

# Rapport de stage PHIMECA

Kristof Attila Simady  
e-mail: kristof.simady@sigma-clermont.fr

05/03/2020

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Recherche bibliographique</b>	<b>2</b>
2.1	Travaux retenus . . . . .	3
2.2	Méthodes et outils de travail . . . . .	3
2.3	Prémices . . . . .	3
2.3.1	Mission confiée et tâches associées . . . . .	3
2.3.2	Prise en main d'openTURNS et de la théorie sur les champs stochastiques . . . . .	3
2.4	Exemple théorique simple, premiers codes . . . . .	6
2.5	Choix des codes et difficultés algorithmiques . . . . .	8
2.6	Premiers essais d'analyse de la sensibilité du modèle . . . . .	10
2.6.1	Interprétation et essais avec le papier : "Time-variant global reliability sensitivity analysis of structures with both input random variables and stochastic processes" . . .	11
2.7	Application au modèle NASTRAN d'échangeur thermique . . . .	19
2.8	Lists . . . . .	22
<b>3</b>	<b>Results</b>	<b>22</b>
<b>4</b>	<b>Tables and figures</b>	<b>23</b>
<b>5</b>	<b>Discussion</b>	<b>23</b>
<b>6</b>	<b>Conclusions</b>	<b>23</b>

## Avant propos

Le cadre d'étude offert par SIGMA Clermont a donné la possibilité d'effectuer une année supplémentaire de stage pour étoffer nos connaissances scientifiques, et découvrir le monde industriel comme des cultures étrangères. De par mon appétence pour la recherche et l'informatique, j'ai eu la chance de pouvoir venir effectuer mon stage au sein de l'entreprise **PHIMECA**, située à Clermont-Ferrand, et depuis longtemps en partenariat avec SIGMA.

Ce stage aura été l'occasion pour moi ... (à suivre)

Je souhaite remercier ... (à suivre)

**Note:** Ce document est autant un rapport de stage qu'une note à qui voudrait se servir des différentes méthodes explorées et codes développés.

## 1 Introduction

Dans le cadre d'analyse probabiliste et fiabiliste, il est d'usage d'effectuer des analyses de sensibilité sur une ou plusieurs grandeurs d'intérêt, grâce à divers moyens comme les indices de Sobol' ou les analyses de corrélation.

Ces analyses sont néanmoins souvent cantonnées à des modèles n'ayant en sortie et entrée que des grandeurs scalaires, pour lesquelles de nombreuses méthodes existent dans la littérature.

L'objectif du travail présenté ici, est d'effectuer ce type d'analyse sur des modèles ayant en entrée plusieurs champs aléatoires et des variables aléatoires scalaires et en sortie aussi des champs et grandeurs scalaires.

Pour compléter cela, il était bien sûr aussi demandé de d'abord créer un modèle simple gouverné par des champs stochastiques et de faire de l'analyse de sensibilité sur ce modèle.

Il sera d'abord présenté différentes méthodes présentes dans la littérature, avec leurs potentiels avantages et défauts.

Cette approche des l'analyse de sensibilité nous renvoie aussi sur des champs de recherche bien différents, car les méthodes peuvent autant venir du champ de la mécanique que de la géologie ou l'étude environnementale.

## 2 Recherche bibliographique

Ce travail de recherche a été l'occasion de parcourir un grand nombre de travaux, liés de plus ou moins loin à l'analyse de sensibilité sur des processus gaussiens. On notera notamment les travaux de [Lilburne and Tarantola, 2009] qui donnent dans leur travail un état de l'art complet sur les différentes méthodes utilisables pour l'analyse de sensibilité sur champs gaussiens.

D'autres techniques plus récentes ont néanmoins vu le jour, et ce sont ces dernières que nous avons explorés en premier. Notamment les travaux de [Wei et al., 2017] sur l'analyse de sensibilité de structures ayant en entrée des variables aléatoires et processus stochastiques variant dans le temps, ou encore le travail de [Pronzato, 2019] qui développe un métamodèle entre l'entrée et sortie d'un modèle gouverné par des champs stochastiques.

## 2.1 Travaux retenus

Etant donné que les missions d'analyse de sensibilité et d'incertitudes se font classiquement en se basant sur la bibliothèque *openTURNS* développé par PHIMECA, le choix a été fait d'adapter techniques de recherche et méthodes déjà existantes dans l'API au problème étudié.

## 2.2 Methodes et outils de travail

L'ensemble des codes ont été écrits dans le langage *Python*, en se servant majoritairement des bibliothèques à caractère scientifique (*SciPy*, *NumPy*) et la bibliothèque développée en partie par **PHIMECA** : *openTURNS* (*open treatment of Uncertainties and Risk & Statistics*) [Michaël Baudin and Popelin, 2015]. **PHIMECA** a fourni un poste de travail Linux, tout comme un accès au serveur de calcul pour les modèles un peu plus conséquents.

## 2.3 Analyse de la problématique, premier tests et prise en main des outils

### 2.3.1 Mission confiée et tâches associées

L'objectif du stage était de développer une méthodologie pour l'analyse de sensibilité sur des modèles gouvernés par des champs aléatoires (Champs Stochastiques) et des variables aléatoires, et qui en sortie sont aussi représentables par une collection de champs et variables aléatoires.

Plus spécifiquement, cette méthodologie était à appliquer à un modèle d'échangeur de chaleur air-air commercialisé par **LIEBHERR** et utilisé pour la régulation thermique de l'air ambiant dans les avions de ligne.

Comme la mission confiée au sein de **PHIMECA** se rapprochait d'un travail de recherche, celui-ci était bien sur accompagné de nombreuses étapes distinctes. En effet, en plus du développement des codes et de l'application analytique, une recherche bibliographique a du être menée en amont pour parvenir à lister les différentes méthodes d'analyse existantes et tester celles qui pourraient convenir au mieux. De plus, une phase d'apprentissage et de tests a du d'abord être nécessaire pour parvenir à prendre en main les différents outils, et comprendre la théorie et les mathématiques sous-jacentes.

### 2.3.2 Prise en main d'openTURNS et de la théorie sur les champs stochastiques

#### Champs Stochastiques

Un champ stochastique est un champ de variables aléatoires toutes corrélées entre elles, et dont l'intensité de la corrélation est déterminée par leur proximité dans l'espace. Par exemple la position de tout objet est corrélée à sa position aux intervalles de temps proches (continuité du temps), ou bien les températures au dessus d'un pays sont corrélés à faible distance mais l'effet aléatoire est plus présent lorsque les distances sont importantes.

Ce type de modélisation des champs est couramment utilisée pour modéliser des

propriétés présentant des variabilités gaussiennes mais une continuité dans leur espace de définition.

Lorsque la corrélation est seulement dépendante de la distance entre deux points de l'espace et non d'une position absolue, on parle d'un champ stationnaire. Lors de cette étude, on se place dans ce cas.



Figure 1: Réalisation d'un champ stochastique en deux dimension. Le champ est continu dans l'espace et la primitive est une gaussienne en deux dimensions.

Mathématiquement, cette corrélation peut être définie par différents modèles. Les deux modèles de corrélation les plus simples sont les fonctions de covariance exponentielles (1) et les exponentielles quadratiques (2).

$$C(d) = \exp(-d/V) \quad (1)$$

$$C(d) = \exp(-(d/V)^2); \quad (2)$$

Avec  $V$  étant le paramètre d'échelle et  $d$  la norme euclidienne de la distance entre deux points de l'espace considéré. *(plus le paramètre d'échelle est grand, plus le champ est lissé, tout comme le modèle quadratique est aussi plus lisse)* Une autre fonction de covariance, celle de *Matérn*, est aussi très utilisée, puisqu'elle présente des comportements limites similaires aux deux fonctions exponentielles présentes plus haut. La dérivabilité de ce modèle est réglable via le paramètre  $\nu$ , ainsi la régularité du champ est facilement modifiable.

$$C_\nu(d) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \sqrt{2\nu} \frac{d}{\rho} \right)^\nu K_\nu \left( \sqrt{2\nu} \frac{d}{\rho} \right); \quad (3)$$

La continuité des processus gaussiens et leur comportement quasi ondulatoire, permet de les traiter de manière un peu analogue à la décomposition de fourrier, et des les décomposer en une somme infinie de variables aléatoires gaussiennes décorréliées.. Cette méthode de décomposition vient du théorème de *Karhunen-Loève*.

La construction de cette série infinie se fait grâce aux vecteurs propres issus de la matrice de corrélation du modèle, et grâce à une base de vecteurs orthonormaux dans un espace Hilbertien. Les détails mathématiques de cette méthode peuvent être trouvées en [Sudret and Kiureghian, 2000].

Lors de l'approximation de *Karhunen-Loève*, la série est tronquée à l'ordre  $\mathbf{M}$ .

$H(\mathbf{x}, \theta)$	Approx. Procéssus Gaussien
$\lambda_i$	Valeur Propre de la matrice de covariance
$\xi_i$	Variable Normale Centrée Réduite
$\varphi_i(\mathbf{x})$	Vecteur propre de la matrice de covariance

$$H(\mathbf{x}, \theta) = \sum_{i=1}^M \sqrt{\lambda_i} \xi_i(\theta) \varphi_i(\mathbf{x}); \quad (4)$$

Cette décomposition permet de représenter l'ensemble de variabilité du champ avec des variables gaussiennes décorréélées, et donc de faire des analyses de sensibilités sur cette nouvelle représentation du champ.

### **openTURNS**

*openTURNS* est initialement un projet de bibliothèque open-source pour le traitement des incertitudes et des risques, commun à trois entreprises fondatrices, **Airbus**, **EDF** et **Phimeca Engineering**, projet ayant débuté en 2005. Depuis, deux autres organismes, **IMACS** et l'**ONERA** ont rejoint le développement d'*openTURNS*, qui se révèle être un grand atout pour le traitement des incertitudes et l'ingénierie fiabiliste. En effet, le projet d'*openTURNS* est un regroupement d'un ensemble d'algorithmes performants écrits en C++, se basant sur la théorie développée pour les traitements des incertitudes, l'optimisation robuste, et les études de sensibilité.

Un *wrapper* a été utilisée pour lier l'ensemble de la bibliothèque C++ à un module Python, et de pouvoir profiter de la facilité d'utilisation du langage Python et de sa flexibilité, tout en gardant les vitesses d'exécution du C++.

**PHIMECA** développe des logiciels commerciaux en se basant sur le module *openTURNS*, qui sont plus ergonomiques dans leur utilisation et automatisent certaines parties moins évidentes en utilisation directe de la librairie.

Néanmoins, la librairie reste quand même extrêmement bien documentée, avec la théorie mathématique sous-jacente à chaque algorithme d'explicitée, tout comme de nombreux exemples. La documentation présente de même des cas d'applications précis montrant la logique à avoir lors de l'écriture de codes avec *openTURNS*.

## **2.4 Exemple théorique simple, premiers codes**

Pour parvenir à tester les méthodes trouvées lors de la recherche bibliographique, et avoir un modèle simple et rapide à exécuter, un exemple simple a été développé : Il s'agit d'un modèle de poutre en appui sur ses deux extrémités



Figure 2: Modélisation d'une poutre en appui sur ses deux extrémités avec la bibliothèque *anastruct*

La particularité de ce modèle de poutre est, en plus d'être une aberration physique, le fait que le diamètre et le module de young sont gouvernés par un processus gaussien en une dimension, suivant la longueur de la barre. Pour parvenir à modéliser cette variation, le modèle est bien-sûr subdivisé en une centaine d'éléments finis, et le champ discrétisé sur un maillage de même longueur. En plus de ces deux grandeurs gouvernées par des champs, la densité du matériau, la position de la force, et la norme de la force sont représentés par des variables aléatoires Gaussiennes. Cette modélisation s'est faite avec l'excellente bibliothèque *Anastruct* de [Vink, 2020].



Figure 3: Échantillon de 5 réalisations de la poutre en flexion. [Sur la gauche : Processus gouvernant le module de young, Processus gouvernant le diamètre, contrainte de Von Mises] | [Sur la droite : Contrainte de cisaillement, moment de flexion, déflexion]

Les grandeurs d'étude choisies ici sont la contrainte de Von Mises, la déflexion de chaque noeud, et la déflexion maximale. On pourrait de même choisir une contrainte de Von Mises maximale pour représenter la défaillance.

Comme chacun des processus stochastiques peut être décomposé en une somme finie de variables aléatoires normales grâce à la décomposition de Karhunen-Loeve, l'on peut à l'inverse, générer des champs stochastiques à partir d'une collection de variables aléatoires, dont les paramètres sont issus de l'analyse même du champ.

On considère donc notre modèle plus comme fonction de variables aléatoires et de processus stochastiques, mais seulement comme une fonction de variables aléatoires scalaires.

Ceci permet de faire l'analyse de Sobol classique (avec l'algorithme de sensibilité de *Saltelli*) sur ces nouvelles variables aléatoires.

La difficulté ici, est d'ensuite relier les indices de sensibilité de Sobol des variables aléatoires définissant le champ au champ stochastique même. (On ne souhaite n'avoir qu'un seul indice de Sobol pour un champ.)

## 2.5 Choix des codes et difficultés algorithmiques

Au vu de choix de développer une méthode algorithmique pour l'analyse de sensibilité sur des champs stochastiques, et non pas de juste faire l'analyse de sensibilité sur une seule problématique, il y avait un besoin de robustesse supplémentaire pour l'écriture des différents codes. Ces derniers doivent pouvoir fonctionner avec tout type de fonction en python prenant en entrée des champs et variables aléatoire, et renvoyant un tuple de variables aléatoires.

Pour pouvoir tester ceci, l'exemple de la poutre en éléments fins a été codée à part, et peut être utilisée directement comme fonction, indépendamment de l'analyse de sensibilité. Cette dernière prend de même en entrée deux champs stochastiques et trois variables aléatoires, et peut renvoyer plusieurs arguments. Les tests ont été faits avec la fonction ne renvoyant qu'une collection de champs (contraintes de von mises) et avec la fonction renvoyant un tuple (collection de champs + vecteur déflexion maximale). Ce choix augmente le nombre de vérification qu'il y a à effectuer au sein du code, mais permet l'utilisation d'un plus grand nombre de fonctions, et l'analyse de sensibilité sur l'ensemble des variables de sortie en une seule fois.

Pour parvenir à bien contrôler les processus stochastiques, une classe a été écrite *NdGaussianProcessConstructor*, qui permet de définir entièrement un processus stochastique, et possède de nombreuses méthodes pour créer des échantillons, faire les décompositions de Karhunen-Loeve, ou encore reconstituer un champ à partir de la dite décomposition. Enfin, comme les échantillons sont nécessaires pour la décomposition de Karhunen-Loeve, et que ceux-ci peuvent être de taille plutôt importante, ils sont enregistrés sous forme de *numpy.memmap* dans un fichier temporaire, pour relâcher un peu de pression sur la mémoire vive.

En interne, le modèle fonction de champs stochastiques, passe par un wrapper qui fait qu'on peut la considérer comme seulement fonction de variables aléatoires gaussiennes. Cela permet d'utiliser les méthodes internes à *openturns* comme l'algorithme de Saltelli pour estimer les indices de Sobol.



Néanmoins, pour pallier certaines difficultés comme les fonctions qui par moment renvoient des ***np.nan*** (donc lorsqu'il y a eu une erreur), on passe d'abord par une étape de correction: l'ensemble des valeurs de sortie contenant des *nan* sont régénérées, et comme les entrées de l'étude sont des combinaison de deux ensembles de variables aléatoires A et B, toutes les autres permutations des entrées ayant entraîné des erreurs sont supprimées et remplacées. Cette étape est expliquée en figure 4

Les codes sont accessibles via ***github***:  
[https://github.com/Kramer84/stochastic\\_process\\_analysis](https://github.com/Kramer84/stochastic_process_analysis)



Figure 4: Organsiation des variables d'entrée, expliquant l'étape de postprocessing

Une fois cette étape finie, on a accès aux indices de sobol pour chaque élément du champ.

Dans le cas de la poutre en flexion, prennant initialement 5 arguments en entrée (champ du module de young et champ du diamètre, densité, position de la force et norme de la force), la décomposition des deux champs en entrée fait que le modèle est ensuite régi par 24 variables aléatoires. Bien que cela fait augmenter la dimensionnalité du problème, nous avons désormais la possiblité de travailler seuelement avec des variables aléatoires gaussiennes décorréliées, et donc faire une analyse de sensibilité classique.

La difficulté qui découle de cette analyse, est de trouver la manière qui permettra de relier les indices de sobol de ces variables aléatoire décorréliées, au champ

stochastique d'origine.

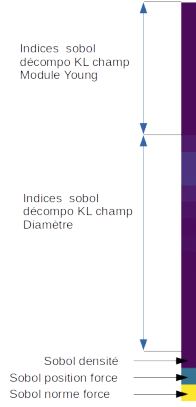


Figure 5: indices de sobol des V.A. de Karhunen-Loève pour la déflexion maximale

Pour parvenir à faire ce lien, nous avons comme base différentes études sur le sujet, notamment le papier par [Wei et al., 2017], sur l'analyse de sensibilité sur des modèles ayant des paramètres dépendants du temps. (Les variables étant gouvernées par le temps sont généralement stochastiques, ex: température, effort etc. )

Mais ayant fait une analyse préliminaire de ces indices de sobol', des incohérences dans ces indices ont pu être remarqués. En effet, la somme de l'ensemble des indices du premier ordre doit être inférieure à 1, et il ne peut y avoir d'indices négatifs. L'apparition d'indices de Sobol' négatifs peut être expliqué par un nombre d'expériences trop faible, nombre d'expériences qui lui même est limité par la mémoire disponible dans l'ordinateur.

## 2.6 Premiers essais d'analyse de la sensibilité du modèle

Comme nous avons désormais accès aux différents indices de sobol de notre modèle basé sur la décomposition de Karhunen-Loève, le travail pouvait désormais se focaliser sur l'interprétation de ces différents indices. En effet, cette décomposition était très classique lors du travail avec des champs stochastiques, et plusieurs papiers de recherche se sont intéressés à comment faire des analyses de sensibilité de ce type de champs avec cette même décomposition. Comme mentionné précédemment, le travail de [Wei et al., 2017] offre une première approche pour l'interprétation de ces résultats, mais d'autres comme [Pronzato, 2019] ont aussi développés des méthodes pour l'analyse de ces indices issus de l'expansion de Karhunen-Loève. Dans le travail de [Shang and Yun, 2013], il est rapporté d'une méthodologie pour lier un modèle en élément finis d'Abaqus avec des champs stochastiques générés avec MATLAB et pour l'analyse probabiliste de ce modèle. L'utilisation de l'expansion de KL y est aussi mentionné, et ce travail offre une analyse intéressante sur la capacité de cette méthode à reproduire



Figure 6: indices de sobol des V.A. de Karhunen-Loève pour le champ des contrainte de Von Mises

le champs stochastique, en fonction de l'élément à partir duquel on tronquait cette expansion.

### 2.6.1 Interprétation et essais avec le papier : "Time-variant global reliability sensitivity analysis of structures with both input random variables and stochastic processes"

Ce papier de recherche datant de 2017, offre une approche pour l'analyse de sensibilité de modèles définis dans le temps et leur analyse de fiabilité. En effet, de nombreuses structures mécaniques (ponts, tours etc.) sont soumis à des grandeurs variant dans le temps, et si celles se trouvent être continues dans celui-ci, alors cette grandeur peut être représentée par un champ stochastique. Bien qu'il y ait une différence notable entre les grandeurs par rapport auxquelles il y a continuité (dans leur papier le temps, et nous l'espace), des analogies peuvent être faites.

L'approche se base sur la décomposition de Karhunen-Loève du champ stochastique, permettant de considérer le problème comme uniquement dépendant de variables aléatoires gaussiennes décorrélées, et de déduire la sensibilité du modèle au champ stochastique à partir de la sensibilité de chaque composant de la décomposition du champ.

La décomposition de KL est particulièrement utile pour le calcul de fonction d'enveloppe et les calculs fiabilistes. Dans ce papier, l'utilisation de l'expérience de Monte-Carlo est essentiellement utilisée pour être comparée aux autres méthodes développées (**FOEF** : First Order Envelope Function et **AK-MCS** Active learning Kriging based Monte Carlo Simulation).

L'utilité de cette approche est de pouvoir identifier les sources principales d'incertitude ayant le plus d'influence sur la fiabilité du modèle. Enfin, elle permet de classer les variables selon leur ordre d'influence.

Néanmoins, leur approche se base sur la définition d'un état limite du modèle (par exemple la différence entre la contrainte maximale et la contrainte limite) et d'une fonction indicatrice de l'état limite, alors que nous cherchons à évaluer la sensibilité d'une grandeur en sortie (par exemple la contrainte) par rapport aux variables aléatoires en entrée. Ceci revient à priori à mesurer l'impact qu'une variable en entrée sur la variance de la grandeur en sortie.

La définition mathématique de leur problème est la suivante :

- $S$  : Etat limite du modèle
- $\mathbf{X}$  : Vecteur de variables aléatoires
- $\mathbf{Y}(t)$  : Vecteur de champs stochastiques dépendants du temps
- $Z$  : Fonction de performance
- $I_S$  : Fonction indicatrice
- $R$  : Probabilité de défaillance sur l'intervalle  $[0, T]$

$$S = \{(\mathbf{X}, \mathbf{Y}) : Z(t) = g(\mathbf{X}, \mathbf{Y}(t), t) < 0 \forall t \in [0, T]\}; \quad (5)$$

$$I_S(t) = I_S(\mathbf{X}, \mathbf{Y}(t), T) = \begin{cases} 1 & (\mathbf{X}, \mathbf{Y}) \in S \\ 0 & (\mathbf{X}, \mathbf{Y}) \notin S \end{cases}; \quad (6)$$

$$R = R(T) = P_r(I_S = 1) = Pr(Z(t) = g(\mathbf{X}, \mathbf{Y}(t)) < 0, \forall t \in [0, T]); \quad (7)$$

Les variables aléatoire et champs stochastiques ici définis sont tous indépendants entre eux.

Soient  $\mu_X = (\mu_{X_1}, \dots, \mu_{X_n})$  et  $\sigma_X = (\sigma_{X_1}, \dots, \sigma_{X_n})$  les vecteurs des moyennes et écarts types des variables aléatoires et  $\mu_Y = (\mu_{Y_1}, \dots, \mu_{Y_n})$  et  $\sigma_Y = (\sigma_{Y_1}, \dots, \sigma_{Y_n})$  les vecteurs des moyennes et écarts types des champs stochastiques.

Comme explicité lors de la définition de la décomposition de Karhunen-Loève, tout champ stochastique  $Y_i(t)$  peut être écrit comme une somme de variables aléatoire gaussiennes décorréliées.

$Y_i(t)$	Approx. Procéssus Gaussien $Y_i$
$\lambda_{ik}$	Valeur Propre de la matrice de covariance
$\xi_{ik}$	Variable Normale Centrée Réduite
$\varphi_{ik}(t)$	Vecteur propre de la matrice de covariance

$$Y_i(t) = \mu_{Y_i}(t) + \sum_{k=1}^M \sqrt{\lambda_{ik}} \xi_{ik} \varphi_{ik}(t); \quad (8)$$

D'où chaque champ gaussien peut être exprimé à partir d'un ensemble de variables aléatoire gaussiennes décorélées  $\xi_i = (\xi_{i1}, \dots, \xi_{iM_i})$  et l'ensemble des champs est représenté par le vecteur  $\xi = (\xi_1, \dots, \xi_m)$ .  $\xi_i$  étant la représentation par des variables auxilliaires du champ  $Y_i$ .

Grâce à la propriété de décomposition de la variance, on a :

$$\begin{aligned} V(I_S) &= \sum_{i=1}^n V_{X_i} + \sum_{i=1}^m V_{\xi_i} + \sum_{i=1}^n \sum_{j \neq i, j=1}^n V_{X_i X_j} \\ &+ \sum_{i=1}^m \sum_{j \neq i, j=1}^m V_{\xi_i \xi_j} + \sum_{i=1}^n \sum_{j=1}^m V_{X_i \xi_j} + \dots \\ &+ V_{X_1 \dots X_n \xi_1, \dots, \xi_m} \end{aligned}$$

Figure 7: Décomposition de la variance de la fonction indicatrice

Ici, c'est la variance de la fonction indicatrice  $I_s$  en sortie qui est traitée, mais ceci est faisable pour tout modèle dépendant d'un ensemble de variables aléatoire. Il est à remarquer ici que :  $V_{X_i} = V[E(I_S|X_i)]$  qui est la variance

conditionnelle de la sortie  $I_S$  lorsque la variable aléatoire  $X_i$  est connue.

Néanmoins, dans le cas du vecteur  $\xi_i = (\xi_{i_1}, \dots, \xi_{i_{M_i}})$  on a:

$$V_{\xi_i} = V[E(I_S|\xi_i)] = V[E(I_S|(\xi_{i_1}, \dots, \xi_{i_{M_i}}))] - \sum_{p=1}^{M_i} V_{\xi_{i_p}} \quad (9)$$

qui est la variance conditionnelle de la sortie lorsque le vecteur aléatoire  $\xi_i$  représentatif du champ stochastique  $Y_i$  est connu.

Pour le calcul du vecteur des variances  $V[E(I_S|(\xi_{i_1}, \dots, \xi_{i_{M_i}}))]$ , dans le cadre d'une expérience de Monte-Carlo, ceci pourrait se faire en rajoutant dans le plan d'expérience, toutes les permutations associées à ces variables annexes aux champs stochastiques.

Avec le plan d'expérience classique proposé par openTURNS, nous sommes en capacité de facilement calculer les indices de Sobol' du 1<sup>er</sup> ordre de toutes les variables  $S_{X_i}$  et  $S_{\xi_{i_p}}$ . Or nous cherchons  $S_{\xi_i}$ . Dans le cas d'un modèle  $F$ , prenant en entrée un vecteur de champs stochastiques  $Y = (Y_1, \dots, Y_m)$  et un vecteur de variables aléatoires  $X = (X_1, \dots, X_n)$  et défini comme suit:

$$\omega = F(X, Y), \omega \in \mathbb{R}^N \quad (10)$$

On peut définir une fonction  $F'$ , utilisant la décomposition de Karhunen-Loève et seulement dépendante d'un ensemble de variables aléatoires décorréliées:

$$\omega = F'(X, \xi) \quad (11)$$

$$\xi = (\xi_1, \dots, \xi_m) \quad (12)$$

$$\xi_i = (\xi_{i_1}, \dots, \xi_{i_{M_i}}) \quad (13)$$

Nous savons que :

$$V_{X_i} = V[E[\omega|X_i]] \quad (14)$$

$$S_{X_i} = \frac{V_{X_i}}{V(\omega)} \quad (15)$$

$$V_{X_i X_j} = V[E[\omega|X_i X_j]] - V_{X_i} - V_{X_j} \quad (16)$$

$$S_{X_i X_j} = \frac{1}{V(\omega)} V[E[\omega|X_i X_j]] - S_{X_i} - S_{X_j} \quad (17)$$

Néanmoins, à la différence des effets d'interactions et les sensibilités qui y sont associées, dans la cas de la sensibilité du champ stochastique, il semblerait qu'on ne soustraie pas l'effet de l'interaction du premier ordre, donc on n'aurait pas :

$$S_{\xi_i} = S_{\xi_{i_1}, \dots, \xi_{i_{M_i}}} = \frac{1}{V(\omega)} V[E[\omega|\xi_{i_1}, \dots, \xi_{i_{M_i}}]] - \sum_{p=1}^{M_i} S_{\xi_{i_p}} \quad (18)$$

Mais plutôt seulement :

$$S_{\xi_i} = \frac{1}{V(\omega)} V[E[\omega|\xi_{i_1}, \dots, \xi_{i_{M_i}}]] \quad (19)$$

Ceci est intéressant dans la sens ou cela diminue grandement la complexité du calcul et qu'il ne faut plus que calculer les effets d'interaction et non plus les indices de sobol du premier ordre associés aux indices de KL des champs stochastiques.

Le calcul à posteriori de ces interactions est possible, vu que nous n'avons qu'à comparer la moyenne des variance des deux échantillons de départ (**A** et **B**) à celle lorsque tout les paramètres du champs stochastique  $i$  sont connus ( $V_{\xi_i}$ ).

Néanmoins, l'on ne peut pas inclure celles-ci dans le plan d'expérience utilisé par openTURNS, vu que l'algorithme utilisé pour calculer les indices de Sobol' n'est conçu que pour les calculs des indices de premier et second ordre de modèles gouvernés par un ensemble de variables aléatoires unitaires, et non des champs stochastiques. Pour parvenir à passer outre cette difficulté, un choix d'algorithme doit être fait pour le calcul de ces indices de sobol d'ordre fortement supérieur. (Vu que pour les indices d'ordre 1, le choix de l'algorithme est entièrement libre.) Au vu du travail fait par PHIMECA sur l'analyse des lois asymptotiques des estimateurs des indices de Sobol [Dumas, 2017], l'on prendra comme estimateur celui de Martinez, mais il sera évidemment possible de rajouter à posteriori les autres estimateurs dans le workflow.

D'autre part, pour parvenir à garder cette complexité faible, nous sommes obligés de revoir la manière dont est fait le plan d'expérience (la manière dont on mélange les différentes colonnes de nos deux samples **A** et **B**) puisque dans le cas de l'exemple de la poutre, nous aurions jusqu'à 5 fois moins de calculs à effectuer qu'en gardant tout les indices inutiles.

### Génération d'expérience pour analyse de sensibilité sur champs stochastiques

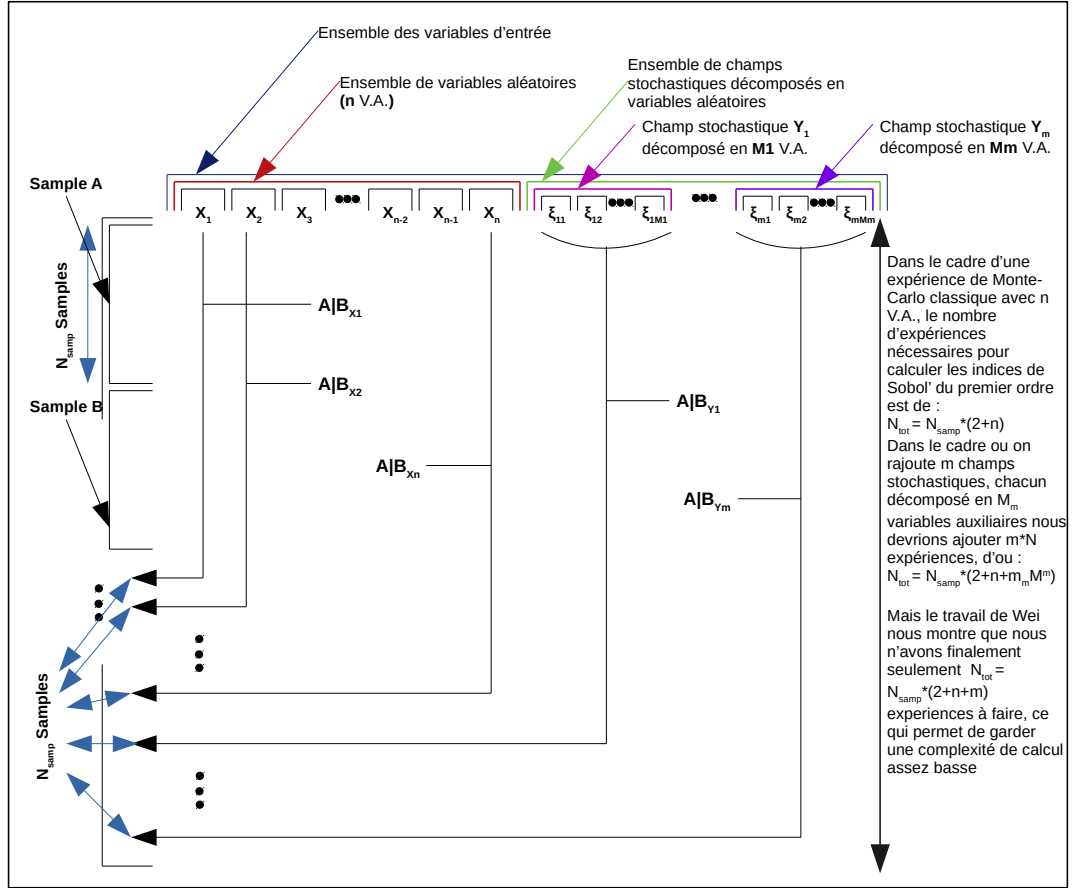


Figure 8: Génération d'expérience pour analyse de sensibilité sur champs stochastiques

Les échantillons sont mélangés comme montré en figure 8. Pour générer les samples de manière plus efficace, et être sûr de couvrir l'ensemble du domaine de variance des lois d'entrée, est inclu dans le code la possibilité de générer l'échantillon en utilisant soit l'échantillonnage LHS (Latin Hypercube Sampling), soit les séries à faible divergence (Low discrepancy series), soit une version optimisée du LHS (Simulated Annealing LHS). Bien sûr une génération entièrement aléatoire reste l'option de base.

Pour le calcul formel de l'estimateur de Sobol', nous allons utiliser la méthode par Saltelli, détaillée dans le papier par [Dumas, 2017]. Ce papier détail aussi la manière de calculer l'erreur dans le calcul de l'estimateur, valeur importante pour connaître la pertinence de l'estimateur.

Détail du calcul de l'indice de Sobol' :

${}^c\mathbf{Y}^A$  : Sortie du modèle associé à l'échantillon A du plan d'expérience

${}^c\mathbf{Y}^B$  : Sortie du modèle associé à l'échantillon B du plan d'expérience

${}^c\mathbf{Y}^E$  : Sortie du modèle associé à l'échantillon issu de A ou l'on a inséré une partie de B

$\mathbf{S}$  : Indice de Sobol'

Les échantillons centrés sont annotés du prescript c

$$S = \frac{Cov[{}^cY^B, {}^cY^E]}{Var[{}^cY^A]} \quad (20)$$

$$= \frac{\frac{1}{n} \sum_{k=0}^n {}^cY^B {}^cY^E - \left(\frac{1}{n} \sum_{k=1}^n {}^cY^B\right) \left(\frac{1}{n} \sum_{k=1}^n {}^cY^A\right)}{\frac{1}{n} \sum_{k=0}^n ({}^cY^B)^2 - \left(\frac{1}{n} \sum_{k=0}^n {}^cY^B\right)^2} \quad (21)$$

Comme les échantillons sont centrés par rapport à leurs moyennes respectives, le numérateur et le dénominateur se simplifient et on obtient l'estimateur suivant :

$$S = \frac{\frac{1}{n} \sum_{k=0}^n {}^cY^B {}^cY^E}{\frac{1}{n} \sum_{k=0}^n ({}^cY^A)^2} \quad (22)$$

Avec l'échantillonnage présenté précédemment, le calcul de l'estimateur de Sobol se fait très directement avec la formule de Saltelli, même lorsque la dimension de la sortie est plus grande que 1.

Pour pouvoir justifier de la pertinence de l'estimateur, on calcule l'intervalle de confiance à 0.975, qui permet de montrer la précision de l'estimateur et de nuancer son analyse. Pour le calcul de cet intervalle de confiance, on utilise la méthode Delta, méthode utilisée par Jansen *et al* (2014) ou de même par [Dumas, 2017]. Le détail de cette méthode est donné ci dessous.

On pose suite à l'équation simplifiée précédente :

$$U_i = \left( {}^cY^B {}^cY^E, ({}^cY^A)^2 \right)^T \quad (23)$$

Ensuite l'on définit :

$$\Psi_S(x, y) = \frac{x}{y} \quad (24)$$



Ceci implique :

$$S_N = \Psi_S(\bar{U}_N) \quad (25)$$

Grâce au théorème central limite:

$$\sqrt{N}(\bar{U}_N - \mu) \xrightarrow[N \rightarrow \infty]{\mathcal{L}} \mathcal{N}_2(0, \Gamma) \quad (26)$$

Où  $\Gamma$  est la matrice de covariance de  $U_1$  et :

$$\mu = \begin{cases} Cov[Y^B, Y^E] \\ Var[Y^A] \end{cases} \quad (27)$$

De l'utilisation de la méthode delta :

$$\sqrt{N}(S_N - S) \xrightarrow[N \rightarrow \infty]{\mathcal{L}} \mathcal{N}_1(0, g^T \Gamma g) \quad (28)$$

ou  $g = \nabla \Psi_S(\mu)$ .

Finalement, pour tout  $x, y$  tel que  $y \neq 0$  :

$$\nabla \Psi_S(x, y) = \left( \frac{1}{y}, \frac{-x}{y^2} \right)^T \quad (29)$$

La matrice de covariance  $\Gamma$ , tout comme le vecteur  $g$  se calculent bien analytiquement dans le cas de l'indicateur de Saltelli', néanmoins le calcul du gradient peut très bien se faire informatiquement, notamment dans le cas d'estimateurs plus complexes.

Pour récupérer l'intervalle de confiance de l'estimateur nous utilisons le quantile de la loi normale centrée réduite à 97.5%, multiplié par la racine de la variance calculée précédemment.

Finalement, l'on récupère les indices de Sobol' pour les différents éléments de la poutre tout comme leur intervalle de confiance:

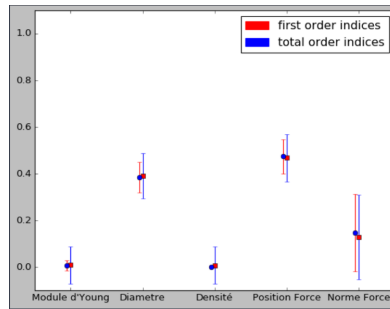


Figure 9: Sensibilité de la déflexion maximale aux champs et variables d'entrée.

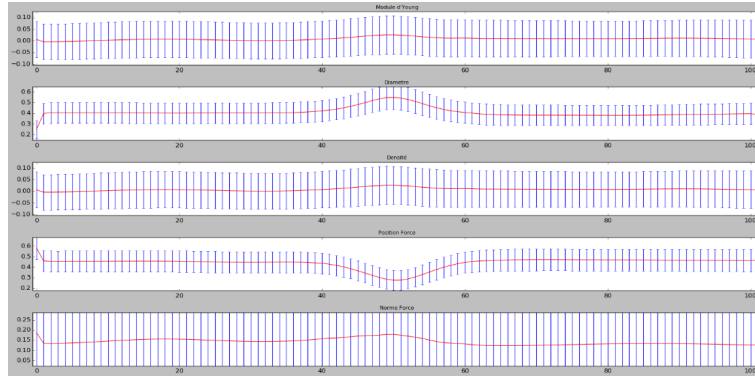


Figure 10: Sensibilité de la contrainte de Von-Mises aux champs (Module d'Young, diamètre) et variables d'entrée.(densité, position, Norme de la force )

L'on peut voir qu'il y a très peu d'écarts entre les indices d'ordre total et premier ordre. Les effets d'interaction sont donc très peu présents entre les variables d'entrée.

Finalement une comparaison entre un exemple de la documentation d'*openTURNS* et les codes développés durant ce stage peut être faite, pour confirmer que les méthodes implémentées sont exemptes d'erreurs.

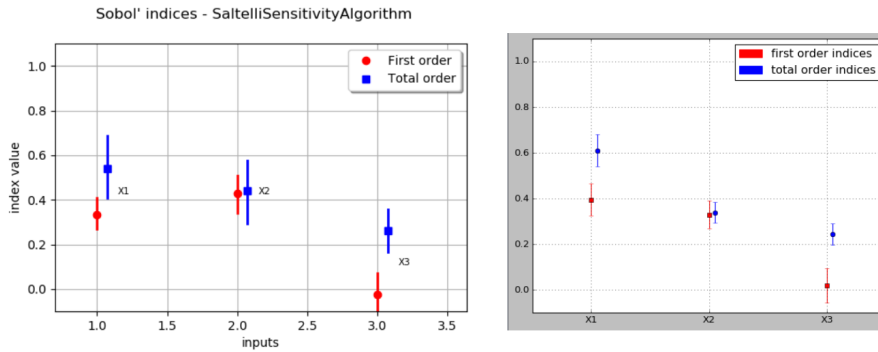


Figure 11: Comparaison entre la solution par openturns (à gauche) et la notre (à droite), pour un échantillon de taille 1000 sur la fonction d'Ishigami

L'on observe, que pour cet échantillon de 1000, l'indice de Sobol' calculé avec nos codes est relativement proche de ceux calculés avec *openTURNS*, même si des écarts subsistent, notamment au niveau de la variance. Cette variance s'explique en partie par l'incertitude dans les variables d'entrées (tout échantillon de 1000 est à priori unique).

Les codes ayant montré leur utilisabilité, il était enfin possible d'essayer d'appliquer cette méthodologie à un système plus complexe, un échangeur de chaleur air-air modélisé avec NASTRAN.

## 2.7 Application au modèle NASTRAN d'échangeur thermique

Comme nous l'avions mentionné avant, la finalité de ce travail de quasi-recherche et développement de codes, était de parvenir à induire des défauts stochastiques dans un modèle numérique, et de mesurer la sensibilité de ce modèle à ces perturbations.

Le modèle en question est celui d'un échangeur thermique d'utilisation aéronautique. Cette pièce doit être capable de supporter et échanger la température de deux flux d'air perpendiculaires, l'un à la température de l'air en haute altitude - jusqu'à  $-50^{\circ}$  - et l'autre aux températures moteurs allant jusqu'à  $600^{\circ}\text{C}$ .

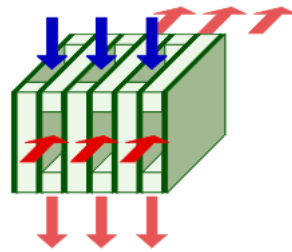


Figure 12: Principe de l'échangeur à chaleur air-air

Ce projet est financé par l'union européenne, et la fabrication est assurée par le fabricant allemand **LIEBHERR**. Ce dernier sous-traite certaines étapes de l'étude, comme les essais physiques sur des échantillons de l'échangeur, tâche réalisée par **Lortek**, la construction d'un modèle paramétrique numérique de l'échangeur, tâche réalisée par **Epsilon**, et enfin la partie analyse de sensibilité et déformation moteur réalisée par **Phimeca**.

Ce projet a donc impliqué l'étroite collaboration de ces différents acteurs, vu que pour l'analyse de sensibilité il nous fallait utiliser le modèle construit par **Epsilon**, et introduire des incertitudes dans les paramètres, dont la nature et les paramètres ont été récupérées dans les mesures faites par **Lortek**.

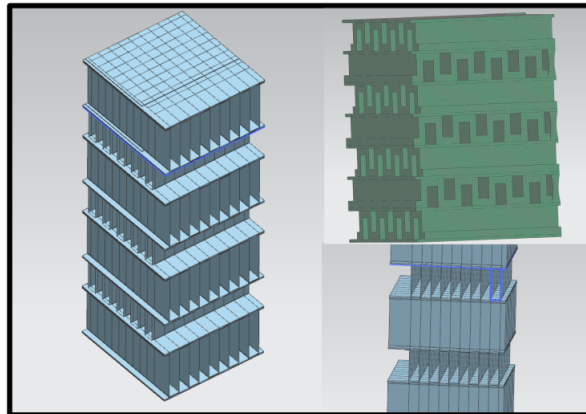


Figure 13: Modèle Nastran de l'échangeur de chaleur - Siemens Simcenter 3D

Pour bien se coordonner sur le projet, des réunions se sont régulièrement tenues, permettant de se tenir à jour sur les avancements de chacun, et de définir les prochaines étapes à atteindre.

Dans le cas de l'étude à réaliser par **Phimeca**, le sujet d'analyse était l'étude de l'influence de la variabilité dans les paramètres des ailettes de l'échangeur de chaleur sur la localisation de la défaillance. Dans le cas de l'échangeur de chaleur, la défaillance correspond à une rupture d'ailette, ou une perte d'étanchéité.

Numériquement, la rupture se traduit par une contrainte maximale supérieure à la contrainte maximale supportée par le matériau constituant de la pièce. L'analyse de la pièce se traduit donc par l'étude de l'effet de la maîtrise des paramètres de l'ailette (épaisseur, planéité, angle...) sur la localisation de la contrainte maximale.

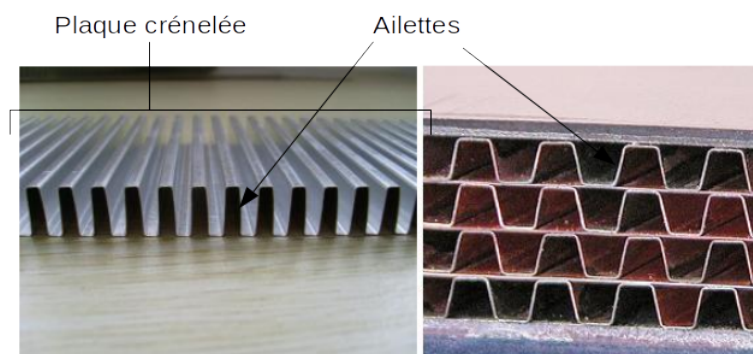


Figure 14: Ailettes d'un échangeur de chaleur

Pour parvenir à introduire ces variabilités différentes approches et modèles de champs stochastiques peuvent être prises en considération, puisque les réali-

sations peuvent fortement varier en fonction des paramètres et modèles. Il est en outre difficile de générer un champ qui serait indépendant du maillage sur lequel il est réalisé, ou du moins qui ressemblerait à un échantillon d'un champ plus grand.

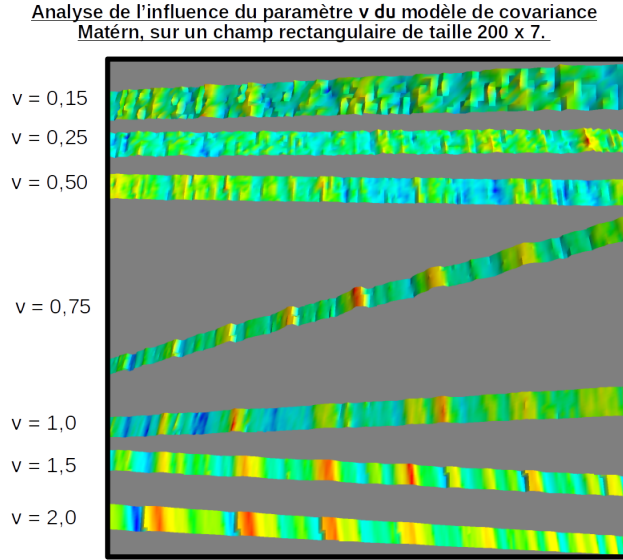


Figure 15: Impacte d'un paramètre sur, un champ

En deux dimensions, un champ stochastique peut être vu comme un maillage uniforme, ou une valeur est associée à chaque noeud, et ou la valeur à chaque noeud est colérée à la distance et valeur des noeuds proches. La première idée aurait donc été d'appliquer directement une déformation générée par un champ de la même taille que le maillage d'une ailette aux noeuds constituant l'ailette, et donc de simuler une déformation stochastique par ailette, mais en écartant du coup toute corrélation inter-ailettes. La déformation se serait fait par translation de chaque noeud suivant l'axe normal au plan de l'ailette. Cette méthode a vite été écartée, puisque la génération de champs sur des maillages sous forme de long rectangle étiré était bien moins précise et contrôlables, allant jusqu'à des réalisations quasi constantes. De plus le nombre de champs générés par plaque serait très important, alors qu'il serait plus simple de ne générer qu'un seul champ par plaque et d'utiliser de déformation paramétriques.

Cette seconde approche est inspirée d'un travail par CITEGOKA, qui traite de la modélisation des déformations réelles de pièces mécaniques. Il découle de son travail, que la déformation d'une pièce physique peut être représentée par une somme de modes de déformation primaires, et qui similairement à une décomposition de Fourier, peut finalement représenter toute déformation de pièce.

En exprimant la déformation de chaque arc de noeuds colinéaire à la direction normale aux deux flux d'air, comme une fonction mathématique représentant cette déformation basique, par exemple un polynôme du second degré possédant

comme racines les deux positions extrêmes de l'arc, et en rendant cette fonction pilotée par un unique paramètre, l'on peut appliquer facilement une déformation à chaque arc de la plaque crénelée. Cette déformation, en plus d'être physique, possède une corrélation intra-ailettes, et ne nécessite que la génération d'un seul champ par plaque crénelée. De plus on peut facilement superposer les déformations par sommation.

Un exemple de déformation qu'il est possible d'obtenir avec cette méthode (un seul mode de déformation):

```
PROGRAM area
REAL base, height, area
PRINT *, 'Enter the values for the base and height of a triangle.'
READ *, base, height
area = (1.0/2.0) * base * height
PRINT *, 'The area of a triangle with base ', base
PRINT *, 'and height ', height, ' is ', area
STOP
END
```

**Note:** In verbatim mode you can easily end up outside the margins, as in the example above: pay attention to that!

## 2.8 Lists

Example of a list with numbered items:

1. Planets, asteroids, moons ...
2. Stars, galaxies, quasars

Example of a list with unnumbered items:

- Planets, asteroids, moons ...
- Stars, galaxies, quasars

## 3 Results

## 4 Tables and figures

Table 1: Example of table caption: opacity sources.

Source	$T/[\text{K}]$
Yorke 1979, Yorke 1980a	$\leq 1700$
Krügel 1971	$1700 \leq T \leq 5000$
Cox & Stewart 1969	$5000 \leq$

## 5 Discussion

## 6 Conclusions

## References

- [Dumas, 2017] Dumas, A. (2017). Lois asymptotiques des estimateurs des indices de sobol. *confidentiel EDF*.
- [Lilburne and Tarantola, 2009] Lilburne, L. and Tarantola, S. (2009). Sensitivity analysis of spatial models. *International Journal of Geographical Information Science*, 23(2):151–168.
- [Michaël Baudin and Popelin, 2015] Michaël Baudin, Anne Dutfoy, B. I. and Popelin, A.-L. (2015). Openturns: An industrial software for uncertainty quantification in simulation.
- [Pronzato, 2019] Pronzato, L. (2019). Sensitivity analysis via Karhunen–Loève expansion of a random field model: Estimation of Sobol’ indices and experimental design. *Reliab. Eng. Syst. Saf.*, 187:93–109.
- [Shang and Yun, 2013] Shang, S. and Yun, G. J. (2013). Stochastic finite element with material uncertainties: Implementation in a general purpose simulation program. *Finite Elements in Analysis and Design*, 64:65–78.
- [Sudret and Kiureghian, 2000] Sudret, B. and Kiureghian, A. D. (2000). Stochastic Finite Element Methods and Reliability A State-of-the-Art Report. *ResearchGate*.
- [Vink, 2020] Vink, R. (2020). A nonlinear water accumulation analysis in Python - Ritchie Vink. [Online; accessed 2. Mar. 2020].
- [Wei et al., 2017] Wei, P., Wang, Y., and Tang, C. (2017). Time-variant global reliability sensitivity analysis of structures with both input random variables and stochastic processes. *Struct. Multidiscip. Optim.*, 55(5):1883–1898.