An overview into the fascinating world of Bayesian optimization

Ricardo López Dawn

LiP6 - Sorbonne Université

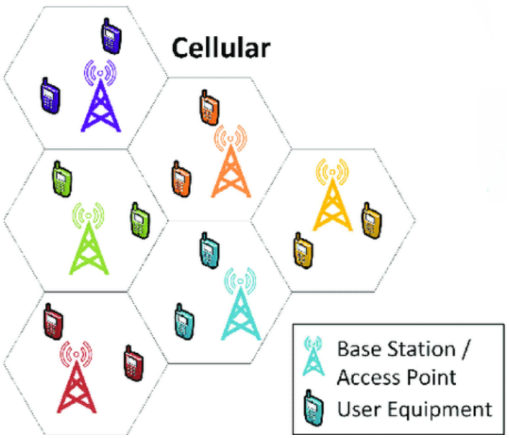
Table of Contents

Motivation

Radio Resource Management

Radio Resource Management (RRM) involves the problem of controlling parameters : transmit power, user allocation, data rates, etc.

The objective is to utilize the radio network infrastructure as efficiently as possible.



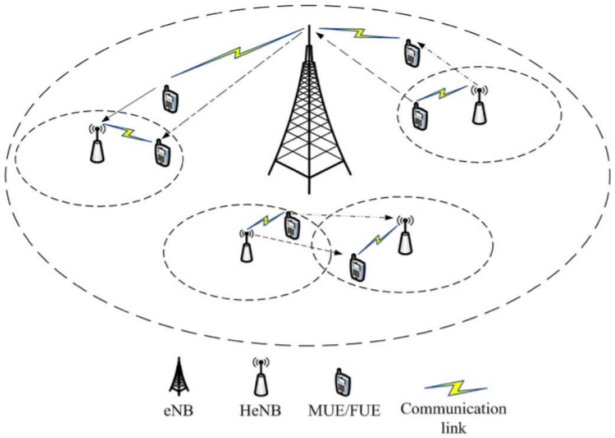


Figure – Illustrative examples

Challenges in RRM

The challenges related to solve RRM problems are :

1. The time or number of trials it takes to find a good set of parameters.
2. The objective function lacks special structure that would make it easy to optimize.
3. We observe the performance, and no first- or second-order derivatives.

Problem formulation

- A controller wants to optimize $x \in \mathcal{X} \subset \mathbb{R}^k$,
- The performance of the system is described by $f(x) \in \mathbb{R}$.
- f is unknown to the controller, possibly noisy, continuous and expensive to evaluate.
- We call $\tilde{f}$ its realizations, and $\mathbb{E}(\tilde{f}) = f$.
- The controller test different values $x(1), x(2), ...$, and observe $\tilde{f}(x(1)), \tilde{f}(x(2)), ...$

The goal is that performance is good at any point in time, while possibly converging to the global optimal configuration $x^* = arg\ max_x f(x)$.

BO in a nutshell

Intuition

- We want to find the optimum of our objective function.
- We fit a Gaussian Process to our observed points and pick our next best point where we believe the maximum will be.
- The next point is determined by a surrogate function called acquisition function - that trades of exploration and exploitation.
- The surrogate function is updated in each step by some hyperparameters.

BO in a nutshell

Intuition

- We want to find the optimum of our objective function.
- We fit a Gaussian Process to our observed points and pick our next best point where we believe the maximum will be.
- The next point is determined by a surrogate function called acquisition function - that trades of exploration and exploitation.
- The surrogate function is updated in each step by some hyperparameters.

BO in a nutshell

Intuition

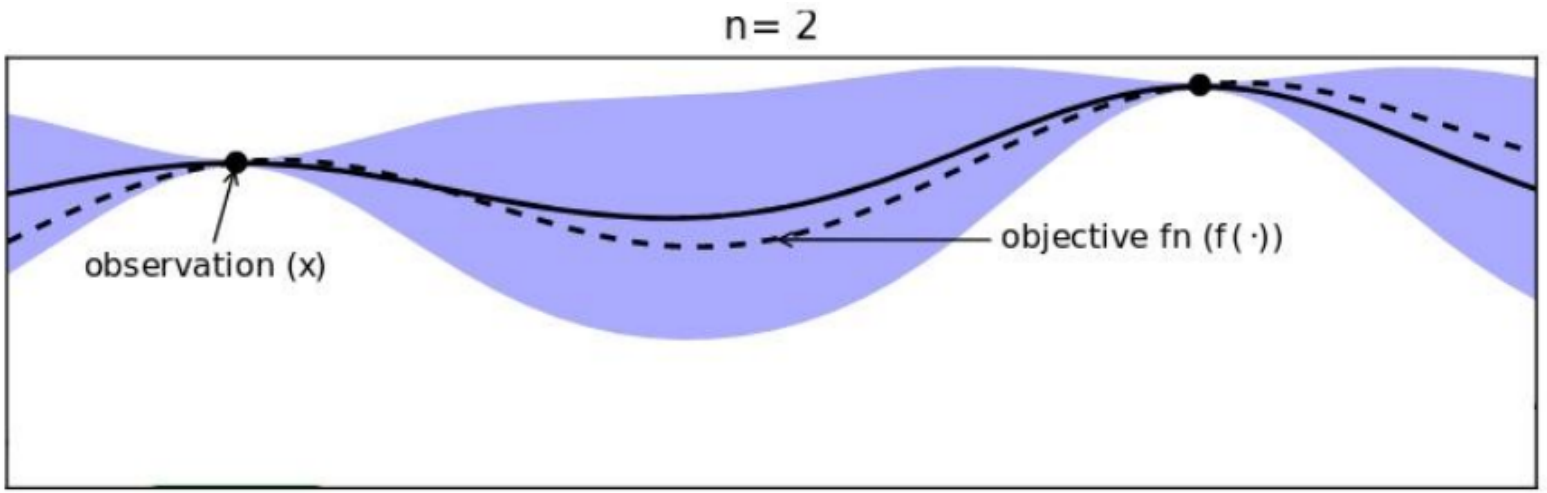
- We want to find the optimum of our objective function.
- We fit a Gaussian Process to our observed points and pick our next best point where we believe the maximum will be.
- The next point is determined by a surrogate function called acquisition function - that trades of exploration and exploitation.
- The surrogate function is updated in each step by some hyperparameters.

BO in a nutshell

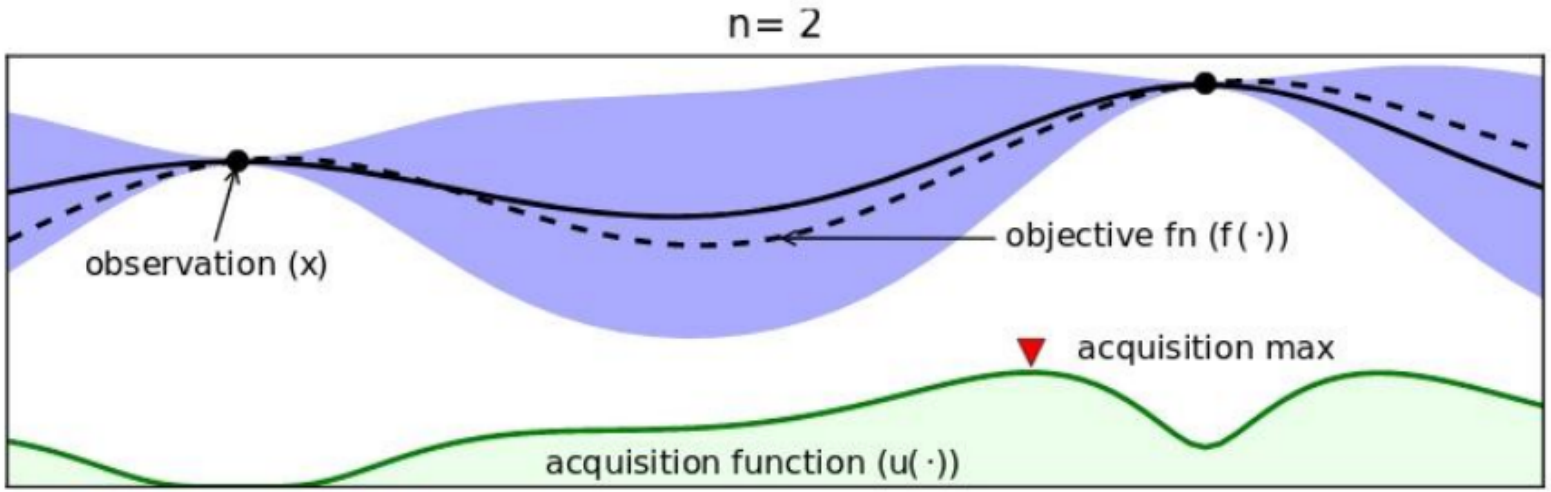
Intuition

- We want to find the optimum of our objective function.
- we fit a Gaussian Process to our observed points and pick our next best point where we believe the maximum will be.
- The next point is determined by a surrogate function called acquisition function - that trades of exploration and exploitation.
- The surrogate function is updated in each step by some hyperparameters.

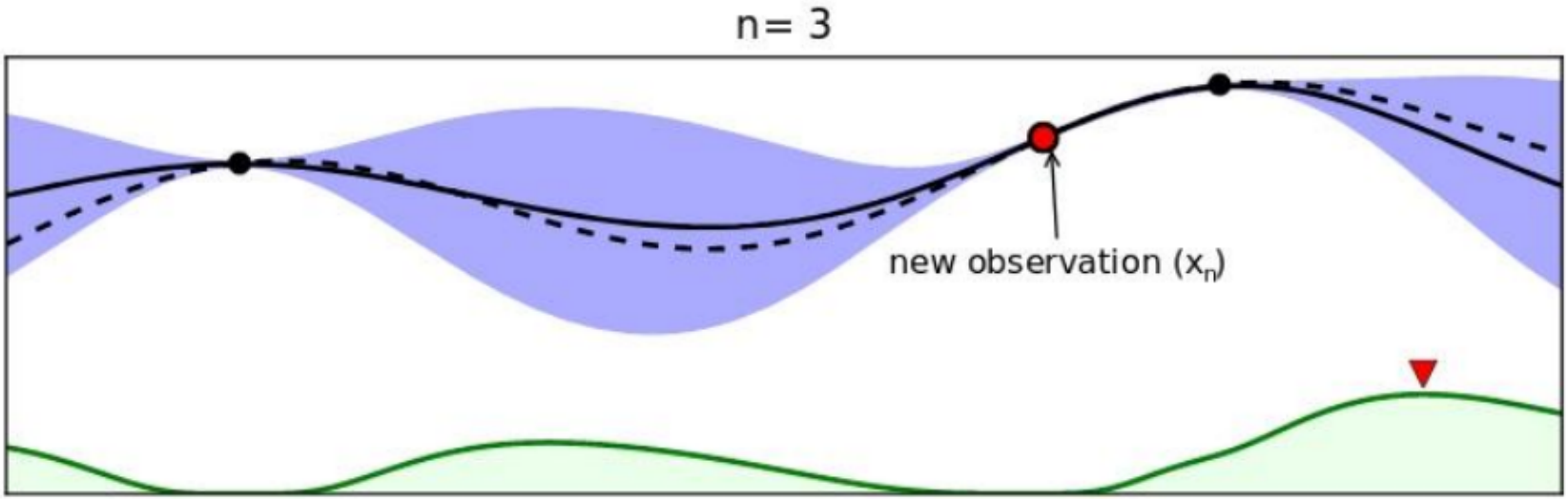
BO in a nutshell



BO in a nutshell



BO in a nutshell



Algorithm of BO in a nutshell

Algorithm Algorithm of BO

Initialization. Initialize hyperparameters
Define a maximum number of exploration steps $\bar{n}$;
while $n \leq \bar{n}$ **do**
  Find the next $x(n+1)$ via the acquisition function ;
  Deploy $x(n+1)$ in the system and observe $\tilde{f}(x(n+1))$;
  Update hyperparameters ;
  Set $n \leftarrow n+1$;
Result: Deploy $x^* = \underset{i=1,\ldots,n}{\arg\max}\, \tilde{f}(x(i))$

The benefits of BO

The benefits of BO

- **Quick convergence :** BO converges to a near optimal configuration in few iterations.
- **Safe exploration :** BO avoids sudden drops.
- **Deals with the exploration-exploitation trade-off :** BO is able to quantify the exploration-exploitation trade-off.

Gaussian process

Conditioning Gaussian process

Let us begin by splitting the components of Y into two disjoint sets A and B and decompose the representation as :

$$p([y_A, y_B]) = \mathcal{N}(\begin{bmatrix}\mu_A \\ \mu_B\end{bmatrix}, \begin{bmatrix}\Sigma_A & \Sigma_{AB} \\ \Sigma_{AB}^T & \Sigma_B\end{bmatrix})$$

Then the posterior probability $p([y_A|y_B])$ is a Gaussian variable as :

$$p([y_A|y_B]) = \mathcal{N}(\tilde{\mu}, \tilde{\Sigma}), \tag{1}$$

With $\tilde{\mu} = \mu_A + \Sigma_{AB}\Sigma_B^{-1}(y_B - \mu_B)$ and $\tilde{\Sigma} = \Sigma_A - \Sigma_{AB}\Sigma_B^{-1}\Sigma_{AB}^T$.

GP for the problem formulation

Assume that the unknown performance f to maximize is modeled GP.

In the past, the controller deployed $x(1), ..., x(n)$ and has observed the performance metrics :

$$o(n) = [\tilde{f}(x(1)), ..., \tilde{f}(x(n))]^T, \tag{2}$$

The value of the performance at any value x via the GP posterior probability is :

$$p(f(x)|o(n)) = \mathcal{N}(\mu_f + \Sigma_{f,o}\Sigma_o^{-1}(o(n) - \mu_o), \Sigma_f - \Sigma_{f,o}\Sigma_o^{-1}\Sigma_{f,o}^T), \tag{3}$$

GPs can infer the value of the performance function for configurations that have never been deployed in the system.

GP for the problem formulation

Assume that the unknown performance f to maximize is modeled GP.

In the past, the controller deployed $x(1), ..., x(n)$ and has observed the performance metrics :

$$o(n) = [\tilde{f}(x(1)), ..., \tilde{f}(x(n))]^T, \tag{3}$$

The value of the performance at any value x via the GP posterior probability is :

$$p(f(x)|o(n)) = \mathcal{N}(\mu_f + \Sigma_{f,o}\Sigma_o^{-1}(o(n) - \mu_o), \Sigma_f - \Sigma_{f,o}\Sigma_o^{-1}\Sigma_{f,o}^T), \tag{4}$$

GPs can infer the value of the performance function for configurations that have never been deployed in the system.

GP for the problem formulation

Assume that the unknown performance f to maximize is modeled GP.

In the past, the controller deployed $x(1), ..., x(n)$ and has observed the performance metrics :

$$o(n) = [\tilde{f}(x(1)), ..., \tilde{f}(x(n))]^T, \tag{3}$$

The value of the performance at any value x via the GP posterior probability is :

$$p(f(x)|o(n)) = \mathcal{N}(\mu_f + \Sigma_{f,o}\Sigma_o^{-1}(o(n) - \mu_o), \Sigma_f - \Sigma_{f,o}\Sigma_o^{-1}\Sigma_{f,o}^T), \tag{4}$$

GPs can infer the value of the performance function for configurations that have never been deployed in the system.

GP for the problem formulation

Assume that the unknown performance f to maximize is modeled GP.

In the past, the controller deployed $x(1), ..., x(n)$ and has observed the performance metrics :

$$o(n) = [\tilde{f}(x(1)), ..., \tilde{f}(x(n))]^T, \tag{3}$$

The value of the performance at any value x via the GP posterior probability is :

$$p(f(x)|o(n)) = \mathcal{N}(\mu_f + \Sigma_{f,o}\Sigma_o^{-1}(o(n) - \mu_o), \Sigma_f - \Sigma_{f,o}\Sigma_o^{-1}\Sigma_{f,o}^T), \tag{4}$$

GPs can infer the value of the performance function for configurations that have never been deployed in the system.

Selection of prior mean function & Covariance function

The mean vectors are defined via an prior mean $m : \mathcal{X} \to \mathbb{R}$, that defines the prior belief of the performance. Ideally, one would like $m \approx f$.

The covariance matrices $\Sigma_f, \Sigma_o, \Sigma_{f,o}$ are construct via a function $C : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that describes how close two parameters are. Ideally, $C(x, x') \approx Cov(f(x), f(x'))$.

C is usually defined via kernel functions at depend on the hyperparameters. A basic example of kernel function is the Radial Basis Function (RBF) :

$$K_\theta^{RBF}(x, x') = a\ exp(-\frac{||x - x'||^2}{2b^2}) \tag{5}$$

With $\theta = [a, b] > 0$ is the set of kernel hyperparameters.

Selection of prior mean function & Covariance function

The mean vectors are defined via an prior mean $m : \mathcal{X} \to \mathbb{R}$, that defines the prior belief of the performance. Ideally, one would like $m \approx f$.

The covariance matrices $\Sigma_f, \Sigma_o, \Sigma_{f,o}$ are construct via a function $C : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that describes how close two parameters are. Ideally, $C(x, x') \approx Cov(f(x), f(x'))$.

C is usually defined via kernel functions at depend on the hyperparameters. A basic example of kernel function is the Radial Basis Function (RBF) :

$$K_\theta^{RBF}(x, x') = a\ exp(-\frac{||x - x'||^2}{2b^2}) \tag{5}$$

With $\theta = [a, b] > 0$ is the set of kernel hyperparameters.

Selection of prior mean function & Covariance function

The mean vectors are defined via an prior mean $m : \mathcal{X} \to \mathbb{R}$, that defines the prior belief of the performance. Ideally, one would like $m \approx f$.

The covariance matrices $\Sigma_f, \Sigma_o, \Sigma_{f,o}$ are construct via a function $C : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that describes how close two parameters are. Ideally, $C(x, x') \approx Cov(f(x), f(x'))$.

C is usually defined via kernel functions at depend on the hyperparameters. An example of kernel function is the Radial Basis Function (RBF) :

$$K_\theta^{RBF}(x, x') = a\ exp(-\frac{||x - x'||^2}{2b^2}) \tag{5}$$

With $\theta = [a, b] > 0$ is the set of kernel hyperparameters.

Computing the GP posterior

Once the prior mean and covariance m, C are defined, one can plug them in the expression of the GP posterior (4) as follows :

We first define $\mu_f = [m(x)]$ the prior mean for the point x at which performance is inferred.

The prior mean vector at observed points $o(n)$ is $\mu_0 = [m(x(1)), ..., m(x(n))]^T$.

The covariance matrix Σ_o of the tested point is such that $[\Sigma_o]_{i,j} = C_{\theta,\sigma}(x(i), x(j))$.

The (mono-dimensional) variance of the performance $f(x)$ to be inferred at the yet-to-be-tested x is $\Sigma_f = C_{\theta,\sigma}(x, x)$.

Finally, $\Sigma_{f,o}$ denotes the covariance column vector between $f(x)$ and the observations $o(n)$, namely, $[\Sigma_{f,o}]_i = C_\theta(x, x(i))$.

Computing the GP posterior

Once the prior mean and covariance m, C are defined, one can plug them in the expression of the GP posterior (4) as follows :

We first define $\mu_f = [m(x)]$ the prior mean for the point x at which performance is inferred.

The prior mean vector at observed points $o(n)$ is $\mu_0 = [m(x(1)), ..., m(x(n))]^T$.

The covariance matrix Σ_o of the tested point is such that $[\Sigma_o]_{i,j} = C_\theta(x(i), x(j))$.

The (mono-dimensional) variance of the performance $f(x)$ to be inferred at the yet-to-be-tested x is $\Sigma_f = C_\theta(x, x)$.

Finally, $\Sigma_{f,o}$ denotes the covariance column vector between $f(x)$ and the observations $o(n)$, namely, $[\Sigma_{f,o}]_i = C_\theta(x, x(i))$.

Computing the GP posterior

Once the prior mean and covariance m, C are defined, one can plug them in the expression of the GP posterior (4) as follows :

We first define $\mu_f = [m(x)]$ the prior mean for the point x at which performance is inferred.

The prior mean vector at observed points $o(n)$ is $\mu_0 = [m(x(1)), ..., m(x(n))]^T$.

The covariance matrix Σ_o of the tested point is such that $[\Sigma_o]_{i,j} = C_\theta(x(i), x(j))$.

The (mono-dimensional) variance of the performance $f(x)$ to be inferred at the yet-to-be-tested x is $\Sigma_f = C_\theta(x, x)$.

Finally, $\Sigma_{f,o}$ denotes the covariance column vector between $f(x)$ and the observations $o(n)$, namely, $[\Sigma_{f,o}]_i = C_\theta(x, x(i))$.

Computing the GP posterior

Once the prior mean and covariance m, C are defined, one can plug them in the expression of the GP posterior (4) as follows :

We first define $\mu_f = [m(x)]$ the prior mean for the point x at which performance is inferred.

The prior mean vector at observed points $o(n)$ is $\mu_0 = [m(x(1)), ..., m(x(n))]^T$.

The covariance matrix Σ_o of the tested point is such that $[\Sigma_o]_{i,j} = C_\theta(x(i), x(j))$.

The (mono-dimensional) variance of the performance $f(x)$ to be inferred at the yet-to-be-tested x is $\Sigma_f = C_\theta(x, x)$.

Finally, $\Sigma_{f,o}$ denotes the covariance column vector between $f(x)$ and the observations $o(n)$, namely, $[\Sigma_{f,o}]_i = C_\theta(x, x(i))$.

Computing the GP posterior

Once the prior mean and covariance m, C are defined, one can plug them in the expression of the GP posterior (4) as follows :

We first define $\mu_f = [m(x)]$ the prior mean for the point x at which performance is inferred.

The prior mean vector at observed points $o(n)$ is $\mu_0 = [m(x(1)), ..., m(x(n))]^T$.

The covariance matrix Σ_o of the tested point is such that $[\Sigma_o]_{i,j} = C_\theta(x(i), x(j))$.

The (mono-dimensional) variance of the performance $f(x)$ to be inferred at the yet-to-be-tested x is $\Sigma_f = C_\theta(x, x)$.

Finally, $\Sigma_{f,o}$ denotes the covariance column vector between $f(x)$ and the observations $o(n)$, namely, $[\Sigma_{f,o}]_i = C_\theta(x, x(i))$.

Computing the GP posterior

Once the prior mean and covariance m, C are defined, one can plug them in the expression of the GP posterior (4) as follows :

We first define $\mu_f = [m(x)]$ the prior mean for the point x at which performance is inferred.

The prior mean vector at observed points $o(n)$ is $\mu_0 = [m(x(1)), ..., m(x(n))]^T$.

The covariance matrix Σ_o of the tested point is such that $[\Sigma_o]_{i,j} = C_\theta(x(i), x(j))$.

The (mono-dimensional) variance of the performance $f(x)$ to be inferred at the yet-to-be-tested x is $\Sigma_f = C_\theta(x, x)$.

Finally, $\Sigma_{f,o}$ denotes the covariance column vector between $f(x)$ and the observations $o(n)$, namely, $[\Sigma_{f,o}]_i = C_\theta(x, x(i))$.

Hyperparameter tuning

The covariance function C is parametrized by the kernel hyperparameters θ.

This is performed by maximum likelihood of the observed function values $o(n)$,

$$\arg\max_{\theta} p(x(1), ..., x(n)) := N(\mu_o, \Sigma_o), \tag{9}$$

μ_o, Σ_o are the mean prior and covariance of $o(n) = [\tilde{f}(x(1)), ..., \tilde{f}(x(n))]^T$,

It is generally a non-convex problem, and one usually looks for a local maximum.

Hyperparameter tuning

The covariance function C is parametrized by the kernel hyperparameters θ.

This is performed by maximum likelihood of the observed function values $o(n)$,

$$\arg\max_{\theta} p(x(1), ..., x(n)) := N(\mu_o, \Sigma_o), \tag{9}$$

μ_o, Σ_o are the mean prior and covariance of $o(n) = [\tilde{f}(x(1)), ..., \tilde{f}(x(n))]^T$,

It is generally a non-convex problem, and one usually looks for a local maximum.

Hyperparameter tuning

The covariance function C is parametrized by the kernel hyperparameters θ.

This is performed by maximum likelihood of the observed function values $o(n)$,

$$\arg\max_{\theta} p(x(1), ..., x(n)) := N(\mu_o, \Sigma_o), \tag{9}$$

μ_o, Σ_o are the mean prior and covariance of $o(n) = [\tilde{f}(x(1)), ..., \tilde{f}(x(n))]^T$,

It is generally a non-convex problem, and one usually looks for a local maximum.

Acquisition function

At step $n+1$, BO optimizes an acquisition function $u(.|o(n))$ and chooses the next $x(n+1)$:

$$x(n+1) = \arg\max_{x \in \mathcal{X}} u(x|o(n))), \tag{10}$$

With u depends on the previous observations via the GP posterior.

There are several ways to define the acquisition function under different assumptions on the observation noise.

Acquisition function

At step $n+1$, BO optimizes an acquisition function $u(.|o(n))$ and chooses the next $x(n+1)$:

$$x(n+1) = \arg\max_{x \in \mathcal{X}} u(x|o(n))), \tag{10}$$

With u depends on the previous observations via the GP posterior.

There are several ways to define the acquisition function under different assumptions on the observation noise.

Acquisition function

At step $n+1$, BO optimizes an acquisition function $u(.|o(n))$ and chooses the next $x(n+1)$:

$$x(n+1) = \arg\max_{x \in \mathcal{X}} u(x|o(n))), \tag{10}$$

With u depends on the previous observations via the GP posterior.

There are several ways to define the acquisition function under different assumptions on the observation noise.

Acquisition function [expected Improvement]

Expected Improvement (EI) is arguably the most widely known acquisition function.

At time $n+2$ we select the best point so far, the performance at time $n+2$ is $\max\{f(x(n+1)), \max_{i=1,\dots,n} f(x(i))\}$.

$x(n+1)$ maximizes the expected performance at time $n+2$ iff maximizes the expected improvement u^{EI} with respect to the best configuration so far, with :

$$u^{EI}(x|o(n)) = \mathbb{E}[f(x) - \max_{i=1,\dots,n} f(x(i))|o(n)]^{+} \tag{11}$$

Acquisition function [expected Improvement]

Expected Improvement (EI) is arguably the most widely known acquisition function.

At time $n+2$ we select the best point so far, the performance at time $n+2$ is $\max\{f(x(n+1)), \max_{i=1,\dots,n} f(x(i))\}$.

$x(n+1)$ maximizes the expected performance at time $n+2$ iff maximizes the expected improvement u^{EI} with respect to the best configuration so far, with :

$$u^{EI}(x|o(n)) = \mathbb{E}[f(x) - \max_{i=1,\dots,n} f(x(i))|o(n)]^{+} \tag{11}$$

Acquisition function [expected Improvement]

Expected Improvement (EI) is arguably the most widely known acquisition function.

At time $n+2$ we select the best point so far, the performance at time $n+2$ is $\max\{f(x(n+1)), \max_{i=1,\ldots,n} f(x(i))\}$.

$x(n+1)$ maximizes the expected performance at time $n+2$ iff maximizes the expected improvement u^{EI} with respect to the best configuration so far, with :

$$u^{EI}(x|o(n)) = \mathbb{E}[f(x) - \max_{i=1,\ldots,n} f(x(i))|o(n)]^{+} \tag{11}$$

Acquisition function [expected Improvement]

u^{EI} has a closed form for GP as :

$$u^{EI}(x|o(n)) = (\mathbb{E}(f(x)|o(n)) - \max_{i=1,\dots,n} f(x(i)))\Phi(Z) + Std[f(x)|o(n)]\varphi(Z), \forall x \in \mathcal{X}, \tag{12}$$

Z is defined as :

$$Z = \frac{\mathbb{E}(f(x)|o(n)) - \max_{i=1,\dots,n} f(x(i))}{Std[f(x)|o(n)]} \tag{13}$$

In general u^{EI} is not well defined.

Acquisition function [expected Improvement]

u^{EI} has a closed form for GP as :

$$u^{EI}(x|o(n)) = (\mathbb{E}(f(x)|o(n)) - \max_{i=1,\dots,n} f(x(i)))\Phi(Z) + Std[f(x)|o(n)]\varphi(Z), \forall x \in \mathcal{X}, \tag{12}$$

Z is defined as :

$$Z = \frac{\mathbb{E}(f(x)|o(n)) - \max_{i=1,\dots,n} f(x(i))}{Std[f(x)|o(n)]} \tag{13}$$

In general u^{EI} is not well defined.

Acquisition function [expected Improvement]

u^{EI} has a closed form for GP as :

$$u^{EI}(x|o(n)) = (\mathbb{E}(f(x)|o(n)) - \max_{i=1,\dots,n} f(x(i)))\Phi(Z) + Std[f(x)|o(n)]\varphi(Z), \forall x \in \mathcal{X}, \tag{12}$$

Z is defined as :

$$Z = \frac{\mathbb{E}(f(x)|o(n)) - \max_{i=1,\dots,n} f(x(i))}{Std[f(x)|o(n)]} \tag{13}$$

In general u^{EI} is not well defined.

Bayesian optimization

Algorithm Bayesian optimization

Initialization. Set $n = 0$;
Define a prior mean function m. By default, $m = 0$;
Choose the covariance kernel function K_θ;
Initialize the hyperparameters θ;
Define a termination threshold ϵ and a maximum number of exploration steps $\bar{n}$;
while $\max_{x \in \mathcal{X}} u(x|o(n)) \geq \epsilon$ **or** $n \leq \bar{n}$ **do**
  Find the next $x(n+1)$ via the acquisition function (10) ;
  Deploy $x(n+1)$ in the system and observe $\tilde{f}(x(n+1))$;
  Update hyperparameters θ via (9) ;
  Set $n \leftarrow n + 1$;

Result: Deploy $x^* = \underset{i=1,\ldots,n}{\arg\max}\, \tilde{f}(x(i))$