



**HAL**  
open science

## Bayesian Optimization

Julien Bect, Emmanuel Vazquez

► **To cite this version:**

Julien Bect, Emmanuel Vazquez. Bayesian Optimization. Doctoral. CEA-EDF-INRIA Numerical analysis Summer school 2017. Design and optimization under uncertainty of large-scale numerical models., Université Pierre et Marie Curie (Paris VI), Paris, France. 2017, pp.224. hal-03277561

**HAL Id: hal-03277561**

**<https://cel.hal.science/hal-03277561v1>**

Submitted on 4 Jul 2021

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

# Bayesian Optimization

Engineering design under uncertainty using  
expensive-to-evaluate numerical models

Julien Bect and Emmanuel Vazquez

Laboratoire des signaux et systèmes, Gif-sur-Yvette, France

CEA-EDF-INRIA Numerical Analysis Summer School  
Université Pierre et Marie Curie (Paris VI)  
Paris, 2017, July 3–7

## Lecture 1 : From meta-models to UQ

- 1.1 Introduction
- 1.2 Black-box modeling
- 1.3 Bayesian approach
- 1.4 Posterior distribution of a quantity of interest
- 1.5 Complements on Gaussian processes

## Lecture 2 : Bayesian optimization (BO)

- 2.1. Decision-theoretic framework
- 2.2. From Bayes-optimal to myopic strategies
- 2.3. Design under uncertainty

## References

## Lecture 1 : From meta-models to UQ

- 1.1 Introduction
- 1.2 Black-box modeling
- 1.3 Bayesian approach
- 1.4 Posterior distribution of a quantity of interest
- 1.5 Complements on Gaussian processes

## Lecture 2 : Bayesian optimization (BO)

- 2.1. Decision-theoretic framework
- 2.2. From Bayes-optimal to myopic strategies
- 2.3. Design under uncertainty

## References

## Lecture 1 : From meta-models to UQ

### 1.1 Introduction

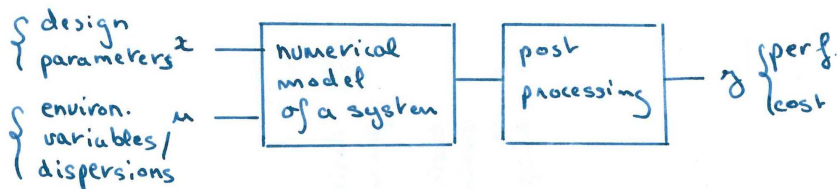
1.2 Black-box modeling

1.3 Bayesian approach

1.4 Posterior distribution of a quantity of interest

1.5 Complements on Gaussian processes

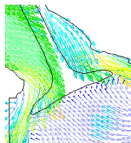
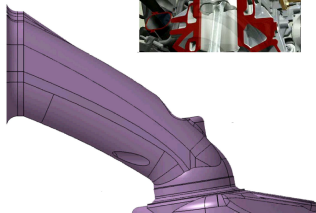
# Computer-simulation based design



- ▶ The numerical model might be time- or resource-consuming
- ▶ Design parameters might be subject to dispersions
- ▶ The system might operate under unknown conditions

# Computer-simulation based design – Example

- ▶ Computer simulations to design a product or a process, in particular
  - ▶ to find the best feasible values for **design parameters** (optimization problem)
  - ▶ to minimize the probability of failure of a product
  
- ▶ To comply with European emissions standards, the design parameters of combustion engines have to be carefully optimized
  
- ▶ The shape of intake ports controls airflow characteristics, which have direct impact on
  - ▶ the performances of the engine
  - ▶ emissions of NO<sub>x</sub> and CO
  
- ▶  $f : \mathbb{X} \subset \mathbb{R}^d \rightarrow \mathbb{R}$  performance as a function of design parameters ( $d = 20 \sim 100$ )
  
- ▶ Computing  $f(x)$  takes 5 ~ 20 hours
  
- ▶ Objective: estimate  $x^* = \operatorname{argmax}_x f(x)$



Simulation of an intake port (Navier-Stokes equ.)  
(courtesy of Renault)

## Lecture 1 : From meta-models to UQ

1.1 Introduction

**1.2 Black-box modeling**

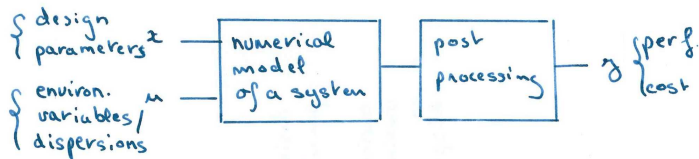
1.3 Bayesian approach

1.4 Posterior distribution of a quantity of interest

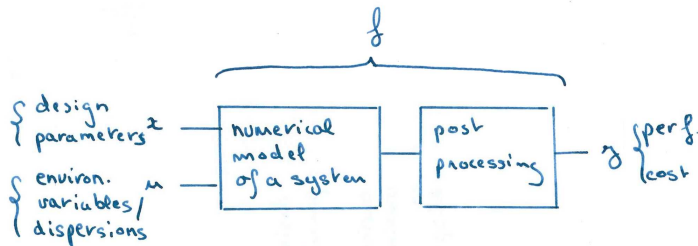
1.5 Complements on Gaussian processes



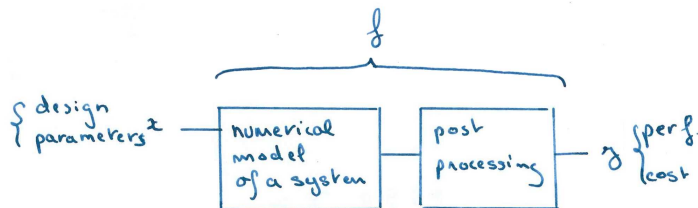
# Black-box modeling



# Black-box modeling



# Black-box modeling



For simplification  $\rightarrow$  drop  $u$

# Black-box modeling

- ▶ Let  $f : \mathbb{X} \rightarrow \mathbb{R}$  be a real function defined on  $\mathbb{X} \subseteq \mathbb{R}^d$ , where
  - ▶  $\mathbb{X}$  is the input/parameter domain of the computer simulation under study, or the **factor** (from Latin, “which acts”) space
  - ▶  $f$  is a **performance** or **cost** function (a function of the outputs of the computer simulation)

# Black-box modeling

- ▶ Let  $x_1, \dots, x_n \in \mathbb{X}$  be  $n$  simulations points
- ▶ Denote by

$$z_1 = f(x_1), \dots, z_n = f(x_n)$$

the corresp. simulation results (observations/evaluations of  $f$ )

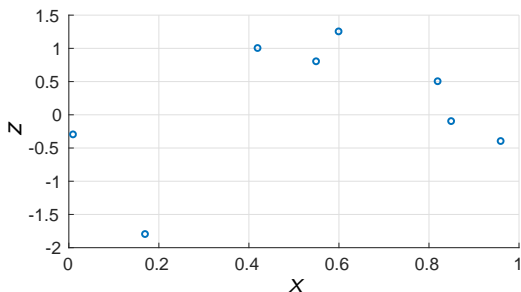
- ▶ Our objective: use the data  $\mathcal{D}_n = (x_i, z_i)_{i=1\dots n}$  to **infer properties** about  $f$
- ▶ Example: given a new  $x \in \mathbb{R}^d$ , predict the value  $f(x)$

# Black-box modeling

- ▶  $f$  is a black-box, only known through evaluation results: query an evaluation at  $x$ , observe the result
  - ▶ Predict the value of  $f$  at a given  $x$ ?
- the problem is that of constructing an **approximation** / an **estimator**  $\hat{f}_n$  of  $f$  from  $\mathcal{D}_n$
- ▶ Such a  $\hat{f}_n$  also called a **model** or a **meta-model** (because the numerical simulator is a model itself) of  $f$

## A simple curve fitting problem

- ▶ Suppose that we are given a data set of  $n$  simulation results, i.e., evaluations results of an unknown function  $f : [0, 1] \rightarrow \mathbb{R}$ , at points  $x_1, \dots, x_n$ .
- ▶ A data set of size  $n = 8$ :



## A simple curve fitting problem

- ▶ Any approximation procedure of  $f$  consists in building a function  $\hat{f}_n = h(\cdot; \theta)$  where  $\theta \in \mathbb{R}^l$  is a vector of parameters, to be estimated from  $\mathcal{D}_n$  and available prior information
- ▶ Fundamental example: **linear model**

$$\hat{f}_n(x) = h(x, \theta) = \sum_{i=1}^l \theta_i r_i(x)$$

where functions  $r_i : \mathbb{X} \rightarrow \mathbb{R}$  are called regressors (e.g.,  $r_1(x) = 1, r_2(x) = x, r_3(x) = x^2 \dots \rightarrow$  polynomial model)

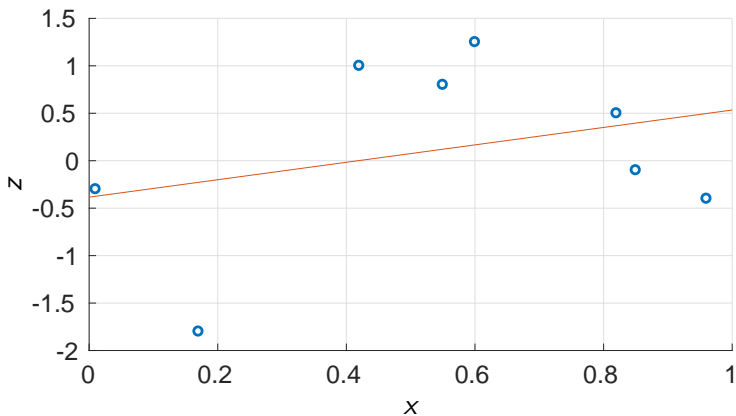
- ▶ Most classical method to obtain a good value of  $b$ :  
**least squares**  $\rightarrow$  minimize the sum of squared errors

$$J(\theta) = \sum_{i=1}^n \left( z_i - \hat{f}_n(x_i) \right)^2 = \sum_{i=1}^n \left( z_i - h(x_i; \theta) \right)^2$$



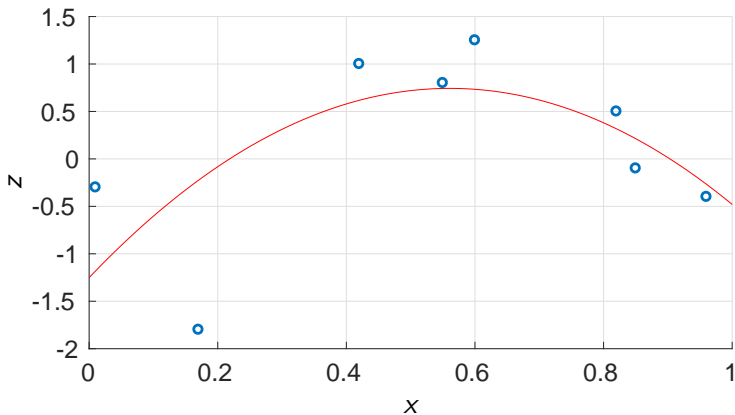
# A simple curve fitting problem

## ► Linear fit



# A simple curve fitting problem

## ► Quadratic fit



## A simple curve fitting problem

- ▶ Poor fit!
- ▶ Why? Model capacity is weak
- ▶ Now, as an example, consider the model

$$\hat{f}_n(x) = \theta^t r(x)$$

with

$$r(x) = (1 \cos(2\pi x) \sin(2\pi x) \dots \cos(2m\pi x) \sin(2m\pi x))^t \in \mathbb{R}^{2m+1}$$

(a truncated Fourier series)

- ▶ The increase in the number of parameters yields an **ill-defined** problem ( $l \gg n$ )

## A simple curve fitting problem

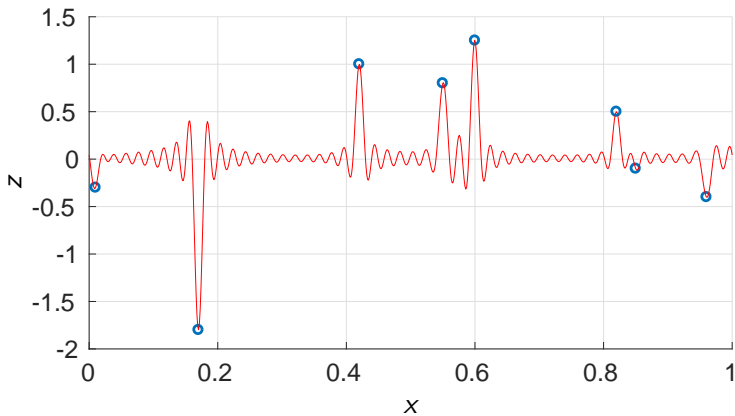
- ▶ When the problem becomes ill-defined (as capacity increases), a classical solution for finding a good value of  $b$  is to minimize the sum of an approximation error and a **regularization** term:

$$J(\theta) = \sum_{i=1}^n (z_i - \theta^t r(x_i))^2 + C \|\theta\|_2^2, \quad C > 0$$

- ▶  $\|\theta\|_2^2$  penalizes vectors  $\theta$  with large elements
- ▶  $C$  strikes a balance between regularization and data fidelity
- ▶ This approach is known as **Tikhonov regularization** (Tikhonov & Arsenin, 1977) → at the basis of numerous approximation methods (ridge regression, splines, RBF, SVM. . .)

## A simple curve fitting problem

- ▶  $n = 8, m = 50, l = 101, C = 10^{-8}$



## A simple curve fitting problem

- ▶ The regularization principle alone is not enough to obtain a good approximation
- ▶ As modeling capacity increases, **overfitting** may arise

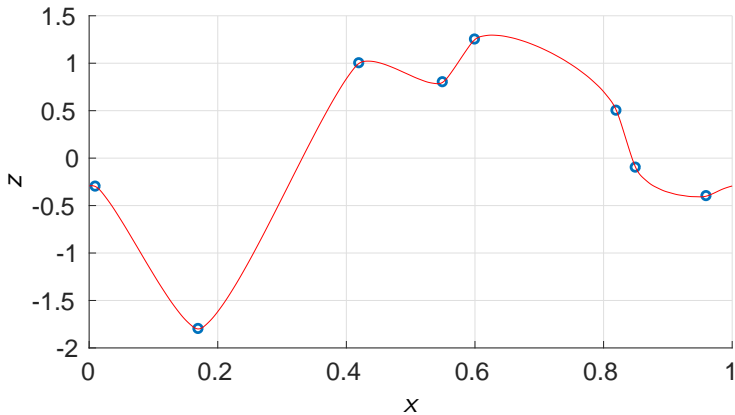
## A simple curve fitting problem

- ▶ To avoid overfitting, we should try a regularization that penalizes high frequencies more
- ▶ For instance, take

$$\|\theta\| = \theta_1^2 + \sum_{k=1}^m \frac{\theta_{2k}^2 + \theta_{2k+1}^2}{(1 + (2k\pi)^\alpha)^2}$$

## A simple curve fitting problem

- ▶  $n = 8, m = 50, l = 101, C = 10^{-8}, \alpha = 1.3$





## A simple curve fitting problem

- ▶ From this example, we can see that the construction of a regularization scheme should result from a procedure that takes into account the data (using cross-validation, for instance) an/or prior knowledge (high frequencies  $\ll$  low frequencies, for instance)

## Black-box modeling

- ▶ A large number of methods are available in the literature: polynomial regression, splines, NN, RBF...
- ▶ All methods are based on mixing prior information and regularization principles
- ▶ Instances of “regularization” in regression:
  - ▶ t-tests, F-tests, ANOVA, AIC (Akaike info criterion)... in linear regression
  - ▶ Early stopping in NN
  - ▶ Regularized reproducing-kernel regression
    - ▶ 1960 : splines, (Schoenberg 1964, Duchon 1976–1979)
    - ▶ 1970 : ridge regression (Hoerl, Kennard)
    - ▶ 1980 : RBF, (Micchelli 1986, Powel 1987)
    - ▶ 1995 : SVM, (Vapnik 1995)
    - ▶ 1997 : SVR, (Smola 1997) & semi-param SVR (Smola 1999)

# Black-box modeling

- ▶ How to choose a regularization scheme?
- ▶ The **Bayesian setting** is a principled approach that makes it possible to construct regularized regressions

## Lecture 1 : From meta-models to UQ

1.1 Introduction

1.2 Black-box modeling

**1.3 Bayesian approach**

1.4 Posterior distribution of a quantity of interest

1.5 Complements on Gaussian processes

# Why a Bayesian approach?

- ▶ Objective: infer properties about  $f : \mathbb{X} \rightarrow \mathbb{R}$  through pointwise evaluations
- ▶ Why a Bayesian approach?
  - ▶ a principled approach to choose a regularization scheme according to prior information
  - ▶ through probability calculus and/or Monte Carlo simulations, the user can infer properties about the unknown function
  - ▶ for instance: given prior knowledge and  $\mathcal{D}_n$ , what is the probability that the global maximum of  $f$  is greater than a given threshold  $u \in \mathbb{R}$ ?
- ▶ **Main idea:** use a **statistical model of the observations**, together with a **probability model for the parameter** of the statistical model

## Some reminders about probabilities

- ▶ Recall that a **random variable** is a function that maps a sample space  $\Omega$  to an outcome space  $E$  (e.g.  $E = \mathbb{R}$ ), and that assigns probabilities (weights) to possible outcomes

### Def.

Formally, let  $(\Omega, \mathcal{A}, P)$  be a probability space, and  $(E, \mathcal{E})$  be a measurable outcome space

→ a random variable  $X$  is a measurable function  $(\Omega, \mathcal{A}, P) \rightarrow (E, \mathcal{E})$

- ▶  $X$  is used to assign probabilities to events: for instance

$$P(X \in [0, 1]) = P^X([0, 1]) = 1/2$$

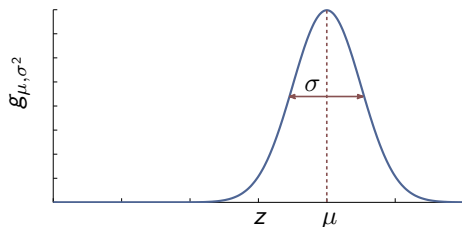
- ▶ Case of a random variable with a density

$$P(X \in [a, b]) = P^X([a, b]) = \int_a^b p^X(x) dx$$

# Gaussian random variables

- ▶ A real-valued random variable  $Z$  is said to be **Gaussian**  $\mathcal{N}(\mu, \sigma^2)$ , if it has the continuous **probability density function**

$$g_{\mu, \sigma^2}(z) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2} \frac{(z - \mu)^2}{\sigma^2}\right)$$



# Gaussian random variables

- ▶ The **mean** of  $Z$  (also called **expectation** or first-order moment) is

$$E(Z) = \int_{\mathbb{R}} z g_{\mu, \sigma^2}(z) dz = \mu$$

and its **second-order** moment is defined as

$$E(Z^2) = \int_{\mathbb{R}} z^2 g_{\mu, \sigma^2}(z) dz = \sigma^2 + \mu^2$$

- ▶ The **variance** of  $Z$  is defined as

$$\text{var}(Z) = E[(Z - E(Z))^2] = E[Z^2] - E[Z]^2 = \sigma^2$$

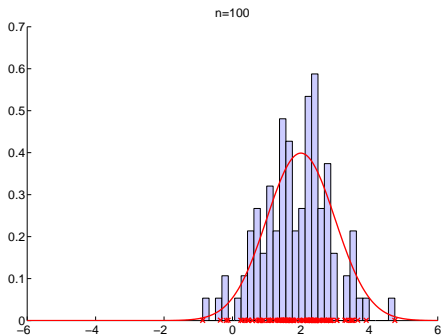


# Gaussian random variables

- ▶ A Gaussian variable  $Z$  can be used as a **stochastic model** of some uncertain real-valued quantity
- ▶ In other words,  $Z$  can be thought as a **prior about some uncertain quantity** of interest

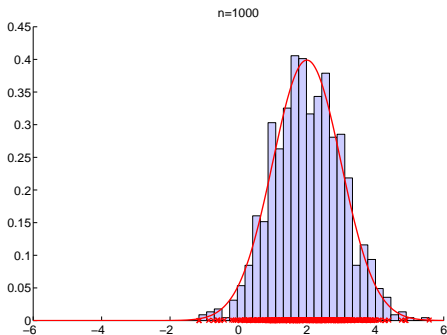
# Gaussian random variables

- ▶ Using a random generator, it is possible to “generate” sample values  $z_1, z_2, \dots$  of our model  $Z \rightarrow$  possible values for our **uncertain quantity of interest**



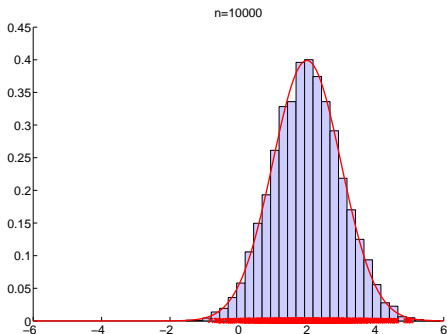
# Gaussian random variables

- ▶ Using a random generator, it is possible to “generate” sample values  $z_1, z_2, \dots$  of our model  $Z \rightarrow$  possible values for our **uncertain quantity of interest**



## Gaussian random variables

- ▶ Using a random generator, it is possible to “generate” sample values  $z_1, z_2, \dots$  of our model  $Z \rightarrow$  possible values for our **uncertain quantity of interest**



(as  $n \rightarrow \infty$ , the empirical distribution of the realizations tends to the normal distribution)

# Bayesian model

Formally, recall that a **statistical model** is a triplet  $\mathcal{M} = (\mathcal{Z}, \mathcal{F}, \mathcal{P})$

- ▶  $\mathcal{Z} \rightarrow$  observation space (typically,  $\mathcal{Z} = \mathbb{R}^n$ )
- ▶  $\mathcal{F} \rightarrow$   $\sigma$ -algebra on  $\mathcal{Z}$
- ▶  $\mathcal{P} \rightarrow$  parametric family  $\{P_\theta; \theta \in \Theta\}$  of probability distributions on  $(\mathcal{Z}, \mathcal{F})$

Def.

A **Bayesian model** is defined by the specification of

- ▶ a **parametric statistical model**  $\mathcal{M}$  (model of the observations)
- ▶ a **prior probability distribution**  $\Pi$  on  $(\Theta, \Xi) \rightarrow$  probability model that describes uncertainty about  $\theta$  before an observation is made

## Example

- ▶ Suppose we repeat measurements of a quantity of interest:  
 $z_1, z_2, \dots \in \mathbb{R}$

- ▶ **Model of observations:**  $Z_i \stackrel{iid}{\sim} \mathcal{N}(\theta_1, \theta_2), i = 1, \dots, n$

- ▶ The statistical model can formally be written as the triplet

$$\mathcal{M} = \left( \mathbb{R}^n, \mathcal{B}(\mathbb{R}^n), \{ \mathcal{N}(\theta_1, \theta_2)^{\otimes n} \}_{\theta_1, \theta_2} \right)$$

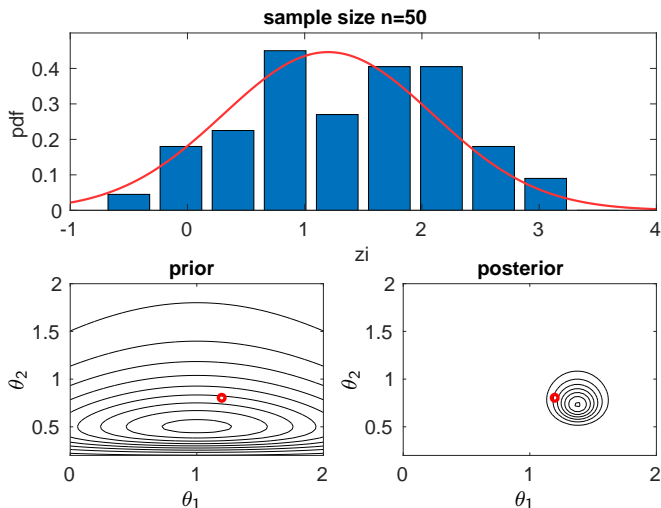
- ▶ Moreover, if we assume a **prior distribution** about  $\theta_1$  and  $\theta_2$   
(e.g.  $\theta_1 \sim \mathcal{N}(1, 1), \theta_2 \sim \mathcal{IG}(3, 2)$ ), we obtain a

**Bayesian model**

- ▶ From this Bayesian model (model of observations + prior), we can compute the **posterior distribution** of  $(\theta_1, \theta_2)$  given  $Z_1, \dots, Z_n$  (will be explained later)

# Example

$$n = 50, \theta_1 = 1.2, \theta_2 = 0.8$$



# Simple curve fitting problem from a Bayesian approach

- ▶ Recall our simple curve fitting model

$$\hat{f}_n(x) = \theta^t r(x)$$

with

$$r(x) = (1 \cos(2\pi x) \sin(2\pi x) \dots \cos(2m\pi x) \sin(2m\pi x))^t \in \mathbb{R}^{2m+1}$$

- ▶ Bayesian model?



- ▶ Assume the following statistical model for the observations:

$$\begin{cases} Z_i = \xi(x_i) + \varepsilon_i, & i = 1, \dots, n \\ \xi(x) = \theta^t r(x), & x \in \mathbb{X} \\ \varepsilon_i \stackrel{iid}{\sim} \mathcal{N}(0, \sigma_\varepsilon^2) \end{cases}$$

or equivalently,

$$Z_i \stackrel{iid}{\sim} \mathcal{N}(\theta^t r(x_i), \sigma_\varepsilon^2), \quad i = 1, \dots, n$$

- ▶ Moreover, choose a prior distribution for  $\theta$ :

$$\theta_j \stackrel{\text{indep}}{\sim} \mathcal{N}(0, \sigma_{\theta_j}^2), \quad j = 1, \dots, 2m + 1$$

- ▶ The rvs  $Z_i$  constitute a Bayesian model of the observations
- ▶  $\xi$  is a random function / random process  $\rightarrow$  prior about  $f$

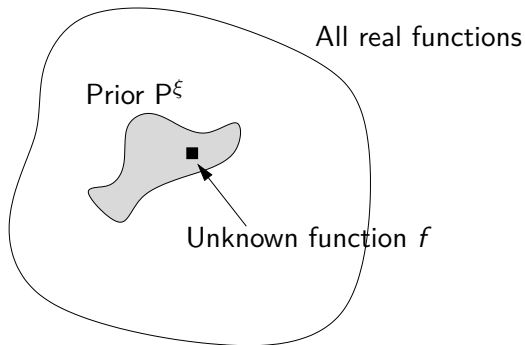
# Random process

Def.

A **random process**  $\xi : (\Omega, \mathbb{X}) \rightarrow \mathbb{R}$  is a collection of random variables  $\xi(\cdot, x) : \Omega \rightarrow \mathbb{R}$  indexed by  $x \in \mathbb{X}$

- ▶ Random processes can be viewed as a generalization of random vectors
- ▶ For a fixed  $\omega \in \Omega$ , the function  $\xi(\omega, \cdot) : \mathbb{X} \rightarrow \mathbb{R}$  is called a **sample path**

- ▶ In our Bayesian setting, we can say that
  - ▶ we use a random process  $\xi$  as a **stochastic model** of the **unknown** function  $f$
  - ▶  $f$  is viewed as as **sample paths** of  $\xi$
  - ▶  $\xi$  represents **our knowledge about  $f$**  before any evaluation has been made
  - ▶ the distribution  $\Pi = P^\xi$  is a **prior** about  $f$

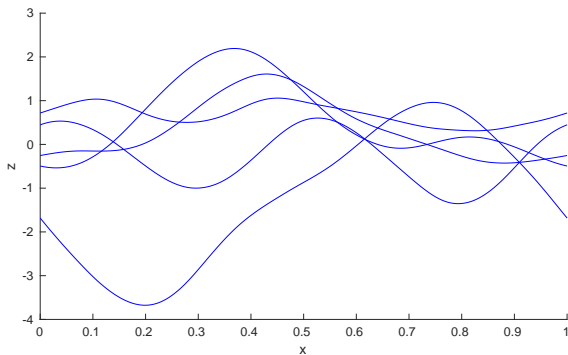


- ▶ Here,  $\xi(\omega, \cdot) = \theta(\omega)^t r(\cdot)$
- ▶ Fixing  $\omega$  (a sample path) amounts to “choosing” a value for the random vector  $\theta$

- ▶ Example of sample paths with

$$\begin{cases} \theta_1 \sim \mathcal{N}(0, 1) \\ \theta_{2k}, \theta_{2k+1} \stackrel{\text{indep}}{\sim} \mathcal{N}\left(0, \frac{1}{1+(\omega_0 k)^\alpha}\right), \quad k = 1, \dots, m \end{cases}$$

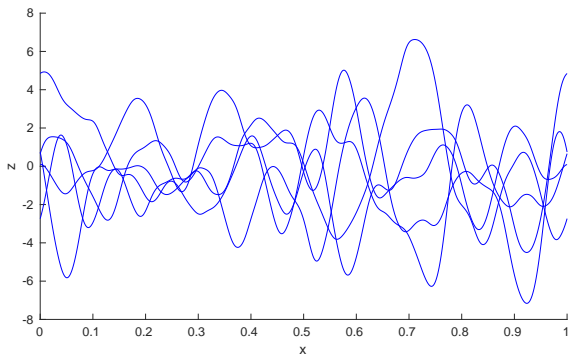
with  $\omega_0 = \frac{2\pi}{10}$ ,  $\alpha = 4$



- ▶ Example of sample paths with

$$\begin{cases} \theta_1 \sim \mathcal{N}(0, 1) \\ \theta_{2k}, \theta_{2k+1} \stackrel{\text{indep}}{\sim} \mathcal{N}\left(0, \frac{1}{1+(\omega_0 k)^\alpha}\right), \quad k = 1, \dots, m \end{cases}$$

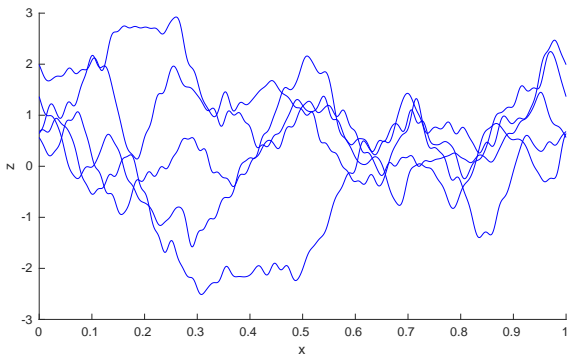
with  $\omega_0 = \frac{2\pi}{50}$ ,  $\alpha = 4$



- ▶ Example of sample paths with

$$\begin{cases} \theta_1 \sim \mathcal{N}(0, 1) \\ \theta_{2k}, \theta_{2k+1} \stackrel{\text{indep}}{\sim} \mathcal{N}\left(0, \frac{1}{1+(\omega_0 k)^\alpha}\right), \quad k = 1, \dots, m \end{cases}$$

with  $\omega_0 = \frac{2\pi}{10}$ ,  $\alpha = 1.5$



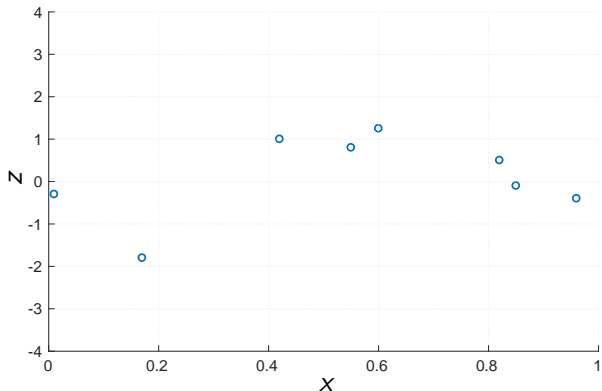
# Bayesian approach

- ▶ The **choice of a prior** in a Bayesian approach reflects the **user's knowledge about uncertain parameters**
- ▶ In the case of function approximation  $\rightarrow$  **regularity** of the function



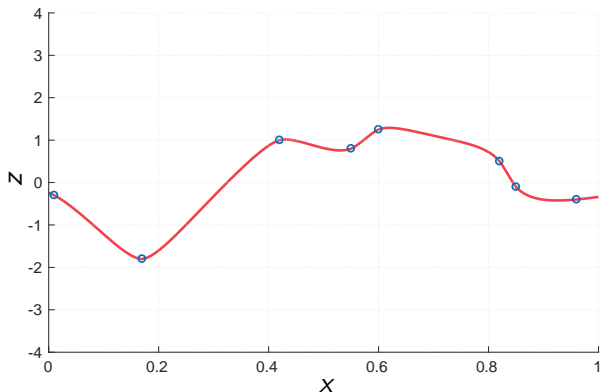
# Bayesian approach

- ▶ Where shall we go now?
- ▶ Objective: compute posterior distributions from data



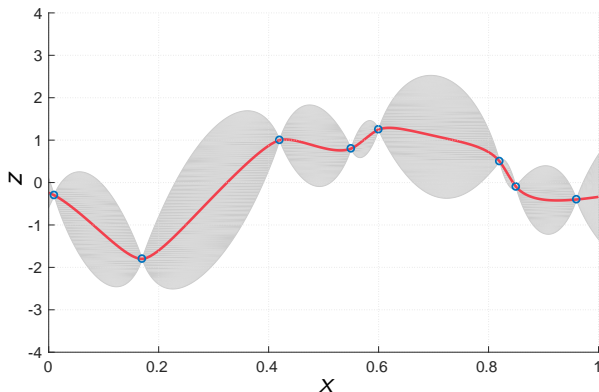
# Bayesian approach

- ▶ Where shall we go now?
- ▶ Objective: compute posterior distributions from data



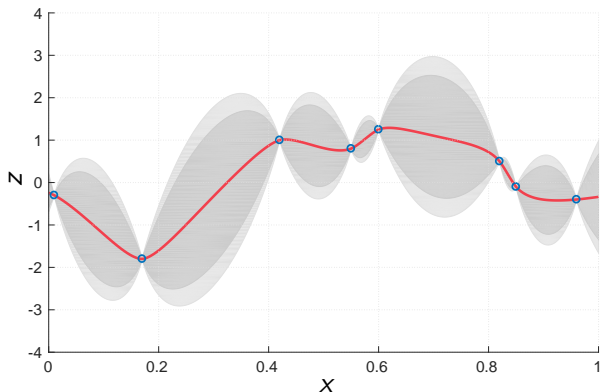
# Bayesian approach

- ▶ Where shall we go now?
- ▶ Objective: compute posterior distributions from data



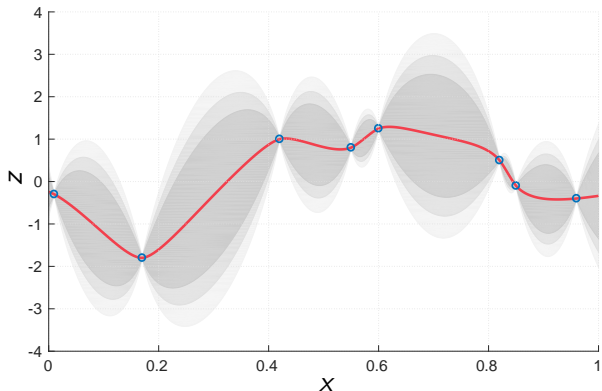
# Bayesian approach

- ▶ Where shall we go now?
- ▶ Objective: compute posterior distributions from data



# Bayesian approach

- ▶ Where shall we go now?
- ▶ Objective: compute posterior distributions from data



# A “simplification” of the Bayesian model for the simple curve fitting problem

- ▶  $\xi = \theta^t r$  (our prior about  $f$ ) is a **Gaussian process**
- ▶ Why?

# Gaussian random vectors

## Def.

A real-valued random vector  $Z = (Z_1, \dots, Z_d) \in \mathbb{R}^d$  is said to be **Gaussian** iff any linear combination of its components  $\sum_{i=1}^d a_i Z_i$ , with  $a_1, \dots, a_d \in \mathbb{R}$ , is a Gaussian variable

- ▶ A Gaussian random vector  $Z$  is characterized by its **mean** vector,  $\mu = (E[Z_1], \dots, E[Z_d]) \in \mathbb{R}^d$ , and the **covariance** of the pairs of components  $(Z_i, Z_j)$ ,  $i, j \in \{1, \dots, d\}$ ,  
 $\text{cov}(Z_i, Z_j) = E[(Z_i - E(Z_i))(Z_j - E(Z_j))]$
- ▶ If  $Z \in \mathbb{R}^d$  is a Gaussian vector with mean  $\mu \in \mathbb{R}^d$  and covariance matrix  $\Sigma \in \mathbb{R}^{d \times d}$ , we shall write  $Z \sim \mathcal{N}(\mu, \Sigma)$

- ▶ Exercise: Let  $Z \sim \mathcal{N}(\mu, \Sigma)$ . Determine  $E\left(\sum_{i=1}^d a_i Z_i\right)$  and  $\text{var}\left(\sum_{i=1}^d a_i Z_i\right)$

- ▶ The correlation coefficient of two components  $Z_i$  and  $Z_j$  of  $Z$  is defined by

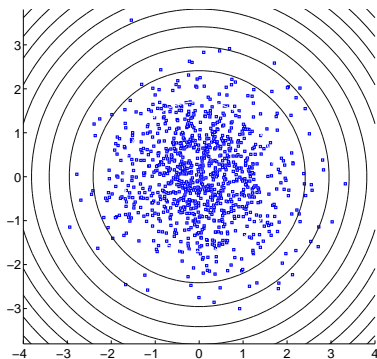
$$\rho(Z_i, Z_j) = \frac{\text{cov}(Z_i, Z_j)}{\sqrt{\text{var}(Z_i)\text{var}(Z_j)}} \in [-1, 1],$$

→ measures the similarity between  $Z_i$  and  $Z_j$

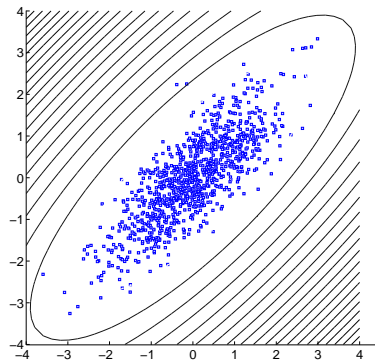


# Gaussian random vectors: correlation

$$\rho = 0$$



$$\rho = 0.8$$



# Gaussian random processes

- ▶ Recall that a random process is a set  $\xi = \{\xi(x), x \in \mathbb{X}\}$  of random variables indexed by the elements of  $\mathbb{X}$
- ▶ Gaussian random process
  - generalization of a Gaussian random vector

Def.

$\xi$  is a **Gaussian random process** iff,  $\forall n \in \mathbb{N}$ ,  $\forall x_1, \dots, x_n \in \mathbb{X}$ , and  $\forall a_1, \dots, a_n \in \mathbb{R}$ , the real-valued random variable

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

is Gaussian

# Application

- ▶ If

$$\begin{cases} \xi(x) = \theta^t r(x), & x \in \mathbb{X} \\ \theta_j \stackrel{\text{indep}}{\sim} \mathcal{N}(0, \sigma_{\theta_j}^2), & j = 1, \dots, 2m + 1 \end{cases}$$

then,  $\forall x_1, \dots, x_n \in \mathbb{X}$ , and  $\forall a_1, \dots, a_n \in \mathbb{R}$ ,

$$\begin{aligned} \sum_{i=1}^n a_i \xi(x_i) &= \sum_i a_i \left( \sum_j \theta_j r_j(x_i) \right) \\ &= \sum_j \left( \sum_i a_i r_j(x_i) \right) \theta_j \sim \mathcal{N} \left( 0, \sum_j \left( \sum_i a_i r_j(x_i) \right)^2 \sigma_{\theta_j}^2 \right) \end{aligned}$$

- ▶ Thus,  $\xi = \theta^t r$  is a Gaussian process

# Gaussian random processes

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

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

- ▶ and its **covariance function**

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

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

▶ Exercise: determine  $E\left(\sum_{i=1}^d a_i \xi(x_i)\right)$  and  $\text{var}\left(\sum_{i=1}^d a_i \xi(x_i)\right)$

▶ What is the distribution of  $\sum a_i \xi(x_i)$ ?

→ The distribution of a linear combination of a Gaussian process  $\mathcal{GP}(m, k)$  can be **simply obtained as a function of  $m$  and  $k$**

# Application

► If

$$\begin{cases} \xi(x) = \theta^t r(x), & x \in \mathbb{X} \\ \theta_j \stackrel{\text{indep}}{\sim} \mathcal{N}(0, \sigma_{\theta_j}^2), & j = 1, \dots, 2m + 1 \end{cases}$$

then,

$$\xi \sim \mathcal{GP}(0, k)$$

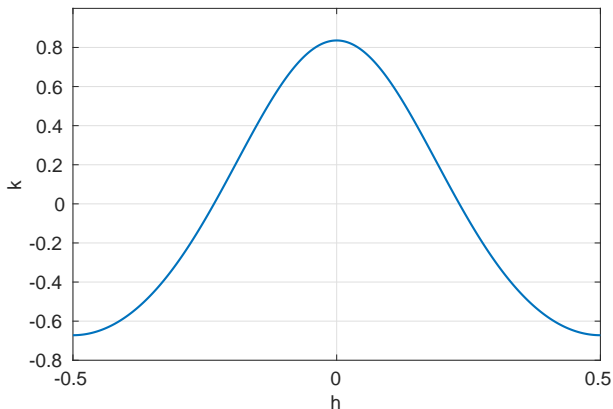
with

$$k : (x, y) \mapsto \sum_j \sigma_{\theta_j}^2 r_j(x) r_j(y)$$

- Covariance function corresponding to

$$\begin{cases} \theta_1 \sim \mathcal{N}(0, 1) \\ \theta_{2k}, \theta_{2k+1} \stackrel{\text{indep}}{\sim} \mathcal{N}\left(0, \frac{1}{1+(\omega_0 k)^\alpha}\right), \quad k = 1, \dots, m \end{cases}$$

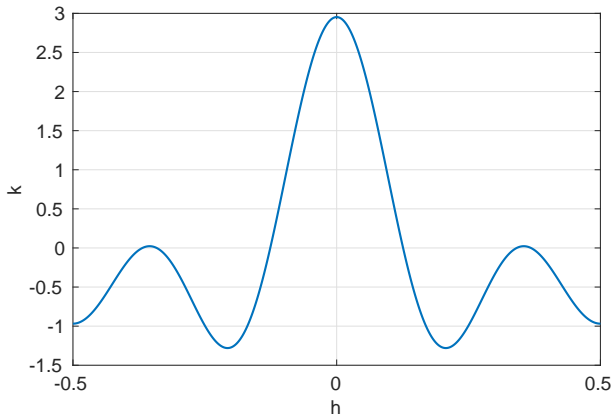
with  $\omega_0 = \frac{2\pi}{10}$ ,  $\alpha = 4$



- Covariance function corresponding to

$$\begin{cases} \theta_1 \sim \mathcal{N}(0, 1) \\ \theta_{2k}, \theta_{2k+1} \stackrel{\text{indep}}{\sim} \mathcal{N}\left(0, \frac{1}{1+(\omega_0 k)^\alpha}\right), \quad k = 1, \dots, m \end{cases}$$

with  $\omega_0 = \frac{2\pi}{50}$ ,  $\alpha = 4$





# The covariance of a Gaussian random process

- ▶ Main properties: a covariance function  $k$  is
  - ▶ **symmetric**:  $\forall x, y \in \mathbb{X}, k(x, y) = k(y, x)$
  - ▶ **positive**:

$$\forall n \in \mathbb{N}, \forall x_1, \dots, x_n \in \mathbb{X}, \forall a_1, \dots, a_n \in \mathbb{R}, \sum_{i,j=1}^n a_i k(x_i, x_j) a_j \geq 0$$

- ▶ In the following, we shall assume that the covariance of  $\xi \sim \mathcal{GP}(m, k)$  is **invariant under translations**, or **stationary**:

$$k(x + h, y + h) = k(x, y), \quad \forall x, y, h \in \mathbb{X}$$

- ▶ When  $k$  is stationary, there exists a **stationary covariance**  $k_{\text{sta}} : \mathbb{R}^d \rightarrow \mathbb{R}$  such that

$$\text{cov}(\xi(x), \xi(y)) = k(x, y) = k_{\text{sta}}(x - y)$$

## Stationary covariances

- ▶ When  $k$  is stationary, the variance

$$\text{var}(\xi(x)) = \text{cov}(\xi(x), \xi(x)) = k(0)$$

does not depend on  $x$

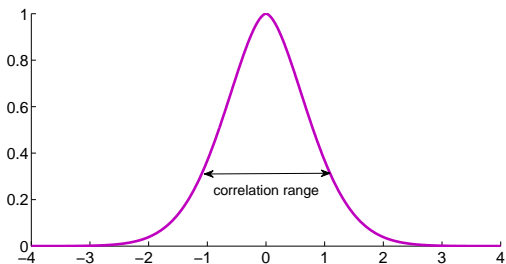
- ▶ The covariance function can be written as

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

with  $\sigma^2 = \text{var}(\xi(x))$ , and where  $\rho$  is the **correlation function** of  $\xi$ .

## Stationary covariances

- ▶ The graph of the correlation function is a symmetric “bell curve” shape



- ▶ We have, by C-S,

$$\begin{aligned}\forall h \in \mathbb{X}, |k(h)| &= |\text{cov}(\xi(x), \xi(x+h))| \\ &= |E[(\xi(x) - m(x))(\xi(x+h) - m(x+h))]| \\ &\leq E((\xi(x) - m(x))^2)^{1/2} E((\xi(x+h) - m(x+h))^2)^{1/2} \\ &= k(0)^{1/2} k(0)^{1/2} = k(0)\end{aligned}$$

- ▶ Recall, Bochner's spectral representation theorem

## Theorem

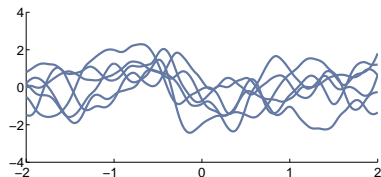
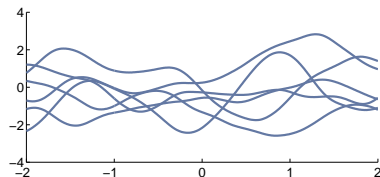
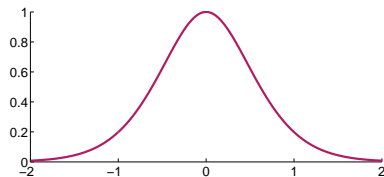
A real function  $k(h)$ ,  $h \in \mathbb{R}^d$  is symmetric positive iff it is the Fourier transform of a finite positive measure, i.e.

$$k(h) = \int_{\mathbb{R}^d} e^{i\langle u, h \rangle} d\mu(u),$$

where  $\mu$  is a finite positive measure on  $\mathbb{R}^d$ .

# Gaussian process simulation

- ▶ Using a random generator, it is possible to “generate” sample paths  $f_1, f_2, \dots$  of a Gaussian process  $\xi$



## Gaussian process simulation

How to simulate sample paths of a zero-mean Gaussian random process?

- ▶ Choose a set of points  $x_1, \dots, x_n \in \mathbb{X}$
- ▶ Denote by  $K$  the  $n \times n$  covariance matrix of the random vector  $\underline{\xi} = (\xi(x_1), \dots, \xi(x_n))^t$  (NB:  $\underline{\xi} \sim \mathcal{N}(0, K)$ )
- ▶ Consider the Cholesky factorization of  $K$

$$K = CC^t,$$

with  $C$  a lower triangular matrix (such a factorization exists since  $K$  is a sdp matrix)

- ▶ Let  $\varepsilon = (\varepsilon_1, \dots, \varepsilon_n)^t$  a Gaussian vector with  $\varepsilon_i \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$
- ▶ Then  $C\varepsilon \sim \mathcal{N}(0, K)$

# “Simplification” of the Bayesian model for the simple curve fitting problem

- ▶ Instead of choosing the model

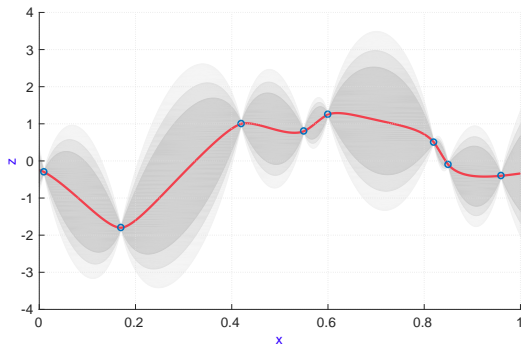
$$\begin{cases} \xi(x) = \theta^t r(x), & x \in \mathbb{X} \\ \theta_j \stackrel{\text{indep}}{\sim} \mathcal{N}(0, \sigma_{\theta_j}^2), & j = 1, \dots, 2m + 1 \end{cases}$$

simply choose a covariance function  $k$  and assume  $\xi \sim \mathcal{GP}(0, k)$

- ▶ More details about how to choose  $k$  will be given below

# Posterior

- ▶ How to compute a posterior distribution, given the Gaussian-process prior  $\xi$  and data?





# Conditional distributions

- ▶ Let  $X$  be a random variable modeling an unknown quantity of interest
- ▶ Assume we observe a random variable  $T$ , or a random vector  $T = (T_1, \dots, T_n)$
- ▶ Provided that  $T$  and  $X$  are **not independent**,  $T$  contains **information about  $X$**
- ▶ Aim: define a **notion of distribution of  $X$  “knowing”  $T$**

# Conditional probabilities

Recall the following

Def.

Let  $(\Omega, \mathcal{A}, P)$  be a probability space. Given two events  $A, B \in \mathcal{A}$  such that  $P(B) \neq 0$ , define the **probability of  $A$  given  $B$**  (or conditional on  $B$ ) by

$$P(A | B) = \frac{P(A \cap B)}{P(B)}$$

# The notion of conditional density

Def.

Assume that the pair  $(X, T) \in \mathbb{R}^2$  has a density  $p^{(X,T)}$ .

Define the **conditional density** of  $X$  given the event  $T = t$  by

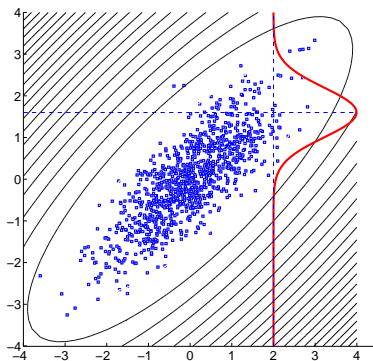
$$p^{X|T}(x | t) = \begin{cases} \frac{p^{(X,T)}(x, t)}{p^T(t)} = \frac{p^{(X,T)}(x, t)}{\int p^{X,T}(x, t) dx} & \text{if } p^T(t) > 0 \\ \text{arbitrary density} & \text{if } p^T(t) = 0. \end{cases}$$

# Application

- ▶ Given a Gaussian random vector  $Z = (Z_1, Z_2) \sim \mathcal{N}(0, \Sigma)$ , what is the distribution of  $Z_2$  “knowing”  $Z_1$ ?
- ▶ Define

$$p^{Z_2|Z_1}(z_2|z_1) = \frac{p^{(Z_2, Z_1)}(z_2, z_1)}{p^{Z_1}(z_1)} = \frac{\sigma_1}{(2\pi)^{1/2}(\det \Sigma)^{1/2}} \exp(-1/2(z^t \Sigma^{-1} z - z_1^2 / \sigma_1^2))$$

→ Gaussian distribution!



- ▶ The random variable denoted by  $Z_2 | Z_1$  with density  $p^{Z_2|Z_1}(\cdot | Z_1)$  represents the **residual uncertainty** about  $Z_2$  when  $Z_1$  has been observed
- ▶ High correlation → small residual uncertainty

## Conditional mean/expectation

- A fundamental notion: conditional expectation

Def.

Assume that the pair  $(X, T)$  has a density  $p^{(X, T)}$ . Define the **conditional mean of  $X$  given  $T = t$**  as

$$E(X | T = t) \triangleq \int_{\mathbb{R}} x p^{X|T}(x|t) dx = h(t)$$

Def.

The random variable  $E(X | T) = h(T)$  is called the **conditional expectation of  $X$  given  $T$**

# Conditional expectation

- ▶ Why conditional expectation is an fundamental notion?
- ▶ We have the following

## Theorem

*Under the previous assumptions, the solution of the problem*

$$\hat{X} = \operatorname{argmin}_Y E[(X - Y)^2]$$

*where the **minimum is taken over all functions of  $T$**  is given by*

$$\hat{X} = E(X | T)$$

- ▶ In other words,  $E(X | T)$  is the **best approximation** (in the sense of the quadratic mean) **of  $X$  by a function of  $T$**

## Important properties of conditional expectation

- (1)  $E(X | T)$  is a random variable depending on  $T \rightarrow$  there exists a function  $h$  such that  $E(X | T) = h(T)$
- (2) The operator  $\pi_H : X \mapsto E(X | T)$  is a (linear) operator of **orthogonal projection** onto the space of all functions of  $T$  (for the inner product  $X, Y \mapsto (X, Y) = E(XY)$ )
- (3) Let  $X, Y, T \in L^2$ . Then
  - i)  $\forall \alpha \in \mathbb{R}, E(\alpha X + Y | T) = \alpha E(X | T) + E(Y | T)$  a.s.
  - ii)  $E(E[X | T]) = E(X)$
  - iii) If  $X \perp\!\!\!\perp T$ ,  $E(X | T) = E(X)$

## Conditional expectation: Gaussian random variables

- ▶ Recall that the space of second-order random variables  $L^2(\Omega, \mathcal{A}, P)$  endowed with the inner product  $X, Y \mapsto (X, Y) = E(XY)$  is a Hilbert space
- ▶ Gaussian linear space

Def.

A **linear subspace**  $G$  of  $L^2(\Omega, \mathcal{A}, P)$  is **Gaussian** iff  $\forall X_1, \dots, X_n \in G$  and  $\forall a_1, \dots, a_n \in \mathbb{R}$  the random variable  $\sum_i a_i X_i$  is Gaussian

- ▶ In what follows, assume that  $G$  is **centered**, i.e., each element in  $G$  is a zero-mean random variable



## Theorem (Projection theorem in centered Gaussian spaces)

Let  $G$  be a centered Gaussian space. Let  $X, T_1, \dots, T_n \in G$ . Then  $E(X \mid T_1, \dots, T_n)$  is the orthogonal projection (in  $L^2$ ) of  $X$  on  $\mathcal{T} = \text{span}\{T_1, \dots, T_n\}$ .

**proof** Let  $\widehat{X} \in G$  be the orthogonal projection of  $X$  on  $\mathcal{T}$ .

- ▶ we have  $X = \widehat{X} + \varepsilon$  where  $\varepsilon \in G$  is orthogonal to  $\mathcal{T}$ .  
In  $G$ , orthogonality  $\Leftrightarrow$  independence. Thus,  $\varepsilon \perp T_i, i = 1, \dots, n$ .
- ▶ Then,

$$\begin{aligned} E(X \mid T_1, \dots, T_n) &= E(\widehat{X} \mid T_1, \dots, T_n) + E(\varepsilon \mid T_1, \dots, T_n) \\ &= \widehat{X} + E(\varepsilon) = \widehat{X} \end{aligned}$$

The result follows. ■

## ■ Application

Let  $Z = (Z_1, Z_2)$  be a zero-mean Gaussian random vector, with covariance matrix

$$\begin{pmatrix} \sigma_1^2 & \sigma_{1,2} \\ \sigma_{2,1} & \sigma_2^2 \end{pmatrix}$$

Then  $E(Z_1 | Z_2)$  is the orthogonal projection of  $Z_1$  onto  $Z_2$ . Thus

$$E(Z_1 | Z_2) = \lambda Z_2$$

with

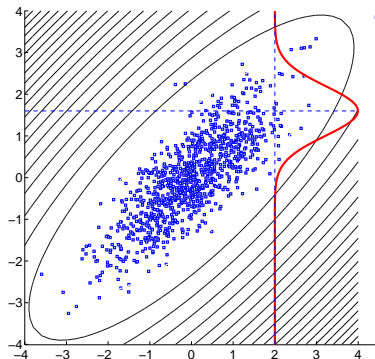
$$(Z_1 - \lambda Z_2, Z_2) = (Z_1, Z_2) - \lambda(Z_2, Z_2) = 0.$$

Hence,

$$\lambda = \frac{\sigma_{1,2}}{\sigma_2^2}$$

# Application

- ▶ Let  $Z = (Z_1, Z_2) \sim \mathcal{N}(0, \Sigma)$  as above  $\rightarrow$  recall that the cond. distrib. of  $Z_2$  given  $Z_1$  is a Gaussian distribution
- ▶ Hence  $Z_2 | Z_1 \sim \mathcal{N}(\mu(Z_1), \sigma(Z_1)^2) \rightarrow \mu ? \sigma ?$



- ▶  $\mu = E(Z_2 | Z_1) = \lambda Z_1$  with  $\lambda = \frac{\sigma_{1,2}}{\sigma_1^2}$
- ▶ Using the property of the orthogonal projection:

$$\begin{aligned}\sigma^2 &= E((Z_2 - E(Z_2 | Z_1))^2 | Z_1) \\ &\stackrel{\perp}{=} E((Z_2 - E(Z_2 | Z_1))^2) \\ &= E((Z_2 - \lambda Z_1)^2) \\ &\stackrel{\perp}{=} E((Z_2 - \lambda Z_1)Z_2) \\ &= \sigma_2^2 - \lambda\sigma_{1,2}\end{aligned}$$

## Generalization

Exercise: Let  $(Z_0, Z_1, \dots, Z_n)$  be a centered Gaussian vector.  
Determine  $E(Z_0 \mid Z_1, \dots, Z_n)$ .

## Application: computation of the posterior distrib. of a GP

- ▶ Let  $\xi \sim \mathcal{GP}(0, k)$
- ▶ Assume we observe  $\xi$  at  $x_1, \dots, x_n \in \mathbb{X}$
- ▶ Given  $x_0 \in \mathbb{X}$ , what is the conditional distrib.—or the posterior distrib. in our Bayesian framework—of

$$\xi(x_0) \mid \xi(x_1), \dots, \xi(x_n) ?$$

- ▶ More generally, what is the **posterior distribution** of the random process

$$\xi(\cdot) \mid \xi(x_1), \dots, \xi(x_n) ?$$

# Computation of the posterior distribution of a GP

## Prop.

Let  $\xi \sim \mathcal{GP}(0, k)$ . The random process  $\xi$  conditioned on  $F_n = \{\xi(x_1), \dots, \xi(x_n)\}$ , denoted by  $\xi | F_n$ , is a **Gaussian process** with

- mean  $\hat{\xi}_n : x \mapsto \mathbb{E}(\xi(x) | \xi(x_1), \dots, \xi(x_n))$
- covariance  $k_n : x, y \mapsto \mathbb{E}((\xi(x) - \hat{\xi}_n(x))(\xi(y) - \hat{\xi}_n(y)))$

## Computation of the posterior distrib. of a GP

- ▶ By property of the conditional expectation in Gaussian spaces, for all  $x \in \mathbb{X}$ ,  $\hat{\xi}_n(x)$  is a linear combination of  $\xi(x_1), \dots, \xi(x_n)$ :

$$\hat{\xi}_n(x) := \sum_{i=1}^n \lambda_i(x) \xi(x_i)$$

- ▶ Moreover, the posterior mean  $\hat{\xi}_n(x)$  is the orthogonal projection of  $\xi(x)$  onto  $\text{span}\{\xi(x_i), i = 1, \dots, n\}$ , such that
  - ▶  $\hat{\xi}_n(x) = \operatorname{argmin}_Y E[(\xi(x) - Y)^2] \rightarrow$  the variance of the prediction error is minimum
  - ▶  $E(\hat{\xi}_n(x)) = E[E(\xi(x) | F_n)] = E(\xi(x)) = 0 \rightarrow$  unbiased estimation
- ▶  $\hat{\xi}_n(x)$  is the **best linear predictor** (BLP) of  $\xi(x)$  from  $\xi(x_1), \dots, \xi(x_n)$ , also called the **kriging predictor** of  $\xi(x)$

# Computation of the posterior distrib. of a GP

- ▶ The posterior covariance, also called **kriging covariance**, is given by

$$\begin{aligned}k_n(x, y) &:= \text{cov}(\xi(x) - \hat{\xi}_n(x), \xi(y) - \hat{\xi}_n(y)) \\ &= k(x - y) - \sum_i \lambda_i(x) k(y - x_i).\end{aligned}$$

- ▶  $k_n$  is the covariance function of the **error of prediction**
- ▶ The posterior variance of  $\xi$ , also called the **kriging variance**, is defined as

$$\sigma_n^2(x) = \text{var}(\xi(x) - \hat{\xi}_n(x)) = k_n(x, x)$$

- ▶  $\sigma_n^2(x)$  is the variance of the **error of prediction**



# Kriging equations

- ▶ How to compute the weights  $\lambda_i(x)$  of the posterior mean/kriging predictor?
- ▶ Weights  $\lambda_i(x)$  are solutions of a **system of linear equations**

$$K\lambda(x) = k(x)$$

with

- $\lambda(x) = (\lambda_1(x), \dots, \lambda_n(x))^t$
- $K$ :  $n \times n$  covariance matrix of the observation vector
- $k(x)$ :  $n \times 1$  vector with entries  $k(x_i, x)$

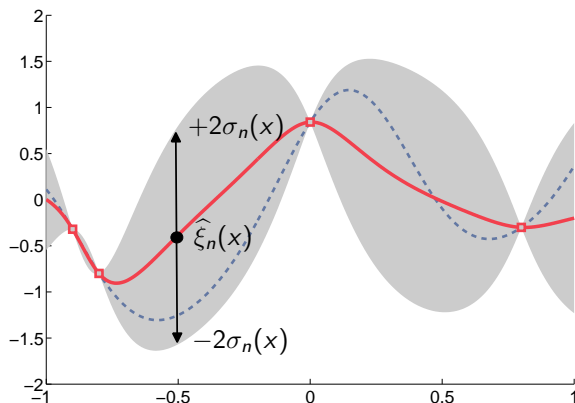
# Kriging equations

**proof**



## Posterior distrib. of a GP

- ▶ For all  $x \in \mathbb{X}$ , the random variable  $\xi(x) \mid F_n$  with distrib.  $\mathcal{N}(\hat{\xi}_n(x), \sigma_n^2(x))$  represents the **residual uncertainty** about  $\xi(x)$  when  $\xi(x_1), \dots, \xi(x_n)$  are observed



# Software for kriging/GP regression

- ▶ R packages

- ▶ **BACCO** Bayesian analysis of computer code software
- ▶ **fanovaGraph** Building Kriging Models from FANOVA Graphs
- ▶ **DiceKriging** **DiceOptim** **GPareto** Dice and ReDice packages
- ▶ **MuFiCokriging** Multi-Fidelity Cokriging models
- ▶ **RobustInv** Robust inversion based on GP (like KrigInv)
- ▶ **tgp** Treed Gaussian processes
- ▶ ...

# Software for kriging/GP regression

## ▶ Matlab/GNU Octave

- ▶ **DACE** DACE, a matlab kriging toolbox.
- ▶ **FERUM** Finite Element Reliability using Matlab
- ▶ **GPML** Gaussian Processes for Machine Learning
- ▶ **GPstuff** GP Models for Bayesian analysis
- ▶ **scalaGAUSS** Kriging toolbox with a focus on large datasets
- ▶ **Matlab Stat & ML toolbox** GP regression from Mathworks
- ▶ **STK** Small (Matlab/GNU Octave) Toolbox for Kriging
- ▶ **SUMO** **ooDACE** Surrogate Modeling Lab
- ▶ **UQLab** Uncertainty quantification framework in Matlab

## ▶ Python

- ▶ **scikit-learn** Machine learning in Python
- ▶ **OpenTURNS** Open source lib for UQ
- ▶ **Spearmint** Bayesian optimization
- ▶ **GPy** Gaussian processes framework in Python

## Generalization: prediction from noisy observations

- ▶ Let  $\xi \sim \text{GP}(0, k)$
- ▶ For  $i = 1, \dots, n$ , we observe  $Z_i = \xi(x_i) + \varepsilon_i$  at points  $x_i$ , where the random variables  $\varepsilon_i$  model an observation noise:
  - ▶  $\varepsilon_i \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2)$
  - ▶ independent from  $\xi$
- ▶ As above, the posterior mean  $\hat{\xi}_n(x)$  of  $\xi(x)$  is obtained as the orthogonal projection of  $\xi(x)$  on the linear subspace  $\text{span}\{Z_i, i = 1, \dots, n\}$ :

$$\hat{\xi}_n(x) = \sum_{i=1}^n \lambda_i(x) Z_i$$

with  $\lambda_i(x)$  such that

$$\forall i, \quad \xi(x) - \hat{\xi}_n(x) \perp Z_i$$

## Generalization: prediction from noisy observations

- ▶ Thus,  $\forall i$

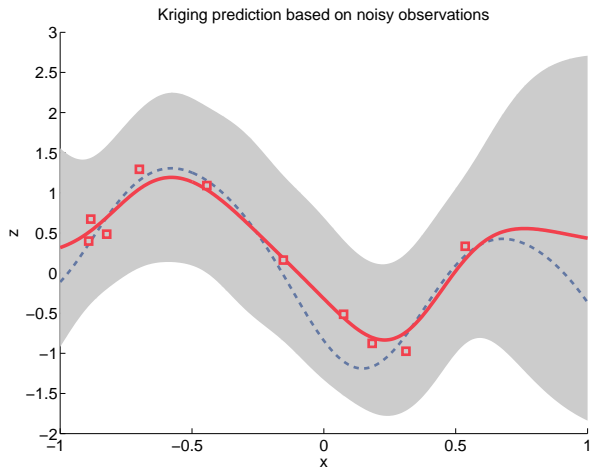
$$\mathbb{E}[(\xi(x) - \hat{\xi}_n(x))Z_i]$$

$$= \mathbb{E}[\xi(x)(\xi(x_i) + \varepsilon_i)] - \sum_{j=1}^n \lambda_j(x) \mathbb{E}[(\xi(x_j) + \varepsilon_j)(\xi(x_i) + \varepsilon_i)]$$

$$= k(x, x_i) - \sum_{j=1}^n \lambda_j(x) (k(x_j, x_i) + \sigma^2 \delta_{i,j})$$

- ▶ Under matrix form (exercise):

# Generalization: prediction from noisy observations





## Generalization: prediction from noisy observations

We should use an **approximation instead of an interpolation** in three cases:

- i) The **observations are noisy** (obviously): the computer code is stochastic (for instance, Monte Carlo is used) and running the code twice does not produce the same output
- ii) The **output of the computer code is very irregular**  $\rightarrow$  a smooth approximation is preferred
- iii) The **covariance matrix is ill-conditioned**  $\rightarrow$  adding a small observation noise will regularize the solution of the linear system (why?)

## Lecture 1 : From meta-models to UQ

1.1 Introduction

1.2 Black-box modeling

1.3 Bayesian approach

**1.4 Posterior distribution of a quantity of interest**

1.5 Complements on Gaussian processes

# Posterior of a quantity of interest

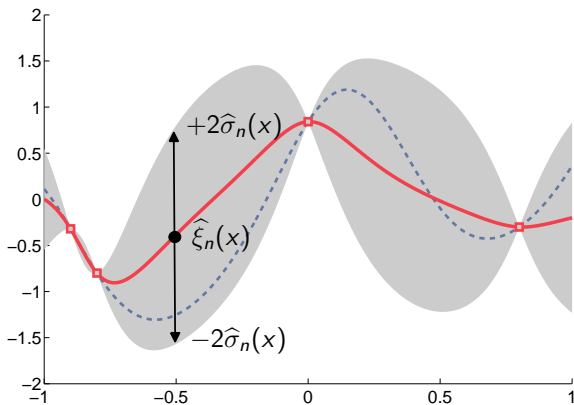
- ▶ Using the posterior distribution of  $\xi$ , we can address questions like
  - ▶ What are plausible values for  $\xi(x)$  at a given  $x$ ? (obviously)
  - ▶ What are plausible values for  $\int g(\xi(x)) d\mu(x)$  for given  $g$  and  $\mu$ ?
  - ▶ What are plausible values for the minimum  $M = \min_x \xi(x)$ ?
  - ▶ Where is the minimizer  $x^* = \operatorname{argmin}_x \xi(x)$ ?
  - ▶ What is the probability that  $\xi(x)$  exceeds a given threshold?
  - ▶ ...

## Example of a quantity of interest: the improvement

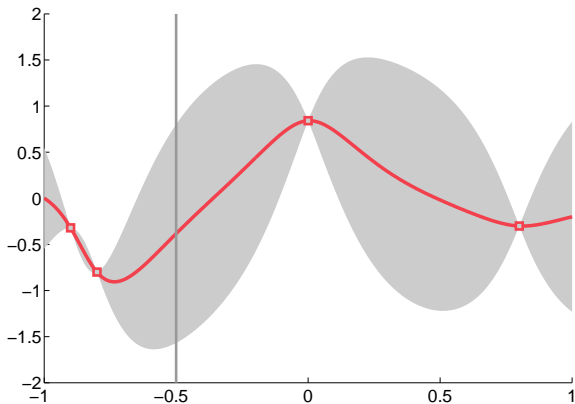
- ▶ Suppose that our objective is to **minimize** an unknown function  $f : \mathbb{X} \rightarrow \mathbb{R}$
- ▶ In our Bayesian approach, we choose a GP prior  $\xi$  for  $f$  (in other words,  $\xi$  is a model of  $f$ )
- ▶ Objective: construct a sequence  $(X_1, X_2, \dots) \in \mathbb{X}$  that converges to  $X^* = \operatorname{argmin}_x \xi(x)$
- ▶ Given  $X_1, \dots, X_{n+1}$ , how to define and choose a “good” point  $X_{n+1}$  in our setting?
- ▶ Let  $m_n = \min_{1 \leq i \leq n} \xi(X_i)$
- ▶ A “good” point  $x \in \mathbb{X}$  is such that  $m_n - \xi(x)$  is large
- ▶ Define the excursion of  $\xi$  at  $x$  below  $m_n$ , a.k.a **the improvement**:

$$I_n = \begin{cases} 0 & \text{if } \xi(x) > m_n \\ m_n - \xi(x) & \text{if } \xi(x) \leq m_n \end{cases}$$

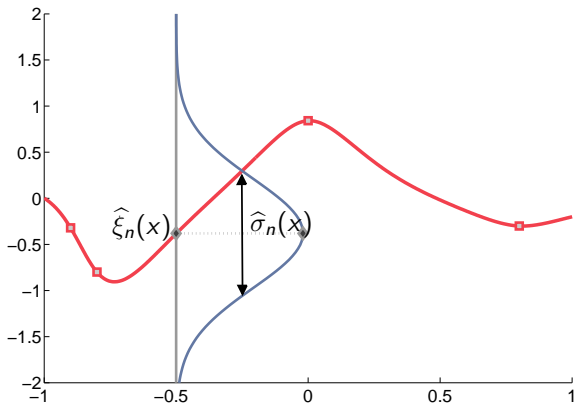
## Example of a quantity of interest: the improvement



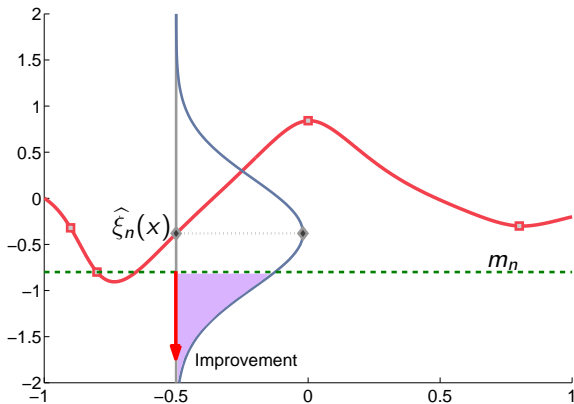
## Example of a quantity of interest: the improvement



## Example of a quantity of interest: the improvement



## Example of a quantity of interest: the improvement





## Example of a quantity of interest: the improvement

- ▶ Regions with high values of the **posterior mean** of  $I_n$  are promising search regions for the minimum of  $\xi$
- ▶ The posterior mean of  $I_n$  may be written as

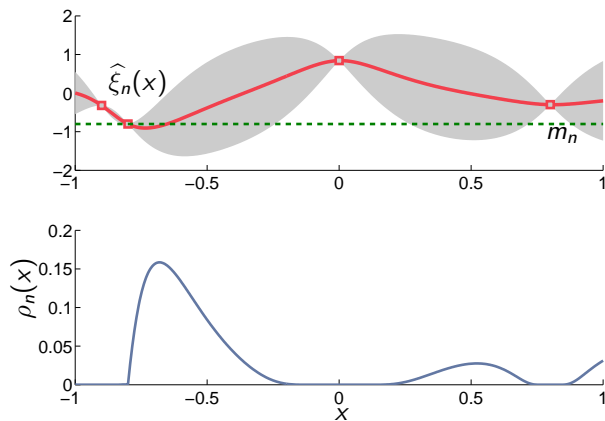
$$\begin{aligned}\rho_n(x) &= E(I_n \mid \xi(X_1), \dots, \xi(X_n)) \\ &= \int_{z=-\inf}^{m_n} (m_n - z) p^{\xi(x) \mid \xi(X_1), \dots, \xi(X_n)}(z) dz \\ &= \gamma(m_n - \hat{\xi}_n(x), \sigma_n^2(x))\end{aligned}$$

with

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

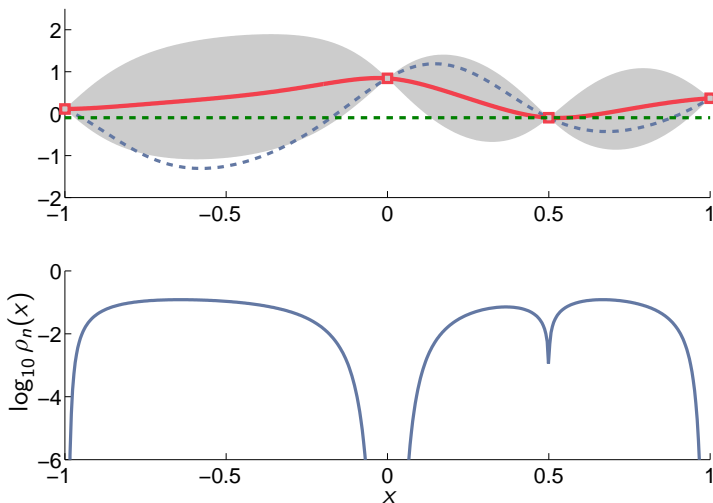
- ▶  $\rho_n$  is called the **expected improvement** [Mockus 78, Schonlau et al. 96, Jones et al. 98]

## Example of a quantity of interest: the improvement



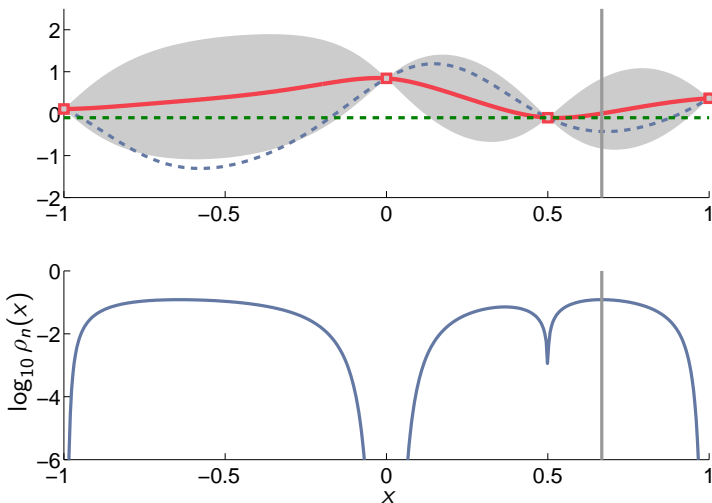
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



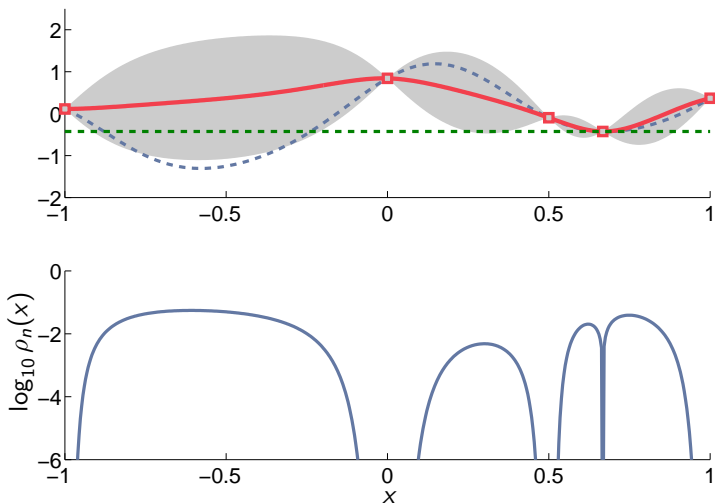
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



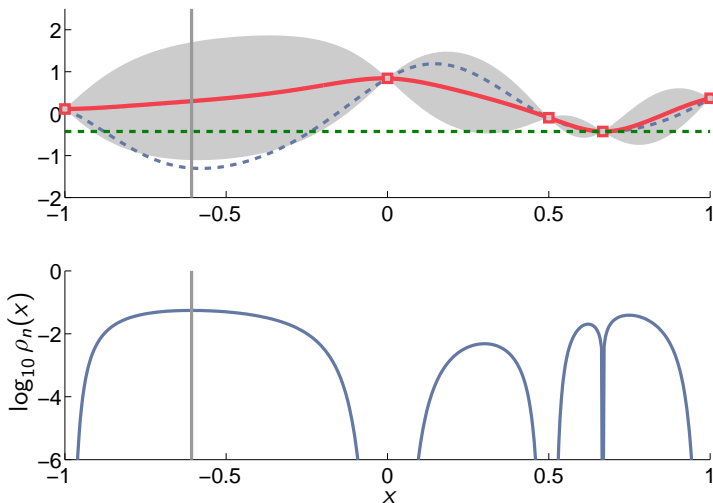
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



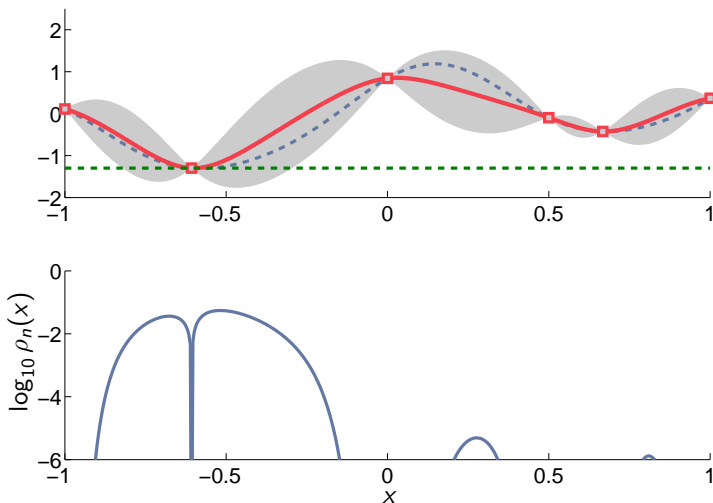
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



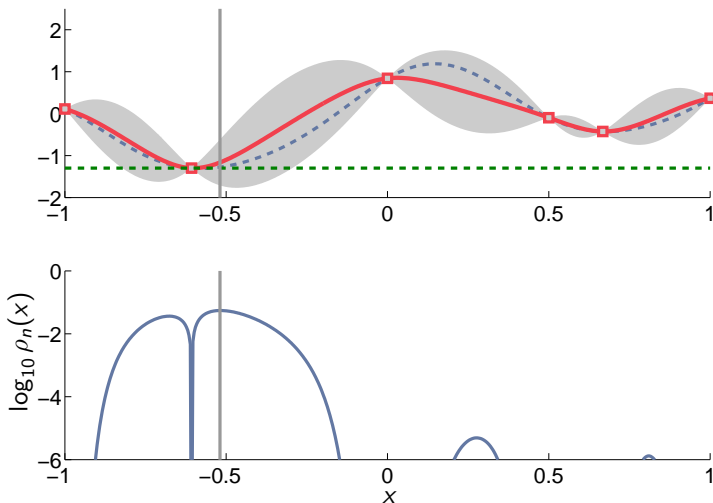
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \arg\max_x \rho_n(x)$



# Insertion into an optimization algorithm

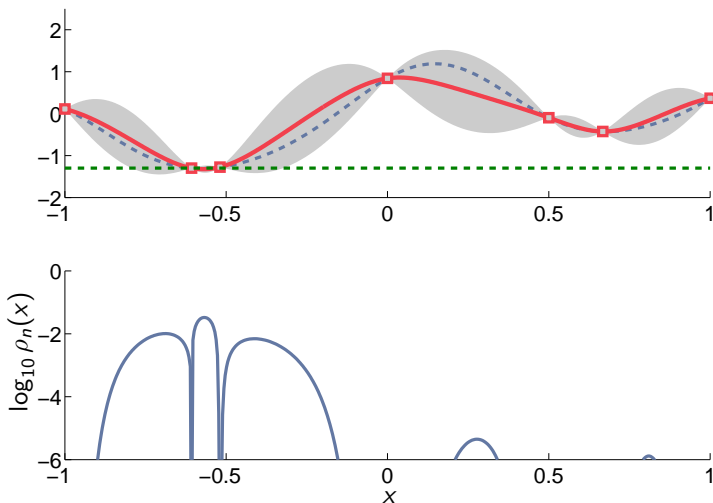
- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$





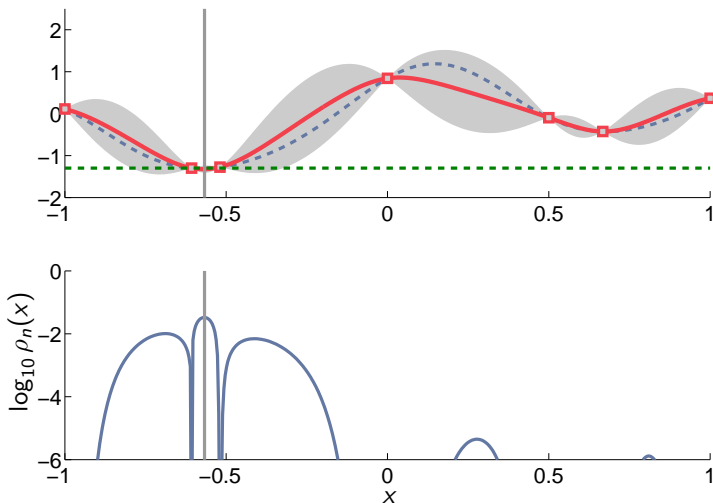
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \arg\max_x \rho_n(x)$



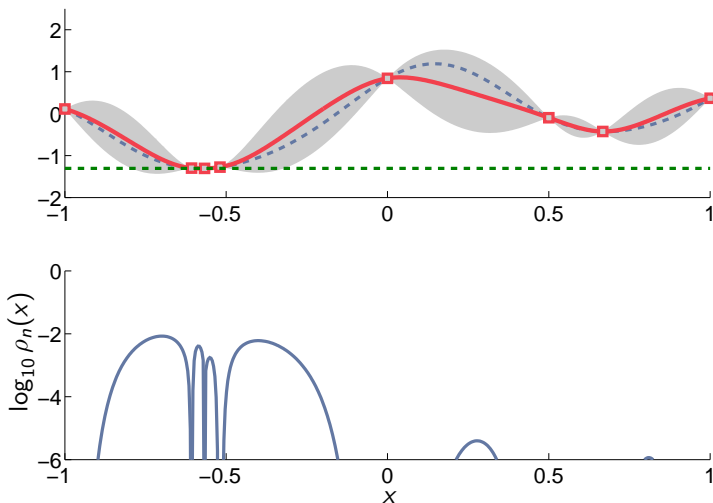
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



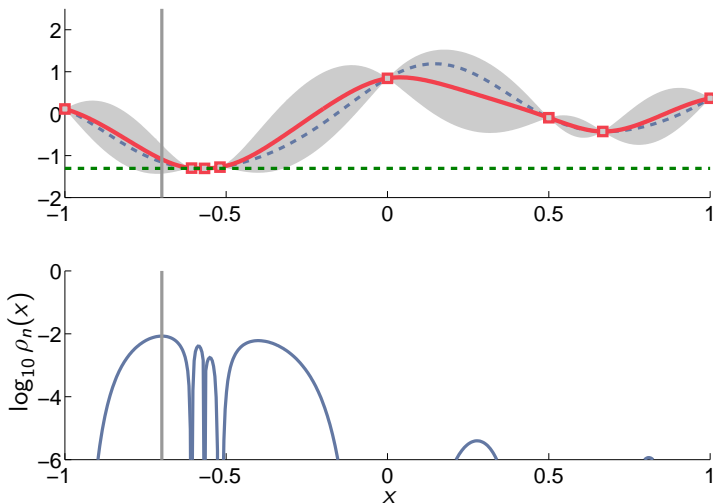
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



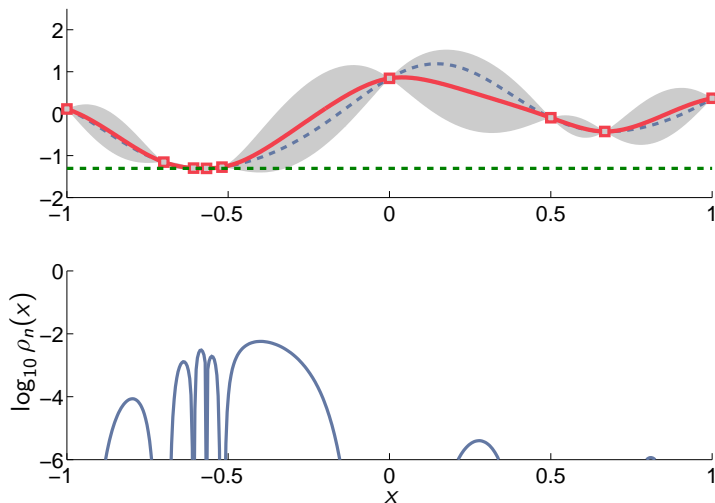
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



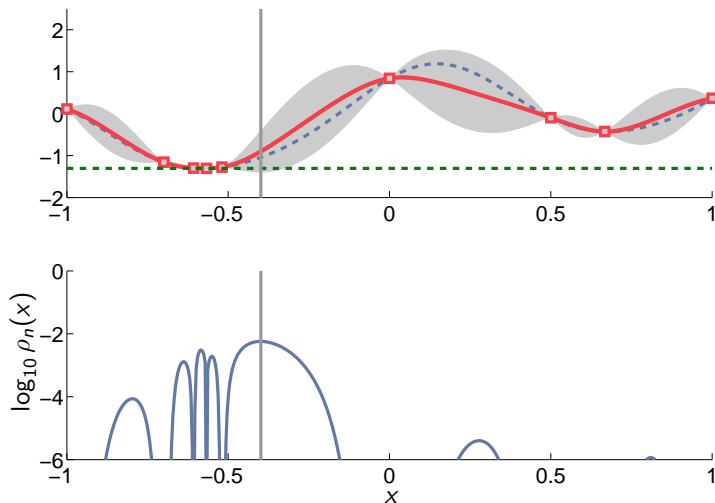
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



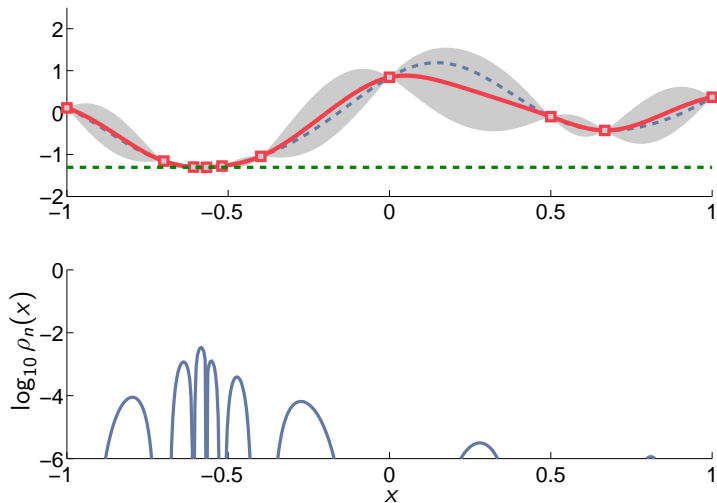
# Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \operatorname{argmax}_x \rho_n(x)$



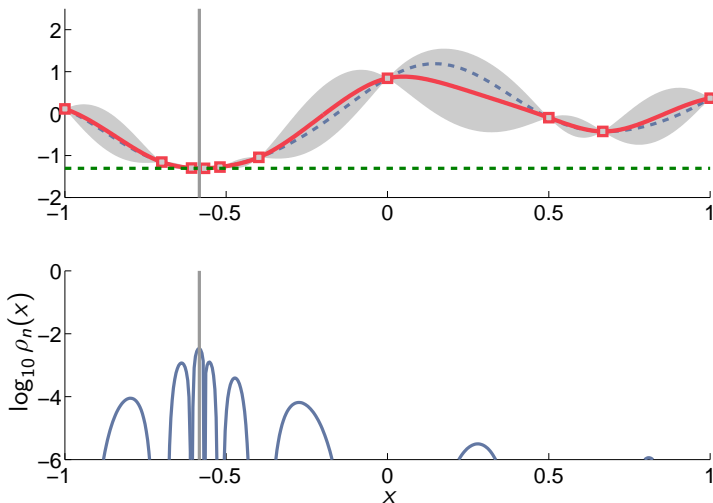
## Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \arg\max_x \rho_n(x)$



## Insertion into an optimization algorithm

- ▶ The EI algorithm:  $X_{n+1} = \arg\max_x \rho_n(x)$





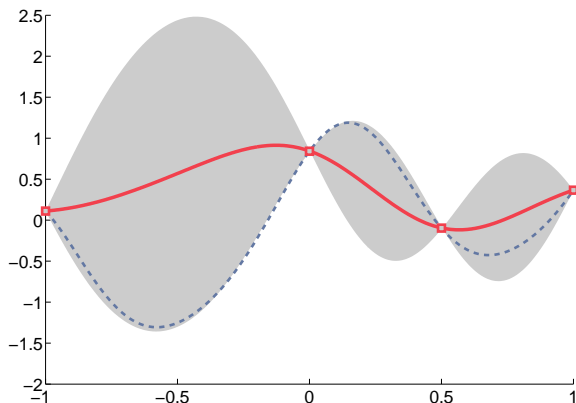
## Example of quantity of interest: the minimizer

- ▶ Assume an unknown function  $f : \mathbb{X} \rightarrow \mathbb{R}$  and suppose we are interested in seeking its **minimizer**:

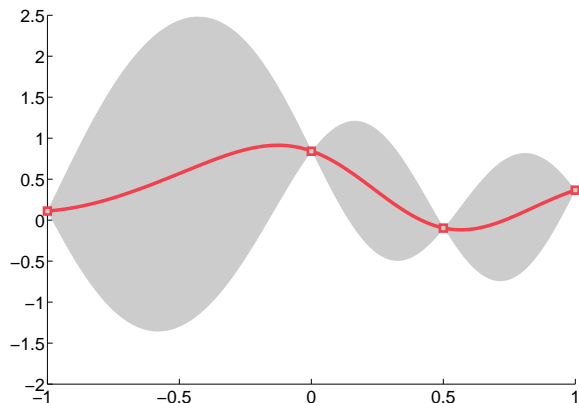
$$x^* = \operatorname{argmin}_x f(x)$$

- ▶ Choose a GP prior  $\xi$  for  $f$ . Given observations  $\xi(x_1), \dots, \xi(x_n)$ , what is the posterior distrib. of  $X^* = \operatorname{argmin}_x \xi(x)$ ?
- ▶ Unlike  $I_n$  above, the distrib. of  $X^*$  does not possess a closed-form expression  $\rightarrow$  resort to an **empirical estimation using conditional sample paths**

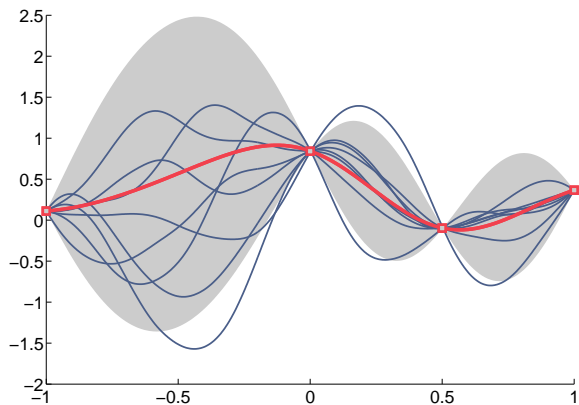
## Empirical posterior density of the minimizer



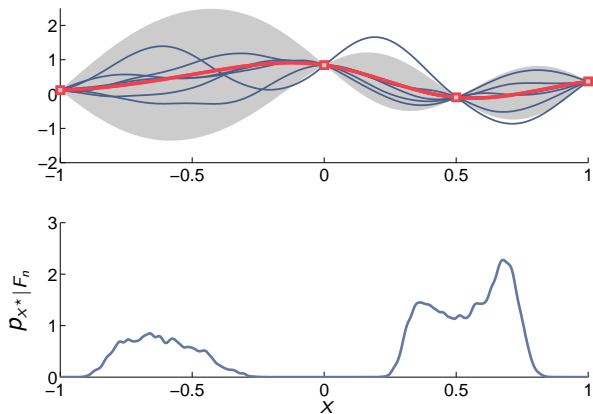
## Empirical posterior density of the minimizer



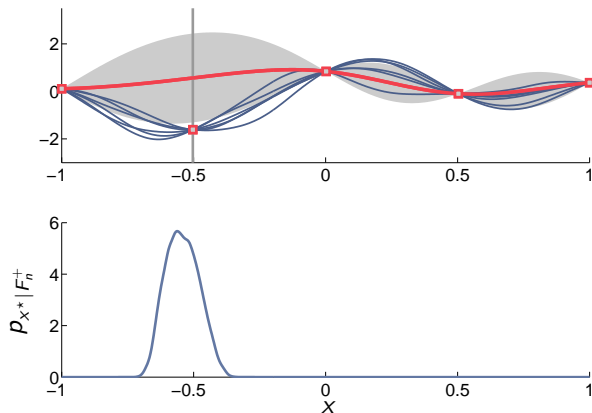
## Empirical posterior density of the minimizer



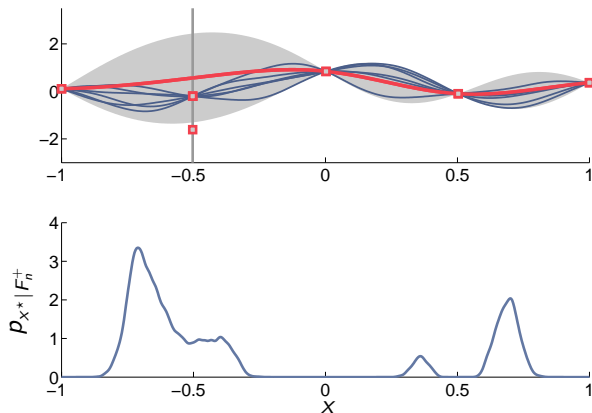
## Empirical posterior density of the minimizer



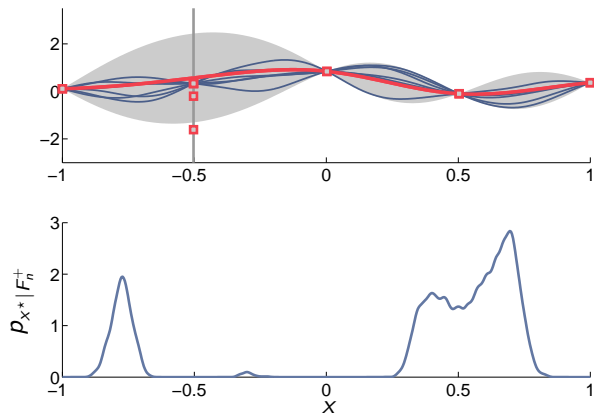
# Empirical posterior density of the minimizer



# Empirical posterior density of the minimizer

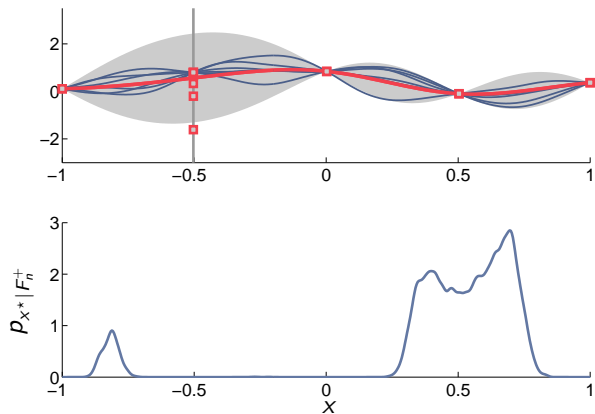


# Empirical posterior density of the minimizer

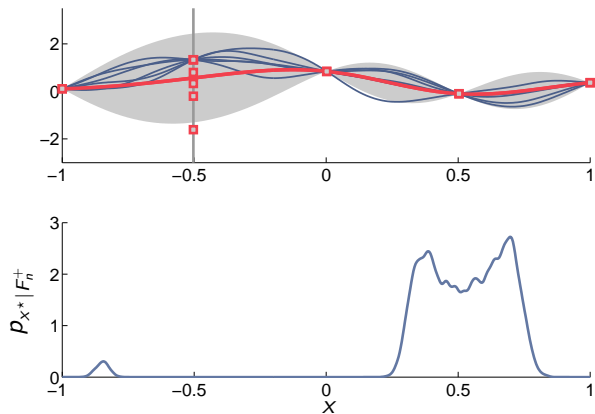




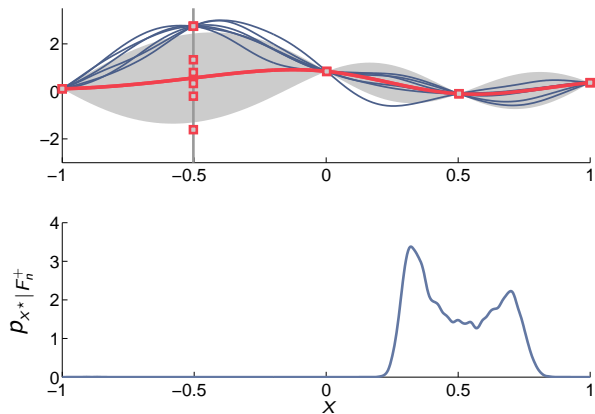
## Empirical posterior density of the minimizer



# Empirical posterior density of the minimizer



## Empirical posterior density of the minimizer



## Lecture 1 : From meta-models to UQ

1.1 Introduction

1.2 Black-box modeling

1.3 Bayesian approach

1.4 Posterior distribution of a quantity of interest

1.5 Complements on Gaussian processes

## Choosing a centered Gaussian random process

How to choose the covariance function of a GP  $\xi \sim \mathcal{N}(0, k)$ ?

# Regularity properties of a random process

Def.

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

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

Prop.

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

# Regularity properties of a random process

Def.

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

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

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

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

Prop.

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

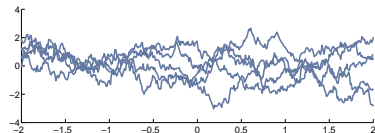
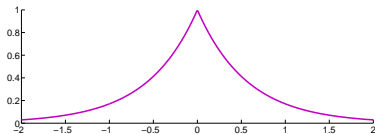
# Regularity properties of a random process

- ▶ Differentiability of the covariance function **at the origin**  $\rightarrow$  **mean-square differentiability** of  $\xi$

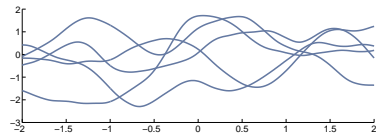
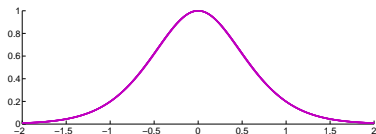


# Influence of the regularity

mean-square continuity



three-time mean-square  
differentiability



## Choice of a covariance

- ▶ A Gaussian process prior carries a high amount of information about  $f$ 
  - it is often difficult to elicit such a prior before any evaluation is made
- ▶ Covariance function of  $\xi$  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 used as if they were known (plug-in approach)
  2. We can assume a prior distrib. 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  
→ offers the possibility to adjust the regularity of  $\xi$  with a single parameter

- ▶ The Matérn function:

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

with

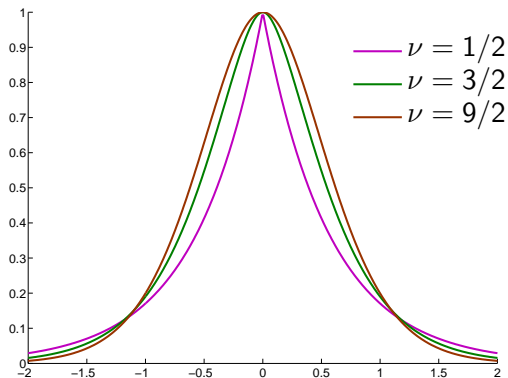
- $\Gamma$  the Gamma function
- $\mathcal{K}_\nu$  the modified Bessel function of the second kind

- ▶ To model a real-valued function defined over  $\mathbb{X} \subset \mathbb{R}$ , we use the Matérn covariance:

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

# Choice of a parametrized covariance function: the Matérn covariance

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



$\xi$  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:

$$k_{\theta}(x, y) = \sigma^2 \kappa_{\nu} \left( \sqrt{\sum_{i=1}^d \frac{(x_{[i]} - y_{[i]})^2}{\rho_i^2}} \right), \quad x, y \in \mathbb{R}^d \quad (3)$$

where  $x_{[i]}, y_{[i]}$  denote the  $i^{\text{th}}$  coordinate of  $x$  and  $y$ , and the positive scalars  $\rho_i$  represent scale parameters

- ▶ Since  $\sigma^2 > 0, \nu > 0, \rho_i > 0, i = 1, \dots, d$ , in practice, we consider the vector of parameters

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

→ makes parameter estimation easier

# Parameter estimation by maximum likelihood

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

$$\ell(\underline{\xi}_n; \theta) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log \det K(\theta) - \frac{1}{2} \underline{\xi}_n^t K(\theta)^{-1} \underline{\xi}_n, \quad (4)$$

where  $K(\theta)$  is the covariance matrix of  $\underline{\xi}_n$ , which depends on the parameter vector  $\theta$

- ▶ The log-likelihood can be maximized using a gradient-based search method

# 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

$$m(\cdot) = \beta^t \varphi(\cdot), \quad (5)$$

with

- $\beta$  a vector of unknown parameters
  - $\varphi = (\varphi_1, \dots, \varphi_l)^t$  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  $\Lambda$  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  $f : \mathbb{X} \rightarrow \mathbb{R}$ , and  $\lambda = \sum_{i=1}^n \lambda_i \delta_{x_i} \in \Lambda$ ,

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

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

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

of  $\xi(x)$  onto  $\text{span}\{\xi(x_i), i = 1, \dots, n\}$  such that the variance of the error  $\xi(x) - \widehat{\xi}_n(x)$  is minimized, under the constraint

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

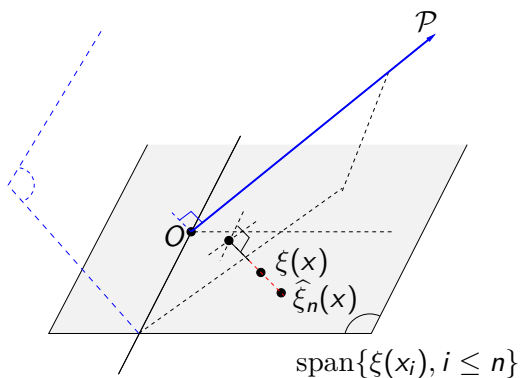
i.e.,

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

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

# Prediction of a Gaussian process with unknown mean function

$\hat{\xi}_n(x)$  is the linear projection of  $\xi(x)$  onto  $\text{span}\{\xi(x_1), \dots, \xi(x_n)\}$  orthogonally to  $\mathcal{P}$



## Prediction of a Gaussian process with unknown mean function

- ▶ The weights  $\lambda_i(x; x_n)$  are again solutions of a **system of linear equations**, which can be written under a matrix form as

$$\begin{pmatrix} K & \underline{\varphi}^t \\ \underline{\varphi} & 0 \end{pmatrix} \begin{pmatrix} \lambda(x) \\ \mu(x) \end{pmatrix} = \begin{pmatrix} k(x) \\ \varphi(x) \end{pmatrix}, \quad (6)$$

with

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

## Prediction of a Gaussian process with unknown mean function

- ▶ When the mean is unknown, the **kriging covariance function** (the covariance of the error of prediction) is given by

$$\begin{aligned}k_n(x, y) &:= \text{cov} \left( \xi(x) - \widehat{\xi}_n(x), \xi(y) - \widehat{\xi}_n(y) \right) \\ &= k(x - y) - \lambda(x)^t k(y) - \mu(x)^t \varphi(y).\end{aligned}$$

### Prop.

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

$$\text{If } \begin{cases} \xi \mid m \sim \mathcal{GP}(m, k) \\ m : x \mapsto \beta^t \varphi(x), \beta \sim U(\mathbb{R}^l) \end{cases} \quad \text{then } \xi \mid F_n \sim \mathcal{GP} \left( \widehat{\xi}_n(\cdot), k_n(\cdot, \cdot) \right)$$

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

- justifies the use of kriging in a **Bayesian framework** provided that the covariance function of  $\xi$  is known

# Parameter estimation with unknown mean function

- ▶ Objective: estimate the covariance parameters of a Gaussian process with unknown mean
- ▶ **Restricted Maximum Likelihood (REML)** approach → maximize the **likelihood of the increments** (or generalized increments) of the data
- ▶ Let  $\xi$  be a Gaussian process with an **unknown mean** function in  $\mathcal{P}$  and  $\underline{\xi}_n$  the random vector of observations at points  $x_i$ ,  $i = 1, \dots, n$
- ▶ Let  $\underline{\varphi} = (\varphi_i(x_j))_{i,j=1}^{l,n}$  be the  $l \times n$  matrix of basis functions of  $\mathcal{P}$  evaluated on  $\{x_1, \dots, x_n\}$ .

## Parameter estimation with unknown mean function

- ▶ Since the dimension of  $\mathcal{P}$  is  $l$ , the dimension of the space of the measures with support  $\{x_1, \dots, x_n\}$  that cancel out the functions of  $\mathcal{P}$  is  $n - l$ .
- ▶ Assume an  $n \times (n - l)$  matrix  $W$  with rank  $n - l$  has been found, such that

$$\underline{\varphi}W = 0.$$

(The columns of  $W$  are in the kernel of  $\underline{\varphi}$ .)

- ▶ Then  $Z = W^T \underline{\xi}_n$  is a Gaussian random vector taking its values in  $\mathbb{R}^{n-l}$ , with **zero mean** and covariance matrix

$$W^T K(\theta) W$$

where  $K(\theta)$  is the covariance matrix of  $\underline{\xi}_n$  with entries  $k_\theta(x_i - x_j)$

# REML

- ▶ The random vector  $Z$  is a **contrast vector**
- ▶ The log-likelihood of the contrasts is given by

$$L(z | \theta) = -\frac{n-l}{2} \log 2\pi - \frac{1}{2} \log \det(W^t K(\theta) W) - \frac{1}{2} z^t (W^t K(\theta) W)^{-1} z.$$

# REML

- ▶ Various methods may be employed to compute the matrix  $W$
- ▶ We favor the QR decomposition of  $\underline{\varphi}^T$

$$\underline{\varphi}^T = (Q_1 \mid Q_2) \begin{pmatrix} R \\ 0 \end{pmatrix},$$

where  $(Q_1 \mid Q_2)$  is an  $n \times n$  orthogonal matrix and  $R$  is a  $l \times l$  upper triangular matrix

- ▶ It is trivial to check that the columns of  $Q_2$  form a basis of the kernel of  $\underline{\varphi}$
- ▶ So we may chose  $W = Q_2$
- ▶ Note that  $W^T W = I_{n-l}$ .



## Books

- ▶ **M. Stein**, Interpolation of Spatial Data: Some Theory for kriging, Springer, 1999
- ▶ **T. Santner, B. Williams and W. Notz**, The Design and Analysis of Computer Experiments, Springer, 2003
- ▶ **C. Rasmussen and C. Williams**, Gaussian processes for Machine Learning, MIT Press, 2006
- ▶ **A. Forrester, A. Sóbester and A. Keane**, Engineering design via surrogate modelling: a practical guide, John Wiley & Sons, 2008

## Lecture 1 : From meta-models to UQ

- 1.1 Introduction
- 1.2 Black-box modeling
- 1.3 Bayesian approach
- 1.4 Posterior distribution of a quantity of interest
- 1.5 Complements on Gaussian processes

## Lecture 2 : Bayesian optimization (BO)

- 2.1. Decision-theoretic framework
- 2.2. From Bayes-optimal to myopic strategies
- 2.3. Design under uncertainty

## References

# What is Bayesian optimization ?

- ▶ “wide sense” definition
  - ▶ optimization using tools from Bayesian UQ

# What is Bayesian optimization ?

- ▶ “wide sense” definition
  - ▶ optimization using tools from Bayesian UQ
  - ▶ started with **Harold Kushner**'s paper: *A New Method of Locating the Maximum Point of an Arbitrary Multipeak Curve in the Presence of Noise*, J. Basic Engineering, 1964.



# What is Bayesian optimization ?

- ▶ a slightly more restrictive definition
  - ▶ sequential **Bayesian decision theory** applied to optimization

# What is Bayesian optimization ?

- ▶ a slightly more restrictive definition
  - ▶ sequential Bayesian decision theory applied to optimization
  - ▶ started with the work of **Jonas Mockus** and **Antanas Žilinskas** in the 70's, e.g., *On a Bayes method for seeking an extremum*, *Avtomatika i Vychislitel'naya Teknika*, 1972 (in Russian)



# What is Bayesian optimization ?

- ▶ a slightly more restrictive definition
  - ▶ sequential Bayesian decision theory applied to optimization
  - ▶ started with the work of Jonas Mockus and Antanas Žilinskas in the 70's, e.g., *On a Bayes method for seeking an extremum*, *Avtomatika i Vychislitel'naya Teknika*, 1972 (in Russian)



- ▶ In this lecture we adopt this second (more constructive !) definition

## Lecture 2 : Bayesian optimization (BO)

2.1. Decision-theoretic framework

2.2. From Bayes-optimal to myopic strategies

2.3. Design under uncertainty



# Decision-theoretic framework

# Decision-theoretic framework

- ▶ Bayesian decision theory (BDT) in a nutshell
  - ▶ a mathematical framework for **decisions under uncertainty**
  - ▶ uncertainty is captured by **probability distributions**
  - ▶ the “Bayesian agent” aims at **minimizing the expected loss**



## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The **agent** is the **optimization algorithm** (or you, if you will)

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature”
- ▶ a prior distribution over the states of nature
- ▶ a description of the decisions we have to make
- ▶ and the corresponding “transitions”
- ▶ a loss function (or utility function)

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature”
- ▶ a prior distribution over the states of nature
- ▶ a description of the decisions we have to make
- ▶ and the corresponding “transitions”
- ▶ a loss function (or utility function)

# States of nature

- ▶ What are the **states of nature** in an optimization problem?
- ▶ Short answer
  - ▶ Everything that “nature” knows but you don't

# States of nature

- ▶ What are the states of nature in an optimization problem?
- ▶ Short answer
  - ▶ Everything that “nature” knows but you don't
- ▶ More practically: depends on the type of problem
  1. the **content of the black box** (expensive numerical model)
  2. for stochastic simulators: **future responses** of the simulator
  3. design under uncertainty: the value of **environmental variables**
  4. ...



# States of nature

- ▶ What are the states of nature in an optimization problem?
- ▶ Short answer
  - ▶ Everything that “nature” knows but you don’t
- ▶ More practically: depends on the type of problem
  1. the content of the black box (expensive numerical model)
  2. for stochastic simulators: future responses of the simulator
  3. design under uncertainty: the value of environmental variables
  4. ...

## Notation

$$\Omega = \{\text{all possible states } \omega \text{ of nature}\}$$

# States of nature: example

- ▶ Consider the following setting
  - ▶ a **deterministic** numerical model
  - ▶ input space  $\mathbb{X} \subset \mathbb{R}^d$ , output space  $\mathbb{R}^p$
  - ▶ **no environmental** variables in the problem
  
- ▶ States of nature for this setting
  - ▶  $\Omega = \{f : \mathbb{X} \rightarrow \mathbb{R}^p \mid f \text{ such that } \dots\}$

# States of nature: example

- ▶ Consider the following setting
  - ▶ a deterministic numerical model
  - ▶ input space  $\mathbb{X} \subset \mathbb{R}^d$ , output space  $\mathbb{R}^p$
  - ▶ no environmental variables in the problem
- ▶ States of nature for this setting
  - ▶  $\Omega = \{f : \mathbb{X} \rightarrow \mathbb{R}^p \mid f \text{ such that } \dots\}$
  - ▶ e.g.,  $d = 1$ ,  $p = 1$ ,  $\mathbb{X} = [0; 1]$  and  $\Omega = C(\mathbb{X}; \mathbb{R})$
- ▶ Until further notice, we will use this simple (but important) setting to illustrate the basics of Bayesian optimization

# Uncertainty quantification (reminder from Lecture #1)

- ▶ The true state of nature  $\omega^* \in \Omega$  is **unknown**
- ▶ Example (cont'd)
  - ▶ a function  $f^* \in \Omega = C(\mathbb{X}; \mathbb{R})$  is inside the black box ( $\omega^* \equiv f^*$ )
  - ▶ we don't “know”  $f^*(x)$  until we run the code with input  $x$

# Uncertainty quantification (reminder from Lecture #1)

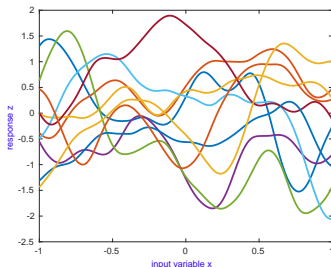
- ▶ The true state of nature  $\omega^* \in \Omega$  is unknown
- ▶ Example (cont'd)
  - ▶ a function  $f^* \in \Omega = C(\mathbb{X}; \mathbb{R})$  is inside the black box ( $\omega^* \equiv f^*$ )
  - ▶ we don't "know"  $f^*(x)$  until we run the code with input  $x$
- ▶ Bayesian approach to UQ
  - ▶ our knowledge of  $\omega^*$  is encoded by a **probability distribution** on the set  $\Omega$  of all possible  $\omega$ 's
  - ▶ technically: proba on  $(\Omega, \mathcal{F})$  for some  $\sigma$ -algebra  $\mathcal{F} \dots$

# Uncertainty quantification (reminder from Lecture #1)

- ▶ The true state of nature  $\omega^* \in \Omega$  is unknown
- ▶ Example (cont'd)
  - ▶ a function  $f^* \in \Omega = C(\mathbb{X}; \mathbb{R})$  is inside the black box ( $\omega^* \equiv f^*$ )
  - ▶ we don't "know"  $f^*(x)$  until we run the code with input  $x$
- ▶ Bayesian approach to UQ
  - ▶ our knowledge of  $\omega^*$  is encoded by a probability distribution on the set  $\Omega$  of all possible  $\omega$ 's
  - ▶ technically: proba on  $(\Omega, \mathcal{F})$  for some  $\sigma$ -algebra  $\mathcal{F} \dots$
- ▶ Sequence of decisions  $\Rightarrow$  **sequence of distributions**  $P_0, P_1, \dots$ 
  - ▶  $P_n$  corresponds to the agent's beliefs after the  $n^{\text{th}}$  decision
  - ▶ A **prior distribution**  $P_0$  needs to be specified

# Uncertainty quantification: example

- ▶ Example (cont'd): if  $f$  is known to look more or less like this:

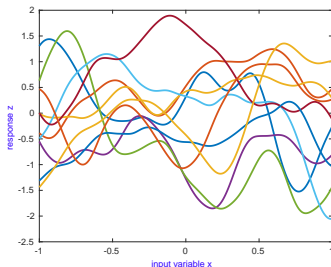


then we can take  $P_0 = \mathcal{GP}(m, k)$

with  $m \sim \mathcal{U}(\mathbb{R})$  and  $k$  a (stationary) Matérn covariance

# Uncertainty quantification: example

- ▶ Example (cont'd): if  $f$  is known to look more or less like this:



then we can take  $P_0 = \mathcal{GP}(m, k)$   
with  $m \sim \mathcal{U}(\mathbb{R})$  and  $k$  a (stationary) Matérn covariance

- ▶ Gaussian process priors are commonly used because
  - ▶ they are **computationally convenient**
  - ▶ while allowing a certain **modeling flexibility**



# Uncertainty quantification: consequences

- ▶ For clarity, consider again the case of a **deterministic model**:
  - ▶ an **unknown function**  $f \in \Omega$  is in the black box

# Uncertainty quantification: consequences

- ▶ For clarity, consider again the case of a deterministic model:
    - ▶ an unknown function  $f \in \Omega$  is in the black box
  - ▶ Given a proba  $P$  on  $\Omega$ , we can
    - ▶ compute the **probability** of any (measurable) statement about  $f$
    - ▶ compute the **expectation** of any (measurable) function of  $f$
- i.e., the unknown  $f$  can be **treated as random**

# Uncertainty quantification: consequences

- ▶ For clarity, consider again the case of a deterministic model:
    - ▶ an unknown function  $f \in \Omega$  is in the black box
  - ▶ Given a proba  $P$  on  $\Omega$ , we can
    - ▶ compute the probability of any (measurable) statement about  $f$
    - ▶ compute the expectation of any (measurable) function of  $f$
- i.e., the unknown  $f$  can be treated as random

## Convenient notation

$\xi$  = random function that represents the unknown  $f$

- ▶ we will write, e.g.,  $E_n(\xi(x))$  instead of  $\int_{\Omega} f(x) P_n(df)$  ☺

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature”
- ▶ a prior distribution over the states of nature
- ▶ a description of the decisions we have to make
- ▶ and the corresponding “transitions”
- ▶ a loss function (or utility function)

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature” ✓
- ▶ a prior distribution over the states of nature ✓
- ▶ a description of the **decisions** we have to make
- ▶ and the corresponding “**transitions**”
- ▶ a loss function (or utility function)

# Decisions

- ▶ Several types of decisions in an optimization procedure:
  - ▶ intermediate decisions
  - ▶ stopping decision
  - ▶ final decision

# Decisions: intermediate decisions

- ▶ Several types of decisions in an optimization procedure:
  - ▶ **intermediate decisions**
  - ▶ stopping decision
  - ▶ final decision
  
- ▶ Intermediate decisions (simple setting)
  - ▶ **running the numerical model** with a given input  $x \in \mathbb{X}$
  - ▶ getting the corresponding output (deterministic or random)

# Decisions: intermediate decisions

- ▶ Several types of decisions in an optimization procedure:
  - ▶ **intermediate decisions**
  - ▶ stopping decision
  - ▶ final decision
- ▶ Intermediate decisions (simple setting)
  - ▶ running the numerical model with a given input  $x \in \mathbb{X}$
  - ▶ getting the corresponding output (deterministic or random)
- ▶ Intermediate decisions (various extensions)
  - ▶ parallel computing: **batches** of input values
  - ▶ multi-fidelity: choosing the right **fidelity level**
  - ▶ variable run-time: choosing when to **stop a computation**
  - ▶ ...



# Decisions: stopping decision

- ▶ Several types of decisions in an optimization procedure:
  - ▶ intermediate decisions
  - ▶ **stopping decision**
  - ▶ final decision
  
- ▶ Stopping decision (standard setting)
  - ▶ a **budget of evaluations**, or computation time, is given
  - ▶ the stopping decision is trivial in this case

# Decisions: stopping decision

- ▶ Several types of decisions in an optimization procedure:
  - ▶ intermediate decisions
  - ▶ **stopping decision**
  - ▶ final decision
- ▶ Stopping decision (standard setting)
  - ▶ a budget of evaluations, or computation time, is given
  - ▶ the stopping decision is trivial in this case
- ▶ Digression: taking the cost of observations into account?
  - ▶ *in principle*, BO can deal with the stopping decision too
  - ▶ in practice, difficult to translate into a loss (see later)
  - ▶ some “BO papers” propose heuristic stopping rule

## Decisions: final decision

- ▶ Several types of decisions in an optimization procedure:
  - ▶ intermediate decisions
  - ▶ stopping decision
  - ▶ **final decision**
  
- ▶ Final decision (ex: single-objective minimization pb.)
  - ▶ an **estimate of the minimizer**  $x^* = \operatorname{argmin}_{x \in \mathbb{X}} f(x)$
  - ▶ and/or an **estimate of the minimum**  $M^* = \min_{x \in \mathbb{X}} f(x)$

with  $\mathbb{X}$  a “known” input space

## Decisions: final decision

- ▶ Several types of decisions in an optimization procedure:
  - ▶ intermediate decisions
  - ▶ stopping decision
  - ▶ **final decision**

- ▶ Final decision (ex: single-objective minimization pb.)

- ▶ an estimate of the minimizer  $x^* = \operatorname{argmin}_{x \in \mathbb{X}} f(x)$
- ▶ and/or an estimate of the minimum  $M^* = \min_{x \in \mathbb{X}} f(x)$

with  $\mathbb{X}$  a “known” input space

- ▶ Other settings

- ▶ **multi-objective**: Pareto set / Pareto front (see later),
- ▶ **inequality constraints** (see later), equality constraints (harder !)
- ▶ quasi-optimal region (sublevel set)...

## Decisions: standard setting and notations

- ▶ From now on we focus on the “standard” setting
  - ▶ **intermediate decisions  $\equiv$  evaluations** (known comput. cost)
  - ▶ stopping: a **budget of  $N$  evaluations** is given

# Decisions: standard setting and notations

- ▶ From now on we focus on the “standard” setting
  - ▶ intermediate decisions  $\equiv$  evaluations (known comput. cost)
  - ▶ stopping: a budget of  $N$  evaluations is given

## Notations

$X_n(\omega)$  = the  $n^{\text{th}}$  evaluation point

$D_{N+1}(\omega)$  = the “final decision” (estimate of the QoI)

$\underline{D}(\omega) = (X_1(\omega), \dots, X_N(\omega), D_{N+1}(\omega))$

# Transitions: conditioning probability measures

- ▶ Recall that
  - ▶ the agent's knowledge at time  $n$  is described by  $P_n$
  - ▶ the  $(n + 1)^{\text{th}}$  decision induces a transition  $P_n \rightarrow P_{n+1}$

# Transitions: conditioning probability measures

- ▶ Recall that
  - ▶ the agent's knowledge at time  $n$  is described by  $P_n$
  - ▶ the  $(n + 1)^{\text{th}}$  decision induces a transition  $P_n \rightarrow P_{n+1}$

## Notations: available information

$I_n$  = the information obtained as a result of the  $n^{\text{th}}$  decision

$F_n$  = the information available at time  $n$

- ▶ Mathematically,  $P_n = P_0(\cdot | F_n) = P_{n-1}(\cdot | I_n)$
- ▶ Various settings can be addressed
  - ▶ depending on what we define as  $I_n$  😊



## Decisions: standard setting and notations

- ▶ From now on we focus on the “standard” setting
  - ▶ intermediate decisions  $\equiv$  evaluations (known comput. cost)
  - ▶ stopping: a budget of  $N$  evaluations is given

### Notations

$X_n(\omega)$  = the  $n^{\text{th}}$  evaluation point

$D_{N+1}(\omega)$  = the “final decision” (estimate of the QoI)

$\underline{D}(\omega) = (X_1(\omega), \dots, X_N(\omega), D_{N+1}(\omega))$

- ▶ We cannot use information that is not yet available
  - ▶  $X_n(\omega)$  depends on  $\omega$  through  $F_{n-1}$  only
  - ▶  $D_{N+1}(\omega)$  depends on  $\omega$  through  $F_N$  only
  - ▶  $\underline{D}$  is a decision strategy (sequence of decision rules)

# Transitions: examples

- ▶ Example (cont'd)
  - ▶ single-output, deterministic code
  - ▶  $I_n = (X_n, \xi(X_n))$  and  $F_n = (X_1, \xi(X_1), \dots, X_n, \xi(X_n))$
  - ▶  $\xi(X_i) = f^*(X_i)$  is the true, scalar, value of the model at  $X_i$

# Transitions: examples

- ▶ Example (cont'd)
  - ▶ single-output, deterministic code
  - ▶  $I_n = (X_n, \xi(X_n))$  and  $F_n = (X_1, \xi(X_1), \dots, X_n, \xi(X_n))$
  - ▶  $\xi(X_i) = f^*(X_i)$  is the true, scalar, value of the model at  $X_i$
- ▶ Many other (interesting) settings are possible !
  - ▶ **stochastic simulators**
    - ▶ the output is a random draw  $Z_n \sim$  some distrib.  $P^{Z_n|X_n}$
    - ▶  $I_n = (X_n, Z_n)$  and  $F_n = (X_1, Z_1, \dots, X_n, Z_n)$

# Transitions: examples

- ▶ Example (cont'd)
  - ▶ single-output, deterministic code
  - ▶  $I_n = (X_n, \xi(X_n))$  and  $F_n = (X_1, \xi(X_1), \dots, X_n, \xi(X_n))$
  - ▶  $\xi(X_i) = f^*(X_i)$  is the true, scalar, value of the model at  $X_i$
- ▶ Many other (interesting) settings are possible !
  - ▶ stochastic simulators
    - ▶ the output is a random draw  $Z_n \sim$  some distrib.  $\mathbb{P}^{Z_n|X_n}$
    - ▶  $I_n = (X_n, Z_n)$  and  $F_n = (X_1, Z_1, \dots, X_n, Z_n)$
  - ▶ availability of **gradients** (e.g., adjoint code)

# Transitions: examples

- ▶ Example (cont'd)
  - ▶ single-output, deterministic code
  - ▶  $I_n = (X_n, \xi(X_n))$  and  $F_n = (X_1, \xi(X_1), \dots, X_n, \xi(X_n))$
  - ▶  $\xi(X_i) = f^*(X_i)$  is the true, scalar, value of the model at  $X_i$
- ▶ Many other (interesting) settings are possible !
  - ▶ stochastic simulators
    - ▶ the output is a random draw  $Z_n \sim$  some distrib.  $\mathbb{P}^{Z_n|X_n}$
    - ▶  $I_n = (X_n, Z_n)$  and  $F_n = (X_1, Z_1, \dots, X_n, Z_n)$
  - ▶ availability of gradients (e.g., adjoint code)
  - ▶ **batch** setting and/or **multiple outputs**

# Transitions: examples

- ▶ Example (cont'd)
  - ▶ single-output, deterministic code
  - ▶  $I_n = (X_n, \xi(X_n))$  and  $F_n = (X_1, \xi(X_1), \dots, X_n, \xi(X_n))$
  - ▶  $\xi(X_i) = f^*(X_i)$  is the true, scalar, value of the model at  $X_i$
- ▶ Many other (interesting) settings are possible !
  - ▶ stochastic simulators
    - ▶ the output is a random draw  $Z_n \sim$  some distrib.  $\mathbb{P}^{Z_n|X_n}$
    - ▶  $I_n = (X_n, Z_n)$  and  $F_n = (X_1, Z_1, \dots, X_n, Z_n)$
  - ▶ availability of gradients (e.g., adjoint code)
  - ▶ batch setting and/or multiple outputs
  - ▶ variable run-times, “simulation failures”...

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature” ✓
- ▶ a prior distribution over the states of nature ✓
- ▶ a description of the **decisions** we have to make
- ▶ and the corresponding “**transitions**”
- ▶ a loss function (or utility function)

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature” ✓
- ▶ a prior distribution over the states of nature ✓
- ▶ a description of the decisions we have to make ✓
- ▶ and the corresponding “transitions” ✓
- ▶ a **loss function** (or utility function)



# Loss function

- ▶ To guide the decisions of the Bayesian agent, we need to specify a **loss function**  $L$

## Notation

$$L : \Omega \times \mathbb{D} \rightarrow \mathbb{R}$$

$$(\omega, \underline{d}) \mapsto L(\omega, \underline{d})$$

where  $\mathbb{D}$  is the set of all possible sequences of decisions

# Loss function

- ▶ To guide the decisions of the Bayesian agent, we need to specify a loss function  $L$

## Notation

$$L : \Omega \times \mathbb{D} \rightarrow \mathbb{R}$$
$$(\omega, \underline{d}) \mapsto L(\omega, \underline{d})$$

where  $\mathbb{D}$  is the set of all possible sequences of decisions

- ▶ The **Bayes-optimal strategy** is, by definition:

$$\begin{aligned} \underline{D} &= \operatorname{argmin} E_0(L(\underline{D})) \\ &= \operatorname{argmin} \int_{\Omega} L(\omega, \underline{D}(\omega)) P_0(d\omega) \end{aligned}$$

where  $\underline{D}$  ranges over all strategies

## Loss function: example

- ▶ Example (cont'd)
  - ▶ Assume that we want to find the minimizer of  $f$ 
    - ▶  $\underline{d} = (x_1, \dots, x_n, \hat{x})$
    - ▶ with  $\hat{x}$  our estimate of  $\operatorname{argmin} f$

# Loss function: example

- ▶ Example (cont'd)

- ▶ Assume that we want to find the minimizer of  $f$

- ▶  $\underline{d} = (x_1, \dots, x_n, \hat{x})$

- ▶ with  $\hat{x}$  our estimate of  $\operatorname{argmin} f$

- ▶ A standard loss function for this situation is the **linear loss**:

$$L(f, \underline{d}) = L(f, \hat{x}) = f(\hat{x}) - \min f$$

(a.k.a. **opportunity cost**, a.k.a. **instantaneous regret**)

# Loss function: example

## ▶ Example (cont'd)

- ▶ Assume that we want to find the minimizer of  $f$

- ▶  $\underline{d} = (x_1, \dots, x_n, \hat{x})$

- ▶ with  $\hat{x}$  our estimate of  $\operatorname{argmin} f$

- ▶ A standard loss function for this situation is the linear loss:

$$L(f, \underline{d}) = L(f, \hat{x}) = f(\hat{x}) - \min f$$

(a.k.a. opportunity cost, a.k.a. instantaneous regret)

## ▶ Remarks

- ▶  $L$  coincides with the  $L^1$  loss of the estimator  $f(\hat{x})$

$$f(\hat{x}) \geq \min f \quad \Rightarrow \quad L(f, \hat{x}) = |f(\hat{x}) - \min f|$$

- ▶  $L$  is a **terminal loss** (does not depend on  $X_1, \xi(X_1), \dots$ )

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature” ✓
- ▶ a prior distribution over the states of nature ✓
- ▶ a description of the decisions we have to make ✓
- ▶ and the corresponding “transitions” ✓
- ▶ a **loss function** (or utility function)

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature” ✓
- ▶ a prior distribution over the states of nature ✓
- ▶ a description of the decisions we have to make ✓
- ▶ and the corresponding “transitions” ✓
- ▶ a loss function (or utility function) ✓

- ▶ Our BDT framework is complete, let's use it 😊

## Decision-theoretic framework (cont'd)

- ▶ How does this relate to optimization ?
- ▶ The agent is the optimization algorithm (or you, if you will)

### Ingredients of a BDT problem

- ▶ a set of all possible “states of nature” ✓
- ▶ a prior distribution over the states of nature ✓
- ▶ a description of the decisions we have to make ✓
- ▶ and the corresponding “transitions” ✓
- ▶ a loss function (or utility function) ✓

- ▶ Our BDT framework is complete, let's use it 😊



## Lecture 2 : Bayesian optimization (BO)

2.1. Decision-theoretic framework

2.2. From Bayes-optimal to myopic strategies

2.3. Design under uncertainty

# Problem statement

- ▶ Assume a standard BO setting
  - ▶ fixed budget  $N$ , terminal cost only
    - ▶  $L(\omega, \underline{d}) = L(\omega, d_{N+1})$
  - ▶ intermediate decisions  $\equiv$  evaluations
    - ▶ one at a time, possibly noisy (stochastic simulator)
    - ▶  $F_n = (X_1, Z_1, \dots, X_n, Z_n)$

## Problem statement

- ▶ Assume a standard BO setting
  - ▶ fixed budget  $N$ , terminal cost only
    - ▶  $L(\omega, \underline{d}) = L(\omega, d_{N+1})$
  - ▶ intermediate decisions  $\equiv$  evaluations
    - ▶ one at a time, possibly noisy (stochastic simulator)
    - ▶  $F_n = (X_1, Z_1, \dots, X_n, Z_n)$
- ▶ Recall the **Bayes-optimal strategy** (algorithm):

$$\begin{aligned}\underline{D}^{\text{Bayes}} &= \operatorname{argmin}_{\underline{D}} E_0(L(D_{N+1})) \\ &= \operatorname{argmin}_{\underline{D}} \int_{\Omega} L(\omega, D_{N+1}(\omega)) P_0(d\omega)\end{aligned}$$

where  $\underline{D}$  ranges over all strategies  $\underline{D} = (X_1, \dots, X_N, D_{N+1})$

## Problem statement

What does this  $\underline{D}^{\text{Bayes}}$  look like ?

Can we actually **build an optimal Bayesian algorithm** ?

## Problem statement (more precisely)

- ▶ Recall that
  - ▶  $X_{n+1}(\omega)$  must depend on  $\omega$  through  $F_n$  only
  - ▶  $D_{N+1}(\omega)$  must depend on  $\omega$  through  $F_N$  only

## Problem statement (more precisely)

- ▶ Recall that
  - ▶  $X_{n+1}(\omega)$  must depend on  $\omega$  through  $F_n$  only
  - ▶  $D_{N+1}(\omega)$  must depend on  $\omega$  through  $F_N$  only

### Notations

$$X_{n+1} = \varphi_n(X_1, Z_1, \dots, X_n, Z_n) = \varphi_n(F_n)$$

$$D_{N+1} = \varphi_N(X_1, Z_1, \dots, X_N, Z_N) = \varphi_N(F_N)$$

## Problem statement (more precisely)

- ▶ Recall that
  - ▶  $X_{n+1}(\omega)$  must depend on  $\omega$  through  $F_n$  only
  - ▶  $D_{N+1}(\omega)$  must depend on  $\omega$  through  $F_N$  only

### Notations

$$X_{n+1} = \varphi_n(X_1, Z_1, \dots, X_n, Z_n) = \varphi_n(F_n)$$

$$D_{N+1} = \varphi_N(X_1, Z_1, \dots, X_N, Z_N) = \varphi_N(F_N)$$

- ▶ Goal: find the functions  $\varphi_0, \dots, \varphi_N$

## Lecture 2 : Bayesian optimization (BO)

### 2.2. From Bayes-optimal to myopic strategies

The optimal terminal decision

Optimal choice of the last evaluation

Bayes-optimal versus “practical Bayes” optimization

Sampling criteria for multi-objective and/or constrained optimization



# Optimal terminal decision

## Notation

$E_n = E(\cdot | F_n)$  = conditional expectation with respect to  $F_n$   
= expectation with respect to the probability  $P_n$

# Optimal terminal decision

## Notation

$E_n = E(\cdot | F_n)$  = conditional expectation with respect to  $F_n$   
= expectation with respect to the probability  $P_n$

- ▶ Consider any incomplete strategy  $X_1, \dots, X_N$
- ▶ Claim: the **optimal terminal decision** is

$$D_{N+1} = \varphi_N^{\text{Bayes}}(X_1, Z_1, \dots, X_N, Z_N) = \text{argmin}_d E_N(L(d))$$

where  $d$  runs over all possible values for the terminal decision

## Optimal terminal decision: proof

- ▶ Take **any strategy**  $\underline{D} = (X_1, \dots, X_N, D_{N+1})$
- ▶ Consider the **modified strategy**  $\underline{D}' = (X_1, \dots, X_N, D'_{N+1})$   
where  $D'_{N+1} = \varphi_N^{\text{Bayes}}(X_1, Z_1, \dots, X_N, Z_N)$

## Optimal terminal decision: proof

- ▶ Take any strategy  $\underline{D} = (X_1, \dots, X_N, D_{N+1})$
- ▶ Consider the modified strategy  $\underline{D}' = (X_1, \dots, X_N, D'_{N+1})$   
where  $D'_{N+1} = \varphi_N^{\text{Bayes}}(X_1, Z_1, \dots, X_N, Z_N)$
- ▶ Then, by definition of  $\varphi_N^{\text{Bayes}}$ ,

$$\begin{aligned} E_N(L(D_{N+1})) &= E_N(L(d))|_{d=D_{N+1}} \\ &\geq \min_d E_N(L(d)) = E_N(L(D'_{N+1})) \end{aligned}$$

## Optimal terminal decision: proof

- ▶ Take any strategy  $\underline{D} = (X_1, \dots, X_N, D_{N+1})$
- ▶ Consider the modified strategy  $\underline{D}' = (X_1, \dots, X_N, D'_{N+1})$   
where  $D'_{N+1} = \varphi_N^{\text{Bayes}}(X_1, Z_1, \dots, X_N, Z_N)$
- ▶ Then, by definition of  $\varphi_N^{\text{Bayes}}$ ,

$$\begin{aligned} E_N(L(D_{N+1})) &= E_N(L(d))|_{d=D_{N+1}} \\ &\geq \min_d E_N(L(d)) = E_N(L(D'_{N+1})) \end{aligned}$$

- ▶ and thus

$$\begin{aligned} E_0(L(D_{N+1})) &= E_0(E_N(L(D_{N+1}))) \\ &\geq E_0(E_N(L(D'_{N+1}))) = E_0(L(D'_{N+1})) \quad \blacksquare \end{aligned}$$

## Vocabulary: posterior (Bayes) risk at time $N$

- ▶ Define the **posterior risk** at time  $N$  for the decision  $d$ :

$$R_N(d) = E_N(L(d))$$

(“risk” is a synonym for “expected loss”)

## Vocabulary: posterior (Bayes) risk at time $N$

- ▶ Define the posterior risk at time  $N$  for the decision  $d$ :

$$R_N(d) = E_N(L(d))$$

(“risk” is a synonym for “expected loss”)

- ▶ Define the **posterior Bayes risk** at time  $N$ :

$$R_N^{\text{Bayes}} = \min_d R_N(d)$$

- ▶ Remember: the min attained for  $d = \varphi_N^{\text{Bayes}}(F_N)$

## Example (cont'd): linear loss

- ▶ Recall the setting
  - ▶ goal: minimize  $f$
  - ▶  $d_{N+1} = \hat{x}$  is an estimate of  $X^*(f) = \operatorname{argmin} f$
  - ▶  $L(f, \hat{x}) = f(\hat{x}) - \min f$



## Example (cont'd): linear loss

- ▶ Recall the setting
  - ▶ goal: minimize  $f$
  - ▶  $d_{N+1} = \hat{x}$  is an estimate of  $X^*(f) = \operatorname{argmin} f$
  - ▶  $L(f, \hat{x}) = f(\hat{x}) - \min f$
- ▶ Compute the **posterior risk** at time  $N$  for a given  $\hat{x} \in \mathbb{X}$ :

$$\begin{aligned}R_N(\hat{x}) &= \mathbb{E}_N(L(\xi, \hat{x})) = \mathbb{E}_N(\xi(\hat{x}) - \min \xi) \\ &= \hat{\xi}_N(\hat{x}) - \mathbb{E}_N(\min \xi)\end{aligned}$$

## Example (cont'd): linear loss

- ▶ Recall the setting
  - ▶ goal: minimize  $f$
  - ▶  $d_{N+1} = \hat{x}$  is an estimate of  $X^*(f) = \operatorname{argmin} f$
  - ▶  $L(f, \hat{x}) = f(\hat{x}) - \min f$
- ▶ Compute the posterior risk at time  $N$  for a given  $\hat{x} \in \mathbb{X}$ :

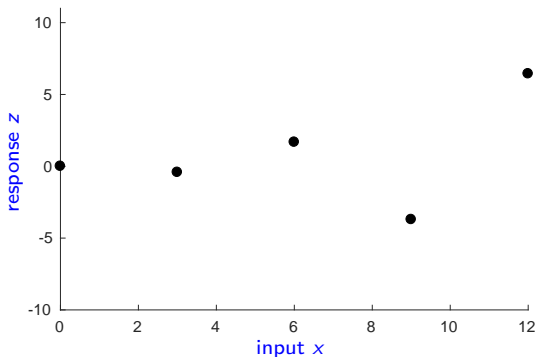
$$\begin{aligned}R_N(\hat{x}) &= E_N(L(\xi, \hat{x})) = E_N(\xi(\hat{x}) - \min \xi) \\ &= \hat{\xi}_N(\hat{x}) - E_N(\min \xi)\end{aligned}$$

- ▶ Thus the optimal terminal decision is

$$D_{N+1}^{\text{Bayes}} = \hat{X}^{\text{Bayes}} = \operatorname{argmin}_{x \in \mathbb{X}} \hat{\xi}_N(x)$$

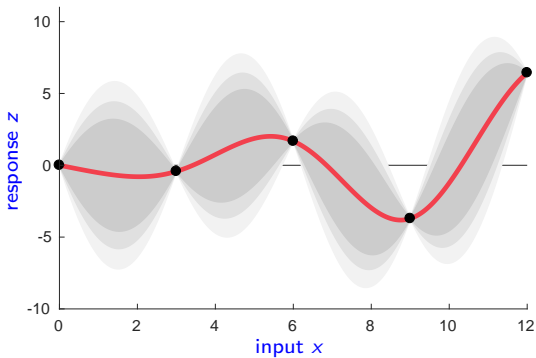
## Example (cont'd): linear loss

Assume that  $n = N = 5$  (a small budget indeed).



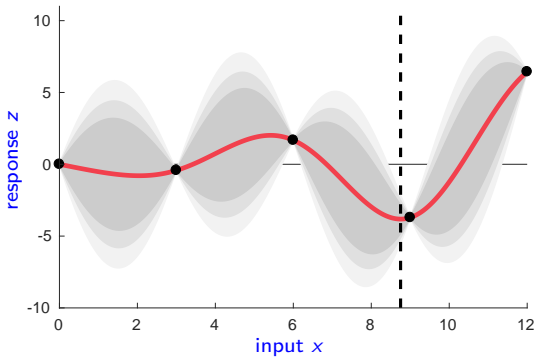
## Example (cont'd): linear loss

Assume that  $n = N = 5$  (a small budget indeed).



## Example (cont'd): linear loss

Assume that  $n = N = 5$  (a small budget indeed).



## Example (cont'd): the $L^1$ loss, variant

- ▶ To summarize, we have for this example

$$\hat{X}^{\text{Bayes}} = \operatorname{argmin} \hat{\xi}_N$$

$$R_N^{\text{Bayes}} = \min \hat{\xi}_N - E_N(\min \xi)$$

- ▶ Remark: in general,  $\hat{X}^{\text{Bayes}} \notin \{X_1, \dots, X_N\}$ 
  - ▶ the value of the function at  $\hat{X}^{\text{Bayes}}$  is not known

## Example (cont'd): the $L^1$ loss, variant

- ▶ To summarize, we have for this example

$$\hat{X}^{\text{Bayes}} = \operatorname{argmin} \hat{\xi}_N$$

$$R_N^{\text{Bayes}} = \min \hat{\xi}_N - E_N(\min \xi)$$

- ▶ Remark: in general,  $\hat{X}^{\text{Bayes}} \notin \{X_1, \dots, X_N\}$ 
  - ▶ the value of the function at  $\hat{X}^{\text{Bayes}}$  is not known

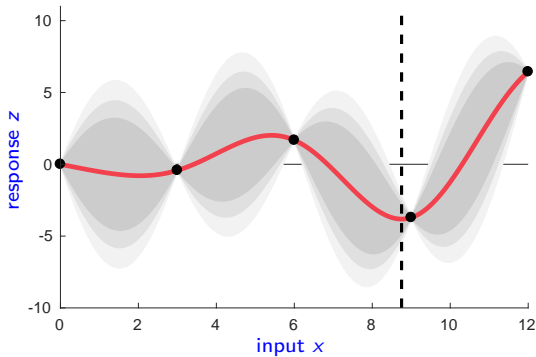
- ▶ Variant: restrict the terminal decision to  $\{X_1, \dots, X_N\}$

$$\hat{X}^{\text{Bayes},1} = \operatorname{argmin}_{x \in \{X_1, \dots, X_N\}} \xi(x)$$

$$R_N^{\text{Bayes},1} = \min_{i \leq N} \xi(X_i) - E_N(\min \xi)$$

## Example (cont'd): linear loss

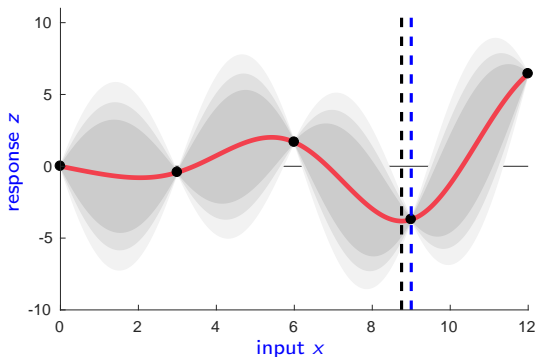
Assume that  $n = N = 5$  (a small budget indeed).





## Example (cont'd): linear loss

Assume that  $n = N = 5$  (a small budget indeed).



Remark: the two estimates are equal when  $\hat{\xi}$  does not “overshoot”  
(e.g., for a Brownian motion prior)

## Lecture 2 : Bayesian optimization (BO)

### 2.2. From Bayes-optimal to myopic strategies

The optimal terminal decision

**Optimal choice of the last evaluation**

Bayes-optimal versus “practical Bayes” optimization

Sampling criteria for multi-objective and/or constrained optimization

## Finding $X_N^{\text{Bayes}}$ (last evaluation point)

- ▶ Let us focus now on the **last evaluation point**
  - ▶ recall that  $\underline{D} = (X_1, \dots, X_{N-1}, X_N, D_{N+1})$

## Finding $X_N^{\text{Bayes}}$ (last evaluation point)

- ▶ Let us focus now on the last evaluation point
  - ▶ recall that  $\underline{D} = (X_1, \dots, X_{N-1}, X_N, D_{N+1})$

### Notation

$E_{n,x}(Y)$  will mean: “compute  $E_n(Y)$ , assuming that  $X_{n+1} = x$ ”

## Finding $X_N^{\text{Bayes}}$ (last evaluation point)

- ▶ Let us focus now on the last evaluation point
  - ▶ recall that  $\underline{D} = (X_1, \dots, X_{N-1}, X_N, D_{N+1})$

### Notation

$E_{n,x}(Y)$  will mean: “compute  $E_n(Y)$ , assuming that  $X_{n+1} = x$ ”

- ▶ For example, if  $Y = g(X_1, Z_1, \dots, X_n, Z_n, X_{n+1}, Z_{n+1})$ ,

$$E_{n,x}(Y) = E_n(g(X_1, Z_1, \dots, X_n, Z_n, x, Z_x))$$

where  $Z_x$  denotes the result of a new evaluation at  $x$

## Finding $X_N^{\text{Bayes}}$ (last evaluation point)

- ▶ Given  $x_N \in \mathbb{X}$ , consider the following strategy at time  $N - 1$ :
  - 1) first, **evaluate at  $X_N = x_N$** ,
  - 2) then, act optimally, i.e., use  **$D_{N+1}^{\text{Bayes}} = \varphi_N^{\text{Bayes}}(F_N)$**

## Finding $X_N^{\text{Bayes}}$ (last evaluation point)

- ▶ Given  $x_N \in \mathbb{X}$ , consider the following strategy at time  $N - 1$ :
  - 1) first, evaluate at  $X_N = x_N$ ,
  - 2) then, act optimally, i.e., use  $D_{N+1}^{\text{Bayes}} = \varphi_N^{\text{Bayes}}(F_N)$
- ▶ The corresponding **posterior risk at time  $N - 1$**  is

$$R_{N-1}(x_N) = \mathbb{E}_{N-1, x_N} \left( L(D_{N+1}^{\text{Bayes}}) \right) = \mathbb{E}_{N-1, x_N} \left( R_N^{\text{Bayes}} \right)$$

## Finding $X_N^{\text{Bayes}}$ (last evaluation point)

- ▶ Given  $x_N \in \mathbb{X}$ , consider the following strategy at time  $N - 1$ :
  - 1) first, evaluate at  $X_N = x_N$ ,
  - 2) then, act optimally, i.e., use  $D_{N+1}^{\text{Bayes}} = \varphi_N^{\text{Bayes}}(F_N)$
- ▶ The corresponding posterior risk at time  $N - 1$  is

$$R_{N-1}(x_N) = \mathbb{E}_{N-1, x_N} \left( L(D_{N+1}^{\text{Bayes}}) \right) = \mathbb{E}_{N-1, x_N} \left( R_N^{\text{Bayes}} \right)$$

- ▶ Claim: the **optimal decision rule** for the last evaluation is

$$X_N^{\text{Bayes}} = \varphi_{N-1}(F_{N-1}) = \operatorname{argmin}_{x_N \in \mathbb{X}} R_{N-1}(x_N)$$



## Finding $X_N^{\text{Bayes}}$ (last evaluation point)

- ▶ Given  $x_N \in \mathbb{X}$ , consider the following strategy at time  $N - 1$ :
  - 1) first, evaluate at  $X_N = x_N$ ,
  - 2) then, act optimally, i.e., use  $D_{N+1}^{\text{Bayes}} = \varphi_N^{\text{Bayes}}(F_N)$

- ▶ The corresponding posterior risk at time  $N - 1$  is

$$R_{N-1}(x_N) = \mathbb{E}_{N-1, x_N} \left( L(D_{N+1}^{\text{Bayes}}) \right) = \mathbb{E}_{N-1, x_N} \left( R_N^{\text{Bayes}} \right)$$

- ▶ Claim: the optimal decision rule for the last evaluation is

$$X_N^{\text{Bayes}} = \varphi_{N-1}(F_{N-1}) = \operatorname{argmin}_{x_N \in \mathbb{X}} R_{N-1}(x_N)$$

- ▶ Remark:  $R_{N-1}$  is used as a “**sampling criterion**” (a.k.a. “infill criterion”, a.k.a. “merit function”...)

## Finding $X_N^{\text{Bayes}}$ : proof

- ▶ For **any strategy**  $\underline{D} = (X_1, \dots, X_{N-1}, X_N, D_{N+1})$ ,

$$\begin{aligned} E_{N-1}(L(D_{N+1})) &= E_{N-1}(R_N(F_N, D_{N+1})) \\ &\geq E_{N-1}(R_N^{\text{Bayes}}(F_N)) = R_{N-1}(F_{N-1}, X_{N-1}) \end{aligned}$$

## Finding $X_N^{\text{Bayes}}$ : proof

- ▶ For any strategy  $\underline{D} = (X_1, \dots, X_{N-1}, X_N, D_{N+1})$ ,

$$\begin{aligned} E_{N-1}(L(D_{N+1})) &= E_{N-1}(R_N(F_N, D_{N+1})) \\ &\geq E_{N-1}(R_N^{\text{Bayes}}(F_N)) = R_{N-1}(F_{N-1}, X_{N-1}) \end{aligned}$$

- ▶ Let  $\underline{D}' = (X_1, \dots, X_{N-1}, X'_N, D'_{N+1})$ ,  
where  $X'_N = \varphi_{N-1}^{\text{Bayes}}(F_{N-1})$  and  $D'_{N+1} = \varphi_N^{\text{Bayes}}(F_{N-1}, X'_N, Z'_N)$

## Finding $X_N^{\text{Bayes}}$ : proof

- ▶ For any strategy  $\underline{D} = (X_1, \dots, X_{N-1}, X_N, D_{N+1})$ ,

$$\begin{aligned} E_{N-1}(L(D_{N+1})) &= E_{N-1}(R_N(F_N, D_{N+1})) \\ &\geq E_{N-1}(R_N^{\text{Bayes}}(F_N)) = R_{N-1}(F_{N-1}, X_{N-1}) \end{aligned}$$

- ▶ Let  $\underline{D}' = (X_1, \dots, X_{N-1}, X'_N, D'_{N+1})$ ,  
where  $X'_N = \varphi_{N-1}^{\text{Bayes}}(F_{N-1})$  and  $D'_{N+1} = \varphi_N^{\text{Bayes}}(F_{N-1}, X'_N, Z'_N)$
- ▶ Then  $E_{N-1}(L(D'_{N+1})) = R_{N-1}^{\text{Bayes}}(F_{N-1}) \leq R_{N-1}(F_{N-1}, X_{N-1})$

## Finding $X_N^{\text{Bayes}}$ : proof

- ▶ For any strategy  $\underline{D} = (X_1, \dots, X_{N-1}, X_N, D_{N+1})$ ,

$$\begin{aligned} E_{N-1}(L(D_{N+1})) &= E_{N-1}(R_N(F_N, D_{N+1})) \\ &\geq E_{N-1}(R_N^{\text{Bayes}}(F_N)) = R_{N-1}(F_{N-1}, X_{N-1}) \end{aligned}$$

- ▶ Let  $\underline{D}' = (X_1, \dots, X_{N-1}, X'_N, D'_{N+1})$ ,  
where  $X'_N = \varphi_{N-1}^{\text{Bayes}}(F_{N-1})$  and  $D'_{N+1} = \varphi_N^{\text{Bayes}}(F_{N-1}, X'_N, Z'_N)$
- ▶ Then  $E_{N-1}(L(D'_{N+1})) = R_{N-1}^{\text{Bayes}}(F_{N-1}) \leq R_{N-1}(F_{N-1}, X_{N-1})$
- ▶ Thus  $E_0(L(D_{N+1})) \geq E_0(L(D'_{N+1}))$  ■

## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

- ▶ Recall our **linear loss** example

$$\hat{X}^{\text{Bayes}} = \operatorname{argmin} \hat{\xi}_N$$

$$R_N^{\text{Bayes}} = \min \hat{\xi}_N - E_N(\min \xi)$$

## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

- ▶ Recall our linear loss example

$$\hat{X}^{\text{Bayes}} = \operatorname{argmin} \hat{\xi}_N$$

$$R_N^{\text{Bayes}} = \min \hat{\xi}_N - E_N(\min \xi)$$

- ▶ Compute the **posterior risk** at time  $N - 1$

$$\begin{aligned} R_{N-1}(F_{N-1}, x_N) &= E_{N-1, x_N} \left( R_N^{\text{Bayes}}(F_N) \right) \\ &= E_{N-1, x_N} \left( \min \hat{\xi}_N \right) - E_{N-1}(\min \xi) \end{aligned}$$

## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

- ▶ Recall our linear loss example

$$\hat{X}^{\text{Bayes}} = \operatorname{argmin} \hat{\xi}_N$$

$$R_N^{\text{Bayes}} = \min \hat{\xi}_N - E_N(\min \xi)$$

- ▶ Compute the posterior risk at time  $N - 1$

$$\begin{aligned} R_{N-1}(F_{N-1}, x_N) &= E_{N-1, x_N} \left( R_N^{\text{Bayes}}(F_N) \right) \\ &= E_{N-1, x_N} \left( \min \hat{\xi}_N \right) - E_{N-1}(\min \xi) \end{aligned}$$

- ▶ The **optimal decision** at time  $N - 1$  is

$$X_N = \operatorname{argmin}_{x_N} E_{N-1, x_N} \left( \min \hat{\xi}_N \right)$$

(first appears (in english) in Mockus, Tiesis & Žilinskas, 1978)



## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

- ▶ Equivalently,

$$X_N = \operatorname{argmax}_{x_N} \underbrace{\min \hat{\xi}_{N-1} - E_{N-1, x_N}(\min \hat{\xi}_N)}_{\rho_{N-1}^{\text{KG}}(x_N) \geq 0}$$

## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

- ▶ Equivalently,

$$X_N = \operatorname{argmax}_{x_N} \underbrace{\min \hat{\xi}_{N-1} - E_{N-1, x_N}(\min \hat{\xi}_N)}_{\rho_{N-1}^{\text{KG}}(x_N) \geq 0}$$

- ▶ Nowadays called the **Knowledge Gradient (KG)** criterion  
(Frazier, Powell & co-authors, 2008, 2009, 2011)

## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

- ▶ Equivalently,

$$X_N = \operatorname{argmax}_{x_N} \underbrace{\min \hat{\xi}_{N-1} - E_{N-1, x_N} \left( \min \hat{\xi}_N \right)}_{\rho_{N-1}^{\text{KG}}(x_N) \geq 0}$$

- ▶ Nowadays called the Knowledge Gradient (KG) criterion  
(Frazier, Powell & co-authors, 2008, 2009, 2011)
- ▶ Remarks
  - ▶ applicable to “noisy” observations as well
  - ▶ a.k.a. simulation-based optimization

## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

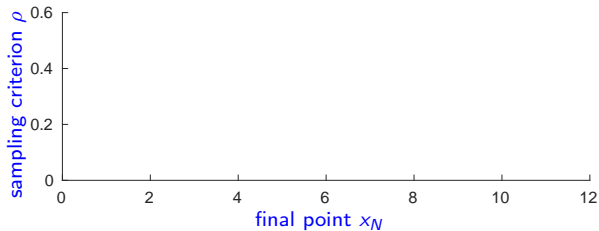
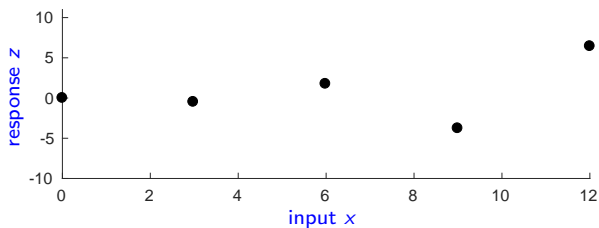
- ▶ Equivalently,

$$X_N = \operatorname{argmax}_{x_N} \underbrace{\min \hat{\xi}_{N-1} - E_{N-1, x_N} \left( \min \hat{\xi}_N \right)}_{\rho_{N-1}^{\text{KG}}(x_N) \geq 0}$$

- ▶ Nowadays called the Knowledge Gradient (KG) criterion  
(Frazier, Powell & co-authors, 2008, 2009, 2011)
- ▶ Remarks
  - ▶ applicable to “noisy” observations as well
    - ▶ a.k.a. simulation-based optimization
  - ▶ even with a GP prior,  $\rho^{\text{KG}}$  is **not exactly computable** in general
    - ▶ idea: approx. max over a finite grid (more about that later)

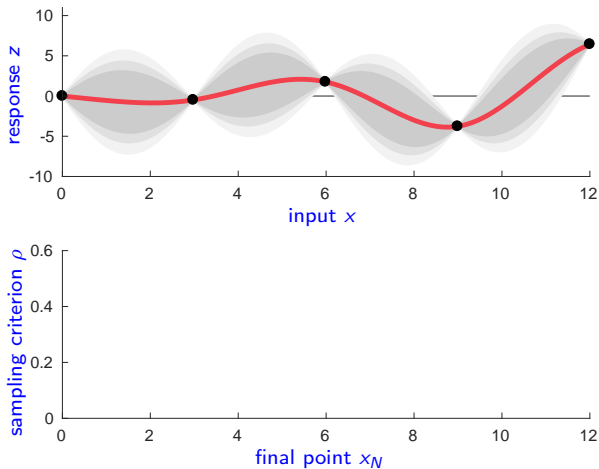
## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

Same example as before,  $n = 5$ , but assume now that  $N = 6$ .



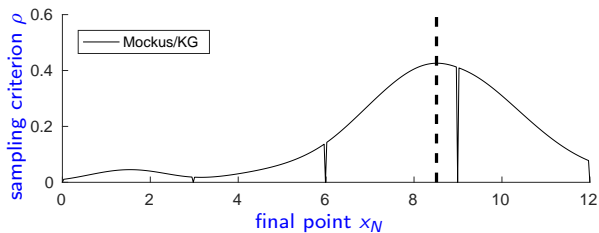
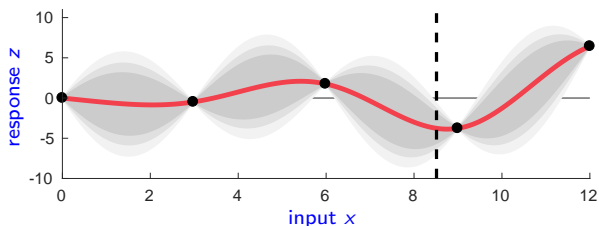
## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

Same example as before,  $n = 5$ , but assume now that  $N = 6$ .



## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

Same example as before,  $n = 5$ , but assume now that  $N = 6$ .



Warning:  $X_N \neq \operatorname{argmax} \hat{\xi}_{N-1}$  (uncertainty is taken into account)

## Finding $X_N^{\text{Bayes}}$ : example, variant (cont'd)

- ▶ Recall the following variant

$$\hat{X}^{\text{Bayes},1} = \operatorname{argmin}_{x \in \{X_1, \dots, X_N\}} \xi(x)$$

$$R_N^{\text{Bayes},1} = \min_{i \leq N} \xi(X_i) - E_N(\min \xi)$$



## Finding $X_N^{\text{Bayes}}$ : example, variant (cont'd)

- ▶ Recall the following variant

$$\hat{X}^{\text{Bayes},1} = \operatorname{argmin}_{x \in \{X_1, \dots, X_N\}} \xi(x)$$

$$R_N^{\text{Bayes},1} = \min_{i \leq N} \xi(X_i) - E_N(\min \xi)$$

- ▶ Set  $M_n = \min_{i \leq n} \xi(X_i)$ . The optimal decision at time  $N - 1$  is

$$X_N = \operatorname{argmax}_{x_N} M_{N-1} - E_{N-1, x_N}(M_N)$$

## Finding $X_N^{\text{Bayes}}$ : example, variant (cont'd)

- ▶ Recall the following variant

$$\hat{X}^{\text{Bayes},1} = \operatorname{argmin}_{x \in \{X_1, \dots, X_N\}} \xi(x)$$

$$R_N^{\text{Bayes},1} = \min_{i \leq N} \xi(X_i) - E_N(\min \xi)$$

- ▶ Set  $M_n = \min_{i \leq n} \xi(X_i)$ . The optimal decision at time  $N - 1$  is

$$\begin{aligned} X_N &= \operatorname{argmax}_{x_N} M_{N-1} - E_{N-1, x_N}(M_N) \\ &= \operatorname{argmax}_{x_N} \underbrace{E_{N-1} \left( (M_{N-1} - \xi(x_N))_+ \right)}_{\rho_n^{\text{EI}}(x_N) \geq 0} \end{aligned}$$

- ▶ This is the **Expected Improvement (EI)** criterion  
(Mockus et al 1978; Jones, Schonlau & Welch, 1998)

## Finding $X_N^{\text{Bayes}}$ : example, variant (cont'd)

- ▶ Recall the following variant

$$\hat{X}^{\text{Bayes},1} = \operatorname{argmin}_{x \in \{X_1, \dots, X_N\}} \xi(x)$$

$$R_N^{\text{Bayes},1} = \min_{i \leq N} \xi(X_i) - E_N(\min \xi)$$

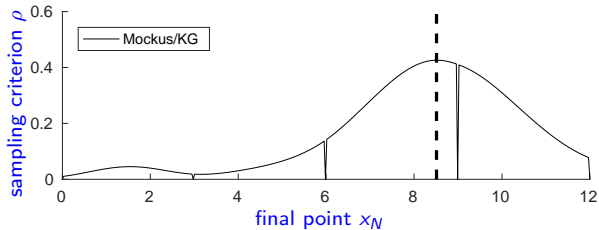
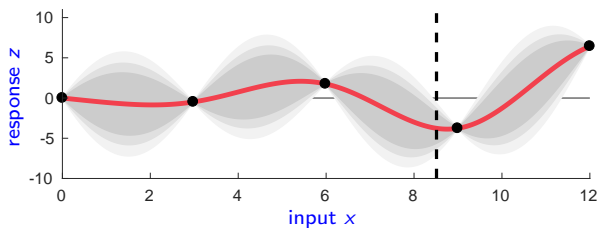
- ▶ Set  $M_n = \min_{i \leq n} \xi(X_i)$ . The optimal decision at time  $N - 1$  is

$$\begin{aligned} X_N &= \operatorname{argmax}_{x_N} M_{N-1} - E_{N-1, x_N}(M_N) \\ &= \operatorname{argmax}_{x_N} \underbrace{E_{N-1} \left( (M_{N-1} - \xi(x_N))_+ \right)}_{\rho_n^{\text{EI}}(x_N) \geq 0} \end{aligned}$$

- ▶ This is the Expected Improvement (EI) criterion  
(Mockus et al 1978; Jones, Schonlau & Welch, 1998)
- ▶ **Computable analytically** for GP priors  $\Rightarrow$  most commonly used  
(for *deterministic* numerical models)

## Finding $X_N^{\text{Bayes}}$ : example (cont'd)

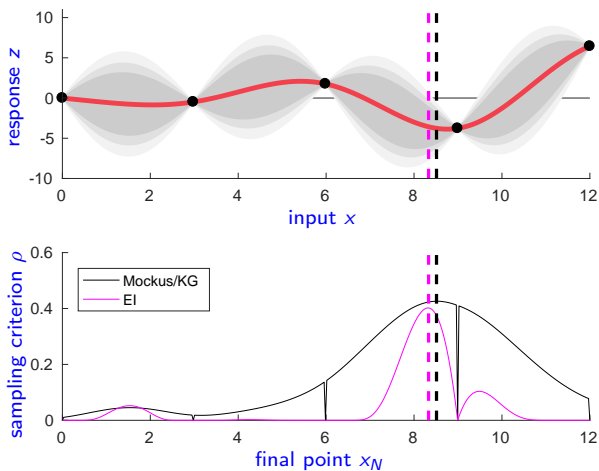
Same example as before,  $n = 5$ , but assume now that  $N = 6$ .



Warning:  $X_N \neq \operatorname{argmax} \hat{\xi}_{N-1}$  (uncertainty is taken into account)

# Finding $X_N^{\text{Bayes}}$ : example (cont'd)

Same example as before,  $n = 5$ , but assume now that  $N = 6$ .



Warning:  $X_N \neq \operatorname{argmax} \hat{\xi}_{N-1}$  (uncertainty is taken into account)

## Lecture 2 : Bayesian optimization (BO)

### 2.2. From Bayes-optimal to myopic strategies

The optimal terminal decision

Optimal choice of the last evaluation

Bayes-optimal versus “practical Bayes” optimization

Sampling criteria for multi-objective and/or constrained optimization

# The Bayes-optimal strategy

- ▶ Recall the **optimal terminal decision rule**

$$\varphi_N^{\text{Bayes}}(F_N) = \operatorname{argmin}_d E_N(L(d))$$

$$R_N^{\text{Bayes}}(F_N) = \min_d E_N(L(d))$$

# The Bayes-optimal strategy

- ▶ Recall the optimal terminal decision rule

$$\varphi_N^{\text{Bayes}}(F_N) = \operatorname{argmin}_d E_N(L(d))$$

$$R_N^{\text{Bayes}}(F_N) = \min_d E_N(L(d))$$

- ▶ Recall the **optimal rule for the last evaluation**

$$\varphi_{N-1}^{\text{Bayes}}(F_{N-1}) = \operatorname{argmin}_{x_N} E_{N-1, x_N} \left( R_N^{\text{Bayes}}(F_N) \right)$$

$$R_{N-1}^{\text{Bayes}}(F_{N-1}) = \min_{x_N} E_{N-1, x_N} \left( R_N^{\text{Bayes}}(F_N) \right)$$



# The Bayes-optimal strategy

- ▶ The entire **Bayes-optimal strategy** can be written similarly:  $\forall n$ ,

$$\varphi_{n-1}^{\text{Bayes}}(F_{n-1}) = \operatorname{argmin}_{x_n} E_{n-1, x_n} \left( R_n^{\text{Bayes}}(F_n) \right)$$

$$R_{n-1}^{\text{Bayes}}(F_{n-1}) = \min_{x_n} E_{n-1, x_n} \left( R_n^{\text{Bayes}}(F_n) \right)$$

- ▶ This is called **backward induction** (or dynamic programming)

# The Bayes-optimal strategy

- ▶ The entire Bayes-optimal strategy can be written similarly:  $\forall n$ ,

$$\varphi_{n-1}^{\text{Bayes}}(F_{n-1}) = \operatorname{argmin}_{x_n} E_{n-1, x_n} \left( R_n^{\text{Bayes}}(F_n) \right)$$

$$R_{n-1}^{\text{Bayes}}(F_{n-1}) = \min_{x_n} E_{n-1, x_n} \left( R_n^{\text{Bayes}}(F_n) \right)$$

- ▶ This is called backward induction (or dynamic programming)
- ▶ So what ? Can we use this ?

# The Bayes-optimal strategy

- ▶ More explicitly, the optimal decision for the first evaluation is

$$X_1 = \operatorname{argmin}_{x_1} E_{0,x_1} \left( \min_{x_2} E_{1,x_2} \left( \dots \right. \right. \\ \left. \left. \min_{x_N} E_{N-1,x_N} \left( \min_d E_N (L(d)) \right) \right) \right)$$

# The Bayes-optimal strategy

- ▶ More explicitly, the optimal decision for the first evaluation is

$$X_1 = \operatorname{argmin}_{x_1} E_{0,x_1} \left( \min_{x_2} E_{1,x_2} \left( \dots \right. \right. \\ \left. \left. \min_{x_N} E_{N-1,x_N} \left( \min_d E_N (L(d)) \right) \right) \right)$$

- ▶ **Very difficult to use in practice** beyond  $N = 1$  or  $2$ 
  - ▶ each “min” is an optim. problem that needs to be solved...
  - ▶ each “ $E_{n,x}$ ” is an integral that needs to be computed...
  - ▶ none of them are tractable, even for the nicest (GP) priors ☹

# Practical Bayesian optimization: myopic strategies

- ▶ Practical BO algorithms use, in general, **myopic strategies**
  - ▶ a.k.a. one-step look-ahead strategies
  - ▶ principle: **make each decision as if it were the last one**
  - ▶ Bayes-optimal if  $N = 1$ , sub-optimal otherwise

# Practical Bayesian optimization: myopic strategies

- ▶ Practical BO algorithms use, in general, myopic strategies
  - ▶ a.k.a. one-step look-ahead strategies
  - ▶ principle: make each decision as if it were the last one
  - ▶ Bayes-optimal if  $N = 1$ , sub-optimal otherwise
- ▶ For any  $n \leq N$ , let  $\bar{L}_n = \min_d E_n(L(d))$

# Practical Bayesian optimization: myopic strategies

- ▶ Practical BO algorithms use, in general, myopic strategies
  - ▶ a.k.a. one-step look-ahead strategies
  - ▶ principle: make each decision as if it were the last one
  - ▶ Bayes-optimal if  $N = 1$ , sub-optimal otherwise
- ▶ For any  $n \leq N$ , let  $\bar{L}_n = \min_d E_n(L(d))$

## Generic myopic BO algorithm

- ▶ For  $n$  from 0 to  $N - 1$ 
  - ▶ Compute  $X_{n+1} = \operatorname{argmin}_x E_{n, x_{n+1}}(\bar{L}_{n+1})$
  - ▶ Make an evaluation at  $X_{n+1}$
- ▶ Output  $D_{N+1} = \operatorname{argmin} E_N(L(d))$

# Practical Bayesian optimization: hyper-parameters

- ▶ GP models have **hyper-parameters**  $\theta$  (variance, range, etc.)
  - ▶ **fully Bayes** approach (see Benassi 2013, chap. III, and refs)
    1. set up prior distributions on the hyper-parameters
    2. use MCMC/SMC to sample from the posterior



# Practical Bayesian optimization: hyper-parameters

- ▶ GP models have **hyper-parameters**  $\theta$  (variance, range, etc.)
  - ▶ **fully Bayes** approach (see Benassi 2013, chap. III, and refs)
    1. set up prior distributions on the hyper-parameters
    2. use MCMC/SMC to sample from the posterior
  - ▶ **plug-in** approach
    - ▶ use  $P_n^\theta \approx \delta_{\hat{\theta}_n}$ , with  $\hat{\theta}_n$  an estimator of  $\theta$  (MML, LOO-CV...)
    - ▶ enough initial data is needed for this approach

# Practical Bayesian optimization: hyper-parameters

- ▶ GP models have hyper-parameters  $\theta$  (variance, range, etc.)
  - ▶ fully Bayes approach (see Benassi 2013, chap. III, and refs)
    1. set up prior distributions on the hyper-parameters
    2. use MCMC/SMC to sample from the posterior
  - ▶ plug-in approach
    - ▶ use  $P_n^\theta \approx \delta_{\hat{\theta}_n}$ , with  $\hat{\theta}_n$  an estimator of  $\theta$  (MML, LOO-CV...)
    - ▶ enough initial data is needed for this approach

## Generic myopic BO algorithm with hyper-parameter estimation

- ▶ Init: (space-filling) DoE of size  $n_0$  (rule of thumb:  $n_0 = 10d$ )
- ▶ For  $n$  from  $n_0$  to  $N - 1$ 
  - ▶ once in a while, Estimate hyper-parameters (plug-in/fully Bayes)
  - ▶ Compute  $X_{n+1} = \operatorname{argmin}_x E_{n, x_{n+1}}(\bar{L}_{n+1})$
  - ▶ Make an evaluation at  $X_{n+1}$
- ▶ Output  $D_{N+1} = \operatorname{argmin} E_N(L(d))$

# Practical Bayesian optimization: EGO

## STK demo

... single-objective box-constrained optimization  
with the EI criterion and a plug-in approach  
(a.k.a. the “EGO” algorithm) ...

## Practical Bayesian optimization: optimization

- ▶ Each iteration involves an **auxiliary optimization problem**

# Practical Bayesian optimization: optimization

- ▶ Each iteration involves an auxiliary optimization problem
- ▶ Various approaches to solve it
  - ▶ **Fix grid** or **IID random search**
    - ▶ OK for low-dimensional, simple problems
    - ▶ if accurate convergence is not needed

# Practical Bayesian optimization: optimization

- ▶ Each iteration involves an auxiliary optimization problem
- ▶ Various approaches to solve it
  - ▶ **Fix grid** or **IID random search**
    - ▶ OK for low-dimensional, simple problems
    - ▶ if accurate convergence is not needed
  - ▶ **External solvers**
    - ▶ ex: DiceOptim → Rgenoud (genetic + gradient)
    - ▶ ex: Janusvekis & Le Riche (2013) → CMA-ES

# Practical Bayesian optimization: optimization

- ▶ Each iteration involves an auxiliary optimization problem
- ▶ Various approaches to solve it
  - ▶ **Fix grid** or **IID random search**
    - ▶ OK for low-dimensional, simple problems
    - ▶ if accurate convergence is not needed
  - ▶ **External solvers**
    - ▶ ex: DiceOptim → Rgenoud (genetic + gradient)
    - ▶ ex: Janusvekis & Le Riche (2013) → CMA-ES
  - ▶ **Sequential Monte Carlo** (Benassi, 2013; Feliot et al, 2017)
    - ▶ sample according to a well-chosen sequence of densities

# Practical Bayesian optimization: optimization

- ▶ Each iteration involves an auxiliary optimization problem
- ▶ Various approaches to solve it
  - ▶ **Fix grid** or **IID random search**
    - ▶ OK for low-dimensional, simple problems
    - ▶ if accurate convergence is not needed
  - ▶ **External solvers**
    - ▶ ex: DiceOptim  $\rightarrow$  Rgenoud (genetic + gradient)
    - ▶ ex: Janusvekis & Le Riche (2013)  $\rightarrow$  CMA-ES
  - ▶ **Sequential Monte Carlo** (Benassi, 2013; Feliot et al, 2017)
    - ▶ sample according to a well-chosen sequence of densities
- ▶ Bayesian optimization  $\Rightarrow$  run-time **overhead**
  - ▶ depends on the model, sampling criterion, optimizer, etc.
  - ▶ BO is appropriate for **expensive-to-evaluate** numerical models



## Lecture 2 : Bayesian optimization (BO)

### 2.2. From Bayes-optimal to myopic strategies

The optimal terminal decision

Optimal choice of the last evaluation

Bayes-optimal versus “practical Bayes” optimization

Sampling criteria for multi-objective and/or constrained optimization

## Multi-objective problems

- ▶ **Several objective functions** to be minimized:  $f = (f_1, \dots, f_p)$ 
  - ▶  $f_j : \mathbb{X} \rightarrow \mathbb{R}, 1 \leq j \leq p$

# Multi-objective problems

- ▶ Several objective functions to be minimized:  $f = (f_1, \dots, f_p)$ 
  - ▶  $f_j : \mathbb{X} \rightarrow \mathbb{R}, 1 \leq j \leq p$

## Pareto domination relation

$$z \prec z' \quad \text{if (def)} \quad \begin{cases} z_j \leq z'_j & \text{for all } j \leq p, \\ z_j < z'_j & \text{for at least one } j \leq p. \end{cases}$$

# Multi-objective problems

- ▶ Several objective functions to be minimized:  $f = (f_1, \dots, f_p)$ 
  - ▶  $f_j : \mathbb{X} \rightarrow \mathbb{R}, 1 \leq j \leq p$

## Pareto domination relation

$$z \prec z' \quad \text{if (def)} \quad \begin{cases} z_j \leq z'_j & \text{for all } j \leq p, \\ z_j < z'_j & \text{for at least one } j \leq p. \end{cases}$$

- ▶ The goal is to find (estimate)
  - ▶ the **Pareto set**  $\mathbb{P} = \{x \in \mathbb{X} : \nexists x' \in \mathbb{X}, f(x') \prec f(x)\}$   
(a.k.a. set of Pareto-efficient solutions)

# Multi-objective problems

- ▶ Several objective functions to be minimized:  $f = (f_1, \dots, f_p)$ 
  - ▶  $f_j : \mathbb{X} \rightarrow \mathbb{R}, 1 \leq j \leq p$

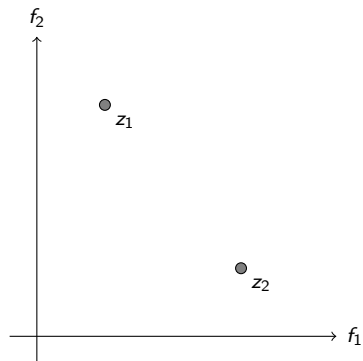
## Pareto domination relation

$$z \prec z' \quad \text{if (def)} \quad \begin{cases} z_j \leq z'_j & \text{for all } j \leq p, \\ z_j < z'_j & \text{for at least one } j \leq p. \end{cases}$$

- ▶ The goal is to find (estimate)
  - ▶ the **Pareto set**  $\mathbb{P} = \{x \in \mathbb{X} : \nexists x' \in \mathbb{X}, f(x') \prec f(x)\}$   
(a.k.a. set of Pareto-efficient solutions)
  - ▶ and/or the **Pareto front**  $\{z \in \mathbb{R}^p : \exists x \in \mathbb{P}, z = f(x)\}$   
(a.k.a. Pareto frontier, Pareto boundary...)

# Multi-objective problems

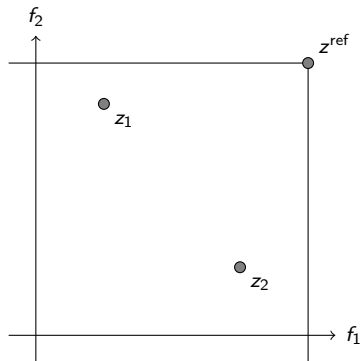
- ▶ EHVI: a natural extension of EI (Emmerich et al, 2006)



Noiseless evaluations

# Multi-objective problems

- ▶ EHVI: a natural extension of EI (Emmerich et al, 2006)

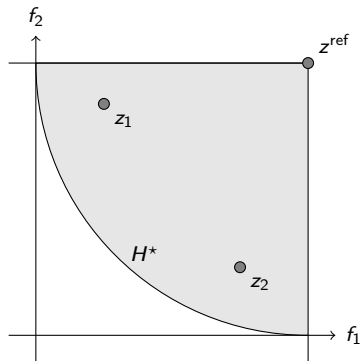


Noiseless evaluations

$$\mathbb{B} = \prod_{j=1}^p ]-\infty; z_j^{\text{ref}}]: \text{ bounding box}$$

# Multi-objective problems

- ▶ EHVI: a natural extension of EI (Emmerich et al, 2006)



Noiseless evaluations

$\mathbb{B} = \prod_{j=1}^p ]-\infty; z_j^{\text{ref}}]$ : bounding box

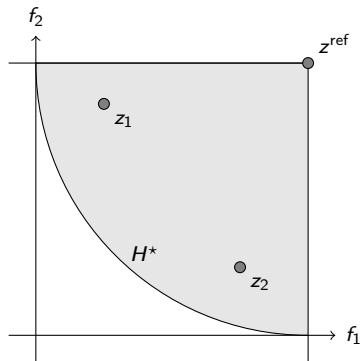
True **dominated region**:

$$H^*(f) = \{z \in \mathbb{B}, \exists x \in \mathbb{X}, f(x) \preceq z\}$$



# Multi-objective problems

- ▶ EHVI: a natural extension of EI (Emmerich et al, 2006)



Noiseless evaluations

$\mathbb{B} = \prod_{j=1}^p ]-\infty; z_j^{\text{ref}}]$ : bounding box

True dominated region:

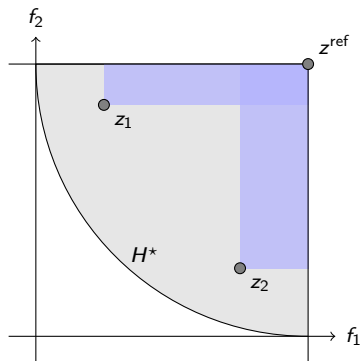
$$H^*(f) = \{z \in \mathbb{B}, \exists x \in \mathbb{X}, f(x) \preceq z\}$$

Loss function:

$$L(f, \hat{H}) = |H^*(f) \Delta \hat{H}|$$

# Multi-objective problems

- ▶ EHVI: a natural extension of EI (Emmerich et al, 2006)



Noiseless evaluations

$\mathbb{B} = \prod_{j=1}^p ]-\infty; z_j^{\text{ref}}]$ : bounding box

True dominated region:

$$H^*(f) = \{z \in \mathbb{B}, \exists x \in \mathbb{X}, f(x) \preceq z\}$$

Loss function:

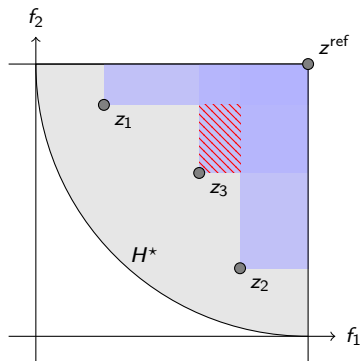
$$L(f, \hat{H}) = |H^*(f) \Delta \hat{H}|$$

Best "safe" estimator:

$$H_n = \{z \in \mathbb{B}, \exists i \leq n, f(X_i) \preceq z\}$$

# Multi-objective problems

- ▶ EHVI: a natural extension of EI (Emmerich et al, 2006)



Noiseless evaluations

$\mathbb{B} = \prod_{j=1}^p ]-\infty; z_j^{\text{ref}}]$ : bounding box

True dominated region:

$$H^*(f) = \{z \in \mathbb{B}, \exists x \in \mathbb{X}, f(x) \preceq z\}$$

Loss function:

$$L(f, \hat{H}) = |H^*(f) \Delta \hat{H}|$$

Best "safe" estimator:

$$H_n = \{z \in \mathbb{B}, \exists i \leq n, f(X_i) \preceq z\}$$

$$\rho_n^{\text{EHVI}}(x_{n+1}) = E_{n, x_{n+1}} ( |H_{n+1} \setminus H_n| )$$

# Multi-objective problems

- ▶ Implementation

- ▶ Exactly computable for independent GP priors,  $2 \leq p \lesssim 5$
- ▶ **Implemented** in STK (Matlab/Octave), GPareto (R)...
- ▶ Dependent priors, larger  $p$ : Monte Carlo approx.

# Multi-objective problems

- ▶ Implementation
  - ▶ Exactly computable for independent GP priors,  $2 \leq p \lesssim 5$
  - ▶ Implemented in STK (Matlab/Octave), GPareto (R)...
  - ▶ Dependent priors, larger  $p$ : Monte Carlo approx.
- ▶ **Many other sampling criteria** have been proposed
  - ▶ See Feliot et al (2017, section 2.2) and references therein

# Multi-objective problems

- ▶ Implementation
  - ▶ Exactly computable for independent GP priors,  $2 \leq p \lesssim 5$
  - ▶ Implemented in STK (Matlab/Octave), GPareto (R)...
  - ▶ Dependent priors, larger  $p$ : Monte Carlo approx.
- ▶ Many other sampling criteria have been proposed
  - ▶ See Feliot et al (2017, section 2.2) and references therein

## STK demo

... bi-objective optimization with the EHVI criterion ...

*code by Etienne Leloup, Guillaume Maistre-Bazin, Lucain Pouget  
CentraleSupélec final year project for CEA DIF*

# Inequality-constrained problems

- ▶ **Single-objective, inequality-constrained** problem:
  - ▶  $f = (f_o, f_{c,1}, \dots, f_{c,q})$ , with
  - ▶  $f_o : \mathbb{X} \rightarrow \mathbb{R}$ , to be minimized,
  - ▶  $f_{c,j} : \mathbb{X} \rightarrow \mathbb{R}$ ,  $1 \leq j \leq q$ , must be  $\leq 0$ .

# Inequality-constrained problems

- ▶ Single-objective, inequality-constrained problem:
  - ▶  $f = (f_o, f_{c,1}, \dots, f_{c,q})$ , with
  - ▶  $f_o : \mathbb{X} \rightarrow \mathbb{R}$ , to be minimized,
  - ▶  $f_{c,j} : \mathbb{X} \rightarrow \mathbb{R}$ ,  $1 \leq j \leq q$ , must be  $\leq 0$ .
- ▶ Consider the following **loss function**

$$L(f, \hat{x}) = \begin{cases} f_o(\hat{x}) - f_o^* & \text{if } f_c(\hat{x}) \leq 0, \\ +\infty & \text{otherwise.} \end{cases}$$

where  $f_o^* = \min_{x: f_c(x) \leq 0} f_o(x)$



# Inequality-constrained problems

- ▶ Assuming
  - ▶ noiseless evaluations,
  - ▶ independent priors on objective and constraint functions,
  - ▶  $\exists i \leq n, \xi_c(X_i) = f_c(X_i) \leq 0$ ,

the following myopic criterion follows (Schonlau et al, 1998)

$$\rho_n^{\text{EIC}}(x_{n+1}) = \rho_{o,n}^{\text{EI}}(x_{n+1}) \cdot \underbrace{\prod_{j=1}^q P_n(\xi_{c,j}(x_{n+1}) \leq 0)}_{\text{Proba of Feasibility (PF)}}.$$

# Inequality-constrained problems

- ▶ Assuming

- ▶ noiseless evaluations,
- ▶ independent priors on objective and constraint functions,
- ▶  $\exists i \leq n, \xi_c(X_i) = f_c(X_i) \leq 0$ ,

the following myopic criterion follows (Schonlau et al, 1998)

$$\rho_n^{\text{EIC}}(x_{n+1}) = \rho_{o,n}^{\text{EI}}(x_{n+1}) \cdot \underbrace{\prod_{j=1}^q P_n(\xi_{c,j}(x_{n+1}) \leq 0)}_{\text{Proba of Feasibility (PF)}}.$$

- ▶ Implementation

- ▶ **Easy for independent GP priors** (most commonly used)
- ▶ Dependent priors: harder... (but see Williams et al, 2010)

# Inequality-constrained problems

- ▶ Assuming
  - ▶ noiseless evaluations,
  - ▶ independent priors on objective and constraint functions,
  - ▶  $\exists i \leq n, \xi_c(X_i) = f_c(X_i) \leq 0$ ,

the following myopic criterion follows (Schonlau et al, 1998)

$$\rho_n^{\text{EIC}}(x_{n+1}) = \rho_{0,n}^{\text{EI}}(x_{n+1}) \cdot \underbrace{\prod_{j=1}^q P_n(\xi_{c,j}(x_{n+1}) \leq 0)}_{\text{Proba of Feasibility (PF)}}.$$

- ▶ Implementation
  - ▶ Easy for independent GP priors (most commonly used)
  - ▶ Dependent priors: harder... (but see Williams et al, 2010)
- ▶ Again, **many other approaches** have been proposed
  - ▶ See Feliot et al (2017, section 2.3) and references therein

# Et maintenant une page de pub !

- ▶ BMOO algorithm (Feliot et al 2017)
  - ▶ **Unified EI/EHVI/EIC criterion**
    - ▶ well-defined even when no feasible point is known
  - ▶ **Efficient SMC technique** for criterion optimization
    - ▶ SMC = Sequential Monte Carlo
    - ▶ extends the work of Benassi (2013)

# Et maintenant une page de pub !

- ▶ BMOO algorithm (Feliot et al 2017)
  - ▶ Unified EI/EHVI/EIC criterion
    - ▶ well-defined even when no feasible point is known
  - ▶ Efficient SMC technique for criterion optimization
    - ▶ SMC = Sequential Monte Carlo
    - ▶ extends the work of Benassi (2013)

## Announcement

Paul Feliot's PhD defense will take place

on **Wednesday, July 12, 2017**, 2 PM,

at CentraleSupélec (Gif). Venez nombreux !

## Miscellaneous references for further reading

- ▶ **Information-based** BO: a different approach
  - ▶ Risk = **entropy** of the minimizer
  - ▶ See Villemonteix et al (2009), Hennig & Schueller (2012), Hernandez-Lobáto and co-authors (2014, 2015...)

## Miscellaneous references for further reading

- ▶ Information-based BO: a different approach
  - ▶ Risk = entropy of the minimizer
  - ▶ See Villemonteix et al (2009), Hennig & Schueller (2012), Hernandez-Lobáto and co-authors (2014, 2015...)
- ▶ Aggregation-based approaches
  - ▶ **Multi-objective**: ParEGO (Knowles, 2006)
  - ▶ **Constrained**: Augmented Lagrangian methods (Gramacy et al, 2016; Picheny et al, 2016)

## Miscellaneous references for further reading

- ▶ Information-based BO: a different approach
  - ▶ Risk = entropy of the minimizer
  - ▶ See Villemonteix et al (2009), Hennig & Schueller (2012), Hernandez-Lobáto and co-authors (2014, 2015...)
- ▶ Aggregation-based approaches
  - ▶ Multi-objective: ParEGO (Knowles, 2006)
  - ▶ Constrained: Augmented Lagrangian methods (Gramacy et al, 2016; Picheny et al, 2016)
- ▶ Batch of evaluations: **multi-point criteria**
  - ▶ Ginsbourger et al (2010), Chevalier & Ginsbourger (2013), Chevalier et al (2014), Marmin et al (2015)



## Miscellaneous references for further reading

- ▶ Information-based BO: a different approach
  - ▶ Risk = entropy of the minimizer
  - ▶ See Villemonteix et al (2009), Hennig & Schueller (2012), Hernandez-Lobáto and co-authors (2014, 2015...)
- ▶ Aggregation-based approaches
  - ▶ Multi-objective: ParEGO (Knowles, 2006)
  - ▶ Constrained: Augmented Lagrangian methods (Gramacy et al, 2016; Picheny et al, 2016)
- ▶ Batch of evaluations: multi-point criteria
  - ▶ Ginsbourger et al (2010), Chevalier & Ginsbourger (2013), Chevalier et al (2014), Marmin et al (2015)
- ▶ **Noisy evaluations** / stochastic simulators
  - ▶ will be discussed in the next part 😊

## Lecture 2 : Bayesian optimization (BO)

2.1. Decision-theoretic framework

2.2. From Bayes-optimal to myopic strategies

2.3. Design under uncertainty

## Lecture 2 : Bayesian optimization (BO)

### 2.3. Design under uncertainty

Overview of possible approaches

Optimization of a mean response

RBDO (and other formulations)

# Design under uncertainty

- ▶ Standard **design optimization** problem:
  - ▶ Minimize one **objective** (“cost”) **function**  $f_o(x)$
  - ▶ or several objective functions  $f_{o,1}(x), \dots, f_{o,p}(x)$
  - ▶ under the **constraints**  $f_{c,j}(x) \leq 0, 1 \leq j \leq q$
- ▶ Some objective/constraint functions are **expensive to evaluate**

# Design under uncertainty

- ▶ Standard design optimization problem:
  - ▶ Minimize one objective (“cost”) function  $f_o(x)$
  - ▶ or several objective functions  $f_{o,1}(x), \dots, f_{o,p}(x)$
  - ▶ under the constraints  $f_{c,j}(x) \leq 0, 1 \leq j \leq q$
- ▶ Some objective/constraint functions are expensive to evaluate

## “Design under uncertainty” framework

- ▶ objective functions:  $f_{o,j}(x, u), 1 \leq j \leq p$
- ▶ constraint functions:  $f_{c,j}(x, u), 1 \leq j \leq q$
- ▶ where  $u$  denotes **factors that the designer cannot control**

## (a few words on the) Worst-case approach

- ▶ Principle of the **worst-case** (minimax) approach
  - ▶ Define an **uncertainty set**  $\mathbb{U}$
  - ▶ Optimize by considering the **worst**  $u \in \mathbb{U}$

## (a few words on the) Worst-case approach

- ▶ Principle of the worst-case (**minimax**) approach
  - ▶ Define an uncertainty set  $\mathbb{U}$
  - ▶ Optimize by considering the worst  $u \in \mathbb{U}$
- ▶ For instance, assuming a single-objective problem:

$$\text{minimize } \max_{u \in \mathbb{U}} f_o(x, u)$$

## (a few words on the) Worst-case approach

- ▶ Principle of the worst-case (minimax) approach
  - ▶ Define an uncertainty set  $\mathbb{U}$
  - ▶ Optimize by considering the worst  $u \in \mathbb{U}$
- ▶ For instance, assuming a single-objective problem:

$$\text{minimize } \max_{u \in \mathbb{U}} f_o(x, u)$$

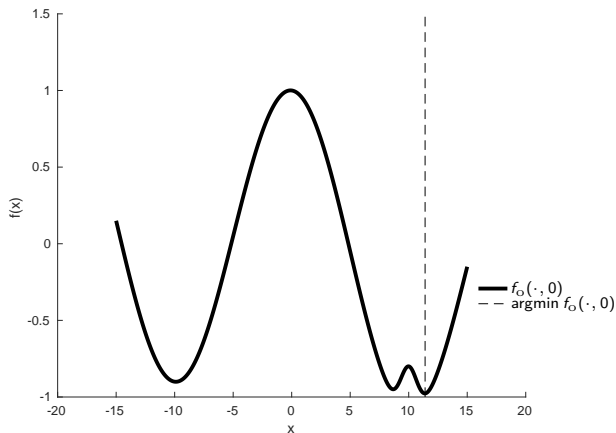
- ▶ If the problem has constraints, they become:

$$\forall j \leq q, \forall u \in \mathbb{U}, f_{c,j}(x, u) \leq 0$$



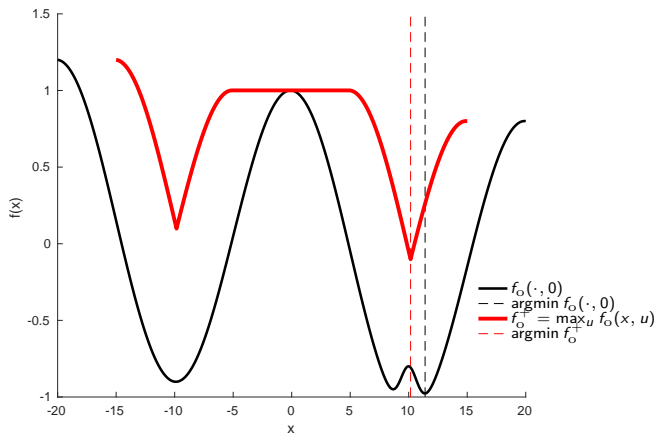
## Example 1: Illustration of the worst-case approach

Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$



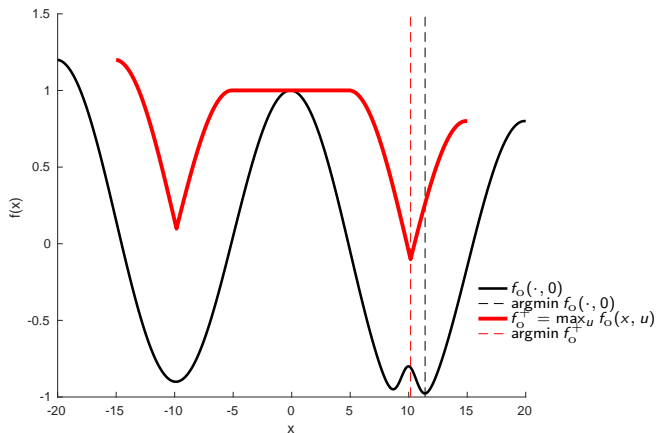
## Example 1: Illustration of the worst-case approach

Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$



## Example 1: Illustration of the worst-case approach

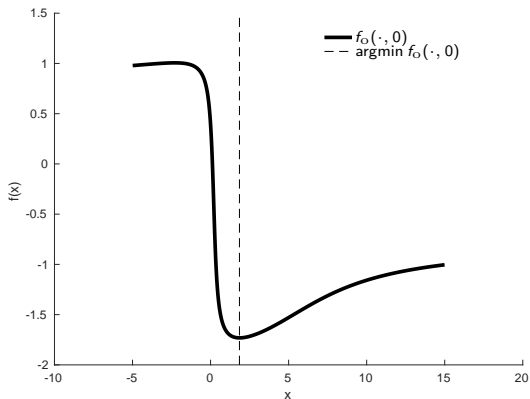
Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$



Remark: very conservative, the **nominal performance** is ignored

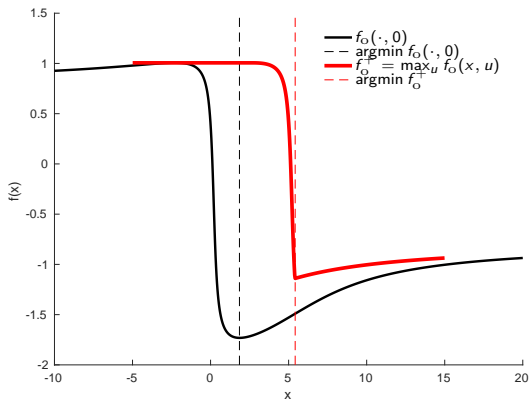
## Example 2: Illustration of the worst-case approach

Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$



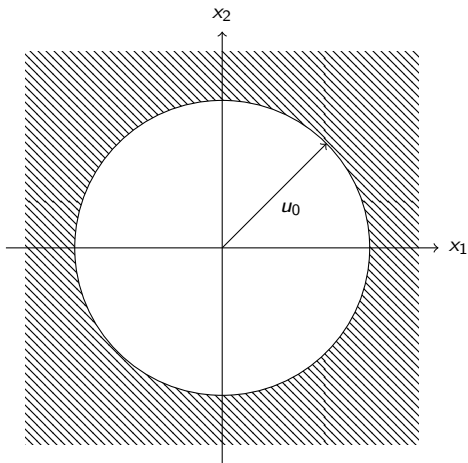
## Example 2: Illustration of the worst-case approach

Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$



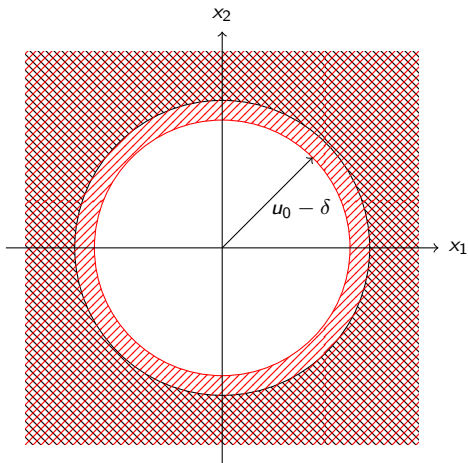
## Another example: worst-case approach for a constraint

Example:  $f_c(x, u) = \|x\|^2 - u^2$



## Another example: worst-case approach for a constraint

Example:  $f_c(x, u) = \|x\|^2 - u^2$ , with  $u \in \mathcal{U} = [u_0 - \delta; u_0 + \delta]$



## (a few more words on the) Worst-case approach

- ▶ In **this lecture**, we will **focus on the probabilistic** approach



## (a few more words on the) Worst-case approach

- ▶ In this lecture, we will focus on the probabilistic approach
- ▶ See Marzat, Walter & Piet-Lahanier (2013, 2016) for a “BO treatment” of the worst-case approach (using relaxation)

## (a few more words on the) Worst-case approach

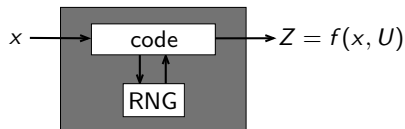
- ▶ In this lecture, we will focus on the probabilistic approach
- ▶ See Marzat, Walter & Piet-Lahanier (2013, 2016) for a “BO treatment” of the worst-case approach (using relaxation)
- ▶ An issue of terminology: in the math literature,
  - ▶ “robust optimization” refers mainly to the worst-case setting (see Ben Tal et al (2009), Bertsimas et al (2011) and refs)
  - ▶ the probabilistic approach is called stochastic programming
- ▶ while engineers use the word “robust” for both 😊

# The probabilistic approach

- ▶ From now, we focus on the **probabilistic approach**
  - ▶  $u$  is considered as **random**  $\rightarrow U \sim P^U$
  - ▶ can be a random vector ( $\in \mathbb{R}^m$ ), or a more complicated object

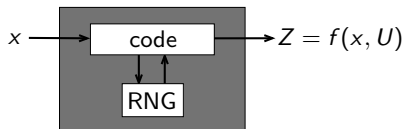


## The “stochastic simulator” setting



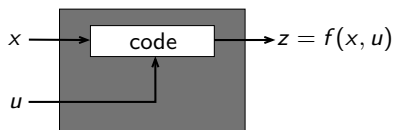
- ▶ Features of the black box
  - ▶  $U$  is **not directly accessible**
  - ▶ only  $x$  **can be chosen** by the algorithm;  $Z = f(x, U)$  is observed
  - ▶  $P^U$  is **not known explicitly**

# The “stochastic simulator” setting



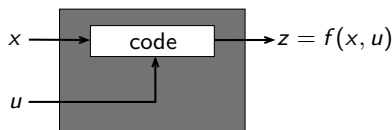
- ▶ Features of the black box
  - ▶  $U$  is not directly accessible
  - ▶ only  $x$  can be chosen by the algorithm;  $Z = f(x, U)$  is observed
  - ▶  $P^U$  is not known explicitly
- ▶ “State of nature” (the things that we don’t know)
  - ▶ the family  $(P_x^Z)$  of conditional distributions
  - ▶ the RV  $U_1, U_2, \dots$  that will be generated when running the computer model with inputs  $x_1, x_2, \dots$
  - ▶ the RV  $U_{\text{real}}$  that defines the  $f(x, U_{\text{real}})$  ultimately realized

# The “environmental variables” setting



- ▶ Features of the black box
  - ▶ the simulator remains **deterministic**
  - ▶  $P^U$  is specified separately, usually **explicitly**
  - ▶ the algorithm can **choose  $(x, u)$  pairs** to be evaluated

# The “environmental variables” setting



- ▶ Features of the black box
  - ▶ the simulator remains deterministic
  - ▶  $P^U$  is specified separately, usually explicitly
  - ▶ the algorithm can choose  $(x, u)$  pairs to be evaluated
- ▶ “State of nature” (the things that we don’t know)
  - ▶ the **deterministic function**  $f : (x, u) \mapsto f(x, u)$
  - ▶ the RV  $U_{\text{real}}$  that defines the  $f(x, U_{\text{real}})$  ultimately realized





# Problem formulations

- ▶ Various “robust” formulations can be considered for the design problem, depending mainly on
  - ▶ the number of objective functions,
  - ▶ the presence of (expensive-to-evaluate) constraints,
  - ▶ and, of course, how we want to deal with  $U_{\text{real}}$ .

# Problem formulations

- ▶ Various “robust” formulations can be considered for the design problem, depending mainly on
  - ▶ the number of objective functions,
  - ▶ the presence of (expensive-to-evaluate) constraints,
  - ▶ and, of course, how we want to deal with  $U_{\text{real}}$ .
- ▶ In the following, we focus on
  - ▶ **single objective** problems
  - ▶ in the “**environmental variables**” setting
- ▶ and discuss two important cases:
  - ▶ optimization of the **averaged objective** function
  - ▶ **reliability-based design optimization** (RBDO), a.k.a. “chance constrained” optimization, and other formulations

## Lecture 2 : Bayesian optimization (BO)

### 2.3. Design under uncertainty

Overview of possible approaches

Optimization of a mean response

RBDO (and other formulations)

# Optimization of a mean response

- ▶ Assume

- ▶ **single objective**  $f = f_o$ , expensive to evaluate
- ▶ **no** (expensive-to-evaluate) **constraints**
- ▶ remark: cheap constraints allowed in the definition of  $\mathbb{X} \subset \mathbb{R}^d$
- ▶ “**environmental variables**” setting

# Optimization of a mean response

- ▶ Assume
  - ▶ single objective  $f = f_o$ , expensive to evaluate
  - ▶ no (expensive-to-evaluate) constraints
  - ▶ remark: cheap constraints allowed in the definition of  $\mathbb{X} \subset \mathbb{R}^d$
  - ▶ “environmental variables” setting
  
- ▶ Consider once again the  $L^1$  loss function

$$\begin{aligned} L((f, u_{\text{real}}), \hat{x}) &= \left| f(\hat{x}, u_{\text{real}}) - \min_x f(x, u_{\text{real}}) \right| \\ &= f(\hat{x}, u_{\text{real}}) - \min_x f(x, u_{\text{real}}) \end{aligned}$$

## Optimization of a mean response

- ▶ Compute the **posterior risk** at time  $N$  for an estimate  $\hat{x} \in \mathbb{X}$

$$E_N(L((\xi, U_{\text{real}}), \hat{x})) = E_N(\xi(\hat{x}, U_{\text{real}})) - E_N(\min \xi(\cdot, U_{\text{real}}))$$

## Optimization of a mean response

- ▶ Compute the posterior risk at time  $N$  for an estimate  $\hat{x} \in \mathbb{X}$

$$\begin{aligned} E_N(L((\xi, U_{\text{real}}), \hat{x})) &= E_N(\xi(\hat{x}, U_{\text{real}})) - E_N(\min \xi(\cdot, U_{\text{real}})) \\ &= E_N(\bar{\xi}(\hat{x})) - E_N(\min \xi(\cdot, U_{\text{real}})) \end{aligned}$$

where  $\bar{\xi}(x) = \int \xi(x, u) P^U(du)$



## Optimization of a mean response

- ▶ Compute the posterior risk at time  $N$  for an estimate  $\hat{x} \in \mathbb{X}$

$$\begin{aligned} E_N(L((\xi, U_{\text{real}}), \hat{x})) &= E_N(\xi(\hat{x}, U_{\text{real}})) - E_N(\min \xi(\cdot, U_{\text{real}})) \\ &= E_N(\bar{\xi}(\hat{x})) - E_N(\min \xi(\cdot, U_{\text{real}})) \end{aligned}$$

where  $\bar{\xi}(x) = \int \xi(x, u) P^U(du)$

- ▶ Same  $L^1$  risk (ignoring last term) as if we were dealing with the

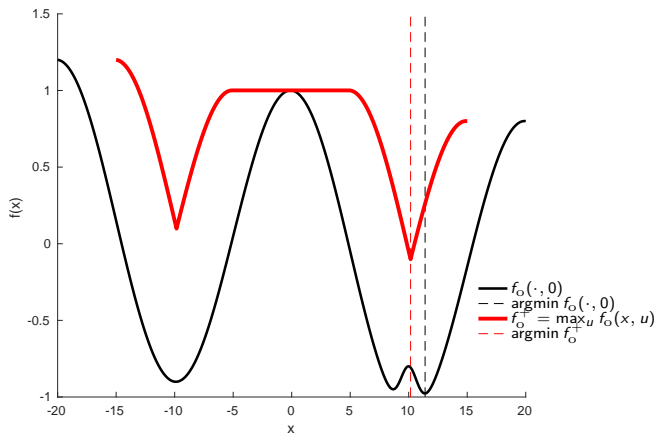
Equivalent “deterministic” problem

$$\min_x \bar{f}(x), \quad \text{with } \bar{f}(x) = \int f(x, u) P^U(du)$$

(Remark: this formulation occurs very naturally in a BO framework 😊)

## Example 1: Illustration of the worst-case approach

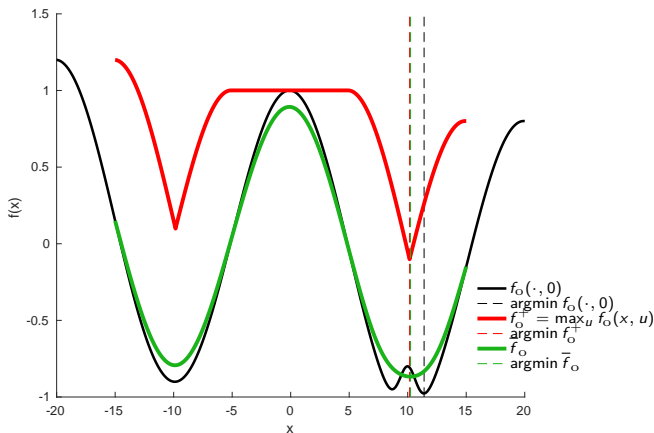
Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$



Remark: very conservative, the **nominal performance** is ignored

## Example 1: Worst-case versus probabilistic approach

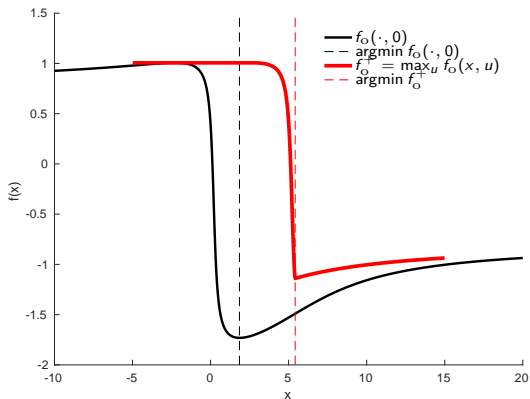
Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$



$\bar{f}_o = E(f_o(\cdot, U))$ , with  $U \sim \mathcal{N}(0, s^2)$ , s.t.  $P(|U| \leq \delta) = 99.9\%$

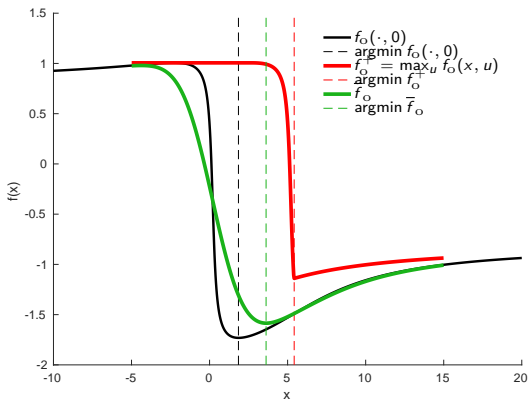
## Example 2: Illustration of the worst-case approach

Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$



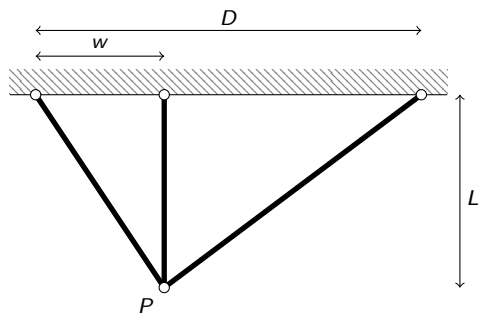
## Example 2: Worst-case versus probabilistic approach

Example:  $f_o(x, u) = \tilde{f}(x + u)$ , with  $u \in \mathbb{U} = [-\delta; \delta]$ ,  $\delta = 5$

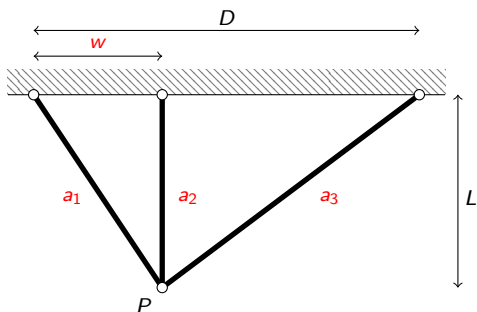


$\bar{f}_o = E(f_o(\cdot, U))$ , with  $U \sim \mathcal{N}(0, s^2)$ , s.t.  $P(|U| \leq \delta) = 99.9\%$

### Example 3: Three-bar truss (Koski, 1985; Das, 1997)

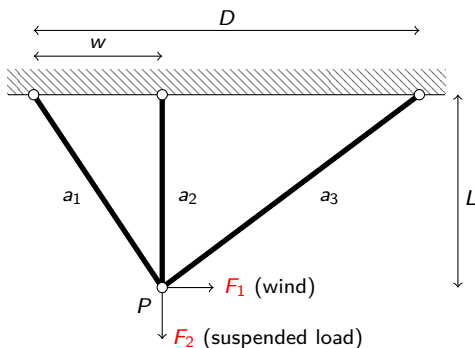


### Example 3: Three-bar truss (Koski, 1985; Das, 1997)



- ▶ Design variables:  $x = (a_1, a_2, a_3, w)$ 
  - ▶  $a_j$ : cross-section of bar  $j$

### Example 3: Three-bar truss (Koski, 1985; Das, 1997)



- ▶ Design variables:  $x = (a_1, a_2, a_3, w)$ 
  - ▶  $a_j$ : cross-section of bar  $j$
- ▶ Environmental variables:  $U = (F_1, F_2)$



## Example 3: Three-bar truss (Koski, 1985; Das, 1997)

- ▶ Our (supposedly expensive) numerical model computes
  - ▶ the displacement  $y = (y_1, y_2)$  of point  $P$ ,
  - ▶ the stress  $\sigma_j$  in each bar ( $1 \leq j \leq 3$ ).

## Example 3: Three-bar truss (Koski, 1985; Das, 1997)

- ▶ Our (supposedly expensive) numerical model computes
  - ▶ the displacement  $y = (y_1, y_2)$  of point  $P$ ,
  - ▶ the stress  $\sigma_j$  in each bar ( $1 \leq j \leq 3$ ).
- ▶ We will consider the following problem:
  - ▶ minimize  $E_U(\|y\|)$
  - ▶ under the constraints:  $x_{\min} \leq x \leq x_{\max}$ ,  $V \leq V_{\max}$

## Example 3: Three-bar truss (Koski, 1985; Das, 1997)

- ▶ Our (supposedly expensive) numerical model computes
  - ▶ the displacement  $y = (y_1, y_2)$  of point  $P$ ,
  - ▶ the stress  $\sigma_j$  in each bar ( $1 \leq j \leq 3$ ).
- ▶ We will consider the following problem:
  - ▶ minimize  $E_U(\|y\|)$
  - ▶ under the constraints:  $x_{\min} \leq x \leq x_{\max}$ ,  $V \leq V_{\max}$
- ▶ Remark about constraints
  - ▶ The constraint  $V \leq V_{\max}$  is **cheap to evaluate**

$$V = a_1 \sqrt{L^2 + w^2} + a_2 L + a_3 \sqrt{L^2 + (D - w)^2}$$

- ▶ Additional constraints:  $|\sigma_j| \leq \sigma_{\max}$  can be checked a posteriori

# Breaking the “double loop”

- ▶ Natural “double loop” approach
  - ▶ outer loop: ordinary **optimization** algorithm applied to  $\bar{f}$
  - ▶ inner loop: **integration** (MC, quadrature. . . ) to compute  $\bar{f}$

# Breaking the “double loop”

- ▶ Natural “double loop” approach
  - ▶ outer loop: ordinary optimization algorithm applied to  $\bar{f}$
  - ▶ inner loop: integration (MC, quadrature. . . ) to compute  $\bar{f}$
- ▶ Drawback: typically require **large number of evaluations**

# Breaking the “double loop”

- ▶ Natural “double loop” approach
  - ▶ outer loop: ordinary optimization algorithm applied to  $\bar{f}$
  - ▶ inner loop: integration (MC, quadrature...) to compute  $\bar{f}$
- ▶ Drawback: typically require large number of evaluations
- ▶ Bayesian optimization **breaks the double loop** 😊
  - ▶ Construct a Bayesian model for  $f$ , not  $\bar{f}$
  - ▶ Remark: can be achieved using other surrogate-model based approaches (see Janusevskis & LeRiche, 2013, and refs therein)

# Prior model

- ▶ There are **two functions of interest** in this setting
  - ▶ the one that can be observed, i.e.,  $f : (x, u) \mapsto f(x, u)$ ,
  - ▶ and the one that want to optimize:  $\bar{f} = \int f(\cdot, u) P^U(du)$
- ▶  $\bar{f}$  is a function of  $f$ 
  - ▶ priors cannot be specified independently

## Prior model

- ▶ There are two functions of interest in this setting
  - ▶ the one that can be observed, i.e.,  $f : (x, u) \mapsto f(x, u)$ ,
  - ▶ and the one that want to optimize:  $\bar{f} = \int f(\cdot, u) P^U(du)$
- ▶  $\bar{f}$  is a function of  $f$ 
  - ▶ priors cannot be specified independently

Gaussian process priors are, again, very convenient

If  $\underbrace{\xi \sim \mathcal{GP}(m, k)}_{\text{defined on } \mathbb{X} \times \mathbb{U}}$ , then  $\underbrace{\bar{\xi} \sim \mathcal{GP}(m_{\bar{\xi}}, k_{\bar{\xi}})}_{\text{defined on } \mathbb{X}}$ ,

with

$$m_{\bar{\xi}}(x) = \int m(x, u) P^U(du)$$
$$k_{\bar{\xi}}(x, y) = \iint k((x, u), (y, v)) P^U(du) P^U(dv)$$



## Proof

- ▶  $\bar{\xi}$  is Gaussian by linearity of the integral
- ▶ Computation of the mean function: exchange  $\int$  and  $E$

$$\begin{aligned}m_{\bar{\xi}}(x) &= E\left(\bar{\xi}(x)\right) = E\left(\int \xi(x, u) P^U(du)\right) \\ &= \int m(x, u) P^U(du)\end{aligned}$$

- ▶ Computation of the covariance function: idem with bilinearity

$$\begin{aligned}k_{\bar{\xi}}(x, y) &= \text{cov}\left(\bar{\xi}(x), \bar{\xi}(y)\right) \\ &= \text{cov}\left(\int \xi(x, u) P^U(du), \int \xi(x, v) P^U(dv)\right) \\ &= \iint k((x, u), (y, v)) P^U(du) P^U(dv)\end{aligned}\quad \blacksquare$$

## Prior model (cont'd)

- ▶ Actually, we can say much better:

### Jointly Gaussian processes

If  $\xi \sim \mathcal{GP}(m, k)$ , then  $\xi$  and  $\bar{\xi}$  are jointly Gaussian, and

$$k_{\xi, \bar{\xi}}((x, u), y) = \text{cov}(\xi(x, u), \bar{\xi}(y)) = \int k((x, u), (y, v)) P^U(dv)$$

## Prior model (cont'd)

- ▶ Actually, we can say much better:

### Jointly Gaussian processes

If  $\xi \sim \mathcal{GP}(m, k)$ , then  $\xi$  and  $\bar{\xi}$  are jointly Gaussian, and

$$k_{\xi, \bar{\xi}}((x, u), y) = \text{cov}(\xi(x, u), \bar{\xi}(y)) = \int k((x, u), (y, v)) P^U(dv)$$

- ▶ Remark:  $m_{\bar{\xi}}$ ,  $k_{\bar{\xi}}$  and  $k_{\xi, \bar{\xi}}$  can be **computed exactly**
  - ▶ if  $P^U$  is **discrete** ( $P^U = \sum_{j=1}^{n_U} w_j \delta_{u_j}$ )
  - ▶ if  $P^U$  is (a mixture of) **Gaussian**(s), for some particular  $k$
  - ▶ see Girard (2004) for exact formulas & approximations

## Prior model (cont'd)

- ▶ Actually, we can say much better:

### Jointly Gaussian processes

If  $\xi \sim \mathcal{GP}(m, k)$ , then  $\xi$  and  $\bar{\xi}$  are jointly Gaussian, and  $k_{\xi, \bar{\xi}}((x, u), y) = \text{cov}(\xi(x, u), \bar{\xi}(y)) = \int k((x, u), (y, v)) P^U(dv)$

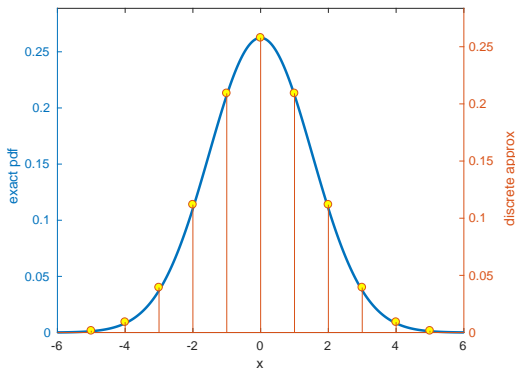
- ▶ Remark:  $m_{\bar{\xi}}$ ,  $k_{\bar{\xi}}$  and  $k_{\xi, \bar{\xi}}$  can be computed exactly
  - ▶ if  $P^U$  is discrete ( $P^U = \sum_{j=1}^{n_U} w_j \delta_{u_j}$ )
  - ▶ if  $P^U$  is (a mixture of) Gaussian(s), for some particular  $k$
  - ▶ see Girard (2004) for exact formulas & approximations
- ▶ Important **special case**:  $\xi(x, u) = \tilde{\xi}(x + u)$ 
  - ▶ If  $\xi$  is a GP iff  $\tilde{\xi}$  is a GP
  - ▶  $m_{\xi}(x, u) = m_{\tilde{\xi}}(x + u)$  and  $k_{\xi}((x, u), (u, v)) = k_{\tilde{\xi}}(x + u, y + v)$

## Examples 1 and 2: discretization of $P^U$

- ▶ In the two examples,  $Prob^U = \mathcal{N}(0, s^2)$ , with  $s = 1.52$ 
  - ▶  $P(|U| \leq 5) \approx 99.9\%$

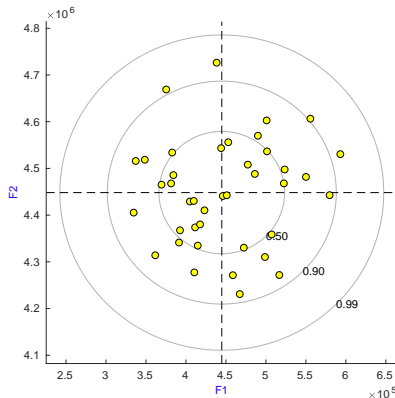
## Examples 1 and 2: discretization of $P^U$

- ▶ In the two examples,  $Prob^U = \mathcal{N}(0, s^2)$ , with  $s = 1.52$ 
  - ▶  $P(|U| \leq 5) \approx 99.9\%$
- ▶ We choose to use a **regular discretization with  $n_U = 11$  points**
  - ▶ points regularly spaced on  $[-5; 5]$
  - ▶ weights computed using the normal cdf (using mid-points)



## Example 3: discretization of $P^U$

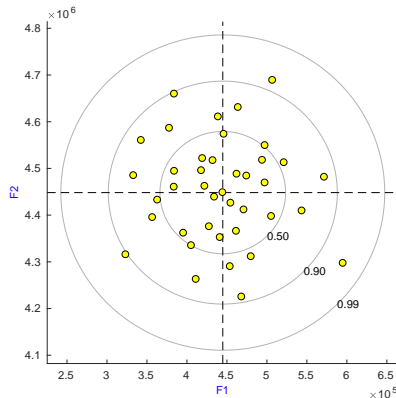
► Here,  $U = \begin{pmatrix} F_1 \\ F_2 \end{pmatrix}$  and  $P^U = \mathcal{N} \left( \begin{pmatrix} \mu_{F_1} \\ \mu_{F_2} \end{pmatrix}, \begin{pmatrix} \sigma_{F_1}^2 & 0 \\ 0 & \sigma_{F_2}^2 \end{pmatrix} \right)$



Monte Carlo sample of size  $n_U = 50$

## Example 3: discretization of $P^U$

► Here,  $U = \begin{pmatrix} F_1 \\ F_2 \end{pmatrix}$  and  $P^U = \mathcal{N} \left( \begin{pmatrix} \mu_{F_1} \\ \mu_{F_2} \end{pmatrix}, \begin{pmatrix} \sigma_{F_1}^2 & 0 \\ 0 & \sigma_{F_2}^2 \end{pmatrix} \right)$



Quasi Monte Carlo (QMC) sample of size  $n_U = 50$



## Sampling strategy: what ?

- ▶ What decision(s) do we have to make at each step ?
  - ▶ i.e, what do we need to provide to run the numerical model ?

# Sampling strategy: what ?

- ▶ What decision(s) do we have to make at each step ?
  - ▶ i.e, what do we need to provide to run the numerical model ?
- ▶ General case
  - ▶ numerical model  $f : (x, u) \mapsto f(x, u)$ , defined on  $\mathbb{X} \times \mathbb{U}$
  - ▶ at each step, we must select a pair  $(X_{n+1}, U_{n+1}) \in \mathbb{X} \times \mathbb{U}$

# Sampling strategy: what ?

- ▶ What decision(s) do we have to make at each step ?
  - ▶ i.e, what do we need to provide to run the numerical model ?
- ▶ General case
  - ▶ numerical model  $f : (x, u) \mapsto f(x, u)$ , defined on  $\mathbb{X} \times \mathbb{U}$
  - ▶ at each step, we must select a pair  $(X_{n+1}, U_{n+1}) \in \mathbb{X} \times \mathbb{U}$
- ▶ Important special case
  - ▶  $f(x, u) = \tilde{f}(x + u)$
  - ▶ in this case, we must simply **select a point**  $X_{n+1} \in \mathbb{X}$
- ▶ In the following slides we assume the general case  
(adaptation to the special case poses no difficulty)

# Sampling strategy: how ?

- ▶ How do we build a sampling strategy for this problem ?
  - ▶ in this lecture, we will **apply the standard BO machinery**
    - ▶  $L^1$  loss  $\rightarrow$  risk  $\rightarrow$  “EI like” myopic strategy

# Sampling strategy: how ?

- ▶ How do we build a sampling strategy for this problem ?
  - ▶ in this lecture, we will apply the standard BO machinery
    - ▶  $L^1$  loss  $\rightarrow$  risk  $\rightarrow$  “EI like” myopic strategy
  - ▶ other strategies are proposed in the literature
    - ▶ Williams et al 2000; Janusevkis et Le Riche 2013
    - ▶ Entropy-based methods could be used as well (Villemonais et al 2009, Hennig & Schueller 2012...)

## Sampling strategy: how ?

- ▶ How do we build a sampling strategy for this problem ?
  - ▶ in this lecture, we will apply the standard BO machinery
    - ▶  $L^1$  loss  $\rightarrow$  risk  $\rightarrow$  “EI like” myopic strategy
  - ▶ other strategies are proposed in the literature
    - ▶ Williams et al 2000; Janusevkiš et Le Riche 2013
    - ▶ Entropy-based methods could be used as well (Villemonteix et al 2009, Hennig & Schueller 2012...)
- ▶ Assume now the  $L^1$  loss
- ▶ Recall the **posterior risk** at time  $N$  for an estimate  $\hat{x} \in \mathbb{X}$ :

$$E_N(L((\xi, U_{\text{real}}), \hat{x})) = E_N(\bar{\xi}(\hat{x})) - E_N(\min_{\mathbb{X}} \xi(\cdot, U_{\text{real}}))$$

## Sampling strategy: one step look-ahead

- ▶ Let  $\bar{L}_n$  denote the **expected loss** that we would get **if we stopped at time  $n$** :

$$\begin{aligned}\bar{L}_n &= \min_{\mathbb{X}} E_n(\bar{\xi}(x)) - E_n(\min_{\mathbb{X}} \xi(\cdot, U_{\text{real}})) \\ &= \min_{\mathbb{X}} m_{\bar{\xi}, n} - E_n(\min_{\mathbb{X}} \xi(\cdot, U_{\text{real}}))\end{aligned}$$

## Sampling strategy: one step look-ahead

- ▶ Let  $\bar{L}_n$  denote the expected loss that we would get if we stopped at time  $n$ :

$$\begin{aligned}\bar{L}_n &= \min_{\mathbb{X}} E_n(\bar{\xi}(x)) - E_n(\min_{\mathbb{X}} \xi(\cdot, U_{\text{real}})) \\ &= \min_{\mathbb{X}} m_{\bar{\xi}, n} - E_n(\min_{\mathbb{X}} \xi(\cdot, U_{\text{real}}))\end{aligned}$$

- ▶ The **one-step look-ahead** (myopic) strategy is

$$(X_{n+1}, U_{n+1}) = \operatorname{argmin}_{x_{n+1}, u_{n+1}} E_{n, (x_{n+1}, u_{n+1})}(\bar{L}_{n+1})$$



## Sampling strategy: one step look-ahead

- ▶ Let  $\bar{L}_n$  denote the expected loss that we would get if we stopped at time  $n$ :

$$\begin{aligned}\bar{L}_n &= \min_{\mathbb{X}} E_n \left( \bar{\xi}(x) \right) - E_n \left( \min_{\mathbb{X}} \xi(\cdot, U_{\text{real}}) \right) \\ &= \min_{\mathbb{X}} m_{\bar{\xi}, n} - E_n \left( \min_{\mathbb{X}} \xi(\cdot, U_{\text{real}}) \right)\end{aligned}$$

- ▶ The **one-step look-ahead** (myopic) strategy is

$$\begin{aligned}(X_{n+1}, U_{n+1}) &= \operatorname{argmin}_{x_{n+1}, u_{n+1}} E_{n, (x_{n+1}, u_{n+1})} \left( \bar{L}_{n+1} \right) \\ &= \operatorname{argmin}_{x_{n+1}, u_{n+1}} E_{n, (x_{n+1}, u_{n+1})} \left( \min_{\mathbb{X}} m_{\bar{\xi}, n+1} \right)\end{aligned}$$

## Sampling strategy: one step look-ahead

- ▶ Equivalently,

$$(X_{n+1}, U_{n+1}) = \operatorname{argmax}_{x_{n+1}, u_{n+1}} \rho_n(x_{n+1}, u_{n+1})$$

where  $\rho_n$  denotes the corresponding “expected improvement”

$$\begin{aligned} \rho_n(x_{n+1}, u_{n+1}) &= \bar{L}_n - \mathbb{E}_{n, (x_{n+1}, u_{n+1})} (\bar{L}_{n+1}) \\ &= \min_{\mathbb{X}} m_{\bar{\xi}, n} - \mathbb{E}_{n, (x_{n+1}, u_{n+1})} \left( \min_{\mathbb{X}} m_{\bar{\xi}, n+1} \right) \end{aligned}$$

## Sampling strategy: one step look-ahead

- ▶ Equivalently,

$$(X_{n+1}, U_{n+1}) = \operatorname{argmax}_{x_{n+1}, u_{n+1}} \rho_n(x_{n+1}, u_{n+1})$$

where  $\rho_n$  denotes the corresponding “expected improvement”

$$\begin{aligned} \rho_n(x_{n+1}, u_{n+1}) &= \bar{L}_n - \mathbb{E}_{n, (x_{n+1}, u_{n+1})} (\bar{L}_{n+1}) \\ &= \min_{\mathbb{X}} m_{\bar{\xi}, n} - \mathbb{E}_{n, (x_{n+1}, u_{n+1})} (\min_{\mathbb{X}} m_{\bar{\xi}, n+1}) \end{aligned}$$

- ▶ Formally, looks like the KG criterion of Frazier & co, but...

## Sampling strategy: one step look-ahead

- ▶ Comparison with KG as presented in the literature

	evaluations	optimization
KG	$\xi(x) + \mathcal{N}\text{noise}$	$\min \xi$
here	$\xi(x, u)$	$\min \bar{\xi}$

## Sampling strategy: one step look-ahead

- ▶ Comparison with KG as presented in the literature

	evaluations	optimization
KG	$\xi(x) + \mathcal{N}\text{noise}$	$\min \xi$
here	$\xi(x, u)$	$\min \bar{\xi}$

- ▶ There is no real difference mathematically: in both cases
  1. the function to be optimized is **not observable directly**,
  2. the evaluation results and the function to be optimized are **jointly Gaussian**.

## Sampling strategy: one step look-ahead

- ▶ Comparison with KG as presented in the literature

	evaluations	optimization
KG	$\xi(x) + \mathcal{N}\text{noise}$	$\min \xi$
here	$\xi(x, u)$	$\min \bar{\xi}$

- ▶ There is no real difference mathematically: in both cases
  1. the function to be optimized is not observable directly,
  2. the evaluation results and the function to be optimized are jointly Gaussian.
- ▶ Good news: we can then derive an implementable **Approximate KG** criterion as in Scott et al (2011)

## Approximate KG criterion (AKG)

- ▶ Let  $\mathbb{X}_n^{\text{ref}} \subset \mathbb{X}$  denote some finite “reference set”
- ▶ Let  $\tilde{x}_{n+1} = (x_{n+1}, u_{n+1})$ . The **AKG criterion** is:

$$\rho_n^{\text{AKG}}(\tilde{x}_{n+1}) = \min m_{\bar{\xi}, n} - E_{n, \tilde{x}_{n+1}} \left( \min m_{\bar{\xi}, n+1} \right) \geq 0$$

where the min runs over  $\mathbb{X}_n^{\text{ref}} \cup \{\tilde{x}_{n+1}\}$ .

## Approximate KG criterion (AKG)

- ▶ Let  $\mathbb{X}_n^{\text{ref}} \subset \mathbb{X}$  denote some finite “reference set”
- ▶ Let  $\tilde{x}_{n+1} = (x_{n+1}, u_{n+1})$ . The AKG criterion is:

$$\rho_n^{\text{AKG}}(\tilde{x}_{n+1}) = \min m_{\bar{\xi}, n} - E_{n, \tilde{x}_{n+1}} \left( \min m_{\bar{\xi}, n+1} \right) \geq 0$$

where the min runs over  $\mathbb{X}_n^{\text{ref}} \cup \{\tilde{x}_{n+1}\}$ .



## Approximate KG criterion (AKG)

- ▶ Let  $\mathbb{X}_n^{\text{ref}} \subset \mathbb{X}$  denote some finite “reference set”
- ▶ Let  $\tilde{x}_{n+1} = (x_{n+1}, u_{n+1})$ . The AKG criterion is:

$$\rho_n^{\text{AKG}}(\tilde{x}_{n+1}) = \min m_{\bar{\xi}, n} - E_{n, \tilde{x}_{n+1}} \left( \min m_{\bar{\xi}, n+1} \right) \geq 0$$

where the min runs over  $\mathbb{X}_n^{\text{ref}} \cup \{\tilde{x}_{n+1}\}$ .

- ▶ Initially proposed by Scott et al (2011)
  - ▶ under the name KGCP (“KG for continuous parameters”)
  - ▶ with  $\mathbb{X}_n^{\text{ref}} = \{X_1, \dots, X_n\}$

# Approximate KG criterion (AKG)

- ▶ Let  $\mathbb{X}_n^{\text{ref}} \subset \mathbb{X}$  denote some finite “reference set”
- ▶ Let  $\tilde{x}_{n+1} = (x_{n+1}, u_{n+1})$ . The AKG criterion is:

$$\rho_n^{\text{AKG}}(\tilde{x}_{n+1}) = \min m_{\bar{\xi}, n} - E_{n, \tilde{x}_{n+1}} \left( \min m_{\bar{\xi}, n+1} \right) \geq 0$$

where the min runs over  $\mathbb{X}_n^{\text{ref}} \cup \{\tilde{x}_{n+1}\}$ .

- ▶ Initially proposed by Scott et al (2011)
  - ▶ under the name KGCP (“KG for continuous parameters”)
  - ▶ with  $\mathbb{X}_n^{\text{ref}} = \{X_1, \dots, X_n\}$
- ▶ Implementation ?
  - ▶ It is **exactly computable** (but not easy to compute...)
  - ▶ Available in STK (Matlab/Octave), DiceOptim (R)...

# Optimization of a mean response: demos

## STK demo

... One dimensional illustration: examples 1 and 2 ...

## STK demo

... Minimization of the mean displacement  
in the 3-bar truss example ...

## (a few words about) The case of stochastic simulators

- ▶ Good news
  - ▶ the **same sampling criteria can be used** in both cases (environmental variables / stochastic simulators)...
  - ▶ ...provided that the observations and the objective function are **jointly Gaussian**.

## (a few words about) The case of stochastic simulators

- ▶ Good news
  - ▶ the same sampling criteria can be used in both cases (environmental variables / stochastic simulators)...
  - ▶ ... provided that the observations and the objective function are jointly Gaussian.
- ▶ **Review/benchmark** of existing criteria: Picheny et al (2013)
  - ▶ AKG emerges has one of the most efficient criteria
  - ▶ Huang et al (2006)'s “augmented EI” also performs well
  - ▶ (Entropy-based criteria not benchmarked)

## (a few words about) The case of stochastic simulators

- ▶ Good news
  - ▶ the same sampling criteria can be used in both cases (environmental variables / stochastic simulators)...
  - ▶ ...provided that the observations and the objective function are jointly Gaussian.
- ▶ Review/benchmark of existing criteria: Picheny et al (2013)
  - ▶ AKG emerges has one of the most efficient criteria
  - ▶ Huang et al (2006)'s “augmented EI” also performs well
  - ▶ (Entropy-based criteria not benchmarked)
- ▶ What about simulators with truly **non-Gaussian output** ?
  - ▶ “batch trick” (CLT)
  - ▶ see also Browne et al (2016)...

## Lecture 2 : Bayesian optimization (BO)

### 2.3. Design under uncertainty

Overview of possible approaches

Optimization of a mean response

RBDO (and other formulations)

# Reliability-based (design) optimization (RBO, RBDO)

- ▶ Assume

- ▶ a **single objective**  $f = f_o$ , often cheap to evaluate
- ▶ one or several **expensive-to-evaluate constraints**  $f_{c,1}, \dots, f_{c,q}$
- ▶ “**environmental variables**” setting



# Reliability-based (design) optimization (RBO, RBDO)

- ▶ Assume
  - ▶ a single objective  $f = f_o$ , often cheap to evaluate
  - ▶ one or several expensive-to-evaluate constraints  $f_{c,1}, \dots, f_{c,q}$
  - ▶ “environmental variables” setting
- ▶ The so-called RB(D)O formulation reads:

## Reliability-based (a.k.a. chance-constrained) optimization

Minimize

$$\bar{f}_o(x), \quad \text{where } f_o(x) = E_U(f_o(x, U))$$

under the constraints:  $x \in \mathbb{X}$  and

$$\forall j \leq q, \quad P_U(f_{c,j}(x, U) > 0) \leq p_j^{\text{tol}}$$

# Reliability-based (design) optimization (RBO, RBDO)

- ▶ See Valdebenito & Schuëller (2010) for a survey

# Reliability-based (design) optimization (RBO, RBDO)

- ▶ See Valdebenito & Schuëller (2010) for a survey
- ▶  $\bar{f}_0$  is often **cheap to evaluate**
  - ▶ e.g. volume / mass / manufacturing cost / ...
  - ▶ Expectation often computed (or approximated) analytically

# Reliability-based (design) optimization (RBO, RBDO)

- ▶ See Valdebenito & Schuëller (2010) for a survey
- ▶  $\bar{f}_0$  is often cheap to evaluate
  - ▶ e.g. volume / mass / manufacturing cost / ...
  - ▶ Expectation often computed (or approximated) analytically
- ▶ Again, algorithms with a “double loop” structure can be used
  - ▶ outer loop: ordinary optimization algorithm with constraints
  - ▶ inner loop: reliability analysis method to compute the constraints

# Reliability-based (design) optimization (RBO, RBDO)

- ▶ See Valdebenito & Schuëller (2010) for a survey
- ▶  $\bar{f}_0$  is often cheap to evaluate
  - ▶ e.g. volume / mass / manufacturing cost / ...
  - ▶ Expectation often computed (or approximated) analytically
- ▶ Again, algorithms with a “double loop” structure can be used
  - ▶ outer loop: ordinary optimization algorithm with constraints
  - ▶ inner loop: reliability analysis method to compute the constraints
- ▶ Again, surrogate-based methods should be able to “break the double loop” by building a model on  $\mathbb{X} \times \mathbb{U}$

# Why RBDO is harder than mean-response optimization

- ▶ because the **thresholds**  $p_j^{\text{tol}}$  are usually **small**
  - ▶ MC-type approx.  $P^U \approx \frac{1}{m} \sum_j \delta_{u_j}$  becomes very expensive or infeasible
  - ▶ Dedicated techniques (e.g., FORM/SORM, IS, subset simulation) needed for an **efficient evaluation of the constraints**

# Why RBDO is harder than mean-response optimization

- ▶ because the thresholds  $p_j^{\text{tol}}$  are usually small
  - ▶ MC-type approx.  $P^U \approx \frac{1}{m} \sum_j \delta_{u_j}$  becomes very expensive or infeasible
  - ▶ Dedicated techniques (e.g., FORM/SORM, IS, subset simulation) needed for an efficient evaluation of the constraints
- ▶ BO: the distribution of  $P_U(\xi_{c,j}(x, U) > 0)$  is intractable
  - ▶ **Posterior mean/variance** can be written as integrals (see, e.g., Villemonteix 2008 chap III), but...
  - ▶ ... the posterior distribution is **not Gaussian** even if  $\xi_c$  is !
  - ▶ Very difficult to derive Bayesian **sampling criteria** that can be implemented efficiently...

# Bayesian RBDO algorithms ?

- ▶ A few GP-based algorithms have been proposed, notably:
  - ▶ Bichon et al (2009): **EGO+EGRA** algorithm
  - ▶ Dubourg and co-authors (2011a, 2011b): **RBDO-N2LA**



# Bayesian RBDO algorithms ?

- ▶ A few GP-based algorithms have been proposed, notably:
  - ▶ Bichon et al (2009): EGO+EGRA algorithm
  - ▶ Dubourg and co-authors (2011a, 2011b): RBDO-N2LA
- ▶ Complex, “weakly Bayesian” algorithms... e.g., RBDO-N2LA:

**Algorithm 2** Adaptive surrogate-based nested RBDO strategy

```

1:  $\theta^{(0)}, \theta^L, \theta^U, j := 0$ 
2: for  $i = 1$  to  $n$  do
3:    $\theta_0^i := \min_{\theta \in \mathcal{D}_\theta} F_N^{-1}(v_i, \theta)$ 
4:    $\theta_0^i := \max_{\theta \in \mathcal{D}_\theta} F_N^{-1}(v_i, \theta)$ 
5: end for
6:  $x := x + 1$ ;  $\mathcal{A}_k(v)$ 
7:  $k := \Phi^{-1}(97.5\%)$ ,  $\epsilon_1 := 10^{-1}$ 
8: Refine := true, Optimize := true
9: while Refine and Optimize do
10:  while Refine do
11:     $G := \text{RefineKrigingModel}(v_i, k, \epsilon_1) \rightarrow \text{Use Algorithm 1}$ 
12:  end while
13:   $\hat{F}^i := \{x : \mu_G(x) \leq 0\}$ 
14:   $\hat{\beta}^i, \nabla_{\mathbf{x}} \hat{\beta}^i := \text{ReliabilityAnalysis}(\hat{F}^i, \theta^{(i)})$ 
15:   $\epsilon^{(i)} := \epsilon(\theta^{(i)})$ ,  $\nabla_{\mathbf{x}} \epsilon^{(i)} := \nabla_{\mathbf{x}} \epsilon(\theta^{(i)})$ 
16:   $f^{(i)} := f(\theta^{(i)})$ ,  $\nabla_{\mathbf{x}} f^{(i)} := \nabla_{\mathbf{x}} f(\theta^{(i)})$ 
17:   $d^{(i)} := \text{SolveQuasiSQP}(\epsilon^{(i)}, f^{(i)}, \hat{\beta}^i, \nabla_{\mathbf{x}} \epsilon^{(i)}, \nabla_{\mathbf{x}} f^{(i)}, \nabla_{\mathbf{x}} \hat{\beta}^i)$ 
18:   $s^{(i)} := \text{GoldsteinAmpeStepSizeRule}(\epsilon, f, \hat{\beta}^i, d^{(i)})$ 
19:   $\theta^{(i+1)} := \theta^{(i)} + s^{(i)} d^{(i)}$ 
20:  for  $l = -1, 0, +1$  do
21:     $\hat{F}^{i+l} := \{x : \mu_G(x) + l k \sigma_G(x) \leq 0\}$ 
22:     $\hat{\beta}^{i+l} := \text{ReliabilityAnalysis}(\hat{F}^{i+l}, \theta^{(i+1)})$ 
23:  end for
24:  Refine :=  $\max(\hat{\beta}^{i+1} - \hat{\beta}^i, \hat{\beta}^0 - \hat{\beta}^{i-1}) > \epsilon_1$ 
25:  Optimize :=  $|\theta^{(i+1)} - \theta^{(i)}| > \epsilon_0$  or  $|\epsilon^{(i+1)} - \epsilon^{(i)}| > \epsilon_2$ 
      or  $3 \mid f_i > 0$  or  $\hat{\beta} < \beta_k$ 
26: end while
    
```

**Algorithm 1** Population-based adaptive refinement strategy

```

1:  $\hat{X} = \emptyset, \mathcal{Y} = \emptyset$ 
2:  $n := x + 1$ ;  $w(x)$ 
3:  $k := \Phi^{-1}(97.5\%)$ ,  $\epsilon_1 := 10^{-1}$ 
4: Refine := true
5:  $\mathcal{M} := \emptyset$ 
6: while Refine do
7:   $\hat{\mathcal{P}} := \text{MCMCAlgorithm}(\mathcal{Y})$ 
8:   $\hat{X}_{\text{new}} := \text{KMMeansAlgorithm}(\hat{\mathcal{P}}, K)$ 
9:   $\mathcal{Y}_{\text{new}} := \mathcal{Y}(\hat{X}_{\text{new}})$ 
10:   $\hat{X} := \{\hat{X}, \hat{X}_{\text{new}}\}$ ,  $\mathcal{Y} := \{\mathcal{Y}, \mathcal{Y}_{\text{new}}\}$ 
11:   $G := \text{MaximumLikelihoodKrigingModel}(\hat{X}, \mathcal{Y}, f, \bullet, R(\bullet, \ell), \ell_1, \ell_2)$ 
12:   $\mathbb{P}\{x \in \mathcal{M}\} := x + \Phi\left(\frac{k \sigma_G(x) - \mu_G(x)}{\sigma_G(x)}\right) - \Phi\left(\frac{-k \sigma_G(x) - \mu_G(x)}{\sigma_G(x)}\right)$ 
13:   $\mathcal{Y} := x + \mathbb{P}\{x \in \mathcal{M}\} w(x)$ 
14:  for  $l = -1, 0, +1$  do
15:     $\hat{F}^{i+l} := \{x : \mu_G(x) + l k \sigma_G(x) \leq 0\}$ 
16:     $\hat{\beta}^{i+l} := \text{ReliabilityAnalysis}(\hat{F}^{i+l})$ 
17:  end for
18:  Refine :=  $\max(\hat{\beta}^{i+1} - \hat{\beta}^i, \hat{\beta}^0 - \hat{\beta}^{i-1}) > \epsilon_1$ 
19: end while
    
```

# Bayesian RBDO algorithms ?

- ▶ A few GP-based algorithms have been proposed, notably:
  - ▶ Bichon et al (2009): EGO+EGRA algorithm
  - ▶ Dubourg and co-authors (2011a, 2011b): RBDO-N2LA
- ▶ Complex, “weakly Bayesian” algorithms... e.g., RBDO-N2LA:

**Algorithm 2** Adaptive surrogate-based nested RBDO strategy

```

1:  $\theta^{(0)}, \theta^L, \theta^U, j := 0$ 
2: for  $i = 1$  to  $n$  do
3:    $\theta_0^i := \min_{\theta \in \mathcal{D}_i(\theta^L)} F_{\theta}^{-1}(x_0, \theta)$ 
4:    $\theta_0^i := \max_{\theta \in \mathcal{D}_i(\theta^U)} F_{\theta}^{-1}(x_0, \theta)$ 
5: end for
6:  $x := x + 1$ ;  $\mathcal{D}_i(x)$ 
7:  $k := \Phi^{-1}(97.5\%)$ ,  $\epsilon_1 := 10^{-1}$ 
8: Refine := true, Optimize := true
9: while Refine and Optimize do
10:  while Refine do
11:     $G := \text{RefineKrigingModel}(x, k, \epsilon_1) \rightarrow \text{Use Algorithm 1}$ 
12:  end while
13:   $\hat{F}^i := \{x : \mu_G(x) \leq 0\}$ 
14:   $\hat{\beta}^i, \nabla_{\theta} \hat{\beta}^i := \text{ReliabilityAnalysis}(\hat{F}^i, \theta^{(j)})$ 
15:   $\epsilon^{(j)} := \epsilon(\theta^{(j)})$ ,  $\nabla_{\theta} \epsilon^{(j)} := \nabla_{\theta} \epsilon(\theta^{(j)})$ 
16:   $f^{(j)} := f(\theta^{(j)})$ ,  $\nabla_{\theta} f^{(j)} := \nabla_{\theta} f(\theta^{(j)})$ 
17:   $d^{(j)} := \text{SolveQuasiSQP}(\epsilon^{(j)}, f^{(j)}, \hat{\beta}^i, \nabla_{\theta} \epsilon^{(j)}, \nabla_{\theta} f^{(j)}, \nabla_{\theta} \hat{\beta}^i)$ 
18:   $s^{(j)} := \text{GoldsteinAmplifyStepSizeRule}(k, f, \hat{\beta}^i, d^{(j)})$ 
19:   $\theta^{(j+1)} := \theta^{(j)} + s^{(j)}$  do
20:    for  $i = -1, 0, +1$  do
21:       $\hat{F}^{i+1} := \{x : \mu_G(x) + ik \sigma_G(x) \leq 0\}$ 
22:       $\hat{\beta}^{i+1} := \text{ReliabilityAnalysis}(\hat{F}^{i+1}, \theta^{(j+1)})$ 
23:    end for
24:    Refine :=  $\max(\hat{\beta}^{+1} - \hat{\beta}^i, \hat{\beta}^i - \hat{\beta}^{-1}) > \epsilon_1$ 
25:    Optimize :=  $|\theta^{(j+1)} - \theta^{(j)}| > \epsilon_0$  or  $|\epsilon^{(j+1)} - \epsilon^{(j)}| > \epsilon_2$ 
        or  $\exists i \mid f_i > 0$  or  $\hat{\beta} < \beta_0$ 
26:  end while
    
```

**Algorithm 1** Population-based adaptive refinement strategy

```

1:  $\hat{X} = \emptyset, \mathcal{Y} = \emptyset$ 
2:  $n := x + 1$ ;  $w(x)$ 
3:  $k := \Phi^{-1}(97.5\%)$ ,  $\epsilon_1 := 10^{-1}$ 
4: Refine := true
5:  $\mathcal{M} := \mathcal{M} \cup \{x\}$ 
6: while Refine do
7:    $\hat{\mathcal{P}} := \text{MCMCAlgorithm}(\mathcal{Y})$ 
8:    $\hat{X}_{\text{mean}} := \text{KMMeansAlgorithm}(\hat{\mathcal{P}}, K)$ 
9:    $\mathcal{Y}_{\text{new}} := \mathcal{Y} \cup \{\hat{X}_{\text{new}}\}$ 
10:   $\hat{X} := \{\hat{X}, \hat{X}_{\text{new}}\}$ ,  $\mathcal{Y} := \{\mathcal{Y}, \mathcal{Y}_{\text{new}}\}$ 
11:   $G := \text{MaximumLikelihoodKrigingModel}(\hat{X}, \mathcal{Y}, f, \bullet, R(\bullet, \ell), \ell_1, \ell_2)$ 
12:   $\mathbb{P}\{x \in \mathcal{M}\} := x + \Phi\left(\frac{k \sigma_G(x)}{\sigma_G(x)}\right) - \Phi\left(\frac{-k \sigma_G(x) - \mu_G(x)}{\sigma_G(x)}\right)$ 
13:   $\mathcal{Y} := x + \mathbb{P}\{x \in \mathcal{M}\} w(x)$ 
14:  for  $j := -1, 0, +1$  do
15:     $\hat{F}^{j+1} := \{x : \mu_G(x) + ik \sigma_G(x) \leq 0\}$ 
16:     $\hat{\beta}^{j+1} := \text{ReliabilityAnalysis}(\hat{F}^{j+1})$ 
17:  end for
18:  Refine :=  $\max(\hat{\beta}^{+1} - \hat{\beta}^0, \hat{\beta}^0 - \hat{\beta}^{-1}) > \epsilon_1$ 
19: end while
    
```

- ▶ RBDO-N2LA is available in FERUM (Bourrinet et al, 2009)

## RBDO and other formulations: alternative approach

- ▶ A general alternative approach in two steps
  1. Explore the design space **efficiently** using **multi-objective** BO,
  2. Evaluate probabilities of failure, quantiles, etc. **a posteriori** for non-dominated (and possibly other) solutions,

## RBDO and other formulations: alternative approach

- ▶ A general alternative approach in two (or three) steps
  1. Explore the design space **efficiently** using **multi-objective** BO,
  2. Evaluate probabilities of failure, quantiles, etc. **a posteriori** for non-dominated (and possibly other) solutions,
  3. optionally **Reduce the uncertainty** on the most promising designs.

## RBDO and other formulations: alternative approach

- ▶ A general alternative approach in two (or three) steps
  1. Explore the design space efficiently using multi-objective BO,
  2. Evaluate probabilities of failure, quantiles, etc. a posteriori for non-dominated (and possibly other) solutions,
  3. optionally Reduce the uncertainty on the most promising designs.
- ▶ In step 1. **constraints** can be taken into account **as objectives**
  - ▶ i.e., the constraint  $P_U(f_{c,j}(x, U) > 0) \leq p_j^{\text{tol}}$
  - ▶ becomes:  **$\min_x f_{c,j}(x, u_0)$  u.c.  $f_{c,j}(x, u_0) \leq 0$**

## RBDO and other formulations: alternative approach

- ▶ A general alternative approach in two (or three) steps
  1. Explore the design space efficiently using multi-objective BO,
  2. Evaluate probabilities of failure, quantiles, etc. a posteriori for non-dominated (and possibly other) solutions,
  3. optionally Reduce the uncertainty on the most promising designs.
- ▶ In step 1. constraints can be taken into account as objectives
  - ▶ i.e., the constraint  $P_U(f_{c,j}(x, U) > 0) \leq p_j^{\text{tol}}$
  - ▶ becomes:  $\min_x f_{c,j}(x, u_0)$  u.c.  $f_{c,j}(x, u_0) \leq 0$
- ▶ In step 2. no new evaluations need to be carried out
  - ▶ the posterior distribution of  $\xi$  is used to **assess uncertainties**

# RBDO and other formulations: alternative approach

STK demo

... Robust design through multi-objective optimization ...

## Lecture 1 : From meta-models to UQ

- 1.1 Introduction
- 1.2 Black-box modeling
- 1.3 Bayesian approach
- 1.4 Posterior distribution of a quantity of interest
- 1.5 Complements on Gaussian processes

## Lecture 2 : Bayesian optimization (BO)

- 2.1. Decision-theoretic framework
- 2.2. From Bayes-optimal to myopic strategies
- 2.3. Design under uncertainty

## References



# References I



Julien Bect, Emmanuel Vazquez, et al.

STK: a Small (Matlab/Octave) Toolbox for Kriging. Release 2.4.2, 2017.

URL <http://kriging.sourceforge.net>.



Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski.

*Robust optimization*.

Princeton University Press, 2009.



Romain Benassi.

*Nouvel algorithme d'optimisation bayésien utilisant une approche Monte-Carlo séquentielle*.

PhD thesis, Supélec, 2013.



Romain Benassi, Julien Bect, and Emmanuel Vazquez.

Bayesian optimization using sequential Monte Carlo.

In *Learning and Intelligent Optimization. 6th International Conference, LION 6, Paris, France, January 16-20, 2012, Revised Selected Papers*, volume 7219 of *Lecture Notes in Computer Science*, pages 339–342. Springer, 2012.



Dimitris Bertsimas, David B Brown, and Constantine Caramanis.

Theory and applications of robust optimization.

*SIAM review*, 53(3):464–501, 2011.

# References II



Barron Bichon, Sankaran Mahadevan, and Michael Eldred.

Reliability-based design optimization using efficient global reliability analysis.

In *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference 17th AIAA/ASME/AHS Adaptive Structures Conference 11th AIAA No*, page 2261, 2009.



Mickael Binois and Victor Picheny.

*GPareto: Gaussian Processes for Pareto Front Estimation and Optimization*.

R package version 1.0.3, 2016.



Jean-Marc Bourinet, Cécile Mattrand, and Vincent Dubourg.

A review of recent features and improvements added to FERUM software.

In H. Furuta, D. M. Frangopol, and M. Shinozuka, editors, *Proc. 10<sup>th</sup> International Conference on Structural Safety and Reliability (ICOSSAR 2009), Osaka, Japan, September 13–17, 2009*. CRC Press, 2009.



Thomas Browne, Bertrand Iooss, Loïc Le Gratiet, Jérôme Lonchamp, and Emmanuel Remy.

Stochastic simulators based optimization by gaussian process metamodels—application to maintenance investments planning issues.

*Quality and Reliability Engineering International*, 32(6):2067–2080, 2016.

# References III



Clément Chevalier and David Ginsbourger.

Fast computation of the multi-points expected improvement with applications in batch selection.

In *International Conference on Learning and Intelligent Optimization*, pages 59–69. Springer, 2013.



Clément Chevalier, Julien Bect, David Ginsbourger, Emmanuel Vazquez, Victor Picheny, and Yann Richet.

Fast parallel kriging-based stepwise uncertainty reduction with application to the identification of an excursion set.

*Technometrics*, 56(4):455–465, 2014.



Indraneel Das.

*Nonlinear multicriteria optimization and robust optimality*.

PhD thesis, Rice University, 1997.



Vincent Dubourg.

*Adaptive surrogate models for reliability analysis and reliability-based design optimization*.

PhD thesis, Université Blaise Pascal-Clermont-Ferrand II, 2011.

# References IV



Vincent Dubourg, Bruno Sudret, and Jean Marc Bourinet.

Reliability-based design optimization using kriging surrogates and subset simulation.

*Structural and Multidisciplinary Optimization*, 44(5):673–690, 2011.



Michael Emmerich, Kyriakos C. Giannakoglou, and Boris Naujoks.

Single- and multi-objective evolutionary optimization assisted by Gaussian random field metamodels.

*IEEE Transactions on Evolutionary Computation*, 10(4):421–439, 2006.



Paul Feliot, Julien Bect, and Emmanuel Vazquez.

A Bayesian approach to constrained single-and multi-objective optimization.

*Journal of Global Optimization*, 67(1-2):97–133, 2017.



Alexander Forrester, András Sóbester, and Andy Keane.

*Engineering design via surrogate modelling: a practical guide*.

John Wiley & Sons, 2008.



Peter Frazier, Warren Powell, and Savas Dayanik.

The knowledge-gradient policy for correlated normal beliefs.

*INFORMS journal on Computing*, 21(4):599–613, 2009.

# References V



Peter I Frazier, Warren B Powell, and Savas Dayanik.

A knowledge-gradient policy for sequential information collection.

*SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.



David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro.

Kriging is well-suited to parallelize optimization.

In *Computational Intelligence in Expensive Optimization Problems*, pages 131–162. Springer Science & Business Media, 2010.



Agathe Girard.

*Approximate methods for propagation of uncertainty with gaussian process models*.

PhD thesis, University of Glasgow, 2004.



Robert B. Gramacy, Genetha A. Gray, Sebastien Le Digabel, Herbert K. H. Lee, Pritam Ranjan, Garth Wells, and Stefan M. Wild.

Modeling an augmented lagrangian for blackbox constrained optimization.

*Technometrics*, 58(1):1–11, 2016.



Philipp Hennig and Christian J. Schuler.

Entropy search for information-efficient global optimization.

*Journal of Machine Learning Research*, 13(Jun):1809–1837, 2012.

# References VI



José Miguel Hernández-Lobato, Matthew W. Hoffman, and Zoubin Ghahramani.  
Predictive entropy search for efficient global optimization of black-box functions.  
*In Advances in Neural Information Processing Systems*, pages 918–926, 2014.



José Miguel Hernández-Lobato, Michael A. Gelbart, Matthew W. Hoffman,  
Ryan P. Adams, and Zoubin Ghahramani.  
Predictive entropy search for Bayesian optimization with unknown constraints.  
*In ICML*, pages 1699–1707, 2015.



Deng Huang, Theodore T. Allen, William I. Notz, and R. Allen Miller.  
Sequential kriging optimization using multiple-fidelity evaluations.  
*Structural and Multidisciplinary Optimization*, 32(5):369–382, 2006.



Janis Janusevskis and Rodolphe Le Riche.  
Simultaneous kriging-based estimation and optimization of mean response.  
*Journal of Global Optimization*, 55(2):313–336, 2013.



Donald R. Jones, Matthias Schonlau, and William J. Welch.  
Efficient global optimization of expensive black-box functions.  
*Journal of Global Optimization*, 13(4):455–492, 1998.

# References VII



Joshua Knowles.

ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems.

*IEEE Transactions on Evolutionary Computation*, 10(1):50–66, 2006.



Juhani Koski.

Defectiveness of weighting method in multicriterion optimization of structures.

*Communications in Applied Numerical Methods*, 1(6):333–337, 1985.



Sébastien Marmin, Clément Chevalier, and David Ginsbourger.

Differentiating the multipoint expected improvement for optimal batch design.

In *International Workshop on Machine Learning, Optimization and Big Data*, pages 37–48. Springer, 2015.



Julien Marzat, Eric Walter, and H el ene Piet-Lahanier.

Worst-case global optimization of black-box functions through kriging and relaxation.

*Journal of Global Optimization*, 55(4):707, 2013.



Julien Marzat, Eric Walter, and H el ene Piet-Lahanier.

A new expected-improvement algorithm for continuous minimax optimization.

*Journal of Global Optimization*, 64(4):785–802, 2016.

# References VIII



Jonas Mockus, Vytautas Tiesis, and Antanas Žilinskas.

The application of Bayesian methods for seeking the extremum.

In L. C. W. Dixon and G. P. Szegö, editors, *Towards Global Optimization*, volume 2, pages 117–129, North Holland, New York, 1978.



Victor Picheny, Tobias Wagner, and David Ginsbourger.

A benchmark of kriging-based infill criteria for noisy optimization.

*Structural and Multidisciplinary Optimization*, 48(3):607–626, 2013.



Victor Picheny, David Ginsbourger, and Roustant Olivier et al.

*DiceOptim: Kriging-Based Optimization for Computer Experiments*, 2016a.

R package version 2.0, 2016.



Victor Picheny, Robert B. Gramacy, Stefan Wild, and Sebastien Le Digabel.

Bayesian optimization under mixed constraints with a slack-variable augmented lagrangian.

In *Advances in Neural Information Processing Systems*, pages 1435–1443, 2016b.



Carl R. Rasmussen and Christopher K. I. Williams.

*Gaussian Processes for Machine Learning*.

MIT Press, 2006.



# References IX



Samee Ur Rehman and Matthijs Langelaar.  
Efficient infill sampling for unconstrained robust optimization problems.  
*Engineering Optimization*, 48(8):1313–1332, 2016.



Samee Ur Rehman, Matthijs Langelaar, and Fred van Keulen.  
Efficient kriging-based robust optimization of unconstrained problems.  
*Journal of Computational Science*, 5(6):872–881, 2014.



Olivier Roustant, David Ginsbourger, and Yves Deville.  
DiceKriging, DiceOptim: Two R packages for the analysis of computer experiments by kriging-based metamodeling and optimization.  
*Journal of Statistical Software*, 51(1), 2012.



Thomas J. Santner, Brian J. Williams, and William Notz.  
*The Design and Analysis of Computer Experiments*.  
Springer Verlag, 2003.



Matthias Schonlau, William J. Welch, and Donald R. Jones.  
Global versus local search in constrained optimization of computer models.  
In *New Developments and Applications in Experimental Design: Selected Proceedings of a 1997 Joint AMS-IMS-SIAM Summer Conference*, volume 34 of *IMS Lecture Notes-Monographs Series*, pages 11–25, 1998.

# References X



Warren Scott, Peter Frazier, and Warren Powell.

The correlated knowledge gradient for simulation optimization of continuous parameters using gaussian process regression.

*SIAM Journal on Optimization*, 21(3):996–1026, 2011.



Michael L. Stein.

*Interpolation of Spatial Data: Some Theory for Kriging*.

Springer, New York, 1999.



Marcos A. Valdebenito and Gerhart I. Schuëller.

A survey on approaches for reliability-based optimization.

*Structural and Multidisciplinary Optimization*, 42(5):645–663, 2010.



Emmanuel Vazquez.

*Modélisation comportementale de systèmes non-linéaires multivariables par méthodes à noyaux et applications*.

PhD thesis, Univ Paris XI, Orsay, France, 2005.



Julien Villemonteix.

*Optimisation de fonctions coûteuses Modèles gaussiens pour une utilisation efficace du budget d'évaluations: théorie et pratique industrielle*.

PhD thesis, Université Paris Sud-Paris XI, 2008.

# References XI



Julien Villemonteix, Emmanuel Vazquez, and Éric Walter.

An informational approach to the global optimization of expensive-to-evaluate functions.

*Journal of Global Optimization*, 44(4):509–534, 2009.



Brian J. Williams, Thomas J. Santner, and William I. Notz.

Sequential design of computer experiments to minimize integrated response functions.

*Statistica Sinica*, pages 1133–1152, 2000.



Brian J. Williams, Thomas J. Santner, William I. Notz, and J. S. Lehman.

Sequential design of computer experiments for constrained optimization.

In *Statistical Modelling and Regression Structures*, pages 449–472.

Physica-Verlag HD, 2010.