Reliability Engineering and System Safety 112 (2013) 67–81

Contents lists available at SciVerse ScienceDirect

Reliability Engineering and System Safety

journal homepage: www.elsevier.com/locate/ress

Smoothing spline analysis of variance approach for global sensitivity analysis of computer codes

Samir Touzani*, Daniel Busby

IFP Energies nouvelles, 92852 Rueil-Malmaison, France

ARTICLE INFO

Article history:
Received 7 September 2011
Received in revised form
6 November 2012
Accepted 7 November 2012
Available online 2 December 2012

Keywords:
Global sensitivity analysis
Metamodel
Smoothing spline ANOVA
Nonparametric regression

ABSTRACT

The paper investigates a nonparametric regression method based on smoothing spline analysis of variance (ANOVA) approach to address the problem of global sensitivity analysis (GSA) of complex and computationally demanding computer codes. The two steps algorithm of this method involves an estimation procedure and a variable selection. The latter can become computationally demanding when dealing with high dimensional problems. Thus, we proposed a new algorithm based on Landweber iterations. Using the fact that the considered regression method is based on ANOVA decomposition, we introduced a new direct method for computing sensitivity indices. Numerical tests performed on several analytical examples and on an application from petroleum reservoir engineering showed that the method gives competitive results compared to a more standard Gaussian process approach.

1. Introduction

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 or oil reservoir forecasting. Simulations are performed with complex computer codes that model diverse complex real world phenomena. Inputs of such computer codes are estimated by experts or by indirect measures and can be highly uncertain. It is important to identify the most significant inputs, which contribute to the model prediction variability. This task is generally performed by the variance-based sensitivity analysis also known as global sensitivity analysis (GSA) (see [1] and [2]).

The aim of GSA for computer codes is to quantify how the variation of an output of the computer code is apportioned to different inputs of the model. The most useful methods that perform sensitivity analysis require stochastic simulation techniques, such as Monte-Carlo methods. These methods usually involve several thousands computer code evaluations that are generally not affordable with realistic models for which each simulation requires several minutes, hours or days. Consequently, meta-modeling methods become an interesting alternative.

A meta-model is an approximation of the computer code's input/output relation, which is fast to evaluate. The general idea of this approach is to perform a limited number of model evaluations (hundreds) at some carefully chosen training input values, and then, using statistical regression techniques to construct an approximation of the model. If the resulting approximation is of a good quality, the meta-model is used instead of the complex and computationally demanding computer code to perform the GSA.

The most commonly used meta-modeling methods are those based on parametric polynomial regression models, which require specifying the polynomial form of the regression mean (linear, quadratic, etc.). 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.

In the last decade many different nonparametric regression models have been used as a meta-modeling method. To name a few of them [3–5] utilized a Gaussian Process (GP). [6,7] used a polynomial chaos expansions to perform a GSA.

In addition [8–10] 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 strategies.

We focus in this work on the modern nonparametric regression method based on smoothing spline ANOVA (SS-ANOVA) model and component selection and smoothing operator (COSSO) regularization, which can be seen as an extension of the LASSO

* Corresponding author.
E-mail addresses: samirtouzani.phd@gmail.com (S. Touzani), daniel.busby@ifpen.fr (D. Busby).

0951-8320/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ress.2012.11.008

[11] variable selection method in parametric models to nonparametric models. Moreover, we use the ANOVA decomposition basis of the COSSO to introduce a direct method to compute the sensitivity indices.

In this paper, we first review the SS-ANOVA, 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 a modified least angle regression algorithm (LARS) (see [12,13]). Next we will describe our new method to compute the sensitivity indices. Finally, numerical simulations and an application from petroleum reservoir engineering will be presented and discussed.

2. Smoothing spline ANOVA approach for metamodels

In mathematical terms, the computer code can be represented as a function $Y=f(\mathbf{X})$ where Y is the output scalar of the computer code, $\mathbf{X}=(X^{(1)},\ldots,X^{(d)})$ the d-dimensional inputs vector which represent the uncertain parameters and $f:\mathbb{R}^d\to\mathbb{R}$ is the unknown function that represents the computer code. Our purpose is to introduce 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 [14]. To remind, the ANOVA expansion is defined as

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

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.

It is important to determine which ANOVA components should be included in the model. Lin and Zhang [15] proposed a penalized least square method with the penalty functional being the sum of component norms. The COSSO is a regularized nonparametric regression method based on ANOVA decomposition.

In the following subsections, we first review the definition of the COSSO method. Then we present the COSSO algorithm proposed in [15]. Finally we introduce three new variations of the COSSO algorithm.

2.1. Definition

Let $f\in\mathcal{F}$, where $\mathcal{F}$ is a reproducing kernel Hilbert space (RKHS) (for more details we refer to [14,16]) corresponding to the ANOVA decomposition (1), and let $\mathcal{H}^j=\{1\}\oplus\overline{\mathcal{H}}^j$ be a RKHS of functions of $X^{(j)}$ over $[0,1]$, where $\{1\}$ is the RKHS consisting of only the constant functions and $\overline{\mathcal{H}}^j$ 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 $\mathcal{H}^j$

$$\mathcal{F}=\bigotimes_{j=1}^{d}\mathcal{H}^j=\{1\}\oplus\sum_{j=1}^{d}\overline{\mathcal{H}}^j\oplus\sum_{j<l}[\overline{\mathcal{H}}^j\otimes\overline{\mathcal{H}}^l]\ldots \tag{2}$$

Each component in the ANOVA decomposition (1) is associated to a corresponding subspace in the orthogonal decomposition (2). We assume that 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 us 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 $d(d+1)/2$ corresponds to the number of ANOVA components) with $1\le j<l\le d$. Using such notation in (2) we obtain

$$\mathcal{F}=\{1\}\oplus\bigoplus_{\alpha=1}^{q}\mathcal{F}^{\alpha}=\{1\}\oplus\sum_{j=1}^{d}\overline{\mathcal{H}}^j\oplus\sum_{j<l}[\overline{\mathcal{H}}^j\otimes\overline{\mathcal{H}}^l] \tag{3}$$

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 COSSO estimate is given by the minimizer of

$$\frac{1}{n}\sum_{i=1}^{n}(y_i-f(\mathbf{x}_i))^2+\lambda^2\sum_{\alpha=1}^{q}\|P_\alpha f\| \tag{4}$$

where λ is the regularization parameter and P_α is the orthogonal projection onto $\mathcal{F}^\alpha$.

2.2. COSSO algorithm

Lin and Zhang [15] have shown that the minimizer of (4) 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 $\mathcal{F}^\alpha$, $\widehat{f}_\alpha\in\mathrm{span}\{K_\alpha(x_i,\cdot),\ i=1,\ldots,n\}$, where K_α is the reproducing kernel of $\mathcal{F}^\alpha$ defined by

$$K_\alpha(x,x')=K_j(x,x')=k_1(x)k_1(x')+k_2(x)k_2(x')-k_4(|x-x'|)$$

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

$$k_1(x)=x-\frac{1}{2}$$

$$k_2(x)=\frac{1}{2}(k_1^2(x)-1/12)$$

$$k_4(x)=\frac{1}{24}\left(k_1^4(x)-\frac{k_1^2(x)}{2}+\frac{7}{240}\right)$$

Moreover, the reproducing kernel K_α for the RKHS $\mathcal{F}^\alpha$ such as $\mathcal{F}^\alpha\equiv\overline{\mathcal{H}}^j\otimes\overline{\mathcal{H}}^l$, 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 [14].

Lin and Zhang [15] have also shown that (4) is equivalent to a more easier form to compute, which is

$$\frac{1}{n}\sum_{i=1}^{n}\{y_i-f(\mathbf{x}_i)\}^2+\lambda_0\sum_{\alpha=1}^{q}\theta_\alpha^{-1}\|P_\alpha f\|^2+v\sum_{\alpha=1}^{q}\theta_\alpha \text{ subject to } \theta_\alpha\ge 0 \tag{5}$$

where λ_0 is a constant and v is a smoothing parameter. If $\theta_\alpha=0$, then the minimizer of (5) is taken to satisfy $\|P_\alpha f\|=0$ and we use the convention $0/0=0$. The penalty term of θ's, $\sum_{\alpha=1}^{q}\theta_\alpha$, controls the sparsity of each component f_α.

For fixed $\boldsymbol{\theta}=(\theta_1,\ldots,\theta_q)^T$ (5) is equivalent to the standard SS-ANOVA [14] and therefore the solution has the form

$$f(\mathbf{x})=b+\sum_{i=1}^{n}c_i\sum_{\alpha=1}^{q}\theta_\alpha K_\alpha(\mathbf{x}_i,\mathbf{x}) \tag{6}$$

and 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 stand for the matrix $\sum_{\alpha=1}^{q}\theta_\alpha K_\alpha$ and $\mathbf{1}_n$ the column vector consisting of n ones. Then $f=K_\theta\mathbf{c}+b\mathbf{1}_n$ with $\mathbf{c}=(c_1,\ldots,c_n)^T$ and (5) can be expressed as

$$\frac{1}{n}\left\|\mathbf{Y}-\sum_{\alpha=1}^{q}\theta_\alpha K_\alpha\mathbf{c}-b\mathbf{1}_n\right\|^2+\lambda_0\mathbf{c}^TK_\theta\mathbf{c}+v\sum_{\alpha=1}^{q}\theta_\alpha \tag{7}$$

For a fixed $\boldsymbol{\theta}$, (7) can be written as

$$\min_{\mathbf{c},b}\|\mathbf{Y}-K_\theta\mathbf{c}-b\mathbf{1}_n\|^2+n\lambda_0\mathbf{c}^TK_\theta\mathbf{c} \tag{8}$$

which is a smoothing spline problem (a quadratic minimization problem) and the solution satisfies

$$(K_\theta + n\lambda_0 I)\mathbf{c} + b\mathbf{1}_n = \mathbf{Y} \tag{9}$$

$$\mathbf{1}_n^T \mathbf{c} = 0 \tag{10}$$

where I is the identity matrix.

Alternately, for fixed $\mathbf{c}$ and b, (7) can be written as

$$\min_\theta \|\mathbf{z} - D\boldsymbol{\theta}\|^2 + n\nu \sum_{\alpha=1}^{q} \theta_\alpha \quad \text{subject to } \theta_\alpha \geq 0 \tag{11}$$

where $\mathbf{z} = \mathbf{Y} - (1/2)n\lambda_0\mathbf{c} - b\mathbf{1}_n$ and D the $n \times q$ matrix with the α th column $\mathbf{d}_\alpha = K_\alpha \mathbf{c}$. Note that this formulation is similar to the nonnegative garrote (NNG) optimization problem introduced in [17].

An equivalent form of (11) is given by

$$\min_\theta \|\mathbf{z} - D\boldsymbol{\theta}\|^2 \quad \text{subject to } \theta_\alpha \geq 0 \quad \text{and} \quad \sum_{\alpha=1}^{q} \theta_\alpha \leq M \tag{12}$$

for some $M \geq 0$. Lin and Zhang [15] noted that the optimal M seems to be close to the number of important components. For computational consideration Lin and Zhang [15] preferred to use (12) rather than (11).

Notice that the COSSO algorithm is a two step procedure. Indeed, it iterates between the standard smoothing spline (8) estimator, which gives a good initial estimate and the NNG (12) estimator, which is a variable selection procedure.

They also observed empirically that after one iteration the result is close to that at convergence. For convenience, in what follow, we call COSSO-solver the algorithm introduced in [15]. Thus the COSSO-solver algorithm is presented as a one step update procedure

Algorithm 1 (*COSSO-solver*).

1. Initialization: Fix $\theta_\alpha = 1$, $\alpha = 1, \ldots, q$
2. For each fixed λ_0 in a chosen range solve for $\mathbf{c}$ and b with (8). Tune λ_0 using v-fold-cross-validation. Set $\mathbf{c}_0$ and b_0 the solution of (8) for the best value of λ_0. Fix λ_0 at the chosen value.
3. For each fixed M in a chosen range, apply the following procedure:
   - 3.1 Solve for $\mathbf{c}$ and b with (8).
   - 3.2 For $\mathbf{c}$ and b obtained in step 3.1, solve for $\boldsymbol{\theta}$ with (12).
   - 3.3 For $\boldsymbol{\theta}$ obtained in step 3.2, solve for $\mathbf{c}$ and b with (8). Tune M using v-fold-cross-validation. Fix M at the chosen value.
4. For λ_0 from steps 2, M from steps 3 and with $\mathbf{c}_0$ and b_0 obtained in step 1, solve for $\boldsymbol{\theta}$ with (12). The solution corresponds to the final estimate of $\boldsymbol{\theta}$.
5. With the new $\boldsymbol{\theta}$ from steps 4 and for each fixed λ_0 in a chosen range solve for $\mathbf{c}$ and b with (8).Tune λ_0 using v-fold-cross-validation. The solution corresponding to the new best value of λ_0 is the final estimate of $\mathbf{c}$ and b.

Notice that we have added the tuning procedure of λ_0 in step 5 with respect to the original COSSO algorithm [15] because we observed empirically that it improved the method's performance, particularly for the high dimensional problems.

2.3. COSSO based on the iterative projected shrinkage algorithm

Consider the (11) regression problem

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

The functional (11) is convex since the matrix D^{TD} is symmetric and positive semidefinite and since the constraints $\theta_\alpha > 0$ also define 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 $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \|z - D\boldsymbol{\theta}\|^2 + n\nu \sum_{\alpha=1}^{q} \theta_\alpha$ subject to $\theta_\alpha \geq 0$. These KKT conditions are defined as

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

which is equivalent to

$$-\mathbf{d}_\alpha^T(\mathbf{Y} - D\boldsymbol{\theta}^*) + \nu = 0 \quad \text{if } \theta_\alpha^* \neq 0 \tag{13}$$

$$-\mathbf{d}_\alpha^T(\mathbf{Y} - D\boldsymbol{\theta}^*) + \nu > 0 \quad \text{if } \theta_\alpha^* = 0 \tag{14}$$

where $\mathbf{d}_\alpha$ denotes the α th column of D. Therefore, from (13) and (14) we can derive the fixed-point equation

$$\boldsymbol{\theta}^* = \mathcal{P}_{\Omega^+}(\delta_\nu^{Soft}(\boldsymbol{\theta}^* + D^T(\mathbf{Y} - D\boldsymbol{\theta}^*))) \tag{15}$$

where $\mathcal{P}_{\Omega^+}$ is the nearest point projection operator onto the nonnegative orthant (closed convex set) $\Omega^+ = \{x : x \geq 0\}$, and δ_λ^{Soft} is the soft-thresholding function defined as

$$\delta_\nu^{Soft}(x) = \begin{cases} 0 & \text{if } |x| \leq \nu \\ x - \nu & \text{if } x > \nu \\ x + \nu & \text{if } x < -\nu \end{cases} \tag{16}$$

Thus, in the framework of Landweber algorithm [18] we introduced in [19] the iterative projected shrinkage (IPS) algorithm to solve (11). This algorithm is defined by

$$\boldsymbol{\theta}^{[p+1]} = \mathcal{P}_{\Omega^+}(\delta_\nu^{Soft}(\boldsymbol{\theta}^{[p]} + D^T(\mathbf{Y} - D\boldsymbol{\theta}^{[p]}))) \tag{17}$$

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

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

where the positive constant c ensures that $\lambda_{max}(D^T D) \leq 1$. 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 various Landweber algorithms, among them [20–22]. The latter brought an efficient procedure, named two-step iterative shrinkage/thresholding (TwIST). We adapted this algorithm in order to solve the NNG optimization problem (11). The accelerated projected iterative shrinkage (AIPS) algorithm is defined as

$$\boldsymbol{\theta}^{[1]} = \mathcal{P}_{\Omega^+}(\delta_\nu^{Soft}(\boldsymbol{\theta}^{[0]})) \tag{18}$$

$$\boldsymbol{\theta}^{[p+1]} = (1-\alpha)\boldsymbol{\theta}^{[p-1]} + (\alpha - \beta)\boldsymbol{\theta}^{[p]} + \beta\mathcal{P}_{\Omega^+}(\delta_\nu^{Soft}(\boldsymbol{\theta}^{[p]} + D^T(\mathbf{Y} - D\boldsymbol{\theta}^{[p]}))) \tag{19}$$

In accordance with Theorem 4 given by [22] 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 $\zeta = \lambda_{min}(D^T D)$ (where λ_{min} is the minimal eigenvalue) if $\lambda_{min}(D^T D) \neq 0$, else $\zeta = 10^{-\kappa}$ with $\kappa = 1, \ldots, 4$ needs to be tuned by running a few iterations. The condition $\kappa = 1$ corresponds 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 [22].

COSSO using IPS or AIPS. Thereby, the COSSO algorithm using IPS or AIPS, which we call respectively COSSO–IPS and COSSO–AIPS, will iterate between (8) and (11) instead of iterating between (8) and (12). Thus the COSSO–IPS algorithm (respectively COSSO–AIPS algorithm) is defined as

Algorithm 2 (*COSSO–IPS/COSSO–AIPS*).

1. Initialization: Fix $\theta_\alpha = 1$, $\alpha = 1,\ldots,q$
2. For each fixed λ_0 in a chosen range solve for $\mathbf{c}$ and b with (8). Tune λ_0 using v-fold-cross-validation. Set $\mathbf{c}_0$ and b_0 the solution of (8) for the best value of λ_0. Fix λ_0 at the chosen value.
3. For each fixed v in a chosen range, apply the following procedure:
   3.1 Solve for $\mathbf{c}$ and b with (8).
   3.2 For $\mathbf{c}$ and b obtained in step 3.1, solve for $\boldsymbol{\theta}$ with (11) by using the IPS (respectively AIPS) algorithm.
   3.3 For $\boldsymbol{\theta}$ obtained in step 3.2, solve for $\mathbf{c}$ and b with (8). Tune v using v-fold-cross-validation. Fix v at the chosen value.
4. For λ_0 from steps 2, M from steps 3 and with $\mathbf{c}_0$ and b_0 obtained in step 1, solve for $\boldsymbol{\theta}$ with (11)by using the IPS (respectively AIPS) algorithm. The solution corresponds to the final estimate of $\boldsymbol{\theta}$.
5. With the new $\boldsymbol{\theta}$ from steps 4 and for each fixed λ_0 in a chosen range solve for $\mathbf{c}$ and b with (8). Tune λ_0 using v-fold-cross-validation. The solution corresponding to the new best value of λ_0 is the final estimate of $\mathbf{c}$ and b

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

2.4. COSSO based on nonnegative LARS algorithm

Consider (11) the NNG regression problem, [13] provided an efficient algorithm similar to LARS for computing the entire path solution as v is varied. We called this algorithm the nonnegative LARS (NN-LARS) and it is described below

Algorithm 3 (*NN-LARS*).

1. Start from $k=1$, $\theta_1^{[0]},\ldots,\theta_q^{[0]} = 0$, $\mathcal{A}_k = \emptyset$ and the residual $\mathbf{r}^{[0]}$ equal to the vector $\mathbf{z}$
2. Update the active set
   $\mathcal{A}_k = \mathcal{A}_{k-1} \cup \{j^*\}$ with $j^* = \arg\max_{j \in \mathcal{A}_k^c}(\mathbf{d}_j^T \mathbf{r}^{[k-1]})$
   where $\mathcal{A}_k^c$ is the complementary of $\mathcal{A}_k$.
3. Compute the current descent direction vectors
   $\mathbf{w}_{\mathcal{A}_k}^{[k]} = (D_{\mathcal{A}_k}^T D_{\mathcal{A}_k})^{-1} D_{\mathcal{A}_k}^T \mathbf{r}^{[k]}$
4. Now, for every $l \in \mathcal{A}_k^c$ compute γ_l that satisfies
   $\mathbf{d}_l^T \mathbf{r}^{[k+1]} = \mathbf{d}_j^T \mathbf{r}^{[k+1]}$. Hence
   $\mathbf{d}_l^T \mathbf{r}^{[k]} - \gamma_l \mathbf{d}_l^T D_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} = \mathbf{d}_j^T \mathbf{r}^{[k]} - \gamma_l \mathbf{d}_j^T D_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}$, with $l \in \mathcal{A}_k^c$ and $j \in \mathcal{A}_k$
   It follows
   $\gamma_l = \frac{\mathbf{d}_j^T \mathbf{r}^{[k]} - \mathbf{d}_l^T \mathbf{r}^{[k]}}{\mathbf{d}_l^T D_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]} - \mathbf{d}_j^T D_{\mathcal{A}_k} \mathbf{w}_{\mathcal{A}_k}^{[k]}}$ with $l \in \mathcal{A}_k^c$ and $j \in \mathcal{A}_k$
5. For every $j \in \mathcal{A}_k$, compute $\gamma_j = \min(\alpha_j, 1)$ where $\alpha_j = -\theta_j^{[k]} / w_j^{[k]}$
6. 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
   $\boldsymbol{\theta}_{\mathcal{A}_k}^{[k+1]} = \boldsymbol{\theta}_{\mathcal{A}_k}^{[k]} + \gamma \mathbf{w}_{\mathcal{A}_k}^{[k]}$
   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 $\mathcal{A}_{k+1} = \mathcal{A}_k - \{j^*\}$.
7. Set $\mathbf{r}^{[k+1]} = \mathbf{z} - D\boldsymbol{\theta}^{[k+1]}$, $k = k+1$ and continue until $\gamma = 1$.

COSSO using NN-LARS. Thus the COSSO algorithm using the NN-LARS algorithm, which we call COSSO–NN-LARS is define as follow

Algorithm 4 (*COSSO–NN-LARS*).

1. Initialization: Fix $\theta_\alpha = 1$, $\alpha = 1,\ldots,q$
2. For each fixed λ_0 in a chosen range solve for $\mathbf{c}$ and b with (8). Tune λ_0 using v-fold-cross-validation. Set $\mathbf{c}_0$ and b_0 the solution of (8) for the best value of λ_0. Fix λ_0 at the chosen value.
3. Apply the following procedure:
   3.1 Solve for $\mathbf{c}$ and b with (8).
   3.2 For $\mathbf{c}$ and b obtained in step 3.1, solve for $\boldsymbol{\theta}$ with (11) by using the NN-LARS algorithm.
   3.3 For each $\boldsymbol{\theta}$ (corresponding to each step k of NN-LARS algorithm) obtained in step 3.2, solve for $\mathbf{c}$ and b with (8). Choose the iteration step k of NN-LARS algorithm which correspond to the solution that minimizes the v-fold-cross-validation criterion. Set K the best chosen step.
4. For λ_0 from steps 2, K from steps 3 and with $\mathbf{c}_0$ and b_0 obtained in step 1, solve for $\boldsymbol{\theta}$ with (11) by using the NN-LARS algorithm. The solution corresponding to the step K is the final estimate of $\boldsymbol{\theta}$.
5. With the new $\boldsymbol{\theta}$ from steps 4 and for each fixed λ_0 in a chosen range solve for $\mathbf{c}$ and b with (8). Tune λ_0 using v-fold-cross-validation. The solution corresponding to the new best value of λ_0 is the final estimate of $\mathbf{c}$ and b.

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. Global sensitivity analysis

3.1. Variance based Sobol' indices

In order to describe this concept, let us suppose that the mathematical model of the computational code is defined on the unit d-dimensional cube ($\mathbf{X} \in [0,1]^d$). The main idea from Sobol' approach [1] is to decompose the response $Y = f(\mathbf{X})$ into summands of different dimensions via ANOVA decomposition (1). The integrals of every summand of this decomposition over any of its own variables are assumed to be equal to zero, i.e.

$$\int_0^1 f_{j_1,\ldots,j_s}(X^{(j_1)},\ldots,X^{(j_s)})\, dX^{(j_k)} = 0 \tag{20}$$

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

$$\int_{\Omega^d} f_{i_1,\ldots,i_s} f_{j_1,\ldots,j_l}\, d\mathbf{X} = 0 \tag{21}$$

Using the orthogonality, Sobol' [1] showed that such decomposition of $f(\mathbf{X})$ is unique and that all the terms in (1) can be evaluated

via multidimensional integrals

$$f_0 = E(Y) \tag{22}$$

$$f_j(X^{(j)}) = E(Y|X^{(j)}) - E(Y) \tag{23}$$

$$f_{j,l}(X^{(j)},X^{(l)}) = E(Y|X^{(j)},X^{(l)}) - f_j - f_l - E(Y) \tag{24}$$

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 (1) over $[0,1]^d$, and by (21), we obtain

$$V = \sum_{j=1}^{d} V_j + \sum_{1 \le j < l \le d} V_{jl} + \cdots + V_{1,2,\ldots,d} \tag{25}$$

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

$$V = E(Y^2) - f_0^2 \tag{26}$$

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

$$S_{j_1,\ldots,j_s} = \frac{V_{j_1,\ldots,j_s}}{V} \tag{27}$$

where $1 \le j_1 < \cdots < j_s \le d$ and $s = 1,\ldots,d$. Thus, $S_j = V_j/V$ 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 $S_{jl} = V_{jl}/V$, for $j \ne l$, 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 (25) has the useful property that all sensitivity indices sum up to one.

$$\sum_{j=1}^{p} S_j + \sum_{1 \le j < l \le p} S_{jl} + \ldots + S_{1,2,\ldots,p} = 1 \tag{28}$$

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.

$$S_{T_j} = \sum_{l \# j} S_l \tag{29}$$

where $\#j$ represents all the $S_{j_1,\ldots,j_s}$ terms that include the index j. Total sensitivity index of an input $X^{(j)}$ measures 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 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 [1]). 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 require huge random samples. For this reason GSA techniques are prohibitive if used directly using the computer code (fluid flow simulator for example).

3.2. Global sensitivity analysis using COSSO

It has been shown in the previous section 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 (25) and the COSSO solution form (6) we have

$$V \approx \sum_{j=1}^{d} V_j + \sum_{1 \le j < l \le d} V_{jl} \tag{30}$$

$$V \approx \sum_{\alpha=1}^{q} E\left(\left[\theta_\alpha \sum_{i=1}^{n} c_i K_\alpha(\mathbf{x}_i,\mathbf{X})\right]^2\right) \tag{31}$$

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

$$\widehat{V}_j = \frac{1}{N}\sum_{m=1}^{N}\left[\theta_j \sum_{i=1}^{n} c_i K_j(x_{i_j},z_{m_j})\right]^2 \tag{32}$$

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

$$\widehat{S}_j = \frac{\widehat{V}_j}{\widehat{V}} \tag{33}$$

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

$$\widehat{V}_{jl} = \frac{1}{N}\sum_{m=1}^{N}\left[\theta_{jl} \sum_{i=1}^{n} c_i K_j(x_{i_j},z_{m_j}) K_l(x_{i_l},z_{m_l})\right]^2 \tag{34}$$

Thus, the second order indices are estimated by

$$\widehat{S}_{jl} = \frac{\widehat{V}_{jl}}{\widehat{V}} \tag{35}$$

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

$$\widehat{S}_{T_j} = \widehat{S}_j + \sum_{l \ne j} \widehat{S}_{jl} \tag{36}$$

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

4. Simulations

The present section is focused on studying the empirical performance of the four different versions of COSSO estimate and compares them to the GP method. In this work we used a standard implementation of the GP (see a brief description in appendix). However, it is important to note that the quality of the GP model is strongly dependent on the covariance function and hyperparameters estimation, so results can differ from an implementation to another. Moreover, more flexible GP method such as [23] may be more effective on the studied examples. The different versions 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 [15]). 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

$$Q_2 = 1 - \frac{\sum_{i=1}^{n_{test}}(y_i - \widehat{f}(\mathbf{x}_i))^2}{\sum_{i=1}^{n_{test}}(y_i - \overline{y})^2} \quad \text{with } n_{test} = 1000 \tag{37}$$

where y_i denotes the ith test observation of the test set, $\overline{y}$ is their empirical mean and $\widehat{f}(\mathbf{x}_i)$ is the predicted value at the design point

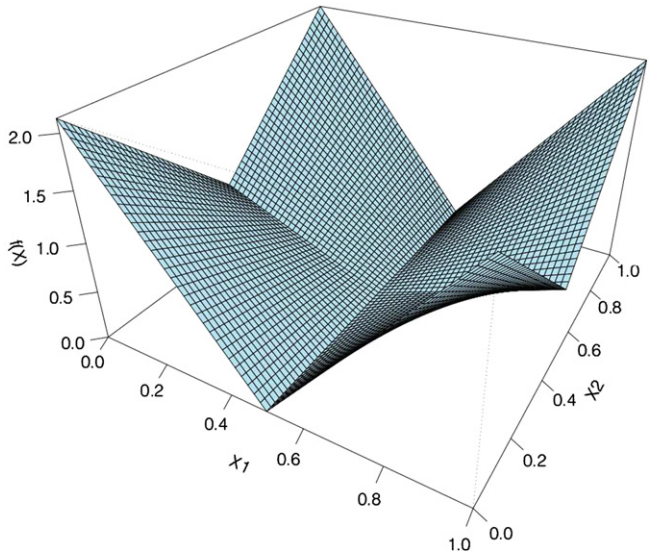

Fig. 1. Plot of g-Sobol function vs. inputs $X^{(1)}$ and $X^{(2)}$ with other inputs fixed at 0.5.

$\mathbf{x}_i$. We also compare the methods for different experimental design sizes, uniformly distributed on $[0,1]^d$ and built by Latin Hypercube Design procedure [24] with maximin criterion [25] (maximinLHD). For each setting of each test example, we run 50 times and average. Thus we define the quantity $\overline{Q}_2 = 1/50 \sum_{k=1}^{50} Q_2^k$.

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 estimation results of the sensitivity indices obtained by the developed method and using COSSO–AIPS to those obtained by Sobol' method (described in [1,2]) for sensitivity indices estimation using COSSO–AIPS and using GP meta-modelling procedure. The main purpose of our comparison to the GP is the fact that GP is widely used in the framework of meta-modelling.

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 code used here is a commercial version implemented using R in the CougarFlow ^{TM} software [26].

4.1. 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. Fig. 1 illustrates the g-Sobol function against the two most influential parameters $X^{(1)}$ and $X^{(2)}$. The g-Sobol function (see [2]) is defined for 8 input factors as

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

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

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

The analytical values of Sobol' indices are given by

$$V_j = \frac{1}{3(1+a_j)^2}, \qquad V = \prod_{k=1}^{d}(V_k+1)-1, \qquad S_{j_1,\ldots,j_s} = \frac{1}{V}\prod_{k=1}^{s} V_k$$

where $1 \le j_1 < \cdots < j_s \le d$ and $s = 1,\ldots,d$. The analytical values of the total effect indices are shown in Table 1.

Table 1
Analytical values of the total effect indices of the g-Sobol function.

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 2
Q_2 results from the g-Sobol function. The estimated standard deviation of Q_2 is given in parentheses.

	n	$\overline{Q}_2$	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

4.1.1. Assessment of the prediction accuracy

Table 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 $\overline{Q}_2$ quantity equal to 0.99 that indicates that those meta-models explain 99% of the model variance. All the COSSO versions provide quite similar results for this example. Moreover, as expected, the AIPS method is clearly faster than IPS. Notice that

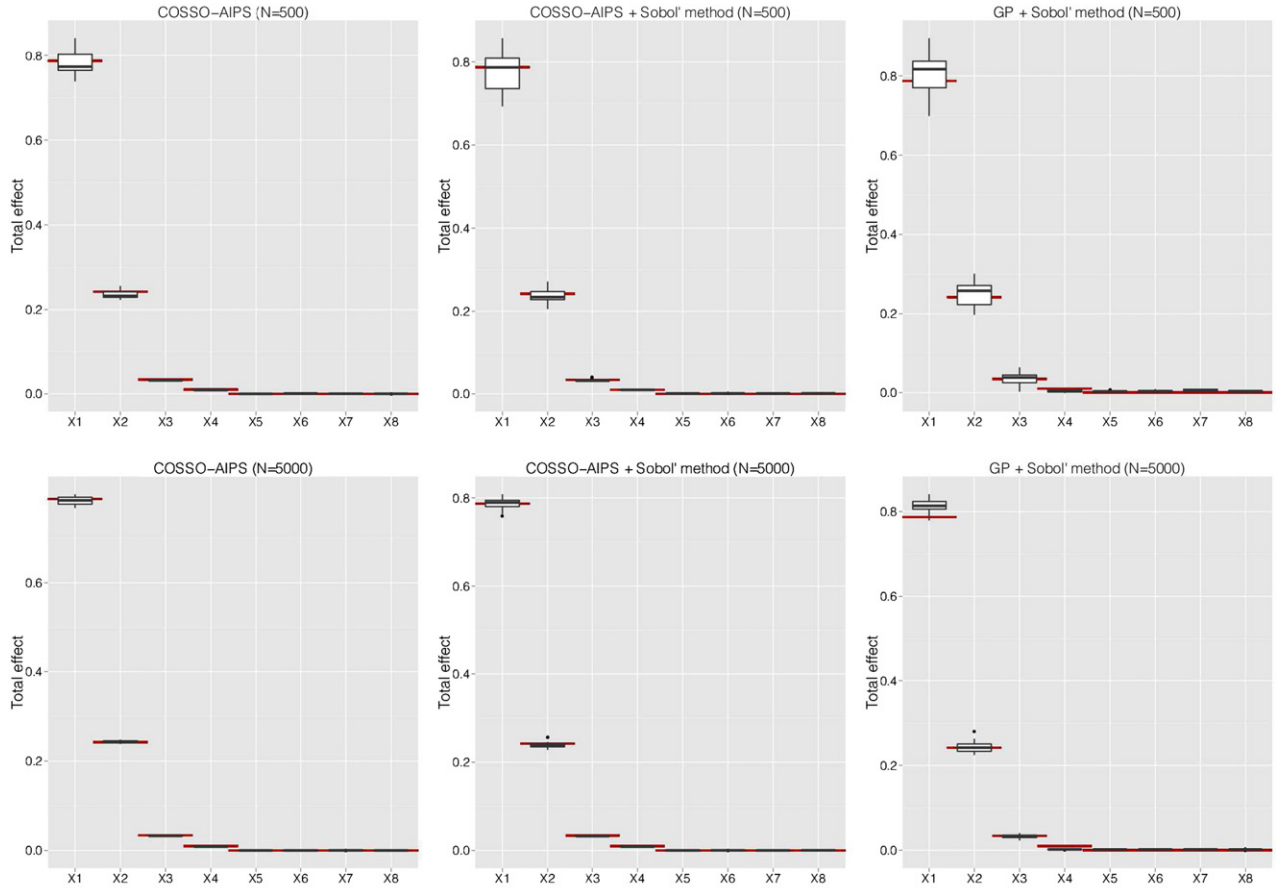

Fig. 2. Total effect indices vs. sample effect (Example 1). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

even if the NN-LARS provides the entire path of the solution, the COSSO–NN-LARS method has the same computational cost as COSSO–AIPS and COSSO-solver, because it uses the best model chosen by v-fold-cross-validation which is computationally costly. Note that the model discontinuities on the first derivative in this example could benefit from using other regression method which can handle discontinuities.

4.1.2. 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 methods was motivated by the good performance of the method and its fast execution on the studied examples in this paper. 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 meta-model 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' method of indices estimation based on the same COSSO–AIPS model and a meta-model build by GP on the same experimental design, the Q_2 for this meta-model is equal to 0.96. As introduced previously Sobol' method to estimate the total effect needs 2 samples, thus we build, using a maximinLHD procedure, 200 samples of two sizes: $N=500$ and $N=5000$. Fig. 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 $\widehat{S}_{T_j}$, $j=1,\ldots,8$. Red 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' method using the COSSO–AIPS and GP meta-models, especially when the sample size is small ($N=500$). Moreover our method needs only N evaluation of the COSSO–AIPS meta-model while Sobol' method needs $2Nd$ evaluations of GP meta-model (for $N=5000$, 80 000 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 $n=400$). Fig. 3 summarizes the results. Each panel is a boxplot of the 50 estimations of $\widehat{S}_{T_j}$, $j=1,\ldots,8$. Red lines are drawn at the corresponding analytical values of the total effects indices. As expected the estimations based on GP meta-models 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.

4.2. Example 2

Let us consider the same example that has been used in [15]. This 10 dimensional regression problem is defined as

$$f(\mathbf{X}) = g_1(X^{(1)}) + g_2(X^{(2)}) + g_3(X^{(3)}) + g_4(X^{(4)}) + g_1(X^{(3)} + X^{(4)}) + g_2\left(\frac{X^{(1)}X^{(3)}}{2}\right) + g_3(X^{(1)}X^{(2)}) \tag{38}$$

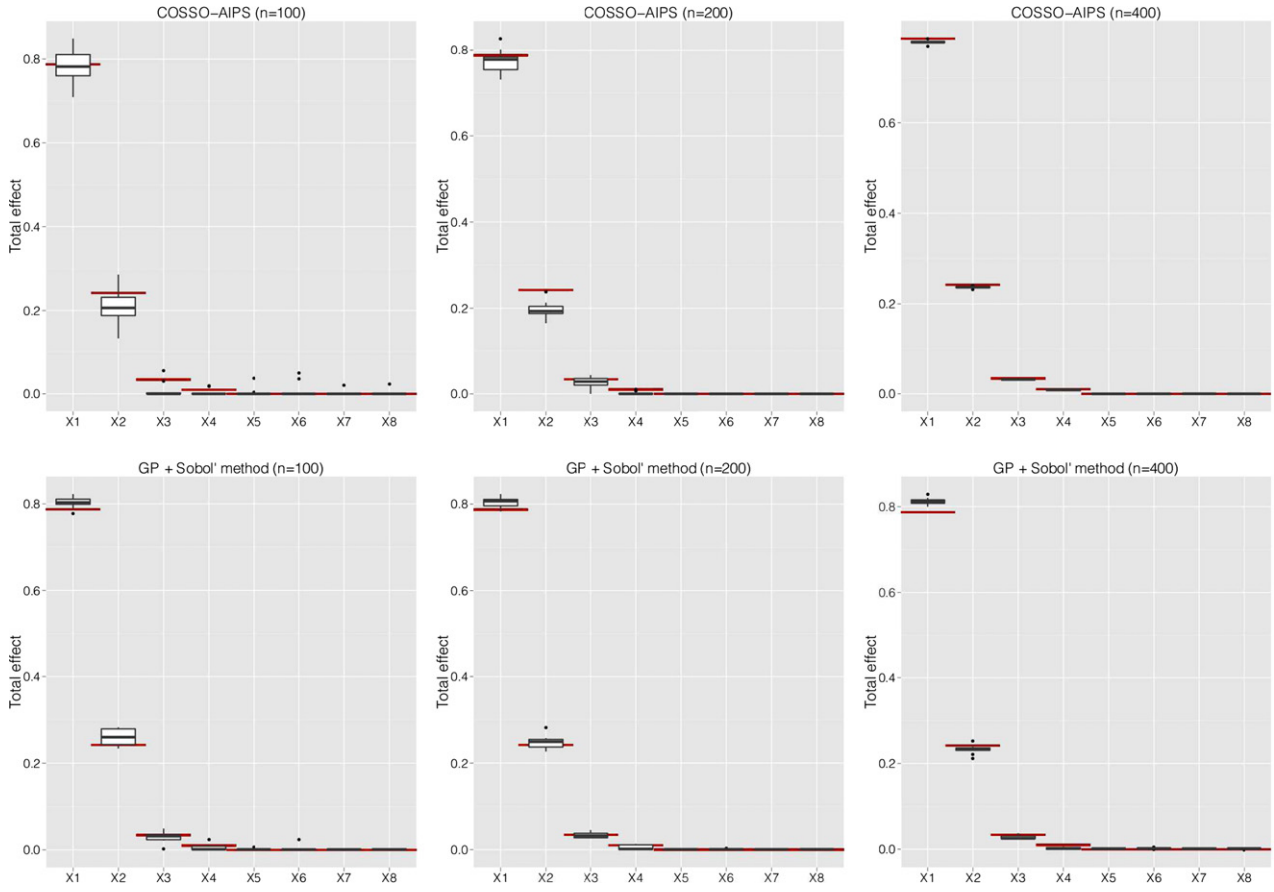

Fig. 3. Total effect indices vs. experimental design size effect (Example 1). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 3
95% CI and the reference values of the total effect indices for Example 2.

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

where

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

Therefore $X^{(5)},\ldots,X^{(8)}$ 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 of the indices are computed by direct Monte-Carlo simulation using Sobol' method (with $N=250\,000$, which correspond to 5×10^6 evaluations). Table 3 shows 95% confidence intervals (95% CI) provided by 100 different samples and the chosen reference values

4.2.1. Assessment of the prediction accuracy

Table 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.

Table 4
Q_2 results from Example 2. The estimated standard deviation of Q_2 is given in parentheses.

	n	$\overline{Q}_2$	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

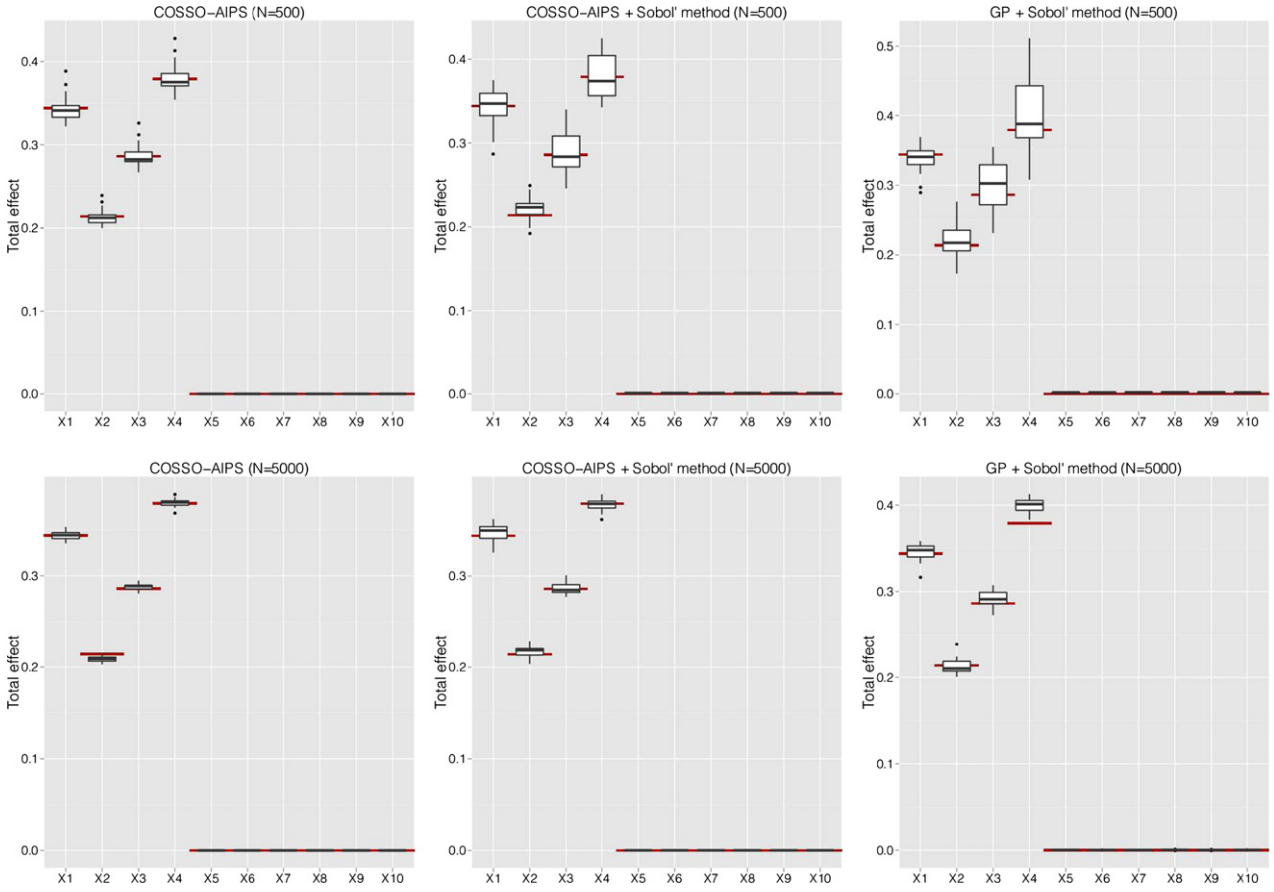

Fig. 4. Total effect indices vs. sample effect (Example 2). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.2.2. 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 by the developed method as well as by Sobol' method using a meta-model 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' method using GP meta-model on the same experimental design, the Q_2 of this meta-model is equal to 0.95. We then build 200 samples of two sizes: $N=500$ and $N=5000$ using a maximinLHD procedure.

Fig. 4 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 effect indices $\widehat{S}_{T_j}$, $j=1,\ldots,10$. Red 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 is more robust than Sobol' one using the meta-models based on GP and COSSO–AIPS 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 Fig. 5. Each panel is a boxplot of the 50 estimations of $\widehat{S}_{T_j}$, $j=1,\ldots,10$. Red 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 performs better than Sobol' method with GP.

4.3. Example 3

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

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

where the functions g_1, g_2, g_3 and g_4 are the same as for example 2. Notice that $X^{(13)},\ldots,X^{(20)}$ are uninformative. The reference values of the total effect indices are computed by direct Monte-Carlo simulation using Sobol' method (with $N=250\,000$, which corresponds to 5×10^6 evaluations). Table 5 shows 95% confidence intervals (95% CI) provided by 100 different samples and the chosen reference values.

4.3.1. Assessment of the prediction accuracy

Table 6 summarizes the results for the 50 realizations of the example 3 model with two different experimental design sizes ($n=200$ and $n=400$) built using maximinLHD procedure. For this example we choose to do not test COSSO–IPS since we shown in the previous tests that AIPS have better computational performance. It can be seen that for this model GP has a bad performance for both sizes of the experimental design. We suspect that the optimization method used to estimate the GP covariance hyperparameters gets stucked in local minima and do not provide an optimal solution. Concerning the COSSO methods

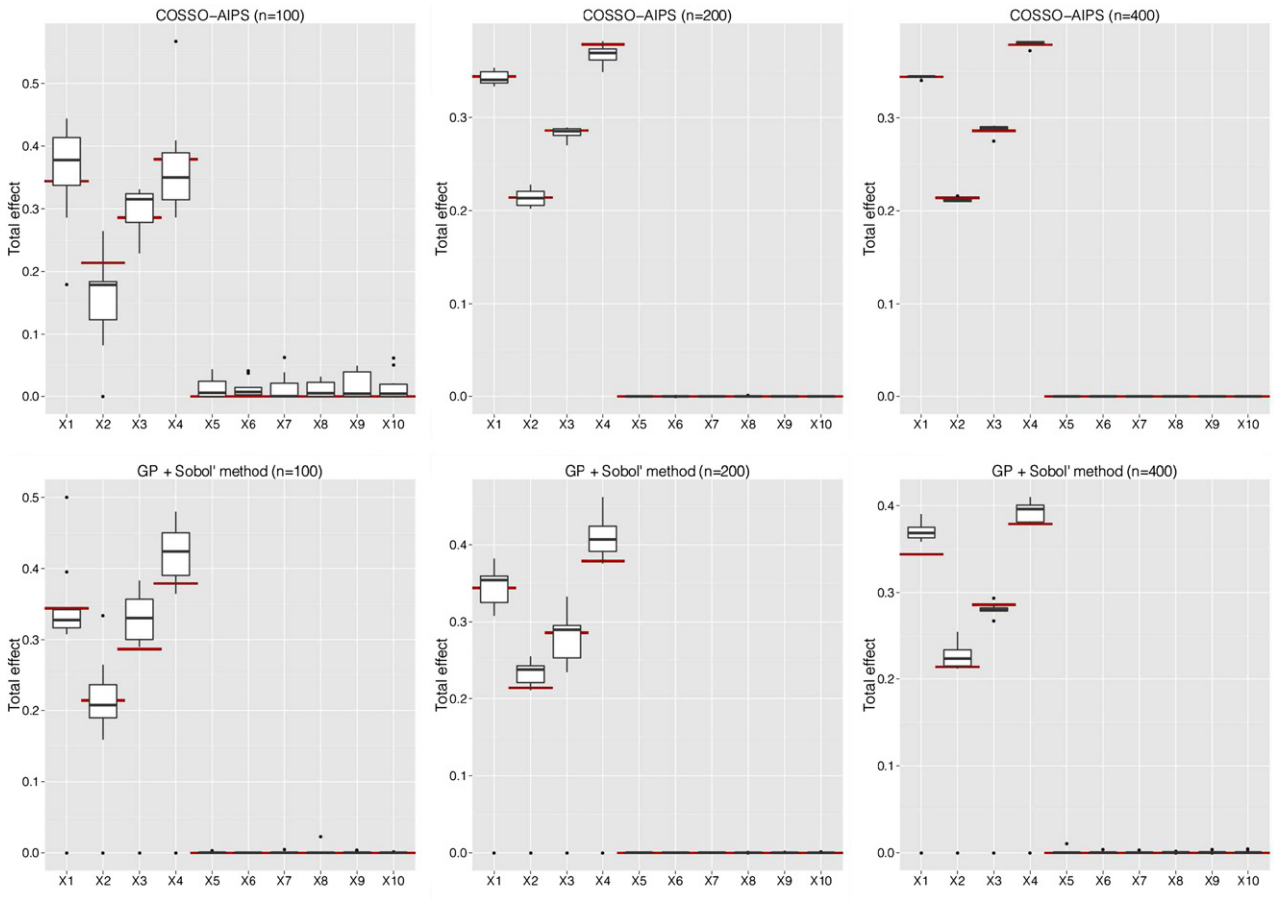

Fig. 5. Total effect indices vs. experimental design size effect (Example 2). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 5
95% CI and the reference values of the total effect indices for Example 3.

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

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.

4.3.2. 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' method with GP meta-model, because of its bad prediction performance (see Table 6). As previously, we first study the effect of the sample size N on indices estimations. We build 100 samples of two sizes: $N=500$ and $N=5000$ with maximinLHD procedure and we compute the indices using our direct method and Sobol' method with a predictive COSSO–AIPS meta-model ($Q_2=0.98$) using on an experimental design of size $n=400$. We can see in Fig. 6 that those estimates are close to the reference values of the indices and that robustness of estimations increases by increasing N. Note that with $N=500$ estimations are still quite good. In addition, as for the previous examples the direct method outperforms the Sobol' method.

Fig. 7 summarizes the results using meta-models built with two different sizes of experimental design ($n=200$ and $n=400$). As one would expect the accuracy of the indices estimations

Table 6
Q_2 results from Example 3. The estimated standard deviation of Q_2 is given in parentheses.

	n	$\overline{Q}_2$	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

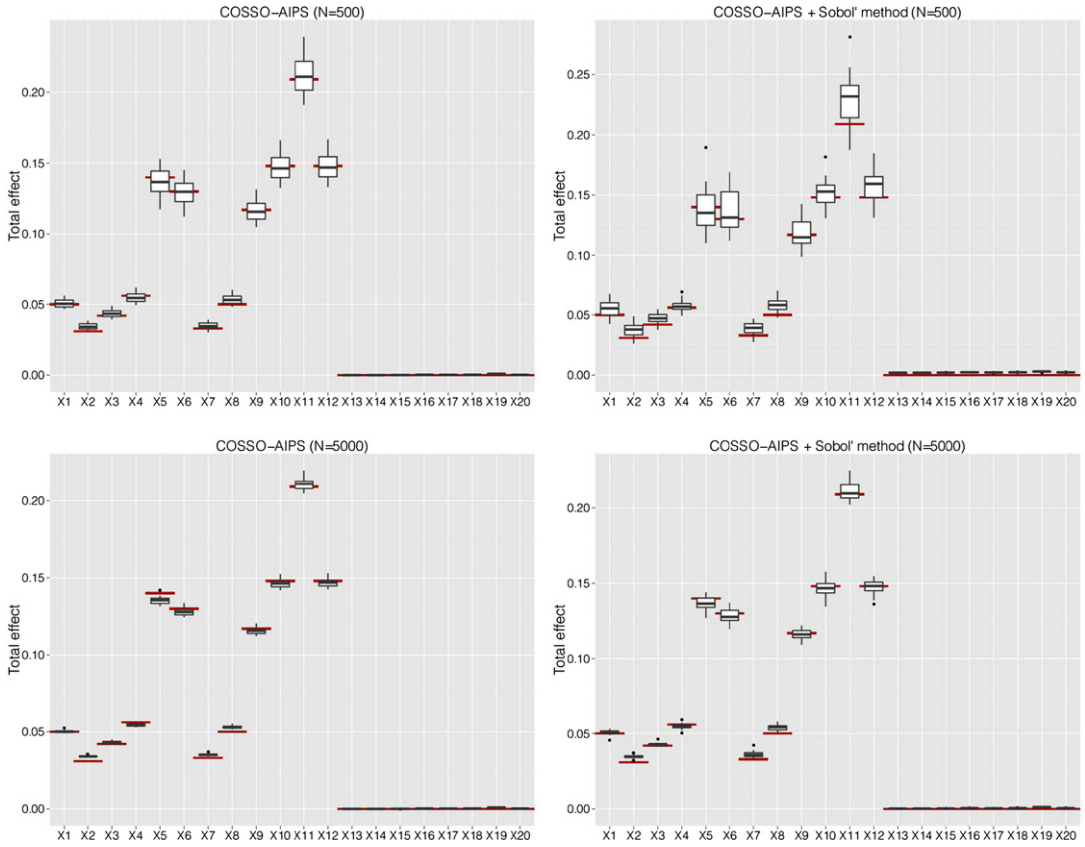

Fig. 6. Total effect indices vs. sample effect (Example 3).

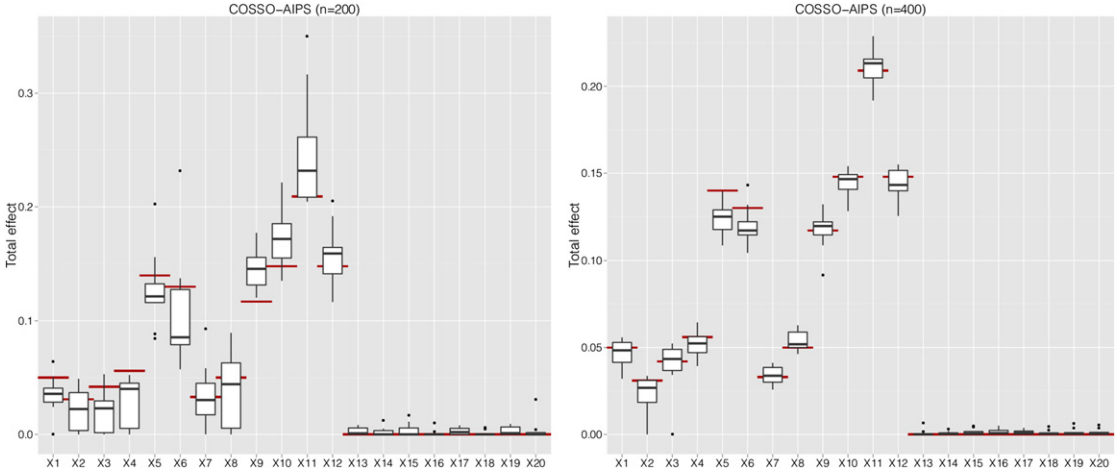

Fig. 7. Total effect indices vs. experimental design size effect (Example 3).

improves as the experimental design increases (in other words as the predictivity improves). This study was done using the 50 meta-models used in the previous section using a $N=5000$ sample to compute the total effect indices.

5. Petroleum reservoir test cases

5.1. PUNQS test case

5.1.1. 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 (Fig. 8). 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

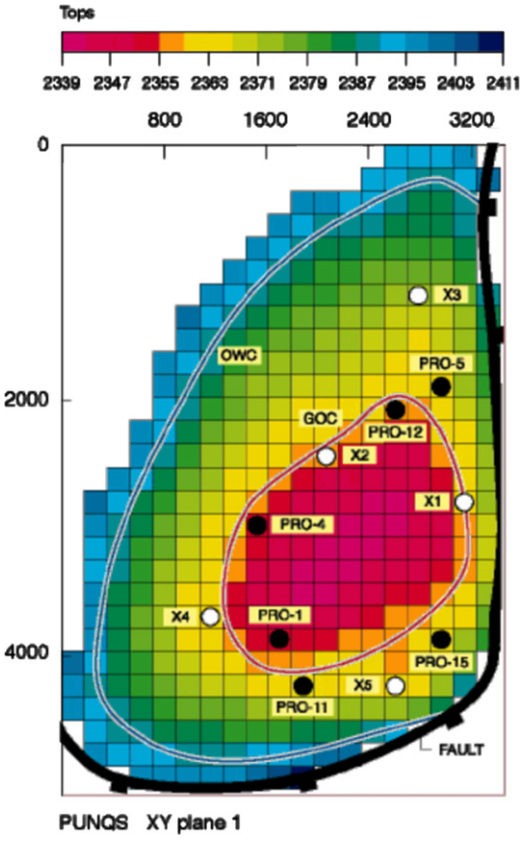

Fig. 8. Top structure map of the reservoir field (PUNQS test case).

required. For more detailed description on the model see [27]. 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)
- *PermAqui*1 $U[100;200]$ mD: analytical permeability of the aquifer 1
- *PermAqui*2 $U[100;200]$ mD: analytical permeability of the aquifer 2
- *PoroAqui*1 $U[0.2;0.3]$: analytical porosity of the aquifer 1
- *PoroAqui*2 $U[0.2;0.3]$: 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

$$OF(X) = \frac{(f(X)-\mathbf{d})^T C_D^{-1}(f(X)-\mathbf{d})}{2} \tag{39}$$

where C_D is the covariance matrix of the observed data and d the observed data. The objective function (39) represents the mismatch between observed and simulated data. This data consists of 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. The observed data is synthetically generated using a random value for the uncertain parameters in the simulator and adding, to each time step of the results, random Gaussian noise, with mean zero and standard deviation of 10% of the average value of the corresponding time series. 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.

Table 7
PUNQS model Q_2 results.

	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

5.1.2. 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 meta-models using COSSO–AIPS, COSSO-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 7 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.

5.1.3. Global sensitivity analysis

Here we use COSSO–AIPS and GP to produce meta-models 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. Table 8 shows the computed indices: we can see that the main effect and the interactions of *MPH*5 explain more than 65% of the model variance, then we have a group of five inputs (*SWCR*, *MPH*1, *SOGCR*, *SGCR* and *PermAqui*1) 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.

5.2. IC fault model

5.2.1. Reservoir model description

The geological model consists of six layers of alternating good and poor quality sands (see Fig. 9). The three good quality layers have identical properties, and three poor quality layers have different set of identical properties. The thickness of the layers

Table 8
GSA from PUNQS model.

Input	Total effect	Main effect
GP		
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		
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

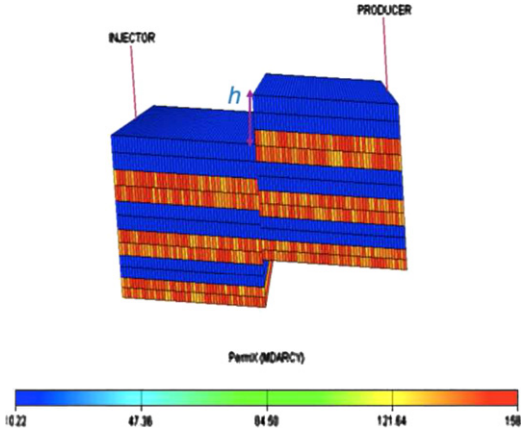

Fig. 9. IC fault model.

has arithmetic progression, with the top layer having a thickness of 12.5 feet, the bottom layer 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 pressure.

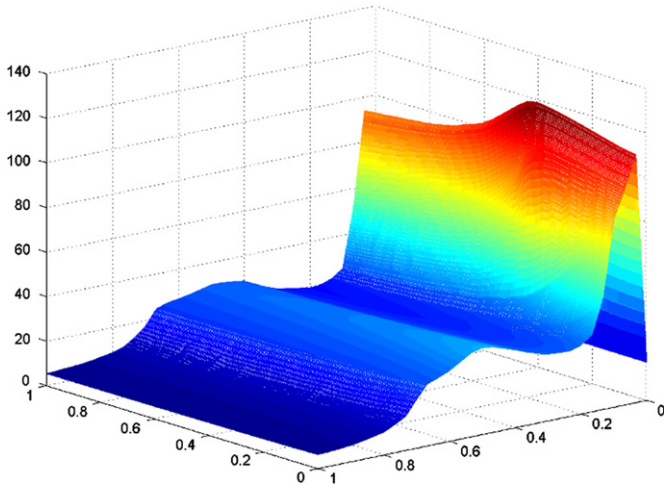

Fig. 10. Oil production rate after 10 years vs. k_g and k_p at a fixed high value of h (obtained with 1000 simulations).

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 8325–8385 feet.

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 analyzed output is in this test case the oil production rate Q_{op} at 10 years. Fig. 10 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 [28].

5.2.2. 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 meta-models using COSSO–AIPS and GP. In order to estimate Q_2, the simulator was run at an additional sample set of size $n_{test} = 25\,000$. Table 9 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. This can be probably explained by the fact that the stationarity hypothesis made here in the GP is not valid [29]. 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 strong discontinuities in the model.

5.2.3. Global sensitivity analysis

As for PUNQS test case we use COSSO–AIPS and GP to produce meta-models which are built using the experimental design of

Table 9
IC fault model Q_2 results.

	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 10
GSA from IC fault model.

Input	Total effect	Main effect
GP		
h	0.381	0.100
k_g	0.173	0.021
k_p	0.809	0.596
COSSO–AIPS		
h	0.375	0.225
k_g	0.030	0.011
k_p	0.733	0.586

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. Table 10 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.

6. Conclusion

In this work, we presented the COSSO regularized nonparametric regression method, which is a model fitting and variable selection procedure. One of the COSSO 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 large size of experimental design (large number of observations). A new iterative algorithm was developed, so-called IPS with its accelerated version (AIPS). Based on the Landweber iterations these procedures are conceptually simple and easy to implement.

We also applied the NN-LARS algorithm to COSSO that has also 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 competitive 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.

It may be possible to improve the performance of COSSO–AIPS by using an adaptive weight in the COSSO penalty [30] which may allows for more flexibility to estimate influential functional components and in the same time providing heavier penalty to non-influential functional components.

Acknowledgments

The authors are grateful to Professor Anestis Antoniadis for many useful suggestions and helpful discussions. We also thank the anonymous referees for their useful comments and constructive suggestions, which helped improve a previous version of the paper.

Appendix A. Gaussian process

In the underlying statistical model of GP (also known as kriging) [31], the deterministic response, say, $f:\mathbb{R}^d\to\mathbb{R}$, is considered as a realization of a random function. Therefore, the simulator f is treated as a realization of a stochastic process, whose form is assumed as

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

The regression part in this stochastic model (A.1) is a linear combination of preselected real-valued functions $h_1,\ldots,h_k$, with coefficients $\boldsymbol{\beta}=(\beta_1,\ldots,\beta_k)^T\in\mathbb{R}^k$, where h_1 is a constant. Moreover, Z in (A.1) is assumed to be a Gaussian random process with mean zero and covariance

$$\mathrm{cov}[\mathbf{x},\mathbf{y}]=E[Z(\mathbf{x})Z(\mathbf{y})]=\sigma^2R(\mathbf{x},\mathbf{y}) \tag{A.2}$$

between $Z(\mathbf{x})$ and $Z(\mathbf{y})$, where σ^2 denotes the process variance, $R(\mathbf{x},\mathbf{y})$ is a specific correlation function, and the symbol E denotes the usual statistical expectation. The selection of the functions $h_2,\ldots,h_k$ of the regression part in the stochastic model (A.1) relies on prior knowledge concerning the model response. For instance, in this work we only used h_1 which is a constant term. The correlation function, R, was selected following the common approach in [32–34], where it is suggested to use a correlation function of the form

$$R(\mathbf{x},\mathbf{y})=r(\mathbf{x}-\mathbf{y})=\exp\left(-\sum_{j=1}^{d}\theta_j|x_j-y_j|^{p_j}\right) \tag{A.3}$$

with $\theta_j>0$, $1\le j\le d$, and $0<p_j\le 2$.

Moreover, by following the recommendations in [32], the unknown coefficients $\boldsymbol{\theta}=(\theta_1,\ldots,\theta_d)\in\mathbb{R}^d$ and p in (A.3) are determined by maximum likelihood estimation (MLE). Likewise, the parameter σ for the process variance in (A.2) and $\boldsymbol{\beta}\in\mathbb{R}^k$ are determined by MLE, see [32] for details. The required numerical optimization relies on the algorithm L-BFGS-B, a constrained version of the quasi-Newton method, due to Byrd et al. [35]. We apply the L-BFGS-B algorithm with different random choices for the initial point.

References

[1] Sobol I. Sensitivity estimates for nonlinear mathematical models. Mathematical Modelling and Computational Experiments 1993;1:407–14.
[2] Saltelli A, Chan K, Scott M. Sensitivity analysis. Wiley; 2000.
[3] Sacks J, Welch WJ, Mitchell TJ, Wynn HP. Design and analysis of computer experiments. Statistical Science 1989;4:409–35.
[4] Busby D, Farmer CL, Iske A. Hierarchical nonlinear approximation for experimental design and statistical data fitting. SIAM Journal on Scientific Computing 2007;29(1):49–69.
[5] Marrel A, Ioss B, Dorpe FV, Volkova E. An efficient methodology for modeling complex computer codes with Gaussian processes. Computational Statistics and Data Analysis 2008;52:4731–44.

[6] Sudret B. Global sensitivity analysis using polynomial chaos expansions. Reliability Engineering and System Safety 2008;93:964–79.
[7] Blatman G, Sudret B. Efficient computation of global sensitivity indices using sparse polynomial chaos expansions. Reliability Engineering and System Safety 2010;95:1216–29.
[8] Storlie CB, Helton JC. Multiple predictor smoothing methods for sensitivity analysis: description of techniques. Reliability Engineering and System Safety 2008;93:28–54.
[9] Storlie CB, Helton JC. Multiple predictor smoothing methods for sensitivity analysis: examples results. Reliability Engineering and System Safety 2008;93:57–77.
[10] Storlie CB, Swiler LP, Helton JC, Sallaberry CJ. Implementation and evaluation of nonparametric regression procedures for sensitivity analysis of computationally demanding models. Reliability Engineering and Systems Safety 2009;94(11):1735–63.
[11] Tibshirani RJ. Regression shrinkage and selection via the lasso. Journal of Royal Statistical Society Series B 1996;58:267–88.
[12] Efron B, Hastie T, Johnstone I, Tibshirani RJ. Least angle regression. The Annals of Statistics 2004;32:407–99.
[13] Yuan M, Lin Y. On the nonnegative garrote estimator. Journal of Royal Statistical Society Series B 2007;69:143–61.
[14] Wahba G. Spline models for observational data. SIAM; 1990.
[15] Lin Y, Zhang H. Component selection and smoothing in smoothing spline analysis of variance models. Annals of Statistics 2006;34(5):2272–97.
[16] Berlinet A, Thomas-Agnan C. Reproducing kernel Hilbert spaces in probability and statistics. Kluwer Academic Publishers; 2003.
[17] Breiman L. Better subset regression using the nonnegative garrote. Technometrics 1995;37:373–84.
[18] Landweber L. An iterative formula for Fredholm integral equations of the first kind. American Journal of Mathematics 1951;73:615–24.
[19] Touzani S. Response surface methods based on analysis of variance expansion for sensitivity analysis. Unpublished doctoral thesis. Joseph Fourier University. URL ⟨http://tel.archives-ouvertes.fr/tel-00614038⟩.
[20] Piana M, Bertero M. Projected Landweber method and preconditioning. Inverse Problems 1997;13:441–63.
[21] Daubechies I, Fornasier M, Loris I. Accelerated projected gradient method for linear inverse problems with sparsity constraints. Journal of Fourier Analysis and Applications 2008;14(5–6):764–92.
[22] Bioucas-Dias JM, Figueiredo MAT. A new twist: two-step iterative shrinkage/thresholding algorithms for image restoration. IEEE Transactions on Image Processing 2007;16:2992–3004.
[23] Gramacy RB, Lee HKH. Bayesian treed Gaussian process models with an application to computer modeling. Journal of the American Statistical Association 2008;103(483):1119–30.
[24] McKay MD, Beckman RJ, Conover WJ. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 1979;21:239–45.
[25] Santner TJ, Williams BJ, Notz WI. The design and analysis of computer experiments. Springer; 2003.
[26] CougarFlow, IFP energies nouvelles uncertainty analysis software. URL ⟨http://www.openflowsuite.com/software/cougarflow⟩.
[27] PUNQS, Production forecasting with uncertainty quantification. URL ⟨http://www.fault-analysis-group.ucd.ie/Projects/PUNQ.html ⟩.
[28] Tavassoli Z, Carter JN, King PR. Errors in history matching. SPE Journal 2004:352–61.
[29] Bastos LS, O'Hagan A. Diagnostics for Gaussian process emulators. Technometrics 2009;51(4):425–38.
[30] Storlie CB, Bondell HD, Reich BJ, Zhang H. Surface estimation, variable selection, and the nonparametric oracle property. Statistica Sinica 2011;21:679–705.
[31] Matheron G. Principles of geostatistics. Economic Geology 1963;58:1246–66.
[32] Welch WJ, Buck RJ, Sacks J, Wynn HP, Mitchell TJ, Morris MD. Screening, predicting, and computer experiments. Technometrics 1992;34(1):15–25.
[33] Oakley JE, O'Hagan A. Probabilistic sensitivity analysis of complex models: a Bayesian approach. Journal of the Royal Statistical Society: Series B 2004;66:751–69.
[34] Busby D. Hierarchical adaptive experimental design for Gaussian process emulators. Reliability Engineering and System Safety 2008;94:1183–93.
[35] Byrd RH, Lu P, Nocedal J, Zhu C. A limited memory algorithm for bound constrained optimization. SIAM Journal on Scientific Computing 1995;16(5):1190–1208.