An efficient methodology for modeling complex computer codes with Gaussian processes

Amandine Marrel, Bertrand Iooss, Francois van Dorpe, Elena Volkova

An efficient methodology for modeling complex computer codes with Gaussian processes

Amandine MARREL[*,1], Bertrand IOOSS[†],

François VAN DORPE[⋆], Elena VOLKOVA[‡]

[*] CEA Cadarache, DEN/DTN/SMTM/LMTE
[†] CEA Cadarache, DEN/DER/SESI/LCFR
[⋆] CEA Cadarache, DEN/D2S/SPR
[‡] Kurchatov Institute, Russia

Abstract

Complex computer codes are often too time expensive to be directly used to perform uncertainty propagation studies, global sensitivity analysis or to solve optimization problems. A well known and widely used method to circumvent this inconvenience consists in replacing the complex computer code by a reduced model, called a metamodel, or a response surface that represents the computer code and requires acceptable calculation time. One particular class of metamodels is studied: the Gaussian process model that is characterized by its mean and covariance functions. A specific estimation procedure is developed to adjust a Gaussian process model in complex cases (non linear relations, highly dispersed or discontinuous output, high dimensional input, inadequate sampling designs, etc.). The efficiency of this algorithm is compared to the efficiency of other existing algorithms on an analytical test case. The proposed methodology is also illustrated for the case of a complex hydrogeological computer code, simulating radionuclide transport in groundwater.

Keywords: Gaussian process, kriging, response surface, uncertainty, covariance, variable selection, computer codes.

[1]Corresponding author: A. Marrel
DEN/DTN/SMTM/LMTE, 13108 Saint Paul lez Durance, Cedex, France
Phone: +33 (0)4 42 25 26 52, Fax: +33 (0)4 42 25 62 72
Email: amandine.marrel@cea.fr

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 or sensitivity studies. Importance measures of each uncertain input variable on the response variability provide guidance to a better understanding of the modeling in order to reduce the response uncertainties most effectively (Saltelli et al. (2000), Kleijnen (1997), Helton et al. (2006)).

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 response surface or a surrogate model or also a metamodel. The response surface method (Box and Draper (1987)) consists in constructing a function that simulates the behavior of real phenomena in the variation range of the influential parameters, starting from a certain number of experiments. Similarly to this theory, some methods have been developed to build surrogates for long running computer codes (Sacks et al. (1989), Osio and Amon (1996), Kleijnen and Sargent (2000), Fang et al. (2006)). Several metamodels are classically used: polynomials, splines, generalized linear models, or learning statistical models such as neural networks, support vector machines, ... (Hastie et al. (2002), Fang et al. (2006)).

For sensitivity analysis and uncertainty propagation, it would be useful to obtain an analytic predictor formula for a metamodel. Indeed, an analytical formula often allows the direct calculation of sensitivity indices or output uncertainties. Moreover, engineers and physicists prefer interpretable models that give some understanding of the simulated physical phenomena and parameter interactions. Some metamodels, such as polynomials (Jourdan and Zabalza-Mezghani (2004), Kleijnen (2005), Iooss et al. (2006)), are easily interpretable but not always very efficient. Others, for instance neural networks (Alam et al. (2004), Fang et al. (2006)), are more efficient but do not provide an analytic predictor formula.

The kriging method (Matheron (1970), Cressie (1993)) has been developed for spatial interpolation problems; it takes into account spatial statistical structure of the estimated variable. Sacks et al. (1989) have extended the kriging principles to computer experiments by considering the correlation between two responses of a computer code depending on the distance between input variables. The kriging model (also called Gaussian process model), characterized by its mean and covariance functions, presents several advantages, especially the interpolation and interpretability properties. Moreover, numerous authors (for example, Currin et al. (1991), Santner et al. (2003) and Vazquez et al. (2005)) show that this model can provide a statistical framework to compute an efficient predictor of code response.

From a practical standpoint, constructing a Gaussian process model implies estimation of several hyperparameters included in the covariance function. This optimization problem is particularly difficult for a model with many inputs and inadequate sampling designs (Fang et al. (2006), O'Hagan (2006)). In this paper, a special estimation procedure is developed to fit a Gaussian process model in complex cases (non linear relations, highly dispersed output, high dimensional input, inadequate sampling designs). Our purpose includes developing a procedure for parameter estimation via an essential step of input parameter selection. Note that we do not deal with the design of experiments in computer code simulations (i.e. choosing values of input parameters). Indeed, we work on data obtained in a previous study (the hydrogeological model of Volkova et al. (2008)) and try to adapt a Gaussian process model as well as possible to a non-optimal sampling design. In summary, this study presents two main objectives: developing a methodology to implement and adapt a Gaussian process model to complex data while studying its prediction capabilities.

The next section briefly explains the Gaussian process modeling from theoretical expression to predictor formulation and model parameterization. In section 3, a parameter estimation procedure is introduced from the numerical standpoint and a global methodology of Gaussian process modeling implementation is presented. Section 4 is devoted to applications. First, the algorithm efficiency is compared to other algorithms for the example of an analytical test case. Secondly, the algorithm is applied to the data set (20 inputs and 20 outputs) coming from a hydrogeological transport model based on waterflow and diffusion dispersion equations. The last

section provides some possible extensions and concluding remarks.

2 GAUSSIAN PROCESS MODELING

2.1 Theoretical model

Let us consider n realizations of a computer code. Each realization $y(\boldsymbol{x})$ of the computer code output corresponds to a d-dimensional input vector $\boldsymbol{x} = (x_1, ..., x_d)$. The n points corresponding to the code runs are called an experimental design and are denoted as $X_s = (\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(n)})$. The outputs will be denoted as $Y_s = (y^{(1)}, ..., y^{(n)})$ with $y^{(i)} = y(\boldsymbol{x}^{(i)}), i = 1, ..., n$. Gaussian process (Gp) modeling treats the deterministic response $y(\boldsymbol{x})$ as a realization of a random function $Y(\boldsymbol{x})$, including a regression part and a centered stochastic process. This model can be written as:

$$Y(\boldsymbol{x}) = f(\boldsymbol{x}) + Z(\boldsymbol{x}). \tag{1}$$

The deterministic function $f(\boldsymbol{x})$ provides the mean approximation of the computer code. Our study is limited to the parametric case where the function f is a linear combination of elementary functions. Under this assumption, $f(\boldsymbol{x})$ can be written as follows:

$$f(\boldsymbol{x}) = \sum_{j=0}^{k} \beta_j f_j(\boldsymbol{x}) = F(\boldsymbol{x})\boldsymbol{\beta},$$

where $\boldsymbol{\beta} = [\beta_0, \ldots, \beta_k]^t$ is the regression parameter vector and $F(\boldsymbol{x}) = [f_0(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x})]$ is the regression matrix, with each f_j $(j = 0, \ldots, k)$ an elementary function. In the case of the one-degree polynomial regression, $(d + 1)$ elementary functions are used:

$$\begin{cases} f_0(\boldsymbol{x}) = 1, \\ f_i(\boldsymbol{x}) = x_i \quad \text{for} \quad i = 1, \ldots, d. \end{cases}$$

In the following, we use this one-degree polynomial for the regression part, while our methodology can be extended to other bases of regression functions. The regression part allows the addition of an external drift. Without prior information on the relation between the model output and the input variables, this quite simple choice appears reasonable. Indeed, adding this simple external drift allows for a nonstation-

ary global model even if the stochastic part Z is a stationary process. Moreover, on our tests of section 4, this simple model does not affect our prediction performance. This simplification is also reported by Sacks et al. (1989).

The stochastic part $Z(\boldsymbol{x})$ is a Gaussian centered process fully characterized by its covariance function: $\mathrm{Cov}(Z(\boldsymbol{x}), Z(\boldsymbol{u})) = \sigma^2 R(\boldsymbol{x}, \boldsymbol{u})$, where σ^2 denotes the variance of Z and R is the correlation function that provides interpolation and spatial correlation properties. To simplify, a stationary process $Z(\boldsymbol{x})$ is considered, which means that correlation between $Z(\boldsymbol{x})$ and $Z(\boldsymbol{u})$ is a function of the difference between $\boldsymbol{x}$ and $\boldsymbol{u}$. Our study is focused on a particular family of correlation functions that can be written as a product of one-dimensional correlation functions:

$$\mathrm{Cov}(Z(\boldsymbol{x}), Z(\boldsymbol{u})) = \sigma^2 R(\boldsymbol{x} - \boldsymbol{u}) = \sigma^2 \prod_{l=1}^{d} R_l(x_l - u_l).$$

Abrahamsen (1994), Sacks et al. (1989), Chilès and Delfiner (1999) and Rasmussen and Williams (2006) give lists of correlation functions with their advantages and drawbacks. Among all these functions, we choose to use the generalized exponential correlation function:

$$R_{\boldsymbol{\theta},\boldsymbol{p}}(\boldsymbol{x} - \boldsymbol{u}) = \prod_{l=1}^{d} \exp(-\theta_l |x_l - u_l|^{p_l}) \text{ with } \theta_l \geq 0 \text{ and } 0 < p_l \leq 2,$$

where $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_d]^t$ and $\boldsymbol{p} = [p_1, \ldots, p_d]^t$ are the correlation parameters. Our motivations stand on the derivation and regularity properties of this function. Moreover, different choices of covariance parameters allow a wide spectrum of possible shapes (Figure 1); $p = 1$ gives the exponential correlation function and $p = 2$ the Gaussian correlation function.

Even for deterministic computational codes (i.e. outputs corresponding to the same inputs are identical), the outputs may be subject to noise (e.g. numerical noise). In this case, an independent white noise $U(\boldsymbol{x})$ is added in the stochastic part of the model:

$$Y(\boldsymbol{x}) = f(\boldsymbol{x}) + Z(\boldsymbol{x}) + U(\boldsymbol{x}), \tag{2}$$

where $U(\boldsymbol{x})$ is a centered Gaussian variable with variance $\varepsilon^2 = \sigma^2 \tau$. In terms of covariance function, this white noise introduces a discontinuity at the origin called

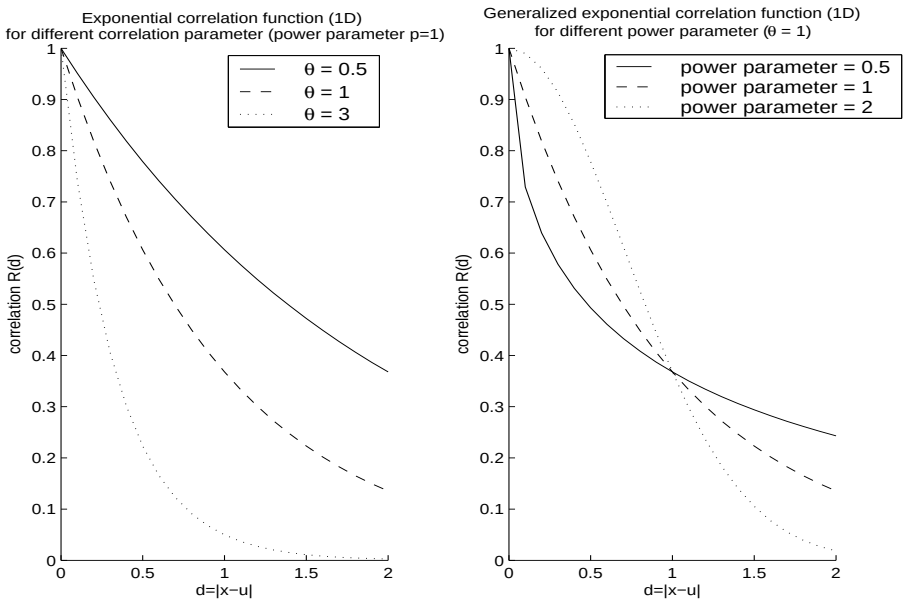

Figure 1: Generalized exponential correlation function for different power and correlation parameters.

the nugget effect (Matheron (1970)):

$$\mathrm{Cov}(Y(\boldsymbol{x}), Y(\boldsymbol{u})) = \sigma^2 \left(R_{\boldsymbol{\theta},\boldsymbol{p}}(\boldsymbol{x} - \boldsymbol{u}) + \tau\delta(\boldsymbol{x} - \boldsymbol{u}) \right),$$

where $\delta(\boldsymbol{v}) = \begin{cases} 1 \text{ if } \boldsymbol{v} = 0, \\ 0 \text{ otherwise.} \end{cases}$

2.2 Joint and conditional distributions

Under the hypothesis of a Gp model, the learning sample Y_s follows the multivariate normal distribution

$$p(Y_s | X_s, \boldsymbol{\beta}, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau) = \mathcal{N}\left(F_s\boldsymbol{\beta}, \boldsymbol{\Sigma}_s\right),$$

where $F_s = [F(\boldsymbol{x}^{(1)})^t, \ldots, F(\boldsymbol{x}^{(n)})^t]^t$ is the regression matrix and

$$\boldsymbol{\Sigma}_s = \sigma^2 \left(R_{\boldsymbol{\theta},\boldsymbol{p}} \left(\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)} \right)_{i,j=1\ldots n} + \tau \boldsymbol{I}_n \right)$$

is the covariance matrix with $\boldsymbol{I}_n$ the n-dimensional identity matrix.

If a new point $\boldsymbol{x}^* = (x_1^*, ..., x_d^*)$ is considered, the joint probability distribution of $(Y_s, Y(\boldsymbol{x}^*))$ is :

$$p(Y_s, Y(\boldsymbol{x}^*)|X_s, \boldsymbol{x}^*, \boldsymbol{\beta}, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau) = \mathcal{N}\left(\begin{bmatrix} F_s \\ F(\boldsymbol{x}^*) \end{bmatrix} \boldsymbol{\beta}, \begin{bmatrix} \boldsymbol{\Sigma}_s & \boldsymbol{k}(\boldsymbol{x}^*) \\ \boldsymbol{k}(\boldsymbol{x}^*)^t & \sigma^2(1+\tau) \end{bmatrix}\right), \quad (3)$$

with

$$\begin{aligned} \boldsymbol{k}(\boldsymbol{x}^*) &= (\ \mathrm{Cov}(y^{(1)}, Y(\boldsymbol{x}^*)), \ldots, \mathrm{Cov}(y^{(n)}, Y(\boldsymbol{x}^*))\ )^t \\ &= \sigma^2(\ R_{\boldsymbol{\theta},\boldsymbol{p}}(\boldsymbol{x}^{(1)}, \boldsymbol{x}^*) + \tau\delta(\boldsymbol{x}^{(1)}, \boldsymbol{x}^*), \ldots, R_{\boldsymbol{\theta},\boldsymbol{p}}(\boldsymbol{x}^{(n)}, \boldsymbol{x}^*) + \tau\delta(\boldsymbol{x}^{(n)}, \boldsymbol{x}^*)\ )^t. \end{aligned} \quad (4)$$

By conditioning this joint distribution on the learning sample, we can readily obtain the conditional distribution of $Y(\boldsymbol{x}^*)$ which is Gaussian (von Mises (1964)):

$$\begin{aligned} &p(Y(\boldsymbol{x}^*)|Y_s, X_s, \boldsymbol{x}^*, \boldsymbol{\beta}, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau) \\ &= \mathcal{N}\left(I\!E[Y(\boldsymbol{x}^*)|Y_s, X_s, \boldsymbol{x}^*, \boldsymbol{\beta}, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau], \mathrm{Var}[Y(\boldsymbol{x}^*)|Y_s, X_s, \boldsymbol{x}^*, \boldsymbol{\beta}, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau]\right), \end{aligned} \quad (5)$$

with

$$\begin{cases} I\!E[Y(\boldsymbol{x}^*)|Y_s, X_s, \boldsymbol{x}^*, \boldsymbol{\beta}, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau] = F(\boldsymbol{x}^*)\boldsymbol{\beta} + \boldsymbol{k}(\boldsymbol{x}^*)^t\boldsymbol{\Sigma}_s^{-1}(Y_s - F_s\boldsymbol{\beta}), \\ \mathrm{Var}[Y(\boldsymbol{x}^*)|Y_s, X_s, \boldsymbol{x}^*, \boldsymbol{\beta}, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau] = \sigma^2(1+\tau) - \boldsymbol{k}(\boldsymbol{x}^*)^t\boldsymbol{\Sigma}_s^{-1}\boldsymbol{k}(\boldsymbol{x}^*). \end{cases} \quad (6)$$

The conditional mean (equation (6)) is used as a predictor. The variance formula corresponds to the mean squared error (MSE) of this predictor and is also known as the kriging variance. This analytical formula for MSE gives a local indicator of the prediction accuracy. More generally, Gp model provides an analytical formula for the distribution of the output variable at an arbitrary new point. This distribution formula can be used for sensitivity and uncertainty analysis, as well as for quantile evaluation (O'Hagan (2006)). Its use can be completely or partly analytical and avoids costly methods based for example on a Monte Carlo algorithm. The variance expression can also be used in sampling strategies (Scheidt and Zabalza-Mezghani (2004)). All these considerations and possible extensions of Gp modeling represent significant advantages (Currin et al. (1991), Rasmussen and Williams (2006)).

2.3 Parameter estimation

To compute the mean and variance of a Gp model, estimation of several parameters is needed. Indeed, the Gp model (2) is characterized by the regression parameter vector $\boldsymbol{\beta}$, the correlation parameters $(\boldsymbol{\theta}, \boldsymbol{p})$ and the variance parameters (σ^2, τ). The maximum likelihood method is commonly used to estimate these parameters. Given a Gp model, the log-likelihood of Y_s can be written as:

$$\begin{aligned} l_{Y_s}(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{p}, \sigma, \tau) = & -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\sigma^2) - \frac{1}{2}\ln(\det(\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}} + \tau \boldsymbol{I}_n)) \\ & -\frac{1}{2\sigma^2}(Y_s - F_s\boldsymbol{\beta})^t(\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}} + \tau \boldsymbol{I}_n)^{-1}(Y_s - F_s\boldsymbol{\beta}). \end{aligned}$$

Given the correlation parameters $(\boldsymbol{\theta}, \boldsymbol{p})$ and the variance parameter τ, the maximum likelihood estimator of $\boldsymbol{\beta}$ is the generalized least squares estimator:

$$\hat{\boldsymbol{\beta}} = ({F_s}^t(\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}} + \tau \boldsymbol{I}_n)^{-1}F_s)^{-1} \ \ {F_s}^t(\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}} + \tau \boldsymbol{I}_n)^{-1}Y_s, \tag{7}$$

and the maximum likelihood estimator of σ^2 is:

$$\widehat{\sigma^2} = \frac{1}{n}(Y_s - F_s\hat{\boldsymbol{\beta}})^t(\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}} + \tau \boldsymbol{I}_n)^{-1}(Y_s - F_s\hat{\boldsymbol{\beta}}). \tag{8}$$

Remark 2.1 If we consider the predictor built on the conditional mean (equation (6)), we replace $\boldsymbol{\beta}$ by its estimator $\widehat{\boldsymbol{\beta}}$. The predictor writes now

$$\widehat{Y(\boldsymbol{x}^*)}_{|Y_s, X_s, \boldsymbol{x}^*, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau} = F(\boldsymbol{x}^*)\widehat{\boldsymbol{\beta}} + \boldsymbol{k}(\boldsymbol{x}^*)^t\boldsymbol{\Sigma}_s^{-1}(Y_s - F_s\widehat{\boldsymbol{\beta}})$$

and its MSE has consequently an additional component (Santner et al. (2003)):

$$Var[\widehat{Y(\boldsymbol{x}^*)}|Y_s, X_s, \boldsymbol{x}^*, \sigma, \boldsymbol{\theta}, \boldsymbol{p}, \tau] = \sigma^2(1+\tau) - \boldsymbol{k}(\boldsymbol{x}^*)^t\boldsymbol{\Sigma}_s^{-1}\boldsymbol{k}(\boldsymbol{x}^*) + u(\boldsymbol{x}^*)({F_s}^t\boldsymbol{\Sigma}_s^{-1}F_s)^{-1}u(\boldsymbol{x}^*)^t$$

with $u(\boldsymbol{x}^*) = F(\boldsymbol{x}^*) - \boldsymbol{k}(\boldsymbol{x}^*)^t\boldsymbol{\Sigma}_s^{-1}F_s$.

Matrix $\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}}$ depends on $\boldsymbol{\theta}$ and $\boldsymbol{p}$. Consequently, $\hat{\boldsymbol{\beta}}$ and $\widehat{\sigma^2}$ depend on $\boldsymbol{\theta}$, $\boldsymbol{p}$ and τ. Substituting $\hat{\boldsymbol{\beta}}$ and $\widehat{\sigma^2}$ into the log-likelihood, we obtain the optimal choice $(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{p}}, \widehat{\tau})$ which maximizes:

$$\phi(\boldsymbol{\theta}, \boldsymbol{p}, \tau) = -\frac{1}{2}\left[n\ln(\widehat{\sigma^2}) + \ln(|\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}} + \tau \boldsymbol{I}_n|)\right] \text{ where } |\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}} + \tau \boldsymbol{I}_n| = \det(\boldsymbol{R}_{\boldsymbol{\theta},\boldsymbol{p}} + \tau \boldsymbol{I}_n).$$

Thus, estimation of $(\boldsymbol{\theta}, \boldsymbol{p})$ and τ consists in numerical optimization of the function ψ defined as follows:

$$(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{p}}, \widehat{\tau}) = \underset{\boldsymbol{\theta}, \boldsymbol{p}, \tau}{\arg\min}\, \psi(\boldsymbol{\theta}, \boldsymbol{p}, \tau) \text{ with } \psi(\boldsymbol{\theta}, \boldsymbol{p}, \tau) = |\boldsymbol{R}_{\boldsymbol{\theta}, \boldsymbol{p}} + \tau \boldsymbol{I}_n|^{\frac{1}{n}} \widehat{\sigma^2}.$$

Our study is focused on complex cases with large dimensions d for the input vector $\boldsymbol{x}$ ($d = 20$ in our second example in section 4), where the sampling design has not been chosen as a uniform grid. In this setting, minimizing function $\psi(\boldsymbol{\theta}, \boldsymbol{p}, \tau)$ is an optimization problem that is numerically costly and hard to solve. Several difficulties guide the choice of the algorithm. First, a large number of parameters imposes the use of a sequential algorithm, where different parameters are introduced step by step. Second, a large parameter domain due to the number of parameters and the lack of prior bounds requires an exploratory algorithm able to explore the domain in an optimal way. Finally, the observed irregularities of $\psi(\boldsymbol{\theta}, \boldsymbol{p}, \tau)$ due, for instance, to a conditioning problem induce local minima, which recommend the use of a stochastic algorithm rather than a descent algorithm.

Several algorithms have been proposed in previous papers. Welch et al. (1992) use the simplex search method and introduce a kind of forward selection algorithm in which correlation parameters are added step by step to reduce function $\psi(\boldsymbol{\theta}, \boldsymbol{p}, \tau)$. In Kennedy and O'Hagan's GEM-SA software (O'Hagan (2006)), which uses the Bayesian formalism, the posterior distribution of hyperparameters is maximized via a conjugate gradient method (the Powel method is used as the numerical recipe). The DACE Matlab free toolbox (Lophaven et al. (2002)) introduces a powerful stochastic algorithm based on the Hooke & Jeeves method (Bazaraa et al. (1993)), which unfortunately requires a starting point and some bounds to constrain the optimization. In complex applications, Welch's algorithm reveals some limitations and for high dimensional input, GEM-SA and DACE software cannot be applied directly on data including all the input variables. To solve this problem, we propose a sequential version (inspired by Welch's algorithm) of the DACE algorithm. It is based on the step by step inclusion of input variables (previously sorted). Our methodology allows progressive parameter estimation by input variables selection both in the regression part and in the covariance function. The complete description of this methodology is the subject of the next section.

Remark 2.2 One of the problems we have to acknowledge in the evaluation of $\psi(\boldsymbol{\theta}, \boldsymbol{p}, \tau)$ *is the condition number of the prior covariance matrix. This condition number affects the numerical stability of the linear system for the* $\hat{\boldsymbol{\beta}}$ *determination and for the evaluation of the determinant. The degree of ill-conditioning not only depends on sampling design but is also sensitive to the underlying covariance model. Ababou et al. (1994) showed, for example, that a Gaussian covariance* ($p = 2$) *implies an ill-conditioned covariance matrix (which leads to a numerically unstable system), while an exponential covariance* ($p = 1$) *gives more stability. Moreover, in our case, the experimental design cannot be chosen and numerical parameter estimation is often damaged by the ill-conditioning problem. The nugget effect represented by* τ *solves this problem. Although the outputs of the learning sample are no longer interpolated, this nugget effect improves the correlation matrix condition number and increases robustness of our estimation algorithm.*

3 MODELING METHODOLOGY

Let us first detail the procedure used to validate our model. Since the Gp predictor is an exact interpolator (except when a nugget effect is included), residuals of the learning data cannot be used directly. So, to estimate the mean squared error in a non-optimistic way, we use either a K-fold cross validation procedure (Hastie et al. (2002)) or a test sample (consisting of new data, unused in the building process of the Gp model). In both cases, the predictivity coefficient Q_2 is computed. Q_2 corresponds to the classical coefficient of determination R^2 for a test sample, i.e. for prediction residuals:

$$Q_2(Y, \hat{Y}) = 1 - \frac{\sum_{i=1}^{n_{test}} \left(Y_i - \hat{Y}_i\right)^2}{\sum_{i=1}^{n_{test}} \left(\bar{Y} - Y_i\right)^2},$$

where Y denotes the n_{test} observations of the test set and $\bar{Y}$ is their empirical mean. $\hat{Y}$ represents the Gp model predicted values, i.e. the conditional mean (equation (6)) computed which the estimated values of parameters $(\widehat{\boldsymbol{\beta}}, \widehat{\sigma}, \widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{p}}, \widehat{\tau})$. Other simple validation criteria can be used: the absolute error, the mean and standard deviation of the relative residuals, ... (see, for example, Kleijnen and Sargent (2000)), which are all global measures. Some statistical and graphical analyses of residuals can provide

more detailed diagnostics.

Our methodology consists in seven successive steps. A formal algorithmic definition is specified for each step. For $i = 1, \ldots, d$, let e_i denote the i^{th} input variable. $\mathcal{M}_0 = \left\{e_1^{(0)}, \ldots, e_d^{(0)}\right\}$ denotes the complete initial model (i.e. all the inputs in their initial ranking). $\mathcal{M}_1 = \left\{e_1^{(1)}, \ldots, e_d^{(1)}\right\}$ and $\mathcal{M}_2 = \left\{e_1^{(2)}, \ldots, e_d^{(2)}\right\}$ refer to the inputs in new rankings after sorting by different criteria (correlation coefficient or variation of Q_2). Finally, $\mathcal{M}_{cov}$ and $\mathcal{M}_{reg}$ denote the current covariance model and the current regression model; i.e. the list of selected inputs appearing in the covariance and regression functions.

Step 0 - Standardization of input variables

The appropriate procedure to construct a metamodel requires space filling designs with good optimality and orthogonality properties (Fang et al. (2006)). However, we are not always able to choose the experimental design, especially in industrial studies when the data have been generated a long time ago. Furthermore, other restrictions can be imposed; for example, a sampling design taking into account the prior distribution of input variables. This can have prejudicial consequences for hyperparameter estimation and metamodel quality.

So, to increase the robustness of our parameter estimation algorithm and to optimize the metamodel quality, we recommend to transform all the inputs into uniform variables. In order to get each transformed input variable following an uniform distribution $\mathcal{U}[0, 1]$, the theoretical distribution (if known) or the empirical ones after a piecewise linear approximation is applied to the original inputs. This approximation is required to avoid transforming a future unsampled $\boldsymbol{x}^*$ to one of the transformed training sites, even if no element of $\boldsymbol{x}^*$ is equal to the corresponding element of any of the untransformed training sites. We empirically observed that this uniform transformation of the inputs seems well adapted to correctly estimate correlation parameters. Choices of bounds and starting points are also simplified by this standardization.

Step 1 - Initial input variables ranking by decreasing coefficient of correlation between e_i and Y

Sorting input variables is necessary to reduce the number of possible models, especially to dissociate regression and covariance models. Furthermore, direct estimation

of all parameters without an efficient starting point gives bad results. So, as a sort criterion, we choose the coefficient of correlation between the input variable and the output variable under consideration. The correlation coefficients between the input parameters and the output variable are the simplest measures of the influence of inputs on the output (Saltelli et al. (2000)). They are valid in the linear relation context, while in the nonlinear context, they give a first idea of the hierarchy among input variables, in terms of their influence on the output. Finally, this simple and intuitive choice does not require any modeling and appears a good initial method to sort the inputs when no other information is available.

For a strongly nonlinear computer code, it could be interesting to use a qualitative method, independent of the model complexity, in order to sort the inputs by influence order (Helton et al. (2006)). Another possibility would be to fit an initial Gp model with an intercept only regression part and all components of $\boldsymbol{p}$ equal to 1 or 2. Only the correlation coefficients vector $\boldsymbol{\theta}$ has to be estimated. Then, sensitivity measures such as the Sobol indices (Saltelli et al. (2000), Volkova et al. (2008)) are computed and used to sort the inputs by influence order.

Algorithm

$$\mathcal{M}_0 = \left\{e_1^{(0)}, \ldots, e_d^{(0)}\right\} \Longrightarrow \mathcal{M}_1 = \left\{e_1^{(1)}, \ldots, e_d^{(1)}\right\}$$

$$\begin{cases} \mathcal{M}_{reg} = \mathcal{M}_1 \\ \mathcal{M}_{cov} = \mathcal{M}_1 \end{cases}$$

Step 2 - Initialization of the correlation parameter bounds and starting points for the estimation procedure

To constrain the ψ optimization, the DACE estimation procedure requires three following values for each correlation parameter: a lower bound, an upper bound and a starting point. These values are crucial for the success of the estimation algorithm, when it is used directly for all the input variables. However, using sequential estimation based on progressive introduction of input variables, we limit the problems associated with these three values, especially with the starting point value. Another way to reduce the importance of starting point and bounds is to increase the number of iterations in DACE estimation algorithm. However, in the case of a high number of inputs, increasing the number of iterations in DACE can become extremely time expensive; a compromise has to be found. As the input variables have been previously

transformed into standardized uniform variables, the initialization and the bounds of the correlation parameters can be the same for all the inputs:

- ◇ lower bounds for each component of $\boldsymbol{\theta}$ and $\boldsymbol{p}$: $lob_{\boldsymbol{\theta}} = 10^{-8}$, $lob_{\boldsymbol{p}} = 0$,
- ◇ upper bounds for each component of $\boldsymbol{\theta}$ and $\boldsymbol{p}$: $upb_{\boldsymbol{\theta}} = 100$, $upb_{\boldsymbol{p}} = 2$,
- ◇ starting points for estimation of each component of $\boldsymbol{\theta}$ and $\boldsymbol{p}$: $\theta^0 = 0.5$, $p^0 = 1$.

Step 3 - Successive inclusion of input variables in the covariance function

For each set of inputs included in the covariance function, all the inputs from the ordered set in the regression function are evaluated. Correlation and regression parameters are estimated by the DACE modified algorithm, with the values, estimated at the $(i-1)^{th}$ step for the same regression model, used as a starting point. More precisely, at step i, input variables numbered from 1 to i are included in the covariance function and the algorithm estimates pairs of the correlation parameters (θ_l, p_l) for $l = 1, \ldots, i$. As the starting point, the algorithm uses correlation parameters obtained at the $(i-1)^{th}$ step for the starting values of $((\theta_1, p_1), \ldots, (\theta_{i-1}, p_{i-1}))$. First starting value of (θ_i, p_i) is fixed to an arbitrary reference value. Then, at each step, selection of variables in the regression part is also made.

Hoeting et al. (2006) recommends the corrected Akaike information criterion (AICC) for input selection in the regression model in order to take spatial correlations into account. Therefore, after the estimation of correlation and regression parameters, the AICC is computed:

$$\text{AICC} = -2l_{Y_s}\left(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}, \hat{\sigma}\right) + 2n\frac{m_1 + m_2 + 1}{n - m_1 - m_2 - 2},$$

where m_1 denotes the number of input variables in the regression function, m_2 those in the covariance function and l_Y the log-likelihood of the sample Y. The required model is the one minimizing this criterion.

Algorithm

For $i = 1 \ldots d$

- ◇ Step 3.1: Variables in covariance function

  $\mathcal{M}_{i,cov} = \mathcal{M}_{cov}(1, \ldots, i)$

◇ Step 3.2: Successive inclusion of input variables in regression function

For $j = 1 \dots d$

- Regression Model:
  $\mathcal{M}_{j,reg} = \mathcal{M}_{reg}(1, \dots, j)$
- Parameter estimation:
  $\boldsymbol{\theta}^{init} = ({\theta_1}^{(i-1),j}, \dots, {\theta_{i-1}}^{(i-1),j}, \theta^0)^t$
  $\boldsymbol{p}^{init} = ({p_1}^{(i-1),j}, \dots, {p_{i-1}}^{(i-1),j}, p^0)^t$
  $[\theta^{i,j}, p^{i,j}] = \text{DACE estimation}(\mathcal{M}_{i,cov}, \mathcal{M}_{j,reg}, [\boldsymbol{\theta}^{init}, \boldsymbol{p}^{init}], [lob_{\boldsymbol{\theta}}, lob_{\boldsymbol{p}}], [upb_{\boldsymbol{\theta}}, upb_{\boldsymbol{p}}])$
- AICC Criterion computation
  $\text{AICC}(i, j) = \text{AICC}(\mathcal{M}_{i,cov}, \mathcal{M}_{j,reg})$

End

◇ Step 3.3: Optimal regression model selection:
$j^{optim}(i) = \arg\min_j (\text{AICC}(i, j))$

◇ Step 3.4: Q_2 evaluation by K-fold cross validation or on test data (with current correlation model and optimal regression model)
$Q_2(i) = Q_2(\mathcal{M}_{i,cov}, \mathcal{M}_{j^{optim}(i),reg})$

End

This order (correlation outer, regression inner) can be justified by minimizing the computer time required for optimization. The selection procedure for the regression part is made by the minimization of AICC criterion which requires, at each step, only one parameter estimation. On the other hand, the covariance selection is made by the maximization of Q_2 which is often computed by a K-fold cross validation. This procedure requires, at each step, K estimation procedures. So, the loop on covariance selection is the more expensive, and consequently has to be outer. The choice of K depends on the number of observations of the data set, on the constraints in term of computer time and on the influence of the learning sample size on prediction quality. If few data are available, a leave-one-out cross-validation could be preferred to a K-fold procedure to avoid an undesirably negative effect of small learning sets on prediction quality.

Remark 3.1 To avoid some biases on the choice of the optimal covariance model in the next two steps, the coefficient Q_2 has to be computed on a test sample (or by a

cross validation procedure), different from the one used for the final validation of the Gp model at step 7.

Other criteria often used in the optimization of the computer experiment designs (Sacks et al. (1989), Santner et al. (2003)) could be considered to select the optimal regression and covariance model. These criteria are based on the variance of Gp model: they produce a model that minimizes the maximum or the integral of predictive variance over input space. However, in the case of a high number of inputs, the optimization of these criteria can be very computer time expensive. The advantage of the Q_2 statistic is its relatively fast evaluation, while producing a final model that optimizes the predictive performance.

Step 4 (optional) - New input variables ranking in the covariance function based on the evolution of Q_2 (inputs sorted by decreasing "jumps" of Q_2)

This optional step improves the selection of inputs, particularly in the covariance function. For each input X_i, the increase of the Q_2 coefficient (denoted $\Delta Q_2(i)$) is computed when this i^{th} variable is added to the covariance function. This value is an indicator of the contribution of the i^{th} input to the accuracy of the Gp model. For this reason, it can be judicious to use values $\Delta Q_2(1), \ldots, \Delta Q_2(d)$ to sort the inputs included in the correlation function. The inputs are sorted by decreasing of values $\Delta Q_2(i)$ and the procedure of parameter estimation is repeated with this new ranking of inputs for the covariance function (step 3 is rerun).

Algorithm

- Evaluation of Q_2 increase for each input variable included in the covariance function:
  $\Delta Q_2(k) = Q_2(1)$
  For $k = 2 \ldots d$
  $\quad \Delta Q_2(k) = Q_2(k) - Q_2(k-1)$
  end
- Sorting input variables by decreasing of ΔQ_2
  $\mathcal{M}_1 \Longrightarrow \mathcal{M}_2$

Step 5 (optional) - Algorithm for parameter estimation with new ranking

of input variables in the covariance function

This optional step improves the selection of inputs, particularly in the covariance function. The procedure of parameter estimation (step 3) is repeated with the inputs sorted by decreasing values of $\Delta Q_2(i)$ in the covariance function. Consequently, correlation parameters related to the inputs that are the most influential for the increase of the Gp model accuracy are estimated in the first place. Furthermore, we can also hope that the use of this new ranking allows a decrease in the number of inputs included in the covariance function and an optimal input selection. The use of this new ranking appears more judicious and justifiable for the covariance function than sorting by decreasing correlation coefficient (cf. step 1). However, the ranking of step 1 is kept for the regression function.

Algorithm

$$\begin{cases} \mathcal{M}_{reg} = \mathcal{M}_1 \\ \mathcal{M}_{cov} = \mathcal{M}_2 \end{cases}$$

Step 6 - Optimal covariance model selection

For each set of inputs in the covariance function, the optimal regression model is selected based on minimization of the AICC criterion (cf. step 3.3). Then, the predictivity coefficient Q_2 is computed either by cross validation or on a test sample (cf. step 3.4). Finally, the selected covariance model is the one corresponding to the highest Q_2 value.

Algorithm

$$i^{optim} = \arg\max_i \left(Q_2(i)\right)$$

$$\begin{cases} \mathcal{M}_{cov}^{\text{optim}} = \mathcal{M}_{cov}(1, \ldots, i^{optim}) \\ \mathcal{M}_{reg}^{\text{optim}} = \mathcal{M}_{reg}(1, \ldots, j^{optim}(i^{optim})) \end{cases}$$

Step 7 - Final validation of the optimal Gp model

After building and selecting the optimal Gp model, a final validation is necessary to evaluate the predictive performance and to eventually compare it to other metamodels. To do this, coefficient Q_2 is evaluated on a new test sample (i.e. data not used in the building procedure). If only few data are available, a cross validation procedure can be considered. So, two cross validation procedures are overlapped; one

for building the model and one for its validation.

Algorithm

$Q_2^{final} = Q_2(\mathcal{M}_{cov}^{\text{optim}}, \mathcal{M}_{reg}^{\text{optim}})$

After all the steps of our algorithm (including the step 5), we can often link the inputs appearing in the covariance and regression functions with the nature of their effects on the output. Indeed, we can generally observed 4 cases: the inputs with only a linear effect which are supposed to appear only in the regression and excluded from the covariance with the step 5, the inputs with only a non-linear effect which are excluded from the regression and can then appear in the covariance with the re-ordering of $\mathcal{M}_{cov}$ at step 5, the inputs with both effects appearing in the regression and covariance functions and, finally, the inactive input variables excluded from both.

4 APPLICATIONS

4.1 Analytical test case

First, an analytical function called the g-function of Sobol is used to illustrate and justify our methodology. The g-function of Sobol is defined for d inputs uniformly distributed on $[0,1]^d$:

$$g_{\text{Sobol}}(X_1, \ldots, X_d) = \prod_{k=1}^{d} g_k(X_k) \text{ where } g_k(X_k) = \frac{|4X_k - 2| + a_k}{1 + a_k} \text{ and } a_k \geq 0.$$

Because of its complexity (strongly nonlinear and non-monotonic relationship) and the availability of analytical sensitivity indices, the g-function of Sobol is a well known test example in the studies of global sensitivity analysis algorithms (Saltelli et al. (2000)). The contribution of each input X_k to the variability of the model output is represented by the weighting coefficient a_k. The lower this coefficient a_k,

the more significant the variable X_k. For example:

$$\begin{cases} a_k = 0 \rightarrow X_k \text{ is very important,} \\ a_k = 1 \rightarrow X_k \text{ is relatively important,} \\ a_k = 9 \rightarrow X_k \text{ is non important,} \\ a_k = 99 \rightarrow X_k \text{ is non significant.} \end{cases}$$

For our analytical test, we choose $a_k = k$.

Applying our methodology to the g-function of Sobol, we illustrate its different steps, especially the importance of rerunning the estimation procedure after sorting the inputs by decreasing ΔQ_2 (cf. steps 4 and 5). At the same time, comparisons are made with other reference software like, for example, the GEM-SA software (O'Hagan (2006), freely available at *http://www.ctcd.group.shef.ac.uk/gem.html*).

To do this, different dimensions of inputs are considered, from 4 to 20: $d = 4, 6, \ldots, 20$. For each dimension d, we generate a learning sample formed by $N_{LS} = d \times 10$ simulations of the g-function of Sobol following the Latin Hypercube Sampling (LHS) method (McKay et al. (1979)). Using these learning data, two Gp models are built: one following our methodology and one using the GEM-SA software. For each method, the Q_2 coefficient is computed on a test sample of $N_{TS} = 1000$ points. For each dimension d, this procedure is repeated 50 times to obtain an average performance in terms of the prediction capabilities of each method (mean of Q_2). The standard deviation of Q_2 is also a good indicator of the robustness of each method.

For each dimension d, the mean and standard deviation of Q_2 computed on the test sample using different methods are presented in Table 1. Three methods are compared: the GEM-SA software, our methodology without steps 4 and 5, and our methodology with steps 4 and 5.

For the values of d higher than 6, our methodology including double selection of inputs (with steps 4 and 5) clearly outperforms the others. More precisely, the pertinence of rerunning the estimation procedure after sorting the inputs by decreasing ΔQ_2 is obvious. The prediction accuracy is much more robust (lower standard deviation of Q_2).

The drawback of our methodology lies in the somewhat costly steps 4 and 5.

g-Sobol simulations		GEM-SA software		Gp methodology without steps 4 and 5		Gp methodology with steps 4 and 5	
d	N_{LS}	$\overline{Q_2}$	sd	$\overline{Q_2}$	sd	$\overline{Q_2}$	sd
4	40	0.82	0.08	0.60	0.21	0.86	0.07
6	60	0.67	0.24	0.59	0.16	0.85	0.05
8	80	0.66	0.13	0.61	0.10	0.85	0.04
10	100	0.59	0.25	0.63	0.13	0.83	0.05
12	120	0.57	0.16	0.61	0.15	0.84	0.05
14	140	0.60	0.17	0.61	0.14	0.83	0.03
16	160	0.62	0.11	0.67	0.06	0.86	0.04
18	180	0.66	0.09	0.67	0.05	0.84	0.03
20	200	0.64	0.09	0.72	0.07	0.86	0.02

Table 1: Mean $\overline{Q_2}$ and standard deviation sd of the predictivity coefficient Q_2 for several implementations of the g-function of Sobol.

Indeed, sequential estimation and rerunning of the procedure require many executions of the Hooke & Jeeves algorithm, particularly in the case of a double cross validation (cf. steps 3.4 and 7 of the algorithm). Consequently, this approach is much more computer time expensive than the GEM-SA software. For example, for a simulation with $d = 10$ and $N_{LS} = 100$, the computing time of our approach is on average ten times larger than that of the GEM-SA software.

For a practitioner, a compromise is usually made between the time to obtain the sampling design points and the time to build a metamodel. As a conclusion of this section, our methodology is interesting for high dimensional input models (more than ten), for inadequate or small sampling designs (a few hundreds) and when simpler methodologies have failed. The data presented in the next section fall into this scope.

Remark 4.1 The Gp model used in the GEM-SA software has a gaussian covariance function. Our model uses a generalized exponential correlation function even if it requires the estimation of twice as many hyperparameters. Indeed, the sequential approach allows to estimate a large number of hyperparameters.

4.2 Application on an hydrogeologic transport code

Our methodology is now applied to the data obtained from the modeling of strontium 90 (noted ^{90}Sr) transport in saturated porous media using the MARTHE software (developed by BRGM, the French Geological Survey). The MARTHE computer code models flow and transport equations in three-dimensional porous formations. In the context of an environmental impact study, this code is used to model ^{90}Sr transport in saturated media for a radwaste temporary storage site in Russia (Volkova et al. (2008)). One of the final purposes is to determine the short-term evolution of ^{90}Sr transport in soils in order to help rehabilitation decision making. Only a partial characterization of the site has been made and, consequently, values of the model input parameters are not known precisely. One of the first goals is to identify the most influential parameters of the computer code in order to improve the characterization of the site in an optimal way. Because of large computing time of the MARTHE code, Volkova et al. (2008) propose to construct a metamodel on the basis of the first learning sample. In the following, our Gp methodology is applied and its results are compared to the previous ones obtained with boosting regression trees and linear regression.

4.2.1 Data presentation

Data simulated in this study are composed of 300 observations. Each simulation consists of 20 inputs and 20 outputs. The 20 uncertain model parameters are permeability of different geological layers composing the simulated field (parameters 1 to 7), longitudinal dispersivity coefficients (parameters 8 to 10), transverse dispersivity coefficients (parameters 11 to 13), sorption coefficients (parameters 14 to 16), porosity (parameter 17) and meteoric water infiltration intensities (parameters 18 to 20). To study sensitivity of the MARTHE code to these parameters, simulations of these 20 parameters have been made by the LHS method.

For each simulated set of parameters, MARTHE code computes transport equations of ^{90}Sr and predicts the evolution of ^{90}Sr concentration. Initial and boundary conditions for the flow and transport models are fixed at the same values for all simulations. So, for an initial map of ^{90}Sr concentration in 2002 and a set of 20 input parameter values, MARTHE code computes a map of predicted concentrations

in 2010. For each simulation, the 20 outputs considered are values of ^{90}Sr concentration, predicted for year 2010, in 20 piezometers located on the waste repository site.

4.2.2 Comparison of three different models

For each output, we choose to compare and analyze the results of three models:

- ▷ Linear regression: it represents a model that provides a reference for the contribution of the Gp model stochastic component to modeling quality. Indeed, comparison between simple linear regression and Gp model will show if considering spatial correlations has significant impact on the modeling results. Moreover, a selection based on the AICC criterion is implemented to optimize the results of the linear regression.
- ▷ Boosting of regression trees: this model was used in the previous study of the data (Volkova et al. (2008)). The boosting trees method is based on sequential construction of weak models (here regression trees with low interaction depth), that are then aggregated. The MART algorithm (Multiple Additive Regression Trees), described in Hastie et al. (2002), is used here. The boosting trees method is relatively complex, in the sense that, as with neural networks, it is a black box model, efficient but quite difficult to interpret. It is interesting to see if a Gp model, that is easier to interpret and offers a quickly computable predictor, can compete with a more complex method in terms of modeling and prediction quality. Note that the boosting trees algorithm also makes its proper input selection.
- ▷ Gaussian process: to implement this model, the methodology previously described in this paper is applied, with the input selection procedure.

4.2.3 Results

To compare prediction quality of the three different models presented above, the coefficient of predictivity Q_2 is estimated by a 6-fold cross validation. Note that for each model the results correspond to the optimal set of inputs included in the model. To avoid some bias in the results, the cross validation used to select variables in the Gp model (see step 6) differs from the cross validation used to validate and

compare prediction capabilities of the three models. Indeed, at each cross validation step (used to validate), data are divided into a learning sample (denoted $LS1$) of 250 observations and a test sample ($TS1$) of the 50 remaining observations. For the Gp model, the procedure of variable selection is then performed by a second cross validation on $LS1$ (for example: a 4-fold cross validation, dividing $LS1$ into a learning set $LS2$ of 210 data and a test set $TS2$ of the 40 others). Then, an optimal set of variables is determined and a Gp model is built based on the 250 data of $LS1$ (with this optimal set of inputs previously selected). Finally, the model is validated on the test set $TS1$ that has never been used for the Gp model construction.

The results are presented in Table 2 and are taken up in a barplot (see Figure 2). Results obtained for the output 8 (piezometer p110) are not considered because of physically insignificant concentration values. For most outputs, the Gp performance is superior to linear regression and boosting, in many cases substantially so. Concerning the outputs 11 ($p27k$) and 19 ($p4a$), the performances of the Gp model are worse than the linear regression ones. However, for these two outputs, the prediction errors are very high and consequently the difference of performance between the two models can be considered as non-significant.

As expected, for most of the outputs, the linear regression presents the worst results. When this model is successfully adapted, the two others are also efficient. When linear regression fails (for example, for output number 12), Gp model presents a real interest, since it gives results as good as those of the boosting trees method. In fact, this is verified for all the ouputs and results are significantly better for several outputs (outputs 1, 2, 4, 9, 12, 13 and 16). To illustrate this, the Figure 3 shows the predicted values vs real values for the output 16, for the Gaussian process and boosting trees models. It clearly shows a better repartition of the Gp model residuals than the boosting trees model ones.

Furthermore, the estimator of MSE, that is expressed analytically (see Equation (6)), can be used as a local prediction interval. To illustrate this, we consider 50 observations of the output 16. Figure 4 shows the observed values, the predicted values and the upper and lower bounds of the 95% prediction interval based on the MSE local estimator. It confirms the good adequacy of the Gp model for this output because all the observed values (except one point) are inside the prediction interval

Output		Linear regression	boosting trees	Gaussian process
Denomination	Number	Q_2	Q_2	Q_2
p1-76	1	0.31	0.59	0.84
p102K	2	0.48	0.64	0.78
p103	3	0.10	0.43	0.5
p104	4	0.69	0.83	0.96
p106	5	0.17	0.29	0.45
p107	6	0.40	0.78	0.86
p109	7	0.40	0.45	0.5
p2-76	9	0.19	0.58	0.86
p23	10	0.74	0.94	0.935
p27K	11	0.52	0.60	0.43
p29K	12	0.55	0.80	0.93
p31K	13	0.27	0.51	0.69
p35K	14	0.26	0.55	0.56
p36K	15	0.54	0.60	0.60
p37K	16	0.59	0.62	0.90
p38	17	0.25	0.43	0.52
p4-76	18	0.67	0.95	0.96
p4a	19	0.16	0.17	0.09
p4b	20	0.39	0.27	0.37

Table 2: Predictivity coefficients Q_2 for the three different models of the MARTHE data.

curves.

4.2.4 Analysis

These results confirm the potential of the Gp model and justify its application for computer codes. Application of our methodology to complex data also confirms the efficiency of our input selection procedure. For a fixed set of inputs in the covariance function, we can verify that this procedure selects the best set of inputs in regression part. Furthermore, the necessity of conducting sequential and ordered procedure estimation has been demonstrated. Indeed, if all the Gp parameters (i.e. considering the 20 inputs) are directly and simultaneously estimated with the DACE algorithm, they are not correctly determined and poor results in terms of Q_2 are obtained. So,

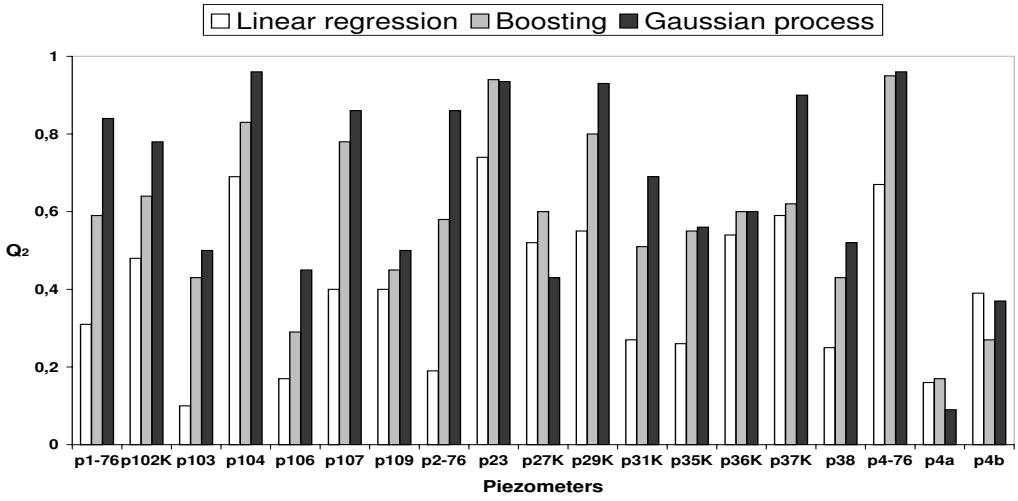

Figure 2: Barplot of the predictivity coefficient Q_2 for the three different models.

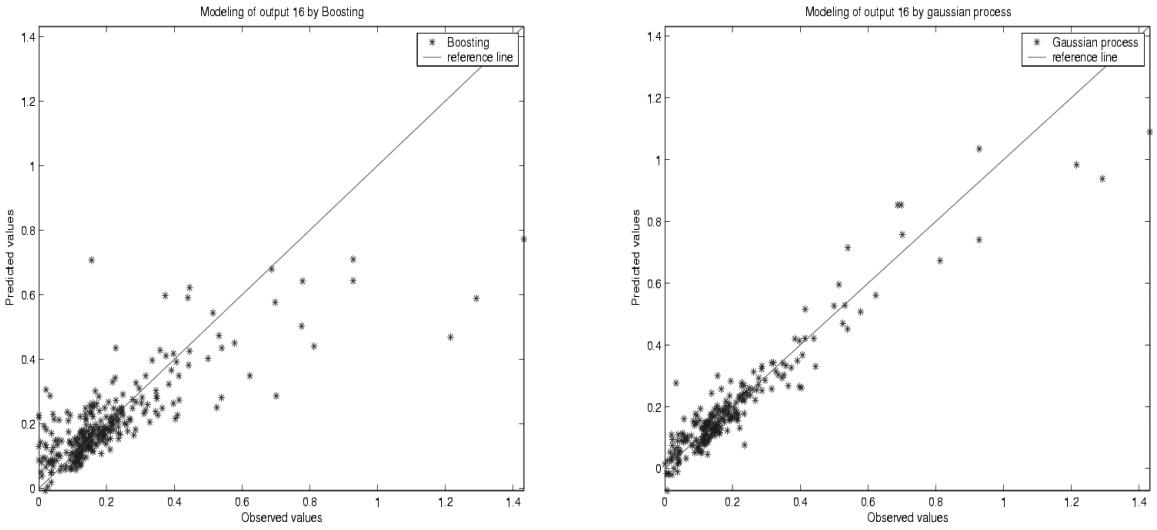

Figure 3: Plot of predicted values vs real values for boosting trees (left) and Gaussian process (right).

in case of a complex model with a large number of inputs, we recommend using a selection procedure such as the algorithm of section 3.

The study of these data have motivated the choice of this methodology. At first, Welch's algorithm (see section 2.3) has been tried. Considering the poor results

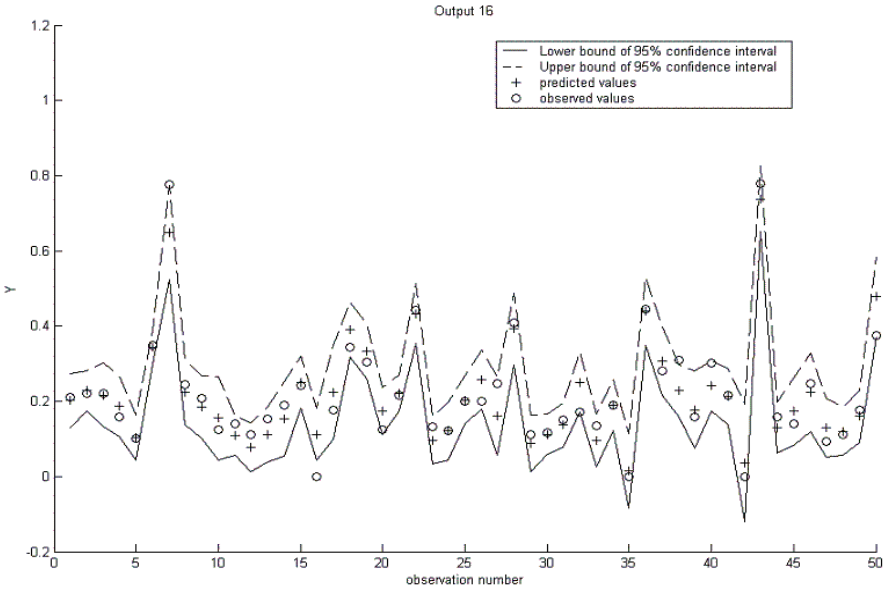

Figure 4: Plot of observed and Gaussian process predicted values for the output 16 with the 95% prediction interval based on $\widehat{MSE}$ formula.

obtained, our methodology based on the DACE estimation algorithm has been developed. To illustrate this, let us detail the different results obtained on the output number 9. With our methodology based on the DACE estimation, the Q_2 coefficient (always computed by a 6-fold cross validation) is 0.86, while with Welch's algorithm (used in its basic version), Q_2 is close to zero. The difference in the results between the two methods can be explained by the value of estimated correlation parameters which are significantly different.

To minimize the number of correlation parameters and consequently reduce computer time required for estimation, the possible values of power parameters p_i ($i = 1, \ldots, d$) can be limited to 0.5, 1 and 2. It can be a solution to optimize computer time. It allows an exhaustive, quick and optimal representation of different kinds of correlation functions (two kinds of inflexion are represented). Furthermore, in many cases, estimation of power parameter with generalized exponential correlation converges to exponential ($p_i = 1$) or Gaussian ($p_i = 2$) correlation.

5 CONCLUSION

The Gaussian process model presents some real advantages compared to other metamodels: exact interpolation property, simple analytical formulations of the predictor, availability of the mean squared error of the predictions and the proved efficiency of the model. The keen interest in this method is testified by the publication of the recent monographs of Santner et al. (2003), Fang et al. (2006) and Rasmussen and Williams (2006).

However, for its application to complex industrial problems, developing a robust implementation methodology is required. In this paper, we have outlined some difficulties arising from the parameter estimation procedure (instability, high number of parameters) and the necessity of a progressive model construction. Moreover, an a priori choice of regression function and, more important, of covariance function is essential to parameterize the Gaussian process model. The generalized exponential covariance function appears in our experience as a judicious and recommended choice. However, this covariance function requires the estimation of $2d$ correlation parameters, where d is the input space dimension. In this case, the sequential estimation and selection procedures of our methodology are more appropriate. This methodology is interesting when the computer model is rather complex (non linearities, threshold effects, etc.), with high dimensional inputs ($d > 10$) and for small size samples (a few hundreds).

Results obtained on the MARTHE computer code are very encouraging and place the Gaussian process as a good and judicious alternative to efficient but non-explicit and complex methods such as boosting trees or neural networks. It has the advantage of being easily evaluated on a new parameter set, independently of the metamodel complexity. Moreover, several statistical tools are available because of the analytical formulation of the Gaussian model. For example, the MSE estimator offers a good indicator of the model's local accuracy. In the same way, inference studies can be developed on parameter estimators and on the choice of the experimental input design. Finally, one possible improvement in our construction algorithm is based on the sequential approach of the choice of input design, which remains an active research domain (Scheidt and Zabalza-Mezghani (2004) for example).

ACKNOWLEGMENTS

This work was supported by the MRIMP project of the "Risk Control Domain" that is managed by CEA/Nuclear Energy Division/Nuclear Development and Innovation Division. We are grateful to the two referees for their comments which significantly improved the paper.

References

Ababou, R., Bagtzoglou, A., Wood, E., 1994. On the condition number of covariance matrices in kriging, estimation, and simulation of random fields. Mathematical Geology 26, 99–133.

Abrahamsen, P., 1994. A review of gaussian random fields and correlation functions. Tech. Rep. 878, Norsk Regnesentral.

Alam, F., McNaught, K., Ringrose, T., 2004. A comparison of experimental designs in the development of a neural network simulation metamodel. Simulation Modelling Practice and Theory 12, 559–578.

Bazaraa, M., Sherali, H., Shetty, C., 1993. Nonlinear programming. John Wiley & Sons, Inc.

Box, G., Draper, N., 1987. Empirical model building and response surfaces. Wiley Series in Probability and Mathematical Statistics. Wiley.

Chilès, J.-P., Delfiner, P., 1999. Geostatistics: Modeling spatial uncertainty. Wiley, New-York.

Cressie, N., 1993. Statistics for spatial data. Wiley Series in Probability and Mathematical Statistics. Wiley.

Currin, C., Mitchell, T., Morris, M., Ylvisaker, D., 1991. Bayesian prediction of deterministic functions with applications to the design and analysis of computer experiments. Journal of the American Statistical Association 86 (416), 953–963.

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

Hastie, T., Tibshirani, R., Friedman, J., 2002. The elements of statistical learning. Springer.

Helton, J., Johnson, J., Salaberry, C., Storlie, C., 2006. Survey of sampling-based methods for uncertainty and sensitivity analysis. Reliability Engineering and System Safety 91, 1175–1209.

Hoeting, J., Davis, R., Merton, A., Thompson, S., 2006. Model selection for geostatistical models. Ecological Applications 16, 87–98.

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

Jourdan, A., Zabalza-Mezghani, I., 2004. Response surface designs for scenario mangement and uncertainty quantification in reservoir production. Mathematical Geology 36 (8), 965–985.

Kleijnen, J., 1997. Sensitivity analysis and related analyses: a review of some statistical techniques. Journal of Statistical Computation and Simulation 57, 111–142.

Kleijnen, J., 2005. An overview of the design and analysis of simulation experiments for sensitivity analysis. European Journal of Operational Research 164, 287–300.

Kleijnen, J., Sargent, R., 2000. A methodology for fitting and validating metamodels in simulation. European Journal of Operational Research 120, 14–29.

Lophaven, S., Nielsen, H., Sondergaard, J., 2002. DACE - A Matlab kriging toolbox, version 2.0. Tech. Rep. IMM-TR-2002-12, Informatics and Mathematical Modelling, Technical University of Denmark, <http://www.immm.dtu.dk/∼hbn/dace>.

Matheron, G., 1970. La Théorie des Variables Régionalisées, et ses Applications. Les Cahiers du Centre de Morphologie Mathématique de Fontainebleau, Fascicule 5. Ecole des Mines de Paris.

McKay, M., Beckman, R., Conover, W., 1979. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21, 239–245.

O'Hagan, A., 2006. Bayesian analysis of computer code outputs: A tutorial. Reliability Engineering and System Safety 91, 1290–1300.

Osio, I., Amon, C., 1996. An engineering design methodology with multistage bayesian surrogates and optimal sampling. Research in Engineering Design 8, 189–206.

Rasmussen, C., Williams, C., 2006. Gaussian processes for machine learning. MIT Press.

Sacks, J., Welch, W., Mitchell, T., Wynn, H., 1989. Design and analysis of computer experiments. Statistical Science 4, 409–435.

Saltelli, A., Chan, K., Scott, E. (Eds.), 2000. Sensitivity analysis. Wiley Series in Probability and Statistics. Wiley.

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

Scheidt, C., Zabalza-Mezghani, I., August 2004. Assessing uncertainty and optimizing production schemes: Experimental designs for non-linear production response modeling. an application to early water breakthrough prevention. In: ECMOR IX. Cannes, France.

Vazquez, E., Walter, E., Fleury, G., 2005. Intrinsic kriging and prior information. Applied Stochastic Models in Business and Industry 21, 215–226.

Volkova, E., Iooss, B., Van Dorpe, F., 2008. Global sensitivity analysis for a numerical model of radionuclide migration from the RRC "Kurchatov Institute" radwaste disposal site. Stochastic Environmental Research and Risk Assesment 22, 17–31.

von Mises, R., 1964. Mathematical Theory of Probability and Statistics. Academic Press.

Welch, W., Buck, R., Sacks, J., Wynn, H., Mitchell, T., Morris, M., 1992. Screening, predicting, and computer experiments. Technometrics 34 (1), 15–25.