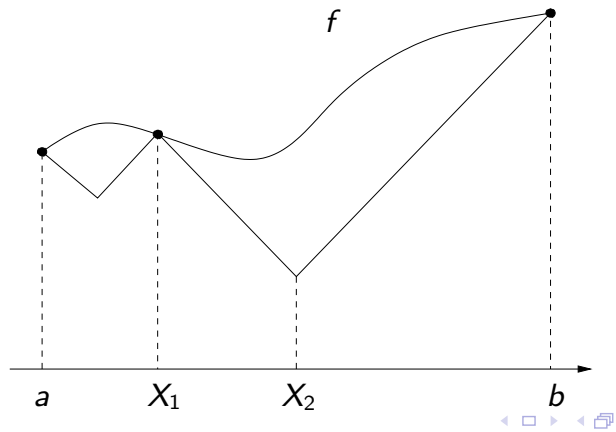
$$f(x) \geq f(x_i) - K(x - x_i)$$
$$f(x) \geq f(x_j) + K(x - x_j).$$



A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

- Assume:
  - $\mathbb{X} = [a, b]$, with $-\infty < a < b < +\infty$
  - f is Lipschitz continuous, with Lipschitz constant K
- For any two points $a \leq x_i < x_j \leq b$, and $\forall x \in [x_i, x_j]$, the following lower-bounds hold

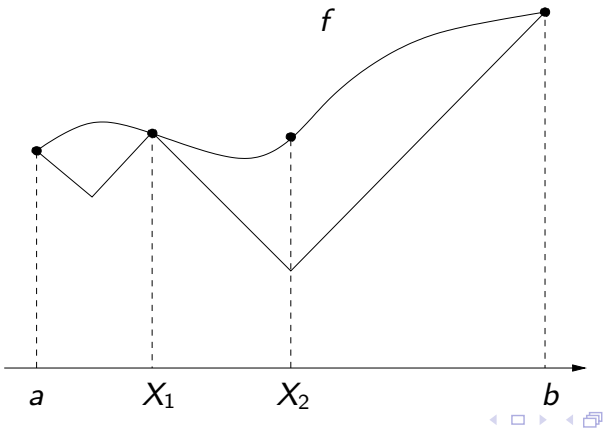
$$f(x) \geq f(x_i) - K(x - x_i)$$
$$f(x) \geq f(x_j) + K(x - x_j).$$



A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

- Assume:
  - $\mathbb{X} = [a, b]$, with $-\infty < a < b < +\infty$
  - f is Lipschitz continuous, with Lipschitz constant K
- For any two points $a \leq x_i < x_j \leq b$, and $\forall x \in [x_i, x_j]$, the following lower-bounds hold

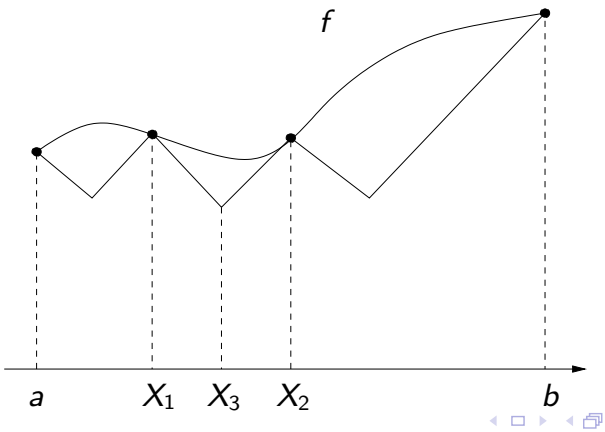
$$f(x) \geq f(x_i) - K(x - x_i)$$
$$f(x) \geq f(x_j) + K(x - x_j).$$



A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

- Assume:
  - $\mathbb{X} = [a, b]$, with $-\infty < a < b < +\infty$
  - f is Lipschitz continuous, with Lipschitz constant K
- For any two points $a \leq x_i < x_j \leq b$, and $\forall x \in [x_i, x_j]$, the following lower-bounds hold

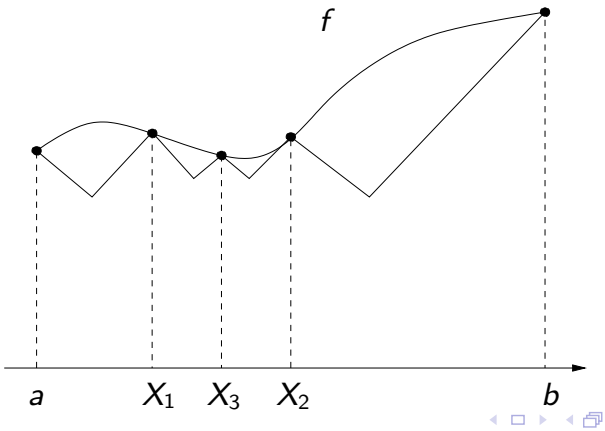
$$f(x) \geq f(x_i) - K(x - x_i)$$
$$f(x) \geq f(x_j) + K(x - x_j).$$



A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

- Assume:
  - $\mathbb{X} = [a, b]$, with $-\infty < a < b < +\infty$
  - f is Lipschitz continuous, with Lipschitz constant K
- For any two points $a \leq x_i < x_j \leq b$, and $\forall x \in [x_i, x_j]$, the following lower-bounds hold

$$f(x) \geq f(x_i) - K(x - x_i)$$
$$f(x) \geq f(x_j) + K(x - x_j).$$



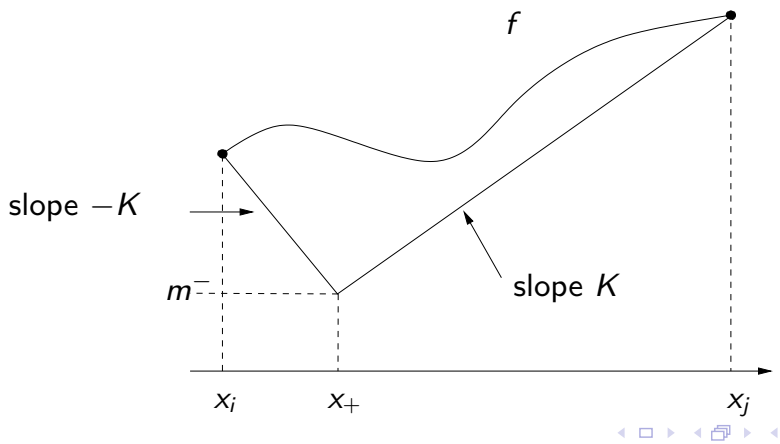
A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

- Exploration vs Exploitation

$$x_+ = \frac{f(x_i) - f(x_j)}{2K} + \frac{x_i + x_j}{2}$$

$$m^- = \frac{f(x_i) + f(x_j)}{2} - K\frac{x_j - x_i}{2}$$

- K can be seen as a parameter to tune the tradeoff local search vs exploration



A sequential method seeking the global maximum of a Lipschitz continuous function [Shubert (1972)]

Pros

- Straightforward implementation
- Global search and local search
- Gives bound on error

Cons

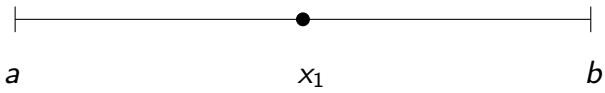
- A low Lipschitz constant has to be known (high $K \rightarrow$ global search)
- Speed of convergence (global vs. local)
- Computational complexity in higher dimensions (Shubert's algorithm is initialized by evaluating the function at the vertices of a hyper-rectangle $\rightarrow O(2^d)$ evaluations)

Lipschitzian Optimization Without the Lipschitz Constant [Jones, Perttunen and Stuckman (1993)]

- The name DIRECT stands fro DIviding RECTangles
- As in Shubert's algorithm, DIRECT balances between global and local search
- Two important ideas:
  - K need not to be known
  - Sample the function at center of rectangles

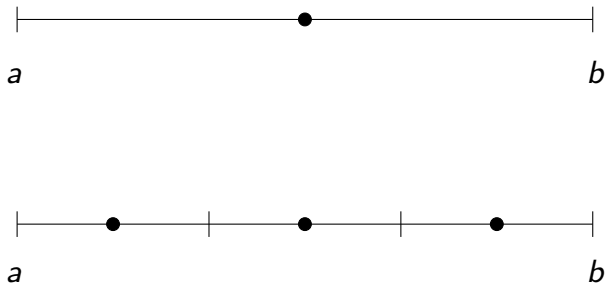
DIRECT in 1D

- Sample at the center of an interval



- When dividing the search domain, we have to make sure that previous function evaluations remain at the center of some interval

  $\rightarrow$ Instead of a bisection, do a trisection.

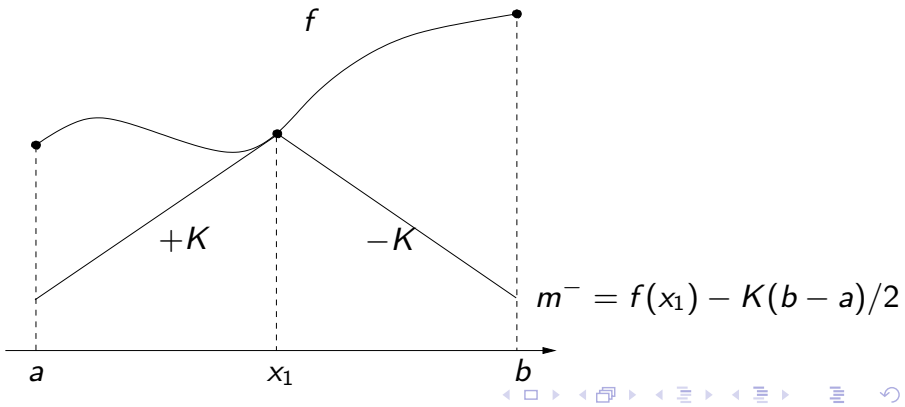


DIRECT in 1D

- Assume:
  - $\mathbb{X} = [a, b]$, with $-\infty < a < b < +\infty$
  - f is Lipschitz continuous, with Lipschitz constant K
- Lipschitz bounds on an interval $[a_i, b_i]$ with midpoint x_i

$$\begin{aligned} f(x) &\geq f(x_i) + K(x - x_i), \quad \text{for } a_i \leq x \leq x_i, \\ f(x) &\geq f(x_i) - K(x - x_i), \quad \text{for } x_i \leq x \leq b_i \end{aligned}$$

- On $[a_i, b_i]$, f is lower-bounded by $m_i^- = f(x_i) - K(b_i - a_i)/2$
- Note that m_i^- only takes into account the function value at the center of the interval



DIRECT in 1D

Definition

- Let S be a partition of $[a, b]$ into subintervals $[a_i, b_i]$ with midpoints x_i, $i = 1, \ldots, n$
- An interval $[a_i, b_i]$ is called potentially optimal if there exists some constant $K \geq 0$ such that the following conditions hold

$$\begin{aligned} m_i^-(K) &\leq m_j^-(K), \quad \text{for all } j \in \{1, \ldots, n\} \\ m_i^-(K) &\leq \min_j f(x_j), \end{aligned}$$

with $m_i^-(K) := f(x_i) - K(b_i - a_i)/2$

(K need not to be a Lipschitz constant)

DIRECT in 1D

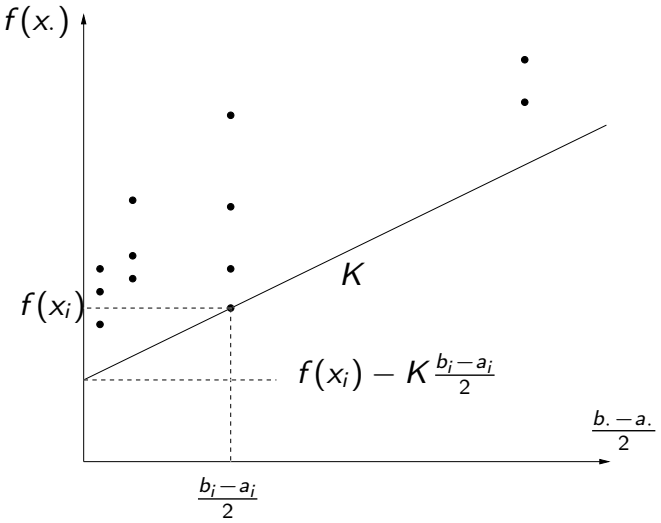
Fix an arbitrary $K > 0$ and consider a potentially optimal interval $[a_i, b_i]$

$$\begin{aligned} f(x_i) - K(b_i - a_i)/2 &\leq f(x_j) - K(b_j - a_j)/2, \quad \text{for all } j \in \{1, \ldots, n\} \\ f(x_i) - K(b_i - a_i)/2 &\leq \min_j f(x_j) \end{aligned}$$

DIRECT in 1D

Fix an arbitrary $K > 0$ and consider a potentially optimal interval $[a_i, b_i]$

$$\begin{aligned} f(x_i) - K(b_i - a_i)/2 &\leq f(x_j) - K(b_j - a_j)/2, \quad \text{for all } j \in \{1, \ldots, n\} \\ f(x_i) - K(b_i - a_i)/2 &\leq \min_j f(x_j) \end{aligned}$$

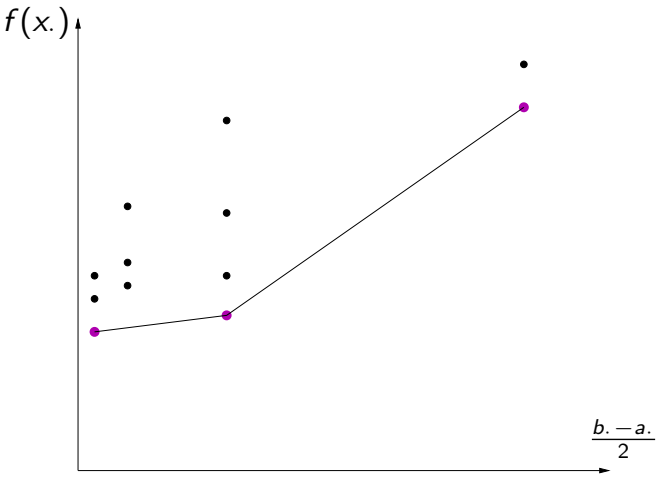


The interval with the lowest lower bound can be found by positioning a line with slope K below the cloud of dots, and shifting it upwards until it touches a dot.

DIRECT in 1D

Lipschitz constant $K > 0$ is unknown

$$\begin{aligned} f(x_i) - K(b_i - a_i)/2 &\leq f(x_j) - K(b_j - a_j)/2, \quad \text{for all } j \in \{1, \ldots, n\} \\ f(x_i) - K(b_i - a_i)/2 &\leq \min_j f(x_j) \end{aligned}$$



We identify the set of intervals that could be selected by using some positive K

DIRECT in 1D

One-dimensional DIRECT algorithm

Require: $a, b \in \mathbb{R}$, $f : [a, b] \to \mathbb{R}$
  Set $[a_1, b_1] = [a, b]$
  while the budget of evaluations is not exhausted; **do**
    1. Identify the set of potentially optimal intervals
    2. Subdivide potentially optimal intervals and evaluate new center points
  end while
  return $\widehat{m}_N$

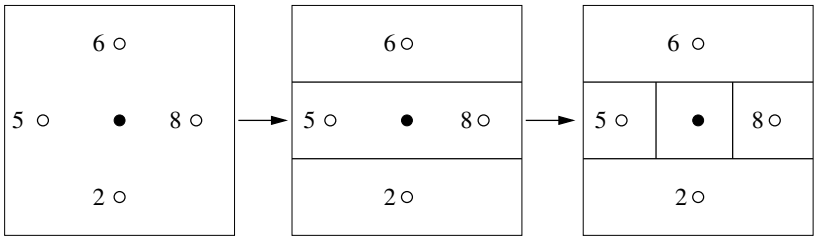
DIRECT in several dimensions

- Consider $f : \mathbb{X} = [0, 1]^d \to \mathbb{R}$, with $d > 1$
- As DIRECT proceeds, $\mathbb{X}$ will be partitioned into hyper-rectangles, each with a sampled point at its center

Division of hypercubes

Assume that x_1 is at the center of the initial hypercube $[0, 1]^d$

1. Evaluate f at $x_1 \pm \frac{1}{3} e_j$, $j = 1, \ldots, d$, where e_j stands for the j^{th} unit vector
2. Subdivide along directions with best function values first



The best values will be at the center of hyper-rectangles

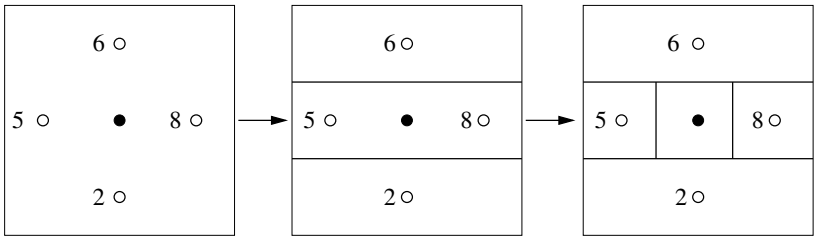
DIRECT in several dimensions

- Consider $f : \mathbb{X} = [0,1]^d \to \mathbb{R}$, with $d > 1$
- As DIRECT proceeds, $\mathbb{X}$ will be partitioned into hyper-rectangles, each with a sampled point at its center

Division of hypercubes

Assume that x_1 is at the center of the initial hypercube $[0,1]^d$

1. Evaluate f at $x_1 \pm \frac{1}{3} e_j$, $j = 1, \ldots, d$, where e_j stands for the j^{th} unit vector
2. Subdivide along directions with best function values first



The best values will be at the center of hyper-rectangles

DIRECT in several dimensions

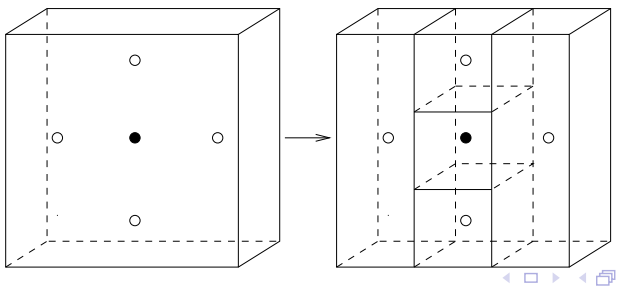
- Once the initial hypercube has been partitioned, some of the subregions will be hyper-rectangles.
- By dividing the hyper-rectangles only along the long dimensions, we ensure that the rectangles shrink on every dimension.

Division of a hyper-rectangle with center x_i

- Identify the set J of dimensions with the maximum edge length. Set δ equal to $1/3$ this maximum edge length.
- Sample the function at $x_i \pm \delta e_j$, $j \in J$.
- Divide the hyper-rectangle containing x_i along the dimensions in J, starting with the dimensions with the lowest value of

$$w_j = \min\{f(x_i - \delta e_j), f(x_i + \delta e_j)\}$$

and continuing with the dimensions with higher w_j.



DIRECT in several dimensions

- The procedure for identifying the set of potentially optimal rectangles in the same as that in one dimension
- For each rectangle with a point x_i at its center, we will know the function value at x_i and the distance d_i from the center point to the vertices.
- We can form a diagram like that on Slide 37, using the distance $d.$ for the horizontal axis, and identify the set of potentially optimal rectangles as before

DIRECT algorithm

Require: $\mathbb{X} \subset \mathbb{R}^d$, $f : \mathbb{X} \to \mathbb{R}$
- Normalize the domain to be the unit hyper-cube with center x_1
- Evaluate f at x_1 and set $\widehat{m}_1 = f(x_1)$, $n = 1$
- Evaluate $f(x_1 \pm \delta e_i)$, $1 \leq i \leq d$, and divide hyper-cube

while the budget of evaluations is not exhausted ($n \leq N$); **do**
- Identify the set of potentially optimal hyper-rectangles

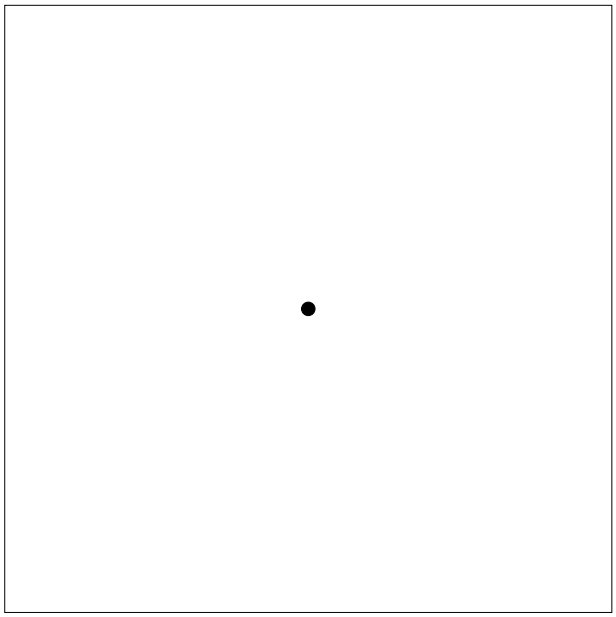
for all potentially optimal rectangles **do**
- Identify the longest side(s) of rectangle
- Divide into smaller rectangles, and evaluate f at centers of new rectangles
- Update $\widehat{m}_n$

end for

end while

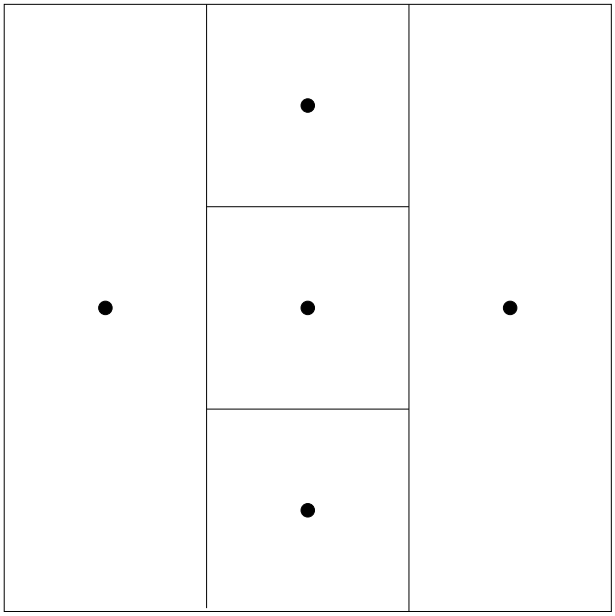
return $\widehat{m}_N$

DIRECT algorithm



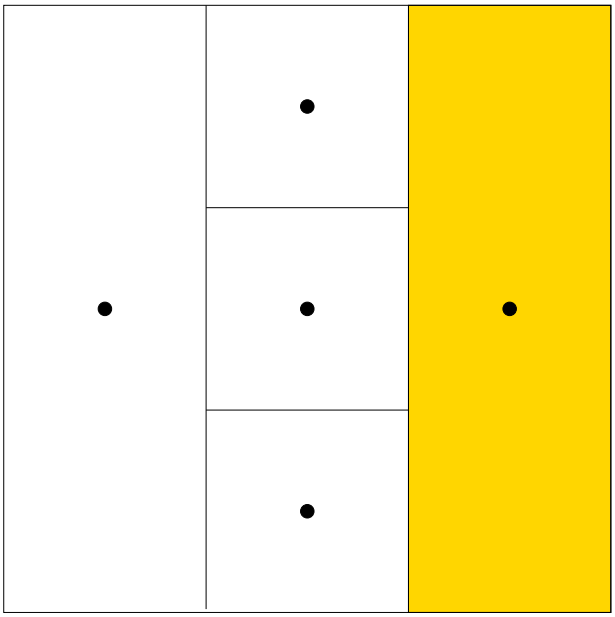
(Only divide along the set of longest sides. Rectangles have edge lengths either 3^{-k} or $3^{-(k+1)}$, $k \in \mathbb{N}$.)

DIRECT algorithm



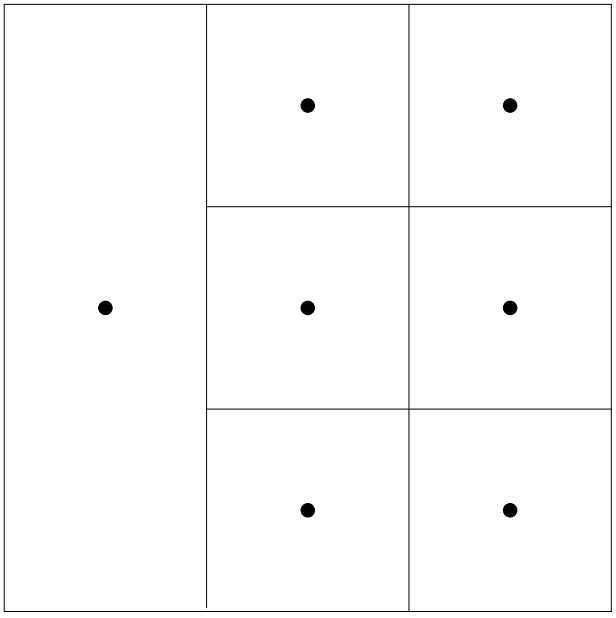
(Only divide along the set of longest sides. Rectangles have edge lengths either 3^{-k} or $3^{-(k+1)}$, $k \in \mathbb{N}$.)

DIRECT algorithm



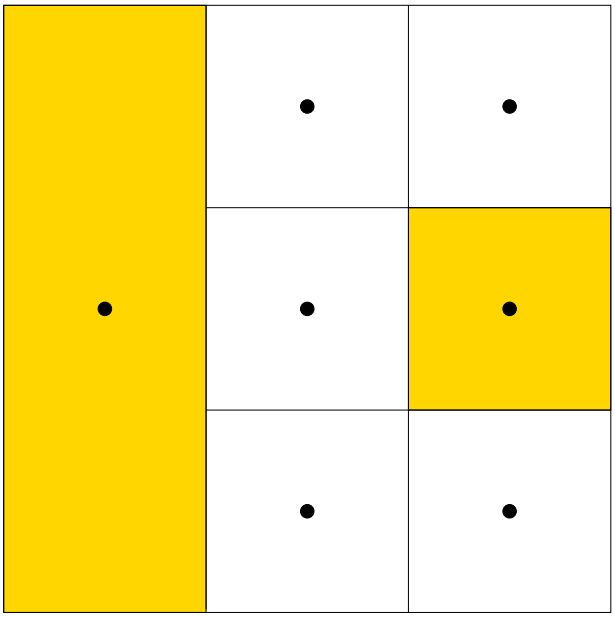
(Only divide along the set of longest sides. Rectangles have edge lengths either 3^{-k} or $3^{-(k+1)}$, $k \in \mathbb{N}$.)

DIRECT algorithm



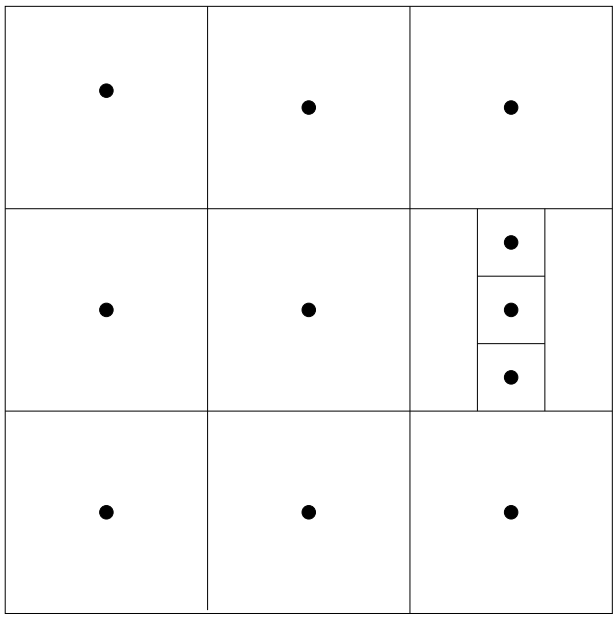
(Only divide along the set of longest sides. Rectangles have edge lengths either 3^{-k} or $3^{-(k+1)}$, $k \in \mathbb{N}$.)

DIRECT algorithm



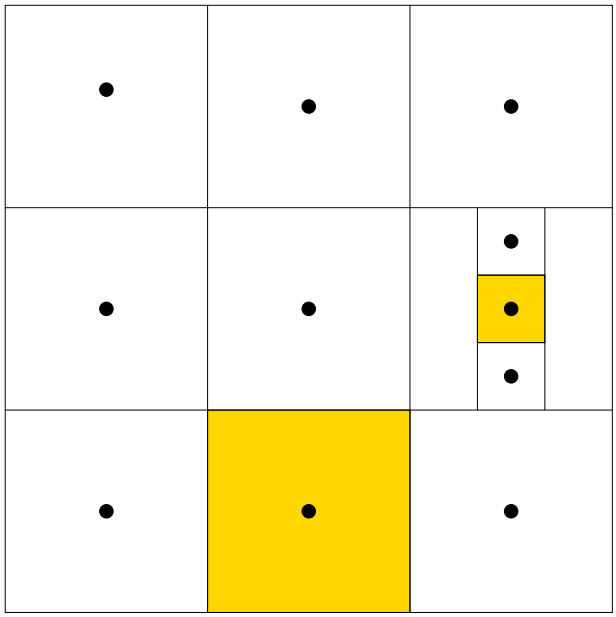
(Only divide along the set of longest sides. Rectangles have edge lengths either 3^{-k} or $3^{-(k+1)}$, $k \in \mathbb{N}$.)

DIRECT algorithm



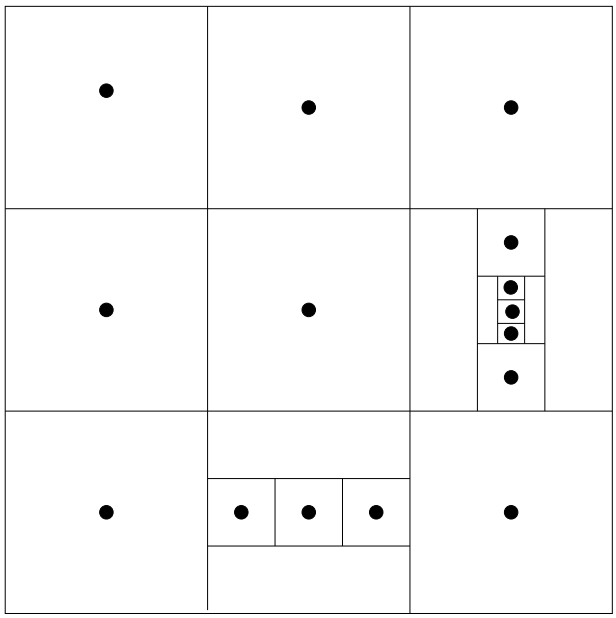
(Only divide along the set of longest sides. Rectangles have edge lengths either 3^{-k} or $3^{-(k+1)}$, $k \in \mathbb{N}$.)

DIRECT algorithm



(Only divide along the set of longest sides. Rectangles have edge lengths either 3^{-k} or $3^{-(k+1)}$, $k \in \mathbb{N}$.)

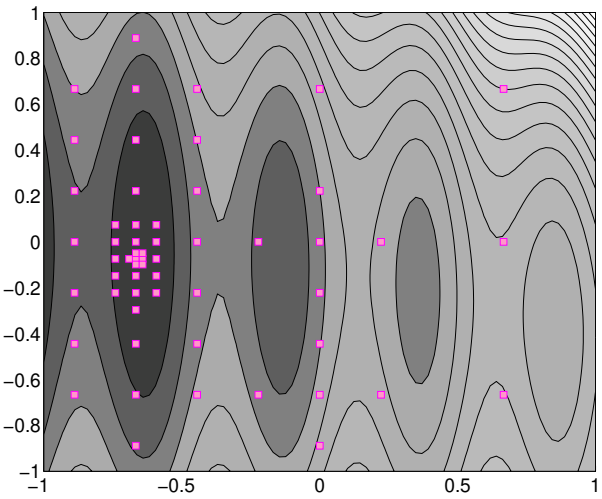
DIRECT algorithm



(Only divide along the set of longest sides. Rectangles have edge lengths either 3^{-k} or $3^{-(k+1)}$, $k \in \mathbb{N}$.)

DIRECT: 2D illustration

(f defined on Slide 19)



	$\widehat{m}_n$ with $N = 60$
LHS	-5.823
DIRECT	-5.839
Global minimum	-5.845

Lipschitzian Optimization Without the Lipschitz Constant [Jones, Perttunen and Stuckman (1993)]

Summing up

- In the context of risk analysis, it is often desirable to assess the worst-case performance of a system $\rightarrow$ this is a global optimization problem
- In this context, we want to use robust optimization algorithms

DIRECT in practice

- Straightforward and efficient global optimization procedure
- Global search and local search
- Known convergence results

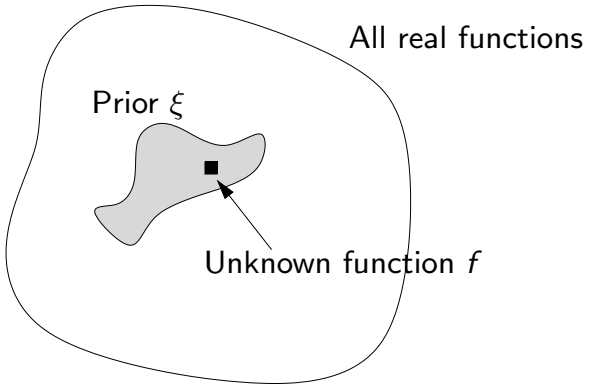
3.5 Average-case approach to the problem of optimization

Average-case approach

- Summing up: the worst case setting may not be the most appropriate framework to assess the performance of an algorithm in practice
- In practice, we need to know how an optimization algorithm performs for "typical" functions f not corresponding to worst cases
- To address this issue, a classical approach is to adopt an average-case point of view

Average-case approach

- Average-case $\rightarrow$ introduction of a probability space $(\Omega, \mathcal{B}, \mathrm{P}_0)$
- We consider methods where f is seen as a sample path of a real-valued random process ξ defined on $(\Omega, \mathcal{B}, \mathrm{P}_0)$ with parameter in $\mathbb{X}$
  $\rightarrow$ there exists $\omega \in \Omega$ such that
  $$f = \xi(\omega, \cdot)$$
- From a Bayesian decision-theoretic point of view, ξ represents prior knowledge about f



Average-case approach

- Under this setting, for a given sequential strategy $\underline{X}_n \in \mathcal{A}_n$
  $$\underline{X}_n(\xi) = (X_1(\xi), \ldots, X_n(\xi))$$
  is a random sequence in $\mathbb{X}$
  (but we still consider deterministic strategies!)
- The performance of a strategy $\underline{X}_n \in \mathcal{A}_N$ is the random variable
  $$\varepsilon(\underline{X}_N, \xi) = \widehat{m}_N(\xi) - m(\xi)$$
- A good strategy is a strategy that achieves, or gets close to, the average (or Bayes) risk
  $$r_{\mathrm{average}} := \inf_{\underline{X}_N \in \mathcal{A}_N} \mathrm{E}_0 \left(\varepsilon(\underline{X}_N, \xi) \right),$$
  where E_0 denotes expectation with respect to the probability P_0

Average-case approach

- This point of view has been widely explored in the domain of optimization and computer experiments.
- Two important issues to address:
  - How to construct an optimal average strategy?
  - How to choose ξ?

3.6 Sequential Bayesian optimization

Optimal Bayesian strategies

- Objective: construct an optimal Bayesian optimization strategy $\underline{X}_N^\star$ such that

$$\mathrm{E}_0\left(\varepsilon(\underline{X}_N^\star, \xi)\right) = r_{\mathrm{average}} = \inf_{\underline{X}_N \in \mathcal{A}_N} \mathrm{E}_0\left(\varepsilon(\underline{X}_N, \xi)\right)$$

- Let E_n, $n = 1, 2, \ldots$, denote the conditional expectation with respect to $\mathcal{I}_n(\xi)$, where
  - $\mathcal{I}_n = \left((X_1, Z_1), \ldots, (X_n, Z_n)\right)$
  - $Z_n(\xi) = \xi(X_n(\xi))$, $1 \leq n \leq N$
- $\underline{X}_N^\star$ can be formally obtained by dynamic programming
- Denote the terminal risk by

$$R_N = \mathrm{E}_N\left(\varepsilon(\underline{X}_N, \xi)\right)$$

  and define by backward induction

$$R_n = \min_{x \in \mathbb{X}} \mathrm{E}_n\big(R_{n+1} \mid X_{n+1} = x\big), \quad n = N-1, \ldots, 0. \tag{1}$$

- To get an insight into (1), notice that R_{n+1}, $n = 0, \ldots, N-1$, depends measurably on $\mathcal{I}_{n+1} = (\mathcal{I}_n, X_{n+1}, Z_{n+1})$, so that

$$\mathrm{E}_n\big(R_{n+1} \mid X_{n+1} = x\big)$$

  is in fact an expectation with respect to Z_{n+1}, and R_n is an $\mathcal{F}_n$-measurable random variable

Optimal Bayesian strategies

- Then, we have $R_0 = r_{\mathrm{average}}$
- The strategy $\underline{X}_N^\star$ defined by

$$X_{n+1}^\star = \operatorname*{argmin}_{x \in \mathbb{X}} \mathrm{E}_n\big(R_{n+1} \mid X_{n+1} = x\big), \quad n = 1, \ldots, N-1, \tag{2}$$

  is the optimal Bayesian strategy
- Unfortunately,

state space is continuous, dim. $n \times (d+1)$
action space is continuous, dim. d

$\Rightarrow$ solving (1)–(2) over an horizon of more than a few steps is not numerically tractable!

- How to construct good sub-optimal strategies?

k-step lookahead strategies

- Using (1), the optimal strategy can be expanded as

$$X_{n+1}^{\star} = \underset{x\in\mathbb{X}}{\operatorname{argmin}}\ \mathrm{E}_n\left(\min_{X_{n+2}} \mathrm{E}_{n+1} \ldots \min_{X_N} \mathrm{E}_{N-1}\, R_N \,\middle|\, X_{n+1} = x\right).$$

- A general approach to construct sub-optimal strategies is to truncate this expansion after k terms, replacing the exact risk R_{n+k} by a surrogate risk $\widetilde{R}_{n+k}$.
- Examples of such surrogates will be given below
- The resulting strategy,

$$X_{n+1} = \underset{x\in\mathbb{X}}{\operatorname{argmin}}\ \mathrm{E}_n\left(\min_{X_{n+2}} \mathrm{E}_{n+1} \ldots \min_{X_{n+k}} \mathrm{E}_{n+k-1}\, \widetilde{R}_{n+k} \,\middle|\, X_{n+1} = x\right).$$

  is called a k-step lookahead strategy
- Both the optimal strategy (2) and the k-step lookahead strategy implicitly define a sampling criterion $J_n(x)$, the minimum of which indicates the next evaluation to be performed.
- For instance, in the case of the k-step lookahead strategy, the sampling criterion is

$$J_n(x) = \mathrm{E}_n\left(\min_{X_{n+2}} \mathrm{E}_{n+1} \ldots \min_{X_{n+k}} \mathrm{E}_{n+k-1}\, \widetilde{R}_{n+k} \,\middle|\, X_{n+1} = x\right).$$

One-step lookahead strategy for the problem of optimization

- In the case of a one-step lookahead strategy the sampling criterion may be written as

$$J_n(x) = \mathrm{E}_n\left(\widetilde{R}_{n+1} \,\middle|\, X_{n+1} = x\right)$$

  and, at step n, the next evaluation point is chosen according to

$$X_{n+1} = \underset{x\in\mathbb{X}}{\operatorname{argmin}}\ J_n(x)$$

- How to define a surrogate $\widetilde{R}_{n+1}$ for the optimization problem?
- For instance, we can choose the next evaluation point as if it were the last one

  $\rightarrow$ in this case, we set $\widetilde{R}_{n+1} = \mathrm{E}_{n+1}(\widehat{m}_{n+1} - m)$

- Note that taking $\widetilde{R}_{n+1} = \mathrm{E}_{n+1}(\widehat{m}_{n+1} - m)$ corresponds to considering an optimal strategy for an horizon of one evaluation only

One-step lookahead strategy for the problem of optimization

- Consider the one-step lookahead strategy for the problem of optimization: each new evaluation point is chosen according to

$$\begin{aligned} X_{n+1} &= \operatorname*{argmin}_{x\in\mathbb{X}} \mathsf{E}_n\left(\mathsf{E}_{n+1}(\widehat{m}_{n+1} - m) \mid X_{n+1} = x\right) \\ &= \operatorname*{argmin}_{x\in\mathbb{X}} \mathsf{E}_n\left(\widehat{m}_{n+1} - m \mid X_{n+1} = x\right) \\ &= \operatorname*{argmin}_{x\in\mathbb{X}} \mathsf{E}_n\left(\widehat{m}_{n+1} \mid X_{n+1} = x\right) \\ &= \operatorname*{argmin}_{x\in\mathbb{X}} \mathsf{E}_n\left(\widehat{m}_n \wedge \xi(X_{n+1}) \mid X_{n+1} = x\right) \\ &= \operatorname*{argmax}_{x\in\mathbb{X}} \mathsf{E}_n\left(0 \wedge (\xi(X_{n+1}) - \widehat{m}_n) \mid X_{n+1} = x\right) \\ &= \operatorname*{argmax}_{x\in\mathbb{X}} \rho_n(x) := \mathsf{E}_n\left((\widehat{m}_n - \xi(X_{n+1}))_+ \;\middle|\; X_{n+1} = x\right), \end{aligned}$$

  where $(z)_+ = 0 \vee z$
- The sampling criterion ρ_n, introduced by J. Mockus and popularized through the EGO algorithm [Jones et al. (1998)], is known as the expected improvement (EI).

One-step lookahead strategy for the problem of optimization

- Heuristic interpretation of the EI sampling criterion

$$X_{n+1} = \operatorname*{argmax}_{x\in\mathbb{X}} \rho_n(x) := \mathsf{E}_n\left((\widehat{m}_n - \xi(X_{n+1}))_+ \;\middle|\; X_{n+1} = x\right),$$

- For $x \in \mathbb{X}$, the random variable

$$(\widehat{m}_n - \xi(x))_+$$

  is called the improvement at x, and represents the excursion of $\xi(x)$ below the current minimum

$$\widehat{m}_n = \xi(X_1) \wedge \cdots \wedge \xi(X_n)$$

- A one-step lookahead strategy selects the point which has the maximum expected improvement

- The next step is to understand:
  - how to choose a random process ξ
  - how to compute a sampling criterion such as the expected improvement ρ_n
- We shall see that restricting ξ to be a Gaussian process makes it possible to obtain a closed-form formula for ρ_n

3.7 Gaussian random models

Gaussian random models

- The idea of modeling an unknown function f by a Gaussian process has originally been introduced in the 60s
  - in time series analysis
  - in optimization theory
  - in geostatistics
- Today, the Gaussian process model plays a central role in the design and analysis of computer experiments
- A Gaussian random process ξ can be used as a stochastic model of some uncertain real-valued function
- In other words, ξ can be thought as a prior about some function

Gaussian random vector

- A real-valued random vector $X = (X_1, \ldots, X_d) \in \mathbb{R}^d$ is said to be Gaussian if and only if any linear combination of its components $\sum_{i=1}^d a_i X_i$, with $a_1, \ldots, a_d \in \mathbb{R}$, is a Gaussian variable
- A Gaussian random vector X is characterized by its mean vector

$$\mu = (\mathrm{E}[X_1], \ldots, \mathrm{E}[X_d]) \in \mathbb{R}^d$$

and the covariance of the pairs of components (X_i, X_j), $i, j \in \{1, \ldots, d\}$

$$\mathrm{cov}(X_i, X_j) = \mathrm{E}[X_i X_j] - \mathrm{E}[X_i]\mathrm{E}[X_j]$$

- If the covariance matrix

$$\Sigma = \left(\mathrm{cov}(X_i, X_j)\right)_{i,j=1,\ldots,d}$$

is nonsingular, X has the probability density function

$$g_{\mu,\Sigma}(x) = \frac{1}{(2\pi)^{d/2}(\det \Sigma)^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^{\mathrm{T}}\Sigma^{-1}(x-\mu)\right)$$

- The correlation coefficient of two components X_i, X_j is defined by

$$\rho(X_i, X_j) = \frac{\mathrm{cov}(X_i, X_j)}{\sqrt{\mathrm{var}(X_i)\,\mathrm{var}(X_j)}} \in [-1, 1],$$

$\rightarrow$ measures the similarity between X_i and X_j

Gaussian random process

- Random process: a set $\xi = \{\xi(x), x \in \mathbb{X}\}$ of random variables indexed by the elements of $\mathbb{X}$
- A random process ξ is Gaussian if and only if, $\forall n \in \mathbb{N}$, $\forall x_1, \ldots, x_n \in \mathbb{X}$, and $\forall a_1, \ldots, a_n \in \mathbb{R}$, the real-valued random variable

$$\sum_{i=1}^{n} a_i \xi(x_i)$$

  is Gaussian
- A Gaussian process is characterized by its mean function

$$x \in \mathbb{X} \mapsto \mathrm{E}[\xi(x)]$$

  and its covariance function

$$(x, y) \in \mathbb{X}^2 \mapsto \mathrm{cov}(\xi(x), \xi(y))$$

- Standing assumption: the covariance function is stationary, i.e., there exists $k : \mathbb{X} \to \mathbb{R}$ such that

$$\mathrm{cov}(\xi(x), \xi(y)) = k(x - y)$$

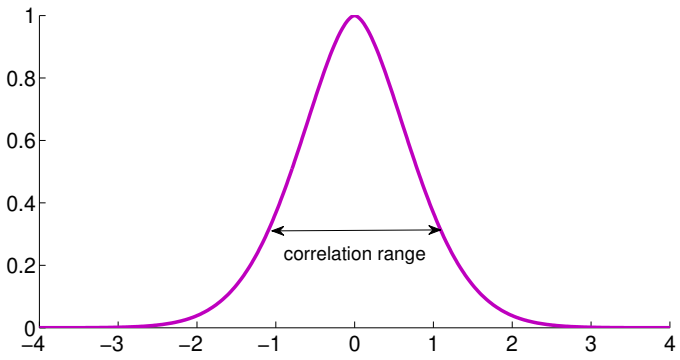
- Notation: $\xi \sim \mathrm{GP}(m, k)$

Gaussian process: correlation structure

- When k is stationary, the variance $\mathrm{var}(\xi(x))$ does not depend on x
- The covariance function can be written as

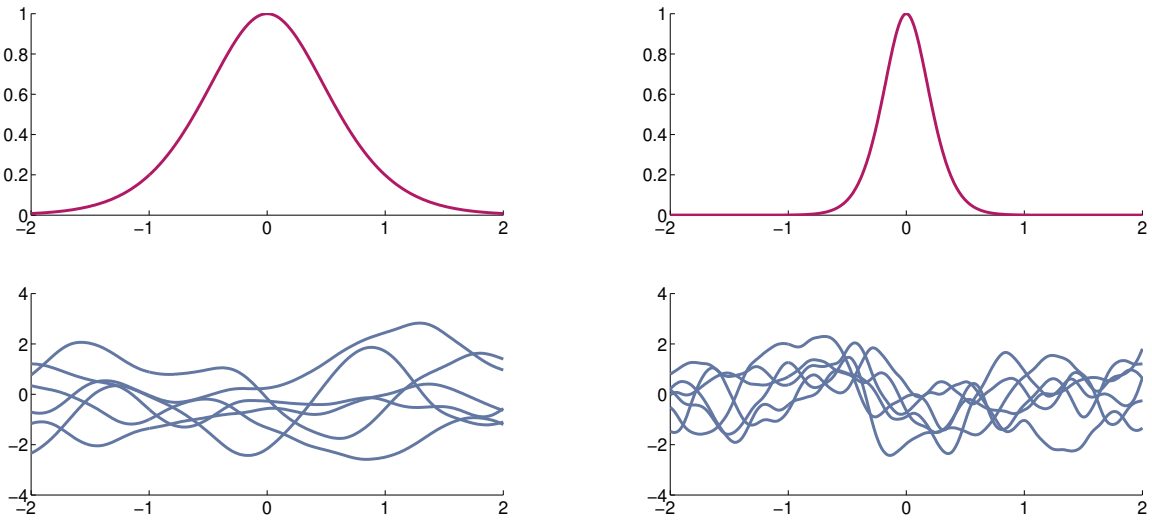
$$k(x - y) = \sigma^2 \rho(x - y),$$

  with $\sigma^2 = \mathrm{var}(\xi(x))$, and where ρ is the correlation function of ξ.
- The graph of the correlation function is a symmetric "bell curve" shape



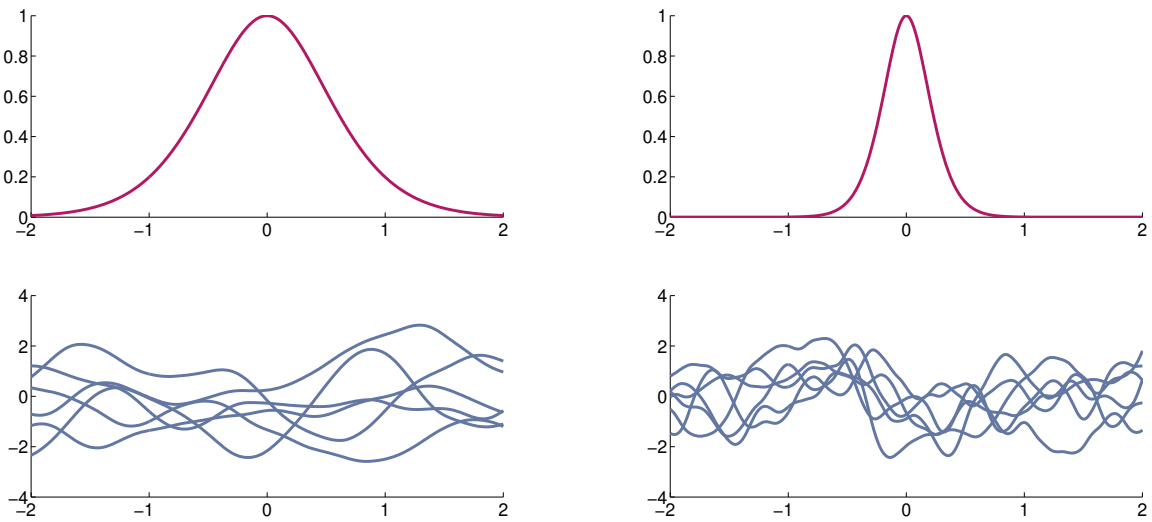
Gaussian process simulation

- Using a random generator, it is possible to "generate" sample paths $f_1, f_2, \ldots$ of a Gaussian process ξ



Gaussian process simulation

- Using a random generator, it is possible to "generate" sample paths $f_1, f_2, \ldots$ of a Gaussian process ξ



Regularity properties of a random process

Definition

Given $x_0 \in \mathbb{R}^d$, a random process ξ is said to be continuous in mean-square at x_0 iff

$$\lim_{x \to x_0} \mathrm{E}[(\xi(x) - \xi(x_0))^2] = 0$$

Proposition

Let ξ be a second-order random process with continuous mean function and stationary covariance function k. ξ is continuous in mean-square iff k is continuous at zero.

Regularity properties of a random process

Definition

For $x, h \in \mathbb{R}^d$, define the random variable

$$\xi_h(x) = \frac{\xi(x_0 + h) - \xi(x_0)}{\|h\|}$$

ξ is mean-square differentiable at x_0 iff there exists a random vector $\nabla\xi(x_0)$ such that

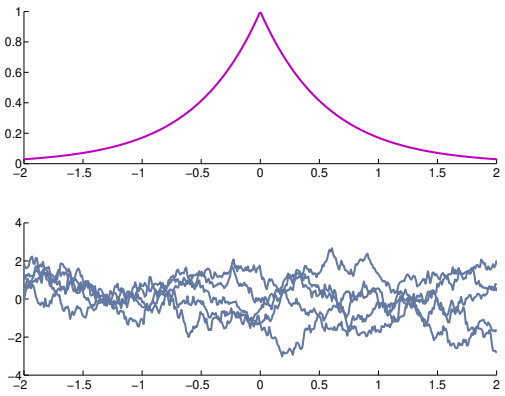
$$\lim_{h \to 0} \mathrm{E}\left[\left(\xi_h(x_0) - (\nabla\xi(x_0), h)\right)^2 \right] = 0$$

Proposition

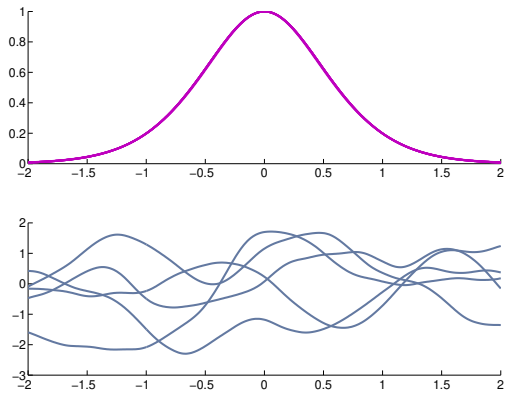
Let ξ be a second-order random process with differentiable mean function and stationary covariance function k. ξ is differentiable in mean-square iff k is two-time differentiable at zero.

Influence of the regularity

mean-square continuity



three-time mean-square differentiability



Choice of a covariance

- A Gaussian process prior carries a high amount of information about f

  $\rightarrow$ it is often difficult to elicit such a prior before any evaluation is made

- Covariance function of ξ is usually assumed to belong to some parametric class of positive definite functions
- Parameter values assumed to be unknown
- Two approaches:

  1. The parameters can be estimated from the evaluation results by maximum likelihood, and then plugged in a sampling criterion
  2. We can assume a prior distribution for the parameters of the covariance and use a fully Bayesian approach

Choice of a parametrized covariance function: the Matérn covariance

- The Matérn covariance function is a conventional covariance function in the literature of computer experiments
  $\rightarrow$ offers the possibility to adjust the regularity of ξ with a single parameter
- The Matérn function:

$$\kappa_\nu(h) = \frac{1}{2^{\nu-1}\Gamma(\nu)} \left(2\nu^{1/2}h\right)^\nu \mathcal{K}_\nu \left(2\nu^{1/2}h\right), \quad h \in \mathbb{R} \tag{3}$$

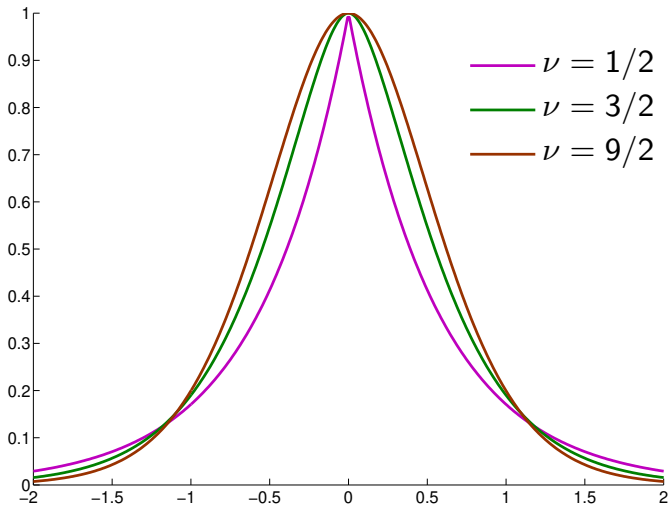
  with
  - Γ the Gamma function
  - $\mathcal{K}_\nu$ the modified Bessel function of the second kind
- To model a real-valued function defined over $\mathbb{X} \subset \mathbb{R}$, we use the Matérn covariance:

$$k_\theta(h) = \sigma^2 \kappa_\nu(|h|/\rho), \quad h \in \mathbb{R} \tag{4}$$

  - $\sigma^2 > 0$ is a variance parameter (we have $k_\theta(0) = \sigma^2$)
  - $\rho > 0$ is a scale or *range* parameter, *i.e.*, characteristic correlation length

Choice of a parametrized covariance function: the Matérn covariance

Matérn covariance in one dimension $\sigma^2 = 1$, $\rho = 0.8$



ξ is p-time mean-square differentiable iff $\nu > p$

Choice of a parametrized covariance function: the Matérn covariance

- To model a function f defined over $\mathbb{X} \subset \mathbb{R}^d$, with $d > 1$, we use the anisotropic form of the Matérn covariance:

$$k_\theta(x, y) = \sigma^2 \kappa_\nu \left(\sqrt{\sum_{i=1}^{d} \frac{(x_{[i]} - y_{[i]})^2}{\rho_i^2}} \right), \quad x, y \in \mathbb{R}^d \tag{5}$$

  where $x_{[i]}, y_{[i]}$ denote the i^{th} coordinate of x and y, and the positive scalars ρ_i represent scale parameters
- Since $\sigma^2 > 0, \nu > 0, \rho_i > 0$, $i = 1, \ldots, d$, in practice, we consider the vector of parameters

$$\theta = \{\log \sigma^2, \log \nu, -\log \rho_1, \ldots, -\log \rho_d\} \in \mathbb{R}^{d+2}$$

  $\rightarrow$ makes parameter estimation easier

Parameter estimation by maximum likelihood

- Assume ξ is a zero-mean Gaussian process
- The log-likelihood of the data $\underline{\xi}_n = (\xi(x_1), \ldots, \xi(x_n))^{\mathsf{T}}$ can be written as

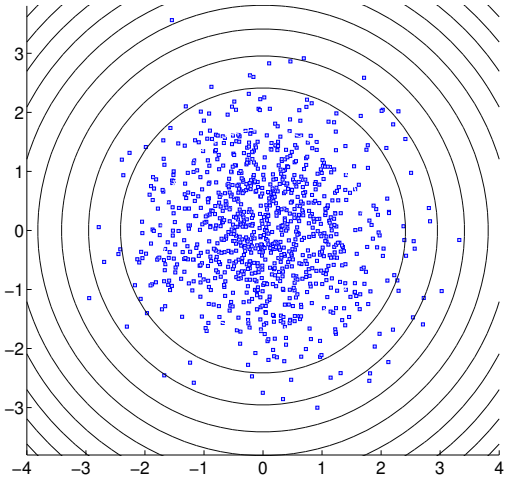
$$\ell(\underline{\xi}_n; \theta) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log \det \mathbf{K}(\theta) - \frac{1}{2} \underline{\xi}_n{}^{\mathsf{T}} \mathbf{K}(\theta)^{-1} \underline{\xi}_n, \tag{6}$$

  where $\mathbf{K}(\theta)$ is the covariance matrix of $\underline{\xi}_n$, which depends on the parameter vector θ
- The log-likelihood can be maximized using a gradient-based search method
- If the mean of ξ is polynomial and unknown, use restricted maximum likelihood instead

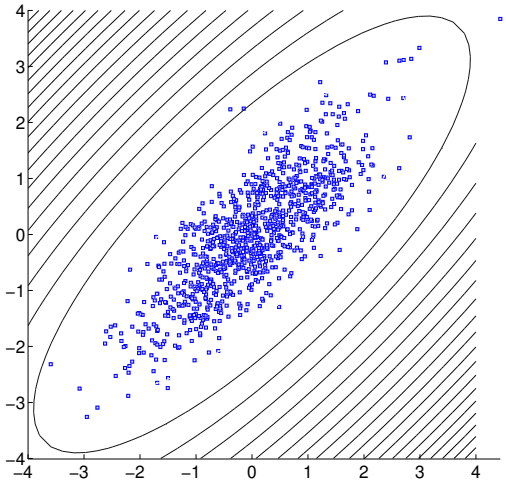
Correlation, prediction, conditioning

- Consider a pair of random variables $(\xi(x_i), \xi(x_j))$, for $x_i \in \mathbb{X}$ and $x_j \in \mathbb{X}$
- If $x_i \in \mathbb{X}$ and $x_j \in \mathbb{X}$ are far apart, $\xi(x_i)$ and $\xi(x_j)$ are typically uncorrelated
- If $x_i \in \mathbb{X}$ and $x_j \in \mathbb{X}$ are close, $\xi(x_i)$ and $\xi(x_j)$ are typically correlated

$\rho(x_i - x_j) = 0$



$\rho(x_i - x_j) = 0.8$

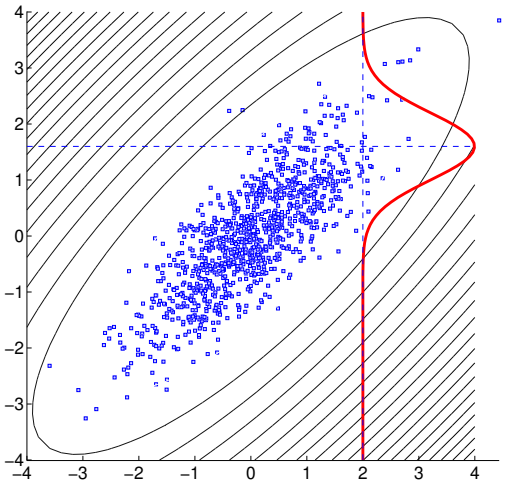


- Correlation makes prediction possible

Correlation, prediction, conditioning

- Correlation $\rightarrow$ we can predict $Z_j = \xi(x_j)$ from the observation of $Z_i = \xi(x_i)$
- Define the conditional density function by

$$f_{Z_j|Z_i}(v|Z_i = u) = \frac{f_{(Z_j,Z_i)(v,u)}}{f_{Z_i}(u)}$$



- The random variable denoted by $Z_j \mid Z_i$ with density $f_{Z_j|Z_i}$ represents the residual uncertainty about Z_j when Z_i has been observed

Correlation, prediction, conditioning

- The conditional mean is defined by

$$\mathrm{E}_0[Z_j \mid Z_i = u] = h(u) = \int_{\mathbb{R}} v\, f_{Z_j|Z_i}(v|Z_i = u)dv$$

- The random variable $\widehat{Z}_j = \mathrm{E}_0[Z_j \mid Z_i] = h(Z_i)$ minimizes $\mathrm{E}_0\left[(\widehat{Z}_j - Z_j)^2\right]$
- Important properties:

$$\exists \lambda \in \mathbb{R}\,, \text{ such that } \widehat{Z}_j = h(Z_i) = \lambda Z_i$$
$$Z_j - \widehat{Z}_j \perp Z_i \iff Z_j - \widehat{Z}_j \perp \mathrm{span}\{Z_i\} \iff \mathrm{E}_0\left\{(Z_j - \widehat{Z}_j)Z_i\right\} = 0$$

- In other words, $\widehat{Z}_j$ is the orthogonal projection of Z_j onto $\mathrm{span}\{Z_i\}$

Prediction of a zero-mean Gaussian process

- Let $\xi \sim \mathrm{GP}(0, k)$
- The best linear unbiased predictor (BLUP) of $\xi(x)$ from observations $\xi(x_i)$, $i = 1, \ldots, n$, also called the kriging predictor of $\xi(x)$, is the orthogonal projection

$$\widehat{\xi}_n(x) := \sum_{i=1}^{n} \lambda_i(x; \underline{x}_n)\, \xi(x_i)$$

of $\xi(x)$ onto $\mathrm{span}\{\xi(x_i), i = 1, \ldots, n\}$

- Weights $\lambda_i(x; \underline{x}_n)$ are solutions of a system of linear equations

$$k(\underline{x}_n, \underline{x}_n)\lambda(x; \underline{x}_n) = k(\underline{x}_n, x) \tag{7}$$

with

  - $\lambda(x; \underline{x}_n) = (\lambda_1(x; \underline{x}_n), \ldots, \lambda_n(x; \underline{x}_n))^{\mathsf{T}}$
  - $k(\underline{x}_n, \underline{x}_n)$: $n \times n$ covariance matrix of the observation vector
  - $k(\underline{x}_n, x)$: $n \times 1$ vector with entries $k(x_i, x)$

- The function $x \mapsto \widehat{\xi}_n(x)$ conditioned on $\xi(x_1) = f(x_1), \ldots, \xi(x_n) = f(x_n)$, is deterministic, and provides a cheap approximation of the function f

Prediction of a zero-mean Gaussian process

- The covariance function of the error of prediction, also called kriging covariance is given by

$$\begin{aligned} k(x, y; \underline{x}_n) &:= \operatorname{cov}\left(\xi(x) - \widehat{\xi}(x; \underline{x}_n), \xi(y) - \widehat{\xi}(y; \underline{x}_n)\right) \\ &= k(x - y) - \sum_i \lambda_i(x; \underline{x}_n)\, k(y - x_i)\,. \end{aligned}$$

- The variance of the prediction error, also called the kriging variance, is defined as

$$\sigma_n^2(x) = k(x, x; \underline{x}_n)$$

Proposition

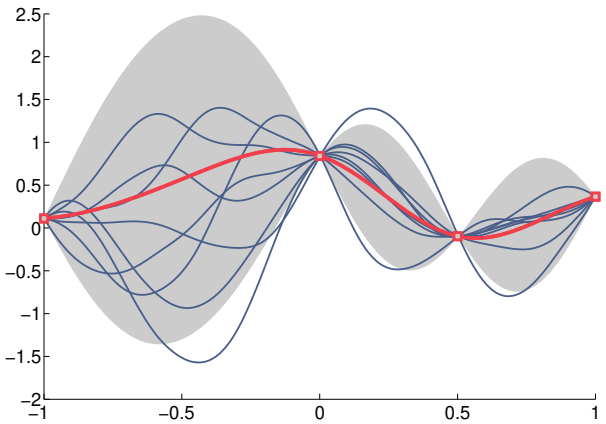
Let $\xi \sim \mathrm{GP}(0,\, k)$. Define $\xi \mid \mathcal{F}_n$ as the random process ξ conditioned on the σ-algebra $\mathcal{F}_n$ generated by $\xi(x_1), \ldots, \xi(x_n)$
$\rightarrow$ $\xi \mid \mathcal{F}_n$ is a Gaussian process with

– mean $\widehat{\xi}_n(\,\cdot\,)$
– covariance $k(\,\cdot\,, \cdot\,; \underline{x}_n)$

- In particular, $\widehat{\xi}_n(x) = \mathrm{E}_0\big(\xi(x) \mid \mathcal{F}_n\big)$ is the best $\mathcal{F}_n$-measurable predictor of $\xi(x)$, for all $x \in \mathbb{X}$.

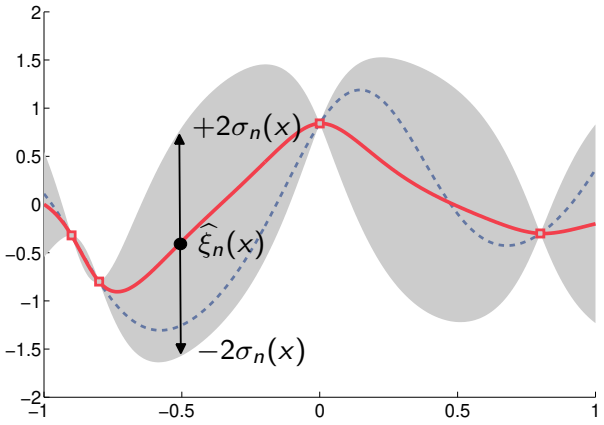
Gaussian process conditioned on observations

- For all $x \in \mathbb{X}$, the random variable $\xi(x) \mid \mathcal{F}_n$ with distribution $\mathrm{N}\big(\widehat{\xi}_n(x), \sigma_n^2(x)\big)$ represents the residual uncertainty about $f(x)$ when $\mathcal{F}_n$ is observed



Gaussian process conditioned on observations

- For all $x \in \mathbb{X}$, the random variable $\xi(x) \mid \mathcal{F}_n$ with distribution $\mathrm{N}\big(\widehat{\xi}_n(x), \sigma_n^2(x)\big)$ represents the residual uncertainty about $f(x)$ when $\mathcal{F}_n$ is observed



Prediction of a Gaussian process with unknown mean function

- In the domain of computer experiments, the mean of a Gaussian process is generally written as a linear parametric function

$$m(\,\cdot\,) = \beta^{\mathsf{T}} \varphi(\,\cdot\,)\,, \tag{8}$$

with
  - β a vector of unknown parameters
  - $\varphi = (\varphi_1, \ldots, \varphi_l)^{\mathsf{T}}$ an l-dimensional vector of functions (in practice, polynomials)

- Simplest case: the mean function is an unknown constant m, in which case $\beta = m$ and $\varphi : x \in \mathbb{X} \mapsto 1$

Prediction of a Gaussian process with unknown mean function

- Define the linear space of functions

$$\mathcal{P} = \left\{ x \mapsto \sum_{i=1}^{l} \beta_i \varphi_i(x);\ \beta_i \in \mathbb{R} \right\},$$

- Define Λ the linear space of finite-support measures on $\mathbb{X}$, i.e.

$$\lambda \in \Lambda \implies \lambda = \sum_{i=1}^{n} \lambda_i \delta_{x_i} \text{ for some } n \in \mathbb{N}$$

- For $f : \mathbb{X} \to \mathbb{R}$, and $\lambda = \sum_{i=1}^{n} \lambda_i \delta_{x_i} \in \Lambda$,

$$\langle \lambda, f \rangle = \int_{\mathbb{X}} f \mathrm{d}\lambda = \sum_{i=1}^{n} \lambda_i f(x_i)$$

- Define the linear subspace $\Lambda_{\mathcal{P}^\perp} \subset \Lambda$ of finite-support measures vanishing on $\mathcal{P}$, i.e.

$$\lambda \in \Lambda_{\mathcal{P}^\perp} \implies \langle \lambda, f \rangle = \int_{\mathbb{X}} f d\lambda = \sum_{i=1}^{n} \lambda_i f(x_i) = 0\,, \quad \forall f \in \mathcal{P}$$

Prediction of a Gaussian process with unknown mean function

- Let ξ be a Gaussian random process with an unknown mean in $\mathcal{P}$, and a covariance function k
- For $x \in \mathbb{X}$, the kriging predictor $\widehat{\xi}_n(x)$ of $\xi(x)$ from $\xi(x_1), \ldots, \xi(x_n)$ is the linear projection

$$\widehat{\xi}_n(x) = \sum_i \lambda_i(x; \underline{x}_n)\xi(x_i)$$

of $\xi(x)$ onto

$$\mathrm{span}\{\xi(x_i), i = 1, \ldots, n\}$$

such that the variance of the error $\xi(x) - \widehat{\xi}_n(x)$ is minimized, under the constraint

$$\delta_x - \sum \lambda_i(x; \underline{x}_n)\delta_{x_i} \in \Lambda_{\mathcal{P}^\perp}$$

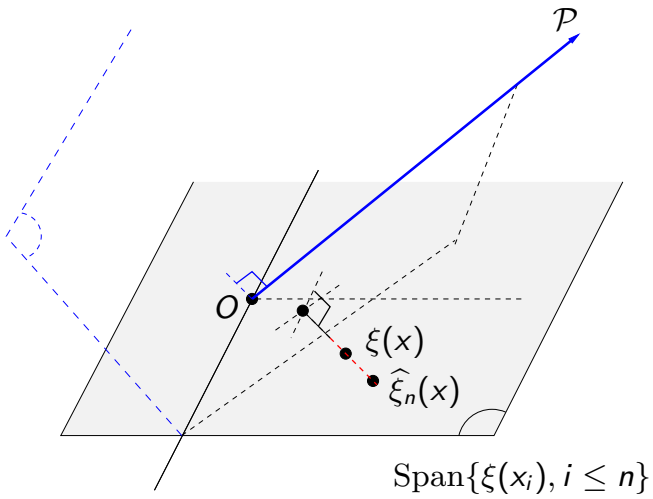
i.e.,

$$\langle \delta_x - \sum \lambda_i(x; \underline{x}_n)\delta_{x_i}, \varphi_j \rangle = \varphi_j(x) - \sum \lambda_i(x; \underline{x}_n)\varphi_j(x_i) = 0\,, \quad j = 1, \ldots, l$$

- The requirement $\delta_x - \sum \lambda_i(x; \underline{x}_n)\delta_{x_i} \in \Lambda_{\mathcal{P}^\perp}$ makes the kriging predictor unbiased, even if the mean of ξ is unknown

Prediction of a Gaussian process with unknown mean function

$\widehat{\xi}_n(x)$ is the linear projection of $\xi(x)$ onto $\mathrm{span}\{\xi(x_1), \ldots, \xi(x_n)\}$ orthogonally to $\mathcal{P}$



Prediction of a Gaussian process with unknown mean function

- The weights $\lambda_i(x; \underline{x}_n)$ are again solutions of a system of linear equations, which can be written under a matrix form as

$$\begin{pmatrix} k(\underline{x}_n, \underline{x}_n) & \varphi(\underline{x}_n)^{\mathsf{T}} \\ \varphi(\underline{x}_n) & 0 \end{pmatrix} \begin{pmatrix} \lambda(x; \underline{x}_n) \\ \mu(x) \end{pmatrix} = \begin{pmatrix} k(x, \underline{x}_n) \\ \varphi(x) \end{pmatrix}, \tag{9}$$

  with

  - $\varphi(\underline{x}_n)$ an $l \times n$ matrix with entries $\varphi_i(x_j)$, $i = 1, \ldots, l$, $j = 1, \ldots, n$
  - μ a vector of Lagrange coefficients
  - $k(\underline{x}_n, \underline{x}_n)$, $\lambda(x; \underline{x}_n)$, $k(x, \underline{x}_n)$ as above

Prediction of a Gaussian process with unknown mean function

- When the mean is unknown, the kriging covariance function is given by

$$\begin{aligned} k(x,y;\underline{x}_n) &:= \operatorname{cov}\left(\xi(x)-\widehat{\xi}_n(x), \xi(y)-\widehat{\xi}_n(y)\right) \\ &= k(x-y) - \lambda(x;\underline{x}_n)^{\mathsf{T}}\, k(y;\underline{x}_n) - \mu(x)^{\mathsf{T}}\varphi(y)\,. \end{aligned}$$

Proposition

Let k be a covariance function and assume $m \in \mathcal{P}$.

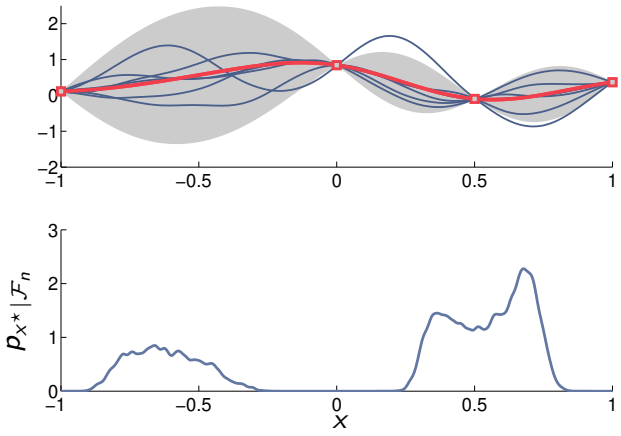
$$\textit{If } \begin{cases} \xi \mid m \sim \mathrm{GP}(m,\, k) \\ m : x \mapsto \beta^{\mathsf{T}}\varphi(x),\ \beta \sim \mathrm{U}(\mathbb{R}^l) \end{cases} \quad \textit{then } \xi \mid \mathcal{F}_n \sim \mathrm{GP}\left(\widehat{\xi}_n(\,\cdot\,),\, k(\,\cdot\,,\,\cdot\,;\underline{x}_n)\right)$$

with $\mathrm{U}(\mathbb{R}^l)$ the (improper) uniform distribution over $\mathbb{R}^l$

→ justifies the use of kriging in a Bayesian framework provided that the covariance function of ξ is known

Summing up

- The framework of Gaussian random processes makes it possible to compute an interpolation of evaluation results, and derive confidence intervals about the interpolation
- The global behavior of f is captured by interpolation
- The regions that may contain a global minimizer can be identified



➜ Moreover, restricting ξ to be a Gaussian process makes it possible to compute the expected improvement with moderate computational effort

3.8 Optimization with the expected improvement criterion

Expected Improvement [Mockus et al. 78, Schonlau et al. 96, Jones et al. 98]

- A well-known Bayesian optimization algorithm
  - proposed by Mockus et al.
  - popularized by the EGO algorithm of Jones et al.
- Idea : explore areas that are likely to contain a global optimizer
- Recall that the expected improvement criterion has been obtained by considering a one-step lookahead strategy for the problem of optimization: each new evaluation point is chosen according to

$$\begin{aligned} X_{n+1} &= \underset{x\in\mathbb{X}}{\operatorname{argmin}}\, \mathrm{E}_n\left(\widehat{m}_{n+1} - m \mid X_{n+1} = x\right) \\ &= \underset{x\in\mathbb{X}}{\operatorname{argmin}}\, \mathrm{E}_n\left(\widehat{m}_{n+1} \mid X_{n+1} = x\right) \\ &= \underset{x\in\mathbb{X}}{\operatorname{argmin}}\, \mathrm{E}_n\left(\widehat{m}_n \wedge \xi(X_{n+1}) \mid X_{n+1} = x\right) \\ &= \underset{x\in\mathbb{X}}{\operatorname{argmax}}\, \mathrm{E}_n\left(0 \wedge (\xi(X_{n+1}) - \widehat{m}_n) \mid X_{n+1} = x\right) \\ &= \underset{x\in\mathbb{X}}{\operatorname{argmax}}\, \rho_n(x) := \mathrm{E}_n\left((\widehat{m}_n - \xi(X_{n+1}))_+ \mid X_{n+1} = x\right) \end{aligned}$$

  with
  - $\widehat{m}_n = \xi(X_1) \wedge \cdots \wedge \xi(X_n)$,
  - $z_+ = \max(z, 0)$
- The sampling criterion ρ_n is the expected improvement (EI)
  $\rightarrow$ average excursion of $\xi(x)$ below the current minimum of past evaluation results

Expected Improvement [Mockus 78, Schonlau et al. 96, Jones et al. 98]

- Assume ξ is a Gaussian process, with known mean and covariance functions
- Then, $\rho_n(x)$ has a closed-form expression:

$$\rho_n(x) \;=\; \gamma\left(\, m_n - \widehat{\xi}_n(x;\underline{X}_n),\, \sigma_n^2(x)\,\right),$$

with

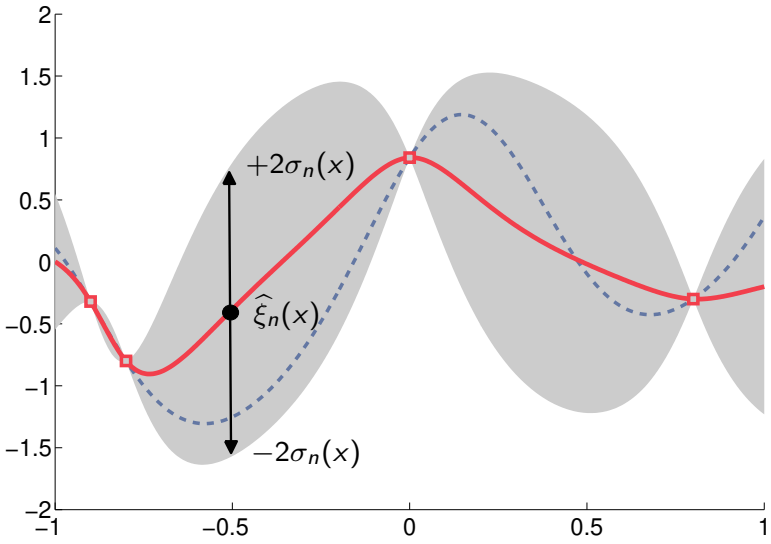
$$\gamma(z,s) \;=\; \begin{cases} \sqrt{s}\,\Phi'\left(\frac{z}{\sqrt{s}}\right) + z\,\Phi\left(\frac{z}{\sqrt{s}}\right) & \text{if } s>0, \\ \max(z,0) & \text{if } s=0. \end{cases}$$

- The EI algorithm:

$$\begin{cases} X_1 &=& x_{\text{init}}\,, \\ X_{n+1} &=& \underset{x\in\mathbb{X}}{\text{argmax}}\ \rho_n(x)\,, \quad n\geq 1\,, \end{cases}$$

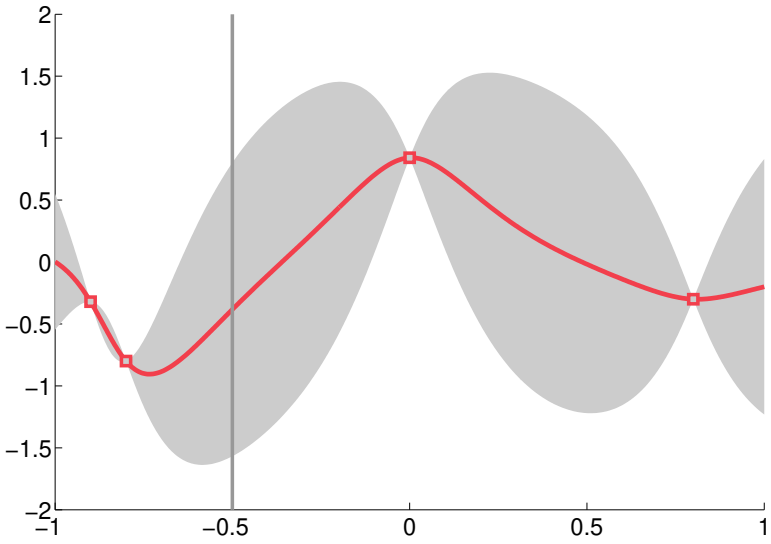
Expected Improvement [Mockus 78, Schonlau et al. 96, Jones et al. 98]

- High $\sigma_n(x)$ → unexplored region
- High $m_n - \widehat{\xi}_n(x)$ → promising region
- Compute the Expected Improvement ρ_n
- Next sample: $X_{n+1} = \arg\max_{x\in\mathbb{X}} \rho_n(x)$



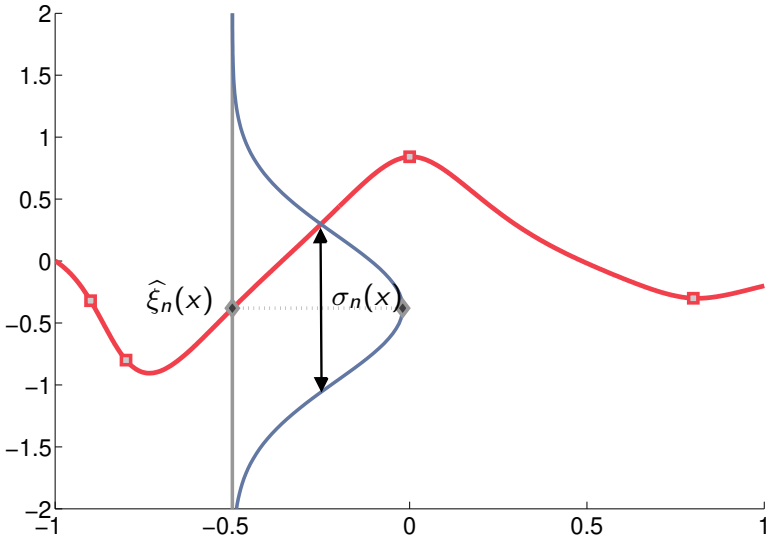
Expected Improvement [Mockus 78, Schonlau et al. 96, Jones et al. 98]

- High $\sigma_n(x) \rightarrow$ unexplored region
- High $m_n - \widehat{\xi}_n(x) \rightarrow$ promising region
- Compute the Expected Improvement ρ_n
- Next sample: $X_{n+1} = \arg\max_{x \in \mathbb{X}} \rho_n(x)$



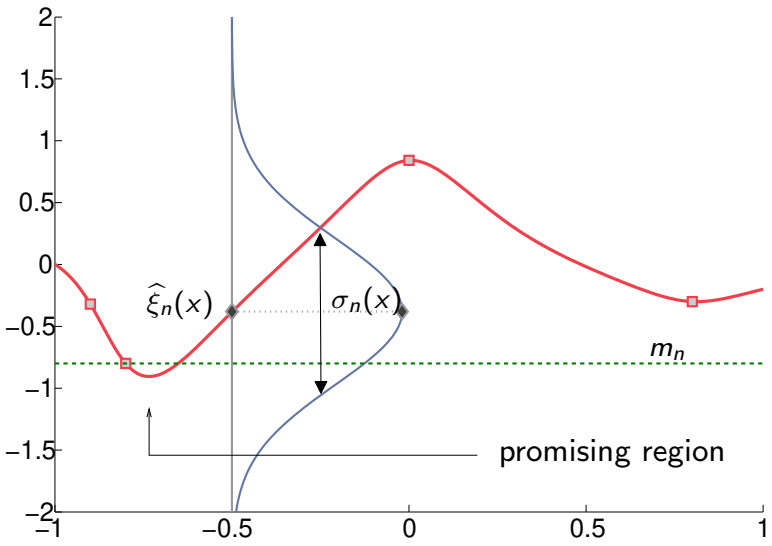
Expected Improvement [Mockus 78, Schonlau et al. 96, Jones et al. 98]

- High $\sigma_n(x) \rightarrow$ unexplored region
- High $m_n - \widehat{\xi}_n(x) \rightarrow$ promising region
- Compute the Expected Improvement ρ_n
- Next sample: $X_{n+1} = \arg\max_{x \in \mathbb{X}} \rho_n(x)$



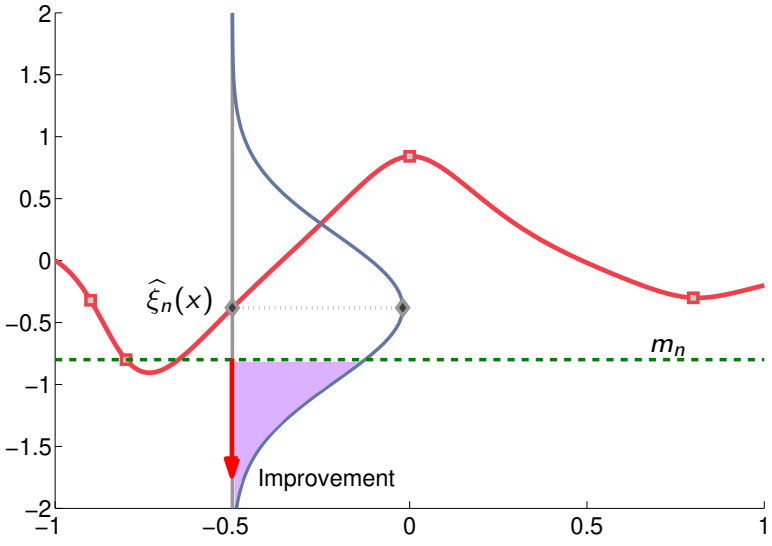
Expected Improvement [Mockus 78, Schonlau et al. 96, Jones et al. 98]

- High $\sigma_n(x) \rightarrow$ unexplored region
- High $m_n - \widehat{\xi}_n(x) \rightarrow$ promising region
- Compute the Expected Improvement ρ_n
- Next sample: $X_{n+1} = \arg\max_{x \in \mathbb{X}} \rho_n(x)$



Expected Improvement [Mockus 78, Schonlau et al. 96, Jones et al. 98]

- High $\sigma_n(x) \rightarrow$ unexplored region
- High $m_n - \widehat{\xi}_n(x) \rightarrow$ promising region
- Compute the Expected Improvement ρ_n
- Next sample: $X_{n+1} = \arg\max_{x \in \mathbb{X}} \rho_n(x)$

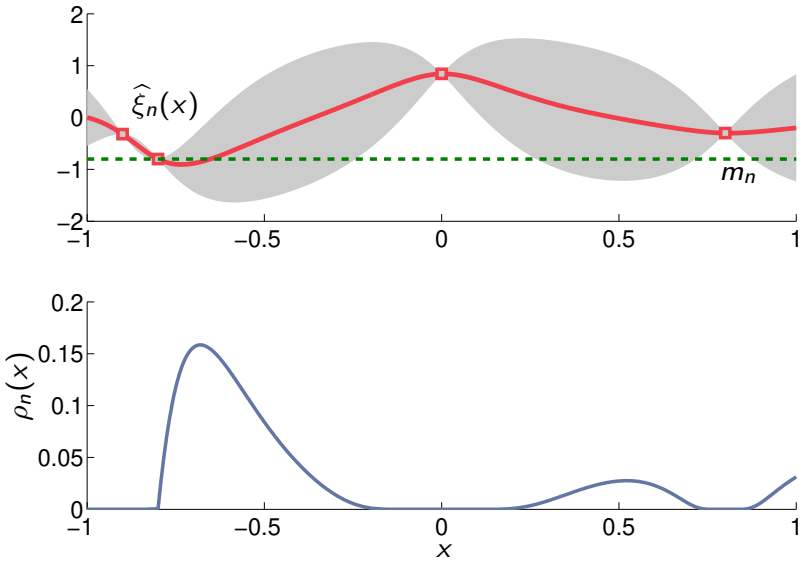


$$\rho_n(x) = \mathrm{E}[\ \max(m_n - \xi(x), 0) \ \mid\ \xi(X_1), \ldots, \xi(X_n)\ ]$$

(cheap to evaluate)

Expected Improvement [Mockus 78, Schonlau et al. 96, Jones et al. 98]

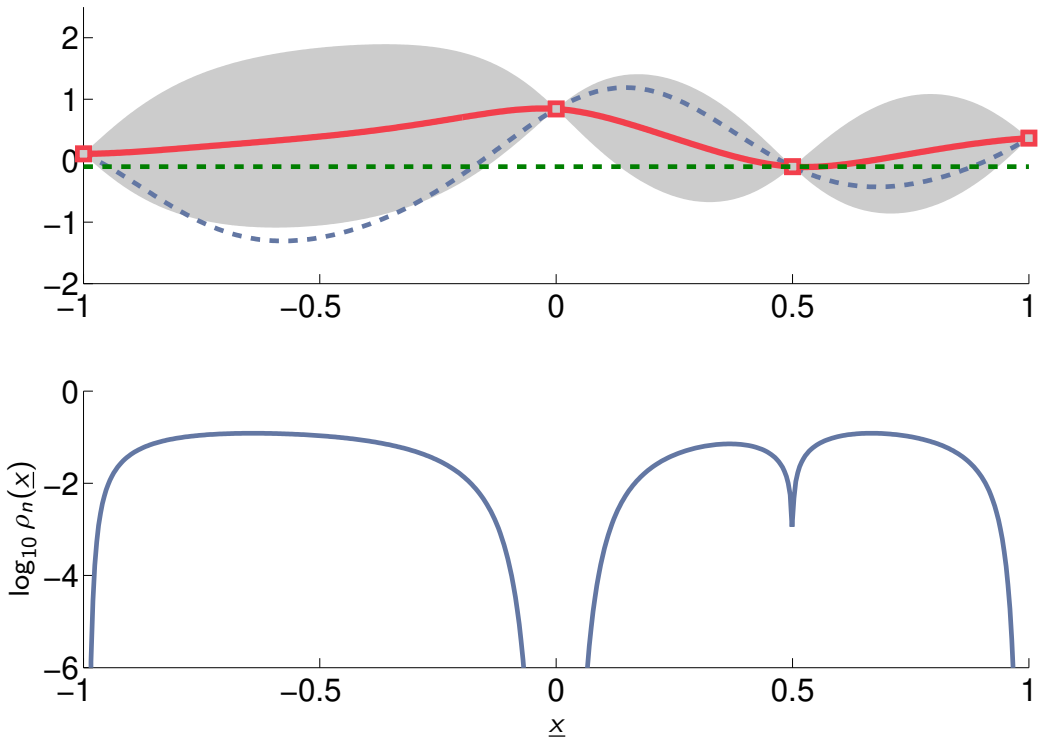
- High $\sigma_n(x) \rightarrow$ unexplored region
- High $m_n - \widehat{\xi}_n(x) \rightarrow$ promising region
- Compute the Expected Improvement ρ_n
- Next sample: $X_{n+1} = \arg\max_{x \in \mathbb{X}} \rho_n(x)$



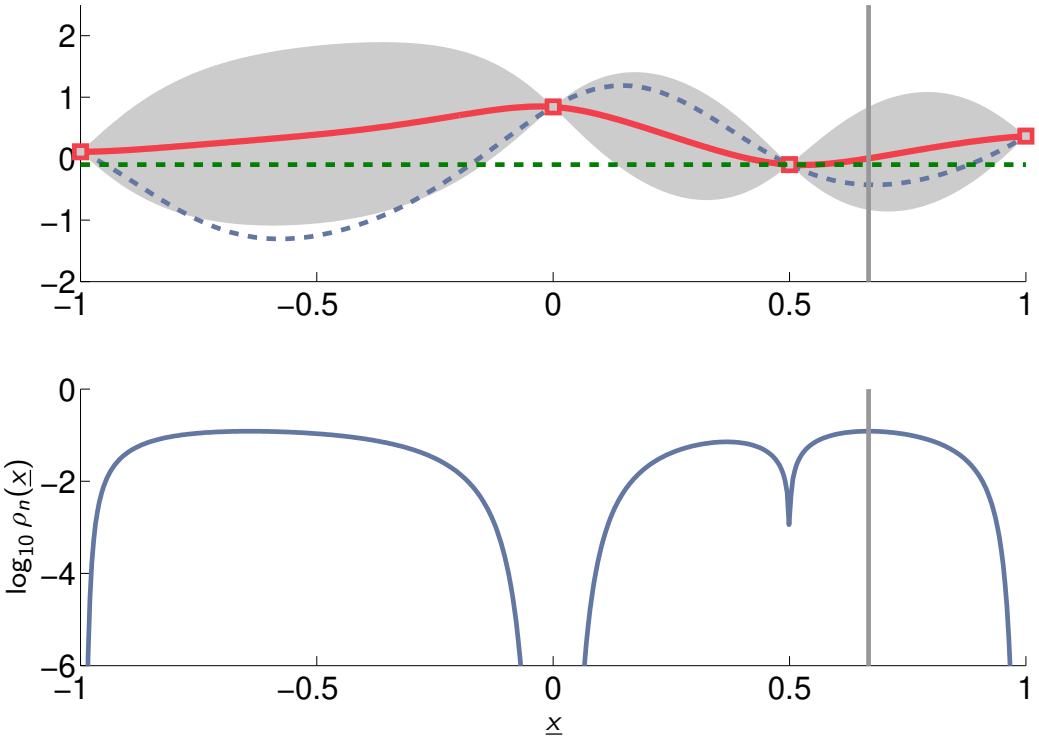
$$\rho_n(x) = \mathrm{E}[\ \max(m_n - \xi(x), 0)\ \mid\ \xi(X_1), \ldots, \xi(X_n)\ ]$$

(cheap to evaluate)

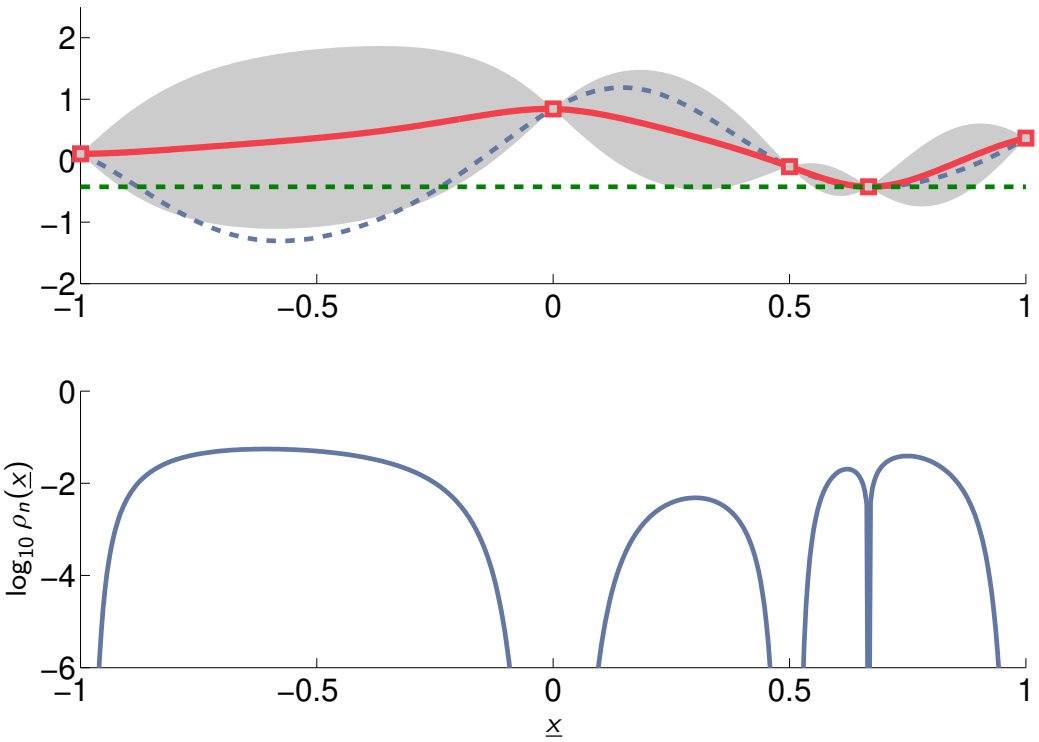
Global optimization based on EI



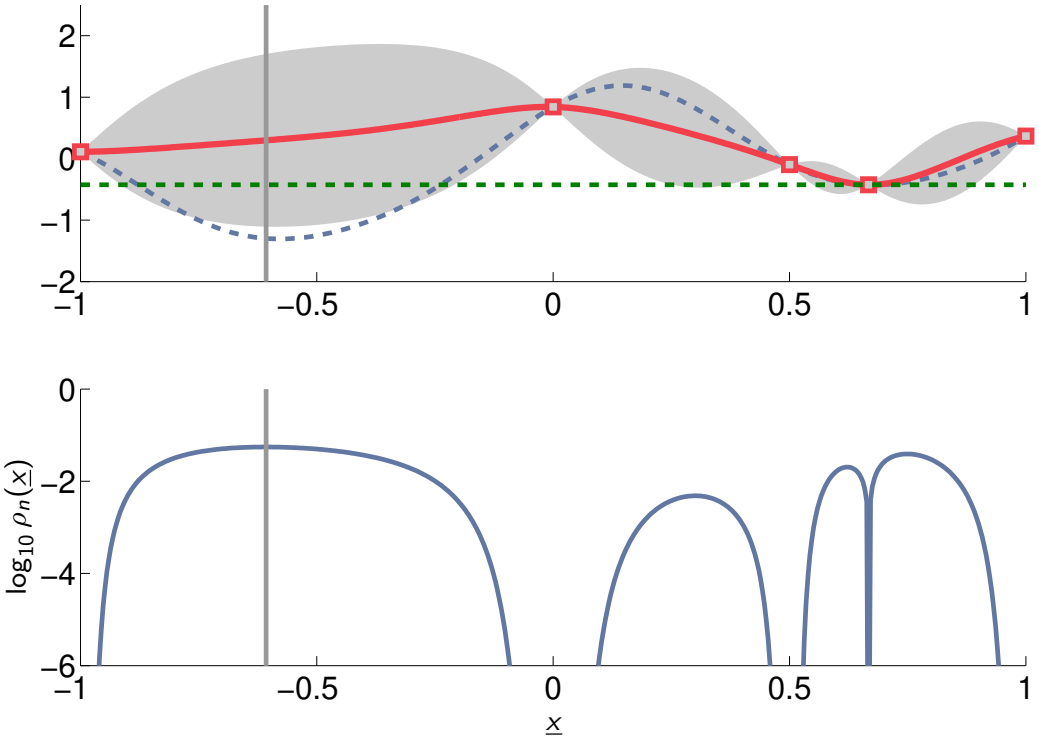
Global optimization based on EI



Global optimization based on EI

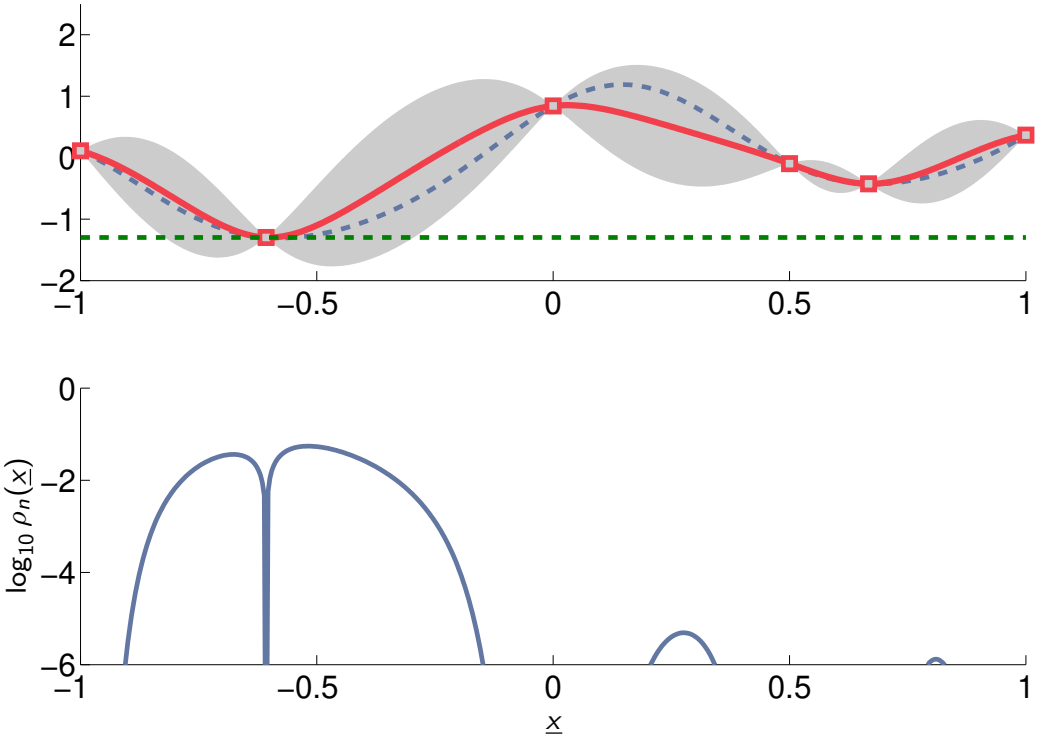


Global optimization based on EI

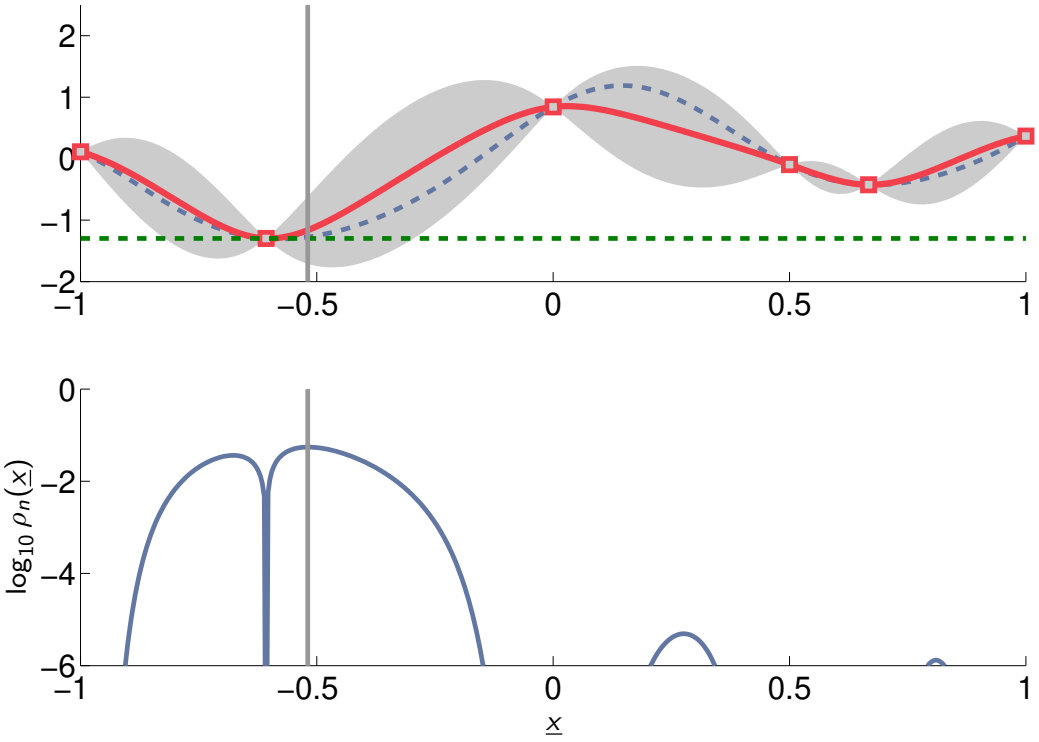


Optimization of an expensive-to-evaluate function Expected Improvement

Global optimization based on EI

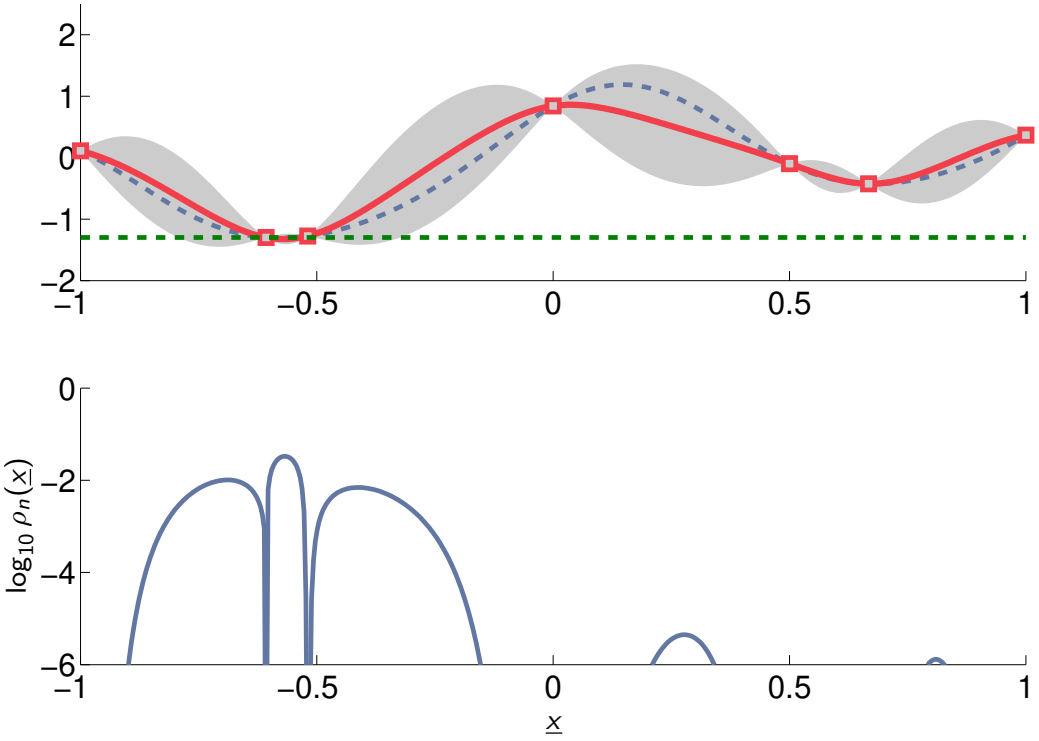


Global optimization based on EI

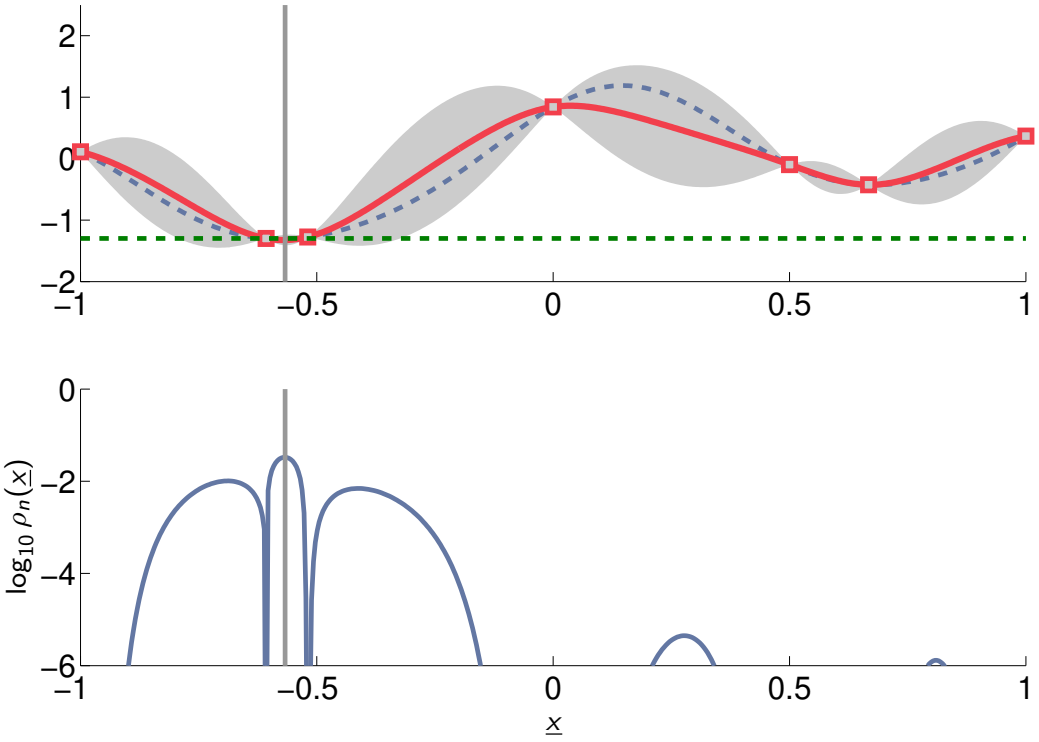


Optimization of an expensive-to-evaluate function Expected Improvement

Global optimization based on EI

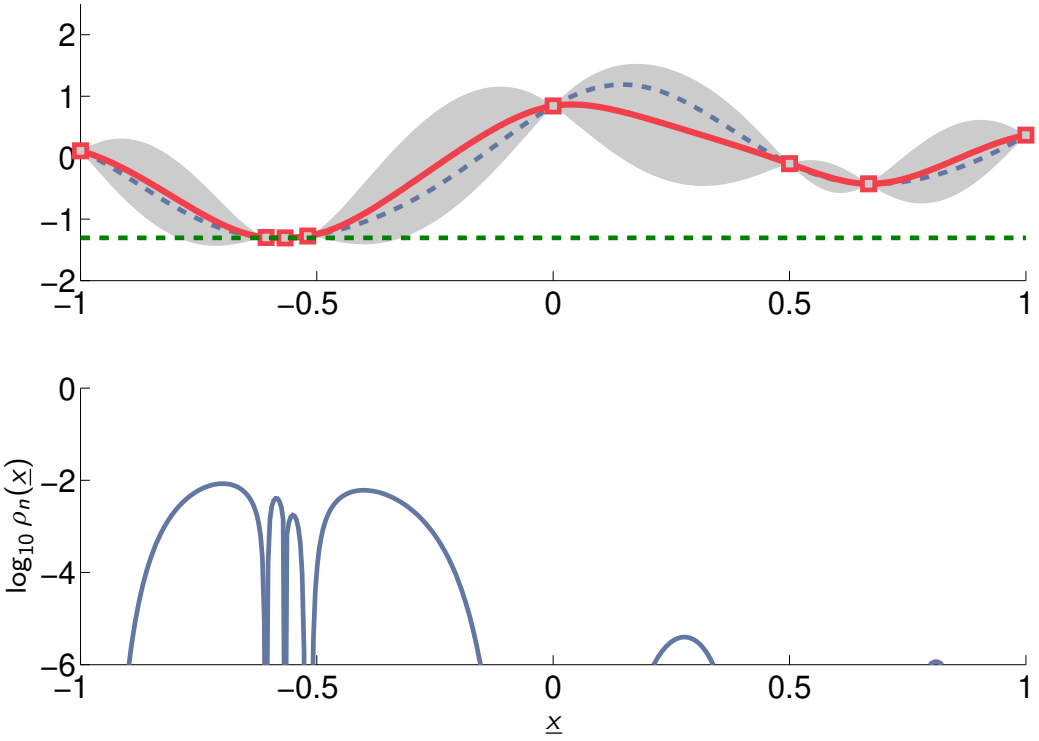


Global optimization based on EI

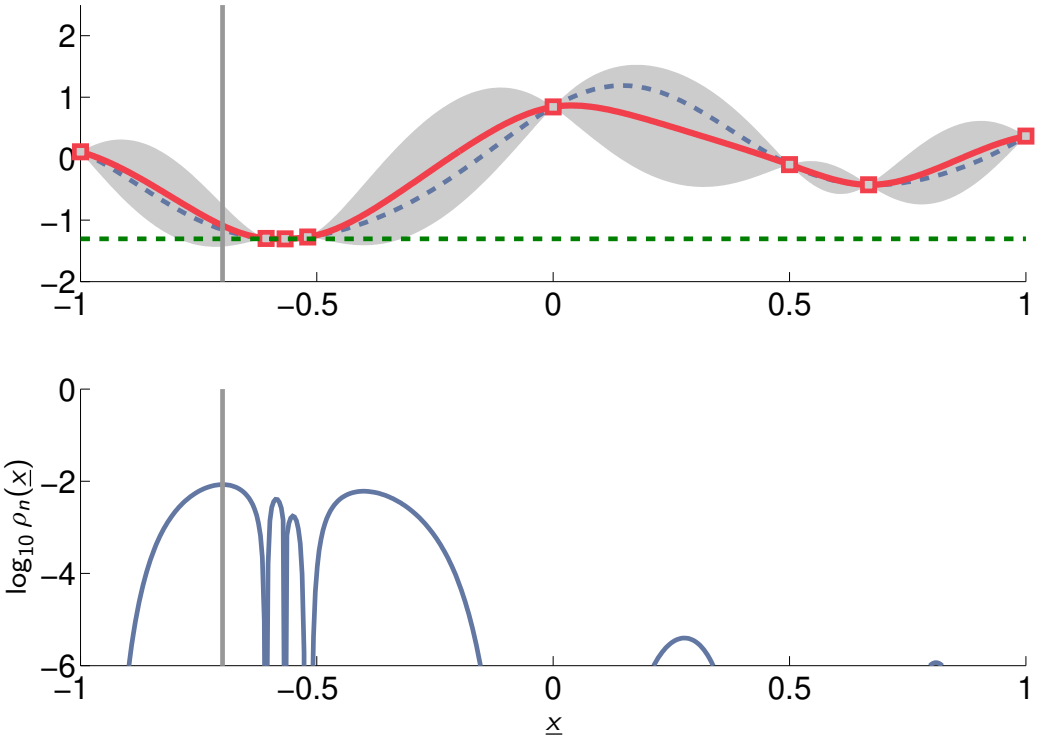


Optimization of an expensive-to-evaluate function Expected Improvement

Global optimization based on EI

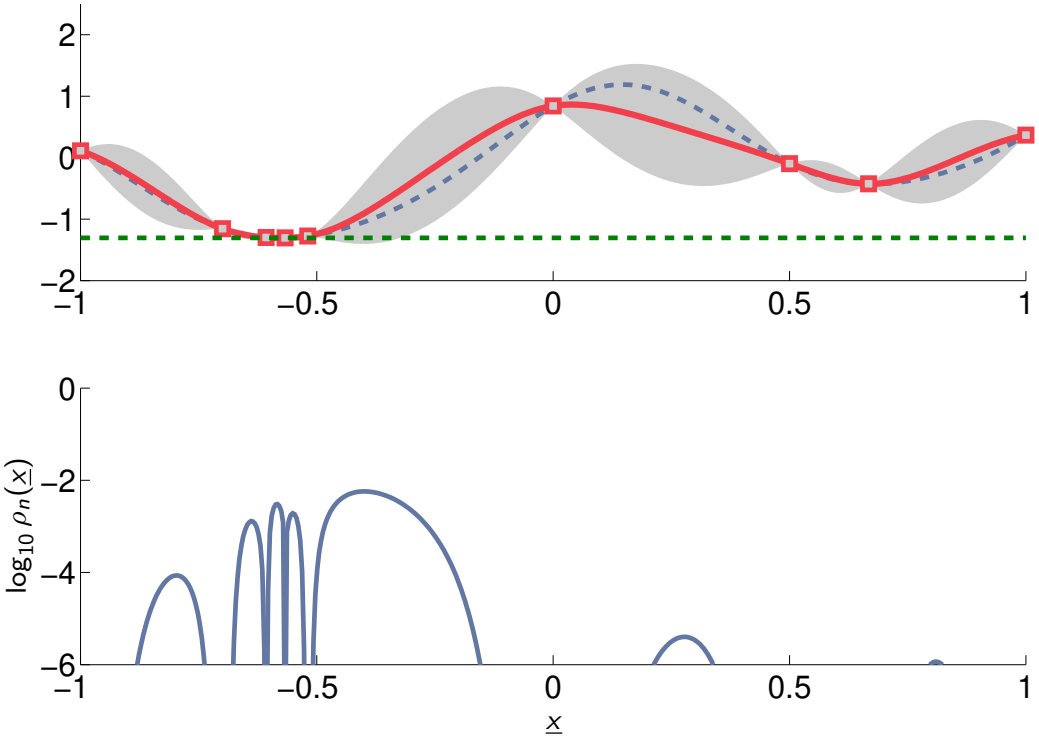


Global optimization based on EI

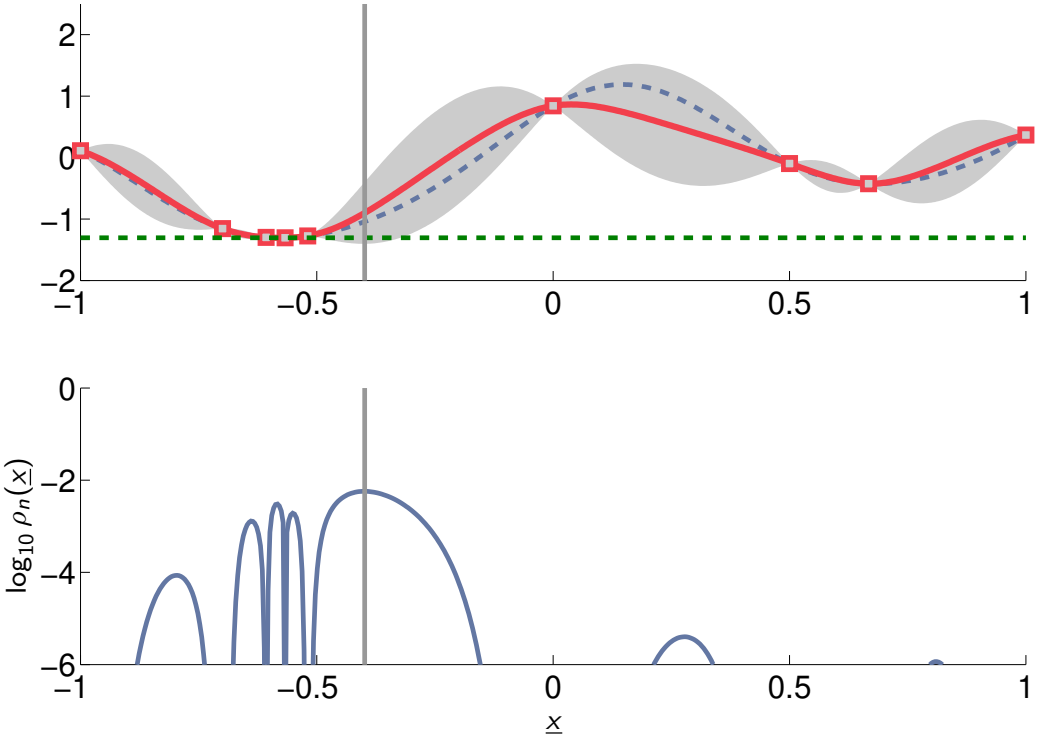


Optimization of an expensive-to-evaluate function Expected Improvement

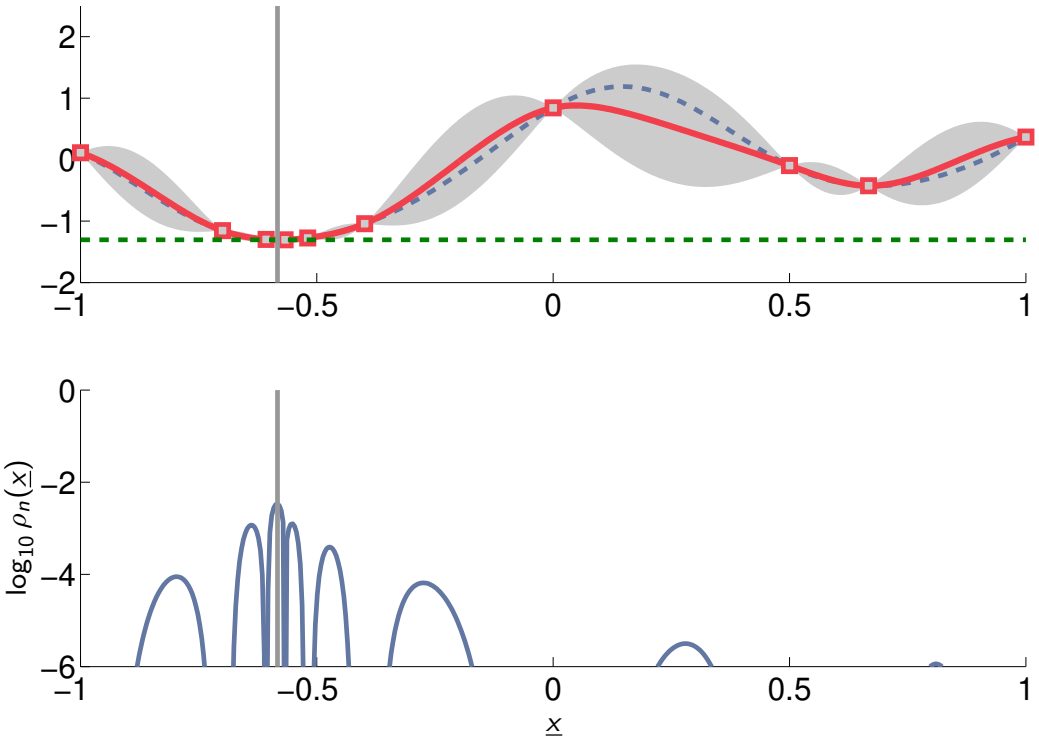
Global optimization based on EI



Global optimization based on EI



Global optimization based on EI

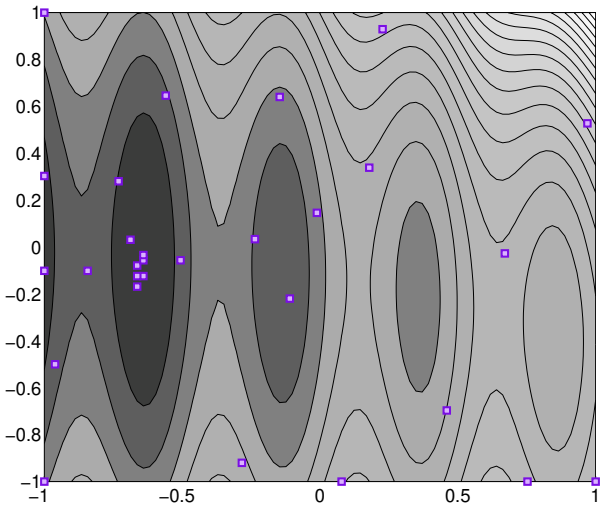


From EI to EGO

- In practice, it is often difficult to choose a covariance function for ξ before any evaluation is made
- As a result, the covariance function of is usually assumed to belong to some parametric class of positive definite functions, the value of the parameters assumed to be unknown
- In the Efficient Global Optimization (EGO) algorithm, the parameters are estimated from the evaluation results by maximum likelihood, and then plugged in the EI sampling criterion (computed for a Gaussian process with known covariance function)

EI/EGO: 2D illustration

(f defined on Slide 19)



	$\widehat{m}_n$ with $N = 60$
LHS	-5.823
DIRECT	-5.839
EI/EGO	-5.845
Global minimum	-5.845

NB: Global minimum found by the EI algorithm in only 31 evaluations (abs. tol. 1.10^{-4})

Global optimization based on EI/EGO: implementation issues

How to find the maximizer of ρ_n at each iteration?

- simple approach: use a finite grid on $\mathbb{X}$ (a set of candidate points)
- refine the grid in regions with a high ρ_n
- in practice, a high precision on the location of the maximizer of ρ_n is not required

How to choose the prior, i.e. the mean and the covariance functions of ξ?

- Usually: consider a parametrized covariance function (e.g. the exponential or the Matérn covariance function) and estimate the parameters by maximum likelihood
- Not necessarily a good idea to estimate the parameters at each iteration (use instead an initial design to estimate the parameters)
- NB: In principle, the uncertainty due to parameter estimation should be taken into account. Very often in practice, a plug-in approach is used (EGO). However, in this case, the variance of the error of prediction is underestimated.

Global optimization based on EI

Pros

- Efficient global optimization procedure (often better than DIRECT in experiments)
- Global search and local search
- Known convergence results

Cons

- Working principle rather involved (?)
- User friendly software yet to come
- Not (yet) in Matlab
- The role of the tuning parameters needs to be understood by the user

Summing up

- In the context of risk analysis, it is often desirable to assess the worst-case performance of a system $\rightarrow$ this is a global optimization problem
- Some working principles of Lipschitzian and Bayesian sequential search algorithms exposed
- Particularly interesting in the context of expensive-to-evaluate functions, very useful and effective in practical situations
- Many applications can be found in the literature
- A great number of methodological and theoretical questions are open and it is an active research domain at present time

4. Estimation of a probability of failure in a Bayesian sequential decision framework

4.1 Statement of the problem

Reminder

- Our objective: to obtain an approximation of

$$\alpha^u(f) = \mathrm{P}_{\mathbb{X}}\{f > u\} = \int_{\mathbb{X}} \mathbb{1}_{f>u} \mathrm{dP}_{\mathbb{X}}$$

- The approximation of $\alpha^u(f)$ has to be built from a set of computer experiments
- Expensive computer experiments: the number of evaluations is limited
- We want to construct an algorithm to estimate a probability of failure, that is a pair $(\underline{X}_N, \widehat{\alpha}_N)$,

$$\begin{aligned} \underline{X}_N &: f \mapsto \underline{X}_N(f) = (X_1(f), X_2(f), \ldots, X_N(f)) \in \mathbb{X}^N, \\ \widehat{\alpha}_N &: f \mapsto \widehat{\alpha}_N(f) \in \mathbb{R}_+, \end{aligned}$$

- $\underline{X}_N$ is called a strategy
- $\widehat{\alpha}_N(f)$ is an estimator of $\alpha^u(f)$
- How to construct a good algorithm?

Estimation of a probability of failure

- Assume that an estimator $\widehat{\alpha}_N$ has been chosen (see how later)
- How to construct the strategy $\underline{X}_N$?
- Let $\mathcal{A}_N$ be the class of all strategies $\underline{X}_N$ that query sequentially N evaluations of f
- Given a loss function

$$L : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$$

define the error of approximation of a strategy $\underline{X}_N \in \mathcal{A}_N$ on f as

$$\epsilon(\underline{X}_N, f) = L(\widehat{\alpha}_N(f), \alpha(f))$$

- Here, we shall consider the quadratic loss function, so that

$$\epsilon(\underline{X}_N, f) = (\widehat{\alpha}_N(f) - \alpha(f))^2$$

(Depending on the problem, there may exist better choices. For instance, if it is more harmful to underestimate a probability of failure than to overestimate it, then the loss function should be chosen accordingly.)

Estimation of a probability of failure

- We adopt a Bayesian approach: the unknown function f is considered as a sample path of a real-valued random process ξ defined on some probability space $(\Omega, \mathcal{B}, \mathrm{P}_0)$ with parameter $x \in \mathbb{X}$
- A good strategy is a strategy that achieves, or gets close to, the Bayes or average risk

$$r_{\text{average}} := \inf_{\underline{X}_N \in \mathcal{A}_N} \mathrm{E}_0 \left(\epsilon(\underline{X}_N, \xi)\right)$$

where E_0 denotes the expectation with respect to P_0

- From a subjective Bayesian point of view, the stochastic model ξ is a representation of our uncertain initial knowledge about f
- From a pragmatic perspective, the prior distribution can be seen as a tool to define a notion of a good strategy in an average sense.

4.2 Optimal and k-step lookahead strategies

Optimal Bayesian strategies

- Let E_n, $n = 1, 2, \ldots$, denote the conditional expectation with respect to $\mathcal{I}_n(\xi)$, where
  - $\mathcal{I}_n = ((X_1, Z_1), \ldots, (X_n, Z_n))$
  - $Z_n(\xi) = \xi(X_n(\xi))$, $1 \leq n \leq N$
- As above, an optimal strategy, i.e. a strategy $\underline{X}_N^\star \in \mathcal{A}_N$ such that

$$\mathrm{E}_0\left(\epsilon(\underline{X}_N^\star, \xi)\right) = r_{\mathrm{average}} = \inf_{\underline{X}_N \in \mathcal{A}_N} \mathrm{E}_0\left(\epsilon(\underline{X}_N, \xi)\right)$$

  can be formally obtained by dynamic programming
- Let $R_N = \mathrm{E}_N\left(\epsilon(\underline{X}_N, \xi)\right) = \mathrm{E}_N\left((\widehat{\alpha}_N - \alpha)^2\right)$ denote the terminal risk
- Define by backward induction

$$R_n = \min_{x \in \mathbb{X}} \mathrm{E}_n\left(R_{n+1} \mid X_{n+1} = x\right), \quad n = N-1, \ldots, 0. \tag{10}$$

Optimal Bayesian strategies

- Then, we have $R_0 = r_{\text{average}}$
- The optimal strategy $\underline{X}_N^\star$ is formally obtained as

$$X_{n+1}^\star = \underset{x \in \mathbb{X}}{\operatorname{argmin}} \, \mathrm{E}_n\left(R_{n+1} \mid X_{n+1} = x\right), \quad n = 1, \ldots, N-1, \tag{11}$$

- Unfortunately, as in the case of optimization, this dynamic programming is not numerically tractable

  (the space of possible actions $\mathbb{X}$ at each step is continuous, the state space $(\mathbb{X} \times \mathbb{R})^n$ at step n is also continuous and of dimension $n(d+1)$)

Optimal Bayesian strategies

- As in the case of optimization, the optimal strategy can be expanded as

$$X_{n+1}^\star = \underset{x \in \mathbb{X}}{\operatorname{argmin}} \, \mathrm{E}_n\left(\min_{X_{n+2}} \mathrm{E}_{n+1} \ldots \min_{X_N} \mathrm{E}_{N-1} \, R_N \;\middle|\; X_{n+1} = x\right).$$

- A k-step lookahead strategy is obtained when truncating the expansion after k terms and replacing the exact risk R_{n+k} by a surrogate $\widetilde{R}_{n+k}$

$$X_{n+1} = \underset{x \in \mathbb{X}}{\operatorname{argmin}} \, J_n(x) := \mathrm{E}_n\left(\min_{X_{n+2}} \mathrm{E}_{n+1} \ldots \min_{X_{n+k}} \mathrm{E}_{n+k-1} \, \widetilde{R}_{n+k} \;\middle|\; X_{n+1} = x\right). \tag{12}$$

- We restrict our attention to the class of one-step lookahead strategies: in this case, the sampling criterion $J_n(x)$ may be written as

$$J_n(x) = \mathrm{E}_n\left(\widetilde{R}_{n+1} \;\middle|\; X_{n+1} = x\right)$$

- How to define a surrogate risk $\widetilde{R}_{n+1}$ for the problem of the estimation of a probability of failure?

One-step lookahead strategy for the problem of estimation of a probability of failure

- A natural and straightforward way of building a one-step lookahead strategy is to select greedily each evaluation as if it were the last one
- When the Bayesian risk provides a measure of the estimation error, we call such a strategy a stepwise uncertainty reduction (SUR) strategy
- Given a sequence of estimators $(\widehat{\alpha}_n)_{n\geq 1}$, a direct application of the above principle using the quadratic loss function yields the sampling criterion

$$J_n(x) = \mathsf{E}_n\left((\alpha - \widehat{\alpha}_{n+1})^2 \mid X_{n+1} = x\right) .$$

- Restricting ξ to be a Gaussian process makes it possible to derive estimators for α and to compute J_n with moderate computational efforts

4.3 Estimators of the probability of failure under a Gaussian prior

Estimators of the probability of failure under a Gaussian prior

- Given a random process ξ and a strategy $\underline{X}_N$, the optimal estimator that minimizes $\mathrm{E}_0\left((\alpha - \widehat{\alpha}_n)^2\right)$ among all $\mathcal{F}_n$-measurable estimators $\widehat{\alpha}_n$, $1 \leq n \leq N$, is

$$\widehat{\alpha}_n \;=\; \mathrm{E}_n\left(\alpha\right) \;=\; \mathrm{E}_n\left(\int_{\mathbb{X}} \mathbb{1}_{\xi>u}\,\mathrm{dP}_{\mathbb{X}}\right) \;=\; \int_{\mathbb{X}} p_n\,\mathrm{dP}_{\mathbb{X}}\,, \tag{13}$$

  where

$$p_n : x \in \mathbb{X} \mapsto \mathrm{P}_n\left\{\xi(x) > u\right\}\,.$$

- When ξ is a Gaussian process, the probability $p_n(x)$ of exceeding u at $x \in \mathbb{X}$, given $\mathcal{I}_n$, has a simple closed-form expression:

$$p_n(x) \;=\; 1 - \Phi\left(\frac{u - \widehat{\xi}_n(x)}{\sigma_n(x)}\right) \;=\; \Phi\left(\frac{\widehat{\xi}_n(x) - u}{\sigma_n(x)}\right)\,,$$

  with Φ the cdf of the normal distribution

- Thus, in the Gaussian case, the estimator (13) is amenable to a numerical approximation, by integrating the excess probability p_n over $\mathbb{X}$

Estimators of the probability of failure under a Gaussian prior

- Another natural way to obtain an estimator of α given $\mathcal{I}_n$ is to approximate the excess indicator $\mathbb{1}_{\xi>u}$ by a hard classifier $\eta_n : \mathbb{X} \to \{0, 1\}$

  ("hard" refers to the fact that η_n takes its values in $\{0, 1\}$)

- If η_n is close (in some sense) to $\mathbb{1}_{\xi>u}$, the estimator

$$\widehat{\alpha}_n = \int_{\mathbb{X}} \eta_n \mathrm{dP}_{\mathbb{X}}$$

  should be close to $\alpha = \int \mathbb{1}_{\xi>u}\mathrm{dP}_{\mathbb{X}}$

Estimators of the probability of failure under a Gaussian prior

- More precisely,

$$\mathsf{E}_n\left((\widehat{\alpha}_n-\alpha)^2\right)=\mathsf{E}_n\left[\left(\int(\eta_n-\mathbb{1}_{\xi>u})\mathrm{dP}_{\mathbb{X}}\right)^2\right]\leq\int\mathsf{E}_n\left((\eta_n-\mathbb{1}_{\xi>u})^2\right)\mathrm{dP}_{\mathbb{X}} \quad (14)$$

- Let

$$\tau_n(x)=\mathsf{P}_n\{\eta_n(x)\neq\mathbb{1}_{\xi(x)>u}\}=\mathsf{E}_n\left((\eta_n(x)-\mathbb{1}_{\xi(x)>u})^2\right)$$

  be the probability of misclassification; that is, the probability to predict a point above (resp. under) the threshold, when the true value is under (resp. above) the threshold
- Thus, (14) shows that it is desirable to use a classifier η_n such that τ_n is small for all $x\in\mathbb{X}$

Estimators of the probability of failure under a Gaussian prior

- The right-hand side of (14) is minimized if we set

$$\eta_n(x)=\mathbb{1}_{p_n(x)>1/2}=\mathbb{1}_{\bar{\xi}_n(x)>u}\,,$$

  where $\bar{\xi}_n(x)$ denotes the posterior median of $\xi(x)$.
- Then, we have

$$\begin{aligned}\tau_n(x) &= p_n(x)+(1-2p_n(x))\,\eta_n(x)\\ &= \min(p_n(x),1-p_n(x))\end{aligned}$$

Estimators of the probability of failure under a Gaussian prior

- In the case of a Gaussian process, the posterior median and the posterior mean are equal
- Then, the classifier that minimizes $\tau_n(x)$ for each $x \in \mathbb{X}$ is $\eta_n = \mathbb{1}_{\widehat{\xi}_n > u}$, in which case

$$\tau_n(x) \;=\; \mathrm{P}_n\left((\xi(x)-u)(\widehat{\xi}_n(x)-u)<0\right) \;=\; 1 \;-\; \Phi\left(\frac{|\widehat{\xi}_n(x)-u|}{\sigma_n(x)}\right). \qquad (15)$$

- For $\eta_n = \mathbb{1}_{\widehat{\xi}_n > u}$, we have

$$\widehat{\alpha}_n = \int_{\mathbb{X}} \mathbb{1}_{\widehat{\xi}_n > u} \mathrm{dP}_{\mathbb{X}} = \alpha(\widehat{\xi}_n)$$

Therefore, this approach to obtain an estimator of α can be seen as a type of plug-in estimation.

Estimators of the probability of failure under a Gaussian prior

- Summing up, the following two estimators of the probability of failure can be considered:

$$\widehat{\alpha}_n = \mathrm{E}_n(\alpha) = \int_{\mathbb{X}} p_n \,\mathrm{dP}_{\mathbb{X}} \qquad \Big| \qquad \widehat{\alpha}_n = \alpha(\widehat{\xi}_n) = \int_{\mathbb{X}} \mathbb{1}_{\widehat{\xi}_n > u} \mathrm{dP}_{\mathbb{X}}$$

4.4 Upper bounds of the SUR sampling criterion

Upper bounds of the SUR sampling criterion

- Recall that the sampling criterion of the one-step lookahead strategy using the quadratic loss function for the problem of estimation of a probability of failure is

$$J_n(x) = \mathsf{E}_n\left((\alpha - \widehat{\alpha}_{n+1})^2 \mid X_{n+1} = x\right) .$$

- Unfortunately, J_n has no analytical expression (setting either $\widehat{\alpha}_n = \mathsf{E}_n(\alpha)$ or $\widehat{\alpha}_n = \alpha(\widehat{\xi}_n)$)
- We seek to replace the minimization of J_n by the minimization of an upper bound of J_n

Upper bounds of the SUR sampling criterion

- Recall that $\tau_n(x) = \min(p_n(x), 1 - p_n(x))$ is the probability of misclassification at x using the classifier $\mathbb{1}_{\widehat{\xi}_n(x)>u}$
- Let us further denote by $\nu_n(x) := p_n(x)\ (1 - p_n(x))$ the variance of the excess indicator $\mathbb{1}_{\xi(x)\geq u}$.

Proposition

Assume that either $\widehat{\alpha}_n = \mathsf{E}_n(\alpha)$ *or* $\widehat{\alpha}_n = \alpha(\widehat{\xi}_n)$.
Define $G_n := \int_{\mathbb{X}} \sqrt{\gamma_n(y)}\mathrm{d}\mathrm{P}_{\mathbb{X}}(y)$ *for all* $n \in \{0, \ldots, N-1\}$, *with*

$$\gamma_n := \begin{cases} \nu_n = p_n(1-p_n) = \tau_n(1-\tau_n), & \text{if } \widehat{\alpha}_n = \mathsf{E}_n(\alpha), \\ \tau_n = \min(p_n, 1-p_n), & \text{if } \widehat{\alpha}_n = \alpha(\widehat{\xi}_n). \end{cases}$$

Then, for all $x \in \mathbb{X}$ *and all* $n \in \{0, \ldots, N-1\}$,

$$J_n(x) \leq \widetilde{J}_n(x) := \mathsf{E}_n\left(G_{n+1}^2 \mid X_{n+1} = x\right).$$

Upper bounds of the SUR sampling criterion

- Note that $\gamma_n(x)$ is a function of $p_n(x)$ that vanishes at 0 and 1, and reaches its maximum at $1/2$; that is, when the uncertainty on $\mathbb{1}_{\xi(x)>u}$ is maximal

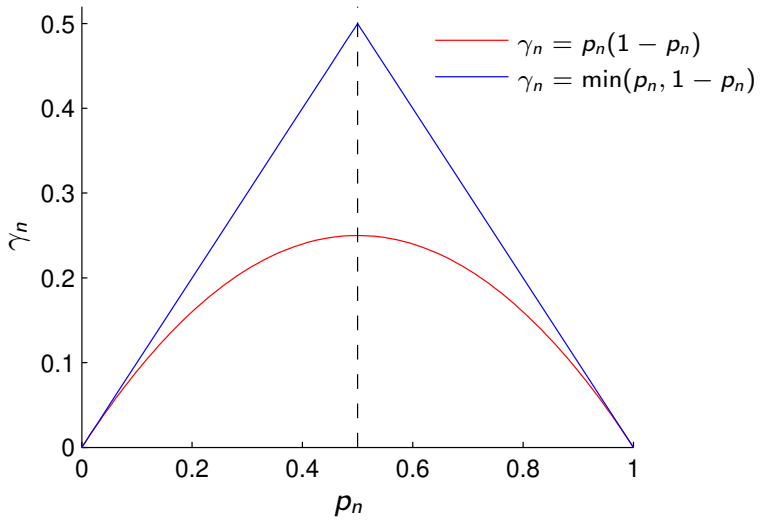


Figure: γ_n as a function of p_n. In both cases, γ_n is maximum at $p_n = 1/2$.

Upper bounds of the SUR sampling criterion

- As a result, we can write two SUR criteria:

1. Setting $\widehat{\alpha}_n = \alpha(\widehat{\xi}_n)$, we obtain

$$J_{1,n}^{\mathrm{SUR}}(x) = \mathrm{E}_n\left(\left(\int \sqrt{\tau_{n+1}}\,\mathrm{dP}_{\mathbb{X}}\right)^2 \,\middle|\, X_{n+1} = x\right)$$

2. Setting $\widehat{\alpha}_n = \mathrm{E}_n(\alpha)$, we obtain

$$J_{2,n}^{\mathrm{SUR}}(x) = \mathrm{E}_n\left(\left(\int \sqrt{\nu_{n+1}}\,\mathrm{dP}_{\mathbb{X}}\right)^2 \,\middle|\, X_{n+1} = x\right)$$

- Each criterion is expressed as the conditional expectation of some squared $\mathcal{F}_{n+1}$-measurable integral criterion, with an integrand that can be expressed as a function of the excess probability p_{n+1}.

4.5 Discretizations of the SUR criteria

Discretizations of the SUR criteria

- At this point, we need to provide numerical approximations of the integrals in the SUR criteria

Example for the criterion $J_{1,n}^{\mathrm{SUR}}$

- For each $y \in \mathbb{X}$, $\tau_{n+1}(y)$ is a function of $\mathcal{I}_{n+1} = (\mathcal{I}_n, X_{n+1}, Z_{n+1})$, with $Z_{n+1} = \xi(X_{n+1})$
- At step n, $\mathcal{I}_n$ is known
- Consider the notation
$$v_{n+1}(y; X_{n+1}, Z_{n+1}) = \sqrt{\tau_{n+1}(y)}$$
to emphasize the fact that, when a new evaluation point must be chosen at step n, $\tau_{n+1}(y)$ depends on the choice of X_{n+1} and the random outcome Z_{n+1}
- For $x \in \mathbb{X}$, let us further denote by $\mathrm{Q}_{n,x}$ the probability distribution of $\xi(x)$ under P_n
- Then,
$$J_{1,n}^{\mathrm{SUR}}(x) = \int_{\mathbb{R}} \left\{ \int_{\mathbb{X}} v_{n+1}(y; x, z) \, \mathrm{dP}_{\mathbb{X}}(y) \right\}^2 \mathrm{dQ}_{n,x}(z)$$
- Given $\mathcal{I}_n$ and a triple (x, y, z), $v_{n+1}(y; x, z)$ can be computed efficiently using kriging

Discretizations of the SUR criteria

- To obtain a numerical approximation of $J_{1,n}^{\mathrm{SUR}}$, we proceed in two steps:
  1. compute the integral on $\mathbb{X}$ with respect to $\mathrm{P}_{\mathbb{X}}$;
  2. compute the integral on $\mathbb{R}$ with respect to $\mathrm{Q}_{n,x}$

Discretizations of the SUR criteria

- To compute the integral on $\mathbb{X}$ with respect to $\mathrm{P}_{\mathbb{X}}$, we can use a MC approach
- Draw an i.i.d. sequence $Y_1, \ldots, Y_m \sim \mathrm{P}_{\mathbb{X}}$ and use the MC approximation:

$$\int_{\mathbb{X}} v_{n+1}(y; x, z)\, \mathrm{dP}_{\mathbb{X}}(y) \;\approx\; \frac{1}{m} \sum_{j=1}^{m} v_{n+1}(Y_j; x, z).$$

- Equivalently, it means that we choose to work from the start on a discretized version of the problem: we replace $\mathrm{P}_{\mathbb{X}}$ by the empirical distribution

$$\widehat{\mathrm{P}}_{\mathbb{X},m} = \frac{1}{m} \sum_{j=1}^{m} \delta_{Y_j}$$

and our goal is to estimate the MC estimator

$$\alpha_m(\xi) = \int \mathbb{1}_{\xi > u} \mathrm{d}\widehat{\mathrm{P}}_{\mathbb{X},m} = \frac{1}{m} \sum_{j=1}^{m} \mathbb{1}_{\xi(Y_j) > u}$$

using either the posterior mean

$$\mathrm{E}_n\left(\alpha_m\right) = \frac{1}{m} \sum_{j} p_n(Y_j)$$

or the plug-in estimate

$$\alpha_m(\widehat{\xi}_n) = \frac{1}{m} \sum_{j} \mathbb{1}_{\widehat{\xi}_n(Y_j) > u}$$

Discretizations of the SUR criteria

- We call this approach meta-estimation: the objective is to estimate the value of a precise Monte Carlo estimator of $\alpha(f)$ (m being large), using prior information on f to alleviate the computational burden of running m times the computer code f
- This point of view also suggests a natural solution for the problem of finding the minimum of $J_{1,n}^{\mathrm{SUR}}$ or $J_{2,n}^{\mathrm{SUR}}$, which is to replace the continuous search for a minimizer $x \in \mathbb{X}$ by a discrete search over the set $\mathbb{X}_m := \{Y_1, \ldots, Y_m\}$.

Discretizations of the SUR criteria

- The second problem is the computation of a one-dimensional integral
- $Q_{n,x}$ is a Gaussian probability distribution with mean $\widehat{\xi}_n(x)$ and variance $\sigma_n^2(x)$
- The integral can be computed using a standard Gauss-Hermite quadrature with Q points
- This is equivalent to replacing (under P_n) the random variable $\xi(x)$ by a quantized random variable with probability distribution

$$\sum_{q=1}^{Q} w_q \delta_{z_{n+1,q}(x)},$$

where w_q are weights of the quadrature and

$$z_{n+1,q}(x) = \widehat{\xi}_n(x) + \sigma_n(x) u_q,$$

where u_q denote quadrature points

- Eventually, the J_1^{SUR} strategy is:

$$X_{n+1} = \operatorname*{argmin}_{1 \le k \le m} \sum_{q=1}^{Q} w_q \left\{ \sum_{j=1}^{m} v_{n+1}\left(Y_j;\, Y_k, z_{n+1,q}(Y_k)\right) \right\}^2.$$

Sequential estimation of a probability of failure

Sketch of an algorithm

1. Construct an initial design of size $n_0 < N$ and evaluate f at the points of the initial design.
2. Choose a Gaussian process ξ (in practice, this amounts to choosing a parametric form for the mean of ξ and a parametric covariance function k_θ)
3. Generate a Monte Carlo sample $\mathbb{X}_m = \{Y_1, \ldots, Y_m\}$ of size m from $P_{\mathbb{X}}$
4. While the evaluation budget N is not exhausted,
   - 4.1 *optional step*: estimate the parameters of the covariance function (case of a plug-in approach);
   - 4.2 select a new evaluation point, using past evaluation results, the prior ξ and $\mathbb{X}_m$;
   - 4.3 perform the new evaluation.
5. Estimate the probability of failure obtained from the N evaluations of f (for instance, by using $E_N(\alpha_m) = \frac{1}{m}\sum_j p_N(Y_j)$).

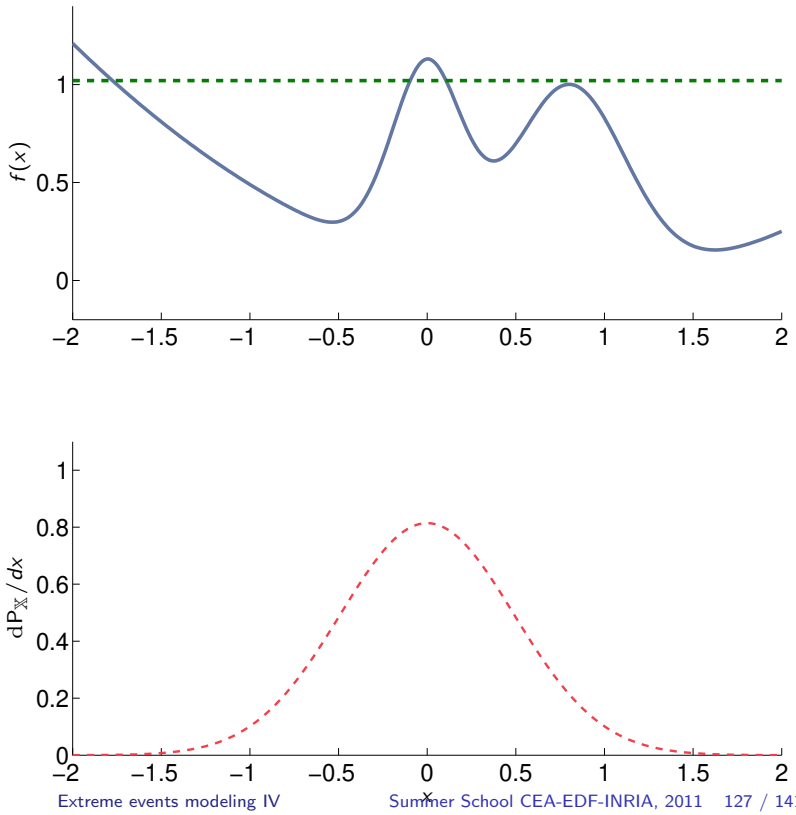
Procedure to select a new evaluation point $X_{n+1} \in \mathbb{X}$ using a SUR strategy

1. Compute the kriging approximation $\widehat{f}_n$ and kriging variance σ_n^2 on $\mathbb{X}_m$ from $\mathcal{I}_n$
2. For each candidate point Y_j, $j \in \{1, \ldots, m\}$,
   - 2.1 for each point Y_k, $k \in \{1, \ldots, m\}$, compute the kriging weights $\lambda_i(Y_k; \{\underline{X}_n, Y_j\})$, $i \in \{1, \ldots, (n+1)\}$, and the kriging variances $\sigma^2(Y_k; \{\underline{X}_n, Y_j\})$
   - 2.2 compute $z_{n+1,q}(Y_j) = \widehat{f}_n(Y_j) + \sigma_n(Y_j) u_q$, for $q = 1, \ldots, Q$
   - 2.3 for each $z_{n+1,q}(Y_j)$, $q \in \{1, \ldots, Q\}$,
     - 2.3.1 compute the kriging approximation $\tilde{f}_{n+1,j,q}$ on $\mathbb{X}_m$ from $\mathcal{I}_n \cup \left(Y_j, f(Y_j) = z_{n+1,q}(Y_j)\right)$, using the weights $\lambda_i(Y_k; \{\underline{X}_n, Y_j\})$, $i = 1, \ldots, (n+1)$, $k = 1, \ldots, m$, obtained at Step 2.1.
     - 2.3.2 for each $k \in \{1, \ldots, m\}$, compute $v_{n+1}\left(Y_k;\ Y_j, z_{n+1,q}(Y_j)\right)$, using u, $\tilde{f}_{n+1,j,q}$ obtained in 2.3.1, and $\sigma^2(Y_k; \{\underline{X}_n, Y_j\})$ obtained in 2.1
   - 2.4 compute $J_n(Y_j) = \sum_{k=1}^{m} \sum_{q=1}^{Q} w_q' \, v_{n+1}\left(Y_k;\ Y_j, z_{n+1,q}(Y_j)\right)$.
3. Find $j^\star = \operatorname{argmin}_j J_n(Y_j)$ and set $X_{n+1} = Y_{j^\star}$

4.6 One-dimensional illustration

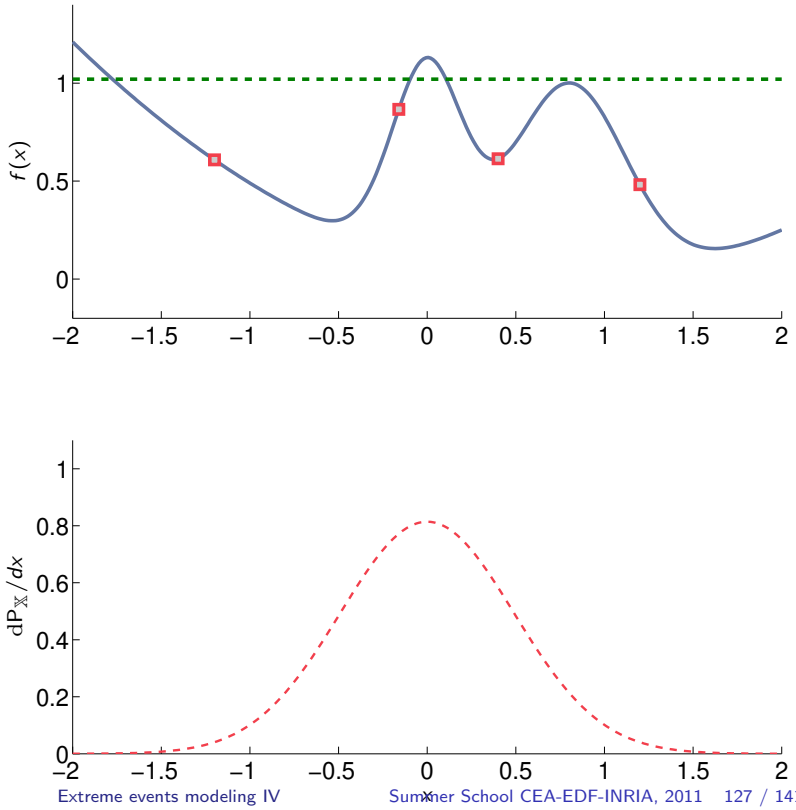
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



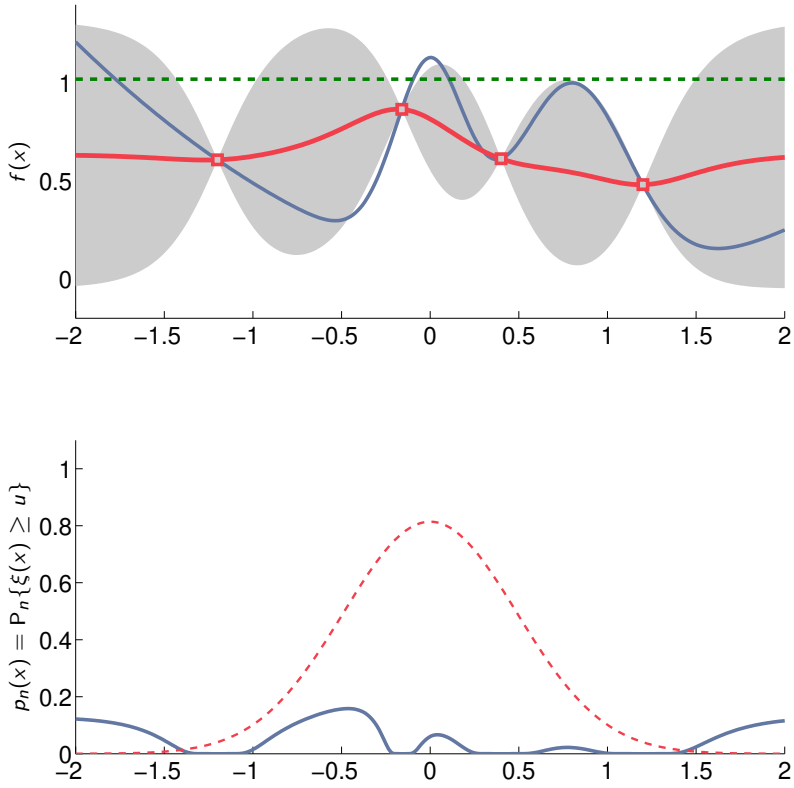
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



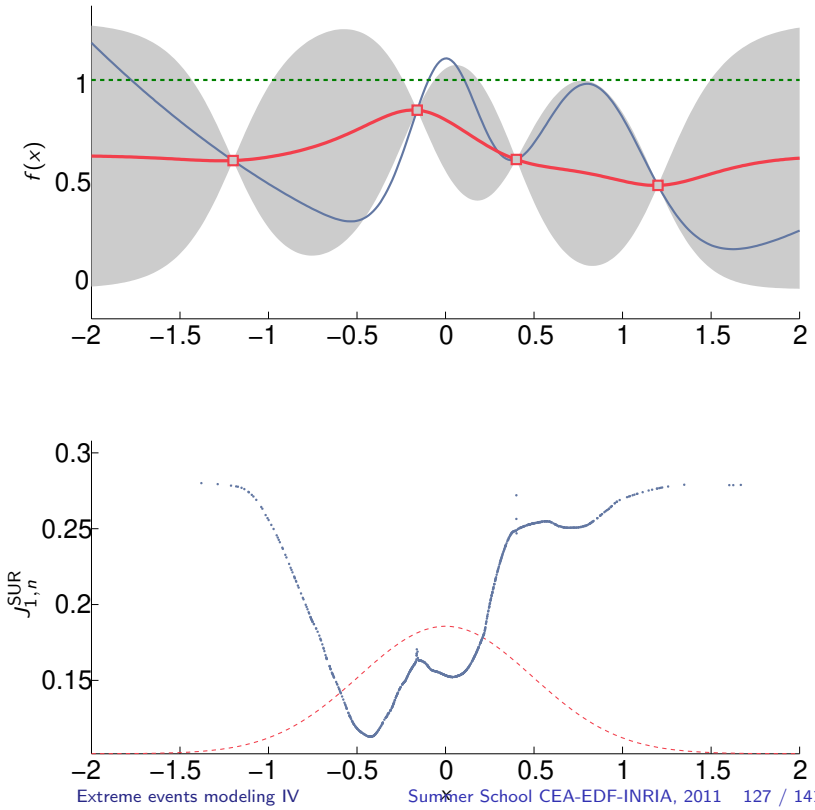
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



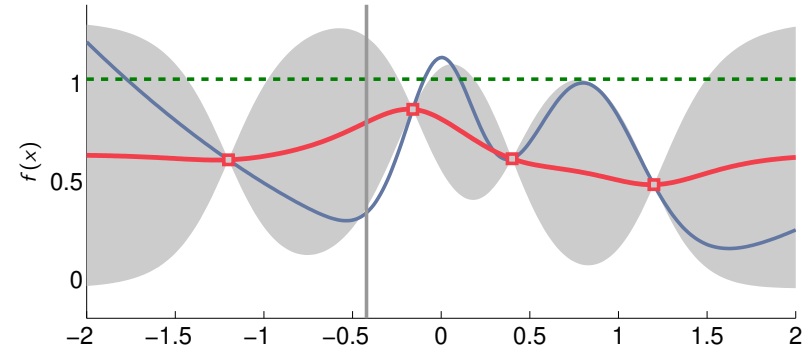
Estimation of the volume of an excursion set

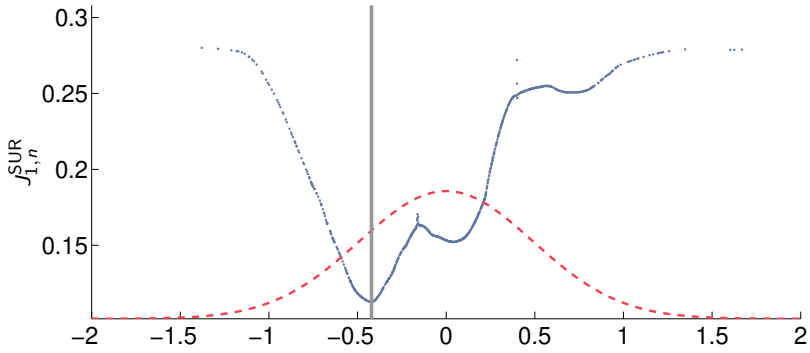
- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



Estimation of the volume of an excursion set

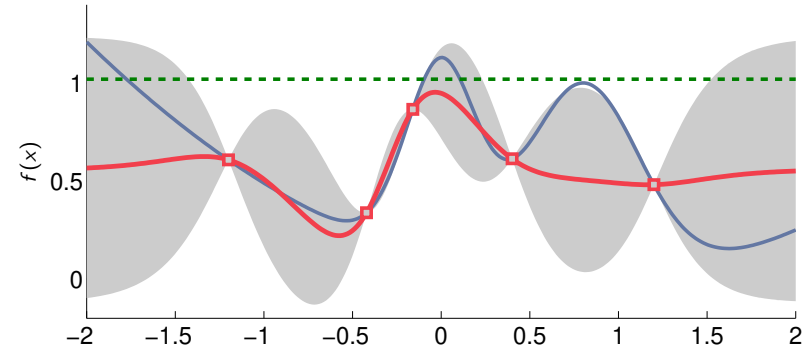
- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation

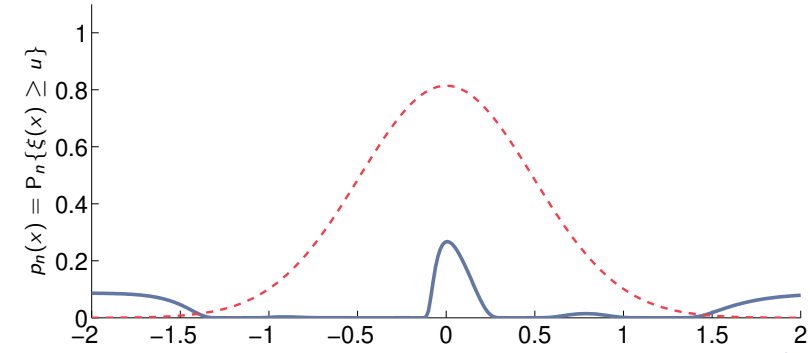




Estimation of the volume of an excursion set

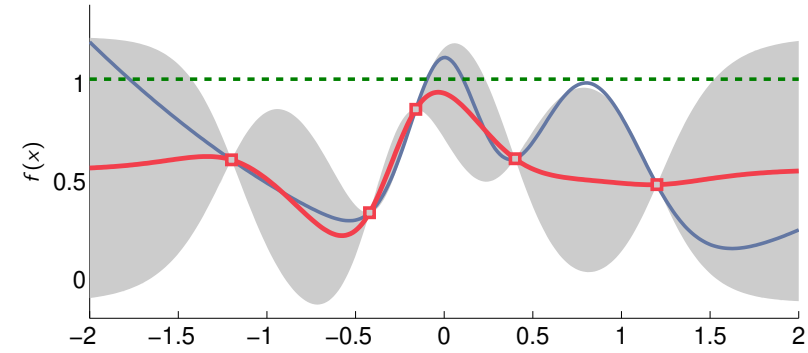
- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation

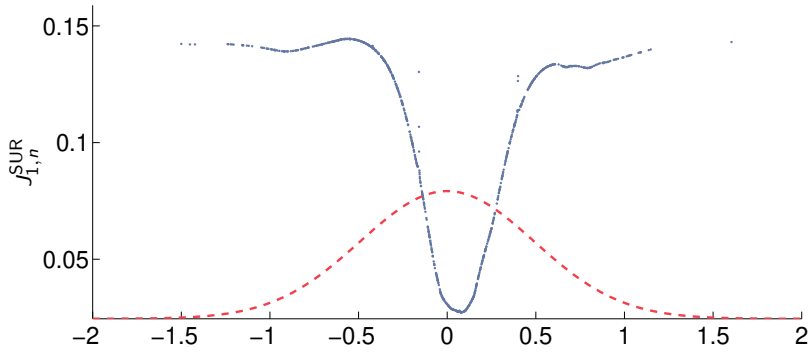




Estimation of the volume of an excursion set

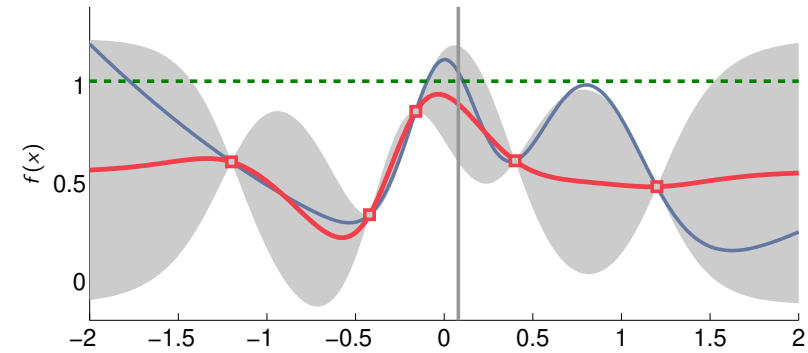
- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- **Computation and minimization of J_n**
- Position of next evaluation

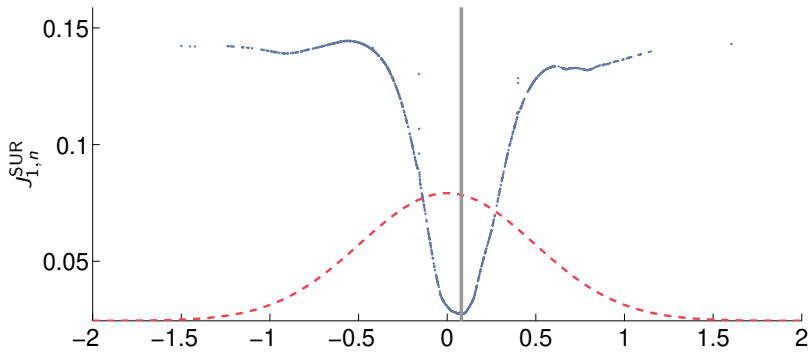




Estimation of the volume of an excursion set

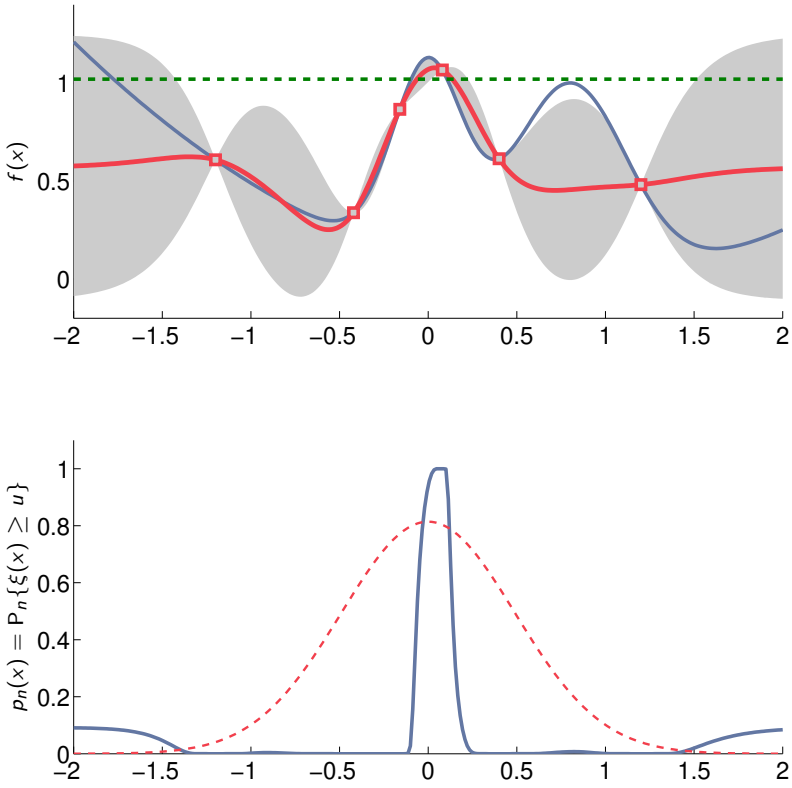
- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- **Position of next evaluation**





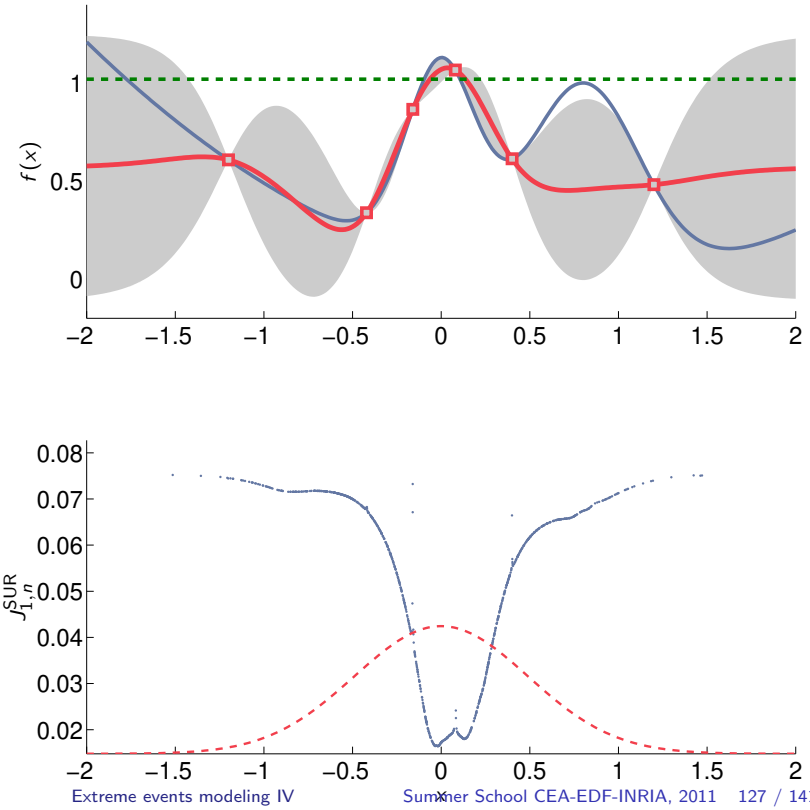
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



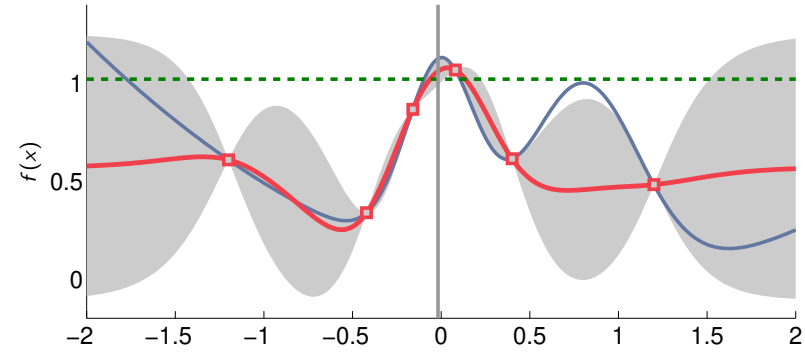
Estimation of the volume of an excursion set

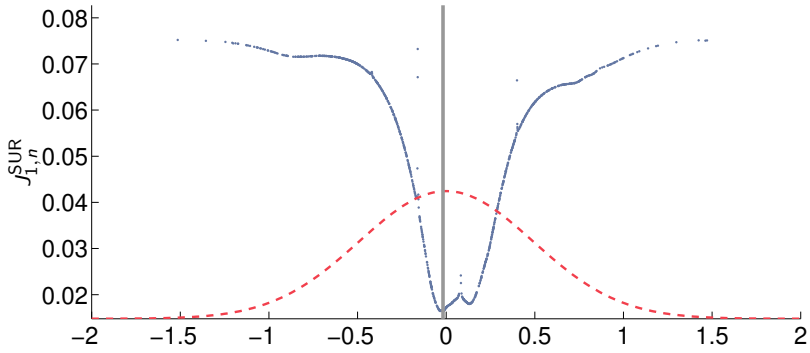
- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



Estimation of the volume of an excursion set

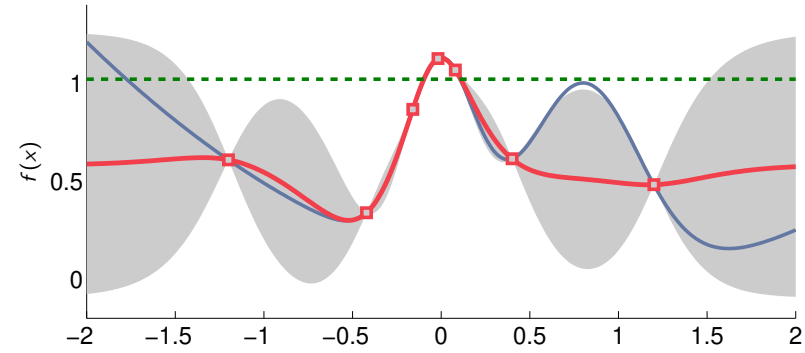
- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation

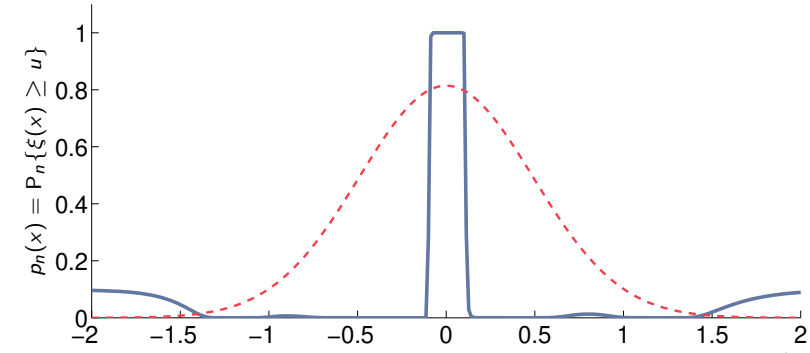




Estimation of the volume of an excursion set

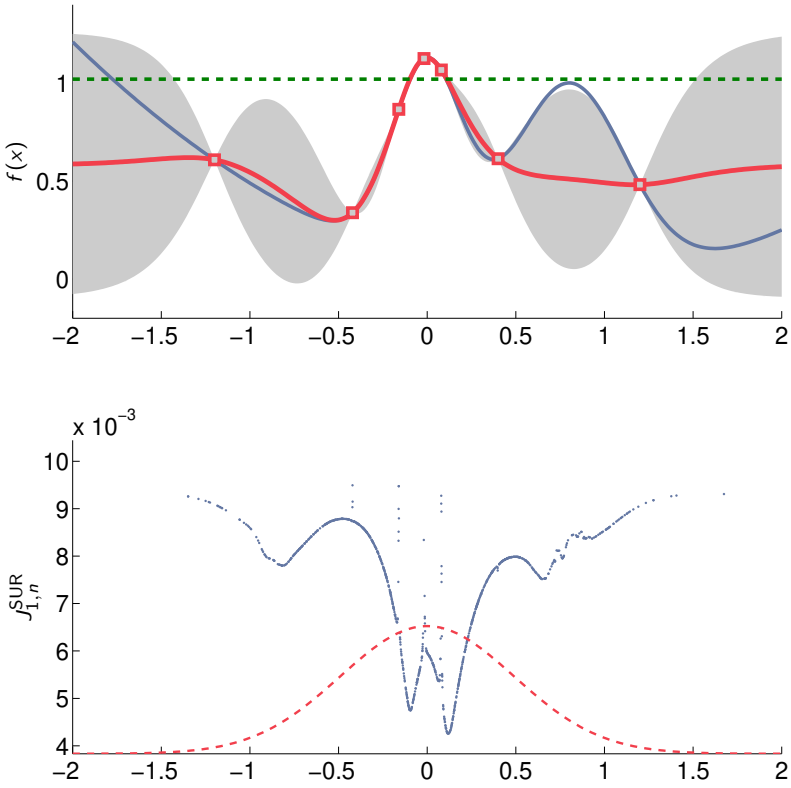
- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation





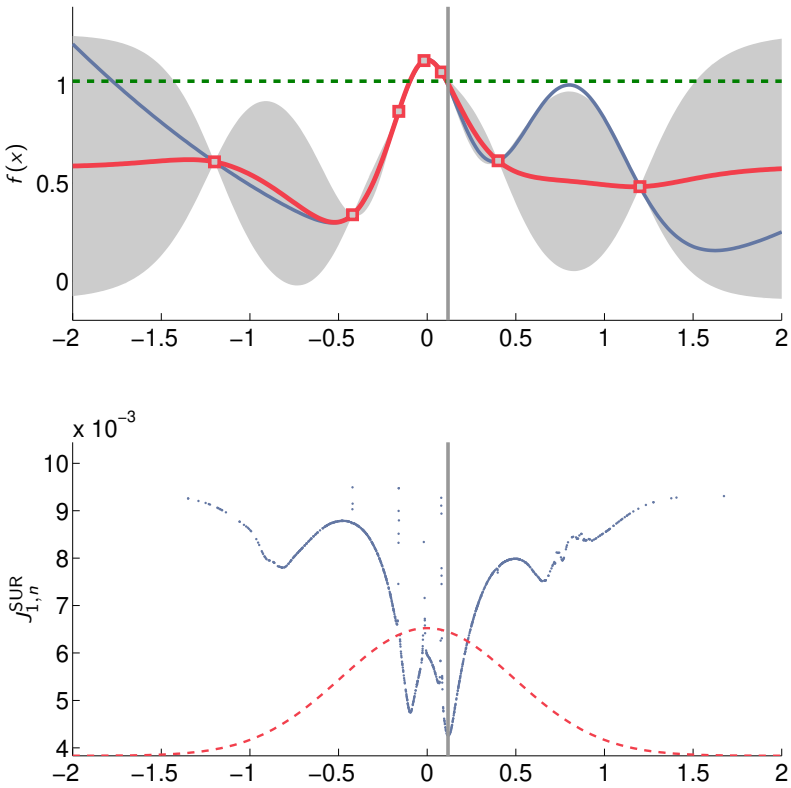
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- **Computation and minimization of J_n**
- Position of next evaluation



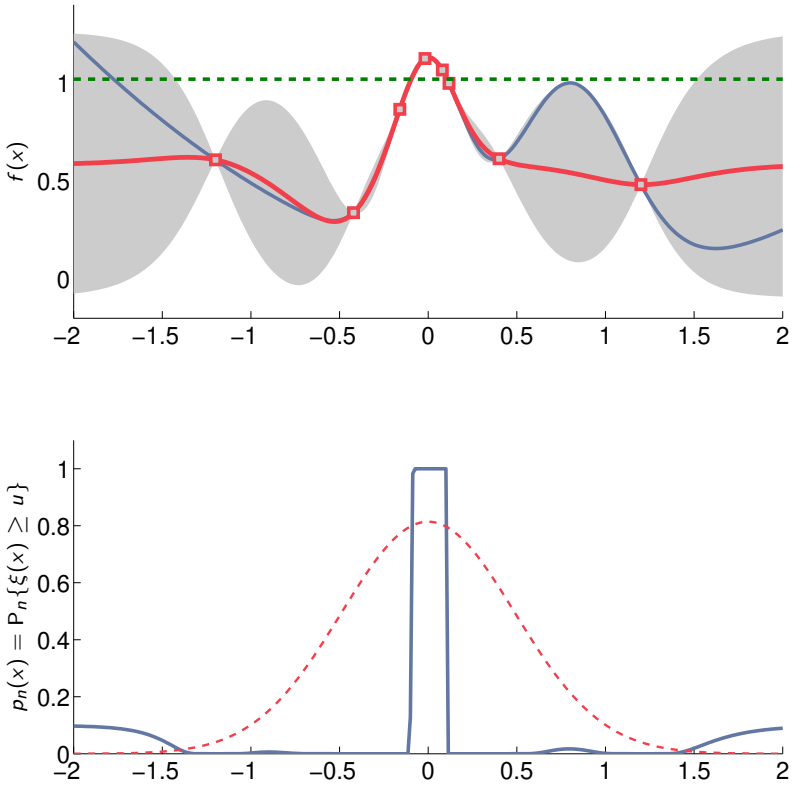
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- **Position of next evaluation**



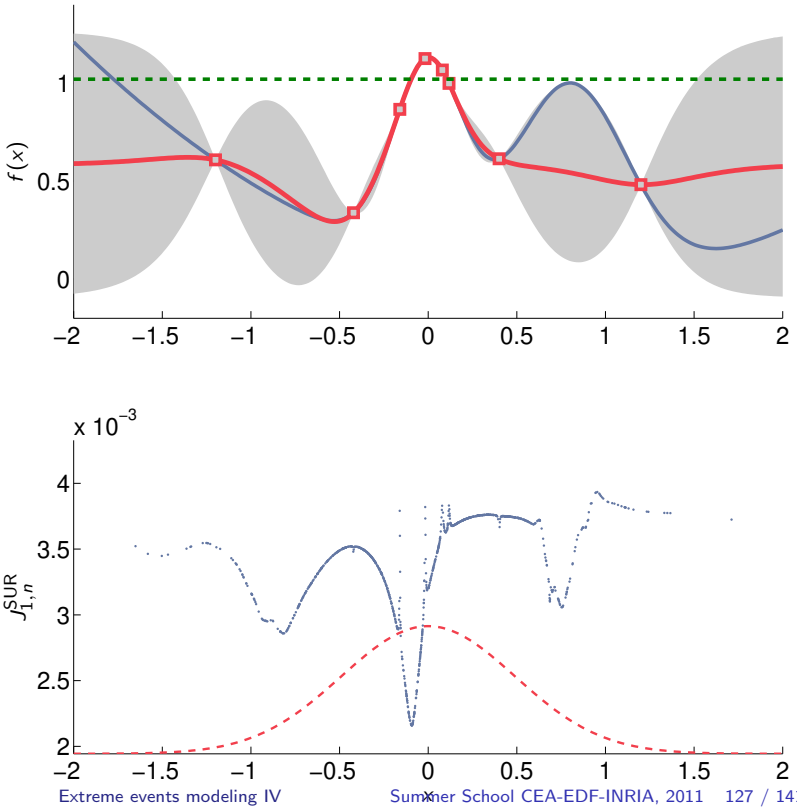
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



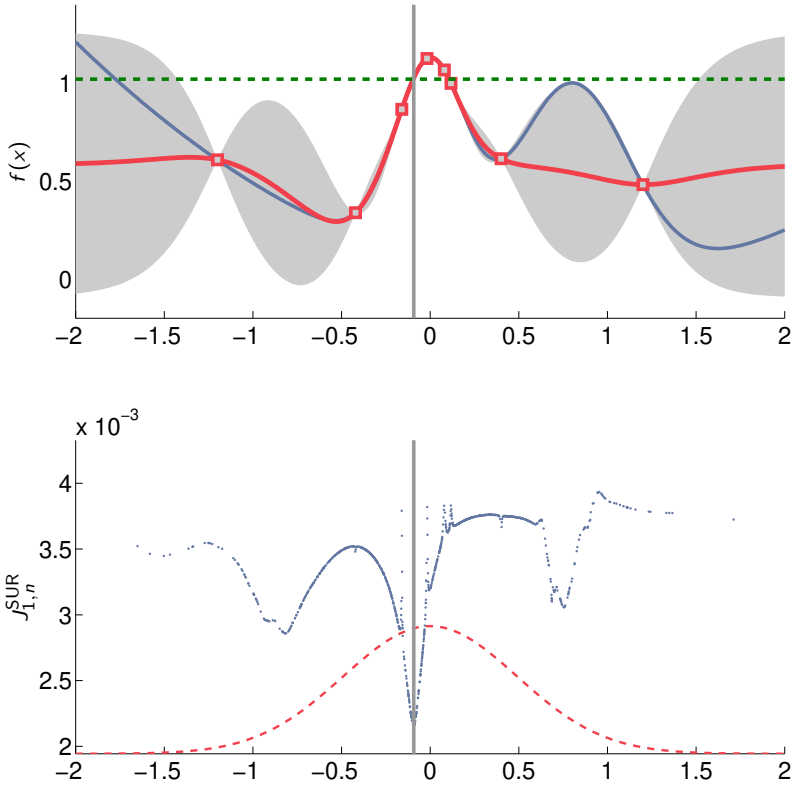
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



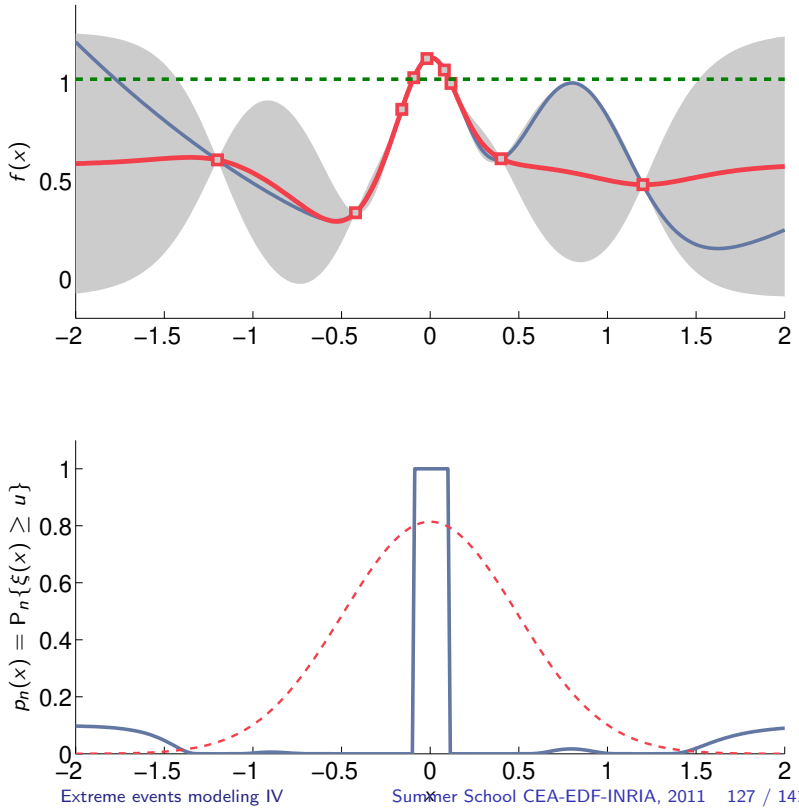
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



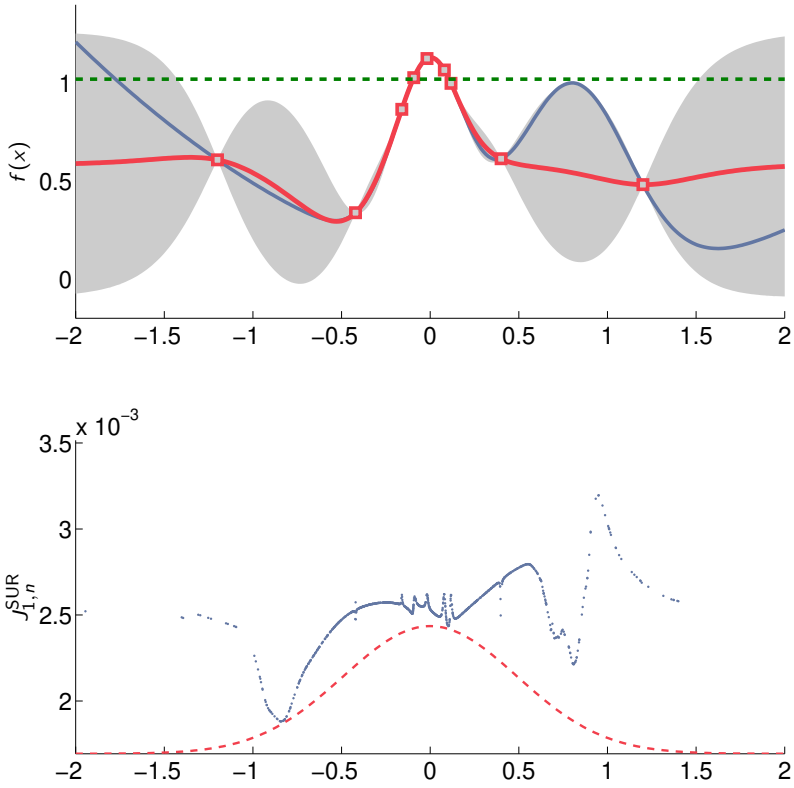
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



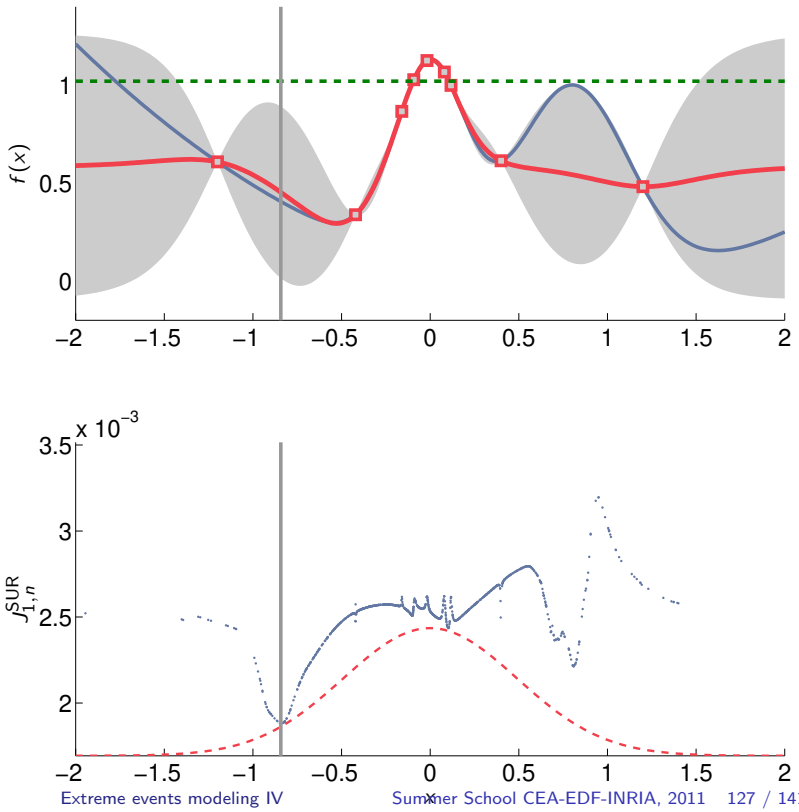
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- **Computation and minimization of J_n**
- Position of next evaluation



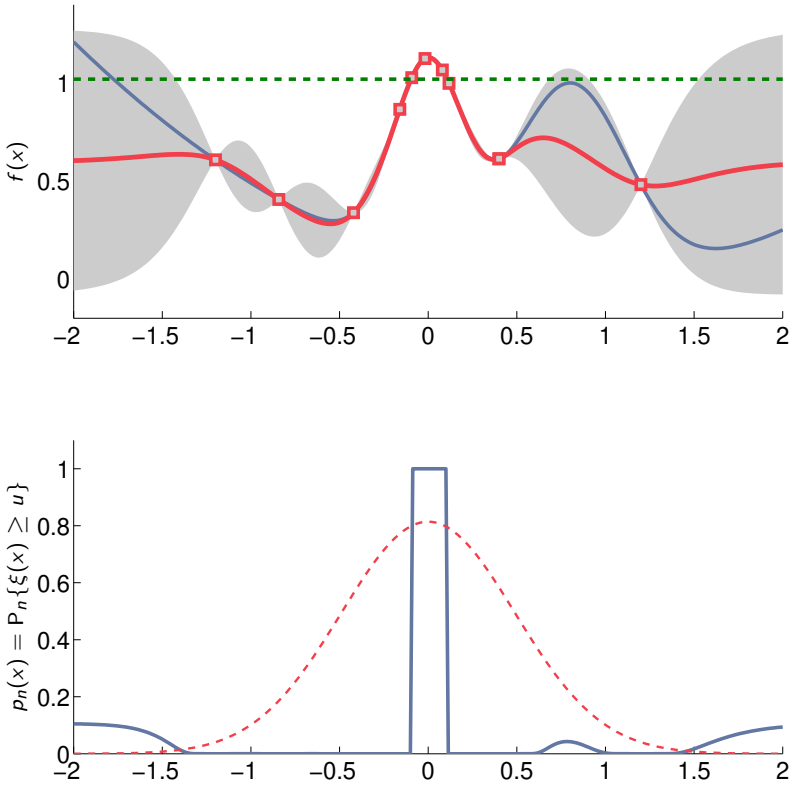
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- **Position of next evaluation**



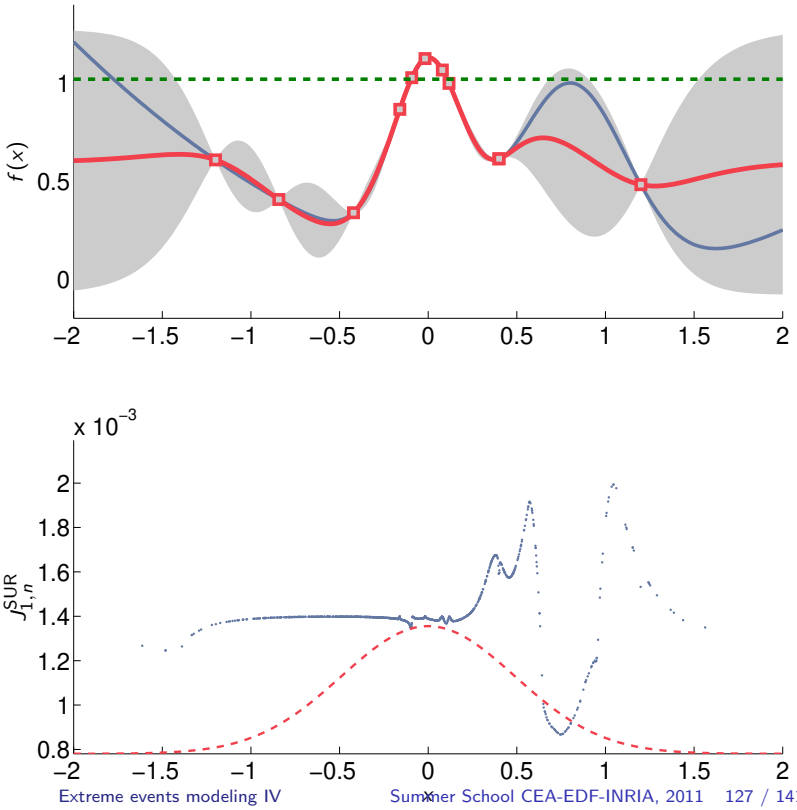
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



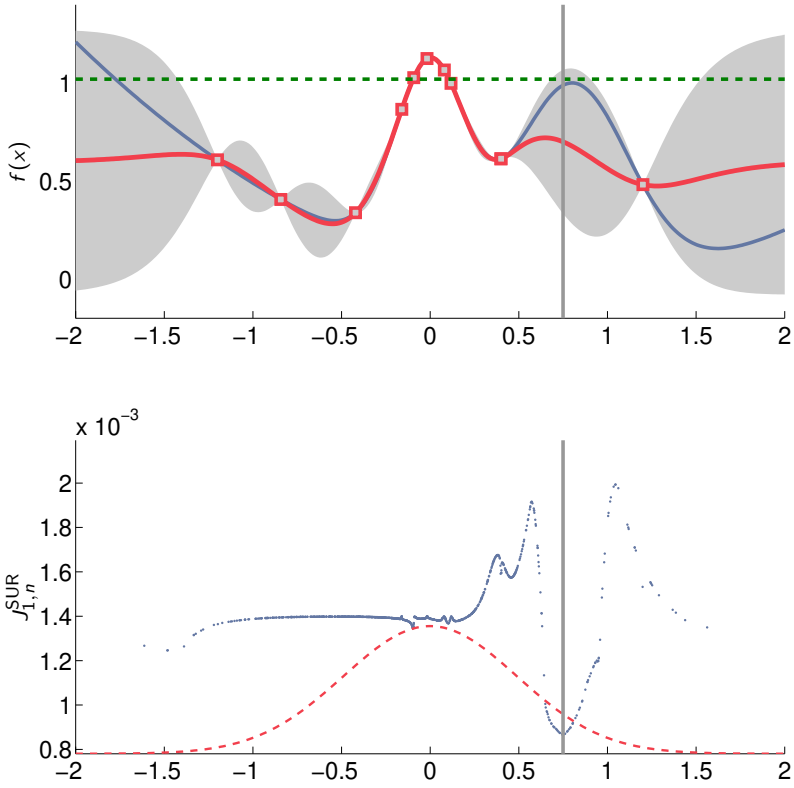
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



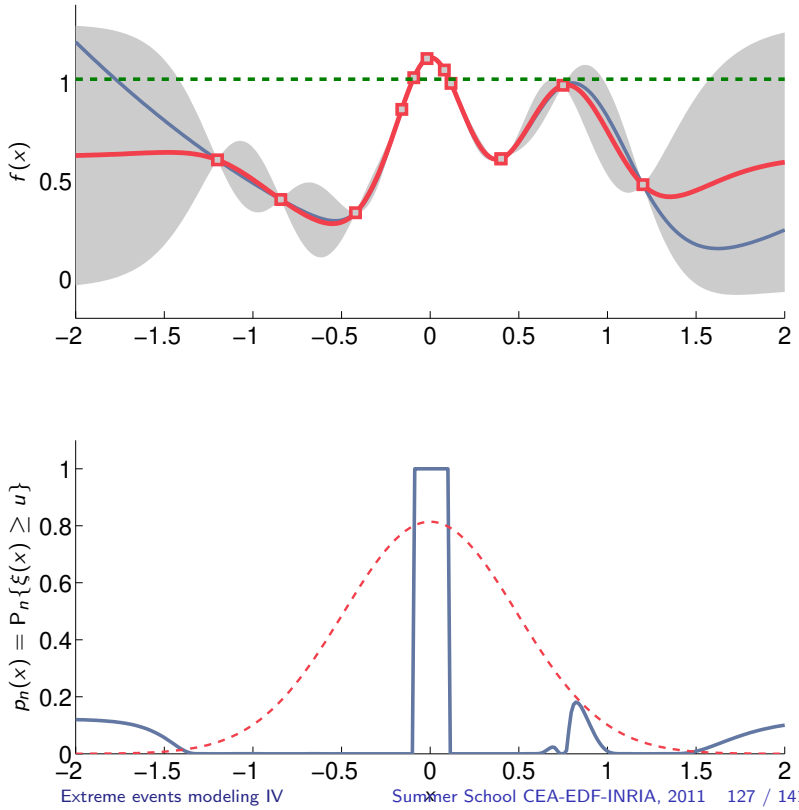
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



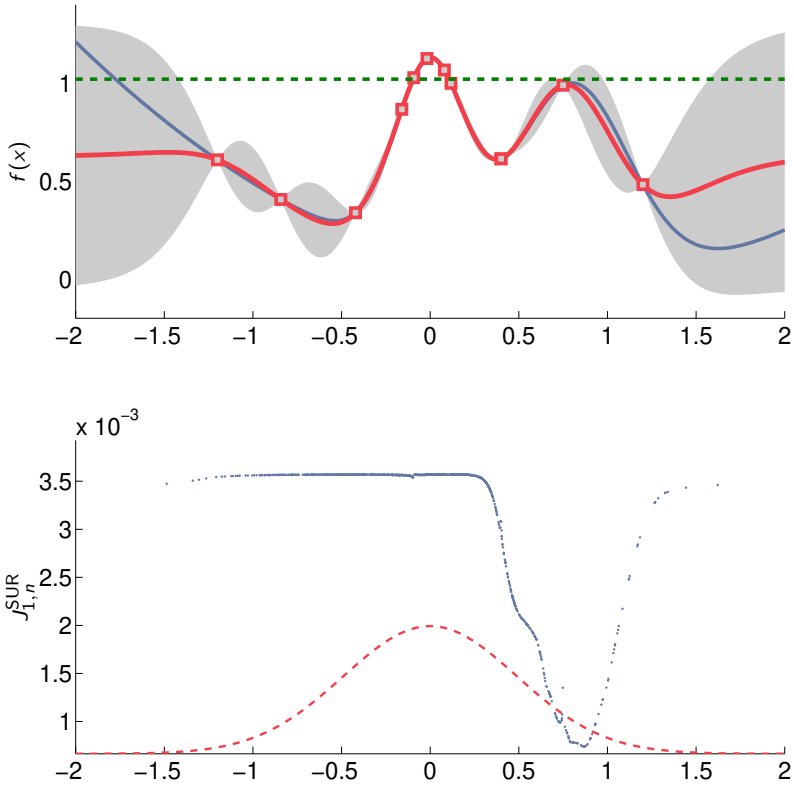
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



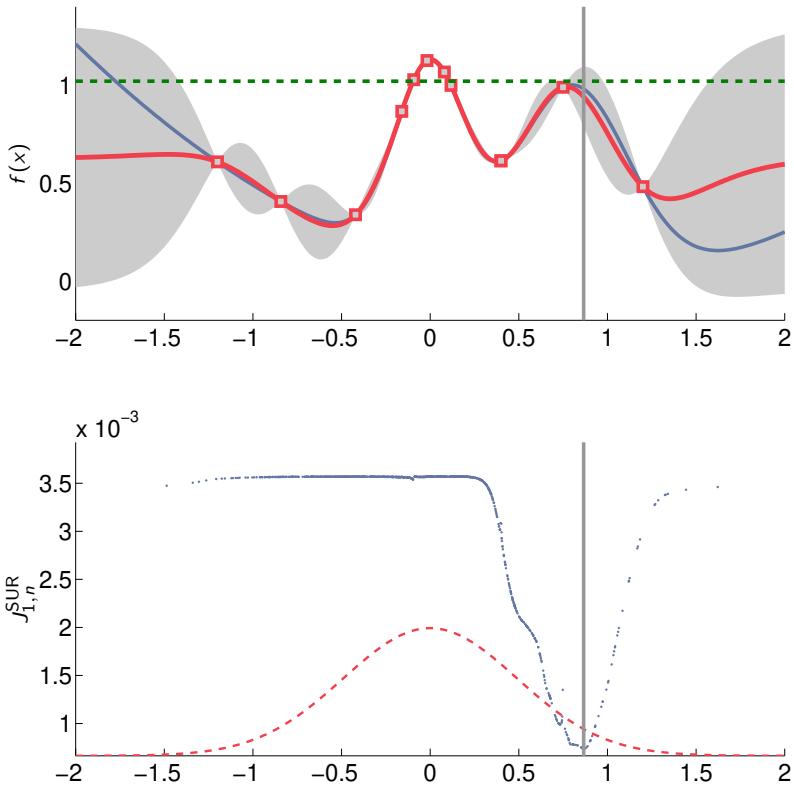
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- **Computation and minimization of J_n**
- Position of next evaluation



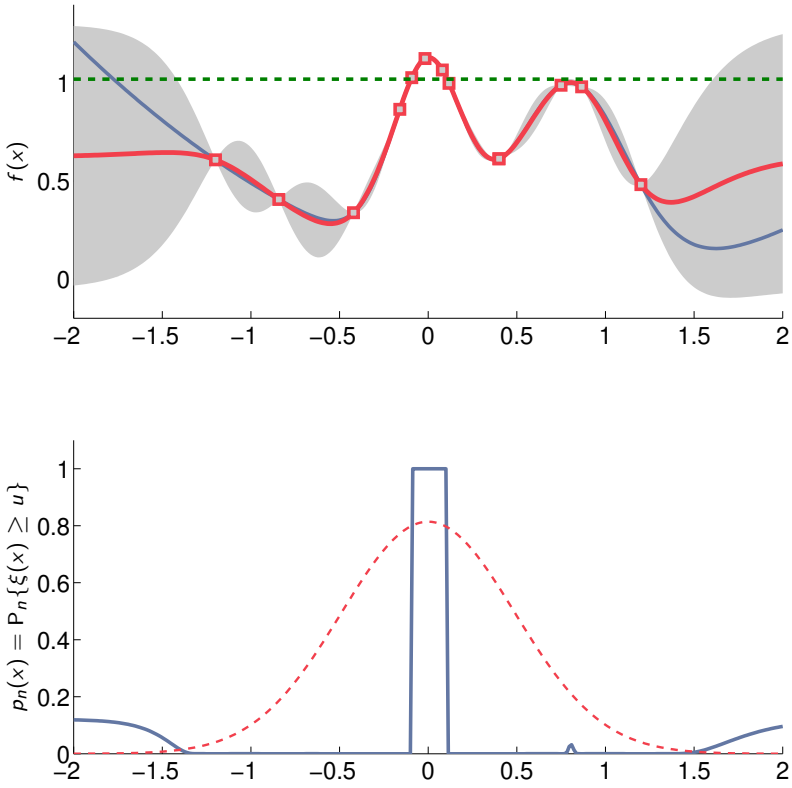
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- **Position of next evaluation**



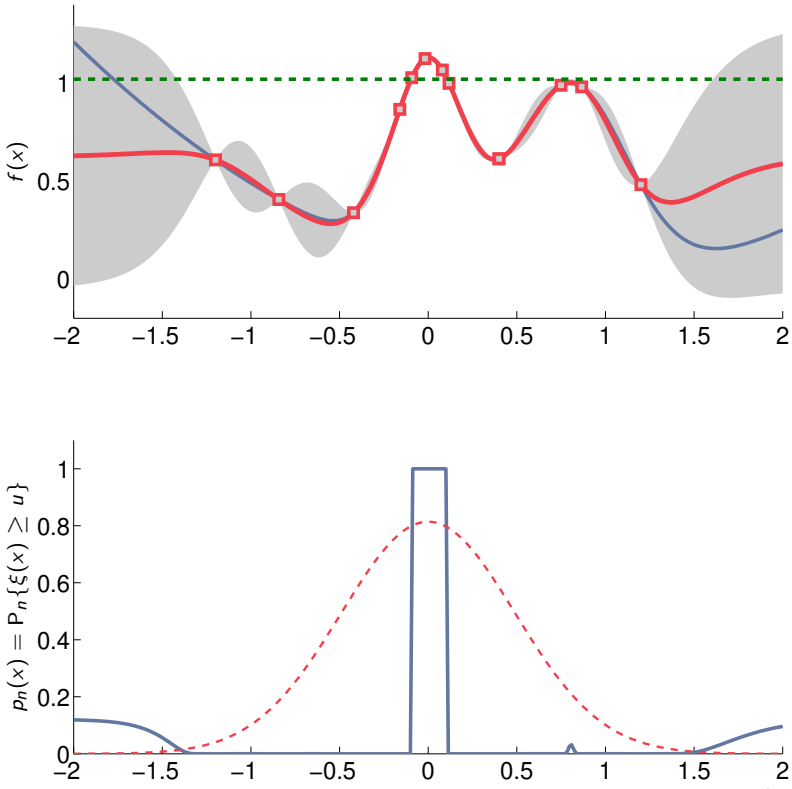
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



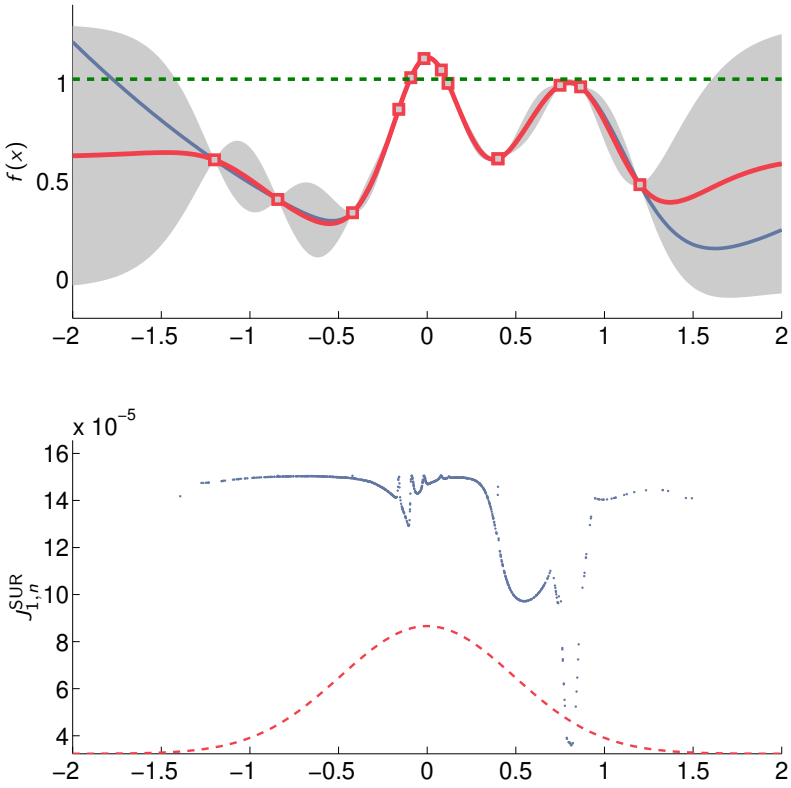
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



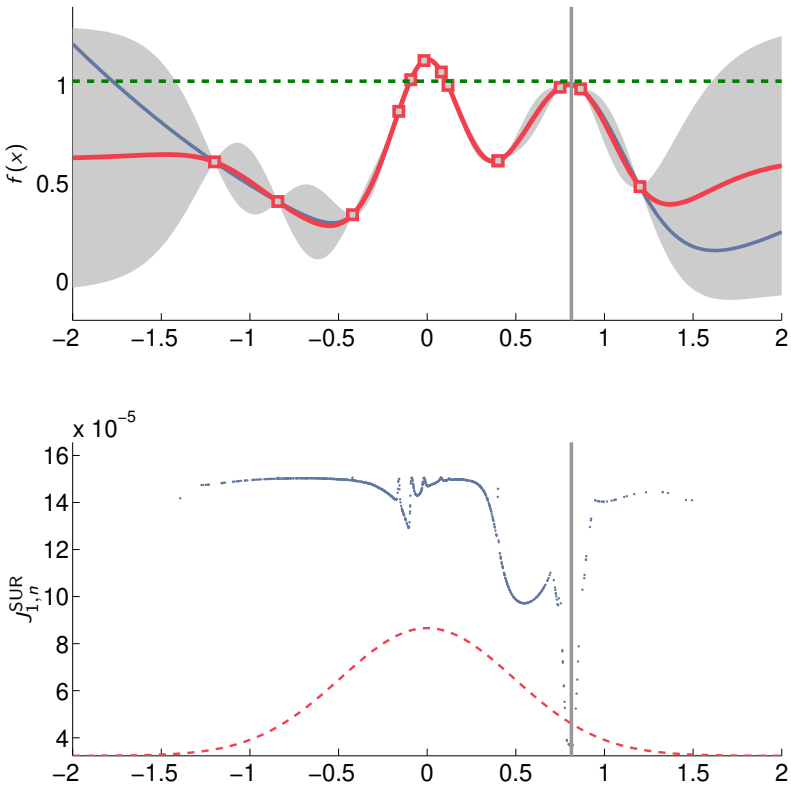
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



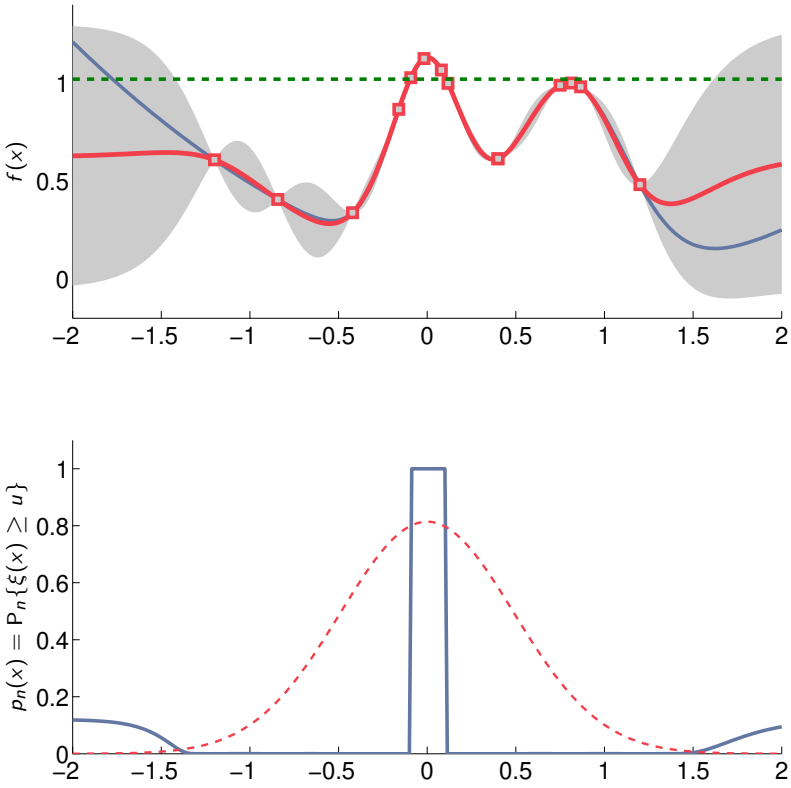
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



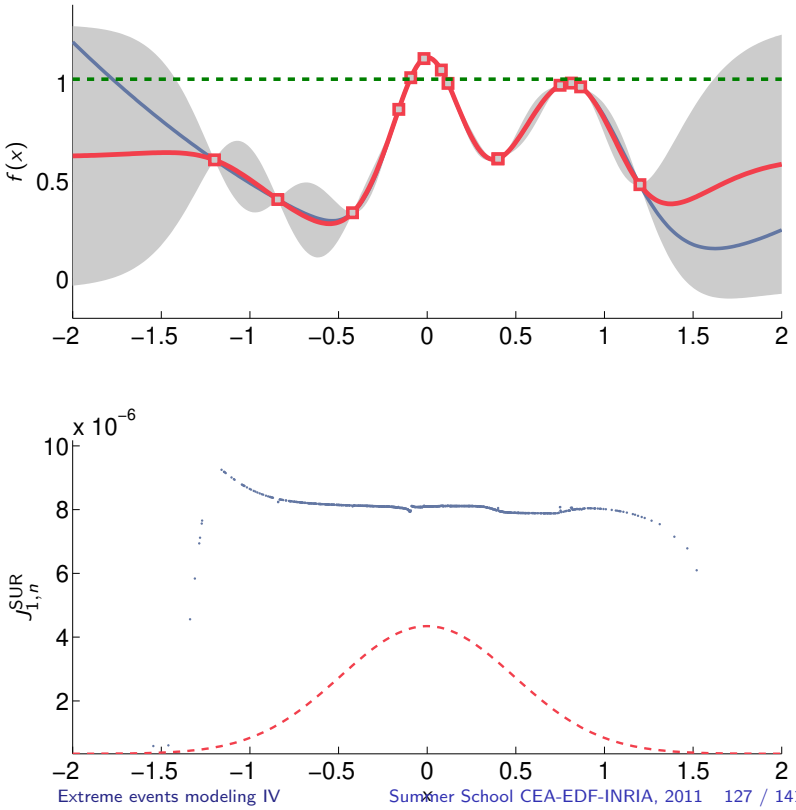
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- **Computation and minimization of J_n**
- Position of next evaluation



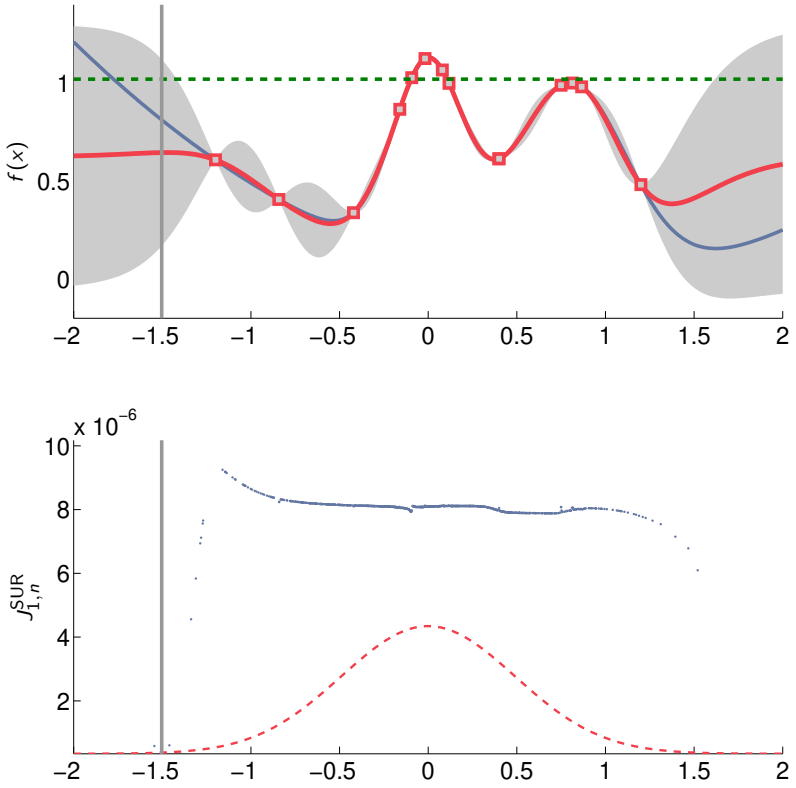
Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- **Position of next evaluation**



Estimation of the volume of an excursion set

- Unknown f, threshold u, and pdf. $\mathrm{dP}_{\mathbb{X}}/dx$ over $\mathbb{X}$
- Initial design
- Construction of f_n, confidence intervals, probability of excursion $\mathrm{P}\{f(x) \geq u\}$, $x \in \mathbb{X}$
- Computation and minimization of J_n
- Position of next evaluation



Additional remarks

- In terms of computational complexity, a SUR strategy to estimate a probability of failure is more expensive than the EI/EGO algorithm
- Kriging takes $\mathrm{O}(mn^2)$ operations to predict the value of f at m locations from n evaluation results of f
- In the procedure to select an evaluation
  - a first kriging prediction is performed at Step 1
  - m different predictions have to performed at Step 2.1.
- The cost becomes rapidly burdensome for large values of n and m
- To work on applications where m must be large (small probabilities of failure), we can avoid dealing with candidate points that have a very low probability of misclassification
  (they are probably far from the frontier of the domain of failure)
- It is also likely that those points with a low probability of misclassification will have a very small contribution to the variance of the error of estimation $\widehat{\alpha}_n - \alpha_m$.
- The idea is to rewrite the sampling strategy, in such a way that the summation over m, and the search set for the minimizer, is restricted to a subset of points Y_j corresponding to the m_0 largest values of $\tau_n(Y_j)$.

5. Estimation of a quantile using a SUR strategy

5.1 Statement of the problem

Quantile estimation

- $\mathbb{X} \subseteq \mathbb{R}^d$ space of uncertain factors
- $\mathrm{P}_{\mathbb{X}}$ probability measure on $\mathbb{X}$
- $f : \mathbb{X} \to \mathbb{R}$ unknown function, whose value is a quantity of interest
- We consider the problem of estimating a quantile

$$q_\alpha(f) = \inf\{u \in \mathbb{R};\ \mathrm{P}_{\mathbb{X}}\{f \leq u\} \geq \alpha\}$$

for a given probability α, that is close to one

(In practice, knowing the value of a quantile makes it possible to assess the safety of a system.)

Quantile estimation by Monte Carlo

- To estimate $q_\alpha(f) \to$ draw an i.i.d m-sample $Y_1, \ldots, Y_m \sim \mathrm{P}_{\mathbb{X}}$, and consider the empirical estimator

$$q_{\alpha,m}(f) = \min\left\{z;\ \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{Z_i \leq z} \geq \alpha\right\} = Z_{\left(\lceil \alpha m \rceil\right)}$$

where

$$Z_i = f(Y_i), \quad i = 1, \ldots, m$$

and $Z_{(i)}$ stands for the ith order statistics of the sample $Z_1, \ldots, Z_m$

- It is well known that

$$\sqrt{m}\left(q_{\alpha,m}(f) - q_\alpha(f)\right) \to_m \mathcal{N}\left(0, \sigma^2\right)$$

with $\sigma^2 = \frac{\alpha(1-\alpha)}{p_Z(q_\alpha(f))^2}$, where p_Z is the pdf of $Z = f(X)$, $X \sim \mathrm{P}_{\mathbb{X}}$
(see, e.g., Wasserman. 2006. All of Nonparametric Statistics. Springer.)

$\Longrightarrow$ a high value of m must be used in order to obtain a good estimator of q_α

- If the evaluation of f is expensive, the budget of evaluations can be very limited

  ➟ we need to find small variance estimators

- Classical approaches : importance sampling, control variate sampling... (see, e.g., Glynn 96, Hesterberg and Nelson 98, Cannamela et al. 08)

- Here: we want to use a SUR approach

5.2 Stepwise uncertainty reduction for the problem of estimating a quantile

Stepwise uncertainty reduction for the problem of quantile estimation

- As in the case of the estimation of a probability of failure, assume a fixed m-sample: $Y_i \overset{\text{i.i.d}}{\sim} \mathrm{P}_{\mathbb{X}}$, $i = 1, \ldots, m$

... we want to approximate the empirical estimator

$$q_{\alpha,m}(f) = \min\left\{y; \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{Y_i \leq y} \geq \alpha\right\} = Y_{\left(\lceil \alpha m \rceil\right)}$$

(meta-estimation)

- Approach: choose sequentially evaluation points of f

$$X_1(f), \ldots, X_n(f) \in \{Y_1, \ldots, Y_m\}$$

to construct a meta-estimator $\widehat{q}_{\alpha,n}$ of $q_{\alpha,m}(f)$ such that

$\widehat{q}_{\alpha,n}$ is close to $q_{\alpha,m}(f)$
with $n \ll m$

Stepwise uncertainty reduction for the problem of quantile estimation

- Choose of a prior about f under the form of Gaussian random process ξ
- Restricting ξ to be a Gaussian process makes it possible to derive the posterior distribution of ξ after n evaluations
- Consider the estimator

$$\widehat{q}_{\alpha,n} = \mathrm{E}_n(q_{\alpha,m}(\xi))$$

- How to compute $\widehat{q}_{\alpha,n}$?

➡ Contrarily to the case of the probability of failure, $\widehat{q}_{\alpha,n}$ does not have a simple expression as a function of the kriging predictor and kriging variance.

- In practice, $\widehat{q}_{\alpha,n}$ can be approximated by simulation

Approximation of $\widehat{q}_{\alpha,n}$ by simulation of sample paths

- For $i = 1, \ldots, M$:
  - (a) Generate a sample path $f^{(n,i)}$ according to the distribution of ξ conditioned on $\mathcal{I}_n(\xi)$
    (Using a conditioning-by-kriging technique, see, e.g., Chiles 99)
  - (b) Compute
    $$q_\alpha^{(n,i)} = q_{\alpha,m}(f^{(n,i)})$$
    based on the m-sample $\left\{f^{(n,i)}(Y_j)\right\}_{j=1,\ldots,m}$.
- Thus, we obtain a sample
  $$q_\alpha^{(n,1)}, \ldots, q_\alpha^{(n,M)}$$
  distributed according to the posterior distribution of $q_{\alpha,m}(\xi)$
- Define $\widehat{q}'_{\alpha,n} = \frac{1}{M}\sum_{i=1}^{M} q_\alpha^{(n,i)} \rightarrow$ approximates $\widehat{q}_{\alpha,n}$ at rate $M^{-1/2}$

SUR strategy to estimate a quantile

- Define a one-step lookahead strategy $\underline{X}_N$ by
  $$X_{n+1} = \underset{x\in\{Y_1,\ldots,Y_m\}}{\operatorname{argmin}} J_n(x) := \mathrm{E}_n\left\{\left(q_{\alpha,m}(\xi) - \widehat{q}_{\alpha,n+1}\right)^2 \,\middle|\, X_{n+1} = x\right\},$$
  where $\widehat{q}_{\alpha,n+1}$ is computed from the observations $\xi(X_i)$, $i = 1, \ldots, n$ and the random outcome $\xi(x)$
- Note that for each n, X_n is $\mathcal{F}_{n-1}$-measurable

Computation of the sampling criterion J_{n+1}

In practice, the sampling criterion J_n can be computed at x using the fact that

$$\begin{aligned} J_n(x) &= \mathrm{E}_n\left\{(q_{\alpha,m}(\xi) - \widehat{q}_{\alpha,n+1})^2 \mid X_{n+1} = x\right\} \\ &= \mathrm{E}_n\left\{\mathrm{E}_n\left\{(q_{\alpha,m}(\xi) - \widehat{q}_{\alpha,n+1})^2\right\} \,\middle|\, X_{n+1} = x\right\}. \end{aligned}$$

➟ the numerical approximation of the inner expectation can be carried out as follows:

1- Compute quantiles $q_\alpha^{(n+1,i)}$ (by simulation, as above), conditioning the sample paths by $\xi(X_1), \ldots, \xi(X_{n-1})$ and $\xi(x) = z$

2- Define $\widehat{q}'_{\alpha,n+1}(x,z) = \frac{1}{M}\sum_{i=1}^M q_\alpha^{(n+1,i)}$ and $\gamma_{n+1}(x,z) = \frac{1}{M-1}\sum_{i=1}^M (q_\alpha^{(n+1,i)} - \widehat{q}'_{\alpha,n+1}(x,))^2$.

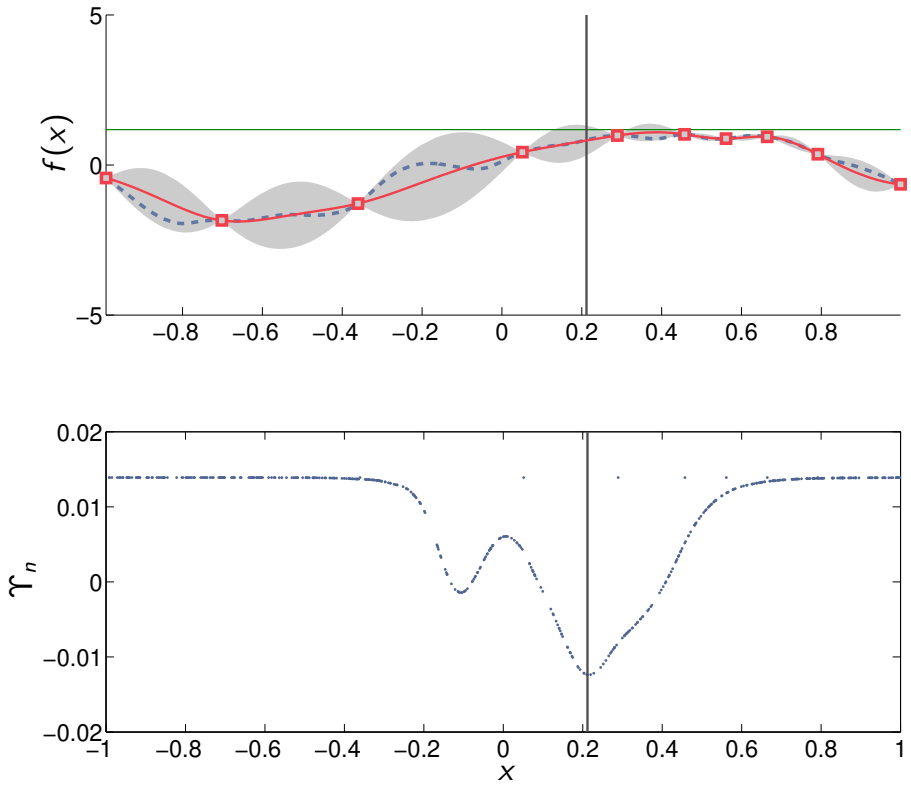
➟ The numerical approximation of the outer expectation consists in approximating the integral

$$\int_{\mathbb{R}} \gamma_{n+1}(x,z)\,\mathrm{dQ}_{n,x}(z)$$

which can be carried out as in the case of the estimation of a probability of failure.

5.3 Example

Example


5
0
−5
f(x)
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
0.02
0.01
0
−0.01
−0.02
Υn
−1
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
1
x