APPENDIX $\mathcal{E}$

Gaussian Processes

Gaussian Processes are a powerful tool for approximating unknown static mapping from an input space into an output one.

E.0.1 Scalar case

Suppose we need to approximate a function g defined as follows:

$$\begin{aligned} g : \mathcal{X} \rightarrow \mathcal{Y} \\ \mathcal{X} \subseteq \mathbb{R}^n \quad \mathcal{Y} \subseteq \mathbb{R} \end{aligned} \tag{E.1}$$

g is unknown and the only available information is represented by a training set S made of N samples $\left[\begin{array}{c} X^i \in \mathcal{X} \\ \hline Y^i \in \mathcal{Y} \end{array}\right]$:

$$S = \langle \begin{bmatrix} X^1 \\ Y^1 \end{bmatrix}, \cdots, \begin{bmatrix} X^N \\ Y^N \end{bmatrix} \rangle \tag{E.2}$$

Since the values in S were generated by the same function g, they are in some way correlated. However, such a correlation is not known precisely. For this reason, Gaussian Processes approximate this correlation, assuming that all the values in S are jointly

Gaussians, *i.e.*:

$$\begin{aligned}
\begin{bmatrix} Y^1 \\ \vdots \\ Y^N \end{bmatrix} &\sim \mathcal{N}(0, K(X^{1,\cdots,N})) \\
\mathbb{P}\left(\begin{bmatrix} Y^1 \\ \vdots \\ Y^N \end{bmatrix} \right) &= \frac{1}{\sqrt{(2\pi)^N |K|}} exp\left(-\frac{1}{2} \begin{bmatrix} Y^1 & \cdots & Y^N \end{bmatrix} K^{-1} \begin{bmatrix} Y^1 \\ \vdots \\ Y^N \end{bmatrix} \right) \\
\mathbb{P}\left(\begin{bmatrix} Y^1 \\ \vdots \\ Y^N \end{bmatrix} \right) &= \frac{1}{\sqrt{(2\pi)^N |K|}} exp\left(-\frac{1}{2} Tr\left(K^{-1} \begin{bmatrix} Y^1 \\ \vdots \\ Y^N \end{bmatrix} \begin{bmatrix} Y^1 & \vdots & Y^N \end{bmatrix} \right) \right)
\end{aligned} \tag{E.3}$$

The covariance matrix K, is a function of the inputs in the training set and it's defined as follows:

$$K = \begin{bmatrix} k(X^1, X^1) & \cdots & k(X^1, X^N) \\ \vdots & \ddots & \vdots \\ k(X^N, X^1) & \cdots & k(X^N, X^N) \end{bmatrix} \tag{E.4}$$

k is the kernel function and it's part of the model. As a general prescription, k must be defined in order to obtain a symmetric positive definite matrix K. For this reason, for any kind of kernel function it holds that $k(x, x^{'}) = k(x^{'}, x)$. k should be defined in order to assume low values for those entries that are strictly correlated. For example, when dealing with periodic function g, the kernel function k should be able to catch the periodicity, assuming a low value for a pair $x, x^{'}$ that is separated by approximately the value of the period. Common adopted functions are Radial Basis Function, Rational Quadratic kernel, Linear kernel, Periodic kernel, etc. [110].
A certain number of tunable parameters $\theta_{1,2,...}$, called hyperparameters, characterize the kernel function $k(\theta_{1,2,...})$. $\theta_{1,2,...}$ together with the training set S are actually what characterize a Gaussian Process model. The values of $\theta_{1,2,...}$ are determined after training, see Section E.0.1 and E.0.2. A Gaussian Process can be exploited for predicting the value assumed by $g(X)$ in a point X not present in S, see E.0.1. In other words, function g is approximated with a Gaussian Process $g_{GP}(X)$.

Prediction

Knowing S and $\theta_{1,2,...}$, a prediction $Y = g(X)$ for a generic entry X can be made. Indeed, $Y(X) = g(X)$ and the population of outputs present in S are assumed as

jointly Gaussian:

$$\left[\begin{array}{c} Y(X) \\ \hline Y^1 \\ \vdots \\ Y^N \end{array}\right] \sim \mathcal{N}\left(\left[\begin{array}{c} 0 \\ \hline 0 \end{array}\right], \left[\begin{array}{c|c} k(X,X) & k_X(X) \\ \hline k_X(X)^T & K \end{array}\right], 0 \right)$$

$$k_X(X) = \begin{bmatrix} k(X,X^1) & \cdots & k(X,X^N) \end{bmatrix} \tag{E.5}$$

Therefore, since $Y^{1,\cdots,N}$ are known, the conditional distribution is assumed as a prediction for Y [1]:

$$(Y|S) \sim \mathcal{N}\left(k_X(X)K^{-1} \begin{bmatrix} Y^1 \\ \vdots \\ Y^N \end{bmatrix}, k(X,X) - k_X(X)K^{-1}k_X(X)^T \right) \tag{E.6}$$

As can be noticed, the prediction is not a value, but is a probability density function. Then, we can assume the mean of the above Gaussian (*i.e.* the value maximising the PDF) as a prediction, *i.e.*:

$$Y(X) \doteq g_{GP}(X) = k_X(X)K^{-1} \begin{bmatrix} Y^1 \\ \vdots \\ Y^N \end{bmatrix} \tag{E.7}$$

Notice that to evaluate the expression in equation (E.6), the inverse of K is required. This is not computationally demanding, since after training K is a constant, meaning that the computation of K^{-1} can be done once for all.

Training

Training has the aim of tuning the hyperparameters $\theta_{1,2,\ldots}$ characterizing the kernel function. The logarithmic likelihood of the model, see Appendix A appendix training, can be obtained considering the joint distribution of the samples in S, equation (E.3) [2]:

$$\begin{aligned} \mathcal{L} &= -\frac{N}{2} log\big(\,|K(\theta)|\,\big) - \frac{1}{2} Tr\left(K(\theta)^{-1} \begin{bmatrix} Y^1 \\ \vdots \\ Y^N \end{bmatrix} \begin{bmatrix} Y^1 & \cdots & Y^N \end{bmatrix} \right) + \cdots \\ &+ log\left((\mathbb{P}(\theta)_{prior} \right) \end{aligned} \tag{E.8}$$

[1] Here the expression of the conditional distribution of a multivariate Gaussian was exploited.
[2] Constant values are omitted

The maximization of $\mathcal{L}$ is typically done through gradient descend. Therefore, the gradient $\frac{\partial \mathcal{L}}{\partial \theta}$ must be evaluated [3]

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \theta_i} &= -\frac{N}{2}\frac{\partial}{\partial \theta_i}\Big(log\big(\,|K(\theta)|\,\big)\Big) - \frac{1}{2}\frac{\partial}{\partial \theta_i}\left(Tr\left(K(\theta)^{-1}\begin{bmatrix}Y^1\\ \vdots \\ Y^N\end{bmatrix}\begin{bmatrix}Y^1 & \cdots & Y^N\end{bmatrix}\right)\right) + \cdots \\
&+ \frac{\partial}{\partial \theta_i}\Big(\mathbb{P}(\theta_i)_{prior}\Big) \\
&= -\frac{N}{2}\frac{1}{|K(\theta)|}\frac{\partial}{\partial \theta_i}\Big(\,|K(\theta)|\,\Big) - \frac{1}{2}\left(\begin{bmatrix}Y^1\\ \vdots \\ Y^N\end{bmatrix}\begin{bmatrix}Y^1 & \cdots & Y^N\end{bmatrix}\right)^T\frac{\partial}{\partial \theta_i}\Big(K(\theta)^{-1}\Big) + \cdots \\
&+ \frac{\partial}{\partial \theta_i}\Big(\mathbb{P}(\theta_i)_{prior}\Big) \\
&= -\frac{N}{2}\frac{1}{|K(\theta)|}Tr\left(K(\theta)^{-1}\frac{\partial K(\theta)}{\partial \theta_i}\right) + \cdots \\
&+ \frac{1}{2}\left(\begin{bmatrix}Y^1\\ \vdots \\ Y^N\end{bmatrix}\begin{bmatrix}Y^1 & \cdots & Y^N\end{bmatrix}\right)^T K(\theta)^{-1}\frac{\partial K(\theta)}{\partial \theta_i}K(\theta)^{-1} + \frac{\partial}{\partial \theta_i}\Big(\mathbb{P}(\theta_i)_{prior}\Big) \qquad \text{(E.9)}
\end{aligned}$$

The expression of $\frac{\partial K(\theta)}{\partial \theta_i}$ depends on the kernel function adopted.

E.0.2 Vectorial case

Also vectorial functions can be approximated by Gaussian Processes. Suppose function g is defined as follows:

$$\begin{aligned} g &: \mathcal{X} \to \mathcal{Y} \\ \mathcal{X} \subseteq \mathbb{R}^n &\quad \mathcal{Y} \subseteq \mathbb{R}^m \end{aligned} \qquad \text{(E.10)}$$

The computations reported so far must be slightly modified for accounting the multidimensionality of $\mathcal{Y}$. Since g is vectorial, it's like having m distinct functions $g_{1,\cdots,m}$:

$$g(X) = \begin{bmatrix} g_1(X) \\ \vdots \\ g_m(X)\end{bmatrix} \qquad \text{(E.11)}$$

Therefore, for approximating g, m distinct Gaussian Processes are required. The learning of such battery of Gaussian Processes, must be done considering a training set S, made of samples $Y^{1,2,\cdots}$:

$$S = \langle \begin{bmatrix} X^1 \\ Y^1 = \begin{bmatrix} Y_1^1 & \cdots & Y_m^1\end{bmatrix}\end{bmatrix}, \cdots, \begin{bmatrix} X^N \\ Y^N\end{bmatrix}\rangle \qquad \text{(E.12)}$$

[3] The derivatives were computed considering what reported in [99].

The single function g_i models the joint density of $\begin{bmatrix} Y_1^1 \\ \vdots \\ Y_1^N \end{bmatrix}$. Therefore, the joint density of $Y^{1,\cdots,N}$ can be computed assuming m independent Gaussians:

$$\mathbb{P}\left(Y_1 = \begin{bmatrix} Y_1^1 \\ \vdots \\ Y_1^N \end{bmatrix} \right) \cdot \cdots \cdot \mathbb{P}\left(Y_m = \begin{bmatrix} Y_m^1 \\ \vdots \\ Y_m^N \end{bmatrix} \right) = \tag{E.13}$$

$$= \left(\frac{1}{\sqrt{(2\pi)^N \left|K\right|}} \right)^m \prod_{i=1}^{m} exp\left(-\frac{1}{2} Tr\left(K(\theta)^{-1} Y_i Y_i^T \right) \right) \tag{E.14}$$

$$= \frac{1}{\sqrt{(2\pi)^{Nm} \left|K\right|^m}} exp\left(-\frac{1}{2} \sum_{i=1}^{m} Tr\left(K(\theta)^{-1} Y_i Y_i^T \right) \right) \tag{E.15}$$

$$= \frac{1}{\sqrt{(2\pi)^{Nm} \left|K\right|^m}} exp\left(-\frac{1}{2} Tr\left(K(\theta)^{-1} \left[Y_1 | \cdots | Y_m \right] \begin{bmatrix} \overline{Y_1^T} \\ \vdots \\ \overline{Y_m^T} \end{bmatrix} \right) \right) \tag{E.16}$$

Prediction

m distinct scalar predictions are made for predicting $g(X)$, leading to:

$$(Y|S) \sim \begin{bmatrix} \mathcal{N}\left(k_X(X) K^{-1} Y_1, k(X,X) - k_X(X) K^{-1} k_X(X)^T \right) \\ \vdots \\ \mathcal{N}\left(k_X(X) K^{-1} Y_m, k(X,X) - k_X(X) K^{-1} k_X(X)^T \right) \end{bmatrix}^T \tag{E.17}$$

The value maximising the above conditioned probability is:

$$Y(X) \doteq g_{GP}(X) = k_X(X) K^{-1} \left[Y_1 | \cdots | Y_m \right] \tag{E.18}$$

Training

Training is done analogously to the scalar case, considering a likelihood function that takes into account the joint distribution in equation (E.16):

$$\mathcal{L} = -\frac{Nm}{2} log\big(\left|K(\theta)\right| \big) - \frac{1}{2} Tr\left(K(\theta)^{-1} \left[Y_1 | \cdots | Y_m \right] \begin{bmatrix} \overline{Y_1^T} \\ \vdots \\ \overline{Y_m^T} \end{bmatrix} \right) + \cdots$$

$$+ \; log\left((\mathbb{P}(\theta)_{prior} \right) \tag{E.19}$$