KPLS : from non-bayesian to bayesian

Thibault Lahire
Department of Mathematics
ENS Paris-Saclay / Master MVA
thibault.lahire@student.isae-supaero.fr

Samuel Gruffaz
Department of Mathematics
ENS Paris-Saclay / Master MVA
samuel.gruffaz@ens-paris-saclay.fr

Abstract

KPLS (for Kriging using Partial Least Squares) is a non-bayesian method proposed in 2015 by Bouhlel et al. [1] aiming at modeling black-box functions through Gaussian processes in an aeronautic framework. A key step in building a Gaussian process is the tuning of the mean and the kernel hyper-parameters. This project aims at exploring a bayesian way of tuning these hyper-parameters, instead of maximizing a likelihood or doing a cross-validation. The choice of a good prior is relevant for industrial purposes, particularly when the teams in charge of modeling the underlying black-box function deal with a parametric family of similar functions.

1 Introduction : setup and contribution

This study is driven by considerations inherent to the aeronautic framework, where optimization processes are a key step for the design of a new plane. Indeed, once the specifications for the new plane are clear, a cascade of optimizations are done to end up with a product satisfying all the constraints and the mission it is made for [2]. To give a simple example, one aims first at optimizing the global shape, then optimizing parts of the structure (wings, tail, landing gear...). The optimization processes go from global to local parts.

To give a concrete example of what is done in industry, one can stay to high level considerations. If a plane able to cross the Atlantic ocean, carry a fix number of passengers, land on international airports, etc..., is desired, the goal is then to design a plane satisfying all these constraints. Even better, one could for example aims at designing the plane satisfying all these constraints with the minimum mass. Hence an optimization problem under constraints appears, where the parameters to be optimized could be the size of the wings, the angle between the wings and the fuselage, etc...

To be more precise, the goal is to optimize $y : B \longrightarrow \mathbb{R}$, with $B \subset \mathbb{R}^d$, d of typical order 20, the number of design variables. For simplicity, we consider that $B = \prod_{j=1}^{d}[a_j,\ b_j]$, $a_j < b_j$. Each $\mathbf{x} \in B$ corresponds to a possible plane, and the aim is to find the best plane with respect to the criterion y and the constraints. Such an optimization is done thanks to an Efficient Global Optimization (EGO) strategy which deals with the trade-off exploration/exploitation with the help of some criteria, namely EI (Expected Improvement) or WB2 [3].

However, before optimizing y, the first difficulty is to know y. Indeed, since each $\mathbf{x} \in B$ is a plane, then evaluating y at point $\mathbf{x}$ is very costly. It could mean, for example, computing fluid-structure interactions to evaluate the lift or the drag, which involves heavy FEM (Finite Element Methods) or

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

CFD (Computational Fluid Dynamics) simulations. There is no hope for an analytic form of y, and no more for a precise knowledge of y. We are forced to find the minimum of y by knowing it on a few points only. Moreover, in such conditions, y is highly non-convex and may contain multiple local optima.

This paper focuses on modeling y with the higher accuracy given the knowledge of y on n points : $\{(\mathbf{x}^{(i)}, y^{(i)}),\ i = 1, ..., n\}$, where $y^{(i)} = y(\mathbf{x}^{(i)})$. This regression problem can be easily framed with the use of Gaussian processes, or Kriging methods,[1] since the number of points n is *small*. Indeed, such a model requires the inversion of a $n \times n$ matrix, which is suitable for *small* n. Moreover, since y is the result of a computer simulation, we know exactly y where it has been evaluated. No noise on the measures has to be added.

This work takes the reference paper [1] as a frequentist baseline, and tries to transform as much as possible the frequentist treatment of the problems encountered in a Bayesian one. Indeed, it is shown experimentally that the choice of a good prior results in a better optimization of hyper-parameters. From an industrial point of view, there is one main advantage to a Bayesian approach. As written above, modeling black-box functions intervene at different steps of the design process, going from a global shape of the plane to precise parts of it. From a set of specifications to another, the functions to model are not identical, but they are *not far* from each other. In other words, there is a sort of *similarity* between them. Indeed, a plane is always made of a fuselage, one right wing and one left wing, a vertical tail, the whole is perfectly symmetric, etc... All this leads us to think that, once a good prior on the hyper-parameters is chosen for a given type of specifications, this prior will be relevant for other types of specifications, due to the invariances written previously.

In section 2, we introduce the Gaussian process model. Even if the main paper [1] does not tackle the issue of optimizing y, we briefly propose some considerations on bayesian optimization. Then, a bayesian manner to adress the issues originally solved in a frequentist way is presented in section 3. The links between the two paradigms are studied. Once the Gaussian process approximating well y is known, the Partial Least Squares (PLS) method is introduced in section 4 to efficiently reduce the dimension of the hyper-parameters. We present how to make use of PLS in a Bayesian framework. Finally, in section 5, the numerical simulations compare the frequentist and the Bayesian approaches on an analytical function.

2 Gaussian processes and Bayesian optimization

Given a training set, the goal is to estimate plausible values of the function y where it has not been evaluated. Once y is estimated, the next point of the optimization process can be selected smartly. This chapter is then divided into two parts : estimating the value of y *everywhere* and selecting the point x_{new} for the optimization process.

2.1 Gaussian processes for estimating the value of $y(\mathbf{x})$

The n points already explored (training set) are stored in the matrix $\mathbf{X} = [\mathbf{x}^{(1)\top}, ..., \mathbf{x}^{(n)\top}]^\top$, and the outputs are stored in the vector $\mathbf{y} = [y^{(1)}, ..., y^{(n)}]^\top$ with $y^{(i)} = y(\mathbf{x}^{(i)})$. We use $\hat{y}(\mathbf{x})$ to denote the prediction of the true function $y(\mathbf{x})$ which is considered as a realization of a stochastic process $Y(\mathbf{x})$ for all $\mathbf{x} \in B$. We consider that we want to know y on n_* points (test set). We store the n_* points of the set in the matrix $\mathbf{X}_*$, just as we did for the training points.

Definition A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution. A Gaussian process is completely specified by its mean function and covariance function, also called kernel.

An isotropic kernel In the reference paper, the authors use an isotropic Gaussian exponential covariance kernel:

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \prod_{i=1}^{d} \exp\left\{-\theta_i (\mathbf{x}_i - \mathbf{x}'_i)^2\right\} \quad \forall\, \theta_i \in \mathbb{R}_+ \tag{1}$$

[1] The phrases *Gaussian processes* and *Kriging method* are seen as equivalent. However, historically, Kriging denoted a Gaussian process in small dimension, typically 2 or 3, by the time Gaussian processes were used to model geological structures.

This choice is common in the literature. Indeed, if the function to be optimized is not isotropic (which is often the case in practice), a mixture of experts (MOE) can be performed for a better analysis [4]. σ^2 and θ are inferred from the data.

Moreover, we would like to have a model which is not zero-mean, for realistic reasons. Using a fixed (deterministic) mean is trivial, but in practice it is often difficult to specify a fixed mean function. In many cases it may be more convenient to specify a basis of functions, whose coefficients, $\boldsymbol{\beta}$, are to be inferred from the data. We note, for $j = 1, ..., m$, f_j the j-th basis function. We introduce the $m \times 1$ vector $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), ..., f_m(\mathbf{x})]^\top$ and the $n \times m$ matrix $\mathbf{F} = \left[\mathbf{f}(\mathbf{x}^{(1)}), ..., \mathbf{f}(\mathbf{x}^{(n)})\right]^\top$. We introduce also $\mathbf{F}_*$ the $n_* \times m$ matrix for the test set.

The correlation between the point $\mathbf{x}$ and the point $\mathbf{x}'$ is written $r_{\mathbf{x}\mathbf{x}'}$. The correlations are the key ingredient to the kernel, we indeed have $k(\mathbf{x}, \mathbf{x}') = \sigma^2 r_{\mathbf{x}\mathbf{x}'}$.

We introduce the $n \times 1$ vector $\mathbf{r}_{\mathbf{x}\mathbf{X}} = [r_{\mathbf{x}\mathbf{x}^{(1)}}, ..., r_{\mathbf{x}\mathbf{x}^{(n)}}]^\top$ and the $n \times n$ covariance matrix $\mathbf{R} = [\mathbf{r}_{\mathbf{x}^{(1)}\mathbf{X}}, ..., \mathbf{r}_{\mathbf{x}^{(n)}\mathbf{X}}]$. Our goal is to infer plausible values of y on a test set composed of n_* points. Hence we introduce the $n \times n_*$ matrix $\mathbf{R}_*$ where the coefficient (i, j) is the value of the correlation function between the i-th point of the training set and the j-th point of the test set. We introduce also the $n_* \times n_*$ matrix $\mathbf{R}_{**}$ where the coefficient (i, j) is the value of the correlation function between the i-th point and the j-th point of the test set.

When there is only one function f_j equals to one everywhere, the model is called "Ordinary Kriging". The linear or quadratic functions are often enough to obtain good models. We could also use the basis $f_j : x \longmapsto \cos(jx)$, which is much more expressive than linear or quadratic functions. However, for computational reasons, we cannot have an infinite number of functions f_j, so, in this case, we have to choose m such that we end up with a good model.

To sum up, the hyper-parameters we have are : m, the number of basis functions to choose, $\boldsymbol{\beta}$, which characterizes the mean at each location $\mathbf{x} \in B$, σ^2 the noise intensity of the process, and θ the length scale of the kernel, characterizing the regularity of the process.

By definition of a Gaussian process, the joint distribution of the training outputs, $\mathbf{y}$, and the test outputs $\mathbf{y}_*$ is :

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{y}_* \end{bmatrix} \mid \mathbf{X}_*, \mathbf{X}, m, \boldsymbol{\beta}, \sigma^2, \theta \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{F}\boldsymbol{\beta} \\ \mathbf{F}_*\boldsymbol{\beta} \end{bmatrix}, \ \sigma^2 \begin{bmatrix} \mathbf{R} & \mathbf{R}_* \\ {\mathbf{R}_*}^\top & \mathbf{R}_{**} \end{bmatrix} \right) \tag{2}$$

Thanks to this Gaussian model, it is very simple to extract the law of $\mathbf{y}_*$ given the sets and all the hyper-parameters. We have :

$$\mathbf{y}_* | \mathbf{X}_*, \mathbf{X}, \mathbf{y}, m, \boldsymbol{\beta}, \sigma^2, \theta \ \sim \ \mathcal{N}\left(\mathbf{F}_*\boldsymbol{\beta} + {\mathbf{R}_*}^\top \mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta}), \ \sigma^2(\mathbf{R}_{**} - {\mathbf{R}_*}^\top \mathbf{R}^{-1}\mathbf{R}_*) \right) \tag{3}$$

2.2 Selecting the next point $\mathbf{x}_{new}$ thanks to an acquisition function

The law of $\mathbf{y}_*$ in (3) is useful to perform predictions on $y(\mathbf{x})$. It is possible to compute two quantities of interest: the mean prediction and the variance prediction.

Given $m, \boldsymbol{\beta}, \sigma^2, \theta$, to obtain the mean prediction presented in the paper [1], we just have to rewrite the prediction of eq. (3) for a particular point $\mathbf{x}$ of the test set :

$$\hat{y}(\mathbf{x}) = \mathbf{f}(\mathbf{x})^\top \boldsymbol{\beta} + {\mathbf{r}_{\mathbf{x}\mathbf{X}}}^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\boldsymbol{\beta}\right) \tag{4}$$

To get the variance prediction, we compute the variance of eq (3) applied to a single point $\mathbf{x}$:

$$s(\mathbf{x}) = \sigma^2 \left(1 - {\mathbf{r}_{\mathbf{x}\mathbf{X}}}^\top \mathbf{R}^{-1} \mathbf{r}_{\mathbf{x}\mathbf{X}}\right) \tag{5}$$

Note the direct influence of θ and σ^2 on the variance prediction. We will see that $\boldsymbol{\beta}$ is important as well because it is present in the estimation of θ and σ^2.

Then, thanks to these two quantities we want to define an acquisition function $a(\mathbf{x}; \{\mathbf{X}, \mathbf{y}\}, \boldsymbol{\beta}, \sigma^2, \theta, m)$ which has an expression depending of the parameters and the training set. This function determines the next point where y should be evaluated:

$$\mathbf{x}_{new} = \text{argmin}_{\mathbf{x}} \, a(\mathbf{x}; \{\mathbf{X}, \mathbf{y}\}, \boldsymbol{\beta}, \sigma^2, \theta, m)$$

The dynamic of Bayesian optimization can be seen here : computation of the activation function by estimating parameters thanks to the training set, selection of $\mathbf{x}_{new}$ by minimizing a, computation of $y(\mathbf{x}_{new})$, addition of $(\mathbf{x}_{new}, y_{new})$ to the training set, computation of the activation function and so on...

There is a wide variety of activation functions and we refer to [8] for further details. Nevertheless, we give an example for a better understanding:

$$a_{LCB}(\mathbf{x}; \{\mathbf{X}, \mathbf{y}\}, \boldsymbol{\beta}, \sigma^2, \theta, m) = \hat{y}(\mathbf{x}) - \kappa s(\mathbf{x})$$

with a tunable κ for balance exploration, (making use of $s(\mathbf{x})$) against exploitation (making use of $\hat{y}(\mathbf{x})$). In that way, underestimating the variance and underestimating the mean will emphasis exploitation at the detriment of exploration. Therefore, it is crucial not to fall in overfitting, otherwise it will give no sense to the Bayesian optimization which should give a smart way of performing exploitation. That's why an Integral Acquisition Function (IAF) was presented in [5] to win in generalization and to adopt a Bayesian approach with a marginalisation on the hyper-parameter θ:

$$\hat{a}(\mathbf{x}; \{\mathbf{X}, \mathbf{y}\}, \boldsymbol{\beta}, \sigma^2, m) = \int a(\mathbf{x}; \{\mathbf{X}, \mathbf{y}\}, \theta, \boldsymbol{\beta}, \sigma^2, m) p(\theta | \mathbf{y}, \mathbf{X}, \hat{\sigma}^2, \hat{\beta}, m) \, \mathrm{d}\, \theta$$

In this paper, a practical way of computing the IAF will be given by making use of the PLS method. However, first, we present how the hyper-parameters $\sigma^2, \beta, \theta, m$ can be estimated by being frequentist (as done in the reference paper [1]) or bayesian. For the following, m is assumed to be fixed for sake of simplicity except if the context implies clearly a random m.

3 From frequentist to Bayesian

The main goal of the articles [1],[6] is to find an efficient way of selecting the hyper-parameters, and in particular the hyper-parameter $\theta \in \Theta \subset \mathbb{R}^d$ which parameterizes the correlation matrix $\mathbf{R} = \mathbf{R}(\theta)$ through the expression of the kernel k. In the proposed approach, everything boils down to optimizing $\log p(\mathbf{y}|\mathbf{X}, m, \boldsymbol{\beta}, \sigma^2, \theta)$ sequentially with respect to the hyper-parameters. The main paper [1] doesn't tackle the regression problem with a bayesian point of view explicitly : all the hyper-parameters are found by computing MLEs (Maximum Likelihood Estimators). In this section, we propose a bayesian approach to handle the problem and we expose how frequentist and bayesian are linked in the different estimations.

First, we need to take a step back to understand why the bayesian approach is legitimate and how we can make the best of it. We could see the GP as a prior on a function space because we see the function we want to recover as a parameter of the conditional law $\mathbf{y}|\mathbf{X}$. The kernel gives the regularity of the function space and the mean function centers the prior on the data. Nevertheless, it seems unfair to call the GP "prior" because the posterior is not computed, we just derive the likelihood thanks to the formula of Gaussian conditioning. Moreover, when it comes to Bayesian approach, it asks to set a prior and a loss to make decisions about selecting kernel's parameters. So we have to pick a loss $\mathcal{L}$ and a hyper-prior for each hyper-parameter. To do so, we have to figure out what we penalize and what are the prior information on our hyper-parameters. Here we want to make predictions on a unknown function to optimize, but in our industrial case, we have some information on its regularity and possible value thanks to previous simulations on similar functions, that's what legitimates the use of priors in our model.

We want to penalize predictions which are too far from the true output without overfitting. But we should keep in mind that the choice of the prior and the loss have to enable practical computations as well, that's why we are sometimes forced to use the MAP estimator which is not a Bayesian estimator in the presented framework, unlike the MP estimator related to the L^2 loss. Secondly, we have to weight up pros and cons in each situation in order to take practical decisions.

3.1 Prior on the mean of the GP

To begin, the basis functions are considered fixed (take for example the linear or quadratic approximations), which is equivalent to set m to a certain value. We are interested in demonstrating how the expression of $\hat{\boldsymbol{\beta}}$ of the original paper has been found. We first present the frequentist method, then we will show that this method can be seen as the limit of a Bayesian technique where the prior becomes non-informative.

Frequentist $\boldsymbol{\beta}$ estimation We simply maximize the likelihood $p(\mathbf{y}|\mathbf{X}, m, \boldsymbol{\beta}, \sigma^2, \theta)$ with respect to $\boldsymbol{\beta}$:

$$\log p(\mathbf{y}|\mathbf{X}, m, \boldsymbol{\beta}, \sigma^2, \theta) = -\log\left(\sqrt{2\pi}^n \sigma^n |\mathbf{R}|\right) - \frac{1}{2\sigma^2}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)$$

We note $g(\boldsymbol{\beta})$ the quantity of interest :

$$g(\boldsymbol{\beta}) = -\frac{1}{2\sigma^2}\left(\boldsymbol{\beta}^\top \mathbf{F}^\top \mathbf{R}^{-1}\mathbf{F}\boldsymbol{\beta} - 2\boldsymbol{\beta}^\top \mathbf{F}^\top \mathbf{R}^{-1}\mathbf{y}\right)$$

We then set the gradient of g with respect to $\boldsymbol{\beta}$ to zero to obtain the MLE:

$$0 = \nabla g(\boldsymbol{\beta}) = -\frac{1}{\sigma^2}(\mathbf{F}^\top \mathbf{R}^{-1}\mathbf{y} - \mathbf{F}^\top \mathbf{R}^{-1}\mathbf{F}\boldsymbol{\beta}) \text{ , and also } \nabla^2 g(\boldsymbol{\beta}) = -\mathbf{F}^\top \sigma^{-2}\mathbf{R}^{-1}\mathbf{F}$$

Note that there is no obvious reason that the Hessian matrix is strictly-non-positive and definite to ensure that 1) it is non-singular and 2) the critical point found is indeed a maximum. Assuming that the basis functions have been chosen such that the Hessian is non-singular, we have :

$$\hat{\boldsymbol{\beta}}_{MLE} = (\mathbf{F}^\top \mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{F}^\top \mathbf{R}^{-1}\mathbf{y} \tag{6}$$

which is what has been obtained in the main paper.

Bayesian $\boldsymbol{\beta}$ estimation As explained in introduction, one may be interested in using a prior on $\boldsymbol{\beta}$. This prior enables us to assume some regularities of the mean function. Indeed, for instance, if we take $(f_j)_{j\leq m} = (x \to \cos(ix))$ and we know that the true function has low frequencies, we can set $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{b}, \boldsymbol{\Gamma})$ with $\boldsymbol{\Gamma} = \mathrm{diag}(C, C/2, \ldots, C/m)$.

In some cases, we don't know how to select m effectively so we could add a layer to the Bayesian estimation by setting a Poisson prior on $m \sim \mathcal{P}(m^*)$ or even use Bayesian non parametric models such as Dirichlet process.

For sake of simplicity, we begin by taking the prior on $\boldsymbol{\beta}$ to be Gaussian: $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{b}, \boldsymbol{\Gamma})$, and we set m to a certain value. We want to estimate $\hat{\boldsymbol{\beta}}$ by taking the Bayesian estimator related to the loss $L(\boldsymbol{\beta}, \boldsymbol{\beta}^*) = ||\mathbf{F}\boldsymbol{\beta} - \mathbf{F}\boldsymbol{\beta}^*||_2^2$ for penalizing differences between the mean functions:

$$\begin{aligned}\hat{\boldsymbol{\beta}} &= \mathrm{argmin}_{\hat{\boldsymbol{\beta}}(\mathbf{y})}\, \mathbb{E}_{\mathbf{y}|\mathbf{X},\sigma^2,\theta}(L(\hat{\boldsymbol{\beta}}(y), \boldsymbol{\beta}^*)) \\ &= \mathrm{argmin}_{\hat{\boldsymbol{\beta}}(\mathbf{y})} \int\int L(\hat{\boldsymbol{\beta}}(y), \beta) p(\beta|\mathbf{y}, \mathbf{X}, \sigma^2, \theta)\,\mathrm{d}\,\beta\; p(\mathbf{y}|\mathbf{X}, \sigma^2, \theta)\,\mathrm{d}\,\mathbf{y}\end{aligned}$$

For each $\mathbf{y}$ we get $\hat{\boldsymbol{\beta}}(\mathbf{y}) = \mathrm{argmin}_\delta \int L(\delta, \beta) p(\beta|\mathbf{y}, \mathbf{X}, \sigma^2, \theta)\,\mathrm{d}\,\beta$ By seeing that $h : \beta \to \mathbf{F}\beta$ is linear, it is equivalent to minimise by using the L^2 loss:

$$\int ||h(\delta - \beta)||_2^2 p(\beta|\mathbf{y}, \mathbf{X}, \sigma^2, \theta)\,\mathrm{d}\,\beta = ||h(\delta - \hat{\boldsymbol{\beta}}_{MP})||_2^2 + \int ||h(\hat{\boldsymbol{\beta}}_{MP} - \beta)||_2^2 p(\beta|\mathbf{y}, \mathbf{X}, \sigma^2, \theta)\,\mathrm{d}\,\beta$$
$$+2\,\underbrace{h\left(\hat{\boldsymbol{\beta}}_{MP} - \int \beta p(\beta|\mathbf{y}, \mathbf{X}, \sigma^2, \theta)\,\mathrm{d}\,\beta\right) h(\delta - \hat{\boldsymbol{\beta}}_{MP})}_{=0}$$

So the minimum is $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}}_{MP} = \int \beta p(\beta|\mathbf{y}, \mathbf{X}, \sigma^2, \theta)\,\mathrm{d}\,\beta$. To derive it, we compute the posterior law of β:

$$p(\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}, \theta, \sigma^2) \propto p(\boldsymbol{\beta})p(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}, \theta, \sigma^2) = \mathcal{N}(\boldsymbol{\beta}; \mathbf{b}, \boldsymbol{\Gamma})\mathcal{N}(\mathbf{y}; \mathbf{F}^\top\boldsymbol{\beta}, \sigma^2\mathbf{R})$$
$$-2\log(p(\mathbf{y}|\mathbf{X}, \boldsymbol{\beta}, \sigma^2, \theta)p(\boldsymbol{\beta})) \propto \boldsymbol{\beta}^\top\boldsymbol{\Gamma}^{-1}\boldsymbol{\beta} + \sigma^{-2}\boldsymbol{\beta}^\top\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F}\boldsymbol{\beta} - 2(\boldsymbol{\Gamma}^{-1}\mathbf{b} + \sigma^{-2}\mathbf{y}^\top\mathbf{R}^{-1}\mathbf{F})\boldsymbol{\beta}$$

By identifying coefficients of the second order polynomial of $\boldsymbol{\beta}$, we find that $\boldsymbol{\beta}|\mathbf{y}, \mathbf{X}, \sigma^2, \theta \sim \mathcal{N}(\hat{\boldsymbol{\beta}}, \Sigma)$ where:

$$\Sigma^{-1} = \sigma^{-2}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F} + \boldsymbol{\Gamma}^{-1}, \text{ and } \hat{\boldsymbol{\beta}} = (\sigma^{-2}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F} + \boldsymbol{\Gamma}^{-1})^{-1}(\boldsymbol{\Gamma}^{-1}\mathbf{b} + \sigma^{-2}\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{y}) \tag{7}$$

Note the nice interpretation of the mean expression, eq. (7): $\hat{\boldsymbol{\beta}}$ is the mean of the global linear model parameters, being a compromise between the data term and prior. The covariance is the harmonic sum of the prior covariance and the covariance due to the kernel. This approach gives a variance for $\boldsymbol{\beta}$ and uses our prior belief to estimate correctly the mean function. Furthermore, we used the MP estimator in the computations herein above. Since the variables are gaussian, a MAP estimation would have given the same result.

The frequentist estimation seen as a limit We now present the results of the main paper by exploring the limit of the above expressions as the prior on the $\boldsymbol{\beta}$ parameter becomes vague, i.e. $\boldsymbol{\Gamma}^{-1} \longrightarrow O$ (where O is the null matrix). We obtain a predictive distribution which is independent of $\mathbf{b}$ conformly to the intuition. Taking the null limit is like taking a uniform prior on $\boldsymbol{\beta}$ living in $\mathbb{R}^m$. We recover the MLE: $\hat{\boldsymbol{\beta}} = (\mathbf{F}^\top \mathbf{R}^{-1} \mathbf{F})^{-1} (\mathbf{F}^\top \mathbf{R}^{-1} \mathbf{y})$.

Taking the MLE is quite simple and unbiased, computations are not so expensive compared to the Bayesian approach which require further matrix inversion. Moreover, we got rid of σ^2 in the $\boldsymbol{\beta}$ estimation through MLE. Nevertheless, there is a bias which comes from the choice of the basis functions and the lack of further information could make our model overfit the data.

An adaptive estimator for $\boldsymbol{\beta}$ As mentioned before, sometimes we might encounter a tricky situation where the mean function is quite sharp and our model is too restricted by the complexity of the basis functions to fit correctly the data. So, our model should be automatically adapted to the situation, by not giving too much complexity when it is unnecessary and by allowing the basis functions to capture the sharpness of the mean function when we have to.

A natural choice is to take m random. By taking a Poisson prior $m \sim \mathcal{P}(m^*)$, it will enable us to perform model selection by penalizing high complexity. Of course, we could simply be bayesian with respect to m, and then take the MLE for $\boldsymbol{\beta}$. But we can also keep a gaussian prior at the same time we take a prior on m. We take $\boldsymbol{\beta}|m \sim \mathcal{N}(\mathbf{b}_m, \boldsymbol{\Gamma}_m)$ and we select $\hat{m}$ by taking the MAP estimator. We then write : $p(m|\mathbf{y}, \mathbf{X}, \sigma^2, \theta) = p(\mathbf{y}|\mathbf{X}, m, \sigma^2, \theta) p(m)$, which means we need the law of $\mathbf{y}|\mathbf{X}, m, \sigma^2, \theta$.

Denoting $\mathbf{J}$ this random variable of interest, it is important to remark that $\mathbf{J}$ is gaussian, but its mean and variance must incorporate the fact that $\boldsymbol{\beta}$ is not fixed. We have $\mathbf{J} = \mathbf{Z} + \mathbf{F}_m \boldsymbol{\beta}$, with $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{b}_m, \boldsymbol{\Gamma}_m)$ as before and $\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{R})$. We then have :

$$
\begin{aligned}
\mathbb{E}[\mathbf{J}] &= \mathbb{E}[\mathbf{Z}] + \mathbb{E}[\mathbf{F}_m \boldsymbol{\beta}] = \mathbf{F}_m \mathbf{b}_m \\
\mathbb{V}ar[\mathbf{J}] &= \mathbb{V}ar[\mathbf{Z}] + \mathbb{V}ar[\mathbf{F}_m \boldsymbol{\beta}] = \sigma^2 \mathbf{R} + \mathbb{E}[(\mathbf{F}_m \boldsymbol{\beta} - \mathbf{F}_m \mathbf{b}_m)(\mathbf{F}_m \boldsymbol{\beta} - \mathbf{F}_m \mathbf{b}_m)^\top] \\
\mathbb{V}ar[\mathbf{J}] &= \sigma^2 \mathbf{R} + \mathbf{F}_m \mathbb{E}[(\boldsymbol{\beta} - \mathbf{b}_m)(\boldsymbol{\beta} - \mathbf{b}_m)^\top] \mathbf{F}_m^\top \\
\mathbb{V}ar[\mathbf{J}] &= \sigma^2 \mathbf{R} + \mathbf{F}_m \boldsymbol{\Gamma}_m \mathbf{F}_m^\top
\end{aligned}
$$

Which implies that: $\mathbf{y}|\mathbf{X}, m, \sigma^2, \theta \sim \mathcal{N}(\mathbf{F}_m \mathbf{b}_m,\ \mathbf{R} + \mathbf{F}_m \boldsymbol{\Gamma}_m \mathbf{F}_m^\top)$. We obtain:

$$
p(m|\mathbf{y}, \mathbf{X}, \sigma^2, \theta) = \underbrace{\frac{(m^*)^m e^{-m^*}}{m!}}_{\text{prior penalty}} \underbrace{\mathcal{N}(\mathbf{y}; \mathbf{F}_m^\top \mathbf{b}_m, \mathbf{R} + \mathbf{F}_m^\top \boldsymbol{\Gamma}_m \mathbf{F}_m)}_{\text{conditional likelihood}}
$$

Therefore, $\hat{m} = \operatorname{argmax}_m p(m|\mathbf{y}, \mathbf{X}, \sigma^2, \theta)$ and $\hat{\boldsymbol{\beta}} = (\sigma^{-2} \mathbf{F}_{\hat{m}}^\top \mathbf{R}^{-1} \mathbf{F}_{\hat{m}} + \boldsymbol{\Gamma}_{\hat{m}}^{-1})^{-1} (\boldsymbol{\Gamma}_{\hat{m}}^{-1} \mathbf{b}_{\hat{m}} + \sigma^{-2} \mathbf{F}_{\hat{m}}^\top \mathbf{R}^{-1} \mathbf{y})$. This model selection is incredibly rich, m^* can be chosen according to precedent simulations and the mean $\mathbf{b}_m$ as well. The covariance matrix $\boldsymbol{\Gamma}_m$ requires a specific attention because it weights up each function of the basis, for example taking decreasing coefficients on the diagonal is useless because high dimensions are already penalized by the penalty term in the estimation of $\hat{m}$. All these possibilities make our predictions more accurate. The other side of the coin is that the computation of conditional likelihood requires new matrix inversions and the $\boldsymbol{\beta}$ estimation as well. In addition, σ^2 has to be estimated before.

3.2 Prior on the noise intensity

The frequentist estimator for noise intensity Note that σ^2, the noise intensity of the process, is usually found by maximizing the likelihood $p(\mathbf{y}|\mathbf{X}, \hat{\boldsymbol{\beta}}, \sigma^2, \theta)$. We now detail the computations of this maximization, and then we propose, unsurprisingly, a Bayesian way to obtain an estimator of σ^2.

$$
p(\mathbf{y}|\mathbf{X}, \hat{\boldsymbol{\beta}}, \sigma^2, \theta) = \frac{1}{(2\pi)^{n/2} \sigma^n |\mathbf{R}|^{1/2}} \exp\left\{ -\frac{1}{2\sigma^2} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) \right\}
$$

$$
\log p(\mathbf{y}|\mathbf{X}, \hat{\boldsymbol{\beta}}, \sigma^2, \theta) = -\frac{n}{2} \log(2\pi) - n \log(\sigma) - \frac{1}{2} \log |\mathbf{R}| - \frac{1}{2\sigma^2} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)
$$

We derive this expression with respect to σ and equals it to zero to find the MLE:

$$0 = \frac{d \log p(y|X, \hat{\boldsymbol{\beta}}, \theta, \sigma)}{d\sigma} = -\frac{n}{\sigma} + \frac{1}{\sigma^3} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)$$

We finally obtain:

$$\hat{\sigma}^2 = \frac{1}{n} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) \tag{8}$$

Bayesian manner for the noise intensity As we wrote previously, we have some experience about the tuning of each hyper-parameter. In this way, we set a prior $\sigma^2 \sim INV\Gamma(\alpha, \delta)$. We recall the density of the law $INV\Gamma(\alpha, \delta)$:

$$p(\sigma^2) = f_{\alpha,\delta}(x) = \frac{\delta^\alpha}{\Gamma(\alpha)} \frac{1}{x^{\alpha+1}} \exp(-\delta/x)$$

This prior is conjugate to the posterior in our model. Then, we can estimate σ^2 by taking the MAP or the MP estimator $\hat{\sigma}^2_{MAP} = \text{argmax}_{\sigma^2}\, p(\sigma^2|\mathbf{y}, \mathbf{X}, \hat{\boldsymbol{\beta}}, \theta))$ or $\hat{\sigma}^2_{MP} = \mathbb{E}_{\sigma^2|\mathbf{y},\mathbf{X},\hat{\boldsymbol{\beta}},\theta}(\sigma^2)$.

$$\begin{aligned}
\log(p(\sigma^2|\mathbf{y}, \mathbf{X}, \hat{\boldsymbol{\beta}}, \theta)) &= \log(p(\mathbf{y}|\mathbf{X}, \hat{\boldsymbol{\beta}}, \sigma^2, \theta)) + \log(p(\sigma)) \\
&\propto -\frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) - \log(p(\sigma^2)) \\
&\propto -\left(\frac{n}{2} + \alpha + 1\right)\log(\sigma^2) - \sigma^{-2}\left(\frac{1}{2}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) + \delta\right)
\end{aligned}$$

Therefore $\sigma^2|\mathbf{y}, \mathbf{X}, \hat{\boldsymbol{\beta}}, \theta \sim INV\Gamma\left(\frac{n}{2} + \alpha, \frac{1}{2}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) + \delta\right)$. We recover the mean and the mode:

$$\hat{\sigma}^2_{MP} = \frac{2\delta + \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)}{2(\alpha - 1) + n} \quad \text{and} \quad \hat{\sigma}^2_{MAP} = \frac{2\delta + \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)}{2(\alpha + 1) + n}$$

Note that when $(\alpha, \delta) = (1, 0)$, we recover the MLE and that in this case, we should be subjective Bayesian. The right choice of the estimator depends on the context and our goals, whether we want our noise intensity to be overestimated, we take the MP estimator, the risk is to overfit our data ; or we want our noise intensity to be underestimated, we take the MAP estimator, the risk is to underfit our data. There were no such decisions to take with the $\hat{\beta}$ estimator because the MP and MAP estimator were equal.

3.3 Prior on the kernel of the GP

So far we have discussed the frequentist and the bayesian ways to select the best values of the hyper-parameters m, $\boldsymbol{\beta}$, and σ^2. We now tackle the most important hyper-parameter : the length-scale θ.

The goal is to tune the (θ_i) in order to predict the variability in each direction. Sadly, the more the variables, the more computationally expensive it is. Thus, we will try to reduce the parameters of this kernel. By applying some elementary operations to existing kernels, we can construct new kernels. In the main paper [1],[6], the authors use the property that the tensor product of covariances is a covariance kernel in the product space to apply in a clever way the PLS method. The Partial Least Squares (PLS) is used to obtain a new set of parameters smaller than the initial one, without doing too much approximations. Before detailing the PLS method, we expose the frequentist and bayesian general solution to tackle the issue of estimating θ.

frequentist approach The frequentist solution consists in maximizing the likelihood $p(\mathbf{y}|\mathbf{X}, \hat{\boldsymbol{\beta}}, \sigma^2, \theta)$ that we have already met with respect to θ. However, this time, we consider σ as given by its MLE, MP or MAP estimation $\hat{\sigma}$. We then want to maximize:

$$\text{log-ML}(\theta) = -\frac{1}{2}\left[n\log(2\pi\hat{\sigma}^2) + \log(|\mathbf{R}(\theta)|) + \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}(\theta)^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)/\hat{\sigma}^2\right] \tag{9}$$

Maximizing the equation above is computationally expensive for high dimensions and when using a large number of sample points because the $n \times n$ matrix $\mathbf{R}(\theta)$ must be inverted. This problem is often solved by using genetic algorithms (see [7] for more details). In the main paper, the authors use the derivative-free optimization algorithm COBYLA that was developed in [8]. COBYLA is a sequential trust- region algorithm that uses linear approximations for the objective and constraint functions. Note that there is no possible simplification in this formula, the influence of θ is highly non linear. That's why we need to add further information.

Bayesian approach If we take a prior on θ, let's see the shape of the posterior:

$$\log(p(\theta|\mathbf{y},\mathbf{X},\hat{\boldsymbol{\beta}},\hat{\sigma}^2)) \propto_+ -\frac{1}{2}\left(\log|\mathbf{R}(\theta)| + \left(\mathbf{y}-\mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}(\theta)^{-1}\left(\mathbf{y}-\mathbf{F}\hat{\boldsymbol{\beta}}\right)/\hat{\sigma}^2 + n\log(\hat{\sigma^2})\right) + \log p(\theta)$$

Here there is no clear way to choose a prior on θ to simplify computation, because the non-linearity of the model makes it impossible to find any conjugation law. We can't expect to compute easily any integrals with respect to the posterior law due to the sharpness of the formula in θ We can only hope to add a clever regularisation effect in the optimisation.

An unclear estimation To compute log-ML(θ) we have to know $\hat{\sigma}^2$ and $\hat{\boldsymbol{\beta}}$, but these latter can be estimated provided that θ is known as well, and the bayesian estimators $\hat{\sigma}^2_{MAP}, \hat{\boldsymbol{\beta}}_{MAP}$ depends mutually to each other. There is no clear way to estimate all these estimators. Nevertheless, we can follow this sequence to get a reasonable approximation of log-ML(θ):

- Pick θ
- Compute the frequentist estimator of $\boldsymbol{\beta}$. $\hat{\boldsymbol{\beta}} = (\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{F})^{-1}(\mathbf{F}^\top\mathbf{R}^{-1}\mathbf{y})$.
- Thanks to $\hat{\boldsymbol{\beta}}, \theta$, derive $\hat{\sigma}^2$ in a frequentist or bayesian way.
- (Optional) Compute a bayesian estimator of $\boldsymbol{\beta}$ thanks to $\hat{\sigma}^2, \theta$.
- Compute log-ML(θ).

4 Partial Least Squares

PLS in [1] Estimating the kriging parameters can be time consuming, especially with dimensions larger than 100. Indeed, for such a case, we have to optimize the function log-ML defined in eq. (9), which has in input dim(θ) variables. However, solving this problem can be accelerated by combing the PLS method and the kriging model.

The θ parameters from the kriging model represent the range in any spatial direction. Assuming, for instance, that certain values are less significant for the response, then the corresponding θ_i $(i = 1, ..., d)$ will be very small compared to the other θ parameters. The PLS method is a well-known tool for high-dimensional problems and consists of maximizing the variance by projecting onto smaller dimensions while monitoring the correlation between input variables and the output variable. In this way, the PLS method reveals the contribution of all variables—the idea being to use this information to scale the θ parameters.

The authors of the main paper propose a new method that can be used to build an efficient kriging model by using the information extracted from the PLS stage. The main steps for this construction are as follows:

1) Use PLS to define weight parameters. 2) To reduce the number of hyper-parameters, define a new covariance kernel by using the PLS weights. 3) Optimize the parameters.

The mathematical details of such a construction are detailed in the main paper. In this work, we do not discuss the way PLS is done. However, as well as there is a Bayesian version of PCA (Principal Component Analysis), there exists a Bayesian version of PLS [9]. After PLS, we know the directions of interest, and we are able to construct a new kernel with hyper-parameters θ of dimension $h \ll d$. We have:

$$k(\mathbf{x},\mathbf{x}') = \sigma^2 \prod_{l=1}^{h}\prod_{i=1}^{d} \exp\left\{-\theta_l\left(\mathbf{w}_{*i}^{(l)}\mathbf{x}_i - \mathbf{w}_{*i}^{(l)}\mathbf{x'}_i\right)^2\right\} \quad \forall\, \theta_l \in [0\,;+\infty[\tag{10}$$

with $\left(\mathbf{w}_{*i}^{(l)}\right)_{i,l}$ which reveals the principal directions. Now, the function log-ML to optimize has h parameters. In the main paper, interesting results were obtained even for very small h, e.g. $h = 3$. However, we are once again maximizing a likelihood.

PLS for Bayesian There are two main options to improve the previous method with a Bayesian approach:

- We can be Bayesian regarding PLS, by putting a prior on h and performing a model selection to choose between the complexity and the accuracy (for instance). But in fact, we can carry out a statistical empirical study to select h nicely without any Bayesian tricks.
- We can make the best of the small number of parameters by sampling the prior to compute interesting Bayesian estimators.

We choose to tackle the second bullet because sampling the posterior of θ gives more probabilistic information about our model than just selecting h according to the empirical variability of our data. Moreover, sampling the posterior enables us to compute the integrated acquisition function as exposed in [5] p.4 and mentioned in section 2.2:

$$\hat{a}(\mathbf{x}; \{\mathbf{X}, \mathbf{y}\}) = \int a(\mathbf{x}; \{\mathbf{X}, \mathbf{y}\}, \theta) p(\theta | \mathbf{y}, \mathbf{X}, \hat{\boldsymbol{\beta}}, \hat{\sigma}^2) \, \mathrm{d}\, \theta$$

There are a wide variety of MCMC methods for sampling hyper-parameters of Gaussian Processes, as presented in [10]. For numerical simulation we rely on the most basic MCMC method: Metropolis-Hastings with a gaussian transition kernel and we compute the posterior density with the method detailed in 3.3.

5 Numerical simulations

In the previous sections, we have transformed as much as possible the frequentist treatments proposed in the main paper by Bayesian ones. We now compare the two approaches on numerical examples. The kriging model with a Gaussian kernel proposed in the main paper provides the benchmark against which the results of the new Bayesian approach are compared. The Python toolbox Scikit-learn [11] is used to implement these numerical tests. The computations were done on an Macbook Air (2017), 1,8 GHz Intel Core i5. The kriging models were built thanks to the SMT library [12].

We did not re-implement PLS or made use of a library containing PLS. Since PLS allows a dimension reduction resulting in a search space for θ of dimension 1, 2 or 3, we decided to do our experiments directly on a function on dimension 1.

We compare the frequentist (usual) kriging with a Bayesian one on the one-dimensional case with the sinus function : $f : \mathbf{x} \longmapsto \sin(2\pi\mathbf{x})$ on the interval $[0; 1]$. We evaluate the sinus on $n = 13$ points through a LHS (Latin Hypercube Sampling) [13]. For the regression of the mean function the basis functions are polynomials of order 2 $(1, \mathbf{x}, \mathbf{x}^2)$, so $\boldsymbol{\beta} \in \mathbb{R}^3$. In the Bayesian kriging, our goal is to find $\hat{\theta}$ by minimizing the log of the posterior of θ with the COBYLA algorithm as performed in the frequentist paradigm. To do so we rely on the method in section 3.3 presented previously. We detail our choice for our practical estimation. We estimate σ^2 by taking the MP estimator associate to the law INV$\Gamma(\alpha = 3, \delta = 15)$, to do so we set $\boldsymbol{\beta} = \hat{\boldsymbol{\beta}}_{MLE}$. Once σ^2 is computed, we use a Gaussian prior on $\boldsymbol{\beta}$: $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{b} = \hat{\boldsymbol{\beta}}_{MLE} + \epsilon, 10 I_3)$ where $\epsilon \sim \mathcal{N}(0, 3 I_3)$.

ϵ is a noise that we artificially add to avoid reusing exactly the MLE. The idea behind this computation is the following. In an industrial context, one might have a rough idea of what the MLE could be (for example thanks to previous simulations on similar functions), so this knowledge can be incorporated in the prior. Then we follow section 3.1 to compute $\hat{\beta}_{MP}$. Finally we compute the log of the posterior of θ by taking a Gaussian prior on θ : $\theta \sim \mathcal{N}(\theta_0 = \hat{\theta}_{MLE} + \epsilon', V > 0)$ where $\epsilon' \sim \mathcal{N}(0, 3)$ is a noise (same reasons as before). Thus, we are able to derive:

$$\log(p(\theta | \mathbf{y}, \mathbf{X}, \hat{\boldsymbol{\beta}}, \hat{\sigma}^2)) \propto_+ -\frac{1}{2} \Bigg(n \log(2\pi\hat{\sigma}^2) + \log|\mathbf{R}(\theta)| + \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^\top \mathbf{R}(\theta)^{-1} \left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) / \hat{\sigma}^2 + (\theta - \theta_0)^2 / V \Bigg)$$

Which represents the quantity to minimize with the COBYLA algorithm. To make a long story short, we follow the same method of [1] but we take Bayesian estimators. This method could be disappointing because we have a lot more probabilistic information in our Bayesian paradigm, that's why we choose to sample from the posterior of θ as well, in order to see the shape of its distribution and to take its mean to get the MP estimator.

In practice, we have two possibilities to sample from the posterior: whether we fix the others hyper-parameters $\sigma^2, \boldsymbol{\beta}$ and we sample from the posterior which is fixed, whether we fix only one of the hyper-parameters, for example σ^2 and we estimate $\boldsymbol{\beta}$ at each step of the sampling according to the last sampled θ. Here the posterior changes at every step, because $\boldsymbol{\beta}$ is updated. This last method enables us to get rid of the influence of the $\boldsymbol{\beta}$ fixed.

The best thing to do now is to see the code we made. All the figures of interests are automatically printed by executing the file main.py. The code works in dimension 1 for a function between $[0; 1]$. It would be easy to adapt it for a different interval or for a dimension higher than one. For sake of simplicity, we stick to this case.

The following relative error is used to compare the performance of the frequentist kriging model with the Bayesian model: Error $= 100||\hat{\mathbf{Y}} - \mathbf{Y}||_2/||\mathbf{Y}||_2$ where $||\ .\ ||_2$ represents the usual L2-norm, and $\hat{\mathbf{Y}}$ and $\mathbf{Y}$ are the vectors containing the prediction and the real values of random points, respectively. The CPU time required to fit models is also reported. For this 1D-case, the test set on which the error is computed is a LHS (Latin Hypercube Sampling) of 50 samples between 0.1 and 0.9 (to get rid of boundary effects inherent to Kriging).

In what follows, the results obtained after averaging over 30 experiments are presented. See code to have more explanations.

Method	**Error**	**CPU time**
Frequentist Kriging	4.2	0.012
Bayesian Kriging	4.4	0.016
Sampling of θ from posterior	7.3	10.1

On the experiments we did, the frequentist approach seems better than the Bayesian one, even if the gap is very tight. It remains difficult to draw general conclusions on the Bayesian approach compared to the frequentist one on only one experiment. Other functions must be tested, in particular the Griewank [14] function seems interesting since it does not have the same behavior in each direction :

$$y_{\text{Griewank}}(\mathbf{x}) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1, \quad -5 \leq x_i \leq 5 \quad \text{for } i = 1, ..., d$$

6 Conclusion

This work takes KPLS as a reference method and proposes a Bayesian version of it. It particularly focuses on turning MLEs into Bayesian estimators, such as MP or MAP estimators, and discusses the way prior knowledge can be used.

The next step consists in doing more numerical experiments, both on analytical and industrial examples. The first simulations done let us think that a clever mix of MLEs and Bayesian estimator associated to good prior should gives better results than a full frequentist treatment.

References

[1] Mohamed Amine Bouhlel, Nathalie Bartoli, Abdelkader Otsmane, and Joseph Morlier. Improving kriging surrogates of high-dimensional design models by partial least squares dimension reduction. *Structural and Multidisciplinary Optimization*, 53(5):935–952, 2016.

[2] P Piperni, A DeBlois, and R Henderson. Development of a multilevel multidisciplinary-optimization capability for an industrial environment. *AIAA journal*, 51(10):2335–2352, 2013.

[3] Donald R Jones and Matthias Schonlau. and welch, w.(1998)," efficient global optimization of expensive black-box functions,". *Journal of global optimization*, pages 455–492.

[4] Dimitri Bettebghor, Nathalie Bartoli, Stéphane Grihon, Joseph Morlier, and Manuel Samuelides. Surrogate modeling approximation using a mixture of experts based on em joint estimation. *Structural and multidisciplinary optimization*, 43(2):243–259, 2011.

[5] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[6] Mohamed Amine Bouhlel, Nathalie Bartoli, Abdelkader Otsmane, and Joseph Morlier. An improved approach for estimating the hyperparameters of the kriging model for high-dimensional problems through the partial least squares method. *Mathematical Problems in Engineering*, 2016, 2016.

[7] Alexander Forrester, Andras Sobester, and Andy Keane. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[8] Michael JD Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in optimization and numerical analysis*, pages 51–67. Springer, 1994.

[9] Yiqi Liu, Yongping Pan, and Daoping Huang. Development of a novel adaptive soft-sensor using variational bayesian pls with accounting for online identification of key variables. *Industrial & Engineering Chemistry Research*, 54(1):338–350, 2015.

[10] Iain Murray and Ryan P Adams. Slice sampling covariance hyperparameters of latent gaussian models. In *Advances in neural information processing systems*, pages 1732–1740, 2010.

[11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[12] Mohamed Amine Bouhlel, John T Hwang, Nathalie Bartoli, Rémi Lafage, Joseph Morlier, and Joaquim RRA Martins. A python surrogate modeling framework with derivatives. *Advances in Engineering Software*, 135:102662, 2019.

[13] Michael Stein. Large sample properties of simulations using latin hypercube sampling. *Technometrics*, 29(2):143–151, 1987.

[14] Rommel G Regis. An initialization strategy for high-dimensional surrogate-based expensive black-box optimization. In *Modeling and Optimization: Theory and Applications*, pages 51–85. Springer, 2013.

[15] Anthony O'Hagan. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society: Series B (Methodological)*, 40(1):1–24, 1978.

[16] CE Rasmussen and C Williams. Gaussian processes for machine learning, ser. adaptive computation and machine learning. *Cambridge, MA, USA: MIT Press*, 38:715–719, 2006.