MascotNum2021 conference - Sequential incrementation of the dimension in computer experiments

T. Gonon
Ecole Centrale de Lyon

Supervisor(s): C. Blanchet-Scalliet (Ecole Centrale de Lyon), C. Helbert (Ecole Centrale de Lyon) and B. Demory (Valeo)

Ph.D. expected duration: Feb. 2019 - Jan. 2022

Adress: Univ Lyon, Ecole Centrale de Lyon, CNRS UMR 5208, Institut Camille Jordan, 36 avenue Guy de Collongue, F-69134 Ecully Cedex, France

Email: thierry.gonon@ec-lyon.fr

Abstract:

Industrial products are studied numerically before being sold. The corresponding numerical codes involve geometrical or environmental inputs and physical outputs. They are complex as they involve lots of input variables. Studies are provided to quantify the influence of the inputs on the behavior of the outputs. They usually use metamodels based on a set of already simulated cases (called a Design Of Experiments DOE). The first studies focus on a small amount of inportant inputs, letting them vary freely to see their influence on the output, while the other inputs are fixed to nominal values. Then, the studies are complexified as some previously fixed inputs are progressively involved.

The classical way of treating the increasing number of inputs is to start from scratch each time new inputs are released, regenerating a DOE and a metamodel independent from the previous ones. This approach can be time consuming and the previous data is lost because unused.

An alternative, presented in our work, is to upgrade gradually the design and the metamodel, based on the previous ones. This enables to exploit all the available data. The surrogate approach is based on gaussian process regression. At each step k, the output is supposed to be the realization of a gaussian process which is the sum of two processes : the first one is defined on the previous input space of step $k-1$ (subspace of the current input space of step k), the second is built independently on the input space of step k but is conditionned to be null on the subspace of step $k-1$.

In order to gain in efficiency, we illustrate the method on an example with an output function of 4 inputs $x = (x_1, x_2, x_3, x_4)$: $y(x_1, x_2, x_3, x_4) = f_1(x_1, x_2) + f_2(x_1, x_2, x_3, x_4)$, with

$$\begin{cases} f_1(x_1, x_2) & = & \left[4 - 2.1(4x_1-2)^2 + \frac{(4x_1-2)^4}{3}\right](4x_1-2)^2 \\ & + & (4x_1-2)(2x_2-1) + \left[-4+4(2x_2-1)^2\right](2x_2-1)^2 \\ f_2(x_1, x_2, x_3, x_4) & = & 4\exp\left(-\|x-0.3\|^2\right) \end{cases}$$

We consider two cases :

- At step 0, we study $y(x_1, x_2, \frac{x_1+x_2}{2}, 0.2x_1 + 0.7)$. Only the first two inputs x_1 and x_2 can vary freely and the other two inputs are fixed : $x_3 = \frac{x_1+x_2}{2}$, $x_4 = 0.2x_1 + 0.7$. For this case, we dispose of a DOE and a metamodel.

- At step 1, we study $y(x_1, x_2, x_3, x_4)$. The last two inputs (x_3, x_4) are freed. We must create a new DOE and a new metamodel depending on all the inputs x_1, x_2, x_3, and x_4, but using the previous information (DOE and metamodel).

We choose to model the output by a Gaussian Process Y. The modelization is inspired from the Multifidelity framework [3].We decompose Y as follows :

$$Y(x_1,x_2,x_3,x_4) = \begin{cases} Y_0(x_1,x_2) & = & Z_0(x_1,x_2) & \text{when } x_3 = \frac{x_1+x_2}{2} \text{ and } x_4 = 0.2x_1+0.7 \\ Y_1(x_1,x_2,x_3,x_4) & = & Y_0(x_1,x_2) + Z_1(x_1,x_2,x_3,x_4) & \text{else} \end{cases}$$

with Z_0 and Z_1 independent Gaussian Processes. The definition of Y implies :

$$Z_1(x_1, x_2, \tfrac{x_1+x_2}{2}, 0.2x_1+0.7) \quad = \quad 0 \quad \forall x_1, x_2$$

We must solve some issues :

- What Gaussian Process can satisfy the property verified by Z_1 ? We propose two nonstationnary processes built from a second order stationnary one (noted $\widetilde{Z}_1$). The candidates are :
  - $Z_1(x_1,x_2,x_3,x_4) = \widetilde{Z}_1(x_1,x_2,x_3,x_4) - \widetilde{Z}_1(x_1,x_2,\frac{x_1+x_2}{2},0.2x_1+0.7)$ that we call Red Process.
  - $Z_1(x_1,x_2,x_3,x_4) = \left[\widetilde{Z}_1(x_1,x_2,x_3,x_4) \mid \widetilde{Z}_1(t_1,t_2,\frac{t_1+t_2}{2},0.2t_1+0.7) = 0 \forall t_1\right]$ (it is the extension of the usual notation of the conditionning on a finite set of points but for an infinite set of points in this case) following the work of [2]. We call it P Process.
- How to estimate the parameters of such a model ? We choose the maximum likelihood estimation. To optimize the likelihood, we adapt the EM algorithm [1] to our case.

We compare our methodology to a classic kriging metamodel trained on the reunion of the samples of all steps. The results are encouraging (see figure 1). The information from the previous step is useful as it enables to decrease the RMSE of the kriging metamodel. The method with the Red Process is a relevant metamodel for that case as it performs better than the kriging metamodel.

References

[1] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[2] Bertrand Gauthier. *Approche spectrale pour l'interpolation à noyaux et positivité conditionnelle.* PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne, 2011.

[3] Marc C Kennedy and Anthony O'Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000.

Short biography – My name is Thierry Gonon. I am at the end of my second year of phd. I previously was an engineer student at Ecole Centrale de Lyon. My last internship at the school took place at Valeo in the Thermal System branch. I was in charge of implementing kriging tools (Prediction, Confidence intervals, EGO...) in their software to treat their numerical CFD simulations. At the end of my internship, we decided to continue with a phd in collaboration between Ecole Centrale de Lyon and Valeo.

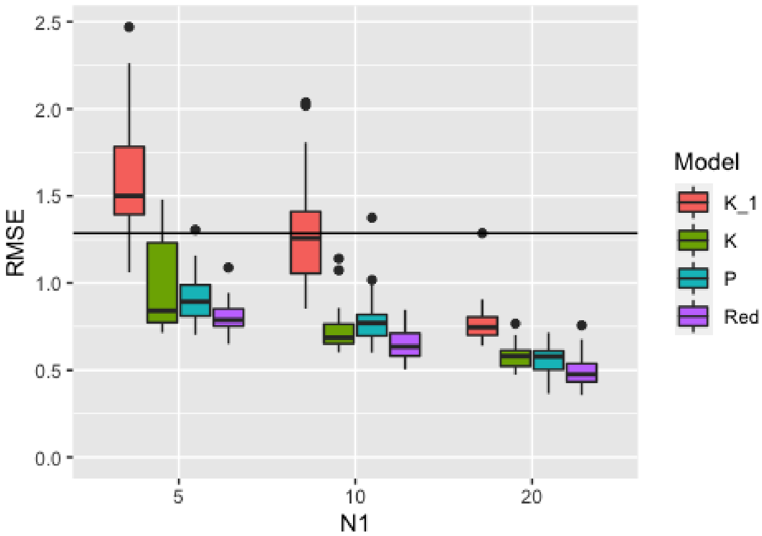

Figure 1: RMSE of different models for the toy function for different sizes of the training sample of step 1 (N1), the size of the training sample of step 0 being 10. K_1 is the kriging metamodel trained only on the training sample of step 1, K is the kriging metamodel trained on the union of the two training samples, P is the method using the P Process, Red is the method using the Red Process.