2009 First International Conference on Advances in System Simulation

Numerical Study of the Metamodel validation process

Bertrand IOOSS
Commissariat à l'Energie Atomique
CEA, DEN, Cadarache
Saint-Paul-lez-Durance, France
bertrand.iooss@cea.fr

Abstract - Complex computer codes are often too time expensive to be directly used to perform uncertainty, sensitivity, optimization and robustness analyses. A widely accepted method to circumvent this problem consists in replacing cpu-time expensive computer models by cpu-inexpensive mathematical functions, called metamodels. In this paper, we focus on the essential step of the metamodel validation phase which consists in evaluating the metamodel predictivity. It allows to allocate some confidence degrees to the results obtained by using the metamodel instead of the initial numerical model. We propose and test an algorithm which optimizes the distance between the validation points and the metamodel training points in order to estimate the true metamodel predictivity with a minimum number of additional calculations. Comparisons are made with classical validation algorithms and application to a nuclear safety computer code is shown. These tests show the relevance of this new validation design called the Feuillard design.

Metamodel; Gaussian process; discrepancy; validation design; computer experiment

1. Introduction

With the advent of computing technology and numerical methods, investigation of computer code experiments remains an important challenge. Complex computer models calculate several output values (scalars or functions) which can depend on a high number of input parameters and physical variables. These computer models are used to make simulations as well as predictions, uncertainty analyses or sensitivity studies [2].

However, complex computer codes are often too time expensive to be directly used to conduct uncertainty propagation studies or global sensitivity analysis based on Monte Carlo methods. To avoid the problem of huge calculation time, it can be useful to replace the complex computer code by a mathematical approximation, called a metamodel [12][7]. Several metamodels are classically used: polynomials, splines, generalized linear models, or learning statistical models like neural networks, regression trees, support vector machines [3]. One particular class of metamodels, the Gaussian process model, extends the kriging principles of geostatistics to computer experiments by considering the correlation between two responses of a computer code depending on the distance between input variables [12]. Numerous studies have shown that this interpolating model provide a powerful statistical framework to compute an efficient predictor of code response [13][8].

The validation of a metamodel is an essential step in practice [7]. By estimating the metamodel predictivity, it gives us a confidence degree associated with the use of the metamodel instead of the initial numerical model. Two validation methods are ordinarily used: the test basis approach [6] and the cross validation method [9][10]. In this paper, we propose to perform numerical studies of the metamodel predictivity with respect to these validation methods. First, we describe these two classical validation methods. Then we present a new validation design (called the Feuillard design) which aims at minimizing the additional number of simulation points. We illustrate the relevance of this new design by performing intensive simulation on two analytical functions. Finally, a real example is addressed in the last section.

2. Classical validation methods

Let us consider the d-dimensional input vector $\boldsymbol{x} = (x_1, \ldots, x_d) \in \mathcal{X}$, where $\mathcal{X}$ is a bounded domain of $\mathbb{R}^d$ and $y(\boldsymbol{x}) \in \mathbb{R}$ is the computer code output. We suppose that a metamodel $\hat{Y}(\boldsymbol{x})$ has been fitted using $\left((\boldsymbol{x}^{(1)}, y(\boldsymbol{x}^{(1)})), \ldots, (\boldsymbol{x}^{(N)}, y(\boldsymbol{x}^{(N)}))\right)$, a N-size training sample of computer code experiments. This sample is also called the learning sample.

The test basis approach consists in comparing the metamodel predictions on simulation points not used in the metamodel fitting process. This gives some prediction residuals (which can be finely analyzed) and global quality measures as the metamodel predictivity coefficient Q_2:

$$Q_2 = 1 - \frac{\sum_{i=1}^{N_{\text{test}}} \left[y(\boldsymbol{x}_t^{(i)}) - \hat{Y}(\boldsymbol{x}_t^{(i)}) \right]^2}{\sum_{i=1}^{N_{\text{test}}} \left[\bar{y} - y(\boldsymbol{x}_t^{(i)}) \right]^2} \qquad (1)$$

with $(\boldsymbol{x}_t^{(1)}, \ldots, \boldsymbol{x}_t^{(N_{\text{test}})})$ the test sample of size N_{test} and $\bar{y}$ the mean of the output test sample

978-0-7695-3773-3/09 $26.00 © 2009 IEEE
DOI 10.1109/SIMUL.2009.8

$(y(\boldsymbol{x}_t^{(1)}), \ldots, y(\boldsymbol{x}_t^{(N_{\text{test}})}))$. Such test points set is called a test basis (or also validation basis or prediction basis). This method requires new calculations with the computer code and the first question we have to face up is the right number of prediction points to obtain the required accuracy of our global validation measures. For cpu time consuming computer code, it can be difficult to provide a sufficient number of test points. Some convergence visualisation tools of the global validation measures can be used to answer to this first question. Another important question is the localization of these test points. The usual practice is to choose an independent Monte Carlo sample for the test basis. However, if the sample size is small, the proposed points can be badly localized, for example near learning points or leaving large space domain unsampled. A fine strategy could be to use a space filling design (which consists to fill the input variables space $\mathcal{X}$ as uniformly as possible [3]) as the test sample. Unfortunately, this solution does not avoid the possibility of too strong proximity between learning points and test points. Such proximity would lead to too optimistic quality measures.

The second solution to validate a metamodel, the cross validation method, is extremely popular in practice because it avoids new calculations on the computer code. The cross validation method proposes to divide the initial sample on a learning sample and a test sample. A metamodel is estimated with the points in the new learning sample and prediction residuals are obtained via the new test sample. This process is repeated several times by using other divisions of the learning sample. Finally all the prediction residuals can be used to compute the global predictivity measures. The leave-one-out procedure is a particular case of the cross validation method where just one observation is left out at each step.

The first drawback of the cross validation method is its cost, which can become large due to many metamodel fitting processes. Moreover, if the initial design has a specific geometric structure (which aimed to optimize the metamodel fitting), the deletion of points (put in the test sample) from the learning sample causes the breakdown of the specific design structure while creating the new learning sample. The new learning sample does not possess the adequate statistical and geometric properties of the initial design and the metamodel fitting process might fail. This could lead to too pessimistic quality measures.

To sum up, the test basis method requires too many new prediction points while the cross-validation method can provide too pessimistic validation criteria. Therefore, to solve this dilemma, an heuristic new solution has been introduced in [5] and is presented in the next section.

3. A new optimized validation design

Retaining the test basis method, our algorithm consists in creating a new test basis and limiting its main drawback by minimizing the number of new prediction points which are required. In this goal, it uses an algorithm which allows the specification of new design points decreasing the discrepancy of an initial design [4]. This sequential algorithm gives us at each step the prediction point furthest away from the other points of the design. The algorithm performs its optimization process in the space $\mathcal{X}$ of the input variables $\boldsymbol{x}$. By choosing the future prediction points in the unfilled zone of the learning sample design, we aim at capturing the right metamodel predictivity using only a small number of additional points. Note that such ideas have also been proposed in [11] for different purposes.

We have not theoretically studied the computational efficiency of this algorithm over the computational efficiency of the traditional methods (introduced in the previous section). However, our intuition is that mean square error computed by this algorithm avoids the biases which could be caused by too strong proximities between the test sample points and between test sample points vs. learning sample points.

Let us consider $X_f(n_f) = (\boldsymbol{x}_f^{(i)})_{i=1..n_f}$ a low discrepancy sequence of n_f points in $[0,1]^d$. A low discrepancy sequence is a deterministic design constructed to uniformly fill the space with regular patterns. Among all the low discrepancy sequence, Halton, Hammersley, Faure and Sobol sequences are the most famous. In the following, we will use the Hammersley sequence which, on a few tests, have shown better properties than the others [4]. The chosen discrepancy measure is the centered L^2 discrepancy D:

$$D^2(X_s(n)) = \left(\frac{13}{12}\right)^d - \frac{2}{n}\sum_{i=1}^{n}\prod_{k=1}^{d}\left(1+\frac{1}{2}|u_k^{(i)}-\frac{1}{2}|-\frac{1}{2}|u_k^{(i)}-\frac{1}{2}|^2\right) + \frac{1}{n^2}\sum_{i,j=1}^{n}\prod_{k=1}^{d}\left(1+\frac{1}{2}|u_k^{(i)}-\frac{1}{2}|+\frac{1}{2}|u_k^{(j)}-\frac{1}{2}|-\frac{1}{2}|u_k^{(i)}-u_k^{(j)}|\right) \quad (2)$$

where d is the dimension of the input vector, $X_s(n)$ denotes the input learning sample with n input vectors and $\left(u_k^{(i)}\right)_{i=1..n,k=1..d}$ are the normalized values in $[0,1]$ of the design $X_s(n) = \left(x_k^{(i)}\right)_{i=1..n,k=1..d}$.

To obtain an additional point of the initial N-size sample, noticed $X(N)$, we use the following algorithm:

1) For $i = 1, \ldots, n_f$,
   - $X(N+1) = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\} \cup \boldsymbol{x}_f{}^{(i)}$;
   - calculation of $\text{Dif}_i = D(X(N+1)) - D(X(N))$;
2) selection of i^* such that $\text{Dif}_{i^*} = \min_{i=1,\ldots,n_f} \text{Dif}_i$;
3) the new point is $\boldsymbol{x}_f{}^{(i^*)}$.

This algorithm is repeated sequentially to obtain N_{test} test points, by updating the initial design and the low discrepancy sequence. For example, for the second point, we reinitalize the design by the following: $X(N+1) = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\} \cup \boldsymbol{x}_f{}^{(i^*)}$ and $X_f(n_f - 1) = \{\boldsymbol{x}_f{}^{(1)}, \ldots, \boldsymbol{x}_f{}^{(n_f)}\} \backslash \boldsymbol{x}_f{}^{(i^*)}$.

This algorithm just consists in adding to the initial design some points of a low discrepancy sequence by minimizing the discrepancies difference between the initial and the new design. The size of the low discrepancy sequence is required to be as large as possible, especially if d is large. Figure 1 gives an example of the specification with our algorithm of $N_{\text{test}} = 4$ new points (the crosses) inside an initial Monte Carlo design ($N = 46$, $d = 2$) One of the advantage of this algorithm is its size-independence (related to the number of added points): the sequence of added points is deterministic and will be always the same for the same $X_f(n_f)$. In the following, the design obtained using this algorithm is called the Feuillard design.

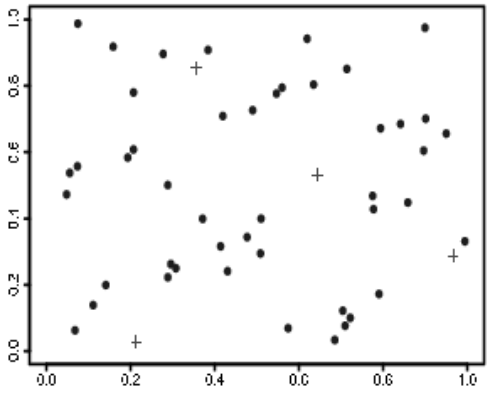

Figure 1. Example of the sequential algorithm: $N = 46$, $d = 2$, $N_{\text{test}} = 4$. The bullets are the points of the initial design while the crosses are the new specified points.

4. Tests on toy functions

To compare the Feuillard design with other test designs for the metamodel validation purpose, we first perform an analytical test using a two-dimensional toy function:

$$f(\boldsymbol{x}) = \cos(10x_1) + \sin(10x_2) + x_1x_2\ ,\ (x_1, x_2) \in [0,1]^2.$$

Gaussian process metamodels are fitted using learning samples of differents sizes N_{BA}: N_{BA} ranges from 10 to 40 allowing a wide variety of metamodel predictivity coefficients Q_2, from 0 (null predictivity) to 1 (perfect predictivity). The initial 10-size design is a maximin Latin Hypercube Sample (LHS) which is known to be a good design for Gaussian process metamodel fitting [3]. The other training designs (of increased size) are obtained by sequentially adding points to the design, while maintaining the Latin properties of the design and keeping some optimality properties (maximizing the mean distance from each design point to all the other points in the design). The black line in Figure 2 shows the evolution of Q_2 in function of the learning sample size. This "exact" predictivity coefficient has been computed for each metamodel by mean of 100 test samples of size $N_{\text{test}} = 1000$. The Q_2 estimation by a leave one out procedure (pink line) strongly underestimates the exact Q_2 for $N_{\text{BA}} < 30$. This is certainly due to the small number of points: leave one out is pessimistic in this case because each point deletion has a strong impact on the metamodel fitting process. The red curve gives the Q_2 estimation using the Feuillard design described in the previous paragraph (with a Hammersley sequence of size $n_f = 10000$). Results are greatly satisfactory for $N_{\text{test}} \geq 20$: the Feuillard design gives precise Q_2 estimates in all cases and outperforms a pure Monte Carlo or LHS design. The green curves correspond to the minimal and maximal values obtained with 100 repetitions of an optimized LHS as the test design. As expected, these intervals are more reduced than the intervals obtained using a pure Monte Carlo sample as the test design (blue curves). As N_{test} increases, these intervals contract, but always show the superiority of the Feuillard design, specially for low metamodel predictivity ($Q_2 < 0.9$ and $N_{\text{BA}} < 25$).

We perform now a second numerical test using an eight-dimensional analytical function (called the g-Sobol function):

$$f(\boldsymbol{x}) = \sum_{i=1}^{8} \frac{|4x_i - 2| + a_i}{1 + a_i}$$

with $a_1 = a_2 = 3$, $a_i = 0$ for ($i = 3, \ldots, 8$), $\boldsymbol{x} \in [0,1]^8$. A Gaussian process model is fitted on a learning sample (maximin LHS) of size $N_{\text{BA}} = 40$. We compute the "exact" predictivity coefficient by mean of 100 test samples of size $N_{\text{test}} = 1000$ and obtain $Q_2^{\text{ref}} = 0.83$. We then apply the Feuillard design described previously (with a Hammersley sequence of size $n_f = 10000$) by adding $N_{\text{test}} = 50$ new points to the design, and we obtain $Q_2^{\text{seq50}} = 0.85$, which is close to the true value. We compare this result with 100 pure Monte-Carlo samples of the same size ($N_{\text{test}} = 50$) which give the 90% confidence interval $[0.79, 0.91]$ for Q_2^{MC}. This last result is rather large and shows the insufficient number of points if we choose a pure Monte-Carlo design. Figure 3 shows the evolution of the estimated Q_2 for test bases with different sizes, ranging from $N_{\text{test}} = 10$ to $N_{\text{test}} = 50$. The solid red line shows the results obtained with the Feuillard design while the dotted blue lines show the 100 sequentially increased pure Monte-Carlo samples. This figure illustrates the poor estimates we obtain when using small size ($N_{\text{test}} < 50$) of Monte-Carlo samples for validation. On the contrary, the Feuillard design allows to obtain a good approximation of the true predictivity coefficient even for small test sample sizes. Results are precise for $N_{\text{test}} \geq 25$.

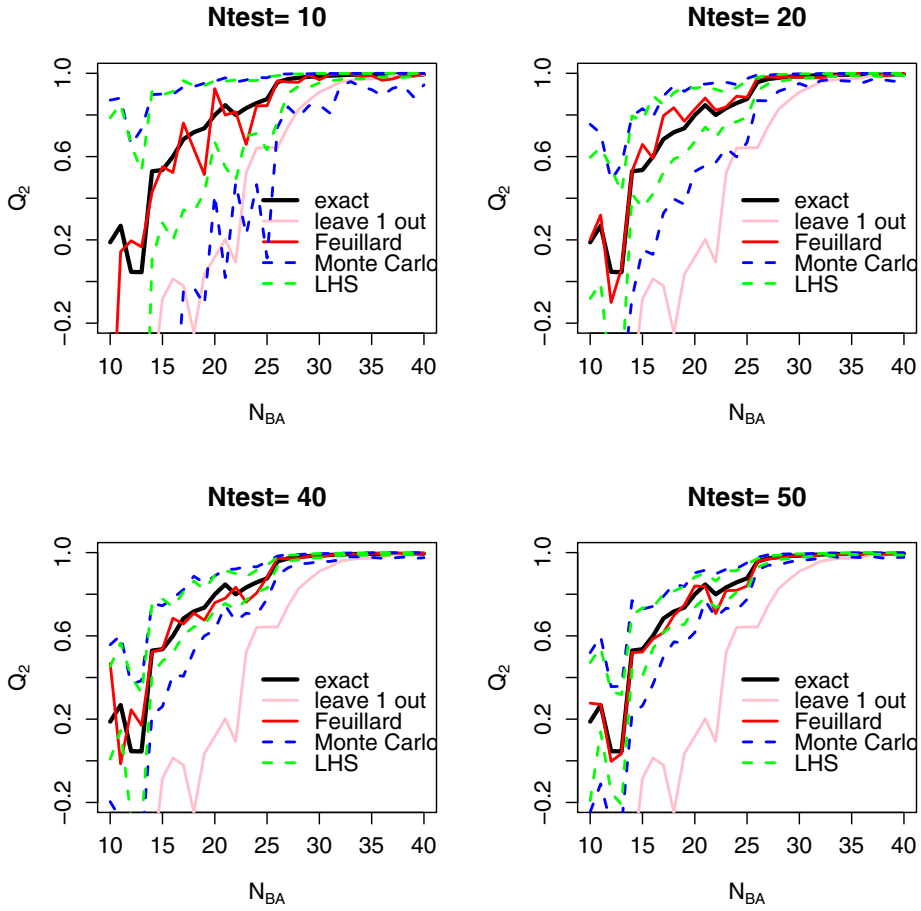

Figure 2. For the two-dimensional toy function, estimation of the metamodel (Gaussian process) predictivity coefficient (Q_2) in function of the learning sample size N_{BA}, for different test sample sizes N_{test}. The dashed curves (blue and green) give the minimal and maximal values obtained with 100 repetitions of the test design.

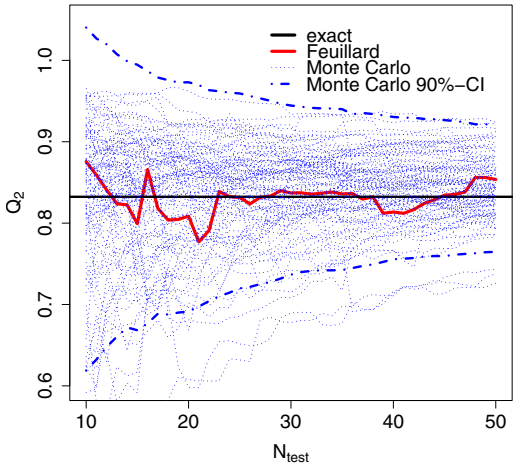

Figure 3. For the g-Sobol function, estimation of the metamodel (Gaussian process) predictivity coefficient (Q_2) in function of the test sample size N_{test}, for two types of validation design: Feuillard (red) and pure Monte Carlo (blue). Dotted blue lines correspond to 100 different pure Monte-Carlo samples).

5. Application to a nuclear safety computer code

In this section we apply our algorithms on a complex computer model used for nuclear reactor safety. It simulates a hypothetic thermal-hydraulic scenario: a large-break loss of coolant accident for which the quantity of interest is the peak cladding temperature. This scenario is part of the Benchmark for Uncertainty Analysis in Best-Estimate Modelling for Design, Operation and Safety Analysis of Light Water Reactors [1] proposed by the Nuclear Energy Agency of the Organisation for Economic Co-operation and Development (OCDE/NEA). It has been implemented on the computer code Cathare of the Commissariat à l'Energie Atomique (CEA).

In our exercise a Gaussian process metamodel of the peak cladding temperature has to be estimated with $N = 100$ computations of the computer model (the input design is a maximin LHS). The CPU time is twenty minutes for each simulation. The complexity of the computer model lies in the high-dimensional input space: $d = 53$ random input variables (physical laws essentially, but also initial conditions, material properties and geometrical modeling) are considered, with normal and log-normal distributions. This number is rather large for the metamodel construction problem. This difficult fit (due to the high dimensionality and small learning sample size) can be realized thanks to the algorithm of [8], specifically devoted to this situation. The obtained Gaussian process metamodel contains a linear regression part (including 7 input variables) and a stochastic Gaussian process with a generalized exponential covariance function (including 6 input variables). The reference quality of this Gaussian process model is measured via an additional 1000-size test sample which gives $Q_2^{\text{ref}} = 0.66$. Figure 4 shows the evolution of the estimated Q_2 for test bases with different sizes, ranging from $N_{\text{test}} = 10$ to $N_{\text{test}} = 95$. The Feuillard design gives coarse estimations for all the test design sizes and begins to give precise results for $N_{\text{test}} \geq 40$. Some inadequacies, which remain when $N_{\text{test}} \in [75, 90]$, have to be finely analyzed in a further work. In any cases, Feuillard design estimations are clearly less hazardous than using a pure Monte Carlo test sample to validate the metamodel: the 90%-confidence intervals obtained using Monte carlo samples show extremely large variation ranges (because of the high dimensionality of the input space: $d = 53$). Q_2 estimation using a Monte Carlo test sample can lead to a strongly erroneous result. Moreover, same results have been obtained using optimized LHS for the test design instead of a pure Monte Carlo sample.

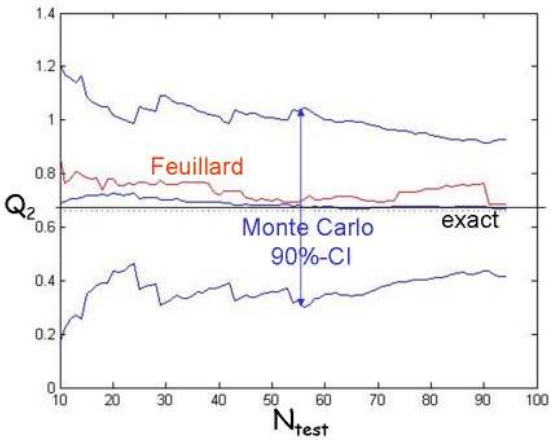

Figure 4. For the nuclear safety computer code application, estimation of the metamodel (Gaussian process) predictivity coefficient (Q_2) in function of the test sample size N_{test}, for two types of validation design: Feuillard (red) and pure Monte Carlo (blue).

6. Conclusion

In this paper, we have looked at the metamodel validation process and have shown that the test basis approach can provide erroneous results for small sizes of the test sample. Moreover, the leave one out approach can strongly underestimate the metamodel predictivity for small sizes of the whole database. We have proposed to use a recent algorithm, called the Feuillard design, which puts prediction points in the unfilled zones of the learning sample design. Therefore, a minimal number of points is required to obtain a good estimation of the metamodel predictivity.

Our numerical tests on analytical functions and real application cases have shown that the Feuillard design outperforms the classical metamodel validation methods, specially in high dimensional context. For our analytical functions, the Feuillard design gives precise estimate of the metamodel predictivity with a test sample size $N_{\text{test}} \geq 25$, while for our industrial application, the minimal bound is $N_{\text{test}} \geq 40$. Further works are necessary to more deeply study the validation designs (other test functions with different effective dimensionality and complexity). Moreover, it would be useful to find a criterion of determining when to terminate the validation using the Feuillard design.

Acknowledgments

The authors would like to thank Vincent Feuillard for providing his R programs (specification algorithms of new points) and Amandine Marrel for the Gaussian process metamodel fitting of the nuclear safety computer code outputs. The analytical tests have been performed using the `lhs` R package for creating specific designs and the `mlegp` R package for the Gaussian process metamodel fitting.

References

[1] A. De Crécy, P. Bazin, H. Glaeser, T. Skorek, J. Joucla, P. Probst, K. Fujioka, B.D. Chung, D.Y. Oh, M. Kyncl, R. Pernica, J. Macek, R. Meca, R. Macian, F. DAuria, A. Petruzzi, L. Batet, M. Perez, and F. Reventos Uncertainty and sensitivity analysis of the LOFT L2-5 test: Results of the BEMUSE programme. *Nuclear Engineering and Design*, 12:3561–3578, 2008.

[2] E. De Rocquigny, N. Devictor, and S. Tarantola, editors. *Uncertainty in industrial practice*. Wiley, 2008.

[3] K-T. Fang, R. Li, and A. Sudjianto. *Design and modeling for computer experiments*. Chapman & Hall/CRC, 2006.

[4] V. Feuillard. *Analyse d'une base de données pour la calibration d'un code de calcul*. Thèse de l'Université Pierre et Marie Curie - Paris VI, 2007.

[5] B. Iooss, L. Boussouf, A. Marrel, and V. Feuillard. Numerical study of algorithms for metamodel construction and validation. In S. Martorell, C. Guedes Soares, and J. Barnett, editors, *Safety, reliability and risk analysis - Proceedings of the ESREL 2008 Conference*, pages 2135–2141, Valencia, Spain, september 2008. CRC Press.

[6] B. Iooss, F. Van Dorpe, and N. devictor. Response surfaces and sensitivity analyses for an environmental model of dose calculations. *Reliability Engineering and System Safety*, 91:1241–1251, 2006.

[7] J.P.C. Kleijnen and R.G. Sargent. A methodology for fitting and validating metamodels in simulation. *European Journal of Operational Research*, 120:14–29, 2000.

[8] A. Marrel, B. Iooss, F. Van Dorpe, and E. Volkova. An efficient methodology for modeling complex computer codes with Gaussian processes. *Computational Statistics and Data Analysis*, 52:4731–4744, 2008.

[9] M. Meckesheimer, A.J. Booker, R.R. Barton, and T.W. Simpson. Computationally inexpensive metamodel assessment strategies. *AIAA Journal*, 40:2053–2060, 2002.

[10] M.I. Reis dos Santos and A.M.O. Porta Nova. Statistical fitting and validation of non-linear simulation metamodels: A case study. *European Journal of Operational Research*, 171:53–63, 2006.

[11] G. Rennen. Subset selection from large datasets for kriging modeling. *Structural and Multidisciplinary Optimization*, 38:545-569, 2009.

[12] J. Sacks, W.J. Welch, T.J. Mitchell, and H.P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4:409–435, 1989.

[13] T. Santner, B. Williams, and W. Notz. *The design and analysis of computer experiments*. Springer, 2003.