Méthodes de surface de réponse basées sur la décomposition de la variance fonctionnelle et application à l'analyse de sensibilité

Samir Touzani

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques Appliquées**

Arrêté ministérial : 7 août 2006

Présentée par

Samir TOUZANI

Thèse dirigée par **Anestis ANTONIADIS**

préparée au sein du **Laboratoire Jean Kuntzmann**
et de l'école Doctorale de **Mathématiques, Sciences et technologies de l'information, Informatique**

Méthodes de surface de réponse basées sur la décomposition de la variance fonctionnelle et application à l'analyse de sensibilité

Thèse soutenue publiquement le **20 Avril 2011**,
devant le jury composé de :

Mme. Clémentine PRIEUR
Professeur, Université de Grenoble, Présidente
M. Fabrice GAMBOA
Professeur, Université Paul Sabatier, Rapporteur
M. Bertrand IOOSS
Ingénieur de recherche, EDF R&D, Rapporteur
M. Alberto PASANISI
Ingénieur de recherche, EDF R&D, Examinateur
M. Daniel BUSBY
Ingénieur de recherche, IFP Energies Nouvelles, Examinateur
M. Anestis ANTONIADIS
Professeur, Université de Grenoble, Directeur de thèse

A ma mère et à la mémoire de mon père.

Remerciements

Je souhaiterais en premier lieu remercier Anestis Antoniadis, qui m'a fait l'honneur de diriger ma thèse. Je tiens à lui exprimer toute ma gratitude pour ses nombreux conseils et remarques avisés, sa confiance en mes capacités, sa bonne humeur et sa gentillesse.

Je remercie Daniel Busby, qui est à l'origine de cette thèse et a encadré mon travail à l'IFP. Je le remercie pour la confiance et la liberté qu'il m'a accordées tout au long de ces trois ans. Je tiens à lui témoigner toute ma reconnaissance pour m'avoir permis de réaliser cette aventure de plus de trois ans.

Je remercie les membres de mon jury de thèse, Clémentine Prieur, Fabrice Gamboa, Bertrand Iooss et Alberto Pasanisi, pour l'intérêt qu'ils ont porté à mon travail et leurs remarques constructives qui ont permis de l'enrichir.

Je remercie également Sébastien Da Veiga pour avoir pris le temps de répondre simplement à mes questions de statistiques les plus naïves et pour les encouragements et conseils qu'il m'a prodigués lors de nos nombreuses discussions.

Je remercie aussi Delphine Sinoquet, Hoël Langouët et Frédéric Delbos de l'équipe Mathématiques Appliquées pour leurs conseils concernant l'optimisation.

Je remercie Abdelaziz Faraj pour son soutien, ces conseils et les qualités humaines dont il a fait preuve à mon égard.

Je remercie particulièrement les membres du département de modélisation de gisement qui ont contribué de près ou de loin au bon déroulement de ce travail. Je pense à Benoit Noetinger, Mathieu Ferrail, Amandine Marrel, Mickaele Le Ravalec, Frédéric Roggero et Sylvie Hoguet.

Je ne peux passer outre dans ces remerciements tous les thésards, apprentis et stagiaires qu'il m'a été donné de rencontrer au cours de ces trois ans. Je pense en premier lieu à mes collègues de bureau, Virginie, Leïla et Alexandre pour avoir fait de cette pièce un agréable endroit où travailler. Je tiens à remercier mes compagnons de pause au bord du " lac " de l'IFP pour avoir égailler de si nombreux moments mémorables et pour leur excellente compagnie : Ekaterina, Zyed, Amine, Monsef, Fakher, Marius, David, Ratiba, Franck, François,

Romain et Erwan. Je remercie également mes autres compagnons de thèse rencontrés au cours de ces trois ans, Céline, Cristina, Salem, ainsi que tous ceux que j'oublie de citer.
Je remercie aussi mes amis Montpelliérains pour leur encouragement durant ces années.
Enfin, je remercie du fond du cœur ma mère et Camille pour leur soutien inconditionnel.

Contents

List of Figures

Chapter 1

Introduction

1.1 Problem statement

The recent significant advances in computational power have allowed computer modeling and simulation to become an integral tool in many industrial and scientific applications, such as nuclear safety assessment, meteorology and oil reservoir forecasting. Simulations are performed with complex computer codes that model diverse complex real world phenomena. However, despite a high level of sophistication of these models, they are only an approximation of the reality and as such they are also subject to uncertainty. This is due to the fact that any model relies upon some hypotheses, inferred from limited information, which can lead to neglecting some significant physical phenomena of the real system. The development of an accurate model can require gathering a large amount of information, which is usually an expensive and sometimes an impossible task.

The context of this thesis's work is oil reservoir forecasting, which consists in predicting the hydrocarbon resources and their production during the operating time of a reservoir. Such predictions are used by engineers and managers to make investment decisions in order to improve oil recovery, or to decide if starting the recovery process is economically viable. The fundamental tools to address this challenging problem is by using reservoir simulators.

Reservoir simulators are complex computer codes that model the physical laws governing the recovery process, and which are mainly modeled by mathematical equations for the three phases flow (oil, gas and water) through porous media. These mathematical equations are solved by numerical methods over discrete computational grids. In order to get to more accurate solutions the reservoir simulators involve higher number of grid cells and

an increasing number of reservoir details. The accuracy of the model predictions depends on the input data accuracy, so if there are uncertainties on input factors, the simulator forecasts will be uncertain. These input parameters are generally related to the geological properties of the reservoir, and the information gathered on them comes from direct measurements, which are clearly very limited and are marred by considerable uncertainty.
Statistical tools to analyse uncertainty have received an increasing interest from scientists and engineers. Uncertainty analysis refers to methods, which attempt to quantify the uncertainty in any quantitative statement, such as for example in estimating the uncertainty in the reservoir simulator output that results from the simulator's input factors. Another important task of uncertainty analysis is to identify the uncertainty sources (factors) that are relevant to a particular reservoir model. This task is generally performed by the so-called sensitivity analysis. We will only consider sensitivity analysis procedures in the work developed in this thesis.
The aim of sensitivity analysis for computer codes is to quantify how the variation (uncertainty) in the output of the computer code is apportioned to different input of the model. The most useful methods that perform sensitivity analysis require stochastic simulation techniques, such as Monte-Carlo methods. Such methods usually necessitate several thousands computer code evaluations that are generally not affordable with reservoir simulators for which each simulation requires several minutes or hours. Consequently, response surface methods become an interesting alternative.
A response surface serves as a simplified model that is intended to approximate the simulator's input/output relation and that is fast to evaluate. The general idea of this approach is to perform a limited number of the simulator evaluations (hundreds) at some carefully chosen training input values, and then, using statistical regression techniques, such as, for example, polynomial regression, Gaussian process or smoothing spline regression, construct an approximation of the simulator. If the resulting approximation is of a good quality, the estimated response surface is used instead of the complex and computationally demanding simulator to perform the sensitivity analysis.
Most common regression techniques used to build response surfaces are computationally efficient and accurate provided that the simulator's input/output relation is smooth enough and the number of input factors is moderate. However, with such methods, the computational cost can become very substantial for high dimensional problems. Since it is more often the case in practice to deal with reservoir simulators that involve a high number of inputs (more than ten), using the usual regression techniques to build appropriate

response surfaces may be a problem.
The efficiency of sensitivity analysis depends on the accuracy of the response surface, on the number of input factors and, as already mentioned, sensitivity analysis techniques can involve Monte-Carlo methods that require huge random samples when dealing with high dimensional cases. Even when the evaluation of the response surface is much faster than the simulator, when applied to hundreds of thousands of input values it becomes computationally demanding and requires an appropriate approach to efficiently perform the analysis.
Another challenging problem, within the response surface framework, arises with functional output. Reservoir simulators model the evolution of time-dependent physical quantities, so their outputs can be composed by several time series. Using classical response surface methods by including time as an extra input factor leads to complications in practice. A first one is the need to deal with extremely large datasets, which results in computationally demanding problems (sometimes intractable). A second one is that most response surface related methods are adapted to quite regular variations of the inputs which is not the case when dealing with time series curves that are irregular.

1.2 Overview of the thesis

The present work aims at developing response surface methods, as well as their application to sensitivity analysis of computer codes, taking into account the above remarks. The organization of the thesis document is as follows.
In Chapter 2, we emphasize and recall some of the main topics that are related to variance based global sensitivity analysis. The concept of experiment design is also briefly discussed and finally, the most common response surface methods available in the literature are introduced and discussed.
In Chapter 3, we present the component selection and smoothing operator (COSSO) regularized nonparametric regression method, which is a general nonparametric model fitting procedure that also performs variable selection and which is based on analysis of variance decompositions (ANOVA). These classes of regression methods (ANOVA based one's) seem to have been underused for building response surfaces and their use is justified from the examples treated in this chapter. One of COSSO's algorithmic steps involves the solution of a nonnegative garrote (NNG) convex optimization problem. Using classical constrained

optimization techniques to solve the NNG problem is efficient but time consuming, especially with high dimensional problems and with large sizes of the experimental design. To bypass this difficulty, we develop a new iterative algorithm based on Landweber iterative algorithm, which are conceptually simple and easy to implement. For comparison purposes we have adapted a nonnegative least angle regression algorithm (reviewed in Chapter 2) which is also known to be efficient. We empirically show, on analytical and reservoir test cases, that COSSO response surface estimation based on our iterative algorithm is the fastest one. Moreover, using the fact that COSSO is based on ANOVA type decompositions allows us to derive a new direct method for computing sensitivity indices.

The response surfaces studied in Chapter 3 involve usually outputs that vary regularly and in a smooth way with respect to the inputs. They are not adapted to cases for which the response involves more roughly outputs. For this purpose, in Chapter 4, we introduce a new regularized nonparametric regression type method, named wavelet kernel ANOVA (WK-ANOVA). This method is similar to COSSO's approach but since it deals with irregular outputs is based on wavelet decompositions instead of splines. Once the wavelet decomposition is adapted to treat irregular designs WK-ANOVA seems as efficient than COSSO, and similarly to what has been done in the previous chapter allows to develop a direct method for computing sensitivity indices. We illustrate the effectiveness of the WK-ANOVA method as well as its limits on analytical and reservoir test cases.

Chapter 5 is devoted to the problem of time series computer code outputs. For such problems, we introduce an original methodology based on an expansion of the time series curves in a wavelet basis, followed by a thresholding procedure, specifically designed for analyzing multiple curve sets. We then use a COSSO-like method to approximate the selected wavelet coefficients and adapt Sobol's Monte-Carlo based estimation methods (presented in chapter 2) to compute time-dependent sensitivity indices. The efficiency is shown on a reservoir test case.

Chapter 2

Response surface for sensitivity analysis

2.1 Introduction

Consider a mathematical model for a computer code simulator

$$Y = f(\mathbf{X}) \tag{2.1}$$

where Y is the output scalar of the computer code realisations, $\mathbf{X} = (X^{(1)}, \ldots, X^{(d)})$ a d-dimensional input vector which represents the uncertain parameters/factors of the simulator and $f : \mathbb{R}^d \to \mathbb{R}$ is a function that models the relationship between the input factors and the output of the computer code.
Sensitivity analysis (SA for short) is the study of how the variation (uncertainty) in the output of the mathematical model can be apportioned, qualitatively or quantitatively, to different sources of variation in the input of the model. Put in another way, it is a technique for systematically changing parameters in a model to determine the effects of such changes on the output. There are several possible procedures to perform sensitivity analysis (SA). Important classes of methods are:

- Local methods, such as the simple derivative of the output Y with respect to a given component $X^{(i)}$ of the input vector $\mathbf{X}$: $|\frac{\partial Y}{\partial X^{(i)}}|_{\mathbf{x}_0}$, where the subscript $\mathbf{x}_0$ indicates that the derivative is taken at some fixed point in the space of the input (hence the 'local' in the name of the class).

- Sampling-based methods in which the model is executed repeatedly for combinations of values sampled from the distribution (assumed known) of the input factors. Once

the sample is generated, several strategies (including simple input-output scatterplots) can be used to derive global sensitivity measures for the factors.

Other methods which are also sampling-based are those based on Monte Carlo filtering whose objective is to identify regions in the space of the input factors corresponding to particular values (e.g. high or low) of the output.
Several work are devoted to global SA (GSA). Among them we cite: Sobol (1993), Saltelli & Sobol' (1995) and Saltelli & Sobol' (1999). Particular instances of global stochastic based methods are regression-based methods, which measure the effects of the input on the output if the mathematical model representing the computer code is linear, and variance-based methods, where the unconditional variance $V(Y)$ of Y is decomposed into terms due to individual factors plus terms due to interaction among the components of the vector of inputs by means of an analysis of variance decomposition (ANOVA) and which are, therefore, nonlinear with respect to the original input parameters. Most variance-based methods are quantitative, and in this work we will focus on this class of methods, and more specifically on Sobol's indices.
One of the main issues with variance based methods is computational time. Indeed, a computer code that is sufficiently realistic is often very costly in terms of computational time. Furthermore such methods require generally several thousands simulations that are usually not affordable in common applications. In order to perform sensitivity analysis with a limited number of runs, response surface methods can then be used. In the latter the simulator input/output relation is approximated using different statistical regression techniques starting from an initial set of carefully chosen training runs. Then, if a reasonably good approximation is obtained, the estimated response surface is used instead of the complex simulator to compute the sensitivity indices. Response surface methods have known a quick development in the last decade and the resulting mathematical approximation is also called a metamodel or a surrogate. Many different approaches have been suggested in many different scientific disciplines. The construction of an efficient response surface involves determining configurations of inputs (experimental design) for running the simulator to build a sufficiently accurate response surface with an as small as possible number of simulator runs.
The purpose of this chapter is to highlight and recall some of the main topics related to variance based global sensitivity methods that are relevant to the rest of this work. The concepts of experiment design will also be shortly discussed and finally the most common

response surface methods for regression will be introduced and discussed together with their potential applications to the problems we are going to deal with in further chapters.

2.2 Global sensitivity analysis

2.2.1 Variance based Sobol's indices

In order to describe this concept, let us suppose that the mathematical model is described by a function $f(\mathbf{X}), \mathbf{X} = (X^{(1)}, \cdots, X^{(d)})$, and is defined on the unit d-dimensional cube ($\mathbf{X} \in [0,1]^d$):

$$\Omega^d = \{\mathbf{X} |\ 0 \leq X^{(j)} \leq 1; j = 1, \ldots, d\}. \tag{2.2}$$

The main idea from Sobol (1993)'s approach is to decompose the response $Y = f(X^{(1)}, \ldots, X^{(d)})$ into summands of different dimensions via a so-called ANOVA (ANalysis Of VAriance) decomposition, defined as follows:

$$f(X^{(1)}, \cdots, X^{(d)}) = f_0 + \sum_{j=1}^{d} f_j(X^{(j)}) + \sum_{1 \leq j < l \leq d} f_{jl}(X^{(j)}, X^{(l)}) + \ldots + f_{1,2,\ldots,d}(X^{(1)}, \cdots, X^{(d)}) \tag{2.3}$$

where f_0 is a constant, f_j's are univariate functions representing the main effects, f_{jl}'s are bivariate functions representing the two way interactions, and so on. The integrals of every summand of the ANOVA decomposition over any of its own variables is assumed to be equal to zero , i.e.

$$\int_0^1 f_{j_1,\ldots,j_s}(X^{(j_1)}, \ldots, X^{(j_s)}) dX^{(j_k)} = 0 \tag{2.4}$$

where $1 \leq j_1 < \ldots < j_s \leq d$, $s = 1, \ldots, d$ and $1 \leq k \leq s$. It follows from this property that all the summands in (2.3) are orthogonal, i.e, if $(i_1, \ldots, i_s) \neq (j_1, \ldots, j_l)$, then

$$\int_{\Omega^d} f_{i_1,\ldots,i_s} f_{j_1,\ldots,j_l} d\mathbf{X} = 0 \tag{2.5}$$

Using the orthogonality, Sobol showed that such decomposition of $f(X^{(1)}, \ldots, X^{(d)})$ is unique and that all the terms in (2.3) can be evaluated via multidimensional integrals:

$$f_0 = E(Y) \tag{2.6}$$

$$f_j(X^{(j)}) = E(Y|X^{(j)}) - E(Y) \tag{2.7}$$

$$f_{j,l}(X^{(j)}, X^{(l)}) = E(Y|X^{(j)}, X^{(l)}) - f_j - f_l - E(Y) \tag{2.8}$$

where $E(Y)$ and $E(Y|X^{(j)})$ are respectively the expectation and the conditional expectation of the output Y. Analogous formulae can be obtained for the higher-order terms. If all the input factors are mutually independent, the ANOVA decomposition is valid for any distribution function of the $X^{(i)}$'s and using this fact, squaring and integrating (2.3) over Ω^d, and by (2.5), we obtain

$$V = \sum_{j=1}^{d} V_j + \sum_{1 \leq j < l \leq d} V_{jl} + \ldots + V_{1,2,\ldots,d} \tag{2.9}$$

where $V_j = V[E(Y|X^{(j)})]$ is the variance of the conditional expectation that measures the main effect of X_j on Y and $V_{jl} = V[E(Y|X^{(j)}, X^{(l)})] - V_j - V_l$ measures the joint effect of the pair $(X^{(j)}, X^{(l)})$ on Y. The total variance V of Y is defined to be

$$V = E(Y^2) - f_0^2 \tag{2.10}$$

Variance-based sensitivity indices, also called Sobol indices, are then defined by:

$$S_{j_1,\ldots,j_s} = \frac{V_{j_1,\ldots,j_s}}{V} \tag{2.11}$$

where S_j is called the first order sensitivity index (or the main effect) for factor $X^{(j)}$, which measures the main effect of $X^{(j)}$ on the output Y, the second order index S_{jl}, for $j \neq l$, is called the second order sensitivity index expresses the sensitivity of the model to the interaction between variables $X^{(i)}$ and $X^{(j)}$ on Y and so on for higher orders effects. The decomposition in (2.9) has the useful property that all sensitivity indices sum up to one.

$$\sum_{j=1}^{p} S_j + \sum_{1 \leq j < l \leq p} S_{jl} + \ldots + S_{1,2,\ldots,p} = 1 \tag{2.12}$$

The total sensitivity index (or total effect) of a given factor is defined as the sum of all the sensitivity indices involving the factor in question.

$$S_{T_j} = \sum_{l \# j} S_l \tag{2.13}$$

where $\#j$ represents all the $S_{j_1,\ldots,j_s}$ terms that include the index j. Total sensitivity indices measure the part of output variance explained by all the effects in which it plays a role. Note however that the sum of all S_{T_j} is higher than one because same interaction terms are counted several times. It is important to note that total sensitivity indices can be computed by a single multidimensional integration and do not require computing all

high order indices (see below). Then comparing the total effect indices provides information about influential parameters. GSA enables to explain the variability of the output response as a function of the input parameters through the definition of total and partial sensitivity indices. The computation of these indices involves the computation of several multidimensional integrals that are estimated by Monte Carlo method and thus requires huge random samples. For this reason GSA techniques are prohibitive if used directly using the computer code (fluid flow simulator for example). In this work the computation of the required sensitivity indices will be performed using response surface techniques that are discussed in following sections.

2.2.2 Monte-Carlo procedure for estimating Sobol' indices

Consider a N i.i.d random sample from the distribution of $\mathbf{X}$, say $\{\mathbf{x}_i = (x_{i_1}, \ldots, x_{i_d})^T, i = 1, \ldots, N\}$. The constant f_0 and the total variance V are then estimated by

$$\widehat{f}_0 = \frac{1}{N}\sum_{i=1}^{N} f(x_{i_1}, \ldots, x_{i_d}) \tag{2.14}$$

$$\widehat{V} = \frac{1}{N}\sum_{i=1}^{N} f^2(x_{i_1}, \ldots, x_{i_d}) - \widehat{f}_0^2 \tag{2.15}$$

The estimation of the Sobol's indices requires the estimation of the variance of the conditional expectation. Sobol (1993) has used a Monte-Carlo procedure to do this. For example, the estimation of V_j involves two independent i.i.d. N-sample random samples sets $\{\mathbf{x}_i = (x_{i_1}, \ldots, x_{i_d})^T, i = 1, \ldots, N\}$ and $\{\mathbf{z}_i = (z_{i_1}, \ldots, z_{i_d})^T, i = 1, \ldots, N\}$ from the distribution of $\mathbf{X}$. The Monte-Carlo estimate of $V_j = V[E(Y|X^{(j)})] = E[E^2(Y|X^{(j)})] - f_0^2$ is then given by

$$\widehat{V}_j = \frac{1}{N}\sum_{i=1}^{N} f(x_{i_1}, \ldots, x_{i_d}) f(z_{i_1}, \ldots z_{i_{j-1}}, x_{i_j}, z_{i_{j+1}}, \ldots, z_{i_d}) - \widehat{f}_0^2 \tag{2.16}$$

Thus, the first order indices are estimated as

$$\widehat{S}_j = \frac{\widehat{V}_j}{\widehat{V}} \tag{2.17}$$

The estimation of $V_{jl} = V[E(Y|X^{(j)}, X^{(l)})] - V_j - V_l = E[E^2(Y|X^{(j)}X^{(l)})] - f_0^2 - V_j - V_l$ are given by the same procedure as

$$\widehat{V}_{jl} = \frac{1}{N}\sum_{i=1}^{N} f(x_{i_1}, \ldots, x_{i_d}) f(z_{i_1}, \ldots z_{i_{j-1}}, x_{i_j}, z_{i_{j+1}}, \ldots, z_{i_{l-1}}, x_{i_l}, z_{i_{l+1}}, \ldots, z_{i_d}) - \widehat{f}_0^2 \tag{2.18}$$

Thus, the second order indices are estimated as

$$\widehat{S}_{jl} = \frac{\widehat{V}_{jl} - \widehat{V}_j - \widehat{V}_l}{\widehat{V}} \tag{2.19}$$

and so on for obtaining the estimates of the sensitivity indices of higher order. The total effect indices S_{T_j} can be estimated directly, without estimating all indices which include the index j. Indeed, total effect indices can be written as

$$S_{T_j} = 1 - \frac{V[E(Y|X^{(-j)})]}{V} = 1 - \frac{V_{-j}}{V} \tag{2.20}$$

where V_{-j} correspond to the variance of the expectation conditioned to all the inputs except $X^{(j)}$. The estimation of V_{-j} is given by

$$\widehat{V}_{-j} = \frac{1}{N}\sum_{i=1}^{N} f(x_{i_1}, \ldots, x_{i_d}) f(x_{i_1}, \ldots x_{i_{j-1}}, z_{i_j}, x_{i_{j+1}}, \ldots, x_{i_d}) - \widehat{f}_0^2 \tag{2.21}$$

Hence the estimation of the total effect indices S_{T_j}

$$\widehat{S}_{T_j} = 1 - \frac{\widehat{V}_{-j}}{\widehat{V}} \tag{2.22}$$

We will not further discuss such simulation based estimation procedures for Sobol's indices since in the following we will mainly deal with metamodels.

2.3 Experimental design

2.3.1 A general framework

An important issue for building predictive response surfaces is the choice of the training data $\mathcal{X}$. Traditionally called experimental design, it is a set of input points at which the computer code is run, selected for the purpose of building the most predictive response surface.

The computer code is considered as a deterministic function, in other words for a given input values the computer code produce the same results. Consequently, design techniques using a replication of points are useless. Moreover, the input/output relation is unknown. Hence, the design based on random samples where the points are spread randomly throughout the experimental region, are usually preferred. This type of design are called space filling design.

Several space filling design techniques have been proposed for computer experiments (Santner *et al.*, 2003), the most popular ones are the Latin Hypercube Designs (LHD) (McKay *et al.*, 1979).

2.3.2 Latin hypercube designs

To introduce LHDs, consider an experimental region of d inputs, where without loss of generality, each input has been rescaled in the interval $[0, 1]$. To obtain a design consisting of N points, divide each one of the p input axes into N equally spaced intervals $\{[0, 1/N], \ldots, [(N-1)/N, 1]\}$. This partitions the scaled experimental region $\Omega \equiv [0, 1]^p$ into N^p cells of equal volume. The design consists first in selecting cells between the N^p and then assigning randomly a design point (by a uniform probability) in the chosen cell. The selection of the cells is done by taking independent permutations of the intervals of each axes. The latin hypercube design has the advantage of producing a sampled point in each of the N partitions of each input. To obtain an LHD with a better space filling criteria we can sample a high number of different LHDs and select the one maximizing the minimum distance between two points, also referred to as the maximin criterion (maximinLHD). Also note that LHDs can be easily generalized to the case of more general input distributions such as, for example, triangular or normal distributions. However we generally prefer using uniform distributions for building response surfaces because otherwise the risk is to obtain a very inaccurate response surface in the low probability regions. The number of design points (simulations) necessary to obtain a reliable response surface generally depends on the number of inputs and on the complexity of the response to analyze.

2.4 Response surface

As previously mentioned the computation of the sensitivity indices requires a huge number of model evaluations. For example, in the framework of reservoir simulation these computer models require several minutes or even several hours to perform one simulation. Thus, computing the sensitivity indices directly is impractical. Therefore, using an approximation of the computer code which is much faster to evaluate than the corresponding simulator seems to be a good reasonable alternative.

Response surface techniques are regression analysis methods for building an approximation of the computer code based on a limited number of evaluations (observations) of this simulator. These approximation models need to be as accurate as possible, in terms of prediction, to provide a reliable GSA results. The most commonly used response surface methods are those based on parametric polynomial regression models, which require to specify the polynomial form of the regression mean (linear, quadratic, ...). However, it

is often the case that the linear (or quadratic) model can fail to identify properly the input/output relation. Thus, in nonlinear situations, nonparametric regression methods are preferred.

Parametric regression models, especially linear models, have many interesting properties such as computational speed and easy interpretability. When the computer code to be approximated is nearly linear in the inputs, there are no better methods. However, when the input/output relationship is not linear, a linear or polynomial parametric regression has a very poor approximation properties. In such cases alternative regression methods may be used as response surface. In the last decade many different nonparametric regression models have been used as a response surface methods. To name a few of them, Sacks *et al.* (1989), Busby *et al.* (2007) and Marrel *et al.* (2008) utilized a Gaussian Process (GP). Sudret (2008) and Blatman & Sudret (2010) used a polynomial chaos expansions to perform a GSA.

In addition, Storlie & Helton (2008a), Storlie & Helton (2008b) and Storlie *et al.* (2009) provide a comparison of various parametric and nonparametric regression models, such as linear regression (LREG), quadratic regression (QREG), projection pursuit regression multivariate adaptive regression splines (MARS), gradient boosting regression, random forest, Gaussian process (GP), adaptive component selection and smoothing operator (ACOSSO), etc ... for providing appropriate metamodel startegies. The authors note that ACOSSO and GP perform well in all cases considered nevertheless suffer from higher computational time.

In the following, we present parametric regression methods and provide a review on a highly popular technique of the variable selection (regularization). In addition, due to its reported good performance, we choose to use GP to compare the methods that we will introduce in the following chapters. For the sake of completeness we will therefore describe in some details the various response surface methodologies that are going to be used and compared.

For each of the following procedures, it is assumed that we have n independent realisations (say observations) $\{(y_i, \mathbf{x}_i = (x_{i_1}, \ldots, x_{i_d})^T), i = 1, \ldots, n\}$ of a computer code, generated via the relation given in (2.1).

2.4.1 Parametric regression

The most frequently used parametric regression models (linear, quadratic, ...) are linear in the coefficients and for that reason are also known as linear regression models in the

statistics literature. A linear regression model has the following form

$$f(\mathbf{X}) = \beta_0 + \sum_{j=1}^{d} \beta_j X^{(j)} + \epsilon \tag{2.23}$$

Here the β_j's are unknown coefficients which we want to estimate and $X^{(j)}$ are the regressors which can be functions of other explanatory variables. The most popular estimation approach is least-squares, in which the coefficients $\widehat{\beta}^{LS} = (\widehat{\beta}_0, \widehat{\beta}_1, \ldots, \widehat{\beta}_d)$ minimize the residual sum of squares

$$\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{d} x_{i_j} \beta_j)^2, \tag{2.24}$$

where $x_{i_j} = X_i^{(j)}$. Denote by X the $n \times (d+1)$ matrix with each column the values of the corresponding regressor (input) (with 1 in the first position corresponding to β_0), and similarly let Y be the vector of outputs in the observation set. Then $\widehat{\beta}^{LS}$ satisfies

$$X^T X \widehat{\boldsymbol{\beta}}^{LS} = X^T \mathbf{Y} \tag{2.25}$$

and, assuming that X has full column rank d ($d \leq n$ and $X^T X$ is positive definite and can be inverted), we obtain a unique solution of the regression coefficients

$$\widehat{\beta}^{LS} = (X^T X)^{-1} X^T \mathbf{Y}$$

Note that $X^{(j)}$ can correspond to:

- linear term $X^{(j)}$ (quantitative input) which quantify the direct influence of the input $X^{(j)}$ on the output Y
- quadratic term $(X^{(j)})^2$ which quantify the quadratic effect of the input $X^{(j)}$ on the output Y
- a product term $X^{(j)}X^{(l)}$ which quantify the interaction influence between inputs $X^{(j)}$ and $X^{(l)}$ on the output Y
- transformation of $X^{(j)}$, such as logarithm or exponential function.

Linear regression (LREG), which involves the linear terms and the product terms, and quadratic regression (QREG), which is a LREG plus the quadratic terms are two of the most commonly used response surface methods in practice.

2.4.2 Regularized parametric regression

When multicolinearity problems among the regressors are present, the X^TX matrix characterizing the system of normal equations (2.25) is ill-conditioned. Moreover, unless one has many more training cases than inputs, least squares estimates for regression coefficients may have a too high variance. Several ways exist to reduce the variance and to enhance the precision of least squares estimates most of them trying to limit the "complexity" of the fitted models by appropriate techniques: either by using a selected subset of input variables (since, in practice, it is plausible that only a small proportion of input are influential on the output), or by finding estimates for the regression coefficients by minimizing the residual sum of squares (RSS) plus a penalty involving the size of the $\boldsymbol{\beta}$s (or equivalently, by minimizing RSS subject to a constraint on the size of the $\boldsymbol{\beta}$s, or by replacing the original inputs with a smaller set of variables that are linear combinations of the original inputs. These methods overlap somewhat: some penalty methods may set some β's exactly to zero, effectively eliminating those input variables. Input selection can be seen as choosing directions restricted to the coordinate axes. When regularizing the regression, the coefficients $\widehat{\boldsymbol{\beta}}$ are defined as the minimum of the penalized least squares functional defined by

$$\sum_{i=1}^{n}(y_i-\beta_0-\sum_{j=1}^{d}x_{i_j}\beta_j)^2+\lambda\mathrm{Pen}(\boldsymbol{\beta})=\mathrm{RSS}+\lambda\mathrm{Pen}(\boldsymbol{\beta}), \tag{2.26}$$

where Pen is the penalty function. Several penalty functions exist and we review some of them below.

2.4.2.1 Ridge regression

Introduced in Hoerl & Kennard (1970), ridge regression adds a penalty of $\lambda\sum_{j=1}^{d}\beta_j^2$ to the residual sum of squares producing a shrinkage on the regression coefficients. More precisely, $\widehat{\beta}^{ridge}$ minimizes the functional

$$\sum_{i=1}^{n}(y_i-\beta_0-\sum_{j=1}^{d}x_{i_j}\beta_j)^2+\lambda\sum_{j=1}^{d}\beta_j^2 \tag{2.27}$$

where the positive scalar λ is a regularization parameter that controls the amount of shrinkage and the penalty function is given by the l_2 norm. An equivalent form of (2.27) is

$$\sum_{i=1}^{n}(y_i-\beta_0-\sum_{j=1}^{d}x_{i_j}\beta_j)^2,\ \text{subject to}\ \sum_{j=1}^{d}\beta_j^2\leq s \tag{2.28}$$

where s is a positive scalar associated to λ. The intercept β_0 is usually not included in the penalty. This can be done by first centering the inputs and response variables (shifting them to have mean zero over the training set), then fitting a model with no intercept. The matrix form of (2.27) is

$$(\mathbf{Y} - X\boldsymbol{\beta})^T(\mathbf{Y} - X\boldsymbol{\beta}) + \lambda\boldsymbol{\beta}^T\boldsymbol{\beta}.$$

That is

$$\widehat{\boldsymbol{\beta}}^{ridge} = (X^TX + \lambda\mathbf{I})^{-1}X^T\mathbf{Y}$$

Note that the addition of the positive constant to the diagonal of X^TX makes the problem nonsingular even when $d > n$. Ridge regression estimate are biased, but have lower variance than least squares estimates. Also, note that the ridge regression usually includes all the predictors in the response surface.

2.4.2.2 LASSO

The Least absolute shrinkage and selection operator method (LASSO), introduced by Tibshirani (1996), is a shrinkage method where the penalty function is based on a l_1 norm of the vector of coefficients. The intercept β_0 is usually not included in the penalty. This can be done by first centering the inputs and response variables (shifting them to have zero mean over the training set), then fitting a model with no intercept. The LASSO estimate $\widehat{\beta}^{lasso}$ minimizes the RSS with the l_1-penalty

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{d}x_{i_j}\beta_j)^2 + \lambda\sum_{j=1}^{d}|\beta_j| \tag{2.29}$$

As for the ridge regression an equivalent form of the lasso in terms of least squares with a constraint is

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{d}x_{i_j}\beta_j)^2, \text{ subject to } \sum_{j=1}^{d}|\beta_j| \leq s. \tag{2.30}$$

For every choice of s, there is a choice of λ that gives the same result, and vice versa. Due to the l_1-penalty, the solution of LASSO is usually sparse when a sufficiently high regularization parameter λ is used. This property makes LASSO a variable selection method. The estimation of LASSO is a convex optimization problem and can be solved, for a given λ, via a quadratic programming algorithm (solver). It is clear that this become computationally expensive since it requires solving the optimization problem for a fine grid of λ's. However, an efficient algorithm introduced by Efron *et al.* (2004), Least angle

regression (LARS), is available for computing the entire path solution as λ is varied with a small computational cost.

LARS builds a model sequentially by adding one variable at a time. Indeed, the procedure starts by identifying the variable that is most correlated with the output and puts it in the active set (at this time the active set contains just one variable). Then LARS moves the coefficient of this variable continuously from 0 toward its least-squares value, until another variable has as much correlation with the current residual as does the first one. The second variable joins the active set and their coefficients are moved jointly in a way that decreases their correlation with the current residual, until some other variable has as much correlation with the residual. If $q \leq n$ the process is continued until all variables are in the model, and if $q > n$ the algorithm stops after $n - 1$ steps.

Let $\mathcal{A}_k$ be the active set of variables at the kth step, its complementary is denoted by $\mathcal{A}_k^c$, and let $\boldsymbol{\beta}_{\mathcal{A}_k}$ be the coefficients vector corresponding to the variables from the active set. Hence, $\mathbf{r}^{[k]} = \mathbf{Y} - X\boldsymbol{\beta}^{[k]}$ is the current residual.

The modified LARS algorithm which provide the entire paths of LASSO coefficients is defined as

1. Start from $k = 1$, $\beta_1^{[0]}, \ldots, \beta_q^{[0]} = 0$, $\mathcal{A}_k = \emptyset$ and the residual $\mathbf{r}^{[0]}$ equal to the vector of the observation $\mathbf{Y}$

2. Update the active set by including the variable $X^{(j^*)}$ most correlated with the current residual $\mathbf{r}^{[k]}$

$$\mathcal{A}_k = \mathcal{A}_{k-1} \cup \{j^*\}, \text{with } j^* = \arg\max_{j \in \mathcal{A}_k^c} (\mathbf{X}_j^T \mathbf{r}^{[k-1]})$$

   where $\mathbf{X}_j$ the jth column of X

3. Given that, all variables from the active set are equally correlated to the current residual, the vector of empirical correlations satisfies

$$X_{\mathcal{A}_k}^T \mathbf{r}^{[k]} = \alpha \mathbb{1}_{\mathcal{A}_k}$$

   where $\mathbb{1}_{\mathcal{A}_k}$ is a vector of 1's of length $card(\mathcal{A}_k)$ and α is a constant. Thus the descent direction vector is defined as

$$\mathbf{w}_{\mathcal{A}_k}^{[k]} = (X_{\mathcal{A}_k}^T X_{\mathcal{A}_k})^{-1} \alpha \mathbb{1}_{\mathcal{A}_k}$$

   Since $\mathbf{w}_{\mathcal{A}_k}^{[k]}$ is a unit descent direction, α is defined as

$$\alpha = \frac{1}{\sqrt{\mathbb{1}_{\mathcal{A}_k}^T (X_{\mathcal{A}_k}^T X_{\mathcal{A}_k})^{-1} \mathbb{1}_{\mathcal{A}_k}}}$$

4. Now as the vector of descent direction is determined, one needs to select the descent step γ. This step must be the smallest positive value such that a new predictor $X^{(j)}$ enters the active set. This condition can be written as

$$\min_{l \in \mathcal{A}_k^c} \mid \mathbf{X}_l^T \mathbf{r}^{[k]} \mid = \mid \mathbf{X}_j^T \mathbf{r}^{[k+1]} \mid, \text{with } j \text{ is arbitrarily chosen from } \mathcal{A}_k$$

where $\mathbf{r}^{[k+1]} = \mathbf{Y} - X\boldsymbol{\beta}^{k+1} = \mathbf{Y} - X\boldsymbol{\beta}^k - \gamma X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}$. In other words, we need to compute γ_l for every $l \in \mathcal{A}_k^c$ that satisfies $\mathbf{X}_l^T \mathbf{r}^{[k+1]} = \mathbf{X}_j^T \mathbf{r}^{[k+1]}$, then the smallest will be chosen. Hence

$$\mathbf{X}_l^T \mathbf{r}^{[k]} - \gamma_l \mathbf{X}_l^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} = \mathbf{X}_j^T \mathbf{r}^{[k]} - \gamma_l \mathbf{X}_j^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}, \text{with } l \in \mathcal{A}_k^c \text{ and } j \in \mathcal{A}_k$$

It follows

$$\gamma_l = \frac{\mathbf{X}_j^T \mathbf{r}^{[k]} - \mathbf{X}_l^T \mathbf{r}^{[k]}}{\mathbf{X}_l^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} - \mathbf{X}_j^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}}, \text{with } l \in \mathcal{A}_k^c \text{ and } j \in \mathcal{A}_k$$

In order to consider the positive and the negative correlation, we also study γ such as

$$\gamma_l = \frac{\mathbf{X}_l^T \mathbf{r}^{[k]} - \mathbf{X}_j^T \mathbf{r}^{[k]}}{\mathbf{X}_j^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} - \mathbf{X}_l^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}}, \text{with } l \in \mathcal{A}_k^c \text{ and } j \in \mathcal{A}_k.$$

As a result, the descent step γ is defined as

$$\gamma = \min_{l \in \mathcal{A}_k^c}{}^{+} \left\{ \frac{\mathbf{X}_j^T \mathbf{r}^{[k]} - \mathbf{X}_l^T \mathbf{r}^{[k]}}{\mathbf{X}_l^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} - \mathbf{X}_j^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}}, \frac{\mathbf{X}_l^T \mathbf{r}^{[k]} - \mathbf{X}_j^T \mathbf{r}^{[k]}}{\mathbf{X}_j^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} - \mathbf{X}_l^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}} \right\}$$

where $\min^+$ indicates that the minimum is taken over only positive components within each choice of l.

5. Update the coefficients vector corresponding to the variables from the active set

$$\boldsymbol{\beta}_{\mathcal{A}_k}^{[k+1]} = \boldsymbol{\beta}_{\mathcal{A}_k}^{[k]} + \gamma \mathbf{w}_{\mathcal{A}_k}^{[k]}$$

6. If a nonzero coefficient β_{j^*} hits zero, in other words if the sign has changed between $\beta_{j^*}^{[k]}$ and $\beta_{j^*}^{[k+1]}$, set γ such as $\beta_{j^*}^{[k+1]} = 0$

$$\gamma = \frac{\beta_{j^*}^{[k+1]} - \beta_{j^*}^{[k]}}{\mathbf{w}_{j^*}^{[k]}} = \frac{-\beta_{j^*}^{[k]}}{\mathbf{w}_{j^*}^{[k]}},$$

update the coefficients vector by using the new γ

$$\boldsymbol{\beta}_{\mathcal{A}_k}^{[k+1]} = \boldsymbol{\beta}_{\mathcal{A}_k}^{[k]} + \gamma \mathbf{w}_{\mathcal{A}_k}^{[k]},$$

and drop the variables $X^{(j^*)}$ from the active set $\mathcal{A}_{k+1} = \mathcal{A}_k - \{j^*\}$. This step of the modified LARS ensures that the solution path corresponds to the LASSO solution.

7. Set $\mathbf{r}^{[k+1]} = \mathbf{Y} - X\boldsymbol{\beta}^{[k+1]}$, $k = k+1$ and continue until $\min(q, n-1)$ variables have been entered

2.4.2.3 Nonnegative garrote

The lasso estimates tend to have larger bias, due to the shrinkage of large coefficients. To remedy this drawback one can use the nonnegative garrote shrinkage method (NNG). Introduced by Breiman (1995) the NNG is a scaled version of the least square estimate. Thus, the shrinking factors $c = (c_1, \ldots, c_d)$ minimize

$$\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{d} x_{i_j} c_j \beta_j^{LS})^2 + \lambda \sum_{j=1}^{d} c_j, \text{ subject to } c_j \geq 0 \tag{2.31}$$

The NNG estimates of the regression coefficients are defined as $c_j \beta_j^{LS}$. In other words, the NNG can be considered as a method that shrinks the least squares estimator by multiplying it by some diagonal matrix with shrinking constants. As for the ridge regression and the LASSO an equivalent formulation of (2.31) is

$$\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{d} x_{i_j} c_j \beta_j^{LS})^2, \text{ subject to } c_j \geq 0 \text{ and } \sum_{j=1}^{d} c_j \leq s \tag{2.32}$$

Yuan & Lin (2007) provided an efficient algorithm similar to the modified LARS for computing the entire path solution as λ is varied.
Let $c_{\mathcal{A}_k}$ be the coefficient vector corresponding to the variables from the active set. Hence, $\mathbf{r}^{[k]} = \mathbf{Y} - Zc^{[k]}$ is the current residual, where $\mathbf{Z}_j = \mathbf{X}_j \beta_j^{LS}$.
The modified LARS algorithm which provide the entire paths of LASSO coefficients is defined as

1. Compute the least-squares coefficients $\boldsymbol{\beta}^{LS}$ and set $\mathbf{Z}_j = \mathbf{X}_j \beta_j^{LS}$
2. Start from $k = 1$, $c_1^{[0]}, \ldots, c_q^{[0]} = 0$, $\mathcal{A}_k = \emptyset$ and the residual $\mathbf{r}^{[0]}$ equal to the vector of the observation $\mathbf{Y}$
3. Update the active set

$$\mathcal{A}_k = \mathcal{A}_{k-1} \cup \{j^*\}, \text{with } j^* = \arg\max_{j \in \mathcal{A}_k^c} (\mathbf{Z}_j^T \mathbf{r}^{[k-1]})$$

4. Compute the current descent direction vectors

$$\mathbf{w}_{\mathcal{A}_k}^{[k]} = (Z_{\mathcal{A}_k}^T Z_{\mathcal{A}_k})^{-1} Z_{\mathcal{A}_k}^T \mathbf{r}^{[k]}$$

5. Now, for every $l \in \mathcal{A}_k^c$ compute γ_l that satisfies $\mathbf{Z}_l^T \mathbf{r}^{[k+1]} = \mathbf{Z}_j^T \mathbf{r}^{[k+1]}$. Hence

$$\mathbf{Z}_l^T \mathbf{r}^{[k]} - \gamma_l \mathbf{Z}_l^T Z_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} = \mathbf{Z}_j^T \mathbf{r}^{[k]} - \gamma_l \mathbf{Z}_j^T X_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}, \text{ with } l \in \mathcal{A}_k^c \text{ and } j \in \mathcal{A}_k$$

   It follows

$$\gamma_l = \frac{\mathbf{Z}_j^T \mathbf{r}^{[k]} - \mathbf{Z}_l^T \mathbf{r}^{[k]}}{\mathbf{Z}_l^T Z_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} - \mathbf{Z}_j^T Z_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}}, \text{ with } l \in \mathcal{A}_k^c \text{ and } j \in \mathcal{A}_k$$

6. For every $j \in \mathcal{A}_k$, compute $\gamma_j = \min(\alpha_j, 1)$ where $\alpha_j = -c_j^{[k]} / \mathbf{w}_j^{[k]}$

7. If for every j we have $\gamma_j \leq 0$ or $\min_j{}^+(\gamma_j) > 1$, set $\gamma = 1$. Otherwise, set $\gamma = \gamma_{j^*} = \min_j{}^+(\gamma_j)$ and update the coefficients vector by using the new γ

$$c_{\mathcal{A}_k}^{[k+1]} = c_{\mathcal{A}_k}^{[k]} + \gamma \mathbf{w}_{\mathcal{A}_k}^{[k]}$$

   If $j^* \notin \mathcal{A}_k$ put the corresponding variable into the active set $\mathcal{A}_{k+1} = \mathcal{A}_k \cup \{j^*\}$, otherwise drop the corresponding variable from the active set $\mathcal{A}_{k+1} = \mathcal{A}_k - \{j^*\}$.

8. Set $\mathbf{r}^{[k+1]} = \mathbf{Y} - Z\mathbf{c}^{[k+1]}$, $k = k + 1$ and continue until $\gamma = 1$.

2.4.3 Criteria for choosing the regularization parameter

Since we are searching for the best response surface model, in term of approximation, a suitable regularization parameter λ must be picked. Several procedures exist to this end, such as the so-called C_p of Mallows' (Mallows (1973), Efron *et al.* (2004) and Zou *et al.* (2007)), the most relevant advantage of this criterion is that it does not require more computation beyond those used for obtaining the estimates. Unfortunately, such a rule requires an estimation of the error standard deviation. Since we assumed that the computer codes are deterministic, we cannot therefore use this criterion. We can also cite AIC and BIC criteria for choosing the regularization parameter. However, empiricaly it seems that the most adapted criteria for our problem is the v-fold-Cross-Validation. This criterion consists in splitting the observation set into v subsamples $\{S_1, \ldots, S_v\}$ roughly of equal size (figure 2.1). For each value of λ, the cross-validation procedure is defined as

1. For $i = 1, \ldots v$

   a. Build the response surface $\widehat{f}_\lambda^{[i]}$ with the training set made up of the observation set except the subsample S_i

   b. Compute the residual sum of squares RSS_i using the observation of the test set S_i.

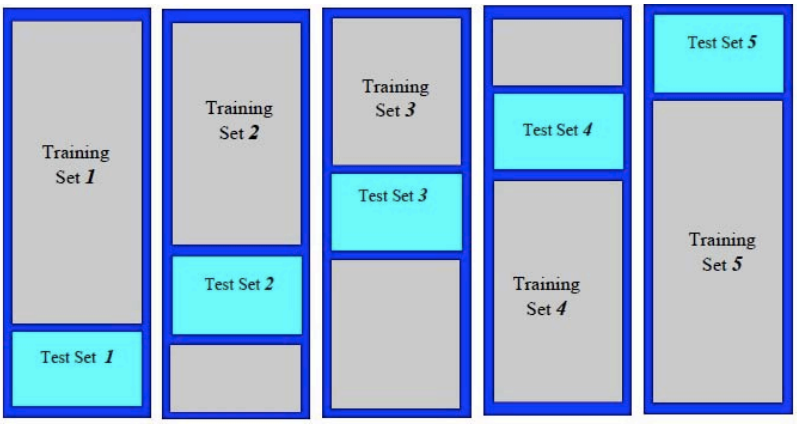

Figure 2.1: Test and training sets

2. Compute the mean residual sum of squares $1/v \sum_{i=1}^{v} RSS_i$

The optimal estimation of λ is λ^* which minimizes the mean residual sum of squares. Values of v between 5 and 10 produce satisfactory results. It is a fact that v-fold-Cross-Validation leads to penalty parameters that perform efficient regularization in nonparametric regression models.

2.4.4 Gaussian process

The Gaussian process (GP), also called Kriging, has been introduced in geostatistics by Matheron (1970). The idea to use this method to analyze a computer code was first proposed by Sacks *et al.* (1989). GP is a statistical method that treats the deterministic output as a realization of a random function, composed by the sum of a deterministic function and a centered stochastic process indexed by $\mathbf{x}$. The model can be written as:

$$S(\mathbf{x}) = \sum_{j=1}^{k} \beta_j h_j(\mathbf{x}) + Z(\mathbf{x}) \tag{2.33}$$

where the deterministic function $f(\mathbf{x}) = \sum_{j=1}^{k} \beta_j h_j(\mathbf{x})$ provides the mean approximation of the computer code and is a linear combination of pre-selected (known) real-valued functions $h_1, \ldots, h_k$, with unknown coefficients $\beta_1, \ldots, \beta_k$ (h_1 is usually a constant function). Z is assumed to be a centered Gaussian random process of covariance:

$$\text{cov}[\mathbf{x}, \mathbf{x}'] = E[Z(\mathbf{x})Z(\mathbf{x}')] = \tau^2 R(\mathbf{x}, \mathbf{x}') \tag{2.34}$$

where $\tau^2 = E[Z(\mathbf{x})^2]$ denotes the process variance, and $R(\mathbf{x}, \mathbf{x})$ is the correlation function. Denote by $\mathbf{X}$ the experimental design (the set $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$) and let $\mathbf{Y}$ be the corresponding

vector of outputs, a linear predictor of the output $f(\mathbf{x})$ of the computer code at a new point $\mathbf{x}$ is given by:

$$\widehat{S}(\mathbf{x}) = \mathbf{a}^T(\mathbf{x})\mathbf{Y}.$$

The coefficient vector $\mathbf{a}(\mathbf{x}) = (a_1(x), \ldots, a_n(x))^T$ is unknown. The best linear unbiased predictor for the deterministic response $f(\mathbf{x})$ is obtained by minimizing the mean square error

$$MSE[\widehat{S}(\mathbf{x})] = E[\mathbf{a}^T(\mathbf{x})\mathbf{Y} - S(\mathbf{x})]^2, \text{ subject to } E[\mathbf{a}^T(\mathbf{x})\mathbf{Y}] = E[S(\mathbf{x})] \tag{2.35}$$

where the constraint corresponds to unbiasedness.
Note that (2.35) can be written as

$$MSE[\widehat{S}(\mathbf{x})] = \tau^2[a^T(\mathbf{x})R\mathbf{a}(\mathbf{x}) - 2\mathbf{a}^T(\mathbf{x})\mathbf{r}(\mathbf{x}) + 1], \text{ subject to } H^T\mathbf{a}(\mathbf{x}) = \mathbf{h}(\mathbf{x}) \tag{2.36}$$

where

$$\begin{aligned}
\mathbf{h}(\mathbf{x}) &= (h_1(\mathbf{x}), h_2(\mathbf{x}), ..., h_k(\mathbf{x}))^T \\
H &= (h_j(\mathbf{x}_i))_{1\leq i\leq n; 1\leq j\leq k} \in \mathbb{R}^{n\times k} \\
R &= (R(\mathbf{x}_i, \mathbf{x}_l))_{1\leq i,l\leq n} \in \mathbb{R}^{n\times n} \\
\mathbf{r}(\mathbf{x}) &= (R(\mathbf{x}_1, \mathbf{x}), \ldots, R(\mathbf{x}_n, \mathbf{x}))^T.
\end{aligned}$$

R is the correlation matrix between the values of Z at the experimental design $\mathbf{X}$ and $\mathbf{r}(\mathbf{x})$ is the correlation vector between Z at $\mathbf{X}$ and the new point $\mathbf{x}$.
There exist different possible correlation functions that can be used. As discussed in the literature (see for instance Busby *et al.* (2007), Sacks *et al.* (1989) and Welch *et al.* (1992)), a commonly used correlation function is the generalized exponential, defined as

$$R(\mathbf{x}, \mathbf{y}) = \exp\left(-\sum_{j=1}^{d}\left(\frac{|x_j - y_j|}{\theta_j}\right)^{p_j}\right) \tag{2.37}$$

where $\theta_j > 0$ and $0 < p_j \leq 2$ are the correlation parameters. With $p_j = 1$ for $j = 1, \ldots, d$ (2.37) yields the exponential correlation function and $p_j = 2$ for $j = 1, \ldots, d$ yields the Gaussian correlation function.
By minimizing (2.36), the best linear unbiased prediction $\widehat{S}(\mathbf{x})$ can be written as

$$\widehat{S}(\mathbf{x}) = \mathbf{h}^T(\mathbf{x})\widehat{\boldsymbol{\beta}} + \mathbf{r}^T(\mathbf{x})R^{-1}(\mathbf{Y} - H\widehat{\boldsymbol{\beta}})$$

where

$$\widehat{\boldsymbol{\beta}} = (H^TR^{-1}H)^{-1}H^TR^{-1}\mathbf{Y}. \tag{2.38}$$

is the usual generalized least squares estimate of coefficients $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)$ in (2.33). The maximum likelihood estimation (MLE) is commonly used to determine the GP parameters $\boldsymbol{\theta}$ and p_j as well as the parameters σ and $\boldsymbol{\beta}$. The log-likelihood for the GP $S(\mathbf{x})$ is given by

$$l(\boldsymbol{\beta}, \tau, \boldsymbol{\theta}, p_j) \sim -1/2(m \ln \tau^2 + \ln \det(R) + (\mathbf{Y} - H\boldsymbol{\beta})^T R^{-1}(\mathbf{Y} - H\boldsymbol{\beta})/\sigma^2) \tag{2.39}$$

Note that for a fixed $\boldsymbol{\theta}$ and p_j the MLE for $\boldsymbol{\beta}$ is given by the generalized least squares estimates (2.38). Moreover, the MLE estimate for τ^2 is given by

$$\widehat{\tau}^2 = 1/n(\mathbf{Y} - H\widehat{\boldsymbol{\beta}})^T R^{-1}(\mathbf{Y} - H\widehat{\boldsymbol{\beta}}).$$

Using these estimates $\widehat{\boldsymbol{\beta}}$ and $\widehat{\tau}^2$ in (2.39), the parameters $\boldsymbol{\theta}$ and p_j are then determined by maximizing

$$l(\boldsymbol{\theta}, p_j) \sim -1/2(m \ln \tau^2 + \ln \det(R)). \tag{2.40}$$

Maximizing (2.40) is a global optimization problem, which became computationally expensive when dealing with high-dimensional models or when many experimental design points are available.

As in the traditional regression models (2.1) we can add an *iid* error term to the GP model (2.33). This can be written as

$$S(\mathbf{x}) = \sum_{j=1}^{k} \beta_j h_j(\mathbf{x}) + Z(\mathbf{x}) + \boldsymbol{\epsilon},$$

where $\boldsymbol{\epsilon}$ is a vector with *iid* $N(0, \sigma^2)$ random components. Since $Z(\mathbf{x})$ and $\boldsymbol{\epsilon}$ are assumed independent, the covariance function is obtained by adding the nugget effect term $\sigma^2 \boldsymbol{I}$

$$\operatorname{cov}[\mathbf{x}, \mathbf{x}] = \tau^2 R(\mathbf{x}, \mathbf{y}) + \sigma^2 \boldsymbol{I}.$$

The nugget effect may represent the measurement errors or the effects of non-deterministic computer codes. The introduction of a very small nugget effect can also be useful in the MLE computation to avoid numerical instability problems. Thus, it can be seen as a regularization parameter (see Pepelyshev (2010)).

2.5 Conclusion

We reviewed in this chapter the concept of sensitivity indices and the ways that may be used to estimate them and have seen that their estimation usually requires computationally intensive Monte Carlo simulations. In the framework of computer experiments and especially with the reservoir simulators, Monte-Carlo simulation based methods lead to intractable calculations. Therefore, in order to overcome this problem, we may replace the computer code by a response surface which may be viewed as an approximation and which is built by appropriate statistical regression methods.

Parametric regression methods have been briefly described and we discussed some of the most popular regularization methods (Ridge, LASSO and NNG), which permit to improve the accuracy of the estimates. These methods involve a regularization parameter whose choice is not straightforward. In addition, the later methods are also used in the nonparametric regression frameworks as we will see in the following chapters.

We also described one of the most popular nonparametric regression methods in the response surface frameworks, namely GP which has been empirically shown to often outperform other nonparametric methods for approximating deterministic computer codes. However, a GP process approach does suffer from its computational demand, especially when many experimental design points are available. Indeed, it requires the inversion of a correlation matrix whose inversion in terms of complexity scales with n^3 elementary operations. Moreover, computing the sensitivity indices by Monte-Carlo procedure described in section 2.2.2 and using the GP response surface can become computationally demanding for high dimensional problems.

In the next two chapters we will investigate some nonparametric regression methods based on ANOVA decomposition that seem to have been underused in the framework of response surface. In addition, we will show that the resulting family of estimates leads to a direct method for estimating the sensitivity indices.

Chapter 3

Component selection and smoothing operator based on iterative regularization algorithms

3.1 Introduction

Consider the mathematical model of the computer code:

$$Y = f(\mathbf{X}) \tag{3.1}$$

where Y is the output scalar of the simulator, $\mathbf{X} = (X^{(1)}, \ldots, X^{(d)})$ the d-dimensional inputs vector which represent the uncertain parameters of the simulator, $f : \mathbb{R}^d \to \mathbb{R}$ the unknown function that represent the computer code. Our purpose is to propose an estimation procedure for f.

A popular approach to the nonparametric estimation for high dimensional problems is the smoothing spline analysis of variance (SS-ANOVA) model (Wahba, 1990). To remind the ANOVA expansion is defined as

$$f(X) = f_0 + \sum_{j=1}^{d} f_j(X^{(j)}) + \sum_{j<l} f_{jk}(X^{(j)}, X^{(l)}) + ... + f_{1,...,d}(X^{(1)}, ..., X^{(d)}) \tag{3.2}$$

where f_0 is a constant, f_j's are univariate functions representing the main effects, f_{jl}'s are bivariate functions representing the two way interactions, and so on. Usually the high-order terms in the ANOVA expansion are negligible and a second-order expansion gives a satisfactory approximation of f.

A common approach to estimation in SS-ANOVA is the minimization of a penalized least

square functional. The goal of SS-ANOVA application is to determine which ANOVA components should be included in the model. Lin & Zhang (2006) proposed a penalized least square method with the penalty functional being the sum of component norms. The component selection and smoothing operator (COSSO) is a regularized nonparametric regression method based on ANOVA decomposition.

In the framework of the response surface, Storlie *et al.* (2009) have applied an adaptive version of COSSO (ACOSSO) for GSA application. This version was introduced in Storlie *et al.* (2011). However, ACOSSO is computationally more demanding than COSSO and in addition, after an empirical study we have remarked that the gain for the prediction accuracy of ACOSSO comparing to COSSO is not obvious for deterministic computer codes. So we will investigate the COSSO instead of ACOSSO.

In this chapter, we first review the (SS-ANOVA) model, then we will describe the COSSO method and its algorithm. Furthermore we will introduce two new algorithms which provide the COSSO estimates, the first one using an iterative algorithm based on Landweber iterations and the second one using the NN-LARS algorithm presented in the previous chapter. Next we will describe a new method to compute the sensitivity indices. Finally, numerical simulations will be presented and discussed.

3.2 Component selection and smoothing operator

3.2.1 Definition

Let $f \in \mathcal{F}$, where $\mathcal{F}$ is a reproducing kernel Hilbert space (RKHS) (for more details on RKHS we refer to the Appendix B) corresponding to the ANOVA decomposition (2.3) , and let $\mathcal{H}^j = \{1\} \oplus \bar{\mathcal{H}}^j$ be a RKHS of functions of $X^{(j)}$ over $[0,1]$, where $\{1\}$ is the RKHS consisting of only the constant functions and $\bar{\mathcal{H}}^j$ is the RKHS consisting of functions $f_j \in \mathcal{H}^j$ such that $\langle f_j, 1\rangle_{\mathcal{H}^j} = 0$. Then the model space $\mathcal{F}$ is the tensor product space of $\mathcal{H}^j$:

$$\mathcal{F} = \bigotimes_{j=1}^{d} \mathcal{H}^j = \{1\} \oplus \sum_{j=1}^{d} \bar{\mathcal{H}}^j \oplus \sum_{j<l} [\bar{\mathcal{H}}^j \otimes \bar{\mathcal{H}}^l] ... \tag{3.3}$$

Each component in the ANOVA decomposition (3.2) is associated to a corresponding subspace in the orthogonal decomposition (3.3). Generally, only second order interactions are considered in the ANOVA decomposition and an expansion to the second order generally provides a satisfactory description of the model.

Let consider the index $\alpha \equiv j$ for $\alpha = 1, \ldots, d$ with $j = 1, \ldots, d$ and $\alpha \equiv (j, l)$ for

$\alpha = d+1, \ldots, d(d+1)/2$ (where $d(d+1)/2$ correspond to the number of ANOVA components) with $1 \leq j < l \leq d$. With such notation in (3.3) when the expansion is truncated to include only interactions up to the second order:

$$\mathcal{F} = \{1\} \oplus \bigoplus_{\alpha=1}^{q} \mathcal{F}^{\alpha} = \{1\} \oplus \sum_{j=1}^{d} \bar{\mathcal{H}}^{j} \oplus \sum_{j<l} [\bar{\mathcal{H}}^{j} \otimes \bar{\mathcal{H}}^{l}] \tag{3.4}$$

where $\mathcal{F}^1, \ldots \mathcal{F}^q$ are q orthogonal subspaces of $\mathcal{F}$ and $q = d(d+1)/2$. We denote by $\| \cdot \|$ the norm in the RKHS $\mathcal{F}$. For some λ the SS-ANOVA estimate is given by the minimizer of

$$\frac{1}{n} \sum_{i=1}^{n} \{y_i - f(\mathbf{x}_i)\}^2 + \lambda^2 \sum_{\alpha=1}^{q} \theta_{\alpha}^{-1} \parallel P_{\alpha} f \parallel^2 \tag{3.5}$$

where P_α is the orthogonal projection onto $\mathcal{F}^\alpha$ and $\theta_\alpha \geq 0$. If $\theta_\alpha = 0$, then the minimizer of (3.5) is taken to satisfy $\parallel P_\alpha f \parallel = 0$ and we use the convention $0/0 = 0$.

The difference between COSSO and SS-ANOVA is that the first one penalizes the sum of the norms instead of the squared norms. Then, the COSSO estimate is given by the minimizer of

$$\frac{1}{n} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2 + \lambda^2 \sum_{\alpha=1}^{q} \parallel P_{\alpha} f \parallel \tag{3.6}$$

where λ is the regularization parameter. Since (3.6) is convex, the existence of the COSSO estimate is guaranteed by the following theorem (Lin & Zhang, 2006):

Theorem 3.2.1 *Let's $\mathcal{F}$ be a RKHS of functions over $[0,1]^d$. Assume that $\mathcal{F}$ can be decomposed as in (3.4). There exists a minimizer of (3.6).*

3.2.2 Algorithm

Lin & Zhang (2006) have shown that the minimizer of (3.6) has the form $\widehat{f} = \widehat{b} + \sum_{\alpha=1}^{q} \widehat{f}_\alpha$, with $\widehat{f}_\alpha \in \mathcal{F}^\alpha$. By the reproducing kernel property of $\mathcal{F}^\alpha$, $\widehat{f}_\alpha \in \text{span}\{K_\alpha(x_i, \cdot), i = 1, \ldots, n\}$, where K_α is the reproducing kernel of $\mathcal{F}^\alpha$. They also have shown that (3.6) is equivalent to a more easier form to compute, which is

$$\frac{1}{n} \sum_{i=1}^{n} \{y_i - f(\mathbf{x}_i)\}^2 + \lambda_0 \sum_{\alpha=1}^{q} \theta_{\alpha}^{-1} \parallel P_{\alpha} f \parallel^2 + \nu \sum_{\alpha=1}^{q} \theta_{\alpha}, \text{ subject to } \theta_\alpha \geq 0 \tag{3.7}$$

where λ_0 is a constant and ν is a smoothing parameter. The penalty term of θ's, $\sum_{\alpha=1}^{q} \theta_\alpha$, controlling the sparsity of each component f_α.

For fixed $\boldsymbol{\theta}$ (3.7) is equivalent to the SS-ANOVA and therefore the solution has the form:

$$f(\mathbf{x}) = b + \sum_{i=1}^{n} c_i \sum_{\alpha=1}^{q} \theta_\alpha K_\alpha(\mathbf{x}_i, \mathbf{x}) \tag{3.8}$$

Let K_α be the $n \times n$ matrix $\{K_\alpha(\mathbf{x}_i, \mathbf{x}_j))\}$,$i = 1, \ldots, n$, $j = 1, \ldots, n$, let K_θ stands for the matrix $\sum_{\alpha=1}^{q} \theta_\alpha K_\alpha$. Then $f = K_\theta \mathbf{c} + b1_n$ with $\mathbf{c} = (c_1, \ldots, c_n)^T$ and (3.7) can be expressed as

$$\frac{1}{n} \parallel \mathbf{Y} - \sum_{\alpha=1}^{q} \theta_\alpha K_\alpha \mathbf{c} - b1_n \parallel^2 + \lambda_0 \mathbf{c}^T K_\theta \mathbf{c} + \nu \sum_{\alpha=1}^{q} \theta_\alpha \tag{3.9}$$

For a fixed θ, (3.9) can be written as

$$\min_{\mathbf{c},b} \parallel \mathbf{Y} - K_\theta \mathbf{c} - b1_n \parallel^2 + n\lambda_0 \mathbf{c}^T K_\theta \mathbf{c} \tag{3.10}$$

which is a smooting spline problem (a quadratic minimization problem) and the solution satify:

$$(K_\theta + n\lambda_0 I)\mathbf{c} + b\mathbb{1}_n = \mathbf{Y} \tag{3.11}$$

$$\mathbb{1}_n^T \mathbf{c} = 0 \tag{3.12}$$

On the other hand, for fixed $\mathbf{c}$ and b, (3.9) can be written as

$$\min_{\theta} \parallel \mathbf{z} - D\boldsymbol{\theta} \parallel^2 + n\nu \sum_{\alpha=1}^{q} \theta_\alpha \text{ subject to } \theta_\alpha \geq 0 \tag{3.13}$$

where $\mathbf{z} = \mathbf{Y} - (1/2)n\lambda_0 \mathbf{c} - b1_n$ and D the $n \times q$ matrix with the αth column $\mathbf{d}_\alpha = K_\alpha \mathbf{c}$. Note that this formulation is similar to (2.31) which is the NNG estimate.
An equivalent form of (3.13) is given by

$$\min_{\theta} \parallel \mathbf{z} - D\boldsymbol{\theta} \parallel^2 + n\nu \sum_{\alpha=1}^{q} \theta_\alpha \quad \text{subject to } \theta_\alpha \geq 0 \text{ and } \sum_{\alpha=1}^{q} \theta_\alpha \leq M \tag{3.14}$$

for some $M \geq 0$. Lin & Zhang (2006) noted that the optimal M seems to be close to the number of important components. For computational consideration Lin and Zhang preferred to use (3.14) rather than (3.13).
Notice that the COSSO algorithm is a two step procedure. Indeed, it iterates between the smoothing spline (3.10) estimator, which gives a good initial estimate and the NNG (3.14) estimator, which is a variable selection procedure.
They also observed empirically that after one iteration the result is close to that at convergence. Thus the COSSO algorithm is presented as a one step update procedure:

1. Initialization: Fix $\theta_\alpha = 1$, $\alpha = 1, ..., q$

2. Tune λ_0 using v-fold-cross-validation.

3. Solve for $\mathbf{c}$ et b with (3.10).

4. For each fixed M in a chosen range, solve for $\boldsymbol{\theta}$ with (3.14) with the c and b, obtained in step 3. Tune M using v-fold-cross-validation. The θ's corresponding to the best M are the final solution at this step.

5. Tune λ_0 using v-fold-cross-validation.

6. With the new $\boldsymbol{\theta}$, solve for $\mathbf{c}$ and b with (3.10)

Notice that we have added step 5 respect to the original COSSO algorithm because we observed empirically that it improved the method's performance.

3.2.3 Kernel

The reproducing kernel K_α for the RKHS $\mathcal{F}^\alpha$ such as $\mathcal{F}^\alpha \equiv \bar{\mathcal{H}}^j$, are given by

$$K_\alpha(\mathbf{X}, \mathbf{X}') = K_j(X^{(j)}, X^{(j)'}) = k_1(X^{(j)})k_1(X^{(j)'}) + k_2(X^{(j)})k_2(X^{(j)'}) - k_4(|X^{(j)} - X^{(j)'}|)$$

where $k_l(x) = B_l(x)/l!$ and B_l is the lth Bernoulli polynomial. Thus, for $x \in [0, 1]$

$$\begin{aligned} k_1(x) &= x - \frac{1}{2} \\ k_2(x) &= \frac{1}{2}(k_1^2(x) - 1/12) \\ k_4(x) &= \frac{1}{24}(k_1^4(x) - \frac{k_1^2(x)}{2} + \frac{7}{240}) \end{aligned}$$

Moreover, the reproducing kernel K_α for the RKHS $\mathcal{F}^\alpha$ such as $\mathcal{F}^\alpha \equiv \bar{\mathcal{H}}^j \otimes \bar{\mathcal{H}}^l$, are given by the following tensor products

$$K_\alpha(\mathbf{X}, \mathbf{X}') = K_j(X^{(j)}, X^{(j)'})K_l(X^{(l)}, X^{(l)'})$$

For more details we refer to Wahba (1990).

3.3 An iterative projected shrinkage algorithm

We consider statistical models (least squares, LASSO, NNG), already mentioned in the previous chapter, which are convex optimization problems. A standard way to solve such problems is via a quadratic programming algorithm. Nevertheless, an iterative algorithm which is conceptually simple, easy to implement and involves no nested matrix inversion

has been proposed by several authors to estimate the solution of the LASSO regression problem, but it seems to have been largely ignored.
In the section that follows, we will first review some iterative algorithms available in the literature that can be used to solve convex optimization problems. Then, within the framework of these procedures we propose an iterative algorithm which estimates the solution of the NNG regression problem. Finally, some numerical results illustrate the performance of our proposed algorithm.

3.3.1 Some iterative optimization algorithms

3.3.1.1 The Landweber algorithm

A simple iterative algorithm have been proposed by Landweber (1951) to solve the linear regression problem $Y = \mathbf{X}\boldsymbol{\beta}$. It generates a sequence that approximates the true solution. The iterative procedure is recursively described as

$$\boldsymbol{\beta}^{[p+1]} = \boldsymbol{\beta}^{[p]} + X^T(\mathbf{Y} - X\boldsymbol{\beta}^{[p]}) \tag{3.15}$$

starting from an arbitrary $\boldsymbol{\beta}^{[0]}$. Each iteration of this algorithm only involves sums and matrix multiplication.
More recently, another version of the Landweber iterative algorithm has been introduced: the projected Landweber algorithm is defined as follows

$$\boldsymbol{\beta}^{[p+1]} = \mathcal{P}_\Omega(\boldsymbol{\beta}^{[p]} + \mathbf{X}^T(Y - \mathbf{X}\boldsymbol{\beta}^{[p]})) \tag{3.16}$$

starting from an arbitrary $\boldsymbol{\beta}^{[0]}$, where $\mathcal{P}_\Omega$ is the orthogonal projection onto a closed convex sets Ω that describes eventual constraints on $\boldsymbol{\beta}$. This algorithm converges to a minimizer of the constrained least square problem when the constraints are expressed in terms of a convex and closed set Ω:

$$\| Y - \mathbf{X}\boldsymbol{\beta} \|_n^2, \text{ subject to } \boldsymbol{\beta} \in \Omega$$

with Ω a given convex and closed subset of $\mathbb{R}^d$. The convergence properties of the projected Landweber algorithm has been investigated in Eicke (1992), Byrne (2002). It also can be easily implemented if the projection operator $\mathcal{P}_\Omega$, can be easily computed.

3.3.1.2 The iterative shrinkage/thresholding algorithm

Consider now the LASSO regression problem. In recent years, several authors have proposed, in different frameworks, an iterative soft-thresholding algorithm to approximate the $\boldsymbol{\beta}^{LASSO}$ (among them Daubechies *et al.* (2004), Friedman *et al.* (2007) and Figueiredo & Nowak (2003)). Assume that the design matrix has been normalized, so $\parallel X^T X \parallel < 1$, in other words $\lambda_{max}(X^T X) < 1$ (where λ_{max} is the maximum eigenvalue) the iterative shrinkage/thresholding algorithm (IST) is defined as

$$\boldsymbol{\beta}^{[p+1]} = \delta_{\lambda}^{soft}(\boldsymbol{\beta}^{[p]} + X^T(\mathbf{Y} - X\boldsymbol{\beta}^{[p]})) \tag{3.17}$$

starting from an initial estimate $\boldsymbol{\beta}^{[0]}$, where δ_{λ}^{Soft} is the soft-thresholding function defined as

$$\delta_{\lambda}^{Soft}(\widehat{\boldsymbol{\beta}}) = \begin{cases} 0 & \text{if } \mid \widehat{\boldsymbol{\beta}} \mid \leq \lambda \\ \widehat{\boldsymbol{\beta}} - \lambda & \text{if } \widehat{\boldsymbol{\beta}} > \lambda \\ \widehat{\boldsymbol{\beta}} + \lambda & \text{if } \widehat{\boldsymbol{\beta}} < -\lambda \end{cases} \tag{3.18}$$

A rigorous convergence proof for this algorithm is provided in Daubechies *et al.* (2004).

3.3.2 Iterative projected shrinkage algorithm

3.3.2.1 Definition

Consider the (3.13) regression problem:

$$\min_{\theta} \parallel \mathbf{z} - D\boldsymbol{\theta} \parallel^2 + n\nu \sum_{\alpha=1}^{q} \theta_{\alpha} \;\; \text{subject to} \;\; \theta_{\alpha} \geq 0$$

The functional (3.13) is convex since the matrix $D^T D$ is symmetric and positive semidefinite and since the constraints $\theta_{\alpha} > 0$ define also a convex feasible set. For the convex optimization problem, the Karush-Kuhn-Tucker (KKT) conditions are necessary and sufficient for the optimal solution $\boldsymbol{\theta}^*$, where $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \parallel z - D\theta \parallel^2 + n\nu \sum_{\alpha=1}^{q} \theta_{\alpha}$ subject to $\theta_{\alpha} \geq 0$. This KKT conditions are defined as

$$\begin{aligned} \{-\mathbf{d}_{\alpha}^T(\mathbf{Y} - D\boldsymbol{\theta}^*) + \nu\}\theta_{\alpha}^* &= 0 \\ -\mathbf{d}_{\alpha}^T(\mathbf{Y} - D\boldsymbol{\theta}^*) + \nu &\geq 0 \\ \theta_{\alpha}^* &\geq 0 \end{aligned}$$

which is equivalent to

$$-\mathbf{d}_{\alpha}^T(\mathbf{Y} - D\boldsymbol{\theta}^*) + \nu = 0, \text{if } \theta_{\alpha}^* \neq 0 \tag{3.19}$$

$$-\mathbf{d}_{\alpha}^T(\mathbf{Y} - D\boldsymbol{\theta}^*) + \nu > 0, \text{if } \theta_{\alpha}^* = 0 \tag{3.20}$$

where $\mathbf{d}_\alpha$ denotes the αth column of D. Therefore, from (3.19) and (3.20) we can derive the fixed-point equation:

$$\boldsymbol{\theta}^* = \mathcal{P}_{\Omega^+}(\delta_\nu^{Soft}(\boldsymbol{\theta}^* + D^T(\mathbf{Y} - D\boldsymbol{\theta}^*))) \tag{3.21}$$

where $\mathcal{P}_{\Omega^+}$ is the nearest point projection operator onto the nonnegative orthant (closed convex set) $\Omega^+ = \{x : x \geq 0\}$ Thus, in the framework of the projected Landweber and of the iterative thresholding algorithms, we propose an iterative algorithm, which is defined by

$$\boldsymbol{\theta}^{[p+1]} = \mathcal{P}_{\Omega^+}(\delta_\nu^{Soft}(\boldsymbol{\theta}^{[p]} + D^T(\mathbf{Y} - D\boldsymbol{\theta}^{[p]}))) \tag{3.22}$$

We named this algorithm the iterative projected shrinkage algorithm (IPS). The following theorem concerns the convergences of IPS algorithm:

Theorem 3.3.1 *IPS algorithm defined by (3.22) converge to the solution of (3.13), whenever such solution exists, for any starting vector* $\boldsymbol{\theta}^{[0]}$.

The proof of this theorem can be found in Appendix A. Note that we have assumed that $\lambda_{max}(D^T D) \leq 1$ (where λ_{max} is the maximum eigenvalue). Otherwise we solve the equivalent minimization problem

$$\min_{\boldsymbol{\theta}} \| \frac{\mathbf{z}}{c} - \frac{D}{c}\boldsymbol{\theta} \|^2 + \frac{n\nu}{c} \sum_{\alpha=1}^{q} \theta_\alpha \;\text{ subject to }\; \theta_\alpha \geq 0$$

where the positive constant c ensures that $\lambda_{max}(D^T D) \leq 1$.

3.3.2.2 Stopping conditions

IPS algorithm is an iterative procedure (3.22) which produces a sequence of solutions $\boldsymbol{\theta}^{[0]}, \boldsymbol{\theta}^{[0]}, \ldots, \boldsymbol{\theta}^{[p]}$ converging to the optimal solution $\boldsymbol{\theta}^*$. There is a need to stop the algorithm when the solution $\boldsymbol{\theta}^{[p]}$ is sufficiently close to the optimal solution $\boldsymbol{\theta}^*$. Several stopping conditions have been proposed in the literature (for example M. Defrise (1987)). We choose to use a stopping condition based on the KKT conditions, which are easy to evaluate. This $\epsilon-$KKT conditions are defined as

$$\mathbf{d}_\alpha^T(\mathbf{Y} - D\boldsymbol{\theta}^*) = \nu - \epsilon, \text{if } \theta_\alpha^* \neq 0$$
$$\mathbf{d}_\alpha^T(\mathbf{Y} - D\boldsymbol{\theta}^*) \leq \nu - \epsilon, \text{if } \theta_\alpha^* = 0$$

where $\epsilon > 0$ is a constant which defines the precision of the solution.

3.3.2.3 Accelerated iterative projected shrinkage algorithm

In practice, slow convergence, particularly when D is ill-conditioned or ill-posed, is an obstacle to a wide use of this method in spite of the good results provided in many cases. Indeed, IPS procedure is a composition of the projected thresholding with the Landweber iteration algorithm, which is a gradient descent algorithm with a fixed step size, known to converge usually slowly. Unfortunately, combining the Landweber iteration with the projected thresholding operation does not accelerate the convergence, especially with a small value of ν. Several authors proposed different methods to accelerate the various algorithms (Landweber, projected Landweber and IST), among them Piana & Bertero (1997), Daubechies *et al.* (2008) and Bioucas-Dias & Figueiredo (2007) , the later brought an efficient procedure, named two-step IST (TwIST), which has faster convergence rates than IST especially for ill-conditioned problems. This procedure is defined as

$$\boldsymbol{\theta}^{[1]} = \delta_{\nu}^{Soft}(\boldsymbol{\theta}^{[0]}) \tag{3.23}$$

$$\boldsymbol{\theta}^{[p+1]} = (1-\alpha)\boldsymbol{\theta}^{[p-1]} + (\alpha-\beta)\boldsymbol{\theta}^{[p]} + \beta\delta_{\nu}^{Soft}(\boldsymbol{\theta}^{[p]} + D^T(\mathbf{Y} - D\boldsymbol{\theta}^{[p]})) \tag{3.24}$$

Observing the equivalence between (3.24) and (3.17) with $\alpha = \beta = 1$, we propose to modify TwIST so it converge to the solution of (3.13). Thus we replace δ_{ν}^{Soft} by $\mathcal{P}_{\Omega^+}(\delta_{\nu}^{Soft})$ in (3.23) and in (3.24). The accelerated projected iterative shrinkage thresholding algorithm is defined as

$$\boldsymbol{\theta}^{[1]} = \mathcal{P}_{\Omega^+}(\delta_{\nu}^{Soft}(\boldsymbol{\theta}^{[0]})) \tag{3.25}$$

$$\boldsymbol{\theta}^{[p+1]} = (1-\alpha)\boldsymbol{\theta}^{[p-1]} + (\alpha-\beta)\boldsymbol{\theta}^{[p]} + \beta\mathcal{P}_{\Omega^+}(\delta_{\nu}^{Soft}(\boldsymbol{\theta}^{[p]} + D^T(\mathbf{Y} - D\boldsymbol{\theta}^{[p]}))) \tag{3.26}$$

In accordance with Theorem 4 given by Bioucas-Dias & Figueiredo (2007) the parameters α and β are set to

$$\alpha = \widehat{\rho}^2 + 1$$

$$\beta = 2\alpha/(1+\zeta)$$

where $\widehat{\rho} = (1-\sqrt{\zeta})/(1+\sqrt{\zeta})$ and $\zeta = \lambda_{min}(D^TD)$ (where λ_{min} is the minimal eigenvalue) if $\lambda_{min}(D^TD) \neq 0$, or $\zeta = 10^{-\kappa}$ with $\kappa = 1, \ldots, 4$ need to be tuned by running a few iterations. The condition $\kappa = 1$ correspond to mildly ill-conditioned problems and $\kappa = 4$ for severely ill-conditioned problems. For more detail about the choice of these parameters we refer to Bioucas-Dias & Figueiredo (2007).

3.3.3 COSSO based on the IPS algorithm

In the previous section we have developed an iterative algorithm to solve like (3.13) regression problems. Indeed, instead of iterating between (3.10) and (3.14) in the COSSO algorithm we will iterate between (3.10) and (3.13). Thus the COSSO algorithm based on IPS (or AIPS) can summarized as:

1. Initialization: Fix $\theta_\alpha = 1$, $\alpha = 1, ..., q$
2. Tune λ_0 using v-fold-cross-validation.
3. Solve for $\mathbf{c}$ et b with (3.10).
4. For each fixed ν, solve (3.13) by using the IPS (or AIPS) algorithm with the c and b, obtained in step 3. Tune ν using v-fold-cross-validation. The θ's corresponding to the best ν are the final solution at this step.
5. With the new $\boldsymbol{\theta}$ tune λ_0 using v-fold-cross-validation.
6. With the new $\boldsymbol{\theta}$ and λ_0, solve for $\mathbf{c}$ and b with (3.10)

Note that it can be shown that $\boldsymbol{\theta} = 0$ for $\nu \geq \nu_{max}$, with $\nu_{max} \equiv \max_\alpha \mid \mathbf{d}_\alpha^T \mathbf{Y} \mid$. Hence, the value of ν, which needs to be estimated, is bounded by ν_{max} and ν_{min}, with ν_{min} small enough. Then, ν is tuned by v-fold-cross-validation.

3.4 COSSO based on NN-LARS algorithm

Previously we have noted that the COSSO is a two step procedure that iterates between the smoothing spline and the NNG. In Chapter 2 we presented an algorithm (NN-LARS) which provides the entire path for the NNG coefficients. To apply NN-LARS we assume that $\mathbf{Z}_j = \mathbf{X}_j \beta_j^{LS} = \mathbf{d}_j^T$, which means that we substitute the step 1 of the NN-LARS algorithm by:

1. Set $D = Z$

Thus the COSSO algorithm based on NN-LARS is presented as

1. Initialization: Fix $\theta_\alpha = 1$, $\alpha = 1, ..., q$
2. Tune λ_0 using v-fold-cross-validation.

3. Solve for $\mathbf{c}$ et b with (3.10).

4. Solve (3.13) by using the NN-LARS algorithm with the $\mathbf{c}$ and b, obtained in step 3. Choose the best model using v-fold-cross-validation. The θ's corresponding to the best model are the final solution at this step.

5. With the new $\boldsymbol{\theta}$ tune λ_0 using v-fold-cross-validation.

6. With the new $\boldsymbol{\theta}$ and λ_0, solve for $\mathbf{c}$ and b with (3.10)

Notice that even if the NN-LARS algorithm provides the entire solution path the choice of the best model (as we will empirically show later) becomes computationally expensive for a high dimensional problem.

3.5 Global sensitivity analysis by COSSO

It has been shown in the previous chapter that, when the input vector components are independently distributed (and $\mathbf{X} \in [0,1]^d$), the component functions in the ANOVA decomposition are orthogonal and contain relevant information on the input/output relationships. Moreover, the total variance V of the model can be decomposed into its input variable contributions. Using the variance decomposition (2.9) and the COSSO solution form (3.8) we have

$$V \approx \sum_{j=1}^{d} V_j + \sum_{1\leq j<l\leq d} V_{jl} \tag{3.27}$$

$$\approx \sum_{\alpha=1}^{q} \int_0^1 \left[\theta_\alpha \sum_{i=1}^{n} c_i K_\alpha(\mathbf{x}_i, \mathbf{X})\right]^2 dX^{(\alpha)} \tag{3.28}$$

where $dX^{(\alpha)} \equiv dX^{(\alpha)}$ for $\alpha = 1, \ldots, d$ and $dX^{(\alpha)} \equiv dX^{(j)}dX^{(l)}$ for $\alpha = d+1, \ldots, q$ with $1 \leq j < l \leq d$.
Let us consider a N i.i.d random sample from the distribution of $\mathbf{X}$, say $\{\mathbf{z}_i = (z_{i_1}, \ldots, z_{i_d})^T, i = 1, \ldots, N\}$. The Monte-Carlo estimate of V_j is given by

$$\widehat{V}_j = \frac{1}{N} \sum_{m=1}^{N} \left[\theta_j \sum_{i=1}^{n} c_i K_j(x_{i_j}, z_{m_j})\right]^2 \tag{3.29}$$

Hence the main effect indices (first order sensitivity indices) are estimated as

$$\widehat{S}_j = \frac{\widehat{V}_j}{\widehat{V}} \tag{3.30}$$

where $\widehat{V}$ is the total variance estimation. The estimation of V_{jl} are given by

$$\widehat{V}_j = \frac{1}{N}\sum_{m=1}^{N}\left[\theta_{jl}\sum_{i=1}^{n} c_i K_j(x_{i_j}, z_{m_j}) K_l(x_{i_l}, z_{m_l})\right]^2 \tag{3.31}$$

Thus, the second order indices are estimated by

$$\widehat{S}_{jl} = \frac{\widehat{V}_{jl}}{\widehat{V}} \tag{3.32}$$

Since we assume that a truncated form of ANOVA decomposition provides a satisfactory approximation of the model, the total effect indices estimation is given by

$$\widehat{S}_{T_j} = \widehat{S}_j + \sum_{l\neq j}\widehat{S}_{jl} \tag{3.33}$$

Notice that to compute all the indices (main effect, interaction and total effect) we need only N evaluations of the response surface.

3.6 Simulations

The present section is focused on studying the empirical performance of the four different versions of COSSO estimate and compares it to the GP method. The four version of COSSO are COSSO-IPS, COSSO-AIPS, COSSO-NN-LARS and COSSO-solver which use a standard convex optimizer (matlab code developed by the COSSO's authors Lin & Zhang (2006)).

The empirical performance of estimators will be measured in terms of prediction accuracy and global sensitivity analysis (GSA). The measure of accuracy is given by Q_2 defined as

$$Q_2 = 1 - \frac{\sum_{i=1}^{n_{test}}(y_i - \widehat{f}(\mathbf{x}_i))^2}{\sum_{i=1}^{n_{test}}(y_i - \bar{y})^2}, \text{with } n_{test} = 1000 \tag{3.34}$$

where y_i denotes the ith test observation of the test set, $\bar{y}$ is their empirical mean and $\widehat{f}(\mathbf{x}_i)$ is the predicted value at the design point $\mathbf{x}_i$. We also compare the methods for different experimental design sizes, uniformly distributed on $[0,1]^d$ and built by maximinLHD procedure. For each setting of each test example, we run 50 times and average. Thus we define the quantity $\bar{Q}_2 = 1/50\sum_{k=1}^{50} Q_2^k$.

Concerning the performance in terms of GSA, we will study the accuracy of the total effect indices estimation. Furthermore, we will study the size effect of the sample used to estimate the total effect indices by Monte-Carlo integration. We will compare the results

to those obtained by Sobol's method described in the previous chapter and combined with the GP.

To fit COSSO models using a standard convex optimizer we have used the matlab code developed by the COSSO's authors Yi Lin and Hao Helen Zhang. COSSO-IPS, COSSO-AIPS and COSSO-NN-LARS are adapted versions of the original matlab code. The GP models are done in R, with IFP code contributed by COUGAR's team.

3.6.1 Example 1

Consider the g-Sobol function which is strongly nonlinear and is described by a non-monotonic relationship. Because of its complexity and the availability of analytical sensitivity indices, this function is a well known test case in the studies of GSA. Figure 3.1 illustrates the g-Sobol function against the two most influential parameters $X^{(1)}$ and $X^{(2)}$. The g-Sobol function (Saltelli *et al.* (2000)) is defined for 8 inputs factors as

$$g_{\text{Sobol}}(X^{(1)},\dots,X^{(8)}) = \prod_{k=1}^{8} g_k(X^{(k)}) \quad \text{with} \quad g_k(X^{(k)}) = \frac{\left|4X^{(k)}-2\right| + a_k}{1+a_k}$$

where $\{a_1,\dots,a_8\} = \{0,1,4.5,9,99,99,99,99\}$. 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:

$$\begin{cases} a_k = 0 \rightarrow x^{(k)} \text{ is very important,} \\ a_k = 1 \rightarrow x^{(k)} \text{ is relatively important,} \\ a_k = 4.5 \rightarrow x^{(k)} \text{ is poorly important,} \\ a_k = 9 \rightarrow x^{(k)} \text{ is non important,} \\ a_k = 99 \rightarrow x^{(k)} \text{ is non significant.} \end{cases}$$

The analytical values of Sobol's indices are given by (Sudret, 2008)

$$V_j = \frac{1}{3(1+a_j)^2}, \qquad V = \prod_{k=1}^{d}(V_k+1) - 1,$$

$$S_{j_1,\dots,j_s} = \frac{1}{V}\prod_{k=1}^{s} V_k$$

where $1 \le j_1 < \dots < j_s \le d$ and $s = 1,\dots,d$. The analytical values of the total effect indices are shown in table (3.1).

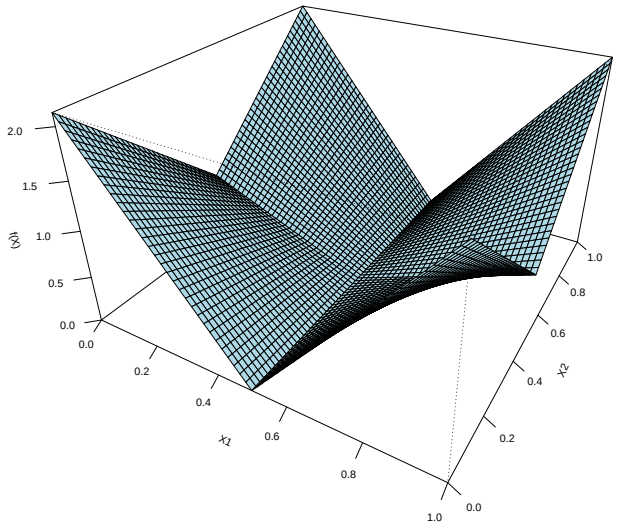

Figure 3.1: Plot of g-Sobol function versus inputs $X^{(1)}$ and $X^{(2)}$ with other inputs fixed at 0.5

Input	Total effect
$X^{(1)}$	0.787
$X^{(2)}$	0.242
$X^{(3)}$	0.034
$X^{(4)}$	0.011
$X^{(5,\ldots,8)}$	0

Table 3.1: Analytical values of the total effect indices of the g-Sobol function

3.6.1.1 Assessment of the prediction accuracy

Table 3.2 summarizes the results for the 50 realizations of the g-Sobol model with three different experimental design sizes ($n = 100$, $n = 200$ and $n = 400$). It appears that for this example the GP outperforms all of the COSSO versions for $n = 100$ and $n = 200$. However, when the experimental design size increases, the performance of the GP does not get much better while all the COSSO methods increase their accuracy by increasing the sample size. Indeed, for $n = 400$ the COSSO methods outperforms GP especially COSSO-NN-LARS, COSSO-AIPS and COSSO-solver which have $\bar{Q}_2$ quantity equal to 0.99 which indicates that those response surfaces explain 99% of the model variance. All

the COSSO versions provide quite similar result for this example. Moreover, as expected, the AIPS method is clearly faster than IPS. Notice that even if the NN-LARS provides the entire path of the solution, the COSSO-NN-LARS method has the same computational cost as COSSO-IPS and COSSO-solver, the reason of that is the choice of the best model by v-fold-cross-validation which is computationally costly.

	n	$\bar{Q}_2$	time (s)
COSSO-NN-LARS	100	0.86(0.03)	4
	200	0.91(0.02)	14
	400	0.99(0.01)	59
COSSO-IPS	100	0.82(0.08)	28
	200	0.90(0.01)	45
	400	0.97(0.02)	195
COSSO-AIPS	100	0.84(0.07)	6
	200	0.90(0.01)	15
	400	0.99(0.01)	53
COSSO-solver	100	0.85(0.06)	8
	200	0.90(0.01)	18
	400	0.99(0.01)	59
GP	100	0.93(0.01)	29
	200	0.96(0.01)	86
	400	0.95(0.01)	342

Table 3.2: Q_2 results from the g-Sobol function. The estimated standard deviation of Q_2 is given in parentheses.

3.6.1.2 Global sensitivity analysis

In this subsection, we apply the COSSO-AIPS method in order to estimate the total effect indices. The choice of COSSO-AIPS instead of other COSSO was motivated by the good performance of the method and it fast execution. We first focus on the robustness of the size effect of the sample used to estimate the indices. To this end, we repeated the experiment 100 times with two different sample size $N = 500$ and $N = 5000$ built using maximinLHD. We estimate the indices using a response surface build by COSSO-AIPS of an experimental design of size $n = 400$ and having a Q_2 equal to 0.99. We compare the results to those obtained by Sobol's method of indices estimation based on response surface

build by GP on an experimental design of size $n = 400$ and having a Q_2 equal to 0.96. As introduced previously Sobol's methods to estimate the total effect needs 2 samples, thus we build, using a maximinLHD procedure, 200 samples of two sizes: $N = 500$ and $N = 5000$. Figure 3.2 summarizes the results for the 100 different samples and the two sizes ($N = 500$ and $N = 5000$). Each panel is a boxplot of the 100 estimations of the total effect index $\widehat{S}_{T_j}$, $j = 1, \ldots, 8$. Dashed lines are drawn at the corresponding analytical values of the total effects indices. We see that our direct method of indices estimation based on COSSO procedure is more robust than Sobol's one using the GP response surface especially when the sample size is small ($N = 500$). Moreover our method needs only N evaluation of the COSSO-AIPS response surface while Sobol's method needs $2Nd$ evaluations of GP response surface (for $N = 5000$, 80000 evaluations are used).

To study the performance of the total effect indices estimations versus the sizes of the experimental design we compute the indices, using sample of size $N = 5000$, for each of the 50's realizations and for the three different experimental design sizes ($n = 100$, $n = 200$ and $n = 400$). Figure 3.3 summarizes the results, each panel is a boxplot of the 50 estimations of $\widehat{S}_{T_j}$, $j = 1, \ldots, 8$. Dashed lines are drawn at the corresponding analytical values of the total effects indices. As expected the estimations based on GP response surface outperforms those based on COSSO-AIPS for $n = 100$ and $n = 200$, which is due to the better performances in terms of Q_2 of the GP for these experimental design sizes. Nevertheless, for $n = 400$ the estimations based on COSSO-AIPS are better than those based on GP.

3.6.2 Example 2

Let consider the same example that has been used in the COSSO paper (Example 3). This 10 dimensional regression problem is defined as

$$f(\mathbf{X}) = g_1(X^{(1)}) + g_2(X^{(2)}) + g_3(X^{(3)}) + g_4(X^{(4)}) + g_1(X^{(3)} + X^{(4)}) + g_2(\frac{X^{(1)}X^{(3)}}{2}) + g_3(X^{(1)}X^{(2)}) \tag{3.35}$$

where

$$g_1(t) = t; \quad g_2(t) = (2t-1)^2; \quad g_3(t) = \frac{\sin(2\pi t)}{2 - \sin(2\pi t)};$$
$$g_4(t) = 0.1\sin(2\pi t) + 0.2\cos(2\pi t) + 0.3\sin^2(2\pi t) + 0.4\cos^3(2\pi t) + 0.5\sin^3(2\pi t)$$

Therefore $X^{(5)}, ..., X^{(8)}$ are uninformative. This analytical model is fast enough to evaluate so we can calculate the total effect indices with great precision. Thus the reference values

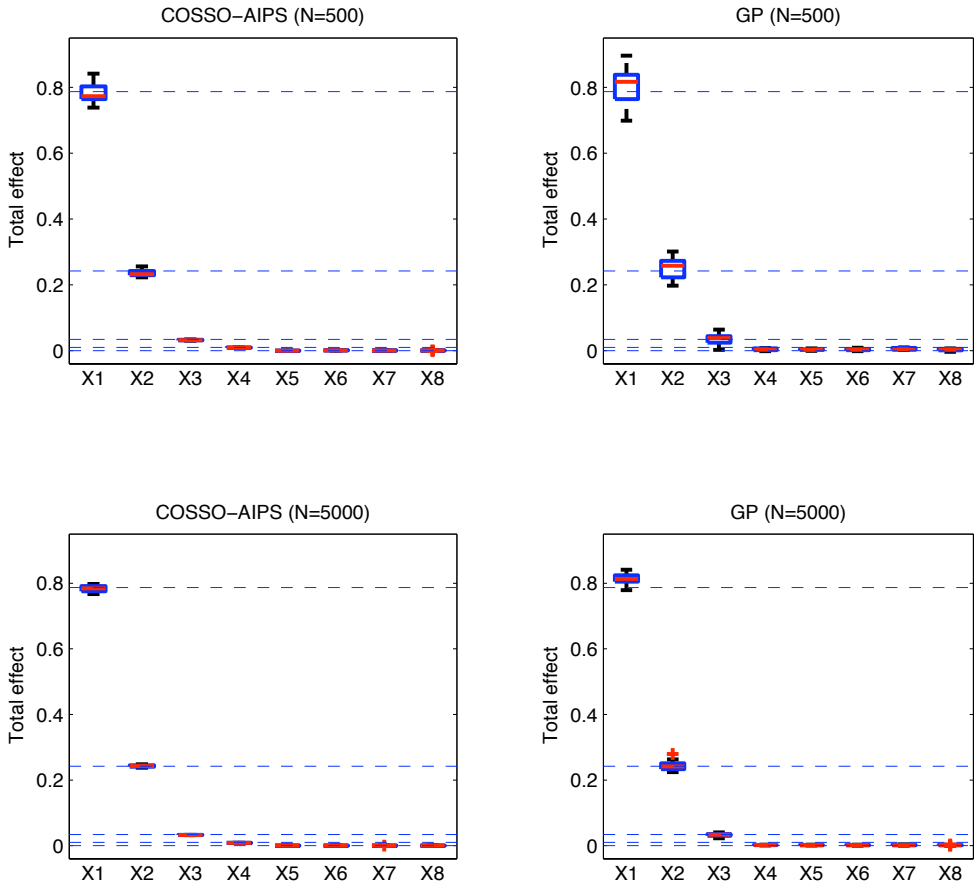

Figure 3.2: Total effect indices vs. sample effect (example 1)

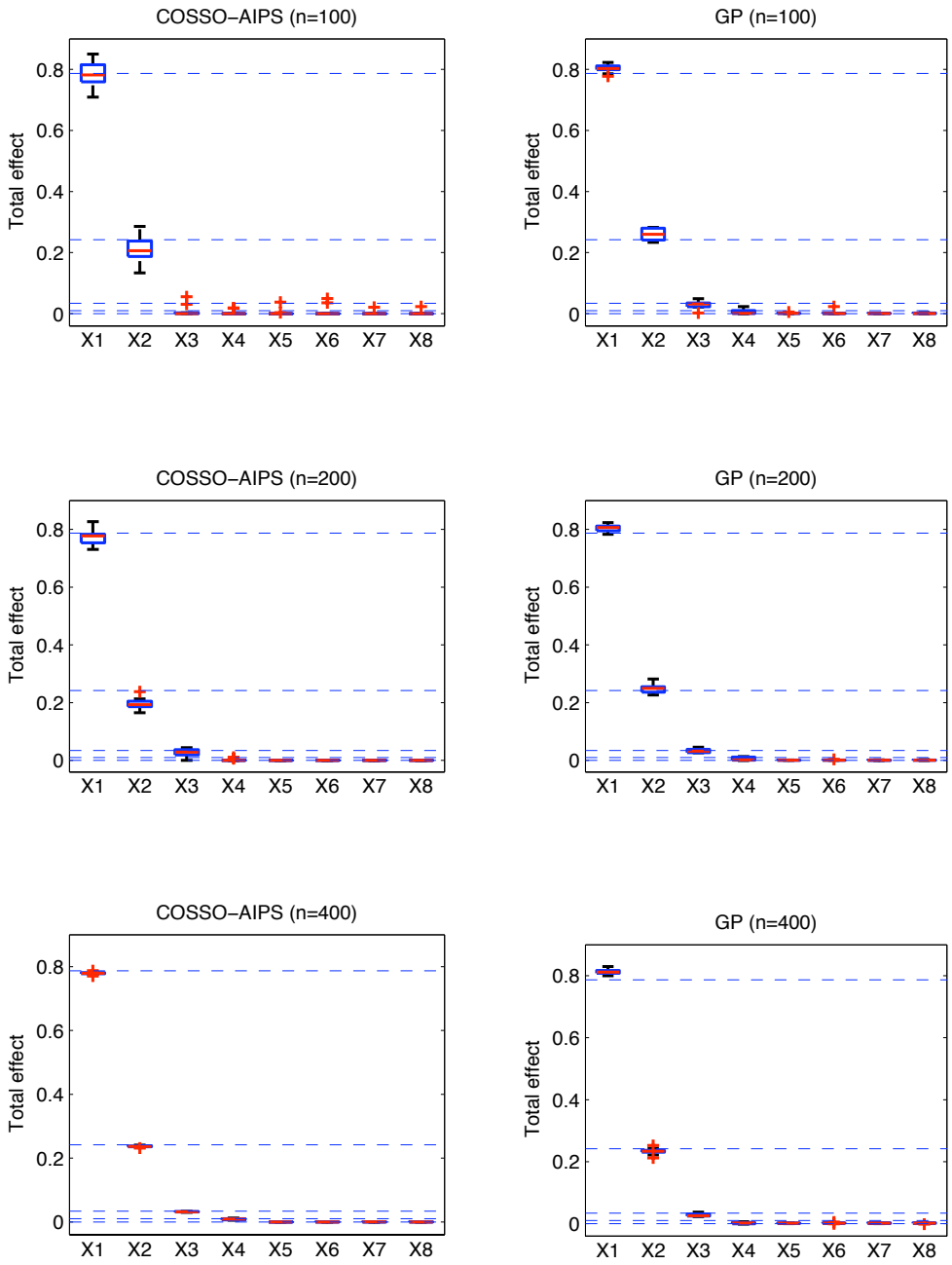

Figure 3.3: Total effect indices vs. experimental design size effect (example 1)

of the indices are computed by direct Monte-Carlo simulation using Sobol's method (with $N = 250000$, which correspond to 5.10^6 evaluations of the example 2). Table 3.3 shows 95% confidence intervals (95% CI) provided by 100 different samples and the chosen reference values

Input	Total effect 95% CI	Reference value
$X^{(1)}$	$[0.343, 0.346]$	0.344
$X^{(2)}$	$[0.213, 0.215]$	0.214
$X^{(3)}$	$[0.285, 0.288]$	0.286
$X^{(4)}$	$[0.377, 0.380]$	0.379
$X^{(5,\ldots,10)}$	0	0

Table 3.3: 95% CI and the reference values of the total effect indices for the example 2

3.6.2.1 Assessment of the prediction accuracy

Table 3.4 summarizes the results for the 50 realizations of the example 2 model with three different experimental design sizes ($n = 100$, $n = 200$ and $n = 400$). Here we see that for all versions and for all sizes of experimental designs the COSSO method outperforms GP. The accuracy for all methods improves as the experimental design increases. Notice that the COSSO-AIPS method is the fastest one, especially with a large experimental design size as opposed to the GP which is the slowest method.

3.6.2.2 Global sensitivity analysis

As in the previous subsection, we apply the COSSO-AIPS method in order to estimate the total effect indices. We first focus on the size effect of the sample used to estimate the indices. Thus we build, using a maximinLHD procedure, 100 samples of two sizes: $N = 500$ and $N = 5000$; then we estimate the indices using a response surface built by COSSO-AIPS of an experimental design of size $n = 400$ and having a Q_2 equal to 0.99. We compare the results to those obtained by Sobol's method of indices estimation based on response surface built by GP on an experimental design of size $n = 400$ and having a Q_2 equal to 0.95. We build, using a maximinLHD procedure, 200 samples of two sizes: $N = 500$ and $N = 5000$.

Figure 3.6 shows the results obtained by the 100 different samples and for the two sizes ($N = 500$ and $N = 5000$). Each panel is a boxplot of the 100 estimations of the total

	n	$\bar{Q}_2$	time (s)
COSSO-NN-LARS	100	0.80(0.09)	6
	200	0.94(0.03)	30
	400	0.99(0.01)	118
COSSO-IPS	100	0.82(0.08)	27
	200	0.95(0.02)	50
	400	0.99(0.01)	140
COSSO-AIPS	100	0.82(0.08)	7
	200	0.94(0.02)	22
	400	0.99(0.01)	84
COSSO-solver	100	0.82(0.08)	19
	200	0.93(0.03)	37
	400	0.98(0.01)	110
GP	100	0.76(0.03)	25
	200	0.88(0.02)	95
	400	0.94(0.02)	490

Table 3.4: Q_2 results from example 2. The estimated standard deviation of Q_2 is given in parentheses.

effect indices $\widehat{S}_{T_j}$, $j = 1, \ldots, 10$. Dashed lines are drawn at the corresponding reference values of the total effects indices. We see that our direct method of indices estimation based on COSSO method is more robust than Sobol's one using the GP response surface especially when the sample size is small ($N = 500$).

A summary of the indices estimation on 50 realizations and for the three different experimental design size ($n = 100$, $n = 200$ and $n = 400$) is shown in figure 3.5. Each panel is a boxplot of the 50 estimations of $\widehat{S}_{T_j}$, $j = 1, \ldots, 10$. Dashed lines are drawn at the corresponding analytical values of the total effects indices. It appears that the indices estimation using COSSO-AIPS suffers more from the small experimental design sizes than GP, especially for those indices corresponding to the uninformative inputs. However, as the sample size increases, our COSSO-AIPS method have performs better than Sobol's with GP.

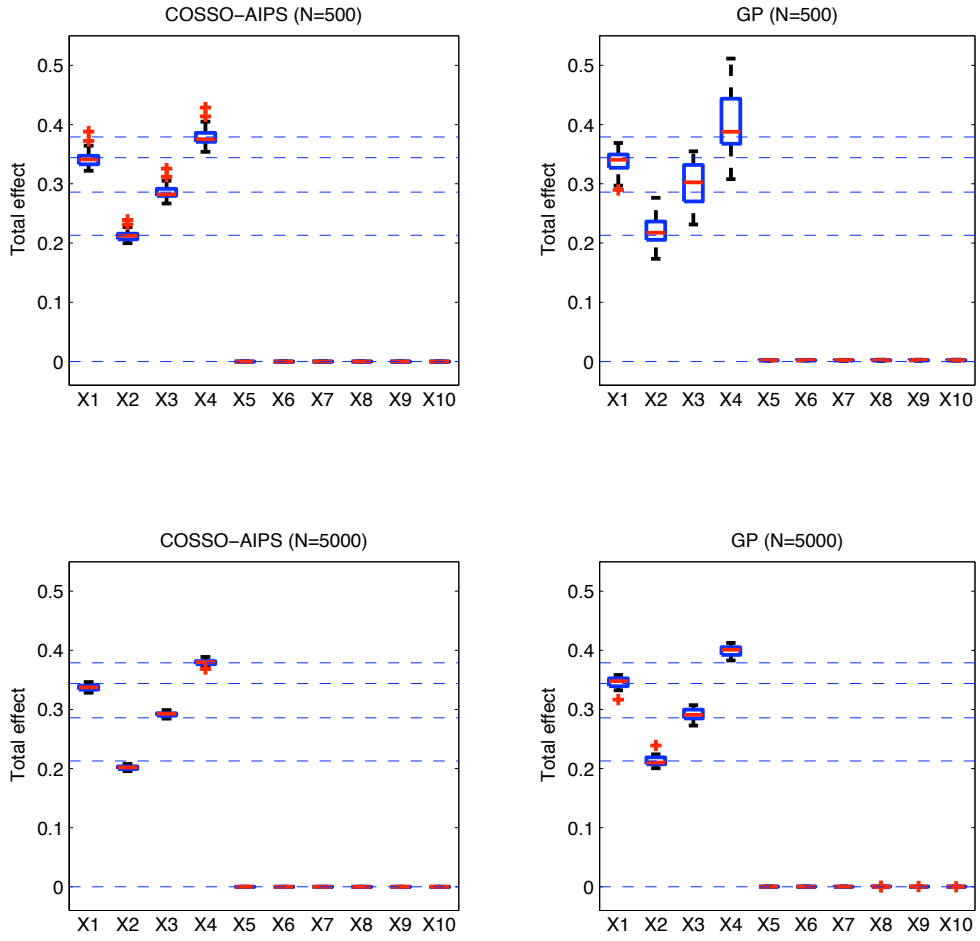

Figure 3.4: Total effect indices vs. sample effect (example 2)

3.6.3 Example 3

This third example is a high dimensional model with $d = 20$. This model is defined as

$$\begin{aligned} f(\mathbf{X}) = {} & g_1(X^{(1)}) + g_2(X^{(2)}) + g_3(X^{(3)}) + g_4(X^{(4)}) + 1.5g_2(X^{(8)}) + 1.5g_3(X^{(9)}) \\ & + 1.5g_4(X^{(10)}) + 2g_3(X^{(11)}) + 1.5g_4(X^{(12)}) + g_3(X^{(1)}X^{(2)}) + g_2(\frac{X^{(1)} + X^{(3)}}{2}) \\ & + g_1(X^{(3)}X^{(4)}) + 2g_3(X^{(5)}X^{(6)}) + 2g_2(\frac{X^{(5)} + X^{(7)}}{2}) \end{aligned}$$

where the functions g_1, g_2, g_3 and g_4 are the same as for example 2. Notice that $X^{(13)}, ..., X^{(20)}$ are uninformative. The reference values of the total effect indices are

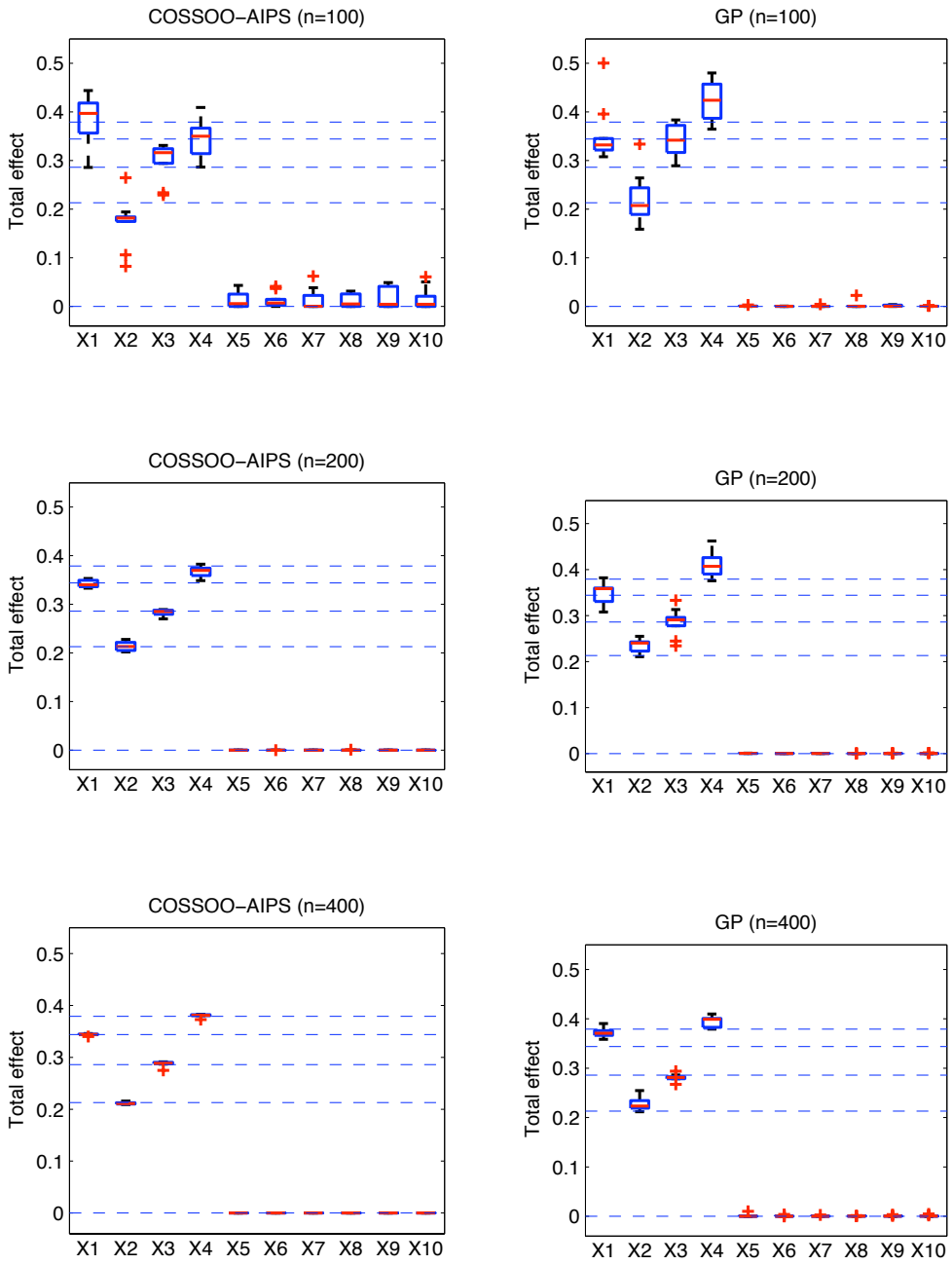

Figure 3.5: Total effect indices vs. experimental design size effect (example 2)

computed by direct Monte-Carlo simulation using Sobol's method (with $N = 250000$, which corresponds to 5.10^6 evaluations of the example 3 model). Table 3.7 shows 95% confidence intervals (95% CI) provided by 100 different samples and the chosen reference values.

Input	Total effect 95%CI	Reference value
$X^{(1)}$	$[0.050, 0.051]$	0.050
$X^{(2)}$	$[0.031, 0.032]$	0.031
$X^{(3)}$	$[0.042, 0.043]$	0.042
$X^{(4)}$	$[0.055, 0.057]$	0.056
$X^{(5)}$	$[0.139, 0.141]$	0.140
$X^{(6)}$	$[0.129, 0.132]$	0.130
$X^{(7)}$	$[0.033, 0.034]$	0.033
$X^{(8)}$	$[0.050, 0.051]$	0.050
$X^{(9)}$	$[0.116, 0.119]$	0.117
$X^{(10)}$	$[0.147, 0.149]$	0.148
$X^{(11)}$	$[0.207, 0.210]$	0.209
$X^{(12)}$	$[0.147, 0.149]$	0.148
$X^{(13,\ldots,20)}$	0	0

Table 3.5: 95% CI and the reference values of the total effect indices for example 3.

3.6.3.1 Assessment of the prediction accuracy

Table 3.6 summarizes the results for the 50 realizations of the example 3 model with two different experimental design sizes ($n = 200$ and $n = 400$) build using maximinLHD procedure. For this example we choose to do not test COSSO-IPS since we shown with the previous tests that AIPS have better computational performance. It can be seen that for this model GP has a bad performance for the both sizes of the experimental design. Concerning the COSSO methods we can see that as the size of the experimental design increases, both COSSO-AIPS and COSSO-solver provide increasingly accurate estimates. However, we can note that COSSO-NN-LARS does not increase its performance as others and as one would expect. As for previous examples, notice that COSSO-AIPS is the fastest method especially comparing to COSSO-solver and GP.

	n	$\bar{Q}_2$	time (s)
COSSO-NN-LARS	200	0.73(0.10)	120
	400	0.75(0.08)	281
COSSO-AIPS	200	0.78(0.08)	78
	400	0.94(0.04)	274
COSSO-solver	200	0.78(0.09)	355
	400	0.94(0.02)	720
GP	200	0.40(0.05)	240
	400	0.56(0.03)	1105

Table 3.6: Q_2 results from example 3. The estimated standard deviation of Q_2 is given in parentheses.

3.6.3.2 Global sensitivity analysis

In this section, total effect indices are computed using COSSO-AIPS. Here we do not compare the results to those using Sobol's method with GP response surface, because of its bad prediction performance (see Table 3.6). As previously, we will first study the effect of the sample size N on indices estimations. Thus, we build using maximinLHD procedure, 100 samples of two sizes $N = 500$ and $N = 5000$ and we compute the indices using our direct method based on predictive COSSO-AIPS response surface ($Q_2 = 0.98$). We can see in the figure 3.6 that those estimates are close to the reference values of the indices and that robustness of estimations increases by increasing N, nevertheless with $N = 500$ estimations are still quite good.

Table 3.7 summarize the results from using response surfaces build with the two different sizes of experimental design ($n = 200$ and $n = 400$). As one would expect the accuracy of the indices estimations improves as the experimental design increases (in other words as the predictivity improves). This study was done using the 50 response surfaces used in the previous section using a $N = 5000$ sample to compute the total effect indices.

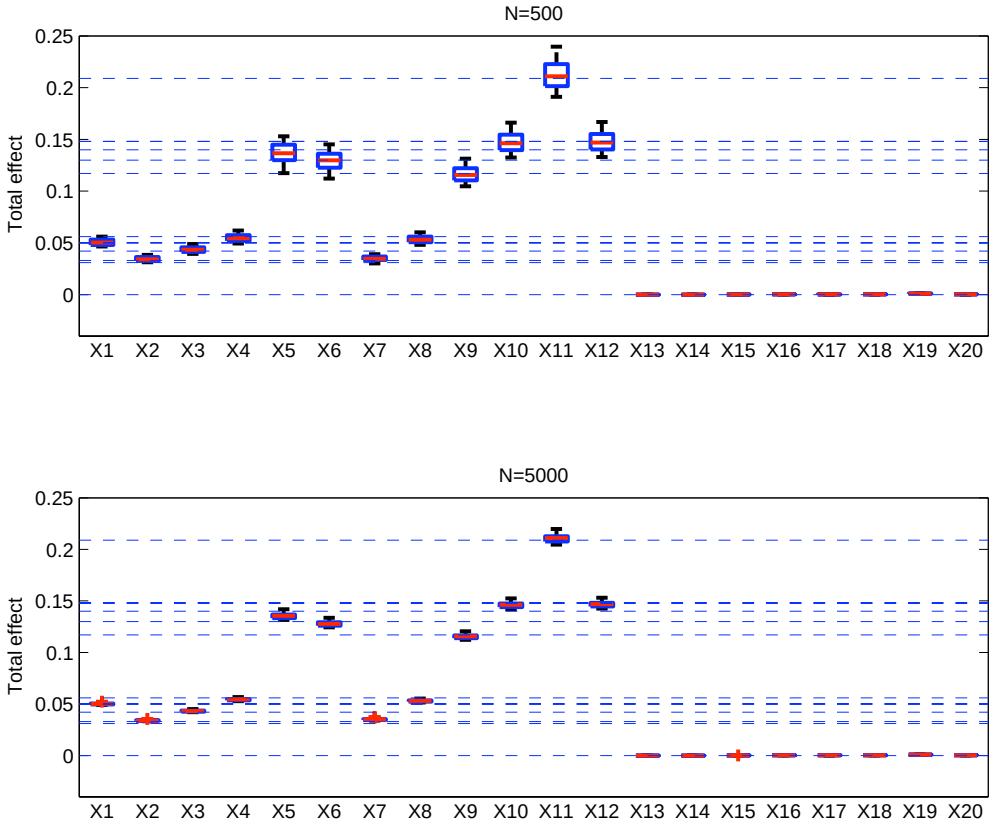

Figure 3.6: Total effect indices vs. sample effect (example 3)

Input	Reference value	$n = 200$	$n = 400$
$X^{(1)}$	0.050	0.035(0.017)	0.047(0.008)
$X^{(2)}$	0.031	0.022(0.019)	0.023(0.012)
$X^{(3)}$	0.042	0.021(0.019)	0.040(0.015)
$X^{(4)}$	0.056	0.029(0.022)	0.052(0.008)
$X^{(5)}$	0.140	0.127(0.033)	0.124(0.009)
$X^{(6)}$	0.130	0.107(0.051)	0.120(0.011)
$X^{(7)}$	0.033	0.034(0.028)	0.034(0.006)
$X^{(8)}$	0.050	0.041(0.034)	0.054(0.006)
$X^{(9)}$	0.117	0.146(0.018)	0.118(0.011)
$X^{(10)}$	0.148	0.172(0.025)	0.145(0.008)
$X^{(11)}$	0.209	0.248(0.05)	0.210(0.011)
$X^{(12)}$	0.148	0.158(0.026)	0.144(0.009)
$X^{(13)}$	0	0.003(0.004)	0.001(0.001)
$X^{(14)}$	0	0.002(0.004)	0.001(0.001)
$X^{(15)}$	0	0.004(0.006)	0.001(0.002)
$X^{(16)}$	0	0.001(0.003)	0.002(0.001)
$X^{(17)}$	0	0.003(0.003)	0.001(0.001)
$X^{(18)}$	0	0.001(0.002)	0.001(0.002)
$X^{(19)}$	0	0.003(0.004)	0.001(0.002)
$X^{(20)}$	0	0.004(0.009)	0.001(0.002)

Table 3.7: Total effect indices vs. experimental design size effect (example 3). The estimated standard deviation of the total effect index are given in parentheses.

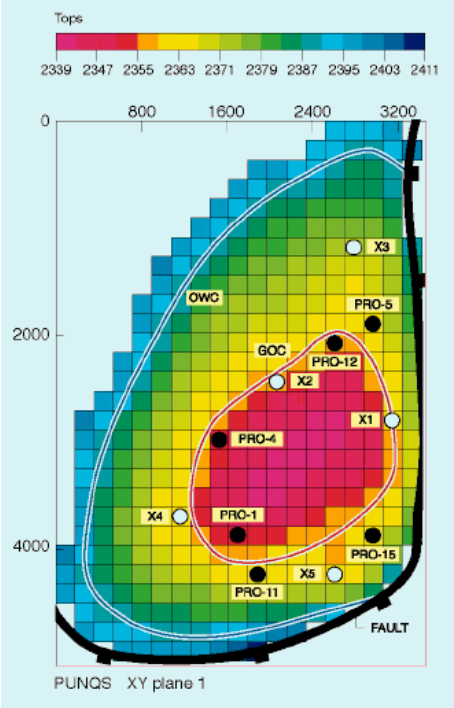

Figure 3.7: Top structure map of the reservoir field (PUNQS test case).

3.7 The reservoir test cases

3.7.1 PUNQS test case

3.7.1.1 Reservoir model description

The PUNQS case is a synthetic reservoir model taken from a real field located in the North Sea. The PUNQS test case, which is qualified as a small-size model, is frequently used as a benchmark reservoir engineering model for uncertainty analysis and for history-matching studies .

The geological model contains $19 \times 28 \times 5$ grid blocks, 1761 of which are active. The reservoir is surrounded by a strong aquifer in the North and the West, and is bounded to the East and South by a fault. A small gas cap is located in the centre of the dome shaped structure. The geological model consists of five independent layers, where the porosity distribution in each layer was modelled by geostatistical simulation. The layers 1, 3, 4 and 5 are assumed to be of good quality, while the layer 2 is of poorer quality. The field contains six production wells located around the gas-oil contact. Due to the strong aquifer, no injection wells are required. For more detailed description on the PUNQS

model, see Dejean & Blanc (1999). Twenty uncertain parameters uniformly distributed and independent, are considered in this study:

- $DensityGas\ U[0.8; 0.9]\ Kg/m^3$: gas density
- $DensityOil\ U[900; 950]\ Kg/m^3$: oil density
- $MPH\ U[0.5; 1.5]$: horizontal transmissibility multipliers for each layers (from 1 to 5)
- $MPV\ U[0.5; 5]$: vertical transmissibility multipliers for each layers (from 1 to 5)
- $PermAqui1\ U[100; 200]\ mD$: analytical permeability of the aquifer 1
- $PermAqui2\ U[100; 200]\ mD$: analytical permeability of the aquifer 2
- $PoroAqui1\ U[0.2; 0.3]$: analytical porosity of the aquifer 1
- $PoroAqui2\ U[0.2; 0.3]$: analytical porosity of the aquifer 2
- $SGCR\ U[0.02; 0.08]$: critical gas saturation
- $SOGCR\ U[0.2; 0.3]$: critical oil gas saturation; largest oil saturation at which oil is immobile in gas
- $SOWCR\ U[0.15; 0.2]$: critical oil water saturation; largest oil saturation at which oil is immobile in water
- $SWCR\ U[0.2; 0.3]$: critical water saturation

For this study we focus on an objective function output, defined as:

$$OF(X) = \frac{(f(X) - \mathbf{d})^T C_D^{-1} (f(X) - \mathbf{d})}{2} \tag{3.36}$$

where C_D is the covariance matrix of the observed data and d the observed data. This OF is given by equation (3.36) and represents the mismatch between observed and simulated data. The observed data is synthetically generated using a random value for the uncertain parameters in the simulator and adding noise (10% of the average value of each time series) to the results. This data consists in time series given with two months frequency during the first 6 years for the following simulator outputs: Gas Oil Ratio, Bottom Hole Pressure, Oil Production Rate, and Water Cut. To define the weights in the objective function definition, we consider independent measurement errors for each time dependent output. This error was taken to be equal to 10% of the average value of each time series.

3.7.1.2 Assessment of the prediction accuracy

Each of the input range has been rescaled to the interval $[0, 1]$ and the reservoir simulator is run on two experimental designs of size $n = 200$ and $n = 400$, which were built using maximinLHD. Then we construct response surfaces using COSSO-AIPS, C0SSO-solver, COSSO-NN-LARS and GP. In order to estimate Q_2 the simulator was run at an additional sample set of size $n_{test} = 500$. Table 3.8 shows the results of this study. We see that for this test case GP outperforms COSSO's methods, but differences between Q_2 given by the used methods are small when the design size is $n = 400$. In addition, as previously shown COSSO-AIPS is less time consuming than others especially if we compare it with GP. Consequently COSSO and particularly COSSO-AIPS is well adapted to perform GSA.

	n	Q_2	time (s)
COSSO-NN-LARS	200	0.67	200
	400	0.81	450
COSSO-AIPS	200	0.69	70
	400	0.82	300
COSSO-solver	200	0.67	280
	400	0.81	700
GP	200	0.75	402
	400	0.84	794

Table 3.8: PUNQS model Q_2 results

3.7.1.3 Global sensitivity analysis

Here we use COSSO-AIPS and GP to produce response surfaces which are built using the experimental design of size $n = 400$. To compute the total effect and main effect indices via COSSO-AIPS we use a sample of size $N = 5000$ and two samples of the same size for the case using GP. We provided here the main effect indices to show the reader the importance of the interaction effects in this model. Tables 3.9 shows the computed indices, thus we can see that the main effect and the interactions of $MPH5$ explain more than 65% of the model variance, then we have a group of five inputs ($SWCR$, $MPH1$, $SOGCR$, $SGCR$ and $PermAqui1$) with relatively important effects and a group of five or six (depending on the method used) inputs with poor importance ($0.05 > \widehat{S}_{T_j} > 0.01$).

While the remaining are considered as uninformative. The GSA results using COSSO-AIPS and GP are almost equivalent, which was expected knowing that their Q_2 are close.

GP		
Input	Total effect	Main effect
$MPH5$	0.656	0.396
$SWCR$	0.193	0.013
$MPH1$	0.143	0.035
$SOGCR$	0.122	0.003
$SGCR$	0.112	0.021
$PermAqui1$	0.060	−0.011
$MPH3$	0.049	0.001
$DensityOil$	0.040	−0.007
$PermAqui2$	0.035	−0.010
$SOWCR$	0.024	−0.019
$MPV4$	0.023	−0.016
$MPV1$	0.005	−0.019
$MPV2$	0.005	−0.019
$MPV5$	0.004	−0.018
$MPV3$	0.002	−0.018
$PoroAqui1$	0.003	−0.017
$DensityGas$	0.001	−0.019
$MPH4$	0.001	−0.018
$MPH2$	0.001	−0.019
$PoroAqui2$	0	−0.019

COSSO-AIPS		
Input	Total effect	Main effect
$MPH5$	0.664	0.402
$SWCR$	0.203	0.034
$MPH1$	0.160	0.058
$SGCR$	0.104	0.041
$SOGCR$	0.091	0.019
$PermAqui1$	0.062	0.003
$MPH3$	0.034	0.007
$PermAqui2$	0.021	0.002
$MPV1$	0.021	0
$DensityOil$	0.019	0.008
$MPV4$	0.018	0.004
$SOWCR$	0.011	0.003
$PoroAqui2$	0.005	0.001
$PoroAqui1$	0.004	0.002
$MPV3$	0.003	0
$MPV5$	0.002	0
$MPV2$	0.002	0
$DensityGas$	0.001	0
$MPH2$	0	0
$MPH4$	0	0

Table 3.9: GSA from PUNQS model

3.7.2 IC Fault Model

3.7.2.1 Reservoir model description

The geological model consists of six layers of alternating good and poor quality sands (see Figure 3.8). The three good quality layers have identical properties, and three poor quality layers have different set of identical properties. The thickness of the layers has arithmetic progression, with the top layer having a thickness of 12.5 feet, the bottom layer

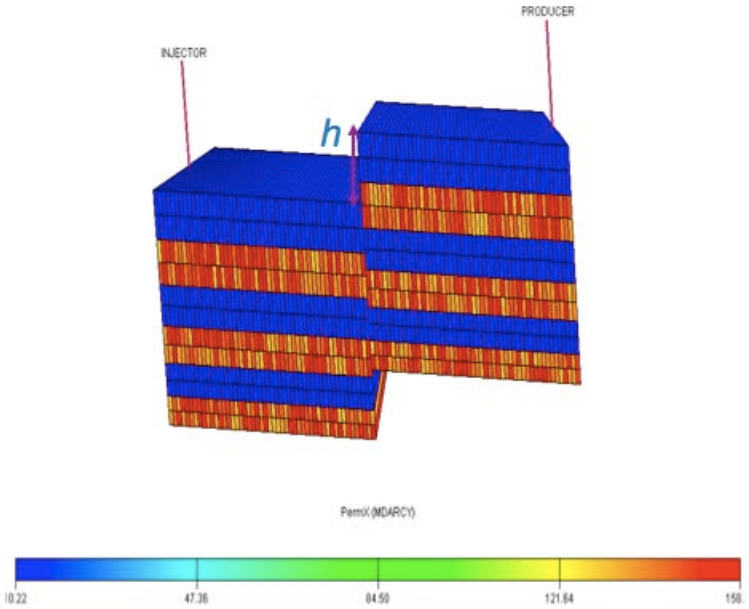

Figure 3.8: IC Fault Model

a thickness of 7.5 feet, and a total thickness of 60 feet. The width of the model is 1000 feet, with a simple fault at the mid-point, which off-sets the layers. There is a water injector well at the left-hand edge, and a producer well on the right-hand edge. Both wells are completed on all layers, and operated at a fixed bottom hole pressures.

The simulation model is 100×12 grid blocks, with each geological layer divided into two simulation layers with equal thicknesses, each grid block is 10 feet wide. The model is constructed such that the vertical positions of the wells are kept constant and equal, even when different fault throws are considered. the well depth is 8325 feet to 8385.

The porosity and permeabilities in each grid block were randomly drawn from uniform distributions with no correlations. The range for the porosities was ± 10 of the mean value, while range for the permeabilities was ± 1 of the mean value. The means for the porosities were 0.30 for the good quality sand and 0.15 for the bad quality sand. The means of the permeabilities were 158.6 mD for the good quality sand and 2.0 mD for the poor quality sand.

This simplified reservoir model has three uncertain input parameters, corresponding to the fault throw h, the good and the poor sand permeability multipliers k_g and k_p. The three parameters are selected independently from uniform distributions with ranges : $h \in [0, 60]$ $k_g \in [100, 200]$ and $k_p \in [0, 50]$. The analysed output is in this test case the oil production

rate Q_{op} at 10 years. Figure 3.9 illustrates this output against k_g and k_p at a fixed high value of h. For more detailed description on the IC Fault model, see Tavassoli *et al.* (2004).

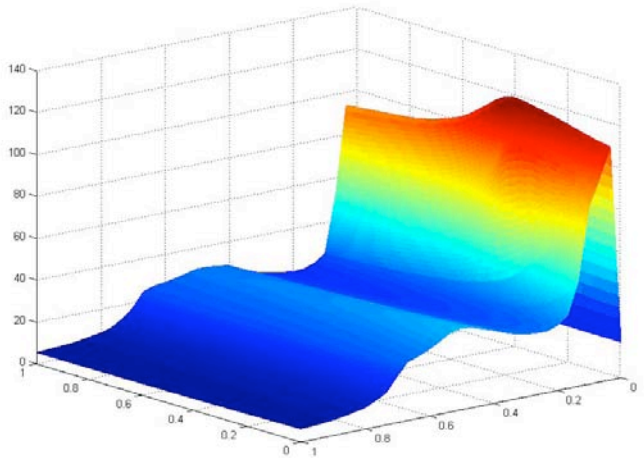

Figure 3.9: Oil production rate after 10 years vs. k_g and k_p at a fixed high value of h (obtained with 1000 simulations)

3.7.2.2 Assessment of the prediction accuracy

The simulator is run on four experimental designs of size $n = 100$, $n = 200$, $n = 400$ and $n = 1600$ generated by maximinLHD procedure. Then we construct response surfaces using COSSO-AIPS and GP. In order to estimate Q_2, the simulator was run at an additional sample set of size $n_{test} = 25000$. Table 3.10 shows the results of this study. Clearly, COSSO-AIPS outperforms GP in this test case, however an experimental design of size $n = 400$ is necessary to provide a reasonably accurate estimate. Moreover, we can note that as the experimental design increases the accuracy of COSSO-AIPS estimate increases, this is not the case for GP as remarked in Example 1. Indeed, by increasing the design from 200 to 400 instead of improving, the estimate becomes worse in terms of predictivity. Even if there are only three uncertain inputs in this test case, the approximation of the input/output relation is a complicated problem, this is due to the presence of the fault that provide discontinuities in the model.

	n	Q_2	time (s)
COSSO-AIPS	100	0.34	1
	200	0.63	4
	400	0.72	15
	1600	0.81	280
GP	100	0.33	8
	200	0.57	25
	400	0.52	63
	1600	0.66	1128

Table 3.10: IC fault model Q_2 results

3.7.2.3 Global sensitivity analysis

As for PUNQS test case we use COSSO-AIPS and GP to produce response surfaces which are built using the experimental design of size $n = 1600$. To compute the total effect and main effect indices via COSSO-AIPS we use a sample of size $N = 5000$ and two samples of the same size for the case using GP. Tables 3.11 shows the computed indices. Following the GSA results produced via COSSO-AIPS, we can see that the variance of the oil production rate mainly depend on the fault throw h and the poor sand permeability k_p. With respect to GSA, results produced via GP gives more interaction effect to the good sand permeability k_g than COSSO-AIPS. The better Q_2 of COSSO-AIPS suggests that its GSA results are more robust.

GP		
Input	Total effect	Main effect
h	0.381	0.100
k_g	0.173	0.021
k_p	0.809	0.596

COSSO-AIPS		
Input	Total effect	Main effect
h	0.375	0.225
k_g	0.030	0.011
k_p	0.733	0.586

Table 3.11: GSA from IC fault model

3.8 Conclusion

In this chapter, we presented the COSSO regularized nonparametric regression method, which is a model fitting and variable selection procedure. One of the COSSO's algorithm

steps is the NNG optimization problem. The original COSSO algorithm uses classical constrained optimization techniques to solve the NNG problem, these techniques are efficient but time consuming, especially with high dimensional problems (as empirically shown) and with big size of experimental design (high number of observations). A new iterative algorithm was developed, so-called IPS with its accelerated version (AIPS). Based on the Landweber iterative algorithms these procedures are conceptually simple and easy to implement.

We also applied the NN-LARS algorithm to COSSO which, as expected, has competitive computation time performance comparing to the original COSSO (COSSO-solver). We empirically show that COSSO based on the AIPS algorithm is the fastest COSSO version. Moreover, we used the ANOVA decomposition basis of the COSSO to introduce a direct method to compute the Sobol' indices. We applied COSSO to the problem of GSA for several analytical models and reservoir synthetic test cases, and we compared its performance to GP method combined with Sobol' Monte-Carlo method. For all the test cases COSSO shows very good performances, especially the COSSO-AIPS version, for which the computational gain was significant compared to COSSO-solver and GP. Consequently, COSSO-AIPS constitutes an efficient and practical approach to GSA. We have also noticed that the COSSO did not provide good results in the example 1, which is a function with discontinuities on its derivatives. To address this type of functions we decoded to investigate the use wavelets basis instead of smoothing splines. Indeed, it is well known that wavelets are well suited to fit functions with discontinuities. In the next chapter a new multivariate nonparametric regression method is presented, which can be seen as a wavelet view of the COSSO.

Chapter 4

Wavelet kernel ANOVA

4.1 Introduction

In recent years there has been an important development in the application of wavelet methods in statistics, especially in signal processing, in image and function representation methods, with many successes in the efficient analysis and compression of noisy data. As a result, wavelets have became another standard tool for the statistician. Broadly speaking, wavelets are functions constructed to satisfy certain mathematical properties, and wavelet algorithms process data at different resolutions (multiresolution analysis). In other words, to notice the gross features of the signal we look at it within a large window and to notice the small features we look at a signal within a small window (zoom-in and zoom-ou property).

The multiresolution analysis provides a good time-frequency localization, which makes wavelet methods efficient to estimate functions with sharp spikes, and discontinuities. Thus wavelets are used in various nonparametric regression methods. However, most of these methods are implemented only for one (signal) or two (image) dimensional problems, the reason for this is that these algorithms are constructed with the assumptions for the data to be of dyadic size and with equally spaced points.

Several algorithms have been proposed to overcome the setting of non-dyadic and non-equispaced design. Among them, Antoniadis *et al.* (1997) transform the random design into equispaced data via binning method. Kovac & Silverman (2000) apply the linear transformation to the data to map it to a dyadic and equispaced set of points. Kerky-acharian & Picard (2004) project the data on an unusual non-orthonormal basis, called warped wavelet basis. Amato *et al.* (2006) suggested a regularization method relying on

wavelet kernel reproducing Hilbert spaces, which does not require a pre-processing of data. The method also achieves optimal convergence rates in the Besov spaces when the estimation error is calculated at the design points only no matter how irregular the design is. Given that, it seems that this method is well adapted to be generalized for the multivariate regression using wavelets. See Appendix C for more details on wavelets.
Inspired by the COSSO (COmponent Selection and Smoothing Operator) (Lin & Zhang, 2006) and the wavelet kernel penalized estimation for non-equispaced design regression proposed by Amato *et al.* (2006), we introduce in this chapter a new approach in estimation of ANOVA components. Given a wavelet type expansion of f we consider a class of wavelet estimators for the nonparametric regression problem using a penalized least-squares approach with penalties. The penalties are chosen in order to control the smoothness of the resulting estimator. For this we use the same penalty as the one used for COSSO, in other words the semi-norm penalty. So we take for penalty, a weighted sum of wavelet details norms.
In this chapter, we first broadly review some definitions which are given in Amato *et al.* (2006). Then we present our nonparametric regression method, named WK-ANOVA, as well as its algorithm. Finally, numerical simulations are presented and discussed.

4.2 Wavelet kernel nonparametric regression for non equispaced design

Consider the univariate regression problem:

$$y_i = f(x_i) + \epsilon_i,\ i = 1, \cdots, n \tag{4.1}$$

where $(x_i)_{i=1,\cdots,n}$ is the irregular design, the ϵ_i are i.i.d. and $N(0, \sigma^2)$ random errors and f an unknown regression function to be estimated.

4.2.1 Wavelet kernels

Let $G_{-1} = \{-1\} \times \{0\}$, $G_0 = \{0\} \times \{0, 1\}$ and for each integer $J \geq 1$ let $G_J = \{J\} \times \{k \in \{0, ..., 2^J\}; k/2 \notin \mathbb{Z}\}$, i.e. G_J is the index set of wavelets at resolution level J. The whole set of indexes pairs (j, k) that describes all wavelets will be denoted by $G = \bigcup_{j \geq -1} G_j$. Therefore, any function $f \in L_2([0, 1))$ admits the infinite wavelet expansion:

$$f = \sum_{g \in G} f_g \psi_g$$

where ψ_g is the wavelet basis function indexed by $g \in G$, f_g is the correponding expansion coefficient and $\psi_{-1,0} = \phi_{0,0}$.
We now define a class of wavelet-based Hilbert spaces. For any function:

$$\Gamma : G \to [0, \infty)$$

define the Hilbert space:

$$\mathcal{H}_\Gamma = \{f \in L_2([0,1]) : \sum_{g \in G} \Gamma(g)|f_g|^2 < \infty\}$$

with scalar product:

$$\langle f, h \rangle_\Gamma = \sum_{g \in G} f_g h_g \Gamma(g)$$

and let be $\| \cdot \|_\Gamma$ the associated norm. As G_J is a finite subset of G, we have $V_J \subset \mathcal{H}_\Gamma$ for every $J \geq 0$. Moreover, for any $f \in \mathcal{H}_\Gamma$,

$$\lim_{J \to \infty} \| f - P_J(f) \|_\Gamma = 0 \tag{4.2}$$

where $P_J(f)$ is the projection of a function f into the space V_J. The space $\mathcal{H}_\Gamma$ is a RKHS and the corresponding reproducing kernels are given by

$$K^\Gamma(x, y) = \sum_{g \in G} \frac{\psi_g(x)}{\Gamma(g)} \psi_g(y), \ x, y \in [0, 1)$$

where $\psi_g(x)$ is a wavelet function (see appendix C for more details). By definition of the index set G, the kernel K can also be written as a sum of the reproducing kernels:

$$K_j^\Gamma(x, y) = \sum_{k=0}^{2^j - 1} \frac{\psi_{j,k}(x)}{\Gamma(j,k)} \psi_{j,k}(y)$$

This implies that the RKHS $\mathcal{H}_\Gamma$, can be decomposed into a direct sum of wavelet RKHS's (spanned by a set of wavelets of scale j) as

$$\mathcal{H}_\Gamma = V_0 \oplus \bigoplus_{j \geq 0} \mathcal{W}_j^\Gamma \tag{4.3}$$

where each space $\mathcal{W}_j^\Gamma$ is the RKHS associated to the kernel K_j^Γ. This representation involves an infinite decomposition of the detail space, in practice we truncate (4.3) up to a maximum resolution J, in other words, the RKHS $\mathcal{H}_{J,\Gamma} = V_0 \oplus \bigoplus_{j=0}^J \mathcal{W}_j^\Gamma$ defines a multiresolution analysis of $\mathcal{H}_\Gamma$ and the associated kernel is

$$K_J^\Gamma(x, y) = \sum_{g \in \cup_{0 \leq j \leq J} G_j} \frac{\psi_g(x)}{\Gamma(g)} \psi_g(y), \ x, y \in [0, 1)$$

Furthermore, from (4.2)

$$\lim_{J\to\infty} \| K^{\Gamma} - K_J^{\Gamma} \|_{\infty} = 0$$

We assume that Γ is only a function of j and equals to 2^{2js} on G_j and $s > 1/2$, then $\mathcal{H}_\Gamma$ equals to the Sobolev space $B^s_{2,2}([0,1])$ of index s. For more mathematical details we refer to Amato *et al.* (2006).

4.2.2 Wavelet kernel penalized estimation

As discussed previously Amato *et al.* (2006) define a least square procedure for estimating the unknown regression function $f \in \mathcal{H}_{J,\Gamma}$ by minimizing:

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda^2 \sum_{j=0}^{J} \|P_j^{\Gamma} f\|_{\mathcal{H}_{j,\Gamma}}$$

where we denote by $P_j^{\Gamma} f$ the orthogonal projection of f onto $\mathcal{W}_j^{\Gamma}$. The penalty term is a sum of the wavelet-based RKHS norm.
Amato *et al.* (2006) have shown that penalizing the norm by blocks produces a better regularization. Thus, they propose finding $f \in \mathcal{H}_{J,\Gamma}$ to minimize:

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda^2 \sum_{j=0}^{J}\sum_{m} \|P_{j,m}^{\Gamma} f\|_{\mathcal{H}_{j,m,\Gamma}} \tag{4.4}$$

where $\mathcal{H}_{j,m,\Gamma}$ is the RKHS corresponding to the kernel defined by

$$K_{j,m}^{\Gamma}(x,y) = \sum_{k\in T_{j,m}} \frac{\psi_{j,k}(x)}{\Gamma_j}\psi_{j,k}(y)$$

where $T_{j,m}$ correspond to the partition on blocks at resolution j of lenght M_j.
The existence of the estimate obtained by the penalization procedure (4.4) is guaranteed by the following theorem (Amato *et al.*, 2006).

Theorem 4.2.1 *Let $\mathcal{H}_{J,\Gamma}$ be the wavelet-based RKHS of functions over $[0,1]$ and consider its decomposition:*

$$\mathcal{H}_{J,\Gamma} = V_0 \oplus \bigoplus_{j=0}^{J} \mathcal{W}_j^{\Gamma}$$

Then there exists a minimizer of (4.4) in $\mathcal{H}_{J,\Gamma}$.

Note that this method does not require the knowledge of the distribution of the design points.

4.3 The wavelet kernel ANOVA

Consider now a multivariate regression problem:

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i,\ i = 1, \cdots, n$$

where $\boldsymbol{x}_i = (x_{i_1}, \ldots, x_{i_d})^T$ is d dimensional vector of inputs, the ϵ_i are i.i.d. and $N(0, \sigma^2)$ random errors and f an unknown univariate regression function to be estimated.
We choose to study here a more general problem, instead of the deterministic computer code, to show the good denoising performance of the method.

4.3.1 Definition

The idea behind the well known smoothing spline ANOVA model is to construct a RKHS $\mathcal{F} = \{f \in L^2([0,1]^d)\}$ corresponding to the decomposition (3.2). Then the model space $\mathcal{F}$ is the tensor product space of $\mathcal{H}_\Gamma^l$:

$$\mathcal{F} = \bigotimes_{l=1}^{d} \mathcal{H}_\Gamma^l = \{1\} \oplus \sum_{l=1}^{d} \bar{\mathcal{H}}_\Gamma^l \oplus \sum_{l<m} [\bar{\mathcal{H}}_\Gamma^l \otimes \bar{\mathcal{H}}_\Gamma^m] \ldots \tag{4.5}$$

where $\mathcal{H}_\Gamma^l = \{1\} \oplus \bar{\mathcal{H}}_\Gamma^l$ and $\bar{\mathcal{H}}_\Gamma^l$ are RKHS associated to the first-order component functions f_l of ANOVA expansion. The tensor products $[\bar{\mathcal{H}}_\Gamma^l \otimes \bar{\mathcal{H}}_\Gamma^m]$ are associated to the second-order component function f_{lm}. We denote by $\mathcal{W}_j^l$ the RKHS associated to wavelet kernel K_j (a detail space at scale j) and the variate $X^{(l)}$, thereby the function space $\mathcal{H}_\Gamma^l$ can be written as

$$\mathcal{H}_\Gamma^l = V_0 \oplus \bigoplus_{j=0}^{J} \Gamma_j^{-1} \mathcal{W}_j^l \tag{4.6}$$

and the tensor product $[\mathcal{H}_\Gamma^l \otimes \mathcal{H}_\Gamma^m]$ as

$$\mathcal{H}_\Gamma^l \otimes \mathcal{H}_\Gamma^m = (V_0^l \otimes V_0^m) \bigoplus_{j=0}^{J} \Gamma_j^{-2} (\mathcal{W}_j^l \otimes \mathcal{W}_j^m) \tag{4.7}$$

It is easy to see that V_0 is also the subspace of $L_2([0,1])$ spanned by the constant function on $[0,1]$, one has $V_0 = V_0^l \otimes V_0^m = \{1\}$.
Thus, the function space $\mathcal{F}$, which is a wavelet-based RKHS, can be also written as

$$\mathcal{F} = \{1\} \oplus \bigoplus_{\gamma=1}^{q} \mathcal{F}_\gamma \tag{4.8}$$

where $\mathcal{F}_\gamma$'s are orthogonal subspaces of $\mathcal{F}$ and correspond to the subspaces $\bar{\mathcal{H}}_\Gamma^l$, $[\bar{\mathcal{H}}_\Gamma^l \otimes \bar{\mathcal{H}}_\Gamma^m]$, etc ... In the additive model $q = d$ where d is the number of input parameters and in the model with two way interaction $q = d(d+1)/2$. We assume that a second order ANOVA expansion gives a satisfactory approximation of f.

We denote by $P_\gamma^\Gamma f$ the orthogonal projection of f onto $\Gamma_j^{-1}\mathcal{W}_j$ and $\|\cdot\|$ the norm in the RKHS $\Gamma_j^{-1}\mathcal{W}_j$. Under the framework of smoothing spline ANOVA one way to estimate f is to find $f \in \mathcal{F}$ that minimizes:

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(\boldsymbol{x}_i))^2 + \lambda^2 \sum_{\gamma=1}^{q}\sum_{j=0}^{J}\sum_{k=0}^{2^j-1}\theta_{\gamma,j,k}^{-1} \parallel P_{\gamma,j,k}^\Gamma f \parallel^2 \tag{4.9}$$

where $\theta_{\gamma,j,k} \geq 0$. If $\theta_{\gamma,j,k} = 0$, then the minimizer is taken to satisfy $\parallel P_{\gamma,j,k}^\Gamma f \parallel^2 = 0$, using the convention $0/0 = 0$. The parameter λ controls the trade-off between the first term in the above expression which discourages the lack of fit of f and the second one which penalizes the roughness of f.

In analogy with COSSO (Lin & Zhang, 2006) and wavelet kernel penalized estimation (Amato *et al.*, 2006) we propose the WK-ANOVA procedure, another way to estimate f, given by $f \in \mathcal{F}$ that minimize:

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(\boldsymbol{x}_i))^2 + \lambda^2 R_q(f) \tag{4.10}$$

with $R_q(f) = \sum_{\gamma=1}^{q}\sum_{j=0}^{J}\sum_{k=0}^{2^j-1} \parallel P_{\gamma,j,k}^\Gamma f \parallel$ is a sum of wavelet-based RKHS norms, instead of the squared RKHS norm employed in (4.9). We note that $R_q(f)$ is not a norm in $\mathcal{F}$ but a pseudo-norm in the following sense: $R_q(f) \geq 0$, $R_q(cf) = \mid c \mid R_q(f)$, $R_q(f+h) \leq R_q(f) + R_q(h)\ \forall\ f, h \in \mathcal{F}$, and, $R_q(f) > 0$ for any non constant $f \in \mathcal{F}$. Futhermore

$$\sum_{\gamma=1}^{q}\sum_{j=0}^{J}\sum_{k=0}^{2^j-1} \parallel P_{\gamma,j,k}^\Gamma f \parallel^2 \leq R_q(f)^2 \leq q\sum_{\gamma=1}^{q}\sum_{j=0}^{J}\sum_{k=0}^{2^j-1} \parallel P_{\gamma,j,k}^\Gamma f \parallel^2 \tag{4.11}$$

Note that there is only one smoothing parameter λ which should be properly chosen, instead of multiple smoothing parameters θ's in (4.9).

The existence of the WK-ANOVA estimate, which is due to the convexity of (4.10), is guaranteed by adapting Theorem 1 of Lin & Zhang (2006).

Theorem 4.3.1 *Let $\mathcal{F}$ be the wavelet-based RKHS of functions over $[0,1]^d$. Assume that $\mathcal{F}$ can be decomposed as (4.8). There exists a minimizer of (4.10).*

Define $\|\cdot\|_n$ as the Euclidian norm in $\mathbb{R}^n$. Under our previous assumption, $s > 1/2$ and $\Gamma_j = 2^{2sj}$. The following theorem is equivalent to theorem 2 of Lin & Zhang (2006) and shows that the WK-ANOVA estimator in the additive model has a rate of convergence $n^{-s/(2s+1)}$, where s is the order of smoothness of the components.

Theorem 4.3.2 *Consider the regression model $y_i = f_0(\boldsymbol{x}_i) + \varepsilon_i$, $i = 1, ..., n$, where $\boldsymbol{x}_i$'s are given deterministic points in $[0,1]^d$, and the ε_i's are independent $N(0, \sigma^2)$ noise variables. Assume f_0 lies in $\mathcal{F} = \{1\} \oplus \bigoplus_{l=1}^{d} \mathcal{H}_\Gamma^l$, with $\mathcal{H}_\Gamma^\gamma = \{1\} \oplus \bar{\mathcal{H}}_\Gamma^\gamma$ being the Sobolev space $B_{2,2}^s([0,1])$ of index s. Consider the WK-ANOVA estimator $\widehat{f}$ at the design points as defined by (4.10). Then (i) if f_0 is not a constant, and $\lambda_n^{-1} = O_p(n^{s/(2s-1)}) R_q^{(2s-1)/(4s+2)}(f_0)$, we have $\| \widehat{f} - f_0 \|_n = O_p(\lambda_n) R_q^{1/2}(f_0)$; (ii) if f_0 is a constant, we have $\| \widehat{f} - f_0 \|_n = O_p(max\{n^{-s/(2s-1)} \lambda_n^{-2/(2s-1)}, n^{-1/2}\})$.*

The following Lemma shows that the solution of (4.10) is in finite dimensional space and the WK-ANOVA estimate can be computed directly from (4.10) by linear programming techniques.

Lemma 4.3.3 *Let $\widehat{f} = \widehat{b} + \sum_{\gamma=1}^{q} \widehat{f}_\gamma$ be a minimizer of (4.10), with $f_\gamma \in \mathcal{F}_\gamma$. Then $\widehat{f}_\gamma \in span\{K_\gamma(\mathbf{x}_i, \cdot), i = 1, ..., n\}$, where $K_\gamma = \sum_{j \geq 0} K^\Gamma$ is the reproducing kernel of the space $\mathcal{F}_\gamma$*

Using the suggestion of Antoniadis & Fan (2001) for solving penalized problems with l_1 penality, we can give an equivalent formulation of (4.10) for computational consideration. Consider the problem of finding $\theta = \{\theta_{\gamma,j,k}, \gamma = 1, ..., q; j = 0, ..., J; k = 1, ..., 2^j - 1\}$ and $f \in \mathcal{F}$ to minimize:

$$\frac{1}{n} \sum_{i=1}^{n} (y_i - f(\boldsymbol{x}_i))^2 + \lambda_0 \sum_{\gamma=1}^{q} \sum_{j=0}^{J} \sum_{k=0}^{2^j-1} \theta_{\gamma,j,k}^{-1} \| P_{\gamma,j,k}^\Gamma f \|^2 + \nu \sum_{\gamma=1}^{q} \sum_{j=0}^{J} \sum_{k=0}^{2^j-1} \theta_{\gamma,j,k} \tag{4.12}$$

subject to $\theta_{\gamma,j,k} \geq 0$,$\gamma = 1, ..., q$, $j = 0, ..., J$, $k = 1, ..., 2^j - 1$, where λ_0 is a fixed positive constant and ν is a smoothing parameter. We fix λ_0 at some value. Then

Lemma 4.3.4 *Set $\nu = \lambda^4/(4\lambda_0)$. (i) if $\widehat{f}$ minimizes (4.10), set $\widehat{\theta}_{\gamma,j,k} = \lambda_0^{1/2} \nu^{-1/2} \| P_{\gamma,j,k}^\Gamma f \|$, then the pair $(\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{f}})$ minimizes (4.12). (ii) On the other hand, if a pair $(\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{f}})$ minimizes (4.12), then $\widehat{\mathbf{f}}$ minimizes (4.10).*

As already introduced by Amato *et al.* (2006) we can penalize the norm of coefficients by blocks, which allows reducing the number of θ's that need to be estimated and can provide a better regularization. Hence, as defined before

$$K_{jm}^\Gamma(x, y) = \sum_m \frac{\psi_{j,k}(x)}{\Gamma_j} \psi_{j,k}(y)$$

where $m = 1, ..., M_j$ and M_j denotes the number of blocks at scale j. In the same way consider the decomposition $\mathcal{H}^l_\Gamma = V_0 \oplus \bigoplus_{j=0}^{J} \sum_m \Gamma_j^{-1} \mathcal{W}^l_{j,m}$ replace (4.12) by

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(\boldsymbol{x}_i))^2 + \lambda_0 \sum_{\gamma=1}^{q}\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{\gamma,j,m}^{-1} \parallel P^{\Gamma}_{\gamma,j,m} f \parallel^2 + \nu \sum_{\gamma=1}^{q}\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{\gamma,j,m} \tag{4.13}$$

We can note that the form of (4.13) is similar to the smoothing spline ANOVA (4.9) with multiple smoothing parameters and an additional penalty on the θ's. There is only one smoothing parameter ν in (4.13) and θ's are part of the estimate, rather than three smoothing parameters. For the WK-ANOVA procedure the sparsity on the detail components is controlled by the additional penalty on θ's in (4.13) makes possible to have some θ's to be zero, thus producing a sparse kernel estimate in sense of Gunn & Kandola (2002).

4.3.2 Algorithm

We will use an iterative optimization algorithm which is equivalent to the one used in Lin & Zhang (2006) and Amato *et al.* (2006). On each step of iteration, for any fixed θ we minimize (4.13) with respect of f, and then for this choice of f we minimize (4.13) with respect of θ. Note that for any fixed θ (4.13) is equivalent to the smoothing spline ANOVA procedure. Therefore from Wahba (1990) the solution f of (4.13) has the following form

$$f(\mathbf{x}) = b + \sum_{i=1}^{n} c_i \sum_{\gamma=0}^{q}\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{\gamma,j,m} K^{\Gamma}_{\gamma,j,m}(\mathbf{x}_i, \mathbf{x}) \tag{4.14}$$

Where $c = (c_1, ..., c_n)^T$, $b \in \mathbb{R}$, $K^{\Gamma}_{\gamma,j,m}$ is the reproducing kernel of $\Gamma_j^{-1}\mathcal{W}^l_{\gamma,j,m}$ if $\gamma \leq d$ and is the reproducing kernel of $\Gamma_j^{-2}\mathcal{W}^l_{\gamma,j,m} \otimes \mathcal{W}^p_{\gamma,j,m}$ else. In what follows, we denote by $K^{\Gamma}_{\gamma,j,m}$ the $n \times n$ matrix $\{K^{\Gamma}_{\gamma,j,m}(\mathbf{x}_i, \mathbf{x}_t)\}$, $i = 1, ..., n$, $t = 1, ..., n$, by K^{Γ}_{θ} the matrix $\sum_{\gamma=0}^{q}\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{\gamma,j,m} K^{\Gamma}_{\gamma,j,m}(\mathbf{x}_i, \mathbf{x})$ and 1_n be the column vector consisting for n ones. Then we can write $\mathbf{f} = K^{\Gamma}_{\theta}\mathbf{c} + b\mathbf{1}_n$, it follows that (4.13) can be expressed as

$$\frac{1}{n} \parallel \mathbf{Y} - \sum_{\gamma=0}^{q}\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{\gamma,j,m} K^{\Gamma}_{\gamma,j,m}\mathbf{c} - b\mathbf{1}_n \parallel_n^2 + \lambda_0 \mathbf{c}^T K^{\Gamma}_{\theta}\mathbf{c} + \nu \sum_{\gamma=0}^{q}\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{\gamma,j,m} \tag{4.15}$$

where $\theta_{\gamma,j,m} \geq 0$, $\gamma = 1, ..., q$, $j = 0, ..., J$, $m = 1, ..., M_j$.
If θ's are fixed, then (4.15) can be written as

$$\min_{\mathbf{c},b} \parallel \mathbf{Y} - K^{\Gamma}_{\theta}\mathbf{c} - b\mathbf{1}_n \parallel_n^2 + n\lambda_0 \mathbf{c}^T K^{\Gamma}_{\theta}\mathbf{c} \tag{4.16}$$

which is a quadratic minimization problem and the solution is given in Wahba (1990). Let b and c were fixed at their values from (4.16), denote $d_{\gamma,j,m} = K^{\Gamma}_{\gamma,j,m}\mathbf{c}$, and let D be the $n \times (\sum_{\gamma}\sum_{j}(2^j - 1))$ matrix with the (γ, j, m)th column being $d_{\gamma,j,m}$. The $\boldsymbol{\theta}$ that minimizes (4.15) is the same as the solution to

$$\min_{\boldsymbol{\theta}} \parallel \mathbf{z} - D\theta \parallel_n^2 + n\nu \sum_{\gamma=0}^{q}\sum_{j=0}^{J}\sum_{m=1}^{M_j}\theta_{\gamma,j,m} \quad subject\ to \quad \theta_{\gamma,j,m} \geq 0 \tag{4.17}$$

where $\mathbf{z} = \mathbf{Y} - (1/2)n\lambda_0\mathbf{c} - b\mathbf{1}_n$. The formulation in (4.17) is a high-dimensional NNG problem for which there exists many algorithms to find the solution (as discussed in chapter 2 and 3).

By starting from a simpler estimate such as the one obtained by penalized least squares with quadratic penalties on the coefficients, a one step update procedure is sufficient to improve on the WK-ANOVA estimator. Then we propose a one step update procedure:

1. Initialization: Fix $\theta_{\gamma,j,k} = 1$, $\gamma = 1, ..., q$, $j = 0, ..., J$, $m = 1, ..., M_j$.
2. Tune λ_0 using v-fold-cross-validation.
3. Solve for $\mathbf{c}$ and b with (4.16).
4. For each fixed ν, solve (4.17) with the $\mathbf{c}$ and b obtained in step 3. Tune ν using v-fold-cross-validation. The θ's corresponding to the best ν are the final solution at this step.
5. With the new $\boldsymbol{\theta}$ tune λ_0 using v-fold-cross-validation.
6. With the new $\boldsymbol{\theta}$ and λ_0, solve for $\mathbf{c}$ and b with (4.16)

A discussion of a one step procedure and fully iterated procedure can be found in Antoniadis & Fan (2001). The performance of the WK-ANOVA estimator depends on the smoothing parameter ν and the chosen resolution J. The choice of these parameters obviously involves an arbitrary decision. In our work we will fix $J = \log_2 n$, but by varying the resolution level we can explore features of the data arising on different scales. We will use v fold cross validation to tune ν. It seems reasonable to take v equal to 5.

We also choose to use compactly supported wavelets, it follows that the numerical algorithm for the kernel computation is based on Daubechies cascade procedures (Daubechies, 1992). Specifically, the cascade algorithm computes the values of wavelets at dyadic points.

In order to evaluate the kernel matrices $K^{\Gamma}_{\gamma,j,m}$ the values of the wavelets have been computed on a fine dyadic grid and stored in a table. Values of wavelets at arbitrary points, necessary for evaluation of $K^{\Gamma}_{\gamma,j,m}$, were then computed by considering the value at the closest point on the tabulated grid. The table construction of wavelet kernel matrices requires $O(n^2 S)$ elementary operations where S denotes the length of with wavelet filter. However, the table is constructed once and stored in memory. In addition, as the dimension of the problem grows, the number of matrices $K^{\Gamma}_{\gamma,j,m}$ also grows as well, and because of the v-fold-cross-validation these matrices must be re-computed several times. All this increases significantly the computational time, and therefore it is necessary to compute the matrices once and stored them in memory.

4.4 Global sensitivity analysis by WK-ANOVA

It has been shown in chapter 2 that the component functions in the ANOVA decomposition are independent and give information on the input/output relationships. Moreover, the total variance V of the model can be decomposed into its input variable contributions. Using the variance decomposition (2.9) and the WK-ANOVA solution form (4.14) we have

$$V \approx \sum_{l=1}^{d} V_l + \sum_{1\le l<p\le d} V_{lp} \tag{4.18}$$

$$\approx \sum_{\gamma=1}^{q} \int_0^1 \left[\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{\gamma,j,m} \sum_{i=1}^{n} c_i K^{\Gamma}_{\gamma,j,m}(\mathbf{x}_i, \mathbf{x})\right]^2 dX^{(\gamma)} \tag{4.19}$$

where $dX^{(\gamma)} \equiv dX^{(\gamma)}$ for $\gamma = 1, \ldots, d$ and $dX^{(\gamma)} \equiv dX^{(l)} dX^{(p)}$ for $\gamma = d+1, \ldots, q$ with $1 \le j < l \le d$.

Let's consider a N i.i.d random sample from the distribution of $\boldsymbol{X}$, say $\{\boldsymbol{x}_i = (x_{i_1}, \ldots, x_{i_d})^T, i = 1, \ldots, N\}$. The Monte-Carlo estimate of V_j is given by

$$\widehat{V}_l = \frac{1}{N} \sum_{\alpha=1}^{N} \left[\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{l,j,m} \sum_{i=1}^{n} c_i K^{\Gamma}_{l,j,m}(x_{i_j}, x_{\alpha_l})\right]^2 \tag{4.20}$$

Hence the main effect indices (first order sensitivity indices) are estimated as

$$\widehat{S}_j = \frac{\widehat{V}_l}{\widehat{V}} \tag{4.21}$$

where $\widehat{V}$ is the total variance estimation. The estimation of V_{lp} are given by

$$\widehat{V}_{\gamma=lp} = \frac{1}{N} \sum_{\alpha=1}^{N} \left[\sum_{j=0}^{J}\sum_{m=1}^{M_j} \theta_{\gamma,j,m} \sum_{i=1}^{n} c_i K^{\Gamma}_{l,j,m}(x_{i_j}, x_{\alpha_l}) K^{\Gamma}_{p,j,m}(x_{i_p}, x_{\alpha_p})\right]^2 \tag{4.22}$$

Thus, the second order indices are defined as

$$\widehat{S}_{jl} = \frac{\widehat{V}_{jl}}{\widehat{V}} \tag{4.23}$$

Assuming that a truncated form of ANOVA decomposition provides a satisfactory description of the model, the total effect indices estimation is given by

$$\widehat{S}_{T_j} = \widehat{S}_j + \sum_{l \neq j} \widehat{S}_{jl} \tag{4.24}$$

Notice that to compute all the indices (main effect, interaction and total effect) we need only N evaluations of the response surface.

4.5 Simulations

In this section we will study the empirical performance of WK-ANOVA, in terms of prediction accuracy and global sensitivity analysis (GSA). The measure of the prediction accuracy is given by Q_2 which is defined as

$$Q_2 = 1 - \frac{\sum_{i=1}^{n_{test}} (y_i - \widehat{f}(\mathbf{x}_i))^2}{\sum_{i=1}^{n_{test}} (y_i - \bar{y})^2}, \text{with } n_{test} = 500 \tag{4.25}$$

where y_i denotes the ith test observation of the test set, $\bar{y}$ is their empirical mean and $\widehat{f}(x_i)$ is the predicted value. We compare the obtained results with those obtained by COSSO-AIPS and GP. We also compare the methods for different experimental design sizes, uniformly distributed on $[0,1]^d$ and built by maximinLHD procedure. Moreover, different signal to noise ratio were applied $SNR \equiv 1:3$ (high noise) $SNR \equiv 1:7$ (medium noise) and $SNR \equiv \infty$ (without noise), with $SNR = [\text{Var}(f(X))]/\sigma^2$. For each setting of some test examples, we perform 50 times the test.

Concerning the performance in terms of GSA, we will study the accuracy of the total effect indices estimation. Furthermore, we will study the size effect of the sample used to estimate the total effect indices by Monte-Carlo integration.

We used the iterative projected shrinkage algorithm (IPS) to solve the NNG step of the algorithm. Moreover, we fixed $M_j = 2^j - 1$ for $\gamma = 1, \ldots, d$ and $M_j = 1$ for $\gamma > d$, in other words we penalize by the translation parameter k for the main effects and by resolution j for the interaction. This assumption permits us reducing significantly the computational time. The wavelets used in our tests were Daubechies wavelets with 3 vanishing moments. The WK-ANOVA was developed in R. We run the simulations on a computer operated by

32bits-Windows OS, this latter imposes limits on the total memory allocation. Knowing that the storage of the matrices $K^{\Gamma}_{\gamma,j,m}$ is memory consuming, we limit the dimension to our examples to 8 and the sample size to estimate Q_2 and the sensitivity indices to 500. The GP models are fitted by a R code contributed by COUGAR team (IFP).

4.5.1 Example 1

Let's consider an additive model with $\mathbf{X} \in [0,1]^6$, with the following function

$$f(\mathbf{X}) = g_1(X^{(1)}) + g_2(X^{(2)}) + g_3(X^{(3)}) + g_4(X^{(4)}) \tag{4.26}$$

where

$$g_1(t) = t; \quad g_2(t) = (2t-1)^2; \quad g_3(t) = \frac{\sin(2\pi t)}{2-\sin(2\pi t)};$$
$$g_4(t) = 0.1\sin(2\pi t) + 0.2\cos(2\pi t) + 0.3\sin^2(2\pi t) + 0.4\cos^3(2\pi t) + 0.5\sin^3(2\pi t)$$

Therefore $X^{(5)}, X^{(6)}$ are uninformative. We use an experimental design of size $n = 200$, built by maxminLHD, and $SNR \equiv \infty$. Figure 4.1 gives the plot of data observation with the true ANOVA component f_l and theirs WK-ANOVA estimates against inputs $X^{(l)}$, $l = 1, \ldots, 6$. The Q_2 of this WK-ANOVA estimate is equal to 0.96 which is a good performance. However, we can note that the estimation of the linear function component f_1 does suffer from using a wavelet method. Part of the reason is the boundary effects caused by using periodic wavelets.

4.5.2 Example 2

In this first test case, consider an additive model with $X \in [0,1]^8$, with the following function

$$f(\mathbf{X}) = g_1(X^{(1)}) + g_2(X^{(2)}) + g_3(X^{(3)}) + g_4(X^{(4)}) + \epsilon$$

where

$$g_1(t) = 0.1\sin(2\pi t) + 0.2\cos(2\pi t) + 0.3\sin(2\pi t) + 0.4\cos^3(2\pi t) + 0.5\sin^3(2\pi t)$$
$$g_2(t) = (2t-1)^2$$
$$g_3(t) = |\sin(3\pi t)| + \frac{0.5|\sin(5\pi t)|}{2-\sin(4\pi t)}$$
$$g_4(t) = \frac{|\sin(2\pi t)|}{2-\sin(2\pi t)}$$

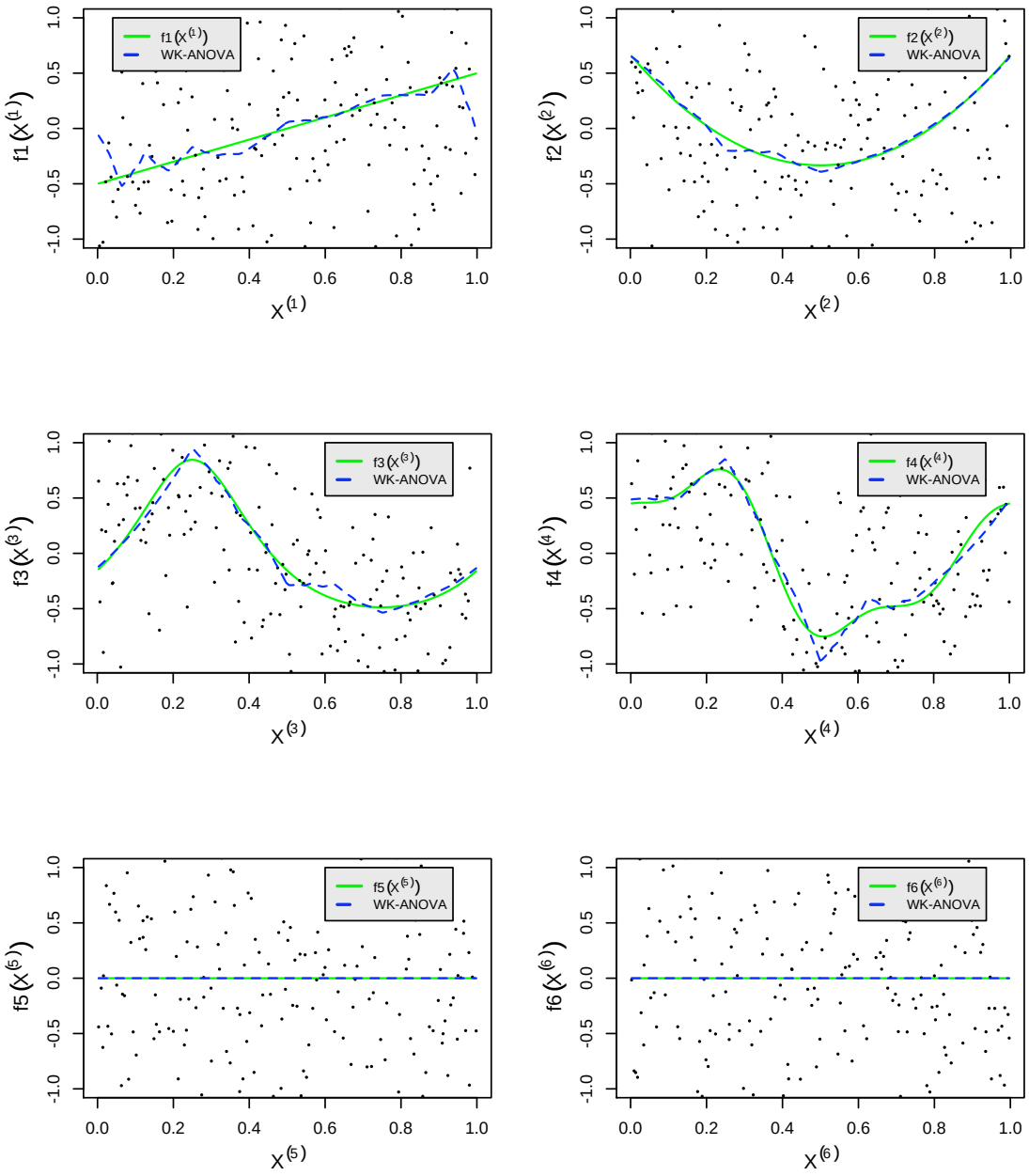

Figure 4.1: Plot of the six true functional components, f_l, $l = 1, \ldots 4$ along with the data observations and theirs estimates given by WK-ANOVA for a realization from example 1

Therefore $X^{(5)}, ..., X^{(8)}$ are uninformative. Note that all informative input of f have nonlinear response, in addition g_3 and g_4 have discontinuities on the derivative. This analytical model is fast enough to evaluate so we can calculate the total effect indices with great precision. Thus the reference values of the indices are computed by direct Monte-Carlo simulation using the Sobol' method (with $N = 250000$, which correspond to 5.10^6 evaluations of the example 2 model). Table 4.1 shows 95% confidence intervals (95%CI) provided by 100 different samples and the reference values that we choose.

Input	Total effect 95%CI	Reference value
$X^{(1)}$	$[0.623, 0.627]$	0.625
$X^{(2)}$	$[0.125, 0.127]$	0.126
$X^{(3)}$	$[0.131, 0.133]$	0.132
$X^{(4)}$	$[0.117, 0.118]$	0.117
$X^{(5,...,8)}$	0	0

Table 4.1: 95% CI and the reference values of the total effect indices for the example 2

4.5.2.1 Assessment of the prediction accuracy

The true ANOVA components f_l, $l = 1, \ldots, n$ with their WK-ANOVA, COSSO-AIPS and GP estimates are given in figure 4.2. These estimates were built with an experimental design of size $n = 200$ and with noise ratio $SNR = 3$. The WK-ANOVA has more fidelity to the reality than COSSO-AIPS and GP especially for the components f_3 and f_4. Indeed, WK-ANOVA captures more the discontinuities of the components f_3 and f_4. This good fit is due to the properties of wavelets analysis. In other words, our algorithm based on wavelets is well suited to this type of functions (with discontinuities on the derivatives).

We run the simulation 50 times for different sizes of experimental design ($n = 50, 100, 200$) and different signal to noise ratio $SNR \equiv 1:3$, $SNR \equiv 1:7$ and $SNR \equiv \infty$. The results are summarized in figure 4.3 each panel is a boxplot of the 50 estimations of Q_2. As expected, the accuracy of WK-ANOVA estimates increases when the sample size raise. We can see that WK-ANOVA procedure outperform COSSO-AIPS and GP in all the studied settings. Moreover, even though there are much more parameters to estimate with WK-ANOVA comparing to COSSO-AIPS, this procedure does not seem to suffer from small sample size effect. For this example, WK-ANOVA has shown better denoising

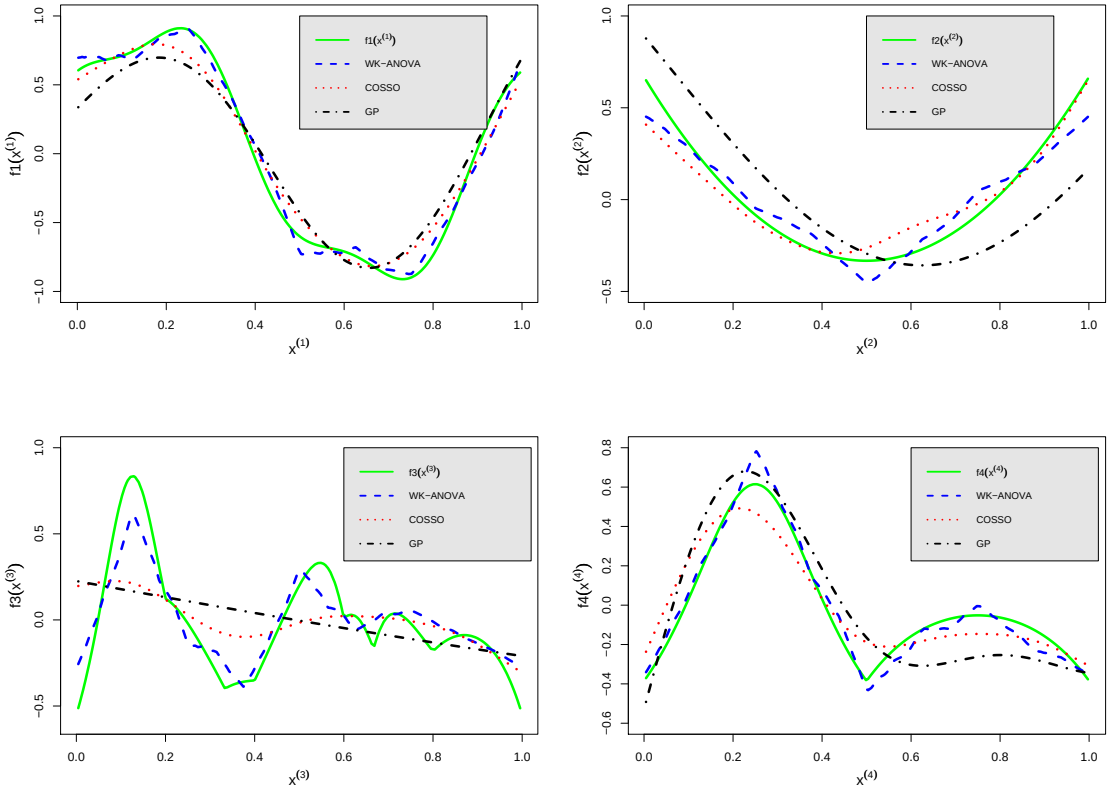

Figure 4.2: Plot of f_l, $l = 1, \dots 4$ along with theirs estimates given by WK-ANOVA, COSSO-AIPS and GP for a realization from example 2

properties and better predictivity. In addition, for $n = 100$ and $n = 200$ WK-ANOVA is the most robust.

4.5.2.2 Global sensitivity analysis

In this section, we apply the WK-ANOVA in order to estimate the total effect indices. We will focus here only on a deterministic problem. We first study the effect of different sample used to compute the indices. Indeed, as discussed previously our WK-ANOVA algorithm has some memory limit, so to avoid this problem we limited the sample size to $N = 500$, which is not sufficient to have a very good robustness for indices estimations. Thus we build using maximinLHD procedure, 100 samples of size $N = 500$, then we estimate the indices using a WK-ANOVA response surface built on an experimental design of size

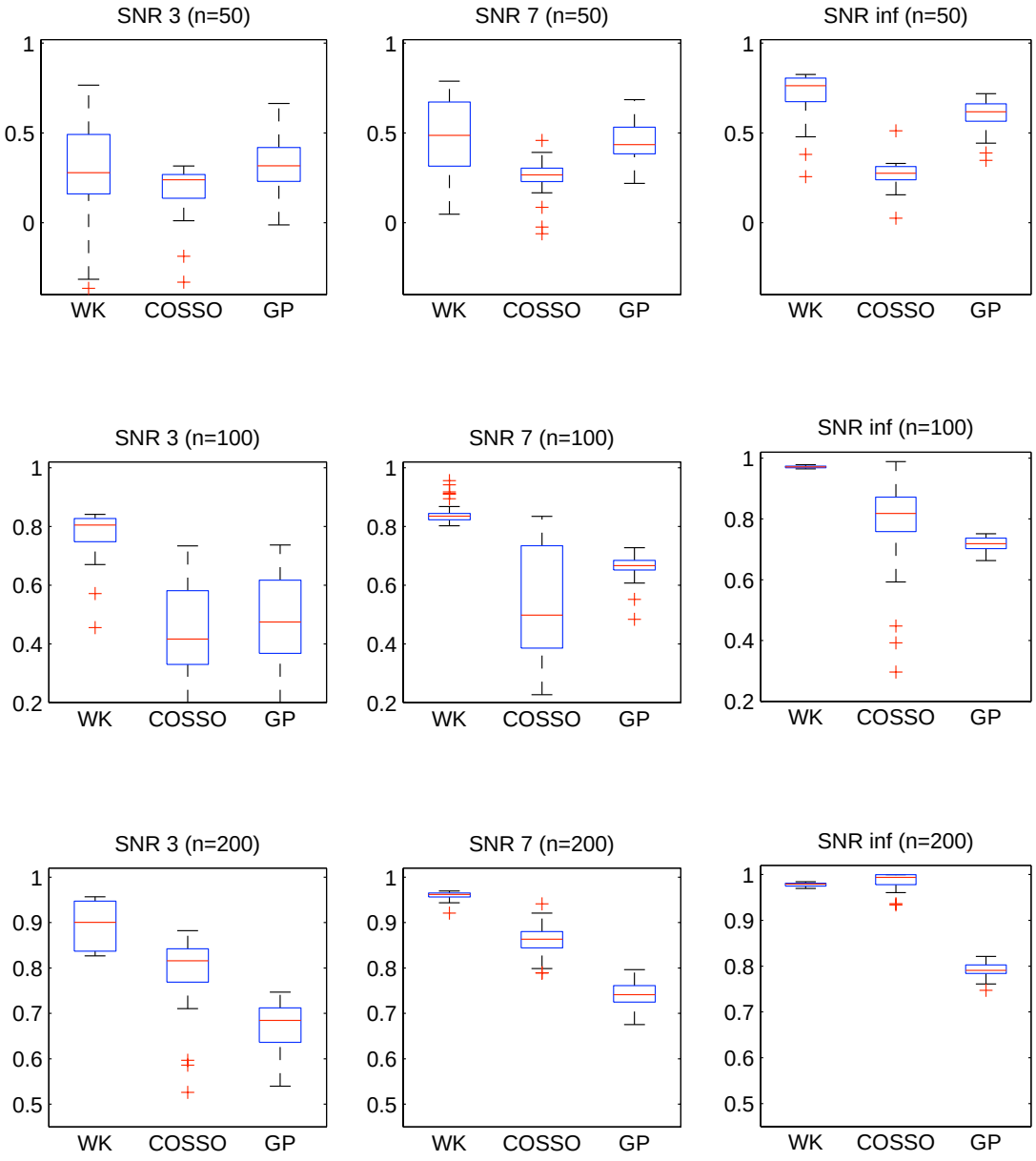

Figure 4.3: Q_2 results from example 2

$n = 200$ and with $Q_2 = 0.96$. Figure 4.4 summarizes the results for this 100 samples. Each panel is a boxplot of the 100 estimations of the total effect index $\widehat{S}_{T_j}$, $j = 1, \ldots, 8$. Dashed lines are drawn at the corresponding reference values of the total effect indices. We see that even if we use a small sample to estimate the indices the robustness of the method remains good.

To study the performance of the estimation of the total effect indices versus sizes of the experimental design, we compute these indices for each of the fifty realizations for $SNR \equiv \infty$ and the for three different sizes ($n = 50$, $n = 100$ and $n = 200$). Figure 4.5 summarizes the results, each panel is a boxplot of the 50 estimations of $\widehat{S}_{T_j}$, $j = 1, \ldots, 8$. Dashed lines are drawn at the corresponding reference values of the total effects indices. It appears that the indices estimation suffers from the small experimental design ($n = 50$). Indeed, WK-ANOVA fails to estimate the input/output relation of $X^{(3)}$, which corresponds to the most non-linear component. In addition, it happens that WK-ANOVA includes some uninformative input into the model. However, the estimation of the total effect indices as well as the variable selection become accurate for experimental designs of sizes $n = 100$ and $n = 200$.

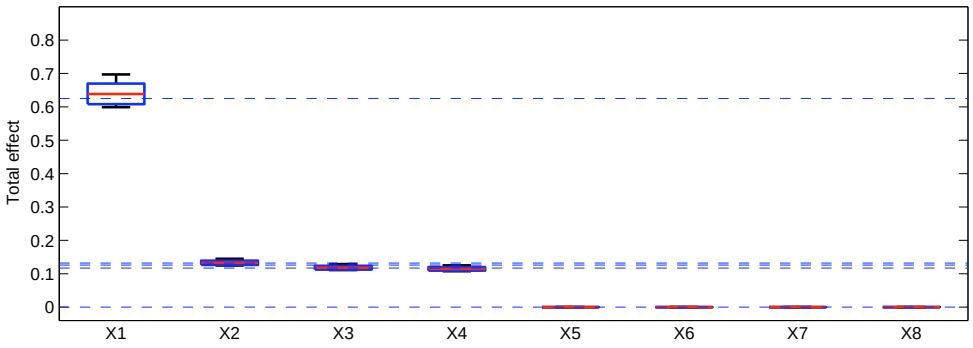

Figure 4.4: Total effect indices vs. sample effect (example 2)

4.5.3 Example 3

Consider the g-Sobol function, described in chapter 3, which is strongly nonlinear, and have non-monotonic relationship. To remind, the g-Sobol is defined for 8 inputs as

$$g_{\text{Sobol}}(X^{(1)}, \ldots, X^{(8)}) = \prod_{k=1}^{8} g_k(X^{(k)}) + \epsilon \quad \text{with} \quad g_k(X_k) = \frac{\left|4X^{(k)} - 2\right| + a_k}{1 + a_k}$$

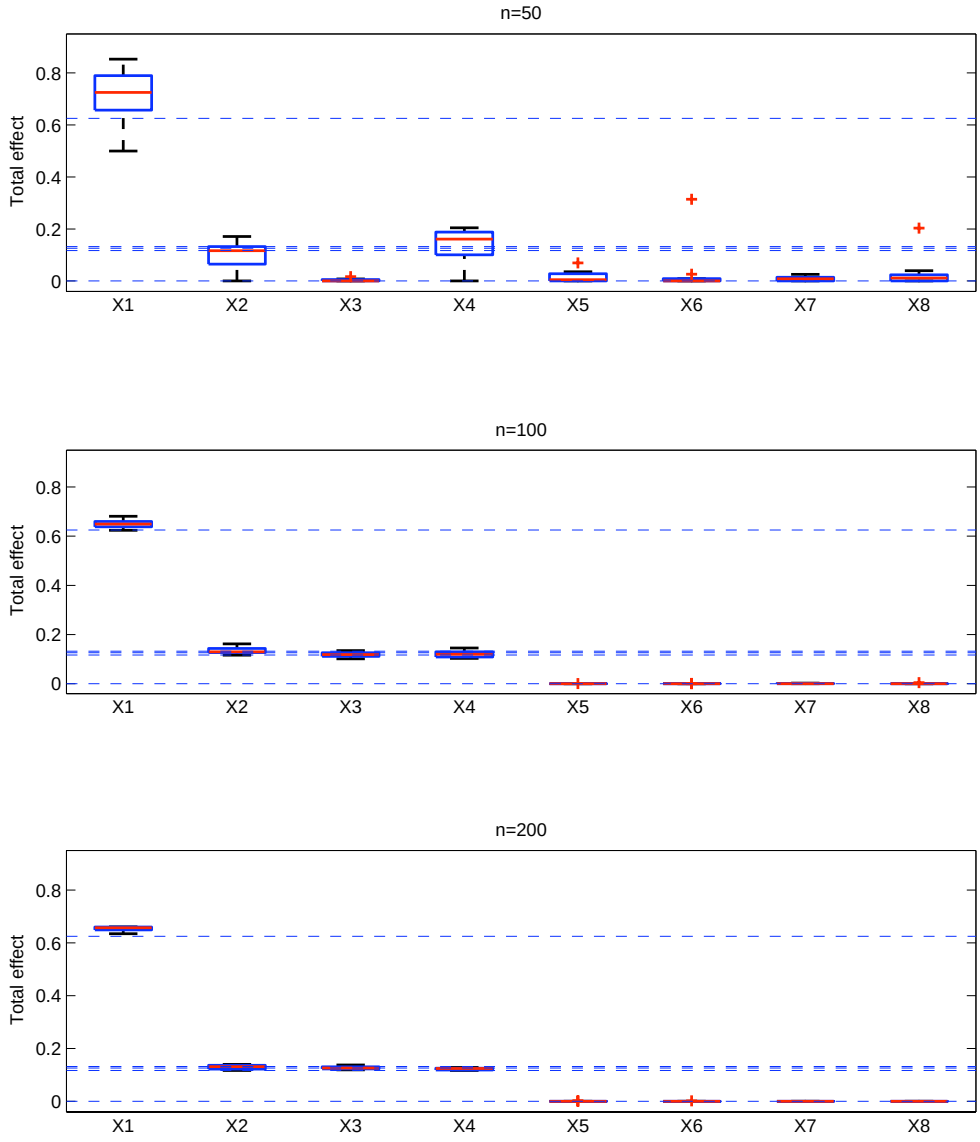

Figure 4.5: Total effect indices vs. experimental design size effect (example 2)

4.5.3.1 Assessment of the prediction accuracy

We run the simulation 50 times for different sizes of experimental design ($n = 50, 100, 200$) and different signal to noise ratio $SNR \equiv 1:3$, $SNR \equiv 1:7$ and $SNR \equiv \infty$. The results are summarized in figure 4.6 each panel is a boxplot of the 50 estimations of Q_2. We can see that for all the tested experimental design sizes and noise ratios WK-ANOVA outperforms COSSO-AIPS and GP. Moreover, the accuracy of the prediction is very good ($\bar{Q}_2 = 0.98$) even with $n = 50$ for the seatting without noise, when a design with $n = 200$ design is necessary to perform a response surface with $\bar{Q}_2 = 0.94$ for GP and with $\bar{Q}_2 = 0.90$ for COSSO. Clearly, for this example WK-ANOVA has the best results in term of predictivity, denoising property and robustness.

4.5.3.2 Global sensitivity analysis

As in the previous example we apply the WK-ANOVA in order to estimate the total effect indices. We will focus here only on a deterministic problem. We first study the effect of different sample used to compute the indices. Thus we build using maximinLHD procedure, 100 samples of size $N = 500$, then we estimate the indices using a WK-ANOVA response surface built on an experimental design of size $n = 200$ and with $Q_2 = 0.99$. Figure 4.7 summarizes the results for these 100 samples. Each panel is a boxplot of the 100 estimations of the total effect index $\widehat{S}_{T_j}$, $j = 1, \ldots, 8$. Dashed lines are drawn at the corresponding analytical values of the total effects indices. We see that even if we use a small sample to estimate the indices the robustness of the method remains good.

To study the performance of the estimation of the total effect indices versus sizes of the experimental design, we compute this indices for each of the fifty realizations for $SNR \equiv \infty$ and for the three different sizes ($n = 50$, $n = 100$ and $n = 200$). Figure 4.8 summarizes the results, each panel is a boxplot of the 50 estimations of $\widehat{S}_{T_j}$, $j = 1, \ldots, 8$. Dashed lines are drawn at the corresponding reference values of the total effects indices. It is clear that GSA with WK-ANOVA response surface provides excellent results with all chosen experimental design sizes.

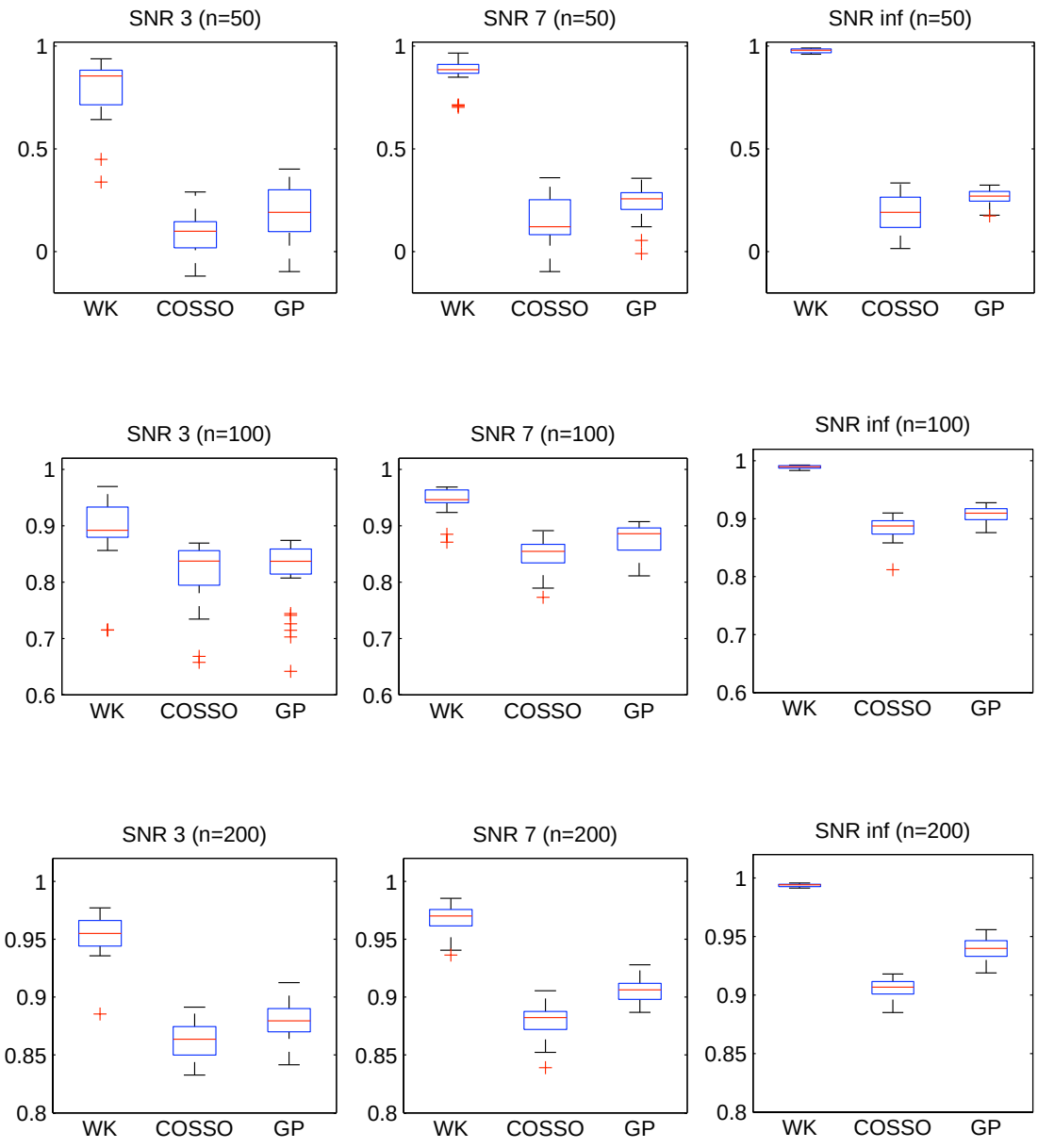

Figure 4.6: Q_2 results from example 3

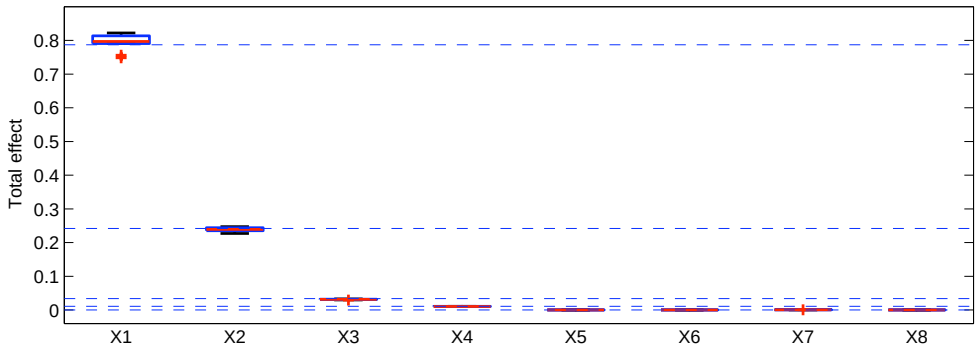

Figure 4.7: Total effect indices vs. sample effect (example 3)

4.6 The reservoir test case

4.6.1 IC Fault model

A description of IC Fault model has been given in section 3.7.2.1. Numerical results performed in the previous chapter demonstrated the complexity of this model due to some discontinuities caused by the presence of a fault. So we expect that using wavelets methods can improve the prediction accuracy.

We remind that the three uncertain parameters are selected independently from uniform distributions with ranges : $h \in [0, 60]$ $k_g \in [100, 200]$ and $k_p \in [0, 50]$. The analysed output is the same as that used in the chapter 3: the oil production rate Q_{op} at 10 years.

4.6.1.1 Assessment of the prediction accuracy

We construct response surfaces using WK-ANOVA of experimental designs of sizes $n = 100$, $n = 200$ and $n = 400$. Q_2 has been estimated on the same sample set ($n_{test} = 25000$) used in the previous chapter for COSSO-AIPS and GP estimate. Table 4.2 shows the performance of WK-ANOVA comparing to the results obtained using COSSO-AIPS and GP. We can see here that COSSO-AIPS outperforms WK-ANOVA and GP for the experimental designs of sizes $n = 200$ and $n = 400$. In addition, WK-ANOVA has very similar results to those obtained by GP. The underperformance of WK-ANOVA comparing to the COSSO-AIPS can be explained by boundary effects caused by the use of periodic wavelets on a non periodic model.

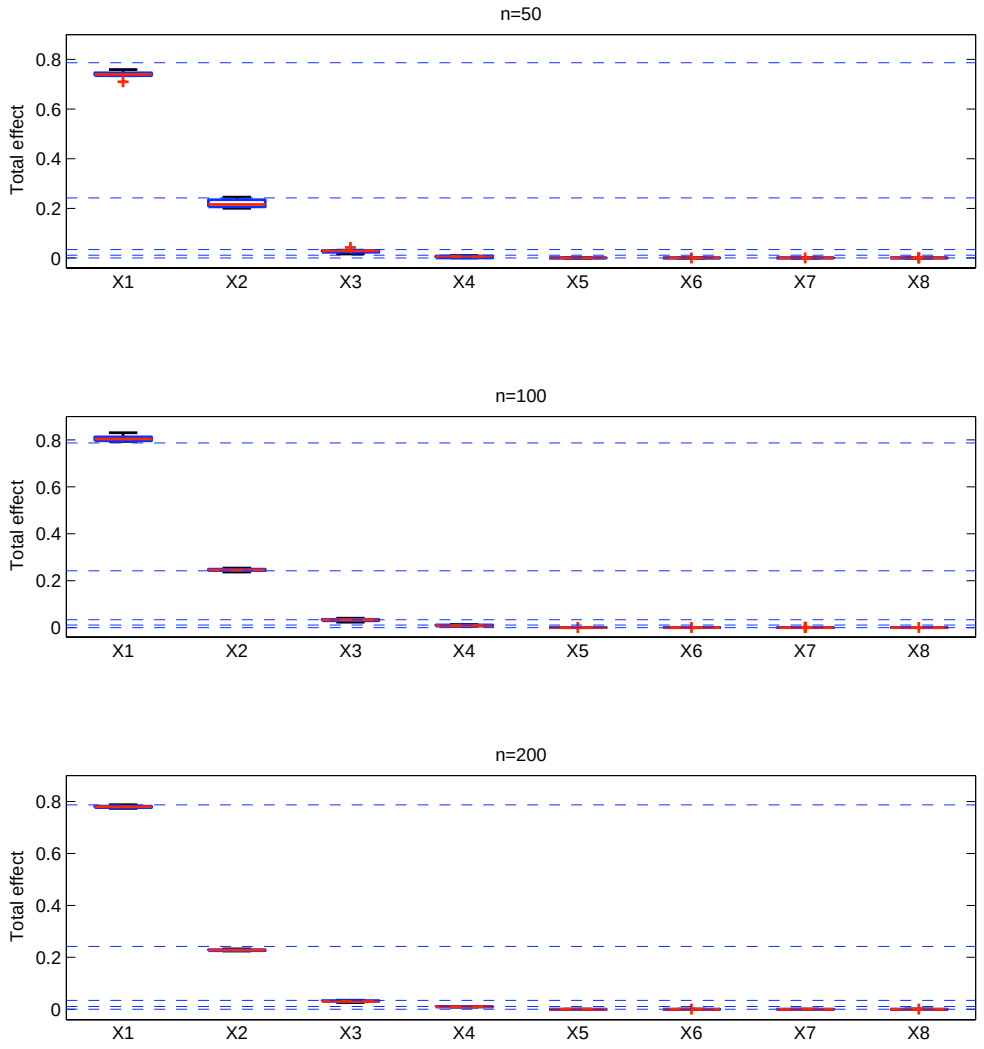

Figure 4.8: Total effect indices vs. experimental design size effect (example 3)

	n	Q_2
COSSO-AIPS	100	0.34
	200	0.63
	400	0.72
	1600	0.81
GP	100	0.33
	200	0.57
	400	0.52
	1600	0.66
WK-ANOVA	100	0.36
	200	0.52
	400	0.59
	1600	–

Table 4.2: Q_2 results from IC Fault model

4.6.1.2 Global sensitivity analysis

To perform GSA we use the response surface built using the $n = 400$ experimental design. To compute the sensitivity indices we use a sample of size $N = 500$. Table 4.3 shows the results compared to those obtained by COSSO-AIPS and GP using the $n = 1600$ experimental design (see chapter 3). Compared with others the GSA results of WK-ANOVA are qualitatively good for this test case.

GP		
Input	Total effect	Main effect
h	0.381	0.100
k_g	0.173	0.021
k_p	0.809	0.596

COSSO-AIPS		
Input	Total effect	Main effect
h	0.375	0.225
k_g	0.030	0.011
k_p	0.733	0.586

WK-ANOVA		
Input	Total effect	Main effect
h	0.261	0.193
k_g	0.085	0.038
k_p	0.753	0.681

Table 4.3: GSA results from IC Fault model

4.7 Conclusion

In this chapter, we introduced a new regularized nonparametric regression method, that we named WK-ANOVA. Differently than other wavelet methods, WK-ANOVA does not require a equispaced experimental design points. In addition, WK-ANOVA is based on ANOVA decomposition, which permits the model output variance to be decomposed into its input contributions. This latter property led us to introduce a direct method to compute Sobol's indices, in the same way as we did for the COSSO method.

We applied WK-ANOVA to a more general multidimensional nonparametric regression problem (by adding a noise term to the model), instead of using WK-ANOVA only as a response surface with deterministic computer codes. This choice was motivated by our desire to show the good denoising property of WK-ANOVA.

For the tested analytical examples which contains some discontinuities, WK-ANOVA outperforms COSSO-AIPS and GP. This good performance is due to the use of wavelets. However, the wavelet methods have undesirable boundary effects (as seen in example 1 and in the IC Fault model) appearing with the estimation of nonperiodic input/output relationship, which are introduced by the use of a periodic wavelets.

Chapter 5

Computer model with time series output

5.1 Introduction

Another challenging problem of response surface frameworks comes up when the analyst plans to study the functional output of the computer models, especially for performing a sensitivity analysis. Some recent works has been done to address this problem. To name few of them, Campbell *et al.* (2006) express the functional output in terms of appropriate functional basis and retain the most important components in its decomposition, then perform the sensitivity analysis on the selected coefficients. Zhang *et al.* (2007) propose a kriging model to investigate computer codes with multiple responses and use an extended functional ANOVA for multiple responses in order to analyze the input effects. Marrel *et al.* (to appear) introduce a method based on wavelet decompositions and Gaussian processes to model the spatial map outputs and then use appropriate Monte-Carlo techniques to estimate the Sobol's indices.

In our considered application, reservoir simulation produce time series datasets. The simplest method, which is widely used, is to include the time as an extra input factor in the model, see, for example, Drignei (2010). In doing so, classical response surface methods can be applied, such as the ones based on Gaussian process modelling. Unfortunately, two complications arise in practice with such an approach: the need to deal with large datasets, which results in a computationally demanding problem (sometimes intractable), and, when the functional outputs are irregular, it may produce oversmooth estimates of the response surface. In a recent work, Bayarri *et al.* (2007) have proposed an original

strategy to handle this problem, which consists in constructing a GP response surface for the selected coefficients of the output's decomposition in a wavelet basis. Indeed, the output corresponding to each design point is a time-series curve, so it is easy to decompose on a wavelet basis. Inspired by this approach, we present in this chapter a methodology based on a COSSO-like method and use a kind of vertical-energy thresholding procedure to select the wavelet coefficients. The goal of this methodology is to perform sensitivity analysis on functional outputs, so we will study sensitivity indices based on a functional ANOVA decomposition.

5.2 The functional response surface methodology

5.2.1 Problem formulation

Consider again a mathematical model of a computer code:

$$\mathbf{Y} = f(\mathbf{X}) \tag{5.1}$$

where $\mathbf{Y} = (Y^{(1)}, \ldots, Y^{(T)})$ is a T-dimensional output vector of the computer code realisations, $\mathbf{X} = (X^{(1)}, \ldots, X^{(d)})$ a d-dimensional input vector which represents the uncertain parameters/factors of the simulator and $f : \mathbb{R}^d \to \mathbb{R}^T$ is an unknown function that models the relationship between the input factors and the output of the computer code.
It is assumed that we have n independent observations $\{(\mathbf{y}_i, \mathbf{x}_i), i = 1, \ldots, n\}$ of a computer code, generated by the relation given in (5.1), where $\mathbf{y}_i = (y_{i_1}, \ldots, y_{i_T})$ and $\mathbf{x}_i = (x_{i_1}, \ldots, x_{i_d})$. It also assumed that each $\mathbf{y}_i = (y_{i_1}, \ldots, y_{i_T})$ represent a time series curve, where T is the number of time sampling points. In many scenarios these curves are irregular, so a wavelet decomposition would be a good choice for basis representation.

5.2.2 Wavelet decomposition

For simplicity, in the following we assume that T is dyadic, which means that $T = 2^J$ for some positive integer J. We also assume that the time sampling points are equally spaced. Then a discrete wavelet transform (DWT) is applied to each $\mathbf{y}_i$ as follow

$$\mathbf{d}^i = W\mathbf{y}_i$$

where $\mathbf{d}^i = (s^i_{0,0}, d^i_{0,0}, \ldots, d^i_{J-1,2^{J-1}-1})^T$ is an T-dimensional vector with components (the discrete) scaling coefficient $s^i_{0,0}$ and (the discrete) wavelet coefficients $d^i_{j,k}$. For more simplicity denote $\mathbf{d}^i = (d^i_1, \ldots, d^i_T)^T$ where d^i_t is the wavelet coefficients at the tth wavelet-position. The matrix W is an orthogonal $T \times T$ matrix associated with the orthonormal

wavelet basis. In addition, $\mathbf{y}_i$ can be recovered using the inverse discrete wavelet transform (IDWT). For more details on DWT and IDWT see appendix C. Note that if there are T time sampling points there will be T wavelet coefficients (include the scaling coefficient at resolution 0).

One of the most important property of wavelet is that many of the coefficients are non-significant so we can use a thresholding procedure to keep only the important ones and set to zero all the others (thresholding). As discussed in appendix C many thresholding procedures exist. The most common strategy in the wavelet literature for treating several data curves, is to apply thresholding procedures to each curve separately. Then, using the union or intersection of wavelet coefficients selected after such individual curves thresholding, the resulting combined set of coefficients is assumed to represent adequately and simultaneously all curves that have been analyzed (see for example Lada *et al.* (2002) and Bayarri *et al.* (2007)). In the framework of meta-modeling the functional outputs of a computer code, Bayarri *et al.* (2007)) apply a hard thresholding to each curve $\mathbf{y}_i$ and then use the union of selected wavelet coefficients to construct a representative set of coefficients for approximating the whole set of original curves. It is clear that this kind of strategy is mainly based on a thresholding criterion that has been developed and is optimal for a single curve analysis, and not for a whole set of curves. Hereafter, we propose to use a criterion that is better adapted to analyze a set of multiple curves.

5.2.3 Vertical energy thresholding

Jung & Lu (2004) introduced the vertical energy thresholding (VET) procedure, which is similar to the classical hard thresholding procedure (see appendix C for more details) and is defined as follows

$$\delta_{\lambda}^{VET}(\mathbf{d}_t) = \begin{cases} \mathbf{0} & \text{if } \| \mathbf{d}_t \|^2 \leq \lambda \\ \mathbf{d}_t & \text{if } \| \mathbf{d}_t \|^2 > \lambda \end{cases} \tag{5.2}$$

where $\mathbf{0}$ is a n-dimensional vector of zeros, $\mathbf{d}_t = (d_t^1, \ldots, d_t^n)^T$ with $t = 1, \ldots, T$ and , for each $t = 1, \ldots, T$, $\| \mathbf{d}_t \|^2$ denotes the squared Euclidian norm of $\mathbf{d}_t$ defined by

$$\| \mathbf{d}_t \|^2 = (d_t^1)^2 + \ldots + (d_t^n)^2.$$

The thresholding parameter λ is chosen by minimizing the following criterion

$$ORRE(\lambda) = \frac{\sum_{t=1}^{T} \| \mathbf{d}_t - \delta_{\lambda}^{VET}(\mathbf{d}_t) \|^2}{\sum_{t=1}^{T} \| \mathbf{d}_t \|^2} + \frac{\sum_{t=1}^{T} I(\| \mathbf{d}_t \|^2 > \lambda)}{T} \tag{5.3}$$

where $I(\cdot)$ denotes the indicator function. Notice that using (5.2) means that when a wavelet coefficient (time index) is selected, the coefficients from all curves at this position will be selected.

Let us define the reconstruction relative error for each curve as follows

$$RE^i = \frac{(\sum_{t=1}^{T}(y_{i_t} - \widehat{y}_{i_t})^2)^{1/2}}{(\sum_{t=1}^{T} y_{i_t}^2)^{1/2}}, \quad i = 1, \ldots, n$$

The criteria ORRE (overall relative reconstruction error) is composed by two components. The first one represents a normalized reconstruction error for the approximated wavelet model. The second one is the normalized number of coefficients selected, which, depending on λ, discourages using a high number of coefficients. It also can be seen as a penalty term for the minimization of the first term in (5.3). Consequently, the idea of the ORRE criterion is to balance the need for approximation accuracy and the need to minimize the complexity of the whole set of curves decompositiont. Ideally, only a small number of coefficients should be selected, which permit us to decrease the computational complexity. For a better approximation accuracy we have decided to retain all the scaling coefficients as well as all wavelets coefficients for levels $j \leq 2$. In other words we threshold the wavelet coefficients vectors $\mathbf{d}_t$ only for $t = 9, \ldots, T$.

5.2.4 Approximating wavelet coefficients with COSSO

Once applied, assume that the VET procedure described above has selected T^* coefficients vectors $\mathbf{d}_t$, and let $\mathcal{D}$ be the index set of the corresponding time indices (i.e. wavelet-positions). Given the fact that a wavelet decomposition has a tendency to decorrelate the resulting coefficients, we will assume that for each time index that has been retained, any component ($i = 1, \ldots, n$) of the corresponding wavelet coefficient $\mathbf{d}_t$, generically say D_t can be considered as a scalar output of a simulator given by the following relation

$$D_t = h^t(\mathbf{X}), \quad t \in \mathcal{D} \tag{5.4}$$

where $\mathbf{X} = (X^{(1)}, \ldots, X^{(d)})$ is a d-dimensional input vector which represents the uncertain parameters/factors of the simulator and $h^t : \mathbb{R}^d \to \mathbb{R}$ is an unknown function that models the relationship between the input factors and the wavelet coefficient corresponding to the tth wavelet-position. It is furthermore assumed that, for each $t \in \mathcal{D}$, the components of corresponding wavelet $\mathbf{d}_t$ form n independent observations $\{(d_t^i, \mathbf{x}_i = (x_{i_1}, \ldots, x_{i_d})), i =$

$1, \ldots, n\}$, generated by the relation (5.4).
The COSSO estimate of h^t is given by the minimizer of

$$\frac{1}{n}\sum_{i=1}^{n}\{d_t^i - h^t(\mathbf{x}_i)\}^2 + \lambda^2 \sum_{\alpha=1}^{q} \parallel P_\alpha h^t \parallel \tag{5.5}$$

The algorithm to solve this convex problem has already been described in chapter 3. Thus, the solution of (5.5) has the following form

$$h^t(\mathbf{x}) = b^t + \sum_{i=1}^{n} c_i^t \sum_{\alpha=1}^{q} \theta_\alpha^t K_\alpha(\mathbf{x}_i, \mathbf{x}) \tag{5.6}$$

where b^t, c_i^t and θ_α^t are the estimates of the parameters corresponding to the approximation of the tth wavelet coefficient. Finally, the approximation of $\mathbf{d}_t$ takes the form $\widehat{\mathbf{d}}_t = K_\theta \mathbf{c} + b\mathbf{1}_n$.

5.2.5 Approximating the computer code

In the previous section we have presented a method to approximate the wavelet coefficients that remain selected after the thresholding procedure. Our goal now is to use these coefficients to build an approximation of the computer code represented by the function f. Indeed, the time series curve $\mathbf{Y}$ can be efficiently approximated using the IDWT by $\widehat{\mathbf{Y}}$ which is defined as follows:

$$\widehat{\mathbf{Y}} = W^T \widehat{\mathbf{d}}$$

where $\widehat{\mathbf{d}} = (\widehat{d}_1, \ldots, \widehat{d}_T)$ with $\widehat{d}_t = 0$ if $t \notin \mathcal{D}$.

5.2.6 The methodology

In summary, to approximate the relationship between the input factors and the functional output of the computer code we apply the following procedure:

1. Wavelet decomposition : apply to each output curve $\mathbf{y}_i$ the DWT.
2. Dimension reduction: perform the VET procedure to the wavelet coefficients obtained in the step 1.
3. Approximation of wavelet coefficients: approximate each h^t functions, which models the relationship between the input factors and the wavelet coefficients retained in the step 2, using COSSO.
4. Approximation of the computer code: apply the IDWT to the approximated wavelet coefficients (in step 3) to obtain the approximation of the time series output $\mathbf{Y}$.

5.3 Monte-Carlo procedure for estimation of time dependent Sobol's indices

In this section we will use the original Sobol's Monte-Carlo estimation method and our methodology to estimate the set of time dependent sensitivity indices. Thus, again, we consider observing an N i.i.d random sample from the distribution of $\mathbf{X}$, say $\{\mathbf{x}_i = (x_{i_1}, \ldots, x_{i_d})^T, i = 1, \ldots, N\}$. For each time index t, the constant $f_0^{(t)}$ and the total variance V^t are estimated by

$$\widehat{f}_0^{(t)} = \frac{1}{N} \sum_{i=1}^{N} \widehat{y}_{i_t} \tag{5.7}$$

$$\widehat{V}^{(t)} = \frac{1}{N} \sum_{i=1}^{N} \widehat{y}_{i_t}^2 - (\widehat{f}_0^{(t)})^2 \tag{5.8}$$

As already described in Chapter 2, Sobol (1993) has used a Monte-Carlo procedure to estimate the variance of the conditional expectations. Thus, the estimation of $V_j^{(t)}$ involves two independent i.i.d. N-sized random samples sets $\{\mathbf{x}_i = (x_{i_1}, \ldots, x_{i_d})^T, i = 1, \ldots, N\}$ and $\{\mathbf{z}_i = (z_{i_1}, \ldots, z_{i_d})^T, i = 1, \ldots, N\}$ from the distribution of $\mathbf{X}$. The Monte-Carlo estimate of V_j^t is then given by

$$\widehat{V}_j^{(t)} = \frac{1}{N} \sum_{i=1}^{N} \widehat{y}_{i_t} \widehat{y}_{i_t}^* - (\widehat{f}_0^{(t)})^2 \tag{5.9}$$

where $\widehat{y}_{i_t}^*$ is the approximated output of the computer code at time t corresponding to the input vector $(z_{i_1}, \ldots z_{i_{j-1}}, x_{i_j}, z_{i_{j+1}}, \ldots, z_{i_d})^T$. Thus, the first order indices are estimated as

$$\widehat{S}_j^{(t)} = \frac{\widehat{V}_j^{(t)}}{\widehat{V}^{(t)}} \tag{5.10}$$

The estimations of $V_{jl}^{(t)}$ are given by the same procedure as

$$\widehat{V}_{jl}^{(t)} = \frac{1}{N} \sum_{i=1}^{N} \widehat{y}_{i_t} \widehat{y}_{i_t}^{**} - (\widehat{f}_0^{(t)})^2 \tag{5.11}$$

where $\widehat{y}_{i_t}^{**}$ is the approximated output of the computer code at time t corresponding to the input vector $(z_{i_1}, \ldots z_{i_{j-1}}, x_{i_j}, z_{i_{j+1}}, \ldots, z_{i_{l-1}}, x_{i_l}, z_{i_{l+1}}, \ldots, z_{i_d})^T$. Thus, the second order indices are estimated by

$$\widehat{S}_{jl}^{(t)} = \frac{\widehat{V}_{jl}^{(t)} - \widehat{V}_j^{(t)} - \widehat{V}_l^{(t)}}{\widehat{V}^{(t)}} \tag{5.12}$$

and so on for obtaining the estimates of the sensitivity indices of higher order. The total effect indices at time t, $S_{T_j}^{(t)}$, can also be estimated directly, without estimating all indices which include the index j. Indeed, once again total effect time dependent indices can be written as

$$S_{T_j}^{(t)} = 1 - \frac{V[E(Y^{(t)}|X^{(-j)})]}{V^{(t)}} = 1 - \frac{V_{-j}^{(t)}}{V^{(t)}} \tag{5.13}$$

where $V_{-j}^{(t)}$ correspond to the variance of the expectation conditioned to all the inputs except $X^{(j)}$. The estimation of $V_{-j}^{(t)}$ is given by

$$\widehat{V}_{-j}^{(t)} = \frac{1}{N}\sum_{i=1}^{N} \widehat{y}_{i_t}\widehat{y}_{i_t}^{***} - (\widehat{f}_0^{(t)})^2 \tag{5.14}$$

where $\widehat{y}_{i_t}^{***}$ is the approximated output of the computer code at time t corresponding to the input vector $(x_{i_1}, \dots x_{i_{j-1}}, z_{i_j}, x_{i_{j+1}}, \dots, x_{i_d})^T$. Hence the estimation of the total effect indices S_{T_j}

$$\widehat{S}_{T_j}^{(t)} = 1 - \frac{\widehat{V}_{-j}^{(t)}}{\widehat{V}^{(t)}} \tag{5.15}$$

5.4 Numerical results

We illustrate here the proposed methodology on an example involving the IC Fault reservoir test case. The empirical performance of the model approximation will be measured at each considered time step using Q_2, which is defined as follows

$$Q_2^{(t)} = 1 - \frac{\sum_{i=1}^{n_{test}}(y_{i_t} - \widehat{y}_{i_t})^2}{\sum_{i=1}^{n_{test}}(y_{i_t} - \bar{y}^{(t)})^2}, \text{with } n_{test} = 1000 \tag{5.16}$$

where y_{i_t} denotes the ith test observation of the test set at time t, $\bar{y}^{(t)}$ is their empirical mean and $\widehat{y}_{i_t}$ is the predicted value at the design point $\mathbf{x}_i$ and time t.

To fit the appropriate COSSO models we have used our COSSO-AIPS algorithm, and to perform the DWT and IDWT we have used the R library "wavethresh" contributed by Guy Nason. The wavelets used in this test were Daubechies wavelets with 3 vanishing moments.

A description of the IC Fault model has been given in section 3.7.2.1. We recall here that the three uncertain parameters are selected independently from uniform distributions with ranges : $h \in [0, 60]$ $k_g \in [100, 200]$ and $k_p \in [0, 50]$. The analyzed output is the time series of the oil production rate reservoir simulator given at the months $5, 6, \dots, 35, 36$, which correspond to $T = 32$.

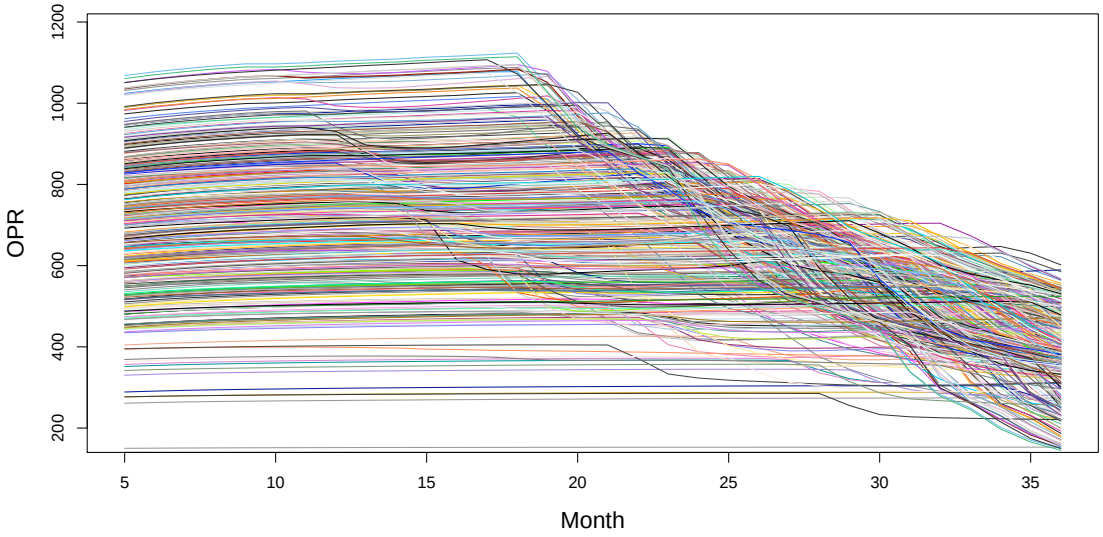

Figure 5.1: Oil production rate from month 5 to month 36

5.4.0.1 Assessment of the prediction accuracy

The simulator is run on an experimental design of size $n = 400$ builts using maximinLHD procedure, figure 5.1 shows the corresponding time series curves. We have then applied our methodology to approximate the model. After performing the VET procedure, 8 of 32 wavelet-positions were selected. To emphasize the good performance of COSSO, we compare the obtained results with those obtained using our methodology but instead of using COSSO we use a GP based approach. Figure 5.2 summarizes the results. We can see here that using COSSO gives us a better predictive approximation than using GP. We can also note that $Q_2^{(t)}$ decreases when approaching the time step 30, part of the reason being that the response becomes more complex in this period of time. However, the accuracy remains satisfactory.

5.4.0.2 Global sensitivity analysis

In addition to the empirical prediction performance we also have studied the empirical GSA performance. To compute the total effect and main effect indices with our COSSO based methodology we use 50 samples of size $N = 10000$ and apply Sobol's Monte-Carlo based estimation method described in the previous section. Figure 5.3 and figure 5.4 show

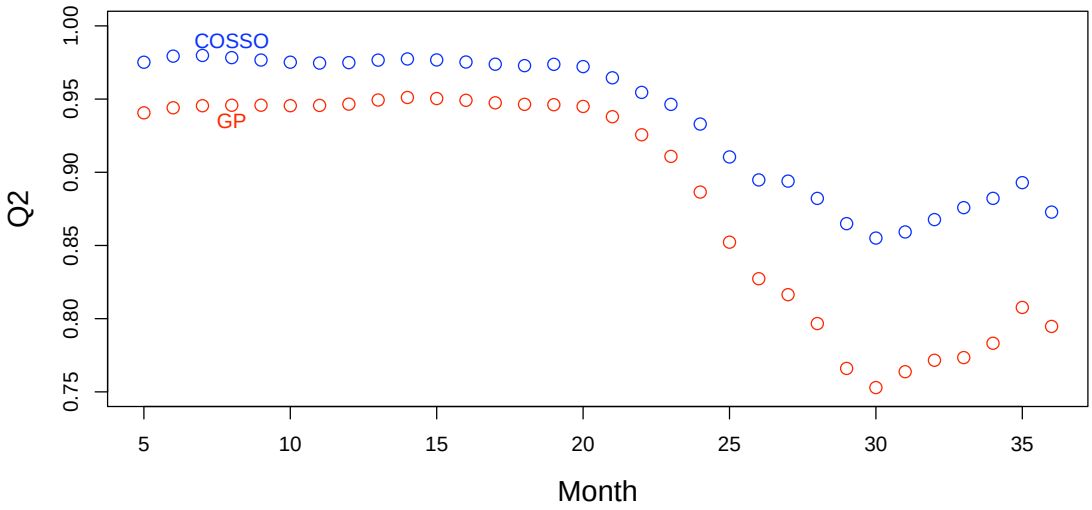

Figure 5.2: Q_2 estimation at each month from month 5 to month 36.

respectively the computed total effect and main effect indices with 95% CI (confidence intervals). We can see that the indices vary differently throughout time. Indeed, when the indices corresponding to k_g increase those associated to h and k_p decrease and the opposite is true. We may also note that until roughly time step 25, the factors h and k_p present small interaction effects and after this time step their main effects decreases to become less important at the end of the studied period.

5.5 Conclusion

The purpose of this chapter has been to introduce an innovative methodology to approximate the computer code with time series output. This methodology is based on an expansion of the time series outputs in a wavelet basis, followed by a vertical energy thresholding procedure (VET), which is designed for analyzing multiple curve sets unlike the classical univariate curve thresholding methods. The latter reduces the dimension of the problem and as such decreases computation complexity. In addition, instead of the widely used GP approach we have used a COSSO-like method, developed and discussed in Chapter 3, to approximate the retained wavelet coefficients. We have also adapted Sobol's Monte-Carlo bases estimation methods to compute time-dependent sensitivity indices. The proposed methodology has been applied to a reservoir test case with success.

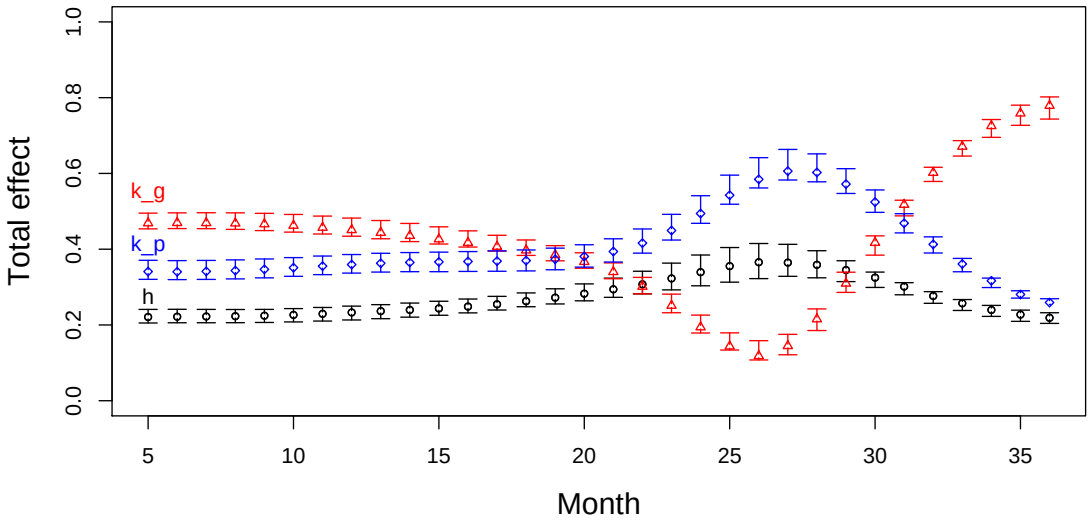

Figure 5.3: Total effect indices estimation with 95% CI at each month from month 5 to month 36

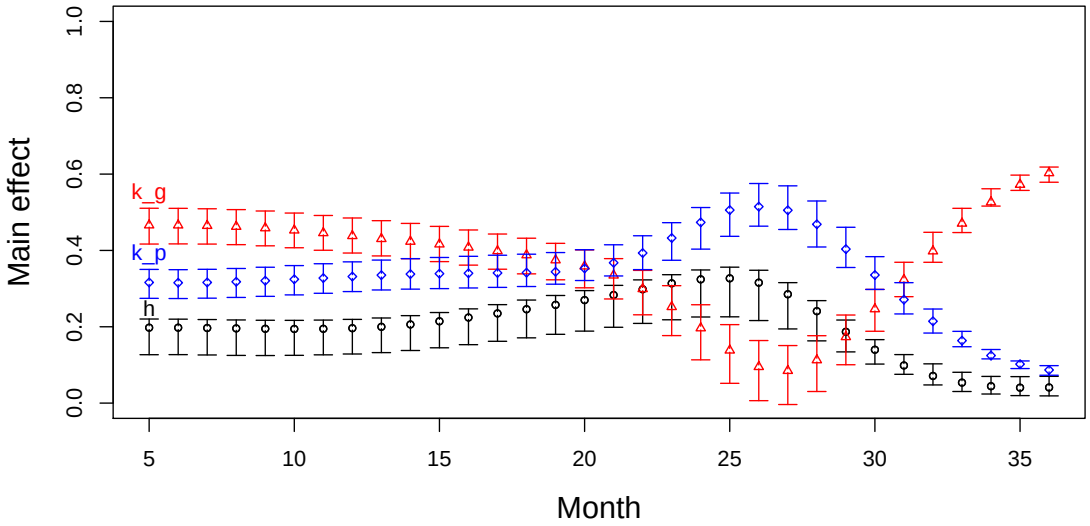

Figure 5.4: Main effect indices estimation with 95% CI at each month from month 5 to month 36

Chapter 6

Conclusion and perspectives

6.1 Conclusion

The purpose of this dissertation is to investigate innovative response surface methods to address the problem of sensitivity analysis for complex and computationally demanding computer codes. To this end, we have focused our research work on methods based on analysis of variance (ANOVA) decomposition. This choice is motivated by the good results of such methods in the framework of nonparametric regression, as well as by the fact that variance based sensitivity analysis (global sensitivity analysis), also relies on ANOVA.
In Chapter 2, we have described Sobol's indices, which are variance based sensitivity indices. These indices quantify the relationship between the variances (uncertainties) of the model inputs and outputs, and provide information on which input factors should be better understood to effectively reduce the uncertainty in the outputs. We have also recalled, the most popular Monte-Carlo simulation based methods that are available to compute Sobol's indices. However, such methods are limited by a high computational burden. Hence, in order to overcome this problem, we may replace the computer code by a response surface that is able of approximating the simulator's output. This approximation, which is generally fast to evaluate, serves to predict outputs from Monte-Carlo simulations and therefore is an efficient tool for computing sensitivity indices. The response surface is constructed by appropriate statistical regression methods, the most common of which have been introduced in Chapter 2.
Chapter 3 was devoted to a regularized nonparametric regression method named component selection and smoothing operator (COSSO). This is an ANOVA based method that is performed using an iterative algorithm, combining a smoothing spline estimation procedure and a nonnegative garrote (NNG) estimation that is somewhat a variable se-

lection procedure. The most common way to solve the NNG problem is to use classical constrained optimization techniques, which are efficient but can become computationally demanding when dealing with high dimensional problems. We propose two remedies to deal with this. First, we develop a new iterative algorithm, called iterative projected shrinkage (IPS) and we also introduce its accelerated version (AIPS). These algorithms are based on a, conceptually simple, Landweber type iterative algorithm. Second, we also adapt in this chapter the nonnegative least angle regression (NN-LARS) algorithm to COSSO, since in standard settings such a type of procedure (LARS) is known to be well-suited for handling high dimensional problems. Extensive simulation results show that our AIPS is the most efficient procedure, both in terms of computational coast and robustness, compared to NN-LARS and other classical solvers.

Using the fact that COSSO is an ANOVA based method, allows us to introduce a new method for computing Sobol's indices. This method seems to be more competitive than the Monte-Carlo Sobol's one because it requires much less response surface evaluations. A comparison is also made with a Gaussian process method, widely used as a response surface technique, and it appears that COSSO-AIPS forms an efficient approach for global sensitivity analysis, especially for high dimensional problems or when a large number of experimental design points is available.

A wavelet approach, expanding COSSO, is introduced in Chapter 4. This new regression method, called wavelet kernel ANOVA (WK-ANOVA), does not require an equi-spaced experimental design, which is generally the rule in the wavelet framework. In the analytical test cases with discontinuities on the derivative, our method produces very good results compared to the COSSO and the Gaussian process approach, and this is due to the multiresolution analysis properties that allow to study a phenomenon in multi-scale fashion. We also show that WK-ANOVA has good denoising properties. Unfortunately, some undesirable boundary effects are present when analyzing models that are nonperiodic and this is due to the fact that we are using periodic wavelet transforms for the decompositions. As a consequence the results from a highly nonlinear reservoir test case do not clearly show the potential of using a wavelet analysis. As in Chapter 3, the WK-ANOVA, by its analogy with COSSO, allows us to develop a direct method for computing Sobol's indices.

Finally, Chapter 5 considers the problem of approximating the computer code when the outputs are time series curves. We propose an original method for performing this task that is based on three main steps: First, an expansion of the time series curves in a wavelet

basis; Second, do reduce the problem's dimension, a vertical energy thresholding procedure (VET) on the resulting decompositions is applied. This procedure, designed for analyzing multiple curve sets use information gives by all time series; Third, an approximation of the selected coefficients is obtained by a COSSO like modeling. This chapter end with an adapted to time series outputs Sobol's Monte-Carlo method for computing sensitivity indices. The efficiency of this methodology is shown on a reservoir test case.
Our work shows the potential of ANOVA based as well as wavelet based decomposition methods to build response surfaces. Of course the proposed methods are not always the most appropriate and need to be improved. Some potential improvements that may be useful and worth developing are discussed below.

6.2 Perspectives

In WK-ANOVA, we have only used Daubechie's periodic wavelet bases for the appropriate transforms, and we have seen that when dealing with nonperiodic phenomena boudary effects can bias the results. It is therefore interesting to take into consideration some other wavelet bases and use boundary adapted wavelet trasforms (Cohen *et al.*, 1993).
In this work, we have assumed that the high-order terms in the ANOVA expansion are negligible compared to the interactions of the second order. However, such an assumption may decrease the accuracy of the approximation in some practical cases. To bypass this limitation, one could use an adaptive strategy for truncating the ANOVA expansions both in COSSO or WK-ANOVA. For instance, such strategy has been discussed in framework of polynomial chaos regression by Blatman (2009). Or else, one can incorporate structural relationships among NNG method, as it has been proposed in Yuan *et al.* (2009).
Knowing that the main objective of a response surface technique is to obtain an accurate approximation which uses as less as possible computer code evaluations, it would be also interesting to explore experimental designs based on active learning strategies, which consist in constructing new observation points located in zones with high uncertainty. Such a strategy has been developed in Gaussian process regression (Busby *et al.*, 2007) and it would be interesting to extend it to our method.
In order to quantify the accuracy of response surfaces, a Q_2 criterion has been employed. This quantity has been estimated on a test sample. However, in practice such estimation

should not require additional computer evaluations. Hence it will be relevant to investigate an inexpensive and robust method to asses the Q_2. It is clear that such assessment is very important for the analyst since the relevance of the sensitivity analysis results will directly depend on the degree of accuracy of the response surface.

At a more practical level, parallelizing the COSSO and WK-ANOVA algorithms could be the most appropriate way to accelerate the methods. Indeed, we have chosen v-fold-cross-validation criterion to tune the regularization parameters. This part of our algorithms is the most computationally demanding, but fortunately it is also very easy to parallelize.

Appendix A

Proofs

Proof of Theorem 3.3.1 The orthogonal projection $\mathcal{P}_\Omega x$ of x onto Ω is characterized by the following useful inequality: for all $a \in \Omega$ and all x we have

$$\langle a - \mathcal{P}_\Omega x, \mathcal{P}_\Omega x - x \rangle \geq 0 \tag{A.1}$$

From the inequality (A.1) we can say that for any Ω, x and z, we have

$$\langle \mathcal{P}_\Omega z - \mathcal{P}_\Omega x, \mathcal{P}_\Omega x - x \rangle \geq 0 \; and \; \langle \mathcal{P}_\Omega z - \mathcal{P}_\Omega x, z - \mathcal{P}_\Omega z \rangle \geq 0 \tag{A.2}$$

Adding, we obtain

$$\langle \mathcal{P}_\Omega z - \mathcal{P}_\Omega x, z - x \rangle \geq \| \mathcal{P}_\Omega z - \mathcal{P}_\Omega x \|^2 \tag{A.3}$$

From the Cauchy inequality we conclude that

$$\| \mathcal{P}_\Omega x - \mathcal{P}_\Omega z \| \leq \| x - z \| \tag{A.4}$$

Let E be nonempty set of all $\boldsymbol{\theta} \in \Omega$ at which the functional (3.13) attains its minimum value over Ω and $\boldsymbol{\theta}^*$ a member of E. Then $\boldsymbol{\theta}^* = \mathcal{P}_\Omega(\delta_\nu^{Soft}(\boldsymbol{\theta}^*))$ and

$$\| \boldsymbol{\theta}^* - \boldsymbol{\theta}^{[p+1]} \| = \| \mathcal{P}_\Omega(\delta_\nu^{Soft}(\boldsymbol{\theta}^*)) - \mathcal{P}_\Omega(\delta_\nu^{Soft}(\boldsymbol{\theta}^{[p]})) \| \leq \| \delta_\nu^{Soft}(\boldsymbol{\theta}^*) - \delta_\nu^{Soft}(\boldsymbol{\theta}^{[p]}) \| \tag{A.5}$$

The convergence of the IPS algorithm follows from the following Lemma.

Lemma A.0.1 *(Lemma* 3.4 *of Daubechies* et al. *(2004))*
$\mathcal{S}_\nu$ is nonexpansive, i.e., for all x and $z \in \mathbb{R}$,

$$\| \delta_\nu^{Soft}(x) - \delta_\nu^{Soft}(z) \| \leq \| x - z \| \tag{A.6}$$

Since (A.9) is convex the Karush-Kuhn-Tucker Theorem suggests that a necessary and sufficient condition for $\boldsymbol{\theta}^*$ to be the solution of model (3.13) is that there is $\lambda \geq 0$ such that, for any $\gamma = 1, ..., q$, $j = 0, ..., J$, $k = 1, ..., 2^j - 1$

$$\{-\mathbf{d}_\alpha^T(\mathbf{Y} - D\boldsymbol{\theta}^*) + \lambda\}\theta_\alpha^* = 0 \tag{A.7}$$

$$-\mathbf{d}_\alpha^T(\mathbf{Y} - D\boldsymbol{\theta}^*) + \lambda \ \geq 0 \tag{A.8}$$

$$\theta_\alpha^* \geq 0 \tag{A.9}$$

There are two possible value for $\boldsymbol{\theta}^{[p+1]}$ in (3.22) :

$$\theta_\alpha^{[p+1]} = \begin{cases} 0 \\ \theta_\alpha^{[p]} + \mathbf{d}_\alpha^T(\mathbf{Y} - D\boldsymbol{\theta}^{[p]}) - \lambda \end{cases} \tag{A.10}$$

It is not difficult to show that the KKT conditions are satisfied when $\theta_\alpha^{[p+1]} = 0$ and that the condition (A.9) is always satisfied. Now we consider the second possibility. If θ_α^* is a fixed point of the map T, with $Tx = \mathcal{P}_\Omega(\mathcal{S}_\nu(x))$, that is, $T\theta_\alpha^* = \theta_\alpha^*$. By (A.10), we have

$$d_\alpha^T(Y - D\boldsymbol{\theta}^*) - \lambda = 0$$

The conclusion follows immediately.

Proof of Theorem 4.3.2 The condition on the unknown regression function f_0 are only active for its wavelets coefficients and do not include the V_0 scaling coefficients of f_0. For any $f \in \mathcal{F}$, write $f(\mathbf{x}) = b + f_1(x^{(1)}) + ... + f_d(x^{(d)}) = b + g(\mathbf{x})$, such that $\sum_{i=1}^n f_l(x_i^{(l)}) = 0$, $l = 1, ..., d$ and where $b \in \{1\}$ and $g \in \bigoplus_{l=1}^d \mathcal{H}_\Gamma^l$. Similarly, write $f_0(\mathbf{x}) = b_0 + g_0(\mathbf{x})$, such that $g_0 \in \bigoplus_{l=1}^d \mathcal{H}_\Gamma^l$. By construction $\sum_{i=1}^n \{g_0(\mathbf{x}_i) - g(\mathbf{x}_i)\} = 0$, we can write $A(f)$ as :

$$(b - b_0)^2 + \frac{2}{n}(b - b_0)\sum_{i=1}^n \varepsilon_i + \frac{1}{n})\sum_{i=1}^n (g_0(\mathbf{x}_i) + \varepsilon_i - g(\mathbf{x}_i))^2 + \lambda_n^2 R_q(g)$$

therefore, the minimizing $\widehat{b}$ is $\widehat{b} = b_0 + 1/n \sum_{i=1}^n \varepsilon_i$, which shows that $\widehat{b}$ converges towards b_0 at rate $n^{-1/2}$. On the other hand, $\widehat{g}$ must minimize over $\bigoplus_{l=1}^d \mathcal{H}_\Gamma^l$ the functional

$$\frac{1}{n}\sum_{i=1}^n \{g_0(\mathbf{x}_i) + \varepsilon_i - g(\mathbf{x}_i)\}^2 + \lambda_n^2 R_q(g)$$

Let $\mathcal{G} = \{g \in \mathcal{F} : g(x) = f_1(x^{(1)}) + ... + f_d(x^{(d)}), with \ \sum_{i=1}^n f_l(x_i^{(l)}) = 0, l = 1, ..., d\}$. the $g_0 \in \mathcal{G}$ $\widehat{g} \in \mathcal{G}$. The conclusion of Theorem 2 follows from the following Lemma.

Lemma A.0.2 *(Theorem* 10.2 *of Van de Geer, lemma* 5.1 *of (Amato* et al.*, 2006) and lemma* 3 *ofLin & Zhang (2006))*
Let $H_\infty(\delta, \mathcal{G})$ *be the* $\delta-$*entropy of* $\mathcal{G}$ *for the supremum norm. Then*

$$H_\infty(\delta, \{g \in \mathcal{G} : R_q(g) \leq 1\} \leq Ad^{(s+1)/s}\delta^{-1/s},$$

for all $\delta > 0$*,* $n \geq 1$*, and some* $A > 0$ *and* $0 < 1/s < 2$*.*

Proof of Lemma Define $\mathcal{G}^l$ as the set of univariate function of $x^{(l)}$.

$$\mathcal{G}^l = \{f_l \in \mathcal{H}_\Gamma^l : R_q(f_l) \leq 1, \sum_{i=1}^n f_\gamma(x_i)^{(l)} = 0\}$$

It follows from Lemma 5.1 of (Amato *et al.*, 2006) that

$$H_\infty(\delta, \mathcal{G}^l) \leq A\delta^{-1/s}$$

for all $\delta > 0$, and $n \geq 1$, some $A > 0$ and $0 < 1/s < 2$. By definition of $\mathcal{G}$ we see that in terms on the supreme norm, if each $\mathcal{G}^l$, $l = 1, .., d$ can be covered by N balls of radius δ, then the set $\{g \in \mathcal{G} : R_q(g) \leq 1\}$ can be covered byN^d balls with radius $d\delta$, and we get :

$$H_\infty(d\delta, \{g \in \mathcal{G} : R_q(g) \leq 1\}) \leq Ad\delta^{-1/s}$$

Proof of Lemma 4.3.3 For any $f \in \mathcal{F}$, write $f = b + \sum_{\gamma=1}^q \sum_{j=0}^J \sum_{k=0}^{2^j-1} f_{\gamma,j,k}$ with $b \in \{1\}$ and $f_{\gamma,j,k} \in \mathcal{W}_{\gamma,j,k}^\Gamma$. Let the projection of $f_{\gamma,j,k}$ onto $span\{K_{\gamma,j,k}^\Gamma(\mathbf{x}_i, \cdot), i = 1, ..., n\} \subset \mathcal{W}_{\gamma,j,k}^\Gamma$ be denoted by $\alpha_{\gamma,j,k}$ and the orthonormal complement by $\beta_{\gamma,j,k}$. Then $f_{\gamma,j,k} = \alpha_{\gamma,j,k} + \beta_{\gamma,j,k}$ and (4.10) can be written as

$$\frac{1}{n}\sum_{i=1}^n \{y_i - b - \sum_{\gamma=1}^q \sum_{j=0}^J \sum_{k=0}^{2^j-1} \langle K_{\gamma,j,k}^\Gamma(\mathbf{x}_i, \cdot), \alpha_{\gamma,j,k}\rangle\}^2 + \lambda^2 \sum_{\gamma=1}^q \sum_{j=0}^J \sum_{k=0}^{2^j-1} (\| \alpha_{\gamma,j,k} \|^2 + \| \beta_{\gamma,j,k} \|^2)^{1/2}$$

Therefore any minimizing f must be such that $\beta_{\gamma,j,k} = 0$, and the result follows immediately.

Proof of Lemma 4.3.4 Denote the functional in (4.12) by $B(\theta, f)$. For any $\gamma = 1, ..., q; j = 0, ..., J; k = 1, ..., 2^j - 1$, we have

$$\lambda_0\theta_{\gamma,j,k}^{-1} \| P_{\gamma,j,k}^\Gamma f \|_{\mathcal{W}_{\gamma,j,k}^\Gamma}^2 + \nu\theta_{\gamma,j,k} \geq 2\lambda_0^{1/2}\nu^{1/2} \| P_{\gamma,j,k}^\Gamma f \|_{\mathcal{W}_{\gamma,j,k}^\Gamma} = \lambda^2 \| P_{\gamma,j,k}^\Gamma f \|_{\mathcal{W}_{\gamma,j,k}^\Gamma} .$$

for any $\theta_{\gamma,j,k} \geq 0$ and $f \in \mathcal{F}$, and the equality holds if and only if $\theta_{\gamma,j,k} = \lambda_0^{1/2}\nu^{-1/2} \| P_{\gamma,j,k}^\Gamma f \|_{\mathcal{W}_{\gamma,j,k}^\Gamma}$. Therefore $B(\theta, f) \geq A(f)$, where $A(f)$ denote the functional of (4.10) for any $\theta_{\gamma,j,k} \geq 0$, $\gamma = 1, ..., q$; $j = 0, ..., J$; $k = 1, ..., 2^j - 1$ and $f \in \mathcal{F}$, with the equality holds only if $\theta_{\gamma,j,k} = \lambda_0^{1/2}\nu^{-1/2} \| P_{\gamma,j,k}^\Gamma f \|_{\mathcal{W}_{\gamma,j,k}^\Gamma}$. The conclusion then follows.

Appendix B

Reproducing kernel Hilbert spaces

The objective of this appendix is to review the basic theory on reproducing kernel Hilbert spaces (RKHS).

B.1 RKHS definition

Let $\mathcal{X}$ be a nonempty set. The RKHS is defined as a Hilbert space $\mathcal{H}$ of functions on a set $\mathcal{X}$ with the following property: $\forall x \in \mathcal{X}$ and $f \in \mathcal{H}$, there exist M_x not depending on f satisfying $\mid f(x) \mid \leq M_x \parallel f \parallel$. The Riesz representation theorem (Akhiezer & Glazman (1963)) states that there exist a unique representer η_x in $\mathcal{H}$ with associated linear functional $\delta_x : \mathcal{H} \to \mathbb{R}$, defined by $\delta_x(f) = f(x)$, such that

$$f(x) =< \eta_x, f >, \qquad \forall f \in \mathcal{H}$$

where $<,>$ is the inner product in $\mathcal{H}$. Let $< \eta_x, \eta_{x'} >= K(x, x')$. Therefore $K(x, x')$ is positive definite on $\mathcal{X} \otimes \mathcal{X}$, i.e, $\sum_{i,j} a_i a_j K(x_i, x'_j) \geq 0$, $\forall x_i, x'_j \in \mathcal{X}$ with $i = 1, \ldots, n$ and $j = 1, \ldots, n$. K is so-called the reproducing kernel for the RKHS $\mathcal{H}$.
In above, we showed that the reproducing kernel K for the RKHS $\mathcal{H}$ is positive definite on $\mathcal{X} \otimes \mathcal{X}$. The Moore-Aronszajn theorem state that for every positive definite kernel K on $\mathcal{X} \otimes \mathcal{X}$ there exists a unique RKHS. The Hilbert space associated with K can be constructed as containing all finite linear combinations of the form $\sum a_i K(x_i, \cdot)$.

B.2 The representer theorem

The representer theorem (Kimeldorf & Wahba, 1971) shows that solution of the optimzation problem defined as finding $f \in \mathcal{H}$ to minimize

$$\sum_{i=1}^{n} \mathcal{L}(y_i, f(x_i)) + \lambda \parallel f \parallel_{\mathcal{H}^2} \tag{B.1}$$

where $\mathcal{L}$ is convex in f, has a representation of the form

$$f(x) = \sum_{i=1}^{n} c_i K(x_i, x) \tag{B.2}$$

Then $B.2$ is substituted in $B.1$ and the $c_i's$ are found numerically. In our work we study the penalized least squares problems. Thus, $B.1$ is equivalent to

$$\sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \parallel f \parallel_{\mathcal{H}}$$

For this special case $\mathcal{L}$ is quadratic and convex. Hence, it is only necessary to solve a linear system. Note that we can replace $\parallel f \parallel^2$ by $\parallel Pf \parallel^2$ where P is the orthogonal projection onto a subspace of small co-dimension, for more detail we refer to Wahba (1990) and Berlinet & Thomas-Agnan (2003).

Appendix C

A short review of wavelet

C.1 Introduction

Joseph Fourier introduces the idea that a signal can be represented using superposition of sines and cosines. A disadvantage of the Fourier expansion is that it has only frequency resolution and no time resolution, in other words it is unable of dealing properly with a signal that is changing over time. Several methods were developed to adapt the usual Fourier method to represent a signal in the time and frequency domain at the same time. The idea behind these representations is to cut the signal into several parts and then analyze the parts separately, but how we should cut the signal? The Heisenberg's uncertainty principle states that in modeling time-frequency phenomena, it is impossible to know the exact frequency and the exact time simultaneously. In other words, the area of rectangles which represent the window of localization in the time-frequency space are bounded by a universal constant. For example, the windowed Fourier transform (Fig C.1) has a single window, which is used for all the frequencies, the resolution of the analysis is constant at all locations in the time-frequency plane.
Wavelets have the advantage that the window trade-off automatically the time-frequency precision, thus solving the problem of cutting the signal. Figure C.1 shows that for different scale a different size of windows are shifted along the signal. In the end the result will be a collection of different resolutions of time-frequency representations. Therefore, we can speak of a multiresolution analysis.

In what follows we will present a brief review on wavelets theory and their utilization in nonparametric regression.

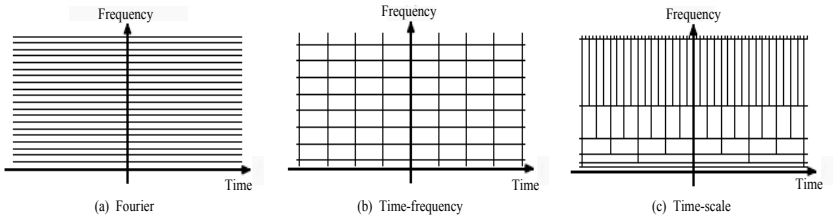

Figure C.1: Time-frequency plane for Fourier, time-frequency and time-scale representation

C.2 Multiresolution analysis

The main idea behind the multiresolution analysis is to define a sequence of closed subspaces V_j, $j \in \mathbb{Z}$ in $L_2(\mathbb{R})$, which possesses the following properties:

1. $\cap_{j\in\mathbb{Z}} V_j = 0$,
2. $\overline{\cup_{j\in\mathbb{Z}} V_j} = \boldsymbol{L}^2(\mathbb{R})$,
3. $\forall f \in \boldsymbol{L}^2(\mathbb{R})$, $\forall j \in \mathbb{Z}$, $f \in V_j$ if and only if $f(2x) \in V_{j+1}$;
4. $\forall f \in \boldsymbol{L}^2(\mathbb{R})$, $\forall k \in \mathbb{Z}$, $f \in V_0$ if and only if $f(x-k) \in V_0$,
5. there exist a scaling function $\phi \in V_0$ whose integer-translates $x \mapsto \phi(x-k)_{k\in(Z)}$ span the space V_0.

This implies that for each resolution j, the set of functions $\{\phi_{j,k}(x) = 2^{j/2}\phi(2^j x - k);\ k \in \mathbb{Z}\}$ constitutes the orthonormal basis of the space V_j for L_2 norm. Remark that each scaling function is indexed by two indices j (resolution) and k (translation). A unit increase in j compresses the function ϕ along the x-axis into half the width. A unit increase in k shifts the location of ϕ along the x-axis.
Since $\phi \in V_0$ and $V_0 \subset V_1$, ϕ can be represented as a linear combination of $\{\phi_{1,k}\}_{k\in(Z)}$. In other words, their exist coefficients a_k, $k \in (Z)$ such as

$$\forall x \in \mathbb{R}, \phi(x) = \sum_{k\in\mathbb{Z}} a_k \sqrt{2}\phi(2x-k) \tag{C.1}$$

As we will see later this equation (two-scale equation) is fundamental in constructing efficient algorithm to perform multiresolution analysis.
If we define $P^j f$ as the projection of a function f into the space V_j, this is expressed

$$P^j f = P^{j-1} f + w^{j-1}$$

The function w^{j-1} represent the residual between the two approximations (on V^j and on V^{j-1}), this function can be written in terms of dilated and translated wavelets:

$$w^{j-1} = \sum_{k\in\mathbb{Z}} \langle f \mid \psi_{j-1,k}\rangle \psi_{j-1,k}$$

where $\{\psi_{j,k}(x) = 2^{j/2}\psi(2^j x - k);\ k \in \mathbb{Z}\}$ is a set of functions that are orthogonal to each function of V_j and span the space W_j which is a detail space. Hence, an important property of multiresolution analysis can be defined as:

$$V_j = V_{j-1} \oplus W_{j-1} \tag{C.2}$$

Following the properties given before, C.2 can be extended recursively until for a given $j_0 \in \mathbb{Z}$, the space $\boldsymbol{L}^2(\mathbb{R})$ can be written as:

$$\boldsymbol{L}^2(\mathbb{R}) = \overline{V_{j_0} \oplus \bigoplus_{j=j_0}^{+\infty} W_j} \tag{C.3}$$

Since $\psi(x) \in W_0$ and $W_0 \subset V_1$ a two-scale equation can be set up. Their exist coefficients b_k, $k \in (Z)$ such that

$$\forall x \in \mathbb{R}, \psi(x) = \sum_{k\in\mathbb{Z}} b_k \sqrt{2}\phi(2x - k) \tag{C.4}$$

Using the decomposiotn of the space $\boldsymbol{L}^2(\mathbb{R})$ given in C.3 and for all $j_0 \in \mathbb{Z}$, we can now write any $\boldsymbol{L}^2(\mathbb{R})$ function f as

$$\forall x \in \mathbb{R}, f(x) = \sum_{k\in\mathbb{Z}} \alpha_{j_0,k}\phi_{j_0,k}(x) + \sum_{j\geq j_0}\sum_{k\in\mathbb{Z}} \beta_{j,k}\psi_{j,k}(x)$$

where $\alpha_{j_0,k} = \int f(x)\phi_{j_0,k}(x)dx$ and $\beta_{j,k} = \int f(x)\psi_{j,k}(x)dx$

C.2.1 Periodic wavelet

Because the functions that we want to estimate are defined in $\boldsymbol{L}^2([0,1])$, we construct an orthonormal wavelet bases that spans $\boldsymbol{L}^2([0,1])$ instead of $\boldsymbol{L}^2(\mathbb{R})$. In other words, we periodize scaling and wavelet functions by the following transformation

$$\phi_{j,k}^{per}(x) = \sum_{l\in\mathbb{Z}} \phi_{j,k}(x + l)$$
$$\psi_{j,k}^{per}(x) = \sum_{l\in\mathbb{Z}} \psi_{j,k}(x + l)$$

Furthermore, we define the spaces V_j^{per} and W_j^{per} as

$$V_j^{per} = \overline{span\{\phi_{j,k}^{per}, k \in \mathbb{Z}\}}$$
$$W_j^{per} = \overline{span\{\psi_{j,k}^{per}, k \in \mathbb{Z}\}}$$

The resulting orthogonal basis provides an orthogonal decomposition of $\boldsymbol{L}^2([0,1])$

$$\boldsymbol{L}^2([0,1]) = \overline{V_{j_0}^{per} \oplus \bigoplus_{j=j_0}^{+\infty} W_j^{per}}$$

which is a multiresolution analysis on $[0,1]$. A major disadvantage of periodized wavelet is the introduction of edge effects at the end points $x = 0$ and $x = 1$. More details about periodized wavelet can be found in Daubechies (1992), Maxim (2003) and Vidakovic (1999).

Then any $\boldsymbol{L}^2([0,1])$ function can be written as

$$\forall x \in \mathbb{R}, f(x) = \sum_{k=0}^{2^{j_0}-1} \alpha_{j_0,k}\phi_{j_0,k}^{per}(x) + \sum_{j \geq j_0} \sum_{k=0}^{2^j-1} \beta_{j,k}\psi_{j,k}^{per}(x) \tag{C.5}$$

where $\alpha_{j_0,k} = \int f(x)\phi_{j_0,k}^{per}(x)dx$ and $\beta_{j_0,k} = \int f(x)\psi_{j_0,k}^{per}(x)dx$,and the restriction on the parameter k is due to the periodicity of the studied function. In what follows, to enhance the interpretability we omit the index *per*.

C.2.2 Some wavelet basis

The different wavelet bases make different compromise between how compactly the basis functions are localized in space and how smooth they are. In statistics, the important properties that wavelet basis should posses is the orthonormality, the functions that we study are scaled on interval $[0,1]$, so it will be important to construct compactly supported wavelet. In this section we will discuss briefly two important families of wavelets.

C.2.2.1 Haar's wavelet

Introduced by Alfred Haar, long before an established wavelets theory was developed, the Haar scaling and wavelet function is defined as

$$\phi(x) = \begin{cases} 1 & \text{if } 0 \leq x < 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\psi(x) = \begin{cases} 1 & \text{if } 0 \leq x < \frac{1}{2} \\ -1 & \text{if } \frac{1}{2} \leq x < 1 \\ 0 & \text{otherwise} \end{cases}$$

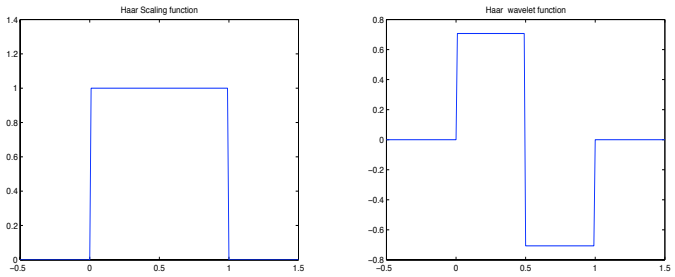

Figure C.2: Haar's scaling and wavelet functions

Figure C.2 illustrate the Haar scaling and wavelet functions . The Haar wavelets have limited applications for many reasons. First it does not have a good time-frquency localization. Furthermore, Harr's functions are discontinuous, which yields them insuitable as a basis classes of smoother functions.

C.2.2.2 Daubechies' wavelet

The work of Daubechies (1988) introduced a family of wavelets that are orthogonal, compactly supported and with a preassigned degree of smoothness. These wavelets are now extensively used in practice. In figure C.3 several Daubechies scaling and wavelet functions are illustrated. A Daubechies' wavelet is indexed by its vanishing moments N, which controls the smoothness of the scaling and the wavelets functions

$$\int x^p \psi(x) dx = 0, p = 0, 1, \cdots, N-1$$

and

$$\int \mid x^N \psi(x) \mid dx < \infty$$

In other words, polynomials of degree up to $N-1$ can be written exactly in terms of the appropriately translated scaling functions.
In this thesis we will use Daubechies family. Note that this family of wavelet is easy to implement and there exist several packages in R and matlab witch construct efficiently these wavelets.
To conclude this brief presentation of different wavelet bases it is important to note that there exist several other families of wavelet with different properties. For rigorous definitions and a detailed study of wavelets families the reader is referred to Daubechies (1992), Vidakovic (1999) and Ogden (1997).

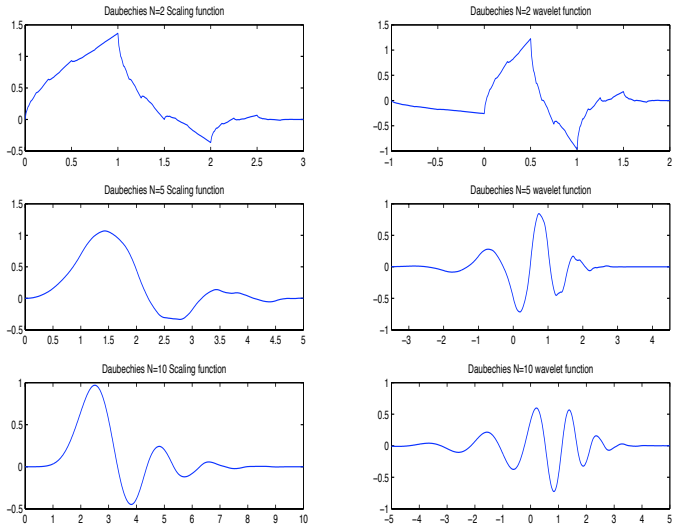

Figure C.3: Daubechies'scaling and wavelet functions for vanishing moments $N = 2, 5, 10$

C.2.3 The discrete wavelet transform

Frequently in statistics we are concerned by discretely sampled data. Thus, by analogy to the discrete Fourier transform we could replace the function f in the definition of the scaling and wavelet coefficients by an estimate corresponding to the function values at equally spaced points t_i, $i = 1, \cdots n$. We define the *discrete wavelet transform* (DWT) of $\mathbf{Y} = (f(t_1), \cdots, f(t_n))$ as

$$s_{j_0,k} \approx \frac{1}{\sqrt{n}} \sum_{i=2}^{n} y_i \phi_{j_0,k}(t_i), k = 0, \cdots, 2_0^j - 1$$

$$d_{j,k} \approx \frac{1}{\sqrt{n}} \sum_{i=2}^{n} y_i \psi_{j,k}(t_i), j = j_0, \cdots, J-1 \text{ and } k = 0, \cdots, 2^j - 1$$

The $s_{j_0,k}$ and $d_{j,k}$ are related to the continuous wavelets coefficients by the relation $\alpha_{j_0,0} \approx \frac{1}{\sqrt{n}} s_{j_0,k}$ and $\beta_{j,0} \approx \frac{1}{\sqrt{n}} d_{j,k}$.
The DWT can be written in matrix form as

$$\begin{aligned} \mathbf{d} &= W\mathbf{Y} \\ &= (s_{j_0,0} \cdots s_{j_0,2^{j_0}-1} d_{j_0,0} \cdots d_{J-1,2^{J-1}-1})^T \end{aligned}$$

where $\mathbf{d}$ is an $n \times 1$ vector including discrete scaling coefficients $s_{j_0,k}$ and discrete wavelet coefficients $d_{j,k}$, W is an orthogonal $n \times n$ matrix associated with the orthonormal wavelet

basis (the projection matrix on the space V_J) and $\mathbf{Y}$ the vector of the function values at the points t_i, $i = 1, \cdots n$. due to the orthogonality of W, the *inverse discrete wavelet transform* (IDWT) is given by

$$\mathbf{Y} = W^T \mathbf{d}$$

If for some positive integer J the number of observation n is dyadic ($n = 2^J$) the DWT and the IDWT can be performed through a computationally fast algorithm developed by Mallat (1989), which requires only order n operations to transform an n-sample vector. For detailed description of the algorithm we refer to Mallat (1989), and Vidakovic (1999).

C.3 Nonparametric regression for equispaced design

In this section we consider the problem of wavelet based univariate nonparametric regression of a function f defined on $[0, 1]$. The goal is to recover the function f from the noisy observation data $(t_i, y_i)_{i=1,\cdots,n}$

$$y_i = f(t_i) + \epsilon_i, \ i = 1, \cdots, n$$

where ϵ i.i.d $N(0, \sigma^2)$ random variables. For simplicity and without loss of generality, we assume that the sample plane is dyadic and its points are equally spaced, i.e. $t_i = 1/n$. For non-equispaced or non-dyadic designs some modifications will be needed to the standard wavelet-based regression.

C.3.1 Linear regression

Following C.5 The linear estimator of f consists in the projection of f on the space V_J

$$P^J f = \hat{f}(t) = \sum_{k=0}^{2^{j_0}-1} \hat{\alpha}_{j_0,k}\phi_{j_0,k} + \sum_{j=j_0}^{J} \sum_{k=0}^{2^j-1} \hat{\beta}_{j,k}\psi_{j,k} \tag{C.6}$$

where the empirical scaling ($\hat{\alpha}_{j_0,k}$) and empirical wavelet ($\hat{\beta}_{j,k}$) coefficients are given by

$$\hat{\alpha}_{j_0,k} = \frac{1}{n}\sum_{i=1}^{n} y_i \phi_{j_0,k}(t_i) \tag{C.7}$$

$$\hat{\beta}_{j,k} = \frac{1}{n}\sum_{i=1}^{n} y_i \psi_{j,k}(t_i) \tag{C.8}$$

The smoothness of the estimation is controlled by the parameter J. Increasing J amounts to decreasing the amounts of smoothing. Thus an appropriate choice of J is important

to perform a good estimation. The optimal choice of J should depend on regularity of the unknown function f. Thereby, too small J will oversmooth the estimation which will face difficulties in estimating functions with local singularities, although too high value of J will damage the estimate in smooth region. A more judicious choice of the empirical wavelet coefficients is reviewed in what follows.

C.3.2 Non-linear regression

C.3.2.1 Wavelet thresholding

Introduced by Donoho & Johnstone (1994), Donoho (1995), Donoho *et al.* (1995) and Donoho & Johnstone (1998) the non-linear wavelet estimator based on wavelet thresholding and shrinkage methods outperforms any linear estimator. Indeed, when the wavelet decomposition is sparse, it is reasonable to assume that only a few $\hat{\beta}_{j,k}$ contain information about the function f. An appropriate choice of the significant value of $\hat{\beta}_{j,k}$ from which we retain the coefficients (all others are set equal to 0) is fundamental to obtain a good approximation of f.

Thresholding methods allow the data to decide itself which wavelet coefficients are significant. The best known thresholding method are the hard thresholding which is a "keep or kill" rule and the soft thresholding which is a "shrink or kill" rule. They are defined respectively by

$$\delta_{\lambda}^{Hard}(\hat{\beta}_{j,k}) = \begin{cases} 0 & \text{if } \mid \hat{\beta}_{j,k} \mid \leq \lambda \\ \hat{\beta}_{j,k} & \text{if } \mid \hat{\beta}_{j,k} \mid > \lambda \end{cases} \tag{C.9}$$

and

$$\delta_{\lambda}^{Soft}(\hat{\beta}_{j,k}) = \begin{cases} 0 & \text{if } \mid \hat{\beta}_{j,k} \mid \leq \lambda \\ \hat{\beta}_{j,k} - \lambda & \text{if } \hat{\beta}_{j,k} > \lambda \\ \hat{\beta}_{j,k} + \lambda & \text{if } \hat{\beta}_{j,k} < -\lambda \end{cases} \tag{C.10}$$

It has been shown in Gao & Bruce (1997) and Marron *et al.* (1998) that hard thresholding results in larger variance in the function estimate, and due to its discontinuity, it is sensitive to small changes in the data, while the soft thresholding has larger bias.

Many others thresholding methods has been developed to compromise the trade-off between variance and bias. For example, Antoniadis & Fan (2001) suggested the SCAD thresholding defined by

$$\delta_{\lambda}^{SCAD}(\hat{\beta}_{j,k}) = \begin{cases} sign(\hat{\beta}_{j,k}) \max(0, \mid \hat{\beta}_{j,k} \mid -\lambda) & \text{if } \mid \hat{\beta}_{j,k} \mid \leq 2\lambda \\ \frac{(a-1)\hat{\beta}_{j,k} - a\lambda sign(\hat{\beta}_{j,k})}{a-2} & \text{if } 2\lambda < \mid \hat{\beta}_{j,k} \mid \leq a\lambda \\ \hat{\beta}_{j,k} & \text{if } \mid \hat{\beta}_{j,k} \mid > a\lambda \end{cases} \tag{C.11}$$

This thresholding metohds is a "shrink or kill" rule. Note that it requires two threshold values (λ and a). Based on a Bayesian argument Antoniadis & Fan (2001) have recommended to use the value of $a = 3.7$.
These three thresholding functions are displayed in figure C.4.

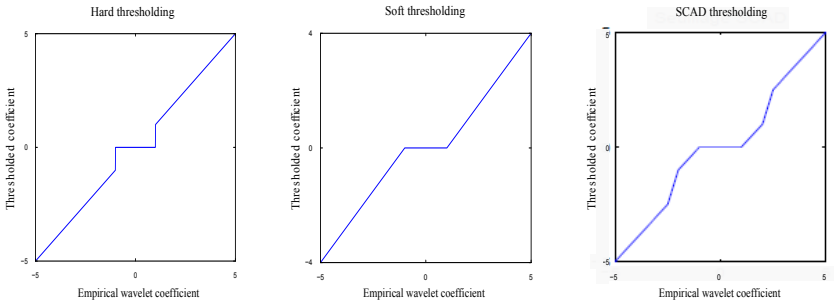

Figure C.4: Hard, Soft and SCAD thresholding for $\lambda = 1$

The effectiveness of the thresholding procedure depends on an appropriate choice of the threshold parameter λ. For too large λ the estimate might omit important parts of the function, whereas for too small λ the estimate retains noise. There are a variety of methods to choose λ, the most common in practice is the universal threshold proposed by Donoho (1995):

$$\lambda_{univ} = \sigma\sqrt{2log(n)}$$

where besides the parameter λ we will have to estimate the noise standard deviation σ. Donoho & Johnstone (1998) proposed an estimate of σ that is based only on the empirical wavelet coefficients at the finest scale associated to the space W_J. The median of absolute deviation (MAD) estimate, which is very common in practice, is defined as

$$\hat{\sigma} = \frac{median(\mid \tilde{\beta_J} - median(\tilde{\beta_J}) \mid)}{0.6745}$$

where $\tilde{\beta_J}$ is the vector of the empirical wavelet coefficients associated to the space W_J.

C.3.2.2 Penalized least-squares wavelet estimators

The thresholding methods can be seen as a regularization process under specific penalty functions. Antoniadis & Fan (2001) proposed the penalty associated to the usual thresholding and showed that the traditional regularization problem can be formulated in the

multiresolution analysis context by minimizing the penalized least-squares functional $l(\boldsymbol{\theta})$, defined as

$$l(\boldsymbol{\theta}) = \|\mathbf{WY} - \boldsymbol{\theta}\|_n^2 + 2\lambda \sum_{i=1}^{n} Pen|\theta_i| \tag{C.12}$$

where $\boldsymbol{\theta} = (\beta_{j_0,0} \cdots \beta_{J,2^J-1})^T$ is the vector of the wavelet coefficients of the unknown regression function f and Pen the penalty function. The L_1-penalty $Pen(|\theta|) = \lambda|\theta|$ corresponding to the soft thresholding rule and the penalty $Pen(|\theta|) = \lambda^2 - (|\theta| - \lambda)^2 \mathbb{1}_{|\theta|<\lambda}(|\theta|)$ corresponding to the hard thresholding rule.
As discussed before, it is clear that the performance of the wavelet estimator depends on the penalty and the regularization parameter λ. For more details we refer the reader to Antoniadis & Fan (2001), Fan & Li (2001) and Antoniadis (2007).

C.4 Note about wavelet regression for multivariate problems

In the previous section we dealt only with the univariate wavelet statistical methods, but we can also use wavelet in multivariate setting. Various constructions of a d-dimensional wavelet basis exist, the most common is the tensor product construction and the corresponding property of multiresolution C.2 is defined as

$$V_j = V_j^1 \otimes V_j^2 \cdots \otimes V_j^d \tag{C.13}$$

Here, we consider two-dimensional regression problems. The multiresolution property C.13 is equivalent to

$$V_j = V_j^1 \otimes V_j^2 = V_{j-1}^1 \otimes V_{j-1}^2 \oplus (V_{j-1}^1 \otimes W_{j-1}^2) \oplus (W_{j-1}^1 \otimes V_{j-1}^2) \oplus (W_{j-1}^1 \otimes W_{j-1}^2)$$

In the two-dimensional case, the function $f(x^{(1)}x^{(2)})$ is sampled on a grid where the number of sample points x_i is supposed to be equal to 2^n ($n = 2^J$) and equispaced. Thus, it is clear that the number of the wavelet coefficients which should be estimated growth exponentially with the dimension d. For these reason the wavelet techniques have been efficiently applied in the multidimensional case only in image processing ($d = 2$ case).
For more details about multivariate wavelet the reader is refered to Daubechies (1992), Ogden (1997),Vidakovic (1999) and Resnikoff & Wells (1998).

References

Akhiezer, N.I. & Glazman, .M. (1963). *Theory of linear operators in Hilbert space*. Ungar, New York.

Amato, U., Antoniadis, A. & Pensky, M. (2006). Wavelet kernel penalized estimation for non-equispaced design regression. *Statistics and Computing*, **16**, 37–56.

Antoniadis, A. (2007). Wavelet methods in statistics: some recent developments and their applications. *Statistics Surveys,*, **1**, 16–55.

Antoniadis, A. & Fan, J. (2001). Regularization of wavelet approximations. *Journal of American Statistical Association*, **96(455)**, 939–967.

Antoniadis, A., Gregoire, G. & Vidal, P. (1997). Random design wavelet curve smoothing. *Statistics and Probability Letters*, **35**, 225–232.

Bayarri, M.J., Berger, J.O., Cafeo, J., Garcia-Donato, G., Liu, F., Palomo, J., Parthasarathy, R.J., Paulo, R., Sacks, J. & Walsh, D. (2007). Computer model validation with functional output. *The Annals of Statistics*, **35**, 1874–1906.

Berlinet, A. & Thomas-Agnan, C. (2003). *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer Academic Publishers.

Bioucas-Dias, J.M. & Figueiredo, M.A.T. (2007). A new twist: two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image Processing*, **16**, 2992 – 3004.

Blatman, G. (2009). *Adaptive sparse polynomial chaos expansions for uncertainty propagation and sensitivity analysis*. Ph.D. thesis, Université Blaise Pascal-Clermont II.

Blatman, G. & Sudret, B. (2010). Efficient computation of global sensitivity indices using sparse polynomial chaos expansions. *Reliability Engineering and System Safety*, **95**, 1216–1229.

Breiman, L. (1995). Better subset regression using the nonnegative garrote. *Technometrics*, **37**, 373–384.

Busby, D., Farmer, C.L. & Iske, A. (2007). Hierarchical nonlinear approximation for experimental design and statistical data fitting. *SIAM J. Sci. Comput*, **29**, 49–69.

Byrne, C. (2002). Iterative oblique projection onto convex sets and the split feasibility problem. *Inverse Problems*, **18**, 441–453.

Campbell, K., McKay, M.D. & Williams, B.J. (2006). Sensitivity analysis when model outputs are functions. *Reliability Engineering and System Safety*, **91**, 1468–1472.

Cohen, A., Daubechies, I. & Vial, P. (1993). Wavelets on the interval and fast wavelet transforms. *Applied and Computational Harmonic Analysis*, **1**, 54–81.

Daubechies, I. (1988). Some results on tchebycheffian spline functions. *Communications in Pure and Applied Mathematics*, **41**, 909–996.

Daubechies, I. (1992). *Ten lectures on wavelets*. SIAM.

Daubechies, I., Defrise, M. & Mol, C.D. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. Pure. Appl. Math*, **57**, 1413–1457.

Daubechies, I., Fornasier, M. & Loris, I. (2008). Accelerated projected gradient method for linear inverse problems with sparsity constraints. *Journal of Fourier Analysis and Applications*, **14**, 764–792.

Dejean, J.P. & Blanc, G. (1999). Managing uncertainties on production predictions using integrated statistical methods. *SPE Journal*.

Donoho, D. & Johnstone, I. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, **83(3)**, 425–455.

Donoho, D.L. (1995). De-noising by soft thresholding. *IEEE Transaction on Information Theory*, **41**, 613–627.

Donoho, D.L. & Johnstone, I.M. (1998). Minimax estimation via wavelet shrinkage. *Annals of statistics*, **26**, 879–921.

Donoho, D.L., Johnstone, I.M., Kerkyacharian, G. & Picard, D. (1995). Wavelet shrinkage: asymptopia?(with discussion). *Journal of Royal Statistical Society*, **Series B 57**, 301–337.

Drignei, D. (2010). Functional anova in computer models with time series output. *Technometrics*, **52**, 430–437.

Efron, B., Hastie, T., Johnstone, I. & Tibshirani, R.J. (2004). Least angle regression. *The annals of statistics*, **32**, 407–499.

Eicke, B. (1992). Iteration methods for convexely constrained ill-posed problems in hilbert spaces. *Numer. Funct. Anal. Optim.*, **13**, 413–129.

Fan, J. & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of American Statistical Association*, **96**, 1348–1360.

Figueiredo, M.A.T. & Nowak, R.D. (2003). An em algorithm for wavelet-based image restortion. *IEEE Transactions on Image Processing*, **12**, 906–916.

Friedman, J., Hastie, T., Hfling, H. & Tibshirani, R. (2007). Pathwise coordinate optimization. *The Annals of Applied Statistics*, **1**, 302–332.

Gao, H.Y. & Bruce, A. (1997). Waveshrink with firm shrinkage. *Statistica Sinica*, **7**, 855–874.

Gunn, S. & Kandola, J. (2002). Structural modeling with sparse kernels. *Machine Learning*, **48**, 115–136.

Hoerl, A. & Kennard, R. (1970). Ridge regression: application to nonorthogonal problems. *Technometrics*, **12(1)**, 69–82.

Jung, U. & Lu, J.C. (2004). A wavelet based random effect model for multiple sets of complicated functional data. Tech. rep.

Kerkyacharian, G. & Picard, D. (2004). Regression in random design and warped wavelets. *Bernoulli*, **10**, 1053–1105.

Kimeldorf, G. & Wahba, G. (1971). Some results on tchebycheffian spline functions. *J. Math. Anal. Applic.*, **33**, 82–95.

KOVAC, A. & SILVERMAN, B.W. (2000). Extending the scope of wavelet regression methods by coefficient-depent tresholding. *Journal of American Statistical Association*, **95**, 172–183.

LADA, E.K., LU, J.C. & WILLSON, J.R. (2002). A wavelet-based procedure for process fault detection. *IEEE Transactions on Semiconductor Manufacturing*, **15**, 79–90.

LANDWEBER, L. (1951). An iterative formula for fredholm integral equations of the first kind. *American journal of mathematics*, **73**, 615–624.

LIN, Y. & ZHANG, H. (2006). Component selection and smoothing in smoothing spline analysis of variance models. *Annals of Statistics*, **34(5)**, 2272–2297.

M. DEFRISE, C.D.M. (1987). *A note on stopping rules for iterative regularization methods and filtered svd*, 261–268. P. C. Sabatier (Ed.).

MALLAT, S. (1989). A theory for multiresolution signal decomposition : wavelet representation. *IEEE Trans. Pattn. Anal. Mach. Intell.*, **11**, 674–693.

MALLOWS, C. (1973). Some comments on c_p. *Technometrics*, **15**, 661–675.

MARREL, A., IOOSS, B., VAN DORPE, F. & VOLKOVA, E. (2008). An efficient methodology for modeling complex computer codes with gaussian processes. *Computational Statistics and Data Analysis*, **52**, 4731–4744.

MARREL, A., IOOSS, B., JULLIEN, M., LAURENT, B. & VOLKOVA, E. (to appear). Global sensitivity analysis for models with spatially dependent outputs. *Environmetrics*.

MARRON, J.S., ADAK, S., JOHNSTONE, I.M., NEUMANN, M.H. & PATIL, P. (1998). Exact risk analysis of wavelet regression. *Journal of Computational and Graphical Statistics*, **7**, 278–309.

MATHERON, G. (1970). La théorie des variables régionalisées, et ses applications. *Les cahiers du centre de morphologie mathématique de Fontainebleau*, **Fascicule 5**, 764–792.

MAXIM, V. (2003). *Restauration de signaux bruité observés sur des plans d'expérience aléatoires*. Ph.D. thesis, University of Grenoble.

McKay, M.D., Beckman, R.J. & Conover, W.J. (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.

Ogden, R. (1997). *Essential wavelets for statistical applications and data analysis*. Birkhäuser.

Pepelyshev, A. (2010). *The Role of the Nugget Term in the Gaussian Process Method*, 149–156. Contributions to Statistics, Physica-Verlag HD.

Piana, M. & Bertero, M. (1997). Projected landweber method and preconditioning. *Inverse Problems*, **13**, 441–463.

Resnikoff, H.L. & Wells, R.O. (1998). *Wavelet analysis : the scalable structure of information*. springer.

Sacks, J., Welch, W.J., Mitchell, T.J. & Wynn, H.P. (1989). Design and analysis of computer experiments. *Statistical science*, **4**, 409–435.

Saltelli, A. & Sobol', I. (1995). About the use of rank transformation in sensitivity of model output. *Reliability Engineering and System Safety*, **50**, 225–239.

Saltelli, A. & Sobol', I. (1999). A quantitative, model independent method for global sensitivity analysis of model output. *Reliability Engineering and System Safety*, **41(1)**, 39–56.

Saltelli, A., Chan, K. & Scott, M. (2000). *Sensitivity analysis*. Wiley.

Santner, T.J., Williams, B.J. & Notz, W.I. (2003). *The design and analysis of computer experiments*. Springer.

Sobol, I. (1993). Sensitivity estimates for nonlinear mathematical models. *Mathematical Modelling and Computational Experiments*, **1**, 407–414.

Storlie, C.B. & Helton, J.C. (2008a). Multiple predictor smoothing methods for sensitivity analysis: Description of techniques. *Reliability engineering and system safety*, **93**, 28–54.

Storlie, C.B. & Helton, J.C. (2008b). Multiple predictor smoothing methods for sensitivity analysis: examples results. *Reliability Engineering and System Safety*, **93**, 57–77.

Storlie, C.B., Swiler, L.P., Helton, J.C. & Sallaberry, C.J. (2009). Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models. *Reliability Engineering and System Safety*, **94, n 11**, 1735–1763.

Storlie, C.B., Bondell, H.D., Reich, B.J. & Zhang, H. (2011). Surface estimation, variable selection, and the nonparametric oracle property. *Statistica Sinica*, **21**, 679–705.

Sudret, B. (2008). Global sensitivity analysis using polynomial chaos expansions. *Reliability Engineering and System Safety*, **93**, 964–979.

Tavassoli, Z., Carter, J.N. & King, P.R. (2004). Errors in history matching. *SPE Journal*, 352–361.

Tibshirani, R.J. (1996). Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society*, **Series B 58**, 267–288.

Vidakovic, B. (1999). *Statistical modeling by wavelets*. Wiley-Interscience.

Wahba, G. (1990). *Spline models for observational data*. SIAM.

Welch, W.J., Sacks, J., Wynn, H.P., Mitchell, T.J. & Morris, M.D. (1992). Screening, prediction , and computer experiments. *Technometrics*, **34**, 15–25.

Yuan, M. & Lin, Y. (2007). On the nonnegative garrote estimator. *Journal of Royal Statistical Society*, **Series B 69**, 143–161.

Yuan, M., Joseph, V.R. & ZOU, H. (2009). Structured variable selection and estimationr. *The Annals of Applied Statistics*, **3**, 1738–1757.

Zhang, Z., Li, R. & Sudjianto, A. (2007). Modeling computer experiments with multiple responses. *SAE*, paper number 2007-01-1655.

Zou, H., Hastie, T. & Tibshirani, R. (2007). On the "degree of freedom" of the lasso. *Annals of Statistics*, **35**, 2173–2192.

Résumé: L'objectif de cette thèse est l'investigation de nouvelles méthodes de surface de réponse afin de réaliser l'analyse de sensibilité de modèles numériques complexes et coûteux en temps de calcul. Pour ce faire, nous nous sommes intéressés aux méthodes basées sur la décomposition ANOVA. Nous avons proposé l'utilisation d'une méthode basée sur les splines de lissage de type ANOVA, alliant procédures d'estimation et de sélection de variables. L'étape de sélection de variable peut devenir très coûteuse en temps de calcul, particulièrement dans le cas d'un grand nombre de paramètre d'entrée. Pour cela nous avons développé un algorithme de seuillage itératif dont l'originalité réside dans sa simplicité d'implémentation et son efficacité. Nous avons ensuite proposé une méthode directe pour estimer les indices de sensibilité. En s'inspirant de cette méthode de surface de réponse, nous avons développé par la suite une méthode adaptée à l'approximation de modèles très irréguliers et discontinus, qui utilise une base d'ondelettes. Ce type de méthode a pour propriété une approche multi-résolution permettant ainsi une meilleure approximation des fonctions à forte irrégularité ou ayant des discontinuités. Enfin, nous nous sommes penchés sur le cas où les sorties du simulateur sont des séries temporelles. Pour ce faire, nous avons développé une méthodologie alliant la méthode de surface de réponse à base de spline de lissage avec une décomposition en ondelettes. Afin d'apprécier l'efficacité des méthodes proposées, des résultats sur des fonctions analytiques ainsi que sur des cas d'ingénierie de réservoir sont présentées.

Abstract: The purpose of this thesis is to investigate innovative response surface methods to address the problem of sensitivity analysis of complex and computationally demanding computer codes. To this end, we have focused our research work on methods based on ANOVA decomposition. We proposed to use a smoothing spline nonparametric regression method, which is an ANOVA based method that is performed using an iterative algorithm, combining an estimation procedure and a variable selection procedure. The latter can become computationally demanding when dealing with high dimensional problems. To deal with this, we developed a new iterative shrinkage algorithm, which is conceptually simple and efficient. Using the fact that this method is an ANOVA based method, it allows us to introduce a new method for computing sensitivity indices. Inspiring by this response surface method, we developed a new method to approximate the model for which the response involves more complex outputs. This method is based on a multiresolution analysis with wavelet decompositions, which is well known to produce very good approximations on highly nonlinear or discontinuous models. Finally we considered the problem of approximating the computer code when the outputs are times series. We proposed an original method for performing this task, combining the smoothing spline response surface method and wavelet decomposition. To assess the efficiency of the developed methods, numerical experiments on analytical functions and reservoir engineering test cases are presented.