Gaussian Processes and Bayesian Optimization

Andrea Ponti
a.ponti5@campus.unimib.it

1 Introduction

Gathering data is often expensive, and sometimes getting exactly the data you want is even impossible. Surrogate model could represent a much cheaper way to explore relationship. A surrogate can be seen as a substitute for the real thing. Gaussian processes are a powerful tool used in Machine Learning to make predictions about data by incorporating some prior knowledge. The most obvious application area of Gaussian processes is fitting a function the the data, i.e., in problems like regression or time series forecasting. Given a set of points, there may be a, potentially infinite, number of functions that could fit the data. Gaussian processes offer an elegant solution to this problem, by assigning a probability to each of these function. Gaussian process models have gained a lot of success for non-linear non-parametric regression in many fields, such as Machine Learning and spatial/geo-statistics, in particular when modelling data believed to be smooth. Gaussian processes are often used also in Bayesian Optimization, as surrogate models, to approximate a, usually black-box and expensive-to-evaluate, objective function.

In the following, after a brief introduction to Gaussian distributions in Section 2, the definition and properties of Gaussian Process Regression are described in Section 3; then, in Section 4, the definition of kernel functions and some examples are presented. In Section 5 is shown how the Gaussian processes are useful in an optimization framework, and in Section 6 some experiments demonstrate the efficacy and the efficiency of Bayesian Optimization. Finally, Section 7 presents the conclusions.

All software used for the experiments and to generate the figures is available on Github[1].

2 Gaussian Distributions

Before introducing Gaussian Process Regression, let review the definition and properties of Gaussian distributions. A random variable X is Gaussian (or normally) distributed, $X \sim \mathcal{N}(\mu, \sigma^2)$, where μ is the mean and σ^2 is the variance of the distribution, if its Probability Density Function (PDF) is:

$$F_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}. \tag{1}$$

[1] https://github.com/andreaponti5/GP-BO

Gaussian distributions occur very often in real world data. Their importance is mainly given by the Central Limit Theorem (CLT). It states that, under some conditions, the arithmetic mean of $m > 0$ samples of a random variable with finite mean and variance is itself a random variable, that converge in distribution to a Gaussian. An illustration of the CLT is given in Figure 1. It is evident that increasing the sample size m, the histogram tends to take the typical bell-shape of a normal distribution.

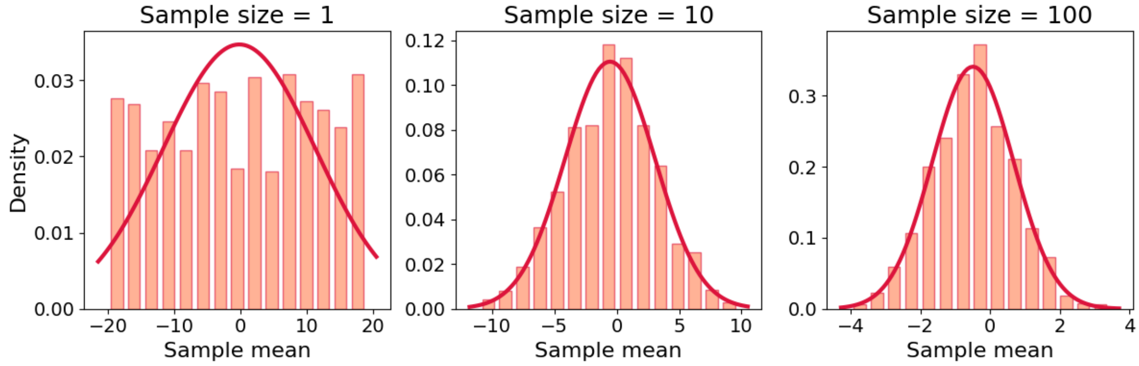

Figure 1: Sample mean distributions of samples with size $m = 1$, $m = 10$ and $m = 100$, respectively.

It is a common situation to have more "feature" variables that are correlated to each other. It is possible to generalize the previous definitions to the multivariate case.
A k-dimensional random vector $X = (X_1, ..., X_k)^T$ is distributed as a multivariate Gaussian distribution, $X \sim \mathcal{N}(\mu, \Sigma)$ if its PDF is:

$$F_X(x) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \exp\left\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right\}, \tag{2}$$

where $\mu = \mathbb{E}[x] \in \mathbb{R}^k$ is the mean vector and $\Sigma = cov(x)$ is the $k \times k$ covariance matrix. In Figure 2 is shown an example of a bi-variate Gaussian distribution. Visually, the distribution is centered around the mean and the covariance matrix defines its shape.
One key property of Gaussian distributions is the closeness under conditioning and under marginalization. This means that the resulting distributions from these operations are also Gaussian, which makes many problem in statics and machine learning tractable. Consider the distribution $X = (X_1, X_2) \sim \mathcal{N}(\mu, \Sigma)$ where $\mu = (\mu_1, \mu_2)^T$ is the mean vector and $\Sigma = \left(\begin{smallmatrix}\Sigma_{11} & \Sigma 12 \\ \Sigma_{21} & \Sigma_{22}\end{smallmatrix}\right)$ is the covariance matrix. The marginal distributions in Equations 3 and 4

$$F(X_1) = \int_{X_2} F(X_1, X_2; \mu, \Sigma) dX_2 \tag{3}$$

$$F(X_2) = \int_{X_1} F(X_1, X_2; \mu, \Sigma) dX_1 \tag{4}$$

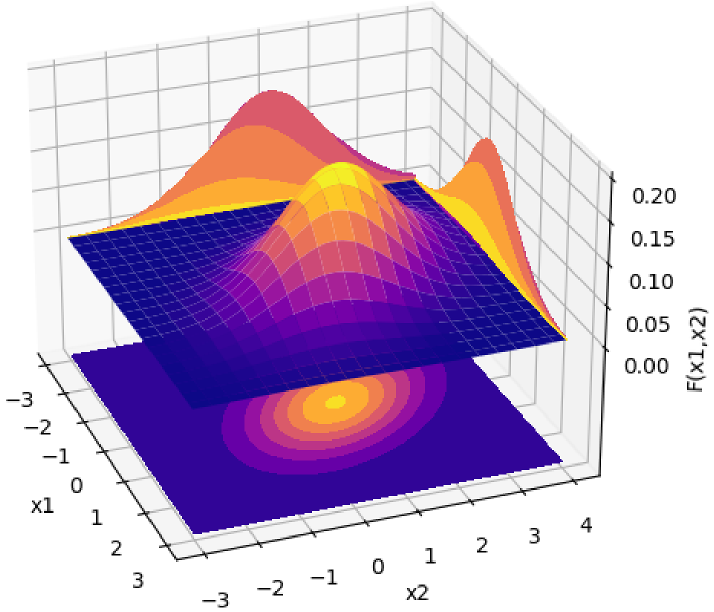

Figure 2: An example of a bi-variate Gaussian distribution $X = (x_1, x_2) \sim \mathcal{N}(\mu, \Sigma)$ where $\mu = (0, 1)^T$ and $\Sigma = \left(\begin{smallmatrix} 1 & -0.5 \\ -0.5 & 1.5 \end{smallmatrix}\right)$.

are Gaussian distributed (Equations 5 and 6), i.e., each partition X_1 and X_2 only depends on its corresponding entries in μ and Σ.

$$X_1 \sim \mathcal{N}(\mu_1, \Sigma_{11}) \tag{5}$$

$$X_2 \sim \mathcal{N}(\mu_2, \Sigma_{22}) \tag{6}$$

The conditional distribution $X_1|X_2$ (Equation 7) is also Gaussian distributed, as in Equation 8.

$$F(X_1|X_2) = \frac{F(X_1, X_2; \mu, \Sigma)}{\int_{X_1} F(X_1, X_2; \mu, \Sigma) dX_1} \tag{7}$$

$$X_1|X_2 \sim \mathcal{N}(\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}) \tag{8}$$

It is important to note that the new mean only depends on the conditioned variable, while the new covariance matrix is independent from this variable.
The closeness of the Gaussian distributions with respect to marginalization and conditioning operations will be useful in deriving Gaussian Process predictions.

3 Gaussian Process Regression

Considering the concepts introduced in the previous section, it is now possible to define the Gaussian processes and to show how they can be used to tackle regression problems. So, let consider an unknown function $f(x)$, the aim of a regression model is to approximate the function given a set of observations $\mathcal{D} = \{(x^{(i)}, y^{(i)} \,|\, i = 1, ..., n\}$, where x denotes an input vector (covariates) of dimension D and y denotes a scalar output, also called target. There are several ways to interpret Gaussian process (GP) regression models. In the so called *function-space view* [3], a GP describes a distribution over functions; formally, a Gaussian process is a (potentially infinite) collection of random variables, any finite number of which have a joint Gaussian distribution. In other words, the joint distribution of every finite subset of random variables is multi-variate Gaussian. A GP is completely specified by its mean function $\mu(x)$ and its covariance function $k(x, x')$

$$\mu(x) = \mathbb{E}[f(x)], \tag{9}$$

$$k(x, x') = \mathbb{E}[(f(x) - \mu(x))(f(x') - \mu(x'))], \tag{10}$$

and it is defined as:

$$f(x) \sim \mathcal{GP}(\mu(x), k(x, x')). \tag{11}$$

The covariance function assumes a critical role in the GP modelling as it specifies the distribution over functions. It describes the "shape" of data: if two points x_i and x_j are considered similar by the covariance function, their observations $f(x_i)$ and $f(x_j)$ are expected to be similar.

Without loss of generality, assume the mean to be zero $\mu = 0$; consider a set of input points $X_{1:n} = (x_1, ..., x_n)^T$ and compute the covariance matrix element-wise; it is then possible to generate a random Gaussian vector as in Equation 12.

$$f(X_{1:n}) \sim \mathcal{N}(0, K(X_{1:n}, X_{1:n})). \tag{12}$$

This is usually known as *sampling from prior*; Figure 3 gives an example of 5 different samples from the prior of a GP.

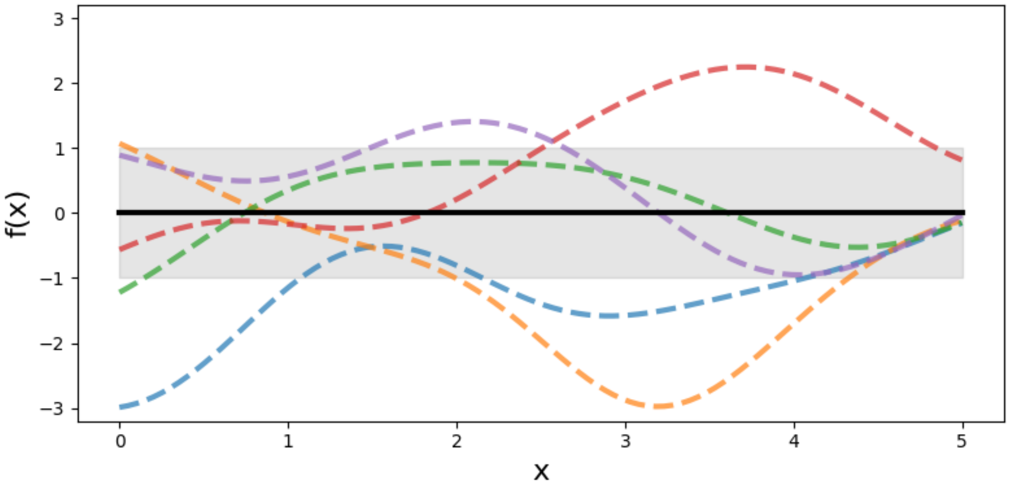

Figure 3: Five samples from the prior of a GP with zero-mean and the squared-exponential kernel.

Usually, the interest is not in drawing random functions from the prior, but it is in incorporate the knowledge about the function obtained through the evaluations performed so far. Consider the set of observations $\mathcal{D} = \{(X_{1:n}, Y_{1:n})\}$ and a new point x_*, then the joint distribution of $Y_{1:n}$ and y_* is expressed as:

$$\begin{bmatrix} Y_{1:n} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(X_{1:n}) \\ \mu(x_*) \end{bmatrix}, \begin{bmatrix} \mathbf{K} & \mathbf{K}_* \\ \mathbf{K}_*^T & \mathbf{K}_{**} \end{bmatrix}\right) \tag{13}$$

where $\mathbf{K} = K(X_{1:n}, X_{1:n})$, $\mathbf{K}_* = K(X_{1:n}, x_*)$ and $\mathbf{K}_{**} = K(x_*, x_*)$. It is now possible to predict the value y_* of the new point x_* using the conditional distribution:

$$y_* \mid Y_{1:n}, X_{1:n}, x_* \sim \mathcal{N}\left(\mathbf{K}_*^T \mathbf{K}^{-1} Y_{1:n},\ \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}^{-1} \mathbf{K}_*\right). \tag{14}$$

In many applications, it is not possible to compute the true function $f(x_i)$ and so the observed values can be noisy $y_i = f(x_i) + \varepsilon_i$. Assuming additive independent and identically distributed Gaussian noise with variance λ^2, the prior on the noisy observations becomes $cov(Y_{1:n}) = \mathbf{K} + \lambda^2 I$. Therefore, the predictive mean $\mu(x_*)$ and the predictive variance σ^2 of the new point x^* can be easily updated by conditioning the joint Gaussian prior distribution on the observations:

$$\mu(x_*) = \mathbf{K}_*^T \left[\mathbf{K} + \lambda^2 I\right]^{-1} Y_{1:n} \tag{15}$$

$$\sigma^2(x_*) = \mathbf{K}_{**} - \mathbf{K}_*^T \left[\mathbf{K} + \lambda^2 I\right]^{-1} \mathbf{K}_* \tag{16}$$

Figure 4 shows five different samples randomly drawn from a GP posterior conditioned to ten function observations.

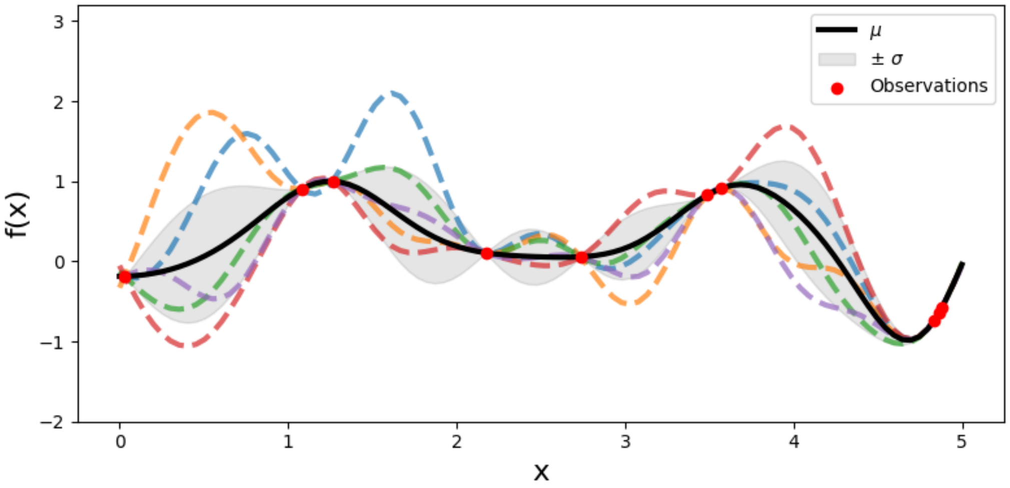

Figure 4: Five samples from the posterior of a GP.

4 Kernel

Covariance functions, often called kernels, are the key element of Gaussian processes, as they encodes assumptions about the function to approximate: points that are close in the input space are likely to have similar objective values. The covariance matrix Σ not only describes the shape of the distribution, but it also determines the characteristics of the function to predict. The entry Σ_{ij} of the covariance matrix describes how much influence the i-th and j-th points have on each other. Therefore, kernels control the possible shape that a fitted function can adopt. The main advantage of using a kernel instead of a standard similarity measure, as the Euclidean distance, is that kernels conceptually embed the input data into a higher dimensional space in which the similarity is then computed.
Kernels can be divided into two main categories. Stationary kernels are functions of $x - x'$, so they are invariant to translations, and the covariance of two points is only dependent on their relative position. In the case in which they are functions of $|x-x'|$ they are invariant to all rigid motions and they are called isotropic. Non-stationary kernels instead are function of the dot-product $x \cdot x'$, and so they depend on the absolute location. They are invariant to the rotation respect to the origin but not to translations.
Finally, a kernel k, to be a covariance function, have to satisfy the following conditions [1]:

- It has to be simmetric: $k(x, x') = k(x', x)$;
- The matrix K, with entries $K_{ij} = k(x_i, x_j)$, must be positive semidefinite.

The most widely used covariance function is the Squared Exponential (SE) kernel, also known as Radial Basis Function or Gaussian kernel, and it is defined as:

$$k_{SE}(x, x') = \sigma^2 \exp\left\{-\frac{||x - x'||^2}{2\ell^2}\right\} \tag{17}$$

where the lenghtscale ℓ determines the smoothness of the function and the output variance σ^2 indicates the average distance of the function away from its mean. This last hyperparameter is common for each kernel function and it is just a scale factor. The SE kernel is infinitely differentiable, bringing to a very smooth Gaussian process.
Another commonly used kernel is the Màtern kernel defined as:

$$k_{Mat}(x, x') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{|x - x'|\sqrt{2\nu}}{\ell}\right)^{\nu} K_{\nu} \left(\frac{|x - x'|\sqrt{2\nu}}{\ell}\right) \tag{18}$$

where Γ is the Gamma function, K_ν is the modified Bessel function, and ℓ and ν are two positive hyperparameters. This covariance function become particularly simple when ν is half-integer: $\nu = p + \frac{1}{2}$ where p is a non-negative integer. In machine learning the most used values are $\nu = \frac{3}{2}$ and $\nu = \frac{5}{2}$. It is important to note that for $\nu \to \infty$ the Màtern kernel corresponds to the Squared Exponential kernel.
The Rational Quadratic kernel instead can be derived as a scale mixture (an infinite sum)

of Squared Exponential covariance functions, and it is defined as

$$k_{RQ}(x, x') = \left[1 + \frac{(x - x')^2}{2\alpha\ell^2}\right]^{-\alpha} \tag{19}$$

where α and ℓ are two positive hyperparameters. Therefore, the sum of kernel functions is still a kernel. Figure 5 shows some examples of the kernel matrices obtained with different covariance functions.

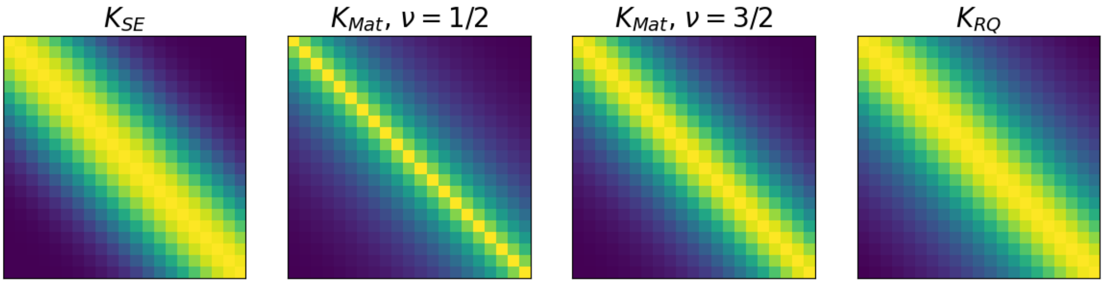

Figure 5: Four examples of kernel matrix $k(X, X)$ with $X = (-10, -9, ..., 9, 10)$ and $\ell = 5.0$.

5 Bayesian Optimization

Gaussian process surrogate models can drive the optimization of black-box function, i.e., functions with unknown structure. Bayesian Optimization (BO) is an active learning or sequential design strategy for global optimization of black-box, usually expensive-to-evaluate, functions. The addressed problem can be stated as follows:

$$x^* = \arg\min_{x \in \mathcal{X}} f(x) \tag{20}$$

where $\mathcal{X}$ is a hyperrectangle, a bounding box or, more in general, a constrained region. BO falls in the *derivative-free* optimization methods, therefore no information about the derivatives of the objective function $f(x)$ is needed. In this context, two kind of uncertainty can be faced:

- *Measurement uncertainty*: usually the observations of the objective function are noisy.
- *Structural uncertainty*: for example, considering three noise-free evaluations of the objective function, there could be an infinite number of functions with different minima, compatible with the three observations.

The general idea of BO is to create a surrogate model of the objective function and build on it a sequential, model-based optimizer. The surrogate model is initially built on a design set $\mathcal{D} = \{(x^{(i)}, y^{(i)}) \,|\, i = 1, ..., n\}$ and the next point to evaluate $x^{(n+1)}$ is chosen optimizing an acquisition (or utility) function α. Then the surrogate model is updated considering the evaluation of this new point $(x^{(n+1)}, y^{(n+1)})$. This process is repeated until a given budget of function evaluations is reached.

The acquisition function is the mechanism to balance exploration and exploitation in BO, where exploiting means to consider the area providing more chance to improve the current solution while exploring means to move towards less explored regions of the search space. In particular, acquisition functions aim to guide the search of the optimum towards points with potentially a small number of function evaluations. Two of the most used acquisition functions are Expected Improvement (EI) and Confidence Bound (Lower Confidence Bound, LCB, for minimization and Upper Confidence Bound, UCB, for maximization), defined as follows:

$$EI(x) = [\mu(x) - y^+]\Phi\left[\frac{\mu(x) - y^+}{\sigma(x)}\right] + \sigma(x)\phi\left[\frac{\mu(x) - y^+}{\sigma(x)}\right] \tag{21}$$

$$LCB(x) = \mu(x) + \sqrt{\beta}\sigma(x) \tag{22}$$

where Φ is the standard normal cumulative distribution function, ϕ is the standard normal probability density function and β is a hyperparameter to balance between exploration and exploitation. Many other acquisition functions exist in the literature [2].

6 Experiments

Consider an optimization problem in the form of Equation 20, with the objective function defined as:

$$f(x) = -x \sin x \tag{23}$$

where $x \in [0, 10]$ and the minima is $f(x) = -7.9167$ in $x = 7.9787$, as shown in Figure 6.

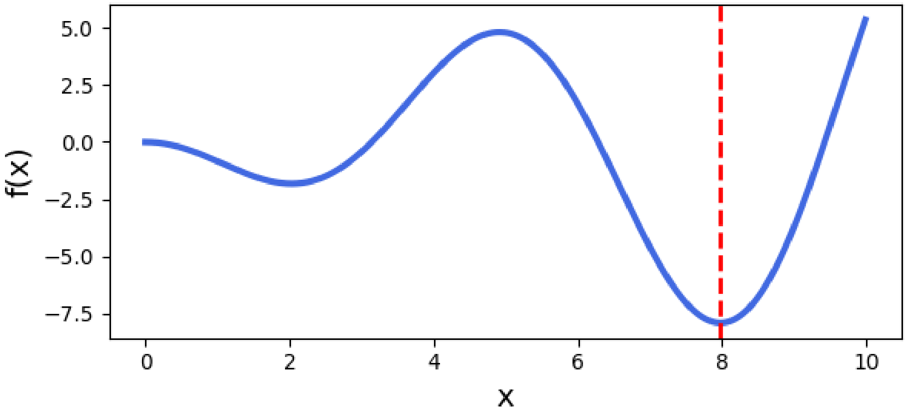

Figure 6: The objective function considered in the experiments. The dotted red line shows the point of minimum.

Let then build a Gaussian Process considering 5 initial evaluation of the objective function, as displayed in the top-left charts in Figure 7. In this experiment the Màtern kernel with $\nu = \frac{5}{2}$ is used. Optimizing the acquisition function, Expected Improvement in this case, it is possible to find the next point to evaluate. Then, the GP is updated and the process is repeated. Figure 7 shows 5 BO iterations. It is clear how, BO, in just a few evaluations is able to find the optimal point of the function. This shows the sample efficiency of BO.

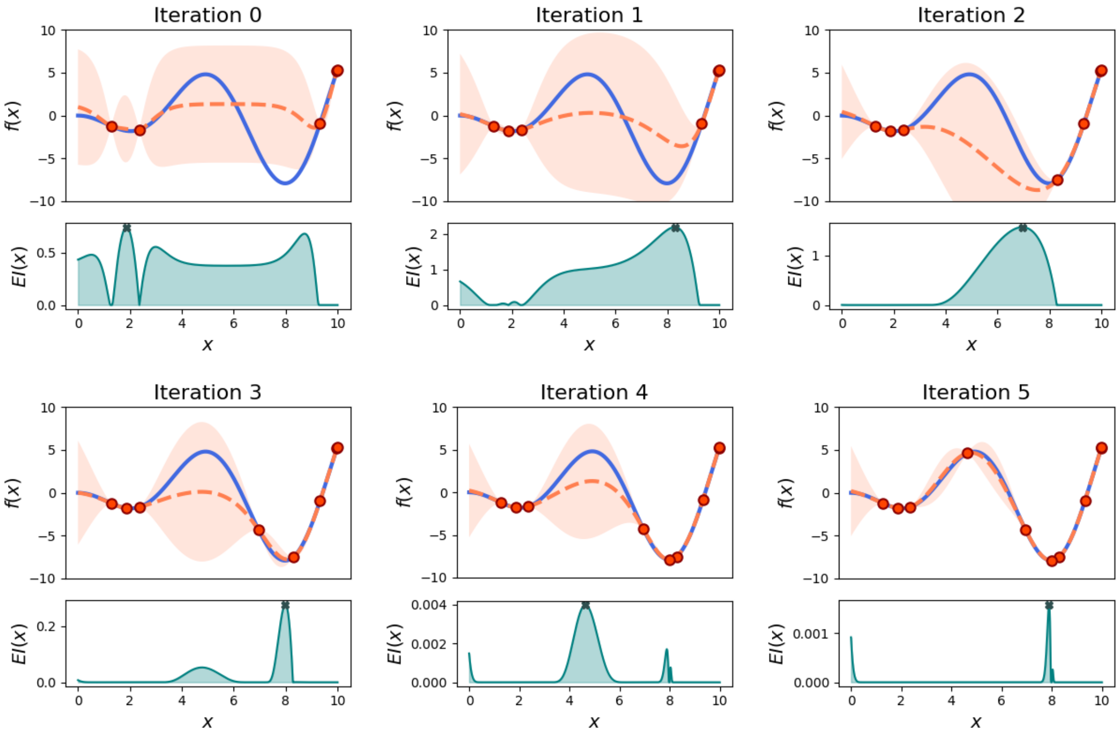

Figure 7: Five iteration of BO. For each iteration the plot shows: *(i)* the true objective function (blue line) and the observation so far (red points); *(ii)* the predicted mean (orange dotted line) and the uncertainty of the prediction, i.e., the standard deviation (shaded orange region); *(iii)* the value of the acquisition function (green) and the next point to evaluate (green cross).

7 Conclusions

Gaussian Processes provide a rich and flexible class of non-parametric statistical models over function spaces. GPs privilege modeling through a covariance structure, rather than through the mean, which allows for a more fine control and for non-linearities to manifest in a relative rather than absolute sense. GPs are a powerful tool used not only in regression problems, but they are also used as surrogate in optimization framework to model black-box and expensive functions. They are the key tool, together with an acquisition function, in Bayesian optimization, whose effectiveness and efficiency is shown in the experiments in the previous section.

Gaussian processes have also some drawbacks, in particular they can be very slow due to the inversion of the kernel matrix, whose computational complexity can explode as the dimension grows. Like every non-parametric methods, another disadvantage of GPs is that inspecting estimated coefficients isn't directly helpful for understanding. In addition, even though GPs are very flexible, particularly compared to ordinary linear models, sometimes they are too rigid and tend to oversmooth. Whether or not these are big deals depends on the specific application domain.

References

[1] F. Archetti and A. Candelieri. *Bayesian optimization and data science*. Springer, 2019.

[2] R. B. Gramacy. *Surrogates: Gaussian process modeling, design, and optimization for the applied sciences*. Chapman and Hall/CRC, 2020.

[3] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006.