High-dimensional sensitivity analysis methods for computationally expensive simulators modeling a severe nuclear accident

Faouzi Hakimi

▶ **To cite this version:**

Faouzi Hakimi. High-dimensional sensitivity analysis methods for computationally expensive simulators modeling a severe nuclear accident. Statistics [math.ST]. Université Paul Sabatier - Toulouse III, 2023. English. NNT : 2023TOU30026 . tel-04116502v2

HAL Id: tel-04116502
https://theses.hal.science/tel-04116502v2

Submitted on 24 Aug 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE
Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue par

Faouzi HAKIMI

Le 8 février 2023

High-dimensional sensitivity analysis methods for computationally expensive simulators modeling a severe nuclear accident

---

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Mathématiques et Applications**

Unité de recherche :
IMT : Institut de Mathématiques de Toulouse

Thèse dirigée par
Fabrice GAMBOA, Amandine MARREL, Claude BRAYER et Benoît HABERT

Jury

M. Jean BACCOU, Rapporteur
M. Alberto PASANISI, Rapporteur
M. Jean-Marc BARDET, Rapporteur
Mme Céline HELBERT, Examinatrice
M. Olivier ROUSTANT, Examinateur
M. Fabrice GAMBOA, Co-directeur de thèse
Mme Amandine MARREL, Co-directrice de thèse
Mme Elena DI BERNARDINO, Présidente

Be like a rocky promontory against which the restless surf continually pounds; it strands fast while the churning sea is lulled to sleep at its feet.

Marcus Aurelius

Remerciements

J'ai fait les efforts et les erreurs qu'il fallait pour être là où je suis aujourd'hui et j'en suis fier ! J'écris mon histoire et la nôtre prend fin. Ainsi, je tiens tout d'abord à remercier Amandine Marrel et Fabrice Gamboa, j'ai beaucoup appris sous votre direction. Je remercie également Claude Brayer et Benoît Habert pour m'avoir accompagné, suivi et aidé pendant ces trois années de thèse.

Je tiens ensuite à remercier Jean Baccou, Jean-Marc Bardet et Alessandro Pasanisi pour avoir rapporté mon travail de thèse et pour la qualité de leurs retours. Je voudrais aussi remercier Céline Helbert, Elena Di Bernadino et Olivier Roustant pour avoir fait partie de mon jury de thèse ainsi que pour les échanges très intéressants que l'ont a pu avoir lors de la soutenance.

Par ailleurs, je remercie mon ami et collègue Anouar Meynaoui pour ses précieux conseils, sa bienveillance et son expertise.

Mes remerciements s'adressent ensuite à l'ensemble des personnes que j'ai pu côtoyer pendant ces trois ans de thèse, qu'il s'agisse de mes collègues du CEA (#MCS2027) ou des personnes rencontrées à l'université Paul Sabatier, en colloques ou en conférences, pour le bon temps passé ensemble. Je remercie également du fond du coeur l'ensemble de mes proches pour m'avoir accompagné et soutenu avant, pendant et j'espère après cette thèse.

Enfin, je tiens à remercier Internet et sa communauté, pour avoir répondu à mes questions, des plus simples aux plus pointues, m'avoir fait rire dans les moments doutes, et m'avoir donné des outils, des ressources et des méthodes qui m'ont servi tout au long de mon doctorat.

Résumé

Une grande part de l'électricité consommée en France étant produite par des réacteurs nucléaires, la recherche dans ce domaine constitue un enjeu industriel important. Ce travail de thèse s'inscrit dans ce contexte et est le résultat d'une collaboration entre le CEA et l'Université Paul Sabatier (Toulouse, France).

Cette thèse a pour objet l'étude d'un code industriel modélisant un accident nucléaire grave, l'interaction corium-eau, et plus précisément le traitement des incertitudes sur les paramètres de modélisation de ce code. Pour étudier ce code, une méthodologie basée sur différents outils statistiques avancés a été proposée. Elle se décompose en cinq grandes étapes :

1. **Échantillonnage de l'espace des entrées.** Connaissant le domaine de variation des paramètres d'entrée, l'espace de ces paramètres a été échantillonné. Une exécution du code a été ensuite effectuée sur chaque point du plan d'expérience ainsi construit. Le jeu de données entrées/sorties obtenu constitue l'échantillon d'apprentissage. Le but ici est d'explorer avec cet échantillon l'espace des entrées et d'obtenir autant d'informations que possible sur le comportement des sorties du code. Ne disposant que d'un budget limité pour les simulations, nous avons utilisé une méthode d'échantillonnage quasi aléatoire, appelée Latin Hypercube Sampling (LHS), au lieu d'une méthode Monte Carlo classique. Cette méthode assure une meilleure couverture spatiale des distributions marginales tout en conservant certaines propriétés de l'échantillonnage purement aléatoire. La convergence asymptotique des statistiques utilisées dans notre méthodologie sous LHS a également été discutée, avec notamment un travail théorique sur la convergence asymptotique de la classe des Z-estimateurs avec un LHS.

2. **Analyse de sensibilité des échecs de code.** Au cours de l'exploration de l'espace d'entrée, nous avons constaté qu'une proportion importante des exécutions du code ne convergeait pas. Nous avons donc analysé ces échecs de code pour voir quelles entrées ont le plus d'influence sur eux, ce qui nous a permis de mieux comprendre le fonctionnement du code.

3. **Sélection des entrées à partir de l'analyse de sensibilité des sorties.** À partir du jeu de données d'apprentissage, un tri a été effectué en utilisant deux outils d'analyse de sensibilité. Le premier outil de sélection est basé sur l'estimation des indices de Sobol' de premier ordre. Ces indices quantifient la contribution de chaque entrée, seule (indépendamment des autres entrées), à la variance de la sortie. En pratique, un estimateur basé sur le rang a été utilisé pour les estimer. Pour détecter plus de formes de dépendance entre les entrées et la sortie (notamment les interactions entre entrées), un autre outil d'analyse sensibilité reposant sur une mesure de dépendance basée sur un noyau, à savoir le critère d'indépendance de Hilbert Schmidt (HSIC), a été utilisé. Ces deux critères de sélection ont permit d'identifier un ensemble d'entrées ayant une influence considérée comme significative sur les sorties du code.

4. **Approximation des sorties par des modèles de régression (appelés métamodèles).** A partir de l'échantillon d'apprentissage, des métamodèles ont été construits pour ajuster les sorties du simulateur. Ces modèles sont destinés à remplacer le code initial dans les études de sensibilité. Pour cela, des modèles de type processus gaussien ont été utilisés. Ces modèles présentent plusieurs avantages par rapport aux autres modèles de régression, comme la possibilité d'évaluer l'incertitude des prédictions effectuées.

5. **Utilisation des modèles de régression (construits à l'étape 4) pour effectuer une analyse de sensibilité plus précise.** Une analyse de sensibilité a ensuite été réalisée en utilisant les modèles de régression construits précédemment. Ces métamodèles, par leur temps d'évaluation négligeable, permettent une exploration approfondie de l'espace d'entrée. Par conséquent, ils ont pu être utilisés pour estimer les indices de Sobol' de premier ordre et les indices totaux. Ces derniers quantifient l'effet total (seul et en interaction) de chaque entrée sur chaque sortie. Les informations obtenues sont donc beaucoup plus complètes que celles fournies par l'estimation des seuls indices de premier ordre effectuée à l'étape 3.

Mots clés: statistiques appliquées, accidents nucléaires graves, analyse sensibilité, simulations numériques, métamodélisation, apprentissage automatique.

Abstract

An important part of the electricity produced in France is supplied by nuclear reactors. Research in this field is therefore an important industrial challenge. The topic of this PhD thesis fits into this context and is the result of a collaboration between the CEA and the Paul Sabatier University (Toulouse, France).

The subject of this thesis is the study of an industrial code modeling a severe nuclear accident, namely the fuel-coolant interaction, and more precisely the treatment of uncertainties on the modeling parameters of this code. To achieve this goal, various statistical tools of the uncertainty quantification framework were used into a global methodology consisting in five main steps:

1. **Exploration of the input space using sampling methods.** Knowing the variation domain of the input variables, the input space was randomly sampled. A code run was then performed on each sampled point of the experimental design. The obtained sample of inputs/outputs constituted the learning sample. The goal is here to explore with this sample the input space and get as much information as possible about the behavior of the code output. Only having a limited budget for simulations, we used a near random sampling method, named Latin hypercube sampling (LHS) instead of a classic random sampling method. This method ensures a better space coverage of the marginal distributions while maintaining some properties of classical Monte-Carlo sampling. Asymptotic convergence of statistics used in our methodology under LHS were also discussed. In particular, a theoretical work on the asymptotic convergence of Z-estimators under LHS has been carried out.

2. **Sensitivity analysis of code failures.** Running the simulations of the experimental design defined in step 1 revealed that a significant portion of the code executions did not converge. We therefore analyzed these code failures to understand which of the inputs have the most influence on them.

3. **Selection of the inputs from sensitivity analysis of the outputs.** From the learning sample, the main influential inputs were identified using two sensitivity analysis tools. The first one is based on the estimation of first-order Sobol' indices. They quantify the contribution of each input, alone (independently from the other inputs), to the output variance. In practice, a rank based estimator is used to compute the indices. These indices identified the impact of the main effects of the variables. In addition, to detect a broader dependence between input and output (including interactions), another sensitivity tool relying on a kernel based dependency measure, namely the Hilbert Schmidt Independence Criterion, was used. From the analysis of Sobol' inidices and HSIC measures, a pool of significantly influential inputs were deduced.

4. **Emulation of the outputs by regression models (called metamodels).** From the learning sample, regression models were built to fit the simulator outputs. These models are intended to replace the initial code. For this, Gaussian Process (GP) regression method was used. This method has several advantages compared to other regression methods notably its ability to propose both a prediction and a quantification of the uncertainty on its predictions.

5. **Use of metamodels (built at Step 4) to perform a more complete sensitivity analysis.** A quantitative sensitivity analysis were then performed using the regression models instead of computer code. These metamodels, by their negligible evaluation time, allow an intensive exploration of the input space. This is made possible by the negligible evaluation time of the metamodels. Hence, they were used to estimate both first-order and total Sobol' indices. The latter quantify the effect of variables alone and in interaction with the other inputs. The information provided is therefore much more complete than that resulting from the previous estimation of the first-order indices alone performed in Step 3.

Key words: applied statistics, sensitivity analysis, numerical simulations, severe nuclear accidents, metamodeling, machine learning.

Contents

List of Figures

List of Tables

List of Acronyms

ANOVA: ANalysis Of Variance

ANR: french National Research Agency (*Agence Nationale de Recherche* in french)

CEA: french Alternative Energies and Atomic Energy Commission (*Commission à l'Energie atomique et aux Energies Alternatives* in french)

DoE: Design of experiments

EDF: Electricity of France (*Electricité de France* in french)

FCI: Fuel-Coolant Interaction

GP: Gaussian Process

GSA: Global Sensitivity Analysis

HSIC: Hilbert Schmidt Independence criterion

i.i.d: independent and identically distributed

ICSCREAM: Identification of penalizing Configurations using SCREening And Metamodel

IRSN: Radioprotection and Nuclear Safety Institute (*Institut de radioprotection et de sûreté nucléaire* in French)

KS: Kolmogorov-Smirnov

LDS: Low Discrepancy Sequences

LHS: Latin Hypercube Sampling

LOO: Leave-One-Out

LR: Linear Regression

LSA: Local Sensitivity Analysis

MASCOT-NUM: Stochastic Analysis Methods for Codes and Numerical Processing (*Méthodes d'Analyse Stochastique pour les COdes et Traitements Numériques* in french)

MC3D: MultiComponent Code 3D

ML: Maximum Likelihood

MSE: Mean Square Error

PWR: Pressurized Water Reactor

SRS: Simple Random Sampling

UQ: Uncertainty Quantification

Introduction

A major part of the electricity produced in France is supplied by nuclear reactors. Research in this field is therefore an important industrial challenge. This work is within this context and is the result of a collaboration between the CEA, a major actor in nuclear research, and the Paul Sabatier University (Toulouse, France).

The subject of this thesis is the study of an industrial code modeling a severe nuclear accident, namely the fuel-coolant interaction, and more precisely the treatment of uncertainties on the modeling parameters of this code. To achieve this goal, various statistical tools of the uncertainty quantification framework are used. Since the industrial context is a crucial factor in the methodological decisions made, it is important to detail it. Therefore, a complete description of the fuel-coolant interaction and its modeling by the simulation code MC3D is first provided in Chapter 1. The operational objective of this work is then presented. Finally, the framework of uncertainty quantification and the methodology proposed to meet these operational objectives are introduced. The next three chapters of the manuscript are devoted to the five main steps of the proposed methodology:

1. Exploration of the input space using sampling methods, in Chapter 2.
2. Sensitivity analysis of code failures to understand which inputs have the most influence on them, in Chapter 3.
3. sélection of significant inputs and global sensitivity analysis of the code outputs of interest, in Chapter 4.
4. Emulation of the outputs by regression models (metamodeling), in Chapter 4.
5. Use of metamodels (built at Step 4) to perform a more complete sensitivity analysis, in Chapter 4.

The deployment of statistical tools to study this code has required the automated launch of many code simulations. As the MC3D code is not initially designed for this, several algorithms in *shell* and *R* have been written during this thesis to fulfill this task. These developments being purely computational and far from statistical and methodological contributions, they are not detailed in this manuscript.

There has been several communications regarding the work presented in this manuscript. Chapter 3 of this manuscript is the subject of an article accepted by the Nuclear Science and Engineering journal. It is also planned to submit a second paper on the convergence of Z-estimators under the Latin Hypercube Sampling method (work detailed in Chapter 2), by early 2023. The main communications (preprints and conferences) are listed below.

Preprints

Hakimi, F., Brayer, C., Marrel, A., Gamboa, F., and Habert, B. (2022), *Statistical methods for the study of computer experiments failures: Application to a fuel-coolant interaction simulation code*, accepted in Nuclear Science and Engineering.

Conferences

Hakimi, F., Brayer, C., Marrel, A., Gamboa, F., and Habert, B. (2022), *High Dimensional Sensitivity Analysis Method for a Computationally Expensive Code*. In SIAM Conference on Uncertainty Quantification (UQ22), Atlanta, USA.

Hakimi, F., Brayer, C., Marrel, A., Gamboa, F., and Habert, B. (2022), *Poster: Statistical methods for the study of computer experiments failures: application to a fuel-coolant interaction simulation code*. In GDR Mascot Num 2022, Clermont Ferrand, France.

Chapter 1

Context, objectives and methodology

In France, around 70% of the electricity consumed in a year is ensured by the nuclear energy (International Energy Agency, 2021). This represents 56 reactors currently in operation. The construction of six new reactors has also been announced for the coming years. Research in nuclear energy is so a major challenge for French electricity production. In France, an important part of the research on this subject is done at the CEA (French Alternative Energies and Atomic Energy Commission).

Almost all existing reactors use the energy released by a fission reaction of a heavy uranium 235 atom by a neutron. This reaction is highly exothermic and can produce new neutrons, leading under certain conditions to a chain reaction. The energy produced is converted into heat by a coolant. All french nuclear power plants are Pressurized Water Reactor (PWR). For this kind of reactor, the coolant is water maintained under high pressure (155 bars) for a temperature that can reach $350°C$. This hot water flows in the core of the reactor in a closed circuit, called primary circuit, and exchanges thermal energy with a second closed water circuit. This second circuit generates steam that is used to drive turbines connected to generators that produce electricity. A third open water circuit cools the secondary circuit to condense the steam after it passes through the turbines. Figure 1.1 depicts the operation of a pressurized water reactor.

Many safety systems have been developed and implemented to allow the proper operation of these nuclear reactors and thus avoid accidental situations or limit their effects. In addition to the basic components mentioned above, reactors are equipped with several independent and redundant safety systems. These systems

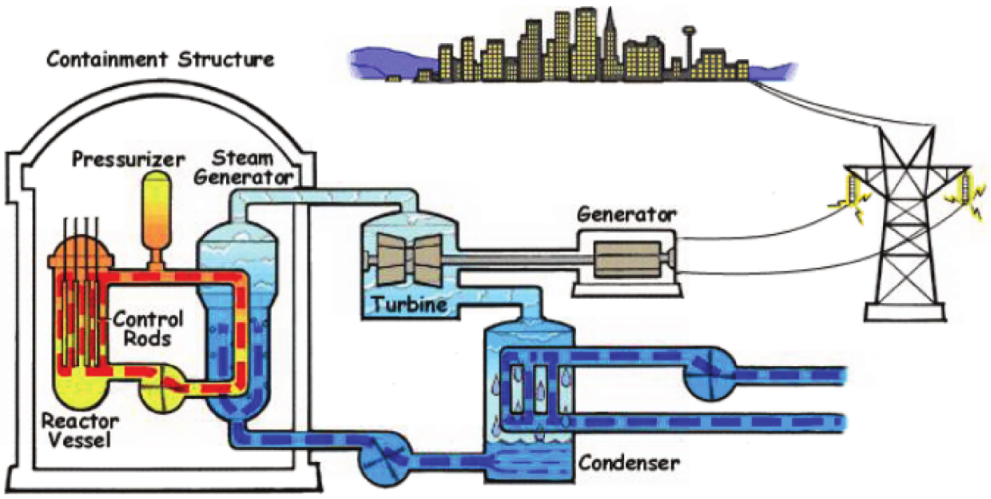

Figure 1.1: *Schematic diagram of the operating principle of a pressurized water nuclear reactor. From wikipedia: Pressurized water reactor.*

operate in parallel to mitigate the risks associated with malfunctions or operational errors. They also prevent such situations from escalating into more important accidents.

However, the effectiveness of any safety system depends on the accident for which it is designed. Accident sequences that are outside the scope of existing safety devices may occur. Indeed, several accidents of this type happened in history. We can mention in particular the accidents of Three Mile Island (USA, 1979), Chernobyl (Ukraine, 1986) and more recently Fukushima Daiichi (Japan, 2011). These events have stressed the need to adopt measures to mitigate the occurrence of scenarios that could lead to what is called *severe accidents*.

An accident is deemed *severe* when the energy released by the core of the reactor is greater than the energy dissipated by the coolant and when safety systems fail to cool down the system. In this case, the core temperature increases and can reach its melting temperature leading to the loss of core integrity. The hot magma (around 3000 K) composed of molten elements of the core and of the structures thus generated is called by the generic name *corium*. This magma may flow down, out of the core, to form a pool in the lower plenum of the reactor pressure vessel. Due to the high temperature of the corium, the vessel wall is ablated by the pool. If the vessel is breached, the corium can then relocate in the reactor pit, reach the basemat and interact with its concrete. The corium also interacts with the water that may be present. This interaction is called the fuel-coolant interaction (FCI). Figure 1.2 depicts these different stages of corium progression in the reactor.

The study of the FCI is the industrial motivation of this work. This chapter aims at presenting this industrial context. We also discuss on how to address industrial stakes of this work. Adapted statistical methods proposed to give some answers to these industrial problems are then presented.

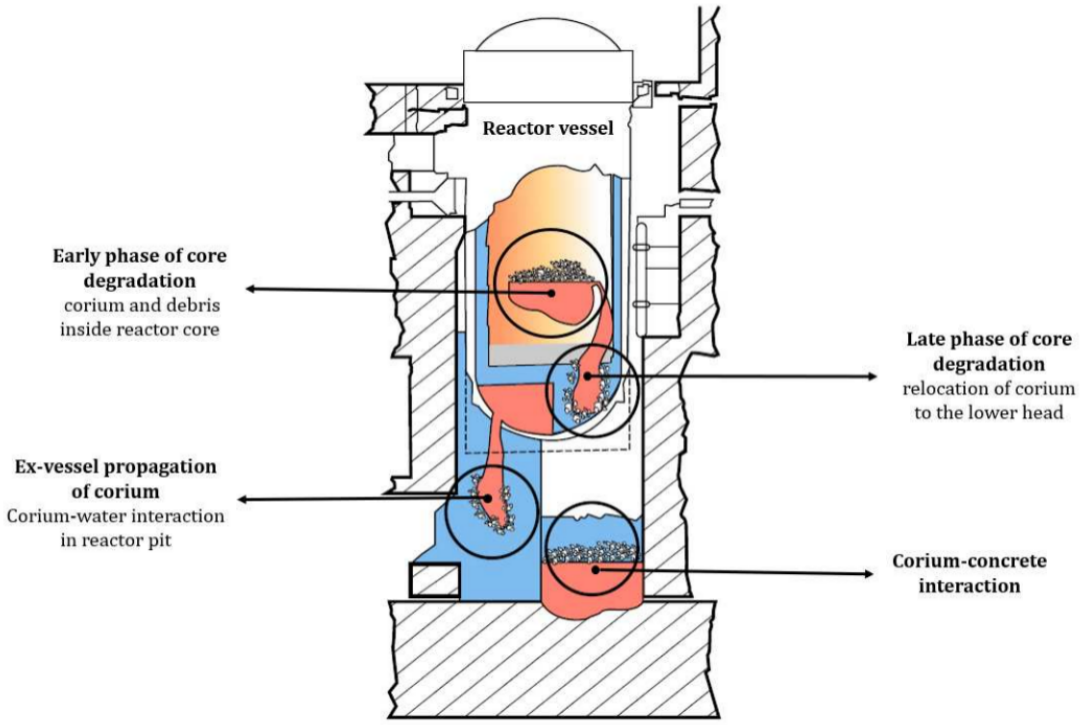

Figure 1.2: *Stages of corium progression during a severe accident in a pressurized water reactor.*

1.1 Industrial context: study of the fuel-coolant interaction (FCI)

The corium-water interaction is one of the important phenomena that can occur during a severe accident. Improving the understanding of this phenomenon is a major issue for nuclear safety.

As for most physical phenomena, two types of tools are available to handle the FCI. On one hand, some experimental facilities are available. They allow direct observation of the phenomenon and its consequences. However, experimental facilities are generally not fully representative of a reactor case. In fact, they are often smaller than an actual nuclear reactor and the materials used are not identical. They can also be expensive to operate: up to several million euros for an experiment simulating corium-water interaction with prototypical materials. Finally, the type and amount of experimental data that can be collected is limited. On the other hand, we have simulation softwares to model and predict physical phenomenon. They allow the process of much more data. However, it is important to make sure that the modeling they provide is faithful to the simulated phenomenon.

This section aims at presenting the FCI phenomenon and the way it is studied. The experimental tool (KROTOS) and the numerical tool (MC3D: Multicomponent Code 3D) used to study this phenomenon at the CEA will also be presented.

1.1.1 Description of the phenomenon

As stated before, the fuel-coolant interaction in a PWR severe accident context corresponds to the contact between the hot corium and the surrounding cooling water. This interaction can lead, under certain conditions, to a fine fragmentation of the corium leading to a violent vaporization of the coolant and the propagation of a pressure wave. This phenomenon, called steam explosion, may threaten the reactor integrity and lead to the dispersion of radio-elements in the environment.

The main steps of this kind of phenomenon, as described in details by Corradini (1988), are the following (cf. Figure 1.3):

- fuel-coolant premixing;
- steam explosion triggering;
- shock wave propagation;
- vapor expansion.

Let us describe the different phases of this phenomenon.

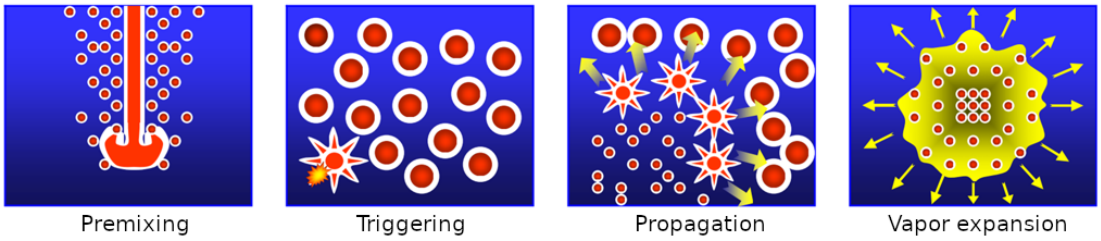

Figure 1.3: *Main steps of steam explosion (Hansson, 2007).*

1.1.1.1 Fuel-coolant premixing

In pressurized water reactor conditions, the fuel-coolant premixing can be defined as the phase where the molten fuel jet breaks up in the coolant into droplets of a few millimeters in diameter (cf. Figure 1.4). Under these conditions, the fragmentation is mainly due to two types of instabilities. The first one is the Kelvin–Helmholtz instability (Helmholtz, 1868; Kelvin, 1871) that develops along the jet. It is a fluid instability that occurs when there is a velocity difference across the interface between two fluids, here fuel jet and the surrounding water. The other one is the Rayleigh-Taylor instability (Rayleigh, 1883), occurring at the jet front. This instability is due to the density difference between corium and water, when the heavier fluid is pushing the lighter fluid. During the fragmentation, the more droplets there are, the larger the fuel-coolant exchange surface is and the greater the chance of a vapor explosion occurs.

During this phase, the corium temperature is high enough to have the coolant in a film boiling regime around the corium jet and droplets. This means that there is a vapor film surrounding the molten corium that separates the two liquids.

1.1.1.2 Steam explosion triggering

Steam explosion triggering is due to the local destabilization and collapse of the vapor film surrounding the melted corium drops. It has two main consequences (Fletcher, 1995):

- Some liquid-liquid contact between fuel and water can occur, enhancing the heat exchanges.
- A fine fragmentation of the corium takes place, leading to an increase of the exchange surface between fuel and coolant.

Both phenomena drastically increase the heat transferred by the molten fuel. These heats transfers reach such intense rapid level that the time scale for heat transfer

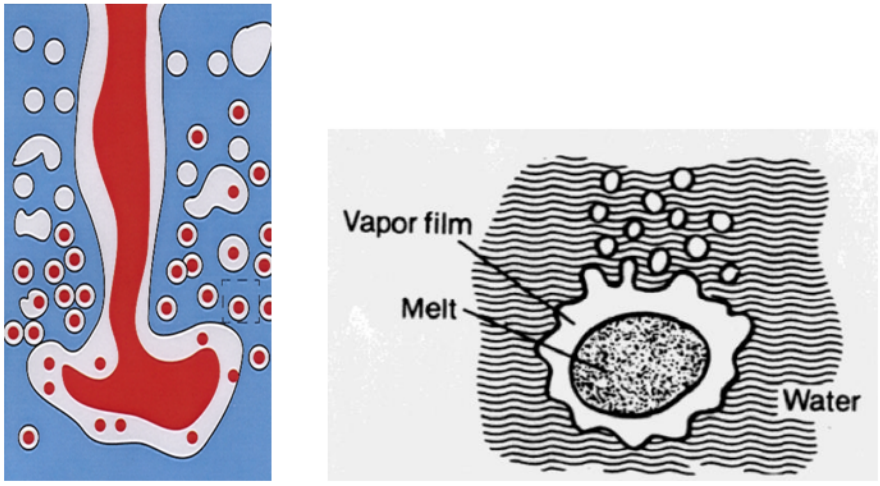

Figure 1.4: *Corium jet overview on the left. Zoom on a molten fuel droplet on the right.*

is shorter than the time scale for pressure relief. This may lead to the initialization of a pressure wave.

Triggering can be spontaneous, meaning that the film-collapse process begins naturally due to local conditions (Corradini et al., 1988). It can also be externally triggered, by a pressure wave or the impact of the jet on structures for example. In experimental conditions, this triggering is manually proceeded in order to ensure the observation of the steam explosion.

Some conditions can make steam explosions more difficult to trigger (Corradini et al., 1988), because of their stabilizing effects on the vapor film:

- ambient pressure in the coolant is high;
- the coolant temperature is close to its saturation temperature (T_{sat});
- the surface temperature of the molten fuel (T_W) is high compared to the saturation temperature of the coolant;
- presence of non condensable gas in the vapor film (eg. H_2 coming from water hydrolysis during corium oxidation).

Figure 1.5 shows, on the left, the experimental results of the temperature conditions to have the spontaneous explosion of a tin/water mixture (Dullforce et al., 1976), and on the right side, the trigger pressure necessary to initiate an explosion in a Fe_2O_3/water mixture (Nelson and Duda, 1985).

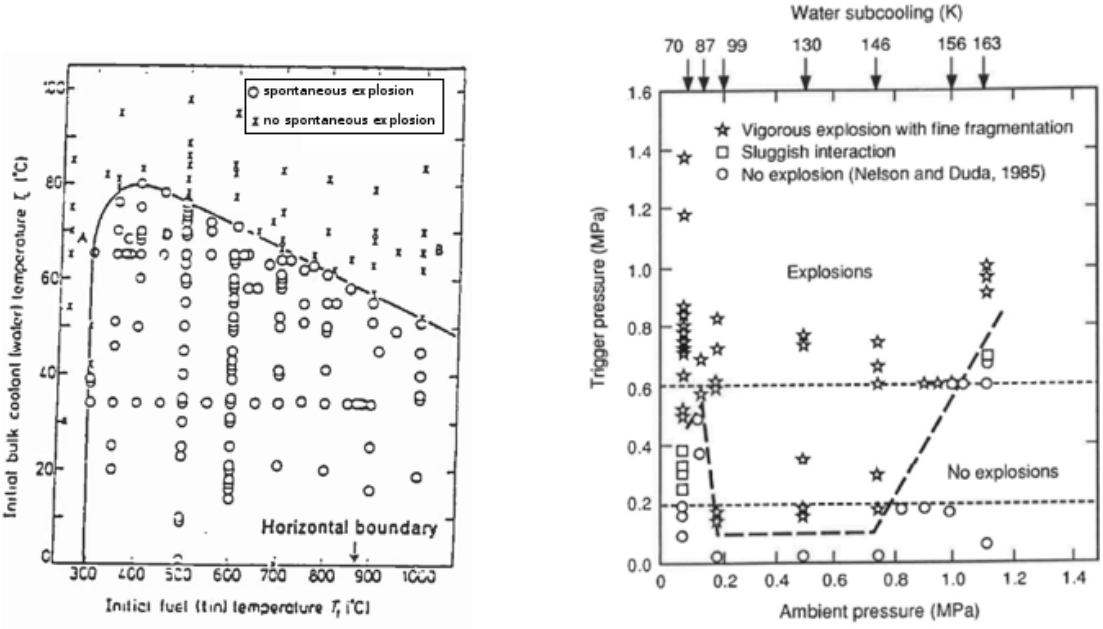

Figure 1.5: *Temperature interaction zone for 12g of molten tin dropped into water on the left (Dullforce et al., 1976). Mandatory trigger pressure depending on ambient pressure for Fe_2O_3 drops on the right (Nelson and Duda, 1985).*

1.1.1.3 Shock wave propagation and vapor expansion

The pressure peak generated by the local destabilization of the vapor gradually propagates in the fuel coolant mixture destabilizing vapor films and fragmenting them into fluid droplets. This may lead to new coolant pressurization cycles. Hence, the shock wave is propagating. The pressure wave is thus maintained by the fragmentation of the fuel as it passes through (cf. Figure 1.6).

An analogy can be made between a chemical explosion and this shock wave propagation. Indeed, both follow a pressure wave behind which a subsonic reaction zone exists. The detonation model of Von Neumann has therefore been used to model the explosion propagation (Board et al., 1975 ; Scott, 1978). But its analogy has some limitations:

- the fuel-coolant mixing is heterogeneous, and the two liquids are in-miscible which has an impact on the sound speed computation and the Von Neumann pressure spike;
- the corium fragmentation might be incomplete, which has an impact on the mass of corium involved in the explosion and on the wave propagation.

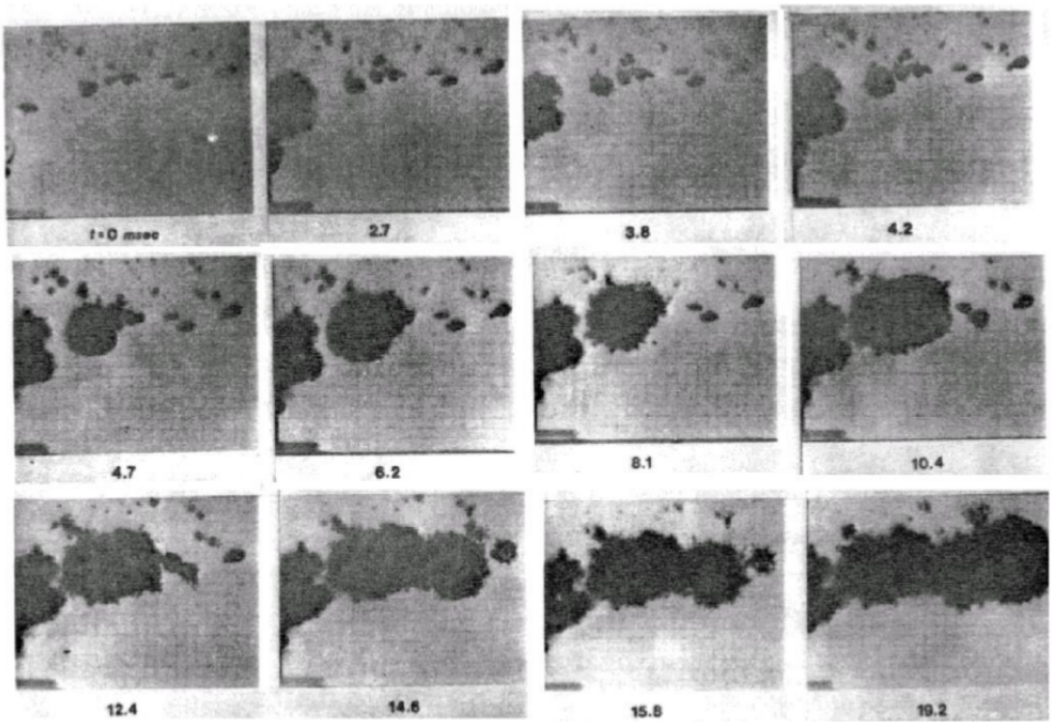

Figure 1.6: *Propagation of a steam explosion in a mixture of molten tin drops and water (Ciccarelli and Frost, 1988).*

These limitations are especially present in reactor conditions where the fuel-coolant mixing is done around the corium jet.

The pressure drop following the passage of the explosion allows vapor generation and expansion. In addition, when the pressure wave reaches the free surface, an expansion wave propagates in the opposite direction, decreasing the pressure. It then enables the vapor generation and expansion.

1.1.2 The KROTOS experimental facility

The KROTOS installation allows to observe a real fuel-coolant interaction involving prototypic corium (same chemical composition as real corium but with depleted uranium). Formerly located at the JRC Ispra (Huhtiniemi et al., 1999 ; Huhtiniemi and Magallon, 2001), the facility has been moved to the CEA Cararache in the early 2000s. At Cadarache, an X-ray radioscopy system has been added to characterize the corium-water mixture during the premixing phase.

This installation is a seven meters high experimental facility built on two levels of the plinius experimental platform located at CEA Cadarache (Brayer et al., 2012). It is composed of 2 main elements:

- a resistive furnace to heat the fuel;
- a test vessel housing the corium jet formation device, the test tube, the trigger and caption tools.

The main measured quantities of the installation and their associated sensors for this experiment are summarized in Table B.1.

Other quantities can be deduced from the aforementioned observables. For instance, we can mention the impulsion of the explosion corresponding to the integral of the pressure over the explosion time. This quantity can be deduced using the measured explosion pressure. Drop properties (mass, size, etc.) during the premixing and the explosion can also be estimated using the collected corium fragments. For more details on the KROTOS installation and KROTOS experiments, please refer to (Brayer et al., 2012 ; Bouyer et al., 2015).

In the context of this work, we mainly focus on the KROTOS experiment KS4. Figure 1.7 locates all of the sensors along the test section of the facility for the KS4 experiment.

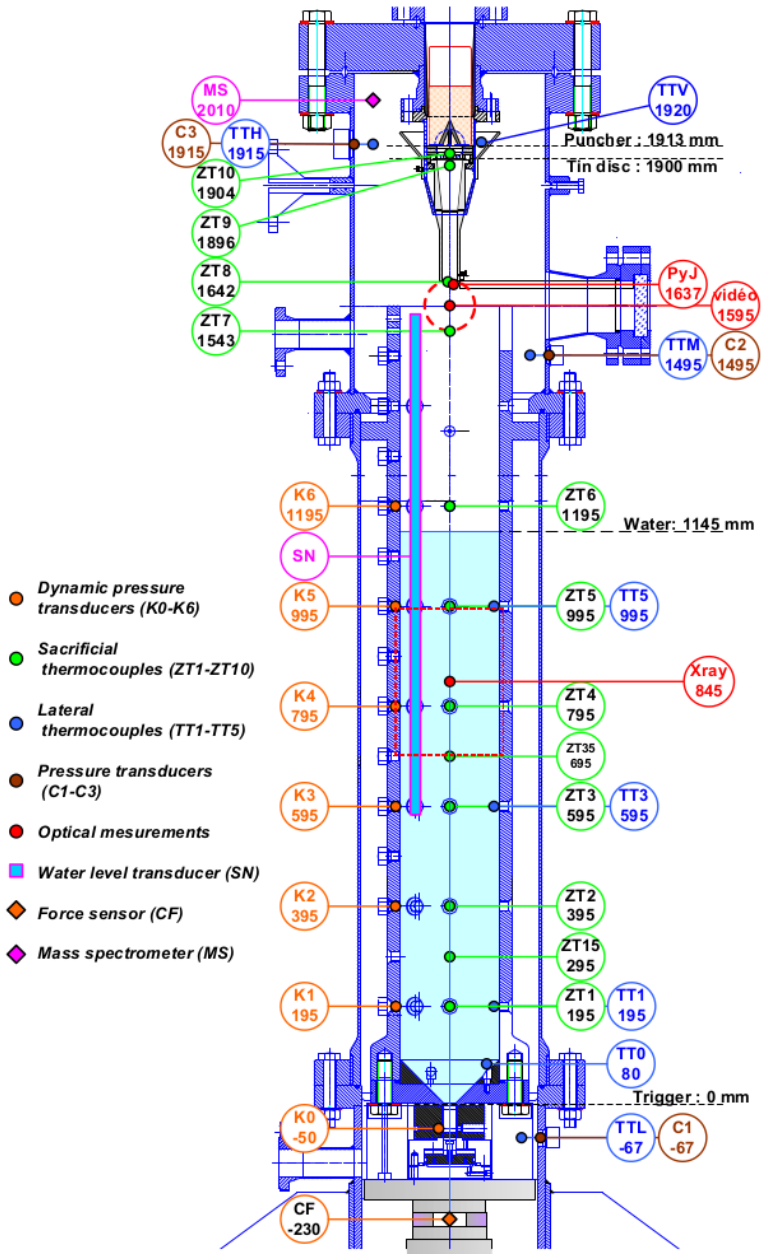

Figure 1.7: *Test section instrumentation of the KROTOS installation for KS-4* (Grishchenko et al., 2011).

Table 1.1: *Main measured quantities on the KROTOS facility.*

Measured quantity	Caption method	Comments
Shape of the jet (continuous/fragmented), jet diameter and Velocity	Video camera	only on reduced view window
Premixing global visualisation	X-ray radioscopy	only on reduced view window
Located explosion pressure	Dynamic pressure transducers	7 sensors distributed along the wall of the test tube, with 1 dedicated to the trigger
Melt front detection	Sacrificial thermocouples	5 sensors on the axis of the test section
Fragment properties (mass,size, etc.)	Collected at the end of the experiment	None
Water level	Water level transducer	None
Water temperature	Thermocouple	None

1.1.3 The MC3D simulation code

Several simulation codes have been developed around the world to model corium-water interaction (Meignen and Magallon, 2005). Initially MC3D was developed by the CEA (Berthoud and Valette, 1994 ; Brayer and Berthoud, 1997). The code is now owned by the IRSN who is pursuing its maintenance and development (Meignen, Picchi, et al., 2014 ; Meignen, Raverdy, et al., 2014). The developments of the last versions were mainly carried out in the context of the multi-year RSNR-ICE project led by IRSN, responding to the so-called *post-Fukushima* project. It involves the CEA and is co-financed by EDF, Framatome and the ANR (french National Research Agency). The version 3.09 of MC3D has been used for this work. A new version of the code, the 3.10, has been released in 2022.

MC3D is an Eulerian multiphase multicomponent computer code devoted to the simulation of corium water interactions. It uses a Cartesian or cylindrical meshing in 2D or 3D. It is organized around a common kernel and modules to model each component. The kernel manages the numerical scheme, the solver and the input/output problem. Each module describes, for a model component, the equations of mass, momentum and energy and the closure equations necessary to describe the model.

MC3D models the FCI through 2 applications. The first one, PREMIX, allows the assessment of the premixing step. However, this application is general enough to be used in other fields. The second application, EXPLO, is restricted in its use to

the study of the steam explosion phase. A complete calculation is thus performed in two steps, a premixing calculation followed by an explosion using as input the output data of the premixing calculation.

1.1.3.1 Format of the inputs and outputs of the code

All the inputs and outputs are defined in the MC3D datafile.

The inputs

In MC3D, the inputs are scalar or categorical. Nevertheless, we can distinguish several types of inputs regarding their role in the simulation process. First we have the geometrical and physical conditions. They allow to describe the situation that we wish to simulate.

Then, we have the numerical parameters: time step, grid size, etc. They influe on the precision of the simulation, but also on its duration and stability. Then we have the modeling parameters. They influe on the behavior of the models implemented on MC3D. In this work, we are interested in this type of inputs. More details are given below.

The outputs

There are two types of outputs:

- time evolution of physical quantities averaged over the whole mesh or specifically located in the calculation domain (to emulate a sensor for instance). They are called *type 1* outputs.
- representation of physical quantities on each cell of the domain at different moments of the numerical experiment. They are called *type 2* outputs.

The direct comparison of *type 2* outputs to experimental results is more difficult because these results are given by located sensor (except for the video film and the X-Ray radioscopy). We don't have access to the evolution of the observed quantities on each location of the domain.

1.1.3.2 Models used by MC3D

For the record, MC3D uses to model the fuel-coolant interaction the classical Eulerian multicomponent method where each component/mixture is described by its mass, momentum and energy equations. These balance equations are solved with an adapted version of the implicit continuous-fluid Eulerian numerical method (Harlow and Amsden, 1975 ; Mercier, 1989). A staggered grid is used (Cartesian or cylindrical) where velocities are expressed at the faces and other variables at the center of cells. In addition, several models are used to describe specific phenomena

of corium-water interaction. This section aims to present the models used in both premixing and explosion applications.

The main specific phenomena modeled in MC3D are the coolant and the fuel flow description, the melt fragmentation, the heat transfers and drop solidification.

Premixing modeling

In the premixing application, MC3D focuses on the jet fragmentation and mixing. Mass, momentum and energy equations are solved for the following components: liquid water, steam and non condensable gas mixture, corium in a continuous phase (corium jet) and the fragmented corium. In addition, physical models are associated with the various phenomena involved in the FCI.

Fuel flow description and fragmentation

The fuel is modeled by a continuous jet breaking into drops. Figure 1.8 depicts this modeling. Two jet fragmentation models are available in MC3D. The first one, *CONST*, assumes a constant jet fragmentation rate. The second one, *KHELMOLTZ*, is based on the Kelvin-Helmholtz theory for tangential instability (Kelvin, 1871 ; Helmholtz, 1868).

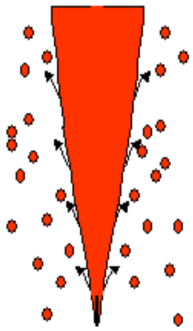

Figure 1.8: *Corium flow modeling in MC3D (Meignen and Magallon, 2005).*

The generated drops can also be fragmented by the surrounding medium (water-gas mixture) (Berthoud, 1996). To model this phenomenon, the code uses a correlation derived from Pilch (1981).

Drops solidification

Drops solidification is an essential mechanism for the reduction of the mass of corium that may be involved in the steam explosion. This phenomenon may therefore reduce the loads of an explosion. The standard approach in MC3D is to

compare the drop temperature to the melt solidification temperature. Below this temperature, the drop is assumed solid and there is no more fragmentation. As it is a purely parametric approach, no hypothesis on the physic of solidification is used.

Coolant flow description

Another model handles the description of the coolant flow (water) and its interaction with the discrete fuel particles. First of all, only the local volume of water (gaz or liquid) is considered. Further, MC3D considers three different coolant flows: the droplet flow, the bubbly flow and a transitional flow between the aforementioned states. Figure 1.9 depicts these different coolant flows.

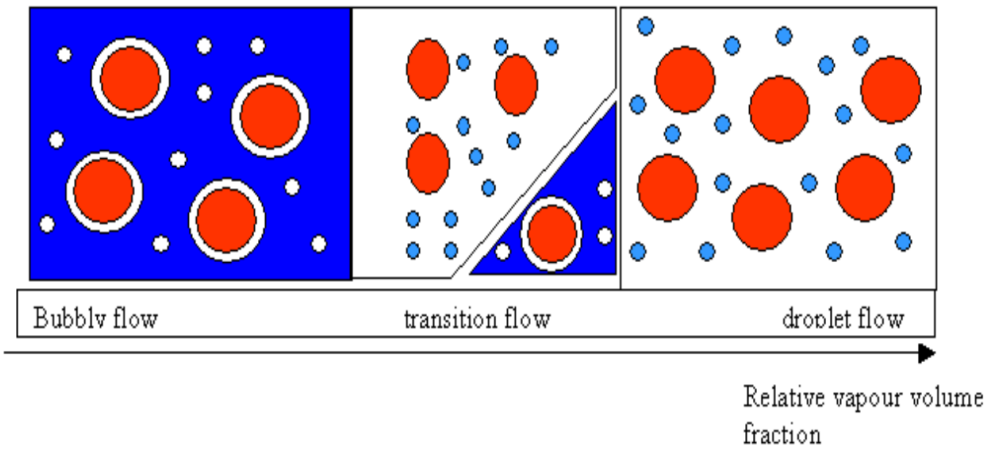

Figure 1.9: *The three regimes of the coolant flow map in MC3D (Meignen and Magallon, 2005).*

Heat and mass transfer

Heat transfers from the fuel to the coolant and the subsequent coolant mass transfers are very important phenomena to model. The two major mechanisms driving those phenomena are film boiling and thermal radiation. In MC3D, the film boiling around fuel drops is modeled through a correlation law derived from Epstein-Hauser correlation (Epstein and Hauser, 1980). Concerning radiation, only in-cell radiation heat transfers are modeled.

Water phase transitions

Mass transfers between liquid water and steam is derived from the energy balance at the water steam interface, assumed to be at saturation temperature.

Corium oxidation

Corium temperature is so high that it may hydrolyze the surrounding water. The chemical reaction generates H_2. It also adds energy to the mixture as the reaction is exothermic. Here, the dioxygene O_2 is the oxidant of the reaction. The modeling of this phenomenon was still under development at the beginning of the thesis. It is now included in the latest version of MC3D (V3.10).

Steam explosion modeling

The basic aspects of the flow patterns established for the PREMIX application is conserved for the explosion phase. However, we have the important following differences:

- Differences in the handling of the fragment field;
- No continuous fuel jet: the eventual jet remaining at the end of the premixing step is transferred into fuel drops.

Moreover, to handle specific phenomena happening during steam explosion, new models are added. The main one is the drop fine fragmentation.

Drop fine fragmentation

The drop fine fragmentation corresponds to the fragmentation into fine drops (diameter of around $50 - 100\mu m$) of the bigger ones (between 1 and $10mm$ diameter) previously generated. It is initiated by a parametric model. The considered driving mechanisms of the fine fragmentation is hydrodynamic. As for the large droplets, the Pilch correlation (Pilch, 1981) is used to calculate the characteristic time for fragmentation.

1.2 Objectives and proposed methodology

In Section 1.1, we described the fuel-coolant interaction, a phenomenon that can happen in a severe nuclear accident sequence. Further, we presented the experimental installation used at CEA to observe this phenomenon. Finally, a global overview of the IRSN software that simulates this phenomenon, MC3D, has been described.

As described, many models interact with each other in MC3D to simulate the corium-water interaction. A set of parameters is required to built each of these models. These modeling parameters are calibrated independently via dedicated experiments. We can cite for example the TREPAM experiment for the study of film boiling (Berthoud and Gros D'Aillon, 2009) and DROPSG or GALAD for drops fragmentation (Achour, 2017 ; Malmazet, 2009). However, these studies

do not take into account the interactions between models. There is therefore epistemic uncertainties concerning these parameters. To account for these model interactions, experimental programs such as KROTOS have been developed. They allow the observation of the whole Fuel Coolant Interaction.

In this context, the main objective of this thesis is to study the simulation of the FCI in the KROTOS facility by MC3D. In particular, we will be interested in the evaluation of the influence of the different modeling parameters and their interactions on the outputs of the MC3D code. This work also prepares the calibration of these parameters on KROTOS experiments in order to improve the reliability of the code.

To meet these objectives, we propose to use some advanced statistical methods within the general framework of uncertainty quantification. The first part of this section aims to give an overview of this framework. Then, we introduce the formalism used in our manuscript. In addition, we provide a description of the studied MC3D configuration and variables. Finally, we present the methodology proposed in this work to achieve our operational goals.

1.2.1 The uncertainty quantification framework

The physical models that describe the process of a severe nuclear accident such as FCI are complex and represented by deterministic equations. They lead, during the numerical modeling phase, to the development of numerical simulation codes.

These software tools take many input parameters characterizing the studied phenomenon. These parameters can be subject to uncertainty according to the degree of knowledge and characterization of the phenomenon. Generally speaking, two main sources make up uncertainty:

- one due to a lack of information (epistemic uncertainty);
- one that are inherent to the random nature of the modeling parameters (stochastic uncertainty).

Whatever their nature, it is important to quantify these uncertainties and study their impact on the code outputs. This analysis allows to validate the mathematical, physical or numerical model, to guide characterization efforts on the most influential parameters and more generally to better understand the modeled phenomenon.

To take into account the uncertainties of the inputs of numerical simulators, a general approach has been developed for more than ten years (Rocquigny, 2008) and remains an active research topic. We can mention in particular the CNRS Research Group MASCOT-Num (French acronym of Méthodes d'Analyse Stochastique pour

les COdes et Traitements Numériques)[1] which brings together most of the academic and technological actors such as the CEA, around the development of stochastic approaches for the analysis of numerical simulators.

In this framework, the usual modeling methodological approach can be divided into several important steps, as depicted in Figure 1.10.

The step A of problem specification consists in defining the system to be studied (model, simulator or measurement process), identifying the uncertain or fixed inputs, as well as the quantities of interest to be studied. These quantities are derived from the output variables of the model. The simulator can be assimilated to a function linking inputs to outputs. See equation (1.1) below.

$$f : \begin{array}{l} \mathcal{X} \to \mathcal{Y} \\ \boldsymbol{X} \longmapsto f(\boldsymbol{X}, \boldsymbol{c}) = \boldsymbol{Y} \end{array} \tag{1.1}$$

Here,

- $\boldsymbol{X}$ is the vector of uncertain inputs evolving in a measurable space $\mathcal{X}$, $d \in \mathbb{N}^*$ its dimension ;
- $\boldsymbol{c} \in \mathbb{R}^{d_c}$ is the vector of fixed inputs considered deterministic and $d_c \in \mathbb{N}^*$ its dimension ;
- f is measurable function representing the model that links the inputs to the outputs;
- $\boldsymbol{Y}$ is the output of interest evolving in a measurable space $\mathcal{Y} \subset \mathbb{R}^p, p \in \mathbb{N}^*$.

The second step B consists in quantifying the inputs uncertainty. In the probabilistic framework, the uncertainties of the random input variables are modeled by fully or partially known probability distributions (Helton, 1997). These probability distributions can be chosen by using any available data and estimating the parameters of the distributions on these data, by formalizing the expert opinion (elicitation) or by using bibliographical data.

The uncertainty quantification is supported by a calibration step, the step B'. In this context, this step aims to characterize the uncertainty on the estimated parameters defining the physical model. Indeed, some parameters are not directly measurable, but must be estimated in a regression framework. This step can also integrate the detection and estimation of a model bias term. This bias takes into account the fact that the model is not an absolutely faithful representation of reality. Classically, regression methods based on maximum likelihood are used

[1] website: http://www.gdr-mascotnum.fr/

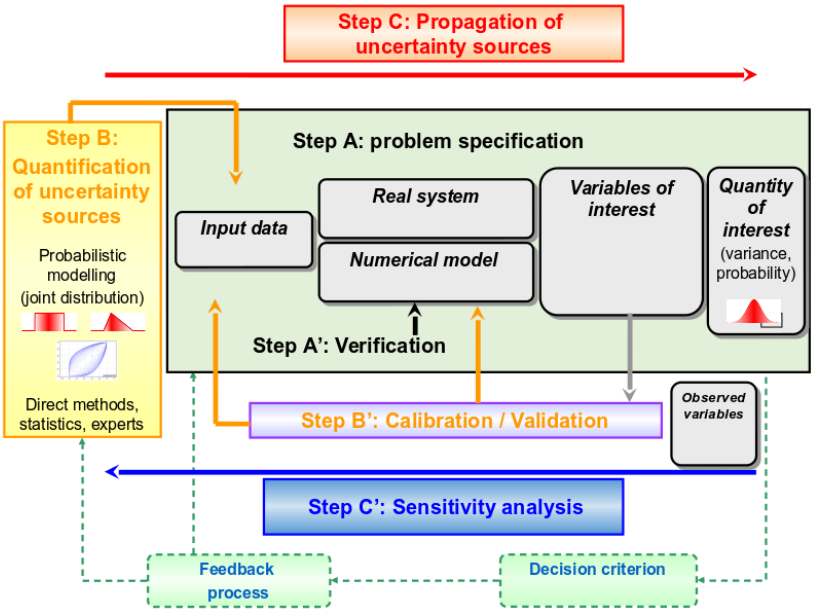

Figure 1.10: *General scheme for the methodology of uncertainty treatment, adapted from De Rocquigny et al. (2008).*

to perform this calibration step (Trucano et al., 2006). More recently, Bayesian calibration methods by data assimilation have also been introduced (Carmassi, 2018). This framework allows us to consider the *a priori* information about the parameters to be calibrated and their uncertainties. In addition, the parameters calibrated using Bayesian methods are associated with a probability distribution that quantifies their post-calibration uncertainties.

The purpose of the step C is to measure how the input variables uncertainty are propagating through the studied model. The aim of this step is to quantify the impact of the input uncertainties on the outputs predicted by the model. More precisely, it mesures the effect on the quantities of interest associated with the outputs. These quantities of interest depend on the objectives of the study. It can be for example a measure of the output mean or dispersion, a quantile or a probability to exceed a critical value (Cannamela, 2007).

In parallel and complement of the uncertainty propagation, a global sensitivity analysis (GSA), the step C' of the approach, is also important to carried out. Sensitivity analysis aims to determine how the variability of the input parameters affects the value of the output or the quantity of interest (Da Veiga et al., 2021; Iooss and Lemaître, 2015; Saltelli et al., 2004). It thus allows to identify and perhaps quantify, for each input parameter or group of parameters, its contribution to the variability of the output.

In support of these major steps, two cross-cutting themes are addressed as part of the process of dealing with uncertainties.

- **Numerical experiments design**: it consists in choosing where to carry out the simulations to sample the input modeled distributions in order to maximize the information collected in a minimum of (numerical) experiments. To do so, many approaches are possible, from simple random draws (Monte Carlo type) to more elaborate constructions (Fang et al., 2005), such as Latin Hypercube Samplings (Mckay et al., 1979) for example.
- **The approximation of the simulator by a regression model**: from simulated observations defined by a design of experiments, a statistical learning model is constructed. These simplified models are sometimes called a metamodel. They approximate as closely as possible the simulator under study and require negligible calculation time. These metamodels are then used to predict the outputs of the simulation code for any set of input values and thus allow a very intensive exploration of the model.

This methodological context adequately meets the objectives of the thesis. Indeed, the tools of global sensitivity analysis can be used to understand the influence of different modeling parameters on the outputs of the MC3D code. In addition, the calibration methods presented above will allow to robustly adjust the modeling parameters of the code to experimental data in a future work.

1.2.2 Formalism and notations

In order to fit in the framework of uncertainty quantification, it is first necessary to present our application case in an adapted formalism. In the following, the model functions of the two applications of the code, PREMIX and EXPLO, will be denoted by $f_{[1]}$ and $f_{[2]}$ respectively.

We distinguish two categories of inputs. First there are the inputs we want to study and thus considered as uncertain for the model functions $f_{[1]}$ and $f_{[2]}$. We denote by $\boldsymbol{X}_{[1]}$ the inputs of the code 1 and $\boldsymbol{X}_{[2]}$ the inputs of the code 2. Inputs common to codes 1 and 2 are denoted by $\boldsymbol{X}_{[1,2]}$. Then, there are the inputs considered as fixed. Similarly, we denote by $\boldsymbol{c}_{[1]}$, $\boldsymbol{c}_{[2]}$ and $\boldsymbol{c}_{[1,2]}$ the fixed inputs respectively associated to the model function $f_{[1]}$, the model function $f_{[2]}$ and common to the two model functions. We also denote by $\boldsymbol{X}$ the vector containing all the inputs of the MC3D code.

Finally, the *type 1* output for the model $f_{[i]}$ will be designated by $\boldsymbol{Y}^{I}_{[i]}$ and the *type 2* outputs will be designated by $\boldsymbol{Y}^{II}_{[i]}$. Figure 1.11 summarizes this formalism and presents the links between the inputs, the codes and the outputs.

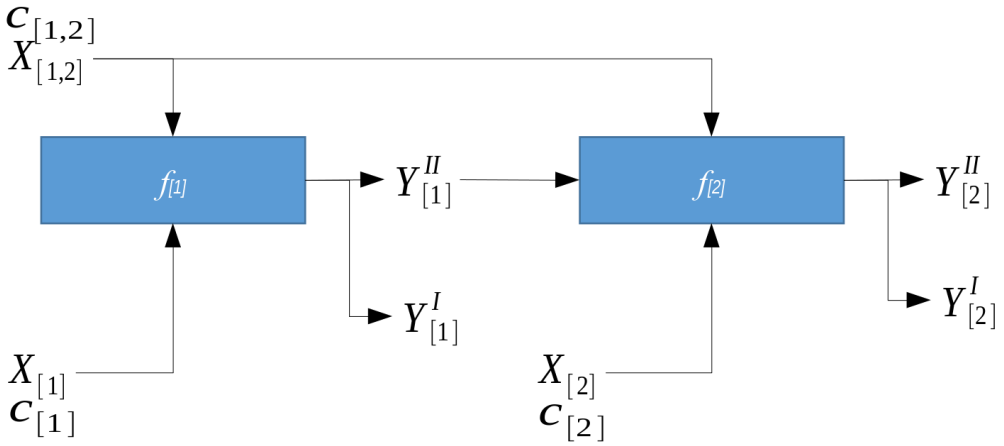

Figure 1.11: *Diagram summarizing the formalism used for this work.*

1.2.3 The application case

Let us give specifications on the application case: the MC3D configuration for this work, the chosen inputs, their variation domain and the studied outputs.

1.2.3.1 MC3D configuration

In this work, we focus on the simulation by MC3D of the KS4 experiment. Thus, the spacial mesh used for simulations corresponds to the facility KROTOS.

Figure 1.12 depicts the used spacial mesh at the beginning of the simulation. It is 2D axisymmetric with $27R$ axis meshes and $103Z$ axis meshes. Working in 2 dimensions limits the calculation time. The global geometry describes the test section of the KROTOS facility. In this figure, the corium pool in the crucible is represented in red and the water pool in the test tube is represented in blue. The gray areas correspond to solid structures (crucible and jet formation device, or conical bottom of the test tube).

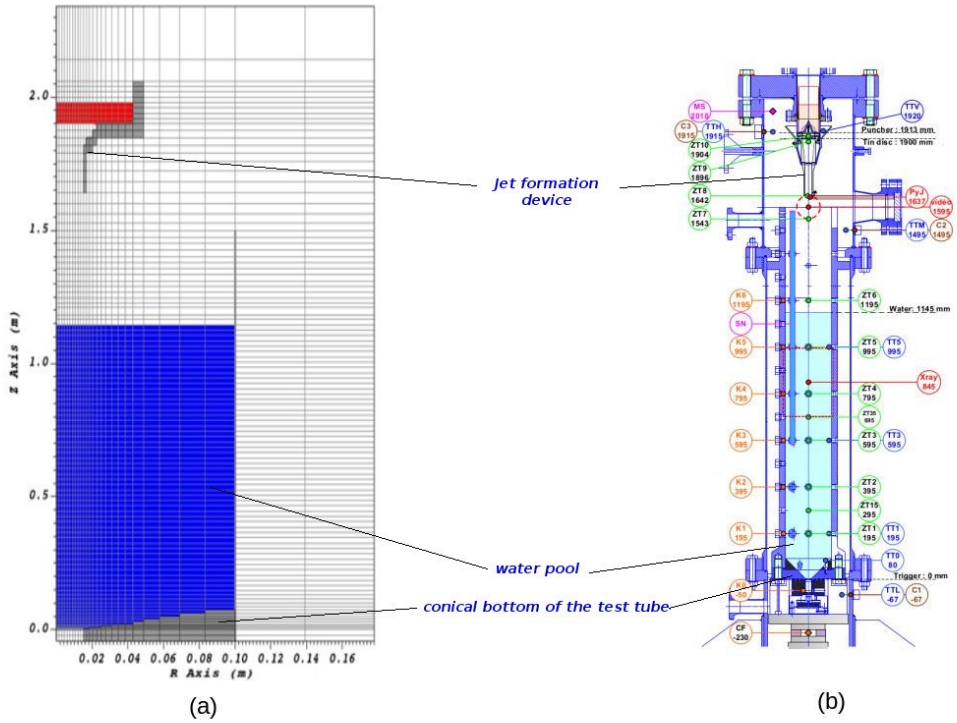

Figure 1.12: *On the left: representation of the spatial mesh for the numerical experiment.The R-axis is expanded with respect to the Z-axis for better visibility. On the right: schematic view of the KROTOS installation.*

1.2.3.2 Considered uncertain inputs

In our work, we focus on the parameters related to the models of the code described above. These parameters can strongly influence the general behavior of the code and the knowledge about their uncertainties is quite limited.

The operating parameters (temperatures, pressures, sizes) have been fixed on the measurements of the KS4 experiment. The uncertainty concerning these parameters are not taken into account in this work. In addition, some of these parameters are known to have a great influence on the stability of the MC3D code and thus on its propensity to converge correctly. The remaining $d = 57$ parameters considered as uncertain here are distributed as follow:

- 39 inputs for $f_{[1]}$;
- 11 inputs of both codes;
- 7 inputs of $f_{[2]}$.

There is very little knowledge about the uncertainty of these inputs. A uniform distribution centered around their nominal value is therefore assumed to characterize their uncertainty. Their variation domains have been discussed with physicists. In particular, these intervals have been chosen considering the physical plausibility and the limits of use of numerical models. The stability of the MC3D code has also been taking into account.

Furthermore, in our work, all the inputs are considered independent,except for two groups of parameters linked with the corium solidification. First, the material solidification temperatures are defined by their solidus and liquidus temperatures, $Tliquidus$ being greater than $Tsolidus$. To remove this dependency, The solidification interval, $TDIFF = Tliquidus - Tsolidus$, with $TDIFF > 0$, has been used instead of $Tliquidus$. Second, MC3D solidification models use a solidification temperature, $Tsolid$, $Tsolid$ being between $Tsolidus$ and $Tliquidus$. To lift the dependency of $Tsolid$ with $Tsolidus$ and $Tliquidus$, we introduced the temperature ratio $KTDROP$, such that $Tsolid = Tsolidus + KTDROP(Tliquidus - Tsolidus)$, with $0 < KTDROP < 1$.

Tables 1.2, 1.3 and 1.4 give a summary of the information concerning respectively the inputs $\boldsymbol{X}_{[1]}$, $\boldsymbol{X}_{[1,2]}$ and $\boldsymbol{X}_{[2]}$.

Table 1.2: *Details about uncertain inputs* $\boldsymbol{X}_{[1]}$. *The unit of the input is specified when it has one.*

Input number	Input name	Nominal value	Variation domain
1	KELMHOLTZ_FRAGNUM (-)	0.15	[0.1,0.2]
2	KELMHOLTZ_CDIACRE (-)	5	[4,6]
3	TXMIN (-)	0.3	[0.3,0.4]
4	TXJMIN (-)	0.8	[0.7,0.9]
5	KELMHOLTZ_C_VELFRA (-)	5	[4,6]
6	KELMHOLTZ_RADIUS (m)	0.03	[0.02,0.04]
7	K (-)	0	[0,1]
8	TXMAX (-)	0.5	[0.4,0.7]
9	COTX (-)	200	[150,250]
10	COEF_TFI (-)	0.99	[0,1]
11	DRAMULLD (-)	1	[0.5,1.5]
12	DRAMULGD (-)	1	[0.5,1.5]
13	CFRAG (-)	0	[0,0.5]
14	SBUTRI (-)	0.3	[0.25,0.35]
15	SLITRI (-)	0.7	[0.65,0.75]
16	DBUMIN (m)	0.0005	[0.0004,0.0006]
17	DLIMIN (m)	0.0005	[0.0004,0.0006]
18	DBUMAX (m)	0.05	[0.04,0.06]
19	DLIMAX (m)	0.05	[0.04,0.06]
20	DDRMIN (m)	0.0002	[0.0001,0.0003]
21	DDRMAX (m)	0.05	[0.04,0.06]
22	CONMUL (-)	1	[0.9,1.1]
23	NU_BUL (-)	150	[10,300]
24	NU_LIQ (-)	10	[10,300]
25	DRAMUL (-)	1	[0.5,1.5]
26	DRAMULLG (-)	1	[0.5,1.5]
27	WECRBUL (-)	12	[3,21]
28	WECRLIQ (-)	12	[3,21]
29	WECRDL (-)	12	[3,21]
30	CRAD (-)	0.1	[0,0.2]
31	CONTACT ($W/m^2/K$)	0	[0,100000]
32	MULTEH (-)	1	[0.5,1.5]
33	APPRWE (-)	20	[10,30]
34	COMPRE (s^2/m^2)	0.000004	[3E-7,4E-7]
35	LIFT (-)	0	[0,1]
36	DISTU (-)	0	[0,1]
37	LIFTB1 (-)	0	[0,1]
38	LIFTB2 (-)	0	[0,1]
39	DISTUB1 (-)	0	[0,1]
40	DISTUB2 (-)	0	[0,1]

Table 1.3: *Details about uncertain inputs* $\boldsymbol{X}_{[1,2]}$. *The unit of the input is specified when it has one.*

Input number	Input name	Nominal value	Variation domain
41	TLIQUIDUS (K)	2920	[2850,2950]
42	TDIFF (K)	50	[30,70]
43	CPLIQUID $(J/Kg/K)$	510	[460,560]
44	CPSOLID $(J/Kg/K)$	450	[400,500]
45	TENSURF (N/m)	0.45	[0.4,0.5]
46	EFUSION (J/kg)	3E5	[2.7E5,3.3E5]
47	CONDUCT (K)	2.88	[2.7,3]
48	VISCODYN $(Pa.s)$	0.008	[0.007,0.009]
49	EMISSIV (-)	0.75	[0.7,0.9]
50	ROSOLIQ (kg/m^3)	7500	[7000,8000]

Table 1.4: *Details about uncertain inputs* $\boldsymbol{X}_{[2]}$. *The unit of the input is specified when it has one.*

Input number	Input name	Nominal value	Variation domain
51	VISCART (-)	5	[4,6]
52	CMFRHY (-)	0.5	[0.3,0.7]
53	DFRAG (m)	7.5E-5	[5E-5,1E-4]
54	DTMFB (K)	10	[5,15]
55	DMINEXP (m)	5E-4	[4.5E-4,5.5E-4]
56	DMAXEXP (m)	0.05	[0.03,0.07]
57	HFRAGFB $(W/m^3/K)$	1E10	[5E9,1.5E10]

In practice and without loss of generality, the statistical tools will be applied here on the inputs scaled on the unit hypercube $[0,1]^d$. A preliminary re-scaling is performed before running the simulation. Hence, we will consider in the following that all the d inputs vary uniformly in $[0,1]$: $\boldsymbol{X} \sim U_{[0,1]^d}$.

As stated before, these parameters are associated to different models in MC3D. Table B.2 presents the parameters associated to each sub-model of MC3D code.

Table 1.5: *List of uncertain inputs associated to each MC3D sub-model.*

MC3D sub-model	inputs concerned
Jet fragmentation	1-6
Drop solidification	7-9
Drop fragmentation and heat transfers	10-33
corium compressibility	34
Drop momentum equations parameters	35-40
Corium physical properties	41-50
artificial viscosity	51
General explosion parameters	51-57

1.2.3.3 Studied outputs

In this work, we decided to focus only on scalar outputs of *type 1*. There are two main reasons for this. First, there are more uncertainty quantification tools developed for this kind of output. They are also easier to set up and test. Second and more importantly, one of the objectives of this thesis is the preparation of the calibration of the MC3D code on KROTOS experiments. However, the vast majority of the experimental outputs which are available on different KROTOS experiments are scalar outputs. Some *time series* (point data measured over time) can also be obtained. Extending the proposed methodology to this type of data could be an interesting prospect.

The output vector is denoted $\boldsymbol{Y} = (Y_1, Y_2, Y_3, Y_4)$. Two outputs of $f_{[1]}$ and two outputs of $f_{[2]}$ are considered. These four outputs are representative of what can be observed experimentally. Table 1.6 gives a brief description of these outputs.

The outputs are processed independently from each other. The notation $Y = f(\boldsymbol{X})$ is used to denote a scalar output without specifying which one. We denote by f the measurable function linking the inputs $\boldsymbol{X}$ to the scalar output Y.

Table 1.6: *Details about the four studied outputs. Their unit is specified.*

Output number	Model	Output name	Description
1	$f_{[1]}$	vm (kg)	vapor mass at the end of the premix
2	$f_{[1]}$	dsauter_p (m)	average corium drop diameter at the end of the premix
3	$f_{[2]}$	im_k $(P.s)$	maximum impulsion estimated using the $k = 6$ pressure captors
4	$f_{[2]}$	mf (kg)	mass of the fragment at the end of the explosion

1.2.4 Aims and methods

The work proposed in this thesis aims to address the two following operational objectives:

1. Understand the influence of modeling parameters on code outputs.
2. Prepare a calibration of these modeling parameters on experimental data.

The purpose is to develop and apply a methodology within the uncertainty quantification framework to meet these objectives. To do so, it necessary to face different challenges. As a matter of fact, the modeling of the fuel-coolant interaction (FCI) involves the use of a large amount of models with different time scales and linked together in a global model simulator. As a consequence, this makes the code simulating the FCI very computationally expensive. That means that only a limited number of code runs can be performed. Further, there is also a large number of input parameters. These two drawbacks limit the exploration of the input space. Moreover, the interweaving of the modeled phenomena with sometimes opposite effects make pre-analysis of physicists difficult to provide. Indeed, the physical relation between the inputs and the outputs is highly non-linear. This can even lead, for a plausible input data set at first glance, to code failures.

To meet the aforementioned objectives taking into account the special frame of our industrial problem, a sequential methodology is proposed. Adapted from the ICSREAM (acronym of Identification of penalizing Configurations using SCREening And Metamodel) methodology of Marrel et al. (2022), it consists in five main step:

- **Step 1: Exploration of the input space.** Knowing the variation domain of the input variables, the inputs space is randomly sampled. A code run is then performed on each sampled point of the experimental design. The

obtained sample of inputs/outputs constitutes the data set (also referred as *learning sample*). The goal is here to explore with this sample the input space and get as much information as possible about the behavior of the code output. Only having a limited budget for simulations, we use a near random sampling method, named Latin hypercube sampling (Mckay et al., 1979) instead of a classic random sampling method. This method ensures a better space coverage of the marginal distributions while maintaining some properties of classical Monte-Carlo sampling.

- **Step 2: Sensitivity analysis of code failure.** During the input space exploration, we find that a significant proportion of the code runs fail to converge. Analyzing these code failures to understand which of the inputs have the most influence on them leads to a better understanding of how the code works. It could also help to better identify the possible input domain of use.

- **Step 3: Screening of the inputs from sensitivity analysis of the outputs.** From the data set, a screening is proceed using two sensitivity analysis indices. The first one is the Sobol' first order index. It is based on the conditional variance of the outputs regarding the inputs (Sobol', 1993). The other one is the Hilbert Schmidt Independence criterion (HSIC). It is based on dependence mesures between inputs and outputs (Gretton et al., 2005). A pool of significant inputs is then selected using these indices.

- **Step 4: Outputs approximation by regression models.** From the data set, regression models are built to fit the simulator outputs. These models are intended to replace the initial code. For this, Gaussian Process (GP) regression method is used (Rasmussen and Williams, 2005). This method has several advantages over other regression methods: probabilistic prediction and exact interpolation property, as well as the evaluated numerical simulator modeling capabilities (Santner et al., 2003 ; Marrel et al., 2008).

- **Step 5: Use of the regression models to perform a more complete sensitivity analysis.** A quantitative sensitivity analysis is performed using the regression models instead of the computer model based on an intensive exploration of the input space. This is made possible by the negligible evaluation time of the metamodels. We are particularly interested in estimating variance-based sensitivity indices, the Sobol' first order (as in step 3) and, much more informative, the total indices. This quantitative sensitivity analysis prepares the calibration by ranking the inputs by level of influence on the outputs.

The following chapters correspond to the different steps of this methodology. More precisely, Chapter 2 deals with the exploration of the inputs space by Latin Hypercube samplings (LHS). An overview of samplings methods and a justification of the use of LHS in our case is given. New and existing results concerning the convergence of different statistics using LHS are also presented. In Chapter 3, we focus on the analysis of code failures. Tools to perform a sensitivity analysis on these code failures are presented and used in our industrial context. Finally, Chapter 4 presents the three last steps of this methodology and their applications.

Chapter 2

Exploration of the input space: the use of Latin Hypercube Sampling Method

Numerous researchers are confronted with the difficult problem of the optimal organization of their numerical or physical experiments. In other words, how to obtain a good amount of information with a reasonable number of experiments (or model evaluation)? Design of experiments (DoE) methods, also called sampling methods, try to give answers to this challenging question.

The experiments for which DoEs are designed can be real (physical experiments) or, as in the context of this work, simulated with a computer code. Numerous methods have been developed in order to explore the space of the input variables (or parameters) involved in the simulation. Numerical designs have several characteristics in the general framework of DoE. Indeed, in the experimental context, it is often assumed that there are random errors due to the measurements or the experimental conditions. In general, this is not the case for numerical simulations, which are primarily deterministic [1]. Moreover, in numerical experiments, it is possible to consider high variations on a very large number of uncertain parameters. Thus, the input space of a numerical experiment can be much bigger than the one feasible for a physical experiment. In the framework of uncertainty treatment, numerical designs of experiments can be support of various and different objectives, among which:

[1] Numerical errors and stochastic codes are not considered here.

- the study of the propagation of uncertainties (see step C of Figure 1.10);
- sensitivity analysis (cf. step C' of 1.10);
- approximation of code outputs by a metamodel.

In the context of our work, sensitivity analysis and metamodeling of the outputs are part of the proposed methodology to study the MC3D simulation code. In order to efficiently perform these two steps, a good numerical design of experiments is needed. This sampling step consists of defining a design of n experiments for the inputs and then performing the corresponding runs with MC3D. The obtained inputs/outputs sample constitutes the data set (also called the learning sample) on which the rest of our analysis is performed. As mentioned before, the MC3D code is very time consuming and the number of input considered is large ($d = 57$ inputs). It is therefore important to effectively explore, with a reasonable number of simulations, this very large variation domain of inputs. A wise choice of the method generating our DoE is then crucial for a good exploration.

In the following section, we provide an overview of the main methods to generate a design of experiments. We also motivate the use of the Latin Hypercube Sampling (LHS) method (Mckay et al., 1979) for our industrial case. Then, in Section 2, we present some new convergence results under LHS for an important class of estimators: the Z-estimators. Indeed, this class of estimators covers a large number of estimation methods, including for instance the maximum likelihood. Finally, we discuss in Section 2.3 the behavior of statistics specifically used in our methodology, under LHS.

2.1 Review of sampling methods

In this section we present a brief overview of the existing sampling methods for the input space of a simulation code. We later provide a rationale for our choice to use LHS in our application.

Before presenting these different methods, let us introduce some notations that will be used throughout the rest of the chapter. We first denote by $\boldsymbol{X} = (X_1, \ldots, X_d)$ the vector of the d independent inputs evolving in a measurable space $\mathcal{X} \subset \mathbb{R}^d$. For simplicity and without loss of generality, we assume that the d inputs vary uniformly in $[0, 1]$ so we have that, for $j = [\![1, d]\!]$, $X_j \sim U_{[0,1]}$. Indeed, one can always work under uniformity and then apply the inverse transformation to place the support back on the original scale and retrieve the original distribution (Devroye, 1986). In our case study, only a preliminary re-scaling before running the simulation is needed.

A DoE associated to $\boldsymbol{X}$ and generated using a sampling method, generically denoted "$METHOD$" is written as follows:

- $\mathbf{X}^{METHOD} = \left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\right)^T \in M_{n,d}([0,1])$. Here, $n \in \mathbb{N}^*$ is the sample size. We also recall that $M_{n,d}([0,1])$ denotes the space of matrices of size $n \times d$ with coefficients in $[0,1]$.
- $\boldsymbol{x}_j^{METHOD} = \left(x_j^{(1)}, \ldots, x_j^{(n)}\right)^T$ with $j \in [\![1,d]\!]$ is the jth column of $\mathbf{X}^{METHOD}$ corresponding to the effective generated sample of the input X_j.

If no sampling method is mentioned, it means that the results presented do not depend on the sampling method.

We also define the measurable function $f : \mathcal{X} \to \mathbb{R}^q$ with $q \in \mathbb{N}^*$ such that $f(\boldsymbol{X}) = \boldsymbol{Y}$. This function represents in practice the studied simulation code. $\mathbf{Y}^{METHOD} = f(\mathbf{X}^{METHOD}) = (\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(n)})^T$ is the matrix of output samples corresponding to the DoE $\mathbf{X}^{METHOD}$.

For $\boldsymbol{a} = (a_1, \ldots, a_q) \in \mathbb{R}^q$ with $q \in \mathbb{N}^*$, we denotes by $||\boldsymbol{a}||$ the Euclidean norm of $\boldsymbol{a}$ such that $||\boldsymbol{a}||^2 = \sum_{i=1}^q a_i^2$. Similarly, for any matrix $\boldsymbol{A} \in M_{q,q}(\mathbb{R})$, we denote by $||\boldsymbol{A}||$ the *pseudo* Euclidean norm (Frobenius norm) such that $||\boldsymbol{A}||^2 = \sum_{1 \leq i,j \leq q} A_{i,j}^2$. Here, $A_{i,j}$ with $i, j \in [\![1,q]\!]$ are the components of the matrix $\boldsymbol{A}$. A *pseudo* Euclidean norm $||.||$ is finally associated to the tensor space $T_{q,q,q}(\mathbb{R})$. This norm is defined, for all $\boldsymbol{T} \in T_{q,q,q}(\mathbb{R})$, by $||\boldsymbol{T}||^2 = \sum_{1 \leq i,j,k \leq q} T_{i,j,k}^2$. Here, $T_{i,j,k}$ with $i, j, k \in [\![1,q]\!]$ are the components of the tensor $\boldsymbol{T}$.

We denote by $o(1)$ ("small oh-one") a deterministic sequence that converges to 0 and $O(1)$ ("big oh-one") a deterministic sequence that is bounded. We denote by $o_p(1)$ ("small oh-P-one") a sequence of random variables that converges in probability to 0. The expression $O_p(1)$ ("big oh-P-one") denotes a sequence of random variables that is bounded in probability. We recall that a sequence of random variables $(\boldsymbol{W}_n)_{n \in \mathbb{N}}$ is bounded in probability if, for any scalar $\epsilon > 0$, there exist M and N such that, $\forall n > N$, $\mathbb{P}(||\boldsymbol{W}_n|| > M) < \epsilon$.

Finally, a multivariate normal distribution of dimension q ($q \in \mathbb{N}^*$) with a mean equal to $\boldsymbol{\mu} \in \mathbb{R}^q$ and a covariance matrix equal to $\boldsymbol{\Sigma} \in M_{q,q}(\mathbb{R})$ is denoted $\mathcal{N}_q(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

2.1.1 Simple Random Sampling (SRS)

Simple Random Sampling (SRS), also called Monte Carlo sampling, is one of the first methods that have been used in the context of uncertainty quantification. It is indeed an intuitive method with good properties.

This method simply consists in independently drawing n realizations of each input random variable. The sample obtained for an input X_j is of the form $\boldsymbol{x}_j^{SRS} = (u_j^{(1)}, \ldots, u_j^{(n)})^T$. Here, $(u_j^{(i)})_{i \in [\![1,n]\!]}$ are *independent and identically distributed* (*i.i.d.*) realizations of the distribution $U_{[0,1]}$. Figure 2.1 shows an example of a Monte Carlo DoE in dimension 2 with $n = 20$ and $n = 100$ points.

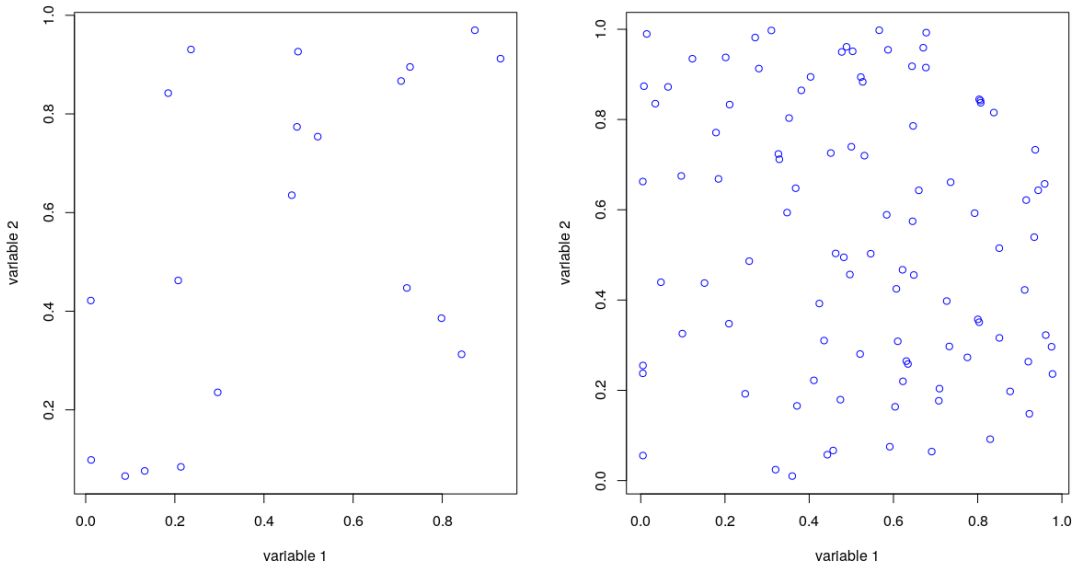

Figure 2.1: *SRS of dimension* $d = 2$ *with* $n = 20$ *on the left and* $n = 100$ *points on the right.*

This method has several advantages. First, its computational cost is low, even for very high sample sizes and input dimensions. Moreover, the obtained samples are by construction *i.i.d.* Due to this property, one can add or remove points to a given data sample without losing the *i.i.d.* characteristic. In addition, the convergence results of statistics are usually settled in this context. Thus, using a SRS design allows the user to write theoretical properties of the objects at hand. In particular, one knows exactly quantities as asymptotic mean, variance or distribution.

Nevertheless, as one can observe in Figure 2.1, the coverage of the input space can be irregular. This is due to the purely random nature of the method. This is even more true when the dimension increases.

2.1.2 Low-discrepancy sequences

The discrepancy is a measure quantifying the difference between a design and the uniform distribution (Thiémard, 2000). In other words, the discrepancy of a DoE $\mathbf{X}$ can be viewed as the largest absolute difference between the continuous uniform probability and the $\mathbf{X}$ taken over all possible sub-cubes of $[0,1]^d$. The lower a discrepancy measure is, the better a sample is distributed. Several discrepancy measures have been established. For instance, the *extreme discrepancy* of $\mathbf{X}$ is defined as follow:

$$\text{Disc}_n(\mathbf{X}) = \sup_{\boldsymbol{s}\in\mathcal{S}} \left| \frac{Q(\boldsymbol{s},\mathbf{X})}{n} - \lambda(\boldsymbol{s}) \right|.$$

Here,

- $\mathcal{S}$ is the set of all d-dimensional rectangles of the form $\prod_{j=1}^{d}[a_j, b_j[= \prod_{j=1}^{d}\{\boldsymbol{x} \in \mathbb{R}^d, a_j \leq x_j < b_j\}$ where $0 \leq a_j < b_j \leq 1, j \in [\![1, d\,]\!]$;
- λ is the d-dimensional Lebesgue measure;
- $Q(\boldsymbol{s}, \mathbf{X})$ is the number of points of the DoE $\mathbf{X}$ in the subset $\boldsymbol{s} \in \mathcal{S}$.

Another discrepancy measure, the *star discrepancy* $\text{Disc}_n^*(\mathbf{X})$ can be defined similarly by replacing the set $\mathcal{S}$ by the set $\mathcal{S}^* = \prod_{j=1}^{d}[0, c_j[$, c_j being in the half-open interval $[0,1[$. Only the sub-class of rectangles containing the origin are taking into account for the *star discrepancy*. An interesting thing is that these two discrepancy measures are related through the following inequality:

$$\text{Disc}_n^*(\mathbf{X}) \leq \text{Disc}_n(\mathbf{X}) \leq 2^d \text{Disc}_n^*(\mathbf{X}).$$

Low-discrepancy suites are designed to minimise a discrepancy measure such as Disc_n or Disc_n^* . More formally, a low-discrepancy sequence $(v^{(i)})_{i\in\mathbb{N}}$ is built such that for any integer k, the sub-sequence $v^{(1)}, \ldots v^{(k)}$ has an optimal discrepancy. To obtain an n-sample of a variable X_j, we simply take the n first elements of the low discrepancy sequence: $\boldsymbol{x}_j^{LDS} = (v^{(1)}, \ldots, v^{(n)})^T$. The samples of the other inputs are obtained similarly, only changing the initialization of the sequence.

Thus, designs generated by low discrepancy sequences well cover, by construction, the input space. They also have the particularity of being purely deterministic. Halton (1964) and Sobol' (1967) gave examples of low discrepancy sequences. Figure 2.2 shows two DoE in dimension $d = 2$ generated by the Sobol' sequence, with $n = 100$ points and $n = 1000$ points respectively.

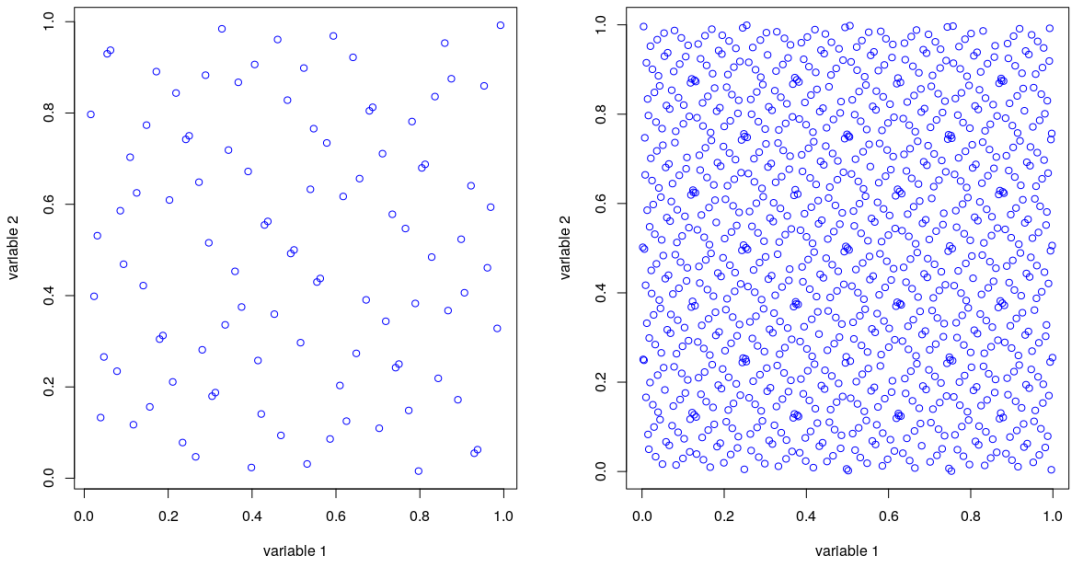

Figure 2.2: *Sobol' sequence of dimension* $d = 2$ *and size* $n = 100$ *on the left and size* $n = 1000$ *on the right.*

These DoE generation methods have a low computational cost, while allowing a very good space coverage. Moreover, as numerical sequences, they guarantee a perfect sequentiality and are easily reproducible. Only the first term of the sequence is needed to rebuild the whole DoE.

However, the main weakness of these methods is the presence of patterns or alignments, when generating many points or when the input space is large. Figure 2.2 illustrates this problem well. These patterns artificially create a form of interdependence between the inputs. In practice, this can affect the analysis of the code studied. Scrambling methods can nevertheless partially address this problem. The idea is to slightly *disturb* the numerical structure of the sequence to mitigate the patterns initially observed. For detailed description of scrambling methods, see (Chi et al., 2005) for instance. Another major disadvantage of this approach is the lack of theoretical convergence properties. This is in contrast to the simple Monte Carlo method, for which most asymptotic results are known.

2.1.3 Latin Hypercube Samplings (LHS)

Latin Hypercube Sampling (LHS), proposed in (Mckay et al., 1979), is another method for generating DoE. The idea of LHS is to divide the range of variation of each input into n intervals of size $1/n$ and then randomly generate a coordinate in

each interval of each dimension. Mapping these draws along each dimension allows the design of experiments to be obtained. For a better understanding, let us give an example. Imagine we are dealing with a square area in dimension 2. The LHS method consists of three steps. The first is to subdivide the domain into a grid of n^2 squares of equal area. The second is to randomly choose n squares such that no pair of squares has the same row or column. Finally, a point is randomly chosen in each square previously selected. Figure 2.3 gives schematic examples of LHS of dimension 2 and size $n = 4$. The process is similar in higher dimensions.

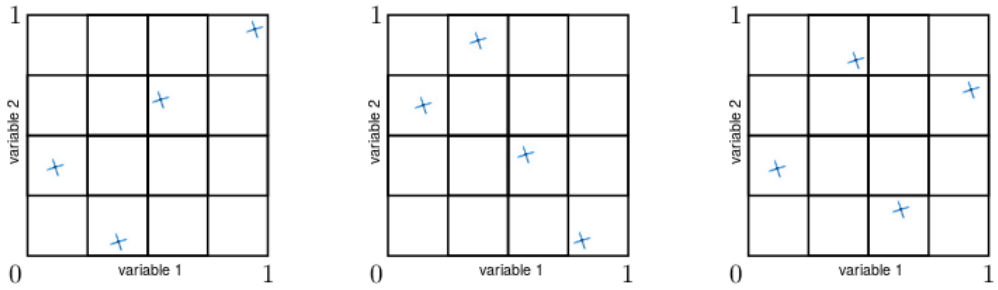

Figure 2.3: *Three schematic examples of LHS of dimension 2 and size* $n = 4$.

Mathematically, to obtain the sample corresponding to an input X_j, we segment the interval $[0, 1]$ into n intervals of the same length and we randomly choose a point in each one. To do so, we denote:

1. $\boldsymbol{\pi}_j = (\pi_j(1) \dots \pi_j(n))^T$, $j \in [\![1, d]\!]$ as a random permutation of $[\![1, n]\!]$, according to the uniform distribution on the set of a all possible permutations of $[\![1, n]\!]$. The random permutations $(\boldsymbol{\pi}_j)_{j \in [\![1,d]\!]}$ are assumed to be independent.
2. $\boldsymbol{u}_j = (u_j^{(1)}, \dots, u_j^{(n)})^T, j \in [\![1, d]\!]$ as an *i.i.d.* sample of the uniform distribution $U_{[0,1]}$. The samples $(\boldsymbol{u}_j)_{j \in [\![1,d]\!]}$ are assumed to be independent.

Note that the random permutations $(\pi_j)_{j \in [\![1,d]\!]}$ and the samples $(\boldsymbol{u}_j)_{j \in [\![1,d]\!]}$ are also assumed to be independent. The n-sized sampling x_j^{LHS} of the input $X_j, j \in [\![1, d]\!]$, is then defined as follow:

$$\boldsymbol{x}_j^{LHS} = \left(x_j^{(1)}, \dots, x_j^{(n)}\right)^T = \left((\pi_j(1) - u_j^{(1)})/n, \dots, (\pi_j(n) - u_j^{(n)})/n\right)^T. \quad (2.1)$$

The corresponding Latin Hypercube of dimension d and size n is then $\mathbf{X}^{LHS} = (\boldsymbol{x}_1^{LHS}, \ldots, \boldsymbol{x}_d^{LHS})$.

The LHS method leads to a good point repartition in the sub-projections of dimension 1. Indeed, one has by definition,

$$\min_{1 \leq i, i' \leq n} (|x_j^{(i)} - x_j^{(i')}|) \leq 2/n.$$

As a result of its stratified nature, the realizations of the LHS design are not *i.i.d.* However, LHS designs still ensure good statistical properties. Several results have been indeed established for the convergence of estimators under LHS. Most of them concern mean statistics (first order U-statistics). For instance, it has been shown that the estimator of $\boldsymbol{\mu}_f = \mathbb{E}(f(\boldsymbol{X}))$ is unbiased (Mckay et al., 1979):

Property 2.1 *Let $f : [0,1]^d \to \mathbb{R}^q$ with $d, q \in \mathbb{N}^*$ be a measurable function such that $\mathbb{E}(||f(\mathbf{X})||) < +\infty$. Denote*

$$\bar{f}_n^{LHS} = \frac{1}{n} \sum_{i=1}^{n} f(\boldsymbol{x}^{(i)}),$$

where $\boldsymbol{x}^{(i)}, i \in [\![1, n]\!]$ is such that $\mathbf{X}^{LHS} = \left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\right)^T$ with $\mathbf{X}^{LHS}$ being defined using Equation (2.1). Then, $\bar{f}_n^{LHS}$ is an unbiased estimator of $\boldsymbol{\mu}_f = \mathbb{E}(f(\boldsymbol{X}))$.

In the sequel, we denote by $\bar{f}_n^{METHOD}$ the empirical mean defined by analogy with the one of Property 2.1, but using the sampling method "METHOD".

A second interesting characteristic of the mean value estimators using LHS is their variance. Indeed, Stein (1987) showed that if f is a real-valued function such that $\mathbb{E}[f^2(\mathbf{X})] < +\infty$, then $\mathbb{V}\mathrm{ar}(\bar{f}_n^{LHS})$ is always asymptotically smaller than $\mathbb{V}\mathrm{ar}(\bar{f}_n^{SRS})$. This result is generalized to multidimensional outputs in Loh (1996). Property 2.2 summarizes the main results regarding the covariance matrix of $\bar{f}_n^{LHS}$, in a similar manner to what is presented in Loh (1996):

Property 2.2 *Let $f : [0,1]^d \to \mathbb{R}^q$, $d, q \in \mathbb{N}^*$ be a measurable function such that $\mathbb{E}(||f(\mathbf{X})||^2) < +\infty$. Let $\boldsymbol{\Sigma}_{\bar{f}_n^{SRS}}$ and $\boldsymbol{\Sigma}_{\bar{f}_n^{LHS}}$ be the covariance matrices of $\bar{f}_n^{SRS}$ and $\bar{f}_n^{LHS}$ respectively, with $\boldsymbol{\Sigma}_{\bar{f}_n^{SRS}} = \frac{1}{n}\mathbb{E}\left(\left(f(\boldsymbol{X}) - \boldsymbol{\mu}_f\right)\left(f(\boldsymbol{X}) - \boldsymbol{\mu}_f\right)^T\right)$.*

We also define, for $\boldsymbol{x} = (x_1, \ldots, x_d) \in [0,1]^d$:

- $f_{-j}(x_j) = \int_{[0,1]^{d-1}} [f(\boldsymbol{x}) - \boldsymbol{\mu}_f] \prod_{1 \leq k \leq d, k \neq j} dx_k$ *with* $j \in [\![1, d]\!]$.
- $f_{rem}(\boldsymbol{x}) = f(\boldsymbol{x}) - \boldsymbol{\mu}_f - \sum_{j=1}^{d} f_{-j}(x_j)$

Then we have, when $n \to +\infty$ *:*

- $\boldsymbol{\Sigma}_{\bar{f}_n^{LHS}} = \frac{1}{n}\int_{[0,1]^d} f_{rem}(\boldsymbol{x}) f_{rem}(\boldsymbol{x})^T \boldsymbol{dx} + \frac{1}{n} o(1)$.
- $\boldsymbol{\Sigma}_{\bar{f}_n^{SRS}} = \frac{1}{n}\int_{[0,1]^d} f_{rem}(\boldsymbol{x}) f_{rem}(\boldsymbol{x})^T \boldsymbol{dx} + \frac{1}{n}\sum_{j=1}^{d}\int_{[0,1]} f_{-j}(x_j) f_{-j}(x_j)^T dx_j$.

We therefore have that $\boldsymbol{\Sigma}_{\bar{f}_n^{SRS}} - \boldsymbol{\Sigma}_{\bar{f}_n^{LHS}}$ *is asymptotically positive semidefinite, that is,*

$$\forall \xi \in \mathbb{R}^p, \quad \lim_{n\to+\infty} n\xi^T(\boldsymbol{\Sigma}_{\bar{f}_n^{SRS}} - \boldsymbol{\Sigma}_{\bar{f}_n^{LHS}})\xi \geq \sum_{j=1}^{d}\int_{[0,1]} \xi^T f_{-j}(x_j) f_{-j}(x_j)^T \xi dx_j \geq 0.$$

Notice that the normalized quantity $n(\boldsymbol{\Sigma}_{\bar{f}_n^{SRS}} - \boldsymbol{\Sigma}_{\bar{f}_n^{LHS}})$ is also asymptotically positive semidefinite, according to the last equation of Property 2.2.

Since $\bar{f}_n^{SRS}$ converges in quadratic mean to $\boldsymbol{\mu}_f$, we can therefore conclude that $\bar{f}_n^{LHS}$ converges in quadratic mean to $\boldsymbol{\mu}_f$, $\lim_{n\to+\infty} \mathbb{E}\Big(||\bar{f}_n^{LHS} - \boldsymbol{\mu}_f||^2\Big) = 0$. Thus, $\bar{f}_n^{LHS}$ converges in probability to $\boldsymbol{\mu}_f$.

In addition, Owen (1992) showed a Central Limit Theorem (CLT) for this class of estimators under LHS when the model function f is bounded. This was generalized to any function with finite third moment in (Loh, 1996). Let us introduce, in the framework of Property 2.2, $\boldsymbol{R}_f = \int_{[0,1]^d} f_{rem}(\boldsymbol{x}) f_{rem}(\boldsymbol{x})^T \boldsymbol{dx}$. Notice that asymptotically we have $\boldsymbol{\Sigma}_{\bar{f}_n^{LHS}} = \frac{1}{n}\Big(\boldsymbol{R}_f + o(1)\Big)$. It is important to note that, unlike $\boldsymbol{\Sigma}_{\bar{f}_n^{LHS}}$, $\boldsymbol{R}_f$ does not depend on n. The CLT can be then expressed as follow:

Theorem 2.1 *Let* $f : [0,1]^d \to \mathbb{R}^q$*, with* $d, q \in \mathbb{N}^*$ *be a measurable function such that* $\mathbb{E}\Big(||f(\mathbf{X})||^3\Big) < +\infty$*. Then assuming that* $\boldsymbol{R}_f$ *is non-singular, we have that* $\sqrt{n}\boldsymbol{R}_f^{-1/2}(\bar{f}_n^{LHS} - \boldsymbol{\mu}_f) \xrightarrow{n\to+\infty} \mathcal{N}_q(0, \boldsymbol{I}_q)$*. Here,* $\boldsymbol{I}_q$ *is the identity matrix of size* q*.*

Notice that we also have that, under the same conditions, $\sqrt{n}(\bar{f}_n^{LHS} - \boldsymbol{\mu}_f) \xrightarrow{n\to+\infty} \mathcal{N}_q(0, \boldsymbol{R}_f)$. Theorem 2.1 will be used in this form later.

Thus, even if LHS samples are not *i.i.d.*, the convergence properties are preserved for the estimators of a mean value. Moreover, the variance of mean estimators is lower under LHS than under SRS.

A way to get even better space-filling properties is to optimize an initial LHS according to a criterion describing the spatial mapping, such as discrepancy measures presented above (Damblin et al., 2013). The counterpart of LHS optimization is that the dependencies between the realizations become more complex to characterize. Moreover, as for low discrepancy sequences, there are no theoretical guarantees concerning the convergence of statistics for this type of DoE.

Figure 2.4 depicts a pure Monte Carlo (SRS), a LHS and an optimized LHS regarding the star discrepancy (Damblin et al., 2013). We can observe from these

figures that the mapping is better with an LHS than with a pure Monte Carlo sample. It is even better with an optimized LHS, but as expected, the repartition of the points seems more structured.

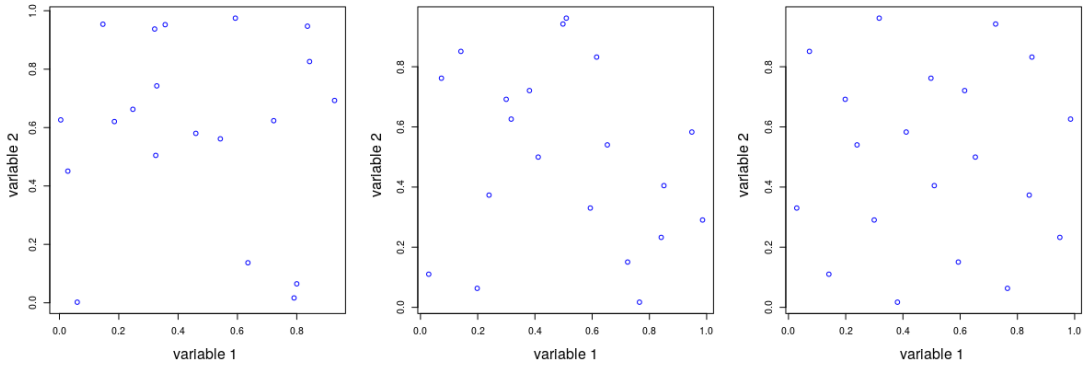

Figure 2.4: *From left to right: examples of SRS, LHS and optimized LHS, all with dimension* $d = 2$ *and size* $n = 20$ *realizations.*

Consequently, for our industrial application, the selected DoE method is the classic LHS. In fact, it allows a good compromise between space coverage regardless of dimension and theoretical convergence properties. The rest of this chapter therefore discusses the convergence of different statistics under LHS.

2.2 M- and Z-estimators under LHS

M-estimators are a broad class of estimators. In general, they are obtained by optimizing an empirical mean. The definition of M-estimators was motivated by robust statistics. We refer to Rousseeuw et al. (1986) for a more general discussion on the subject. The Z-estimator class is directly related to the M-estimator class. The brief overview given here is mainly inspired by the fifth chapter of Van der Vaart (1998).

Let $\mathrm{X} = (\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n)})^T$ be the vector of n realizations of a random variable $\boldsymbol{X}$ evolving in a measurable space $\mathcal{X} \subset \mathbb{R}^p$, with $p \in \mathbb{N}^*$. Its law is parameterized by a vector $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^q, q \in \mathbb{N}^*$. The statistic $\hat{\boldsymbol{\theta}}_n = \hat{\boldsymbol{\theta}}_n(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n)})$ is a M-estimator of $\boldsymbol{\theta}$ if it maximises a function of the type

$$\boldsymbol{\theta} \to M_n(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} m_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}). \tag{2.2}$$

Here, $m_{\boldsymbol{\theta}}$ is a known real-valued function on $\mathcal{X}$. The maximum value of a function is often found by setting a derivative (or a set of derivative) to 0. In this case, the maximisation problem can be reformulated by the following vectorial equations, with $\psi_{\boldsymbol{\theta}}$ corresponding to the vector of derivatives of $m_{\boldsymbol{\theta}}$:

$$\Psi_n(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} \psi_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) = 0. \tag{2.3}$$

More generally, we may consider in Eq. (2.3) the function $\psi_{\boldsymbol{\theta}}$ as a known measurable and vector-valued map evolving from $\mathbb{R}^p$ to $\mathbb{R}^q$. $\psi_{\boldsymbol{\theta}}$ is not necessarily a derivative of another function in the general case. The name Z-estimator is used for estimators $\hat{\boldsymbol{\theta}}_n$ satisfying the vectorial equations (2.3). In this section, we are focusing on this kind of estimator. Thus in the following, the notation $\hat{\boldsymbol{\theta}}_n$ will always refer to an estimator defined by Equation (2.3). This equation may not have an exact solution. Then it is natural to use as estimator a value that is close to zero. Estimators that are sufficiently close to being a zero often have the same asymptotic behavior.

Many known estimators can be defined as Z-estimator. For instance, let $\boldsymbol{X}$ have a distribution function $f_{\boldsymbol{\theta}}$ with a continuous first derivative in $\boldsymbol{\theta} \in \Theta$. In this case, the maximum likelihood estimator of $\boldsymbol{\theta}$ can be written as a Z-estimator as defined by 2.3 with, for $\boldsymbol{x} \in \mathbb{R}^p, p \in \mathbb{N}^*$, $\psi_{\boldsymbol{\theta}}(\boldsymbol{x}) = (\frac{\partial \log(f_{\boldsymbol{\theta}}(\boldsymbol{x}))}{\partial \theta_1}, \ldots, \frac{\partial \log(f_{\boldsymbol{\theta}}(\boldsymbol{x}))}{\partial \theta_q})^T$.

The aim of this section is to extend the results on the convergence of empirical mean estimators under LHS, namely Properties 2.1, 2.2 and Theorem 2.1, to the class of Z-estimators. First, useful general results concerning the convergence of Z-estimators are discussed. Then, we give original results regarding the asymptotic normality of this class of estimators under LHS.

2.2.1 Known properties of Z-estimators

The first useful properties regarding Z-estimators concern the link between the consistency of $\Psi_n(\boldsymbol{\theta})$ and the consistency of $\hat{\boldsymbol{\theta}}_n$. These properties set assumptions on the objective function $\Psi_n(\boldsymbol{\theta})$ that ensure convergence of the parameter $\boldsymbol{\theta}$ we aim to estimate.

For instance, if the parameter θ is a scalar, Proposition 2.3 presented in (Van der Vaart, 1998) gives sufficient conditions for the convergence of $\hat{\theta}_n$:

Property 2.3 *Let Θ be a subset of the real line. Let $\Psi_n(\theta)$ be random functions and $\Psi(\theta)$ a fixed function of $\theta \in \Theta$ such that $\Psi_n(\theta)$ converges to $\Psi(\theta)$ in probability for every θ. Assume also that each map $\theta \to \Psi_n(\theta)$ is continuous and has exactly one zero $\hat{\theta}_n$, or is nondecreasing and converges to 0 in probability. Let θ_0 be a point*

such that $\Psi(\theta_0 - \epsilon) < 0 < \Psi(\theta_0 + \epsilon)$ *for every* $\epsilon > 0$.

Then $\hat{\theta}_n$ *is a consistent estimator of* θ_0*, meaning that* $\hat{\theta}_n \xrightarrow[n\to+\infty]{p} \theta_0$.

This property has the advantage of holding under simple assumptions. However, as mentioned, it is limited to the scalar case. In Dacunha-Castelle and Duflo (1986), one can find assumptions for which the consistency of $\hat{\boldsymbol{\theta}}_n$ is ensured in the multidimensional case:

Property 2.4 *Let* Θ *be a compact subset of* $\mathbb{R}^q$ *with* $q \in \mathbb{N}^*$*. Let also assume that the following hypotheses are true, for any* $\boldsymbol{\theta} \in \Theta$*, :*

- *the functions* $\Psi_n(\boldsymbol{\theta})$ *and* $\Psi(\boldsymbol{\theta})$ *are continuous functions of* $\boldsymbol{\theta} \in \Theta$*;*
- *each function* $\boldsymbol{\theta} \to \Psi_n(\boldsymbol{\theta})$ *has exactly one zero* $\hat{\boldsymbol{\theta}}_n \in \Theta$*;*
- $\Psi_n(\boldsymbol{\theta})$ *converges to* $\Psi(\boldsymbol{\theta})$ *in probability;*
- $\Psi(\boldsymbol{\theta})$ *vanishes only at* $\boldsymbol{\theta}_0$ *with* $\boldsymbol{\theta_0} \in \Theta$*;*
- *denoting, for* $\eta \geq 0$, $w_n(\eta) = \sup\{||\Psi_n(\boldsymbol{\theta}_1) - \Psi_n(\boldsymbol{\theta}_2)||; ||\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2|| \leq \eta, \boldsymbol{\theta_1}, \boldsymbol{\theta_2} \in \Theta\}$ *; there exists two sequences* (η_k) *and* (ϵ_k) *both decreasing to* 0 *such that, for all* $k \in \mathbb{N}$, $\mathbb{P}(w_n(\eta_k) > \epsilon_k) \xrightarrow[n\to+\infty]{} 0$.

Then $\hat{\boldsymbol{\theta}}_n$ *is a consistent estimator of* $\boldsymbol{\theta}_0$*, that is* $\hat{\boldsymbol{\theta}}_n \xrightarrow[n\to+\infty]{p} \boldsymbol{\theta}_0$.

The assumption on $w_n(\eta)$ seems difficult to grasp at first glance. However, as mentioned in (Dacunha-Castelle and Duflo, 1986), if we find a function ϕ from $\mathbb{R}_+$ to $\mathbb{R}$ such that $\lim_{\eta\to 0^+} \phi(\eta) = 0$, this assumption on w_n can be obtained through: $\mathbb{P}(w_n(\eta) \geq 2\phi(\eta)) \xrightarrow[n\to+\infty]{} 0$ for each $\eta \geq 0$. For instance, $w_n(\eta) \xrightarrow[n\to+\infty]{} \phi(\eta)$, or $\lim_{n\to+\infty} w_n(\eta) \leq \phi(\eta)$ give both sufficient conditions.

In addition to these convergence properties, several central limit theorems for Z-estimators have been proved. Here we give one of them, proposed in (Van der Vaart, 1998). Theorem 2.2 relies on the so-called *classic conditions*. They were formulated in the 1930s and 1940s to mathematically tighten the informal derivation of the asymptotic normality of maximum likelihood proposed earlier by Fisher (1922). These conditions are stringent but they are simple. They lead to a simple proof of the central limit theorem. This simplicity will allow us to adapt this theorem to the LHS case.

In particular, a needed assumption for the application of this theorem concerns the existence of a first and a second order derivatives in $\boldsymbol{\theta}$ for $\psi_{\boldsymbol{\theta}}$. Let us introduce these terms, assuming they exist.

We consider that Θ is an open subset of an Euclidean space of dimension $q, q \in \mathbb{N}^*$. The first order partial derivative $\psi_{\boldsymbol{\theta}}$ in $\boldsymbol{\theta} \in \Theta$ is a size $q \times q$ matrix is denoted $\dot{\psi}_{\boldsymbol{\theta}}$. Its components are such that $\dot{\psi}_{\boldsymbol{\theta},j,k} = \frac{\partial \psi_{\boldsymbol{\theta},j}}{\partial \theta_k}$.

The second order partial derivative is a tensor of size $q \times q \times q$ (a q-vector of $q \times q$ matrices) denoted $\ddot{\psi}_{\boldsymbol{\theta}}$ such that $\ddot{\psi}_{\boldsymbol{\theta},j,k,l} = \frac{\partial^2 \psi_{\boldsymbol{\theta},j}}{\partial \theta_k \partial \theta_l}$.

Theorem 2.2 *Let Θ be an open subset of an Euclidean space of dimension $q, q \in \mathbb{N}^*$ and $\mathcal{X}$ be a measurable subspace of $\mathbb{R}^p, p \in \mathbb{N}^*$. Assume that, for all $\boldsymbol{\theta} \in \Theta$ and for all $\boldsymbol{x} \in \mathcal{X}$, the function $\boldsymbol{\theta} \to \psi_{\boldsymbol{\theta}}(\boldsymbol{x})$ is twice continuously differentiable in $\boldsymbol{\theta}$.*

Let $X^{SRS} = (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)})^T$ be the vector of i.i.d realizations of a random variable $\boldsymbol{X} = (X_1, \ldots X_p)$ evolving in $\mathcal{X}$.

Suppose also that the following assumptions are fulfilled:

1. $\Psi_n^{SRS}(\hat{\boldsymbol{\theta}}_n^{SRS}) = \frac{1}{n}\sum_{i=1}^n \psi_{\hat{\boldsymbol{\theta}}_n^{SRS}}(\boldsymbol{x}^{(i)}) = 0, \forall n \in \mathbb{N}$;
2. *there exists a unique* $\boldsymbol{\theta}_0 \in \Theta$ *such that* $\mathbb{E}(\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})) = \Psi(\boldsymbol{\theta}_0) = 0$ *with* $\boldsymbol{\theta}_0 \in \Theta$;
3. $\mathbb{E}(||\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})||^2) < +\infty$;
4. $\mathbb{E}(\dot{\psi}_{\theta_0}(\boldsymbol{X}))$ *exists and is nonsingular;*
5. *the function* $\boldsymbol{\theta} \to \ddot{\psi}_{\boldsymbol{\theta}}(\boldsymbol{x})$ *is dominated in norm by a fixed integrable function* $\ddot{\psi}(\boldsymbol{x})$ *for every* $\boldsymbol{\theta}$ *in the neighborhood of* $\boldsymbol{\theta}_0$.

Then, if $\hat{\boldsymbol{\theta}}_n^{SRS}$ is a consistent $\boldsymbol{\theta}_0$, we have:

$$(\hat{\boldsymbol{\theta}}_n^{SRS} - \boldsymbol{\theta}_0) = -[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1} \frac{1}{n}\sum_{i=1}^n \psi_{\boldsymbol{\theta}_0}(\boldsymbol{x}^{(i)}) + \frac{1}{\sqrt{n}} o_p(1).$$

Moreover, we have that the sequence $\sqrt{n}(\hat{\boldsymbol{\theta}}_n^{SRS} - \boldsymbol{\theta}_0)$ is asymptotically normal with mean zero and a covariance matrix equal to $[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}\mathbb{E}\Big(\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})^T\Big)[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-T}$.

For the following, it is important to note that we have

$$\frac{1}{n}\mathbb{E}\Big(\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})^T\Big) = \boldsymbol{\Sigma}_{\Psi_n^{SRS}(\boldsymbol{\theta}_0)}.$$

Here, $\boldsymbol{\Sigma}_{\Psi_n^{SRS}(\boldsymbol{\theta}_0)}$ is the covariance matrix of $\Psi_n^{SRS}(\boldsymbol{\theta}_0)$. It is also important to notice that among the results presented in this subsection, only Theorem 2.2 shown here requires the use of an *i.i.d* sample, since its proof relies on the classical Central Limit Theorem (CLT).

2.2.2 Convergence of Z-estimators under LHS

In this section, we extend the convergence properties of Z-estimators to LHS designs. The idea is to combine all the properties mentioned above. Indeed, we notice that the Z-function $\Psi_n(\boldsymbol{\theta})$ is written as an empirical mean of $\psi_{\boldsymbol{\theta}}$. Moreover, as mentioned in Subsection 1.1.3, the convergence of this kind of statistics under LHS holds. This is what we exploit here. In particular, a central limit theorem for Z-estimators under LHS is proposed.

As before, the covariance matrix of $\Psi_n^{LHS}(\boldsymbol{\theta})$ and $\Psi_n^{SRS}(\boldsymbol{\theta})$ are denoted by $\boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta})}$ and $\boldsymbol{\Sigma}_{\Psi_n^{SRS}(\boldsymbol{\theta})}$ respectively. We also introduce the term $\boldsymbol{R}_{\psi_{\boldsymbol{\theta}}} = \int_{[0,1]^p} \psi_{\boldsymbol{\theta}rem}(\boldsymbol{x})\psi_{\boldsymbol{\theta}rem}(\boldsymbol{x})^T d\boldsymbol{x}$, with $\psi_{\boldsymbol{\theta}rem}$ being defined as in Property 2.2. Let us now give some noteworthy convergence properties on $\Psi_n^{LHS}(\boldsymbol{\theta})$.

Property 2.5 *Let Θ be an open bounded subset of $\mathbb{R}^q$ and $\mathcal{X} = [0,1]^p$ $(q, p \in \mathbb{N}^*)$. Let $X^{LHS} = (\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n)})^T$ be the vector of LHS realizations of a random variable $\boldsymbol{X} = (X_1, \dots X_p)$ evolving in $\mathcal{X}$ such that $\boldsymbol{X} \sim U_{[0,1]^p}$. Assume also that, for all $\boldsymbol{\theta} \in \Theta$, $\psi_{\boldsymbol{\theta}}$ is a measurable function from $\mathcal{X}$ to $\mathbb{R}^q$. We then have the following properties on $\Psi_n^{LHS}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^n \psi_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})$:*

1. *If, for all $\boldsymbol{\theta} \in \Theta$, $\mathbb{E}(||\psi_{\boldsymbol{\theta}}(\boldsymbol{X}))||) < +\infty$, $\Psi_n^{LHS}(\boldsymbol{\theta})$ is an unbiased estimator of $\Psi(\boldsymbol{\theta}) = \mathbb{E}(\psi_{\boldsymbol{\theta}}(\boldsymbol{X}))$;*
2. *If, for all $\boldsymbol{\theta} \in \Theta$, $\mathbb{E}(||\psi_{\boldsymbol{\theta}}(\boldsymbol{X}))||^2) < +\infty$, we also have, when $n \to +\infty$:*

$$\boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta})} = \frac{1}{n}\boldsymbol{R}_{\psi_{\boldsymbol{\theta}}} + \frac{1}{n}o(1).$$

   Moreover, we have that $n(\boldsymbol{\Sigma}_{\Psi_n^{SRS}(\boldsymbol{\theta})} - \boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta})})$ is asymptotically positive semi-definite and that $\Psi_n^{LHS}(\boldsymbol{\theta})$ converges in quadratic mean to $\Psi(\boldsymbol{\theta})$. In other words, we have $\lim_{n\to+\infty} \mathbb{E}\Big(||\Psi_n^{LHS}(\boldsymbol{\theta}) - \Psi(\boldsymbol{\theta})||^2\Big) = 0$;
3. *If, for all $\boldsymbol{\theta} \in \Theta$, $\mathbb{E}(||\psi_{\boldsymbol{\theta}}(\boldsymbol{X}))||^3) < +\infty$ and if $\boldsymbol{R}_{\psi_{\boldsymbol{\theta}}}$ is non-singular, then $\sqrt{n}(\Psi_n^{LHS}(\boldsymbol{\theta}) - \Psi(\boldsymbol{\theta}))$ is asymptotically normal with mean 0 and covariance matrix equal to $\boldsymbol{R}_{\psi_{\boldsymbol{\theta}}}$.*

Proof. Let us show these properties one by one:

1. Since, for all $\boldsymbol{\theta} \in \Theta$, the function $\boldsymbol{x} \to \psi_{\boldsymbol{\theta}}(\boldsymbol{x})$ with $\boldsymbol{x} \in \mathcal{X}$ is measurable and $\mathbb{E}(||\psi_{\boldsymbol{\theta}}(\boldsymbol{X}))||) < +\infty$, $\Psi_n^{LHS}(\boldsymbol{\theta})$ is an unbiased estimator of $\Psi(\boldsymbol{\theta})$ by Property 2.1.
2. This is a direct consequence of Property 2.2.
3. This is a direct consequence of Theorem 2.1. ■

All these properties on $\Psi_n^{LHS}(\boldsymbol{\theta})$ allow to show the convergence of $\hat{\boldsymbol{\theta}}_n^{LHS}$. Indeed, the assertion 2 of Property 2.5 ensures the convergence in probability of $\Psi_n^{LHS}(\boldsymbol{\theta})$ to $\Psi(\boldsymbol{\theta})$. As mentioned before, Properties 2.3 and 2.4 do not impose any other conditions on the sampling scheme. We therefore have, under the conditions of application of at least one of these properties, that $\hat{\boldsymbol{\theta}}_n^{LHS}$ converges to $\boldsymbol{\theta}_0$ in probability. Let us now establish a central limit theorem for Z-estimators under LHS.

Theorem 2.3 *Let Θ be an open subset of an Euclidean space of dimension q with $q \in \mathbb{N}^*$ and $\mathcal{X} = [0,1]^p$ with $p \in \mathbb{N}^*$. Assume that, for all $\boldsymbol{\theta} \in \Theta$ and for all $\boldsymbol{x} \in \mathcal{X}$, the function $\boldsymbol{\theta} \to \psi_{\boldsymbol{\theta}}(\boldsymbol{x})$ is twice continuously differentiable in $\boldsymbol{\theta}$. The first order derivative is a size $q \times q$ matrix denoted $\dot{\psi}_{\boldsymbol{\theta}}$ and the second order partial derivative is a size $q \times q \times q$ tensor denoted $\ddot{\psi}_{\boldsymbol{\theta}}$. Let $X^{LHS} = (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)})^T$ be the vector of LHS realizations of a random variable $\boldsymbol{X} = (X_1, \ldots X_p)$ evolving in $\mathcal{X}$ such that $\boldsymbol{X} \sim U_{[0,1]^p}$.*

Suppose also that the following hypotheses are fulfilled:

1. $\forall n \in \mathbb{N}, \Psi_n(\hat{\boldsymbol{\theta}}_n^{LHS}) = \frac{1}{n}\sum_{i=1}^n \psi_{\hat{\boldsymbol{\theta}}_n^{LHS}}(\boldsymbol{x}^{(i)}) = 0$;
2. *there is $\boldsymbol{\theta}_0 \in \Theta$ such that $\mathbb{E}(\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})) = \Psi(\boldsymbol{\theta}_0) = 0$;*
3. $\mathbb{E}(||\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})||^2) < +\infty$;
4. $\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))$ *is nonsingular and such that* $\mathbb{E}(||\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X})||^2) < +\infty$;
5. *the function $\boldsymbol{\theta} \to \ddot{\psi}_{\boldsymbol{\theta}}(\boldsymbol{x})$ is dominated in norm by a fixed integrable function $\ddot{\psi}(\boldsymbol{x})$ for every $\boldsymbol{\theta}$ in the neighborhood of $\boldsymbol{\theta}_0$.*

Then, if $\hat{\boldsymbol{\theta}}_n^{LHS}$ is a consistent estimator of $\boldsymbol{\theta}_0$, we have:

$$(\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0) = -[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}\frac{1}{n}\sum_{i=1}^n \psi_{\boldsymbol{\theta}_0}(\boldsymbol{x}^{(i)}) + \frac{1}{\sqrt{n}}o_p(1).$$

In particular, we have that the covariance matrix of $\hat{\boldsymbol{\theta}}_n^{LHS}$, $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{LHS}} = [\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}\boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta_0})}[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-T} + \frac{1}{n}o(1)$, with $\boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta_0})} = \frac{1}{n}\boldsymbol{R}_{\psi_{\boldsymbol{\theta_0}}} + \frac{1}{n}o(1)$ asymptotically. Moreover, we have that $n(\boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{SRS}} - \boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{LHS}})$ is asymptotically positive semi-definite.

In addition, if the function $\psi_{\boldsymbol{\theta}_0}$ is such that $\mathbb{E}(||\psi_{\boldsymbol{\theta_0}}(\boldsymbol{X})||^3) < +\infty$ and if $\boldsymbol{R}_{\psi_{\boldsymbol{\theta_0}}}$ is non-singular, we have that the sequence $\sqrt{n}(\hat{\boldsymbol{\theta}}^{LHS} - \boldsymbol{\theta}_0)$ is asymptotically normal with mean zero and a covariance matrix equal to $[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}\boldsymbol{R}_{\psi_{\boldsymbol{\theta_0}}}[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-T}$.

Proof. The reasoning of this proof is adapted from the one given in (Van der Vaart, 1998) to demonstrate Theorem 2.2.

By Taylor's theorem, as $\Psi_n(\boldsymbol{\theta})$ is continuous and twice differentiable, $\exists\ \tilde{\boldsymbol{\theta}}_n^{LHS}$ between $\boldsymbol{\theta}_0$ and $\hat{\boldsymbol{\theta}}_n^{LHS}$ such that:

$$\Psi_n^{LHS}(\hat{\boldsymbol{\theta}}_n^{LHS}) = 0 = \Psi_n^{LHS}(\boldsymbol{\theta}_0) + \dot{\Psi}_n^{LHS}(\boldsymbol{\theta}_0)(\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0) + \frac{1}{2}(\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0)^T \ddot{\Psi}_n^{\ LHS}(\tilde{\boldsymbol{\theta}}_n^{LHS})(\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0).$$

We have, since $\mathbb{E}(||\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})||^2) < +\infty$ that $\Psi_n^{LHS}(\boldsymbol{\theta}_0) = \frac{1}{n}\sum_{i=1}^n \psi_{\boldsymbol{\theta_0}}(\boldsymbol{x}^{(i)})$ converges in probability to $\mathbb{E}(\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})) = 0$, thanks to Property 2.5: $\Psi_n^{LHS}(\boldsymbol{\theta}_0) \xrightarrow[n\to+\infty]{p} \mathbb{E}(\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})) = 0$.

Similarly, since $\dot{\Psi}_n^{LHS}(\boldsymbol{\theta})$ is the empirical mean of the matrix $\dot{\psi}_{\boldsymbol{\theta}}$ over $\mathrm{X}^{LHS} = (\boldsymbol{x}^{(1)} \dots \boldsymbol{x}^{(n)})^T$, then $\dot{\Psi}_n^{LHS} \xrightarrow[n\to+\infty]{p} \mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))$ (with $\mathbb{E}(||\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X})||^2) < +\infty$). In addition, thanks to assumption 4, we also have that $\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))$ is nonsingular.

The term $\ddot{\Psi}_n^{\ LHS}(\boldsymbol{\theta})$ corresponds to the empirical mean of the $q \times q \times q$ tensor $\ddot{\psi}_{\boldsymbol{\theta}}$ over $\mathrm{X}^{LHS} = (\boldsymbol{x}^{(1)} \dots \boldsymbol{x}^{(n)})^T$. By assumption 5, there is a ball $\mathcal{B}$ around $\boldsymbol{\theta}_0$ such that $\ddot{\psi}_{\boldsymbol{\theta}}$ is dominated in norm by $||\ddot{\psi}||$. Since we have $\hat{\boldsymbol{\theta}}_n^{LHS} \xrightarrow[n\to+\infty]{p} \boldsymbol{\theta}_0$, we also have $\mathbb{P}(\tilde{\boldsymbol{\theta}}_n^{LHS} \in \mathcal{B}) \xrightarrow[n\to+\infty]{} 1$. Moreover, if $\tilde{\boldsymbol{\theta}}_n^{LHS} \in \mathcal{B}$, we have:

$$||\ddot{\Psi}_n^{LHS}(\tilde{\boldsymbol{\theta}}_n^{LHS})|| = ||\tfrac{1}{n}\sum_{i=1}^n \ddot{\psi}_{\tilde{\boldsymbol{\theta}}_n^{LHS}}(\boldsymbol{x}^{(i)})|| \leq \tfrac{1}{n}\sum_{i=1}^n ||\ddot{\psi}(\boldsymbol{x}^{(i)})||.$$

Since $||\ddot{\psi}||$ is integrable and the right term is an empirical mean, it converges to a finite value thanks to Property 2.4. Hence it is the case for the left term.

So that, we can rewrite the Taylor's expansion as follow:

$$-\Psi_n^{LHS}(\boldsymbol{\theta}_0) = \Big[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X})) + o_p(1) + \frac{1}{2}(\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0)^T O_p(1)\Big](\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0).$$

Since $\hat{\boldsymbol{\theta}}_n$ converges in probability to $\boldsymbol{\theta}_0$, we have that $(\hat{\boldsymbol{\theta}}_n - \boldsymbol{\theta}_0)O_p(1)$ converges to 0 and thus:

$$-\Psi_n^{LHS}(\boldsymbol{\theta}_0) = \Big[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X})) + o_p(1)\Big](\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0).$$

This leads to the equation given in 2.3, considering that $\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0})$ is nonsingular and that we have asymptotically $\Psi_n^{LHS}(\boldsymbol{\theta}_0) = \frac{1}{\sqrt{n}}O_p(1)$:

$$(\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0) = -[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}\frac{1}{n}\sum_{i=1}^n \psi_{\boldsymbol{\theta}_0}(\boldsymbol{x}^{(i)}) + \frac{1}{\sqrt{n}}o_p(1).$$

Using this result, we directly have $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{LHS}} = [\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}\boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta_0})}[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-T}$. Moreover, we have that $n(\boldsymbol{\Sigma}_{\Psi_n^{SRS}(\boldsymbol{\theta_0})} - \boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta_0})})$ is asymptotically positive semi-definite thanks to Property 2.5. Thus, this is also the case for $n(\boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{SRS}} - \boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{LHS}})$.

Finally, if we suppose that $\mathbb{E}(||\psi_{\theta_0}(\boldsymbol{X})||^3) < +\infty$ and that $\boldsymbol{R}_{\psi_{\boldsymbol{\theta}_0}}$ is nonsingular, we have thanks to the assertion 3 of Property 2.5 that $\sqrt{n}(\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0)$ is asymptotically normal with mean zero and a covariance matrix equal to $[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}\boldsymbol{R}_{\psi_{\boldsymbol{\theta}_0}}[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}$. ■

Recall that, even if the results of Property 2.5 and Theorem 2.3 are given for $\boldsymbol{X} \sim U_{[0,1]^p}$ for simplicity, they can be easily generalized using the quantile transformation (Devroye, 1986).

These results give an asymptotic convergence for $\hat{\boldsymbol{\theta}}_n^{LHS}$ with, in the univariate case, a lower asymptotic variance of estimation than $\hat{\boldsymbol{\theta}}_n^{SRS}$ (corresponding to $n(\boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{SRS}} - \boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{LHS}})$ being asymptotically positive semi-definite in the multivariate case). Moreover, it gives a central limit theorem for Z-estimators under LHS. While strong regularity conditions on $\psi_{\boldsymbol{\theta}}$ are needed for these results to be valid, it remains very useful in many practical cases (eg. for the estimation of maximum likelihood). In the next subsection we give an example of application.

2.2.3 Application for the parameters estimation of generalized linear models: the example of the logistic regression

In the context of the statistical analysis of a computational code, it is common that one wants to approximate its outputs with a regression model. This is an important step in the methodology presented in this work (see Section 4.2 for more details). If the estimation of the modeling parameters can be expressed as a Z-estimator and the other conditions of use are satisfied, Theorem 2.3 ensures that the estimation variance of these parameters is asymptotically lower under LHS than under SRS. It also provides a central limit theorem under LHS.

Consider the case of generalized linear models (GLM), proposed in (Nelder and Wedderburn, 1972). They were formulated as a way of unifying various statistical models, including linear regression, logistic regression and Poisson regression. To estimate the parameters of a GLM, one generally uses a Maximum Likelihood Estimator (MLE). It is therefore a special case of Z-estimation. Thus, the results presented above can be applied to parameters estimation of a GLM. To illustrate this, we focus here logistic regression, which is a simple example of GLM.

Logistic regression or logit model is a binary regression model. This model aims to explain a binary variable Z with a covariable $\boldsymbol{X}$ evolving in a measurable space $\mathcal{X} \subset \mathbb{R}^d, d \in \mathbb{N}^*$. For simplicity and without loss of generality we assume that $\boldsymbol{X} \sim U_{[0,1]^d}$. Logistic Regression is widely used as a classification algorithm in machine learning.

Z is supposed to follow a Bernoulli distribution such that $P(Z = 1|\boldsymbol{X} = \boldsymbol{x}, \boldsymbol{\theta})$ is a function of $\boldsymbol{x} \in \mathbb{R}^d$ and the modeling parameter is $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)^T \in \Theta$ with Θ being an open bounded subset of $\mathbb{R}^d$. This function, called $p_{\boldsymbol{\theta}}(\boldsymbol{x})$, is defined as follow:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\sum_{j=1}^{d} x_j \theta_j}} = \frac{1}{1 + e^{-\boldsymbol{x}\boldsymbol{\theta}}}.$$

Let $\mathbf{X} = \left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\right)^T$ be the n realizations of X and $\mathbf{Z} = \left(z^{(1)}, \ldots, z^{(n)}\right)^T$ be the n realizations modeled by the logistic regression.

Suppose that one wants to estimate the optimal value $\boldsymbol{\theta_0} = (\theta_{0,1}, \ldots, \theta_{0,d})^T$ of the parameter $\boldsymbol{\theta}$ regarding the observations of $\boldsymbol{x}$ and $\mathbf{Z}$. The likelihood of an observation $z^{(i)}$ with $i \in [\![1, n]\!]$ and a vector of parameters $\boldsymbol{\theta}$ is

$l(z^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta}) = p_\theta(\boldsymbol{x}^{(i)})^{z^{(i)}}(1 - p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}))^{1-z^{(i)}}$.

If the n realizations are independent, the global likelihood of the observations corresponds to $L(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^{n} l(z^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta})$. The maximum likelihood estimator corresponds then to $\hat{\boldsymbol{\theta}}_n = \underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmax}}(L(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}))$. Since the log function is strictly increasing on $]0, +\infty[$, we also have that $\hat{\boldsymbol{\theta}}_n = \underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmax}}(\sum_{i=1}^{n} \log\left[l(z^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta})\right])$. The log-likelihood can be written as follow:

$\log\left[L(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta})\right] = \sum_{i=1}^{n} \log\left(l(z^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta})\right)$

$= \sum_{i=1}^{n} \left[z^{(i)} \log(p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})) + (1 - z^{(i)}) \log(1 - p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}))\right]$

$= \sum_{i=1}^{n} \left[z^{(i)}\boldsymbol{x}^{(i)}\boldsymbol{\theta} - \log(1 + e^{\boldsymbol{x}^{(i)}\boldsymbol{\theta}})\right]$.

Since $\log[l(z^{(i)}|x^{(i)}, \boldsymbol{\theta})]$ is differentiable in $\boldsymbol{\theta}$, the maximum likelihood estimator $\hat{\boldsymbol{\theta}}_n$ is a solution of the vectorial equation:

$$\sum_{i=1}^{n} \nabla \log[l(z^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta})] = 0 \tag{2.4}$$

with $\nabla \log[l(z^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta}) = (\frac{\partial \log[l(z^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta})]}{\partial \theta_1}, \dots, \frac{\partial \log[l(z^{(i)}|\boldsymbol{x}^{(i)}, \boldsymbol{\theta})]}{\partial \theta_d})^T$.

The quantity $\hat{\boldsymbol{\theta}}_n$ is therefore a Z-estimator of $\boldsymbol{\theta_0}$ as defined in Equation (2.3), with $\psi_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) = (\psi_{\theta_1}(\boldsymbol{x}^{(i)}), \dots, \psi_{\theta_d}(\boldsymbol{x}^{(i)}))^T = \nabla \log[l(z^{(i)}|\boldsymbol{x}^{(i)}, \theta)$. Here we have, for $j \in [\![1, d]\!]$, $\psi_{\theta_j}(\boldsymbol{x}^{(i)}) = \frac{\partial \log[l(z^{(i)}|\boldsymbol{x}^{(i)}, \theta_j)]}{\partial \theta_j} = x_j^{(i)}\Big(z^{(i)} - p_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})\Big)$.

We can see that the estimation of the parameters of a logistic regression by MLE fits into the framework of Z-estimation. Thus, let us suppose that the observations of X are obtained by a LHS. We consider the Z-estimator defined by the equation 2.4, even if in this case the realization are not independent anymore. Let us discuss the convergence of this estimator under LHS.

Let $\mathrm{X}^{LHS} = (\boldsymbol{x}^{(i)}, \dots \boldsymbol{x}^{(n)})^T$ be the realizations of $\boldsymbol{X}$ generated by a LHS. If we suppose that $\boldsymbol{X}$ and $\boldsymbol{\theta}$ are different from zero and since they are bounded, we have that $\Psi_n^{LHS} = \sum_{i=1}^n \psi_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)})$ converges in probability to $\mathbb{E}(\psi_{\boldsymbol{\theta}}(\boldsymbol{X})) = \Psi(\boldsymbol{\theta})$. As we have seen, the other conditions concerning the convergence of $\hat{\boldsymbol{\theta}}_n$ to $\boldsymbol{\theta}_0$ are not specific to the type of DoE. The conditions of application of Property 2.3 (resp 2.4) are verified both in the case of an SRS or a LHS design. We can thus conclude, thanks to Property 2.3 that $\hat{\boldsymbol{\theta}}_n$ converges in probability in $\boldsymbol{\theta}_0$.

Let us now verify that the conditions of application of theorem 2.3 are fulfilled. First, we see that Ψ_n is continuous and infinitely differentiable in $\boldsymbol{\theta}$. Plus, $\mathbb{E}(\psi_{\boldsymbol{\theta}_0}) = 0$ by construction.

We also have, for $j, j' \in [\![1, d]\!]$, that $\frac{\partial \psi_{\boldsymbol{\theta}_{0,j}}(\boldsymbol{x})}{\partial \theta_{j'}} = \frac{x_j x_{j'} e^{-\boldsymbol{x}\boldsymbol{\theta_0}}}{(1+e^{-\boldsymbol{x}\boldsymbol{\theta_0}})^2}$. Thus, we have that the matrix of partial derivatives $\dot{\psi}_{\boldsymbol{\theta}_0}$ is such that $\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(X))$ is defined and nonsingular since $\boldsymbol{X} \neq 0$. Since the values of $\boldsymbol{X}$ and $\boldsymbol{\theta}$ are bounded in norm, the function $\psi_{\boldsymbol{\theta}_0}$ is bounded and thus $\mathbb{E}(||\psi_{\boldsymbol{\theta}_0}(X)||^3) < +\infty$. Finally, we have that $||\ddot{\psi}_{\boldsymbol{\theta}}(\boldsymbol{x})||$ can be bounded by an integrable function since the values of $\boldsymbol{\theta}$ and $\boldsymbol{X}$ are bounded.

All of these allows us to apply Theorem 2.3. We therefore have that the covariance matrix of estimation of $\hat{\theta}_n^{LHS}$ is equal to $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{LHS}} = [\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1} \boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta_0})} [\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-T}$ and that $n(\boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{SRS}} - \boldsymbol{\Sigma}_{\hat{\boldsymbol{\theta}}_n^{LHS}})$ is asymptotically positive semidefinite. Note that we have, with the previously introduced notations, $\boldsymbol{\Sigma}_{\Psi_n^{LHS}(\boldsymbol{\theta_0})} = \frac{1}{n}(\boldsymbol{R}_{\psi_{\boldsymbol{\theta}_0}} + o(1))$ asymptotically.

Finally, if we suppose that $\mathbb{E}(||\psi_{\boldsymbol{\theta}_0}(\boldsymbol{X})||^3) < +\infty$, we have that $\sqrt{n}(\hat{\boldsymbol{\theta}}_n^{LHS} - \boldsymbol{\theta}_0)$ is asymptotically normal with mean zero and a covariance matrix equal to $[\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1} \boldsymbol{R}_{\psi_{\boldsymbol{\theta}_0}} [\mathbb{E}(\dot{\psi}_{\boldsymbol{\theta}_0}(\boldsymbol{X}))]^{-1}$, assuming that $\boldsymbol{R}_{\psi_{\boldsymbol{\theta}_0}}$ is non-singular.

A similar reasoning can be made regarding the maximum likelihood estimation of other models belonging to the GLM family. However, it will be necessary to check the applicability conditions of Theorem 2.3.

2.3 Some other statistics under LHS

In the previous section, we established convergence properties for the class of Z-estimators. However, within the presented work, some other statistics have to be estimated. The objective of this section is to discuss their convergence under LHS. A complete presentation of each of these tools is made in the next chapters. Thus, we first discuss the Kolmogov statistic (see Chapter 3 for more details). Then, we focus on the estimation of the first order Sobol' indices by the Chatterjee method (Gamboa et al., 2022) (see Chapter 4 for more details on Sobol' indices).

2.3.1 Kolmogorov-Smirnov statistic under LHS

The Kolmogorov–Smirnov (KS) statistic (Shiryayev, 1992) is a rescaled distance between a known reference continuous distribution and the sample under study. A goodness-of-fit test is built upon this statistic. In this section, we focus on the convergence of the KS statistic under LHS. For more information about this statistic and its use, one can refer to Section 3.1 of this manuscript.

Let X be a continuous random variable with values in a set $\mathcal{X} \subset \mathbb{R}$. Let $F : \mathbb{R} \to [0,1]$ be the cumulative distribution function of X. We define $\boldsymbol{x} = (x^{(1)}, \dots, x^{(n)})^T$ as a sampling of X. Thus, the empirical cumulative distribution function of X is $F_n(t) = \sum_{i=1}^n \mathbf{1}_{x^{(i)} \leq t}, t \in \mathbb{R}$. Here, the notation $\mathbf{1}$ corresponds to the indicator function. We also define $\tilde{\boldsymbol{x}} = (x^{(\sigma(1))}, \dots, x^{(\sigma(n))})^T$ as the ordered realizations of X such as $x^{(\sigma(1))} \leq \dots \leq x^{(\sigma(n))}$.

The KS statistic is equal to $S_n^{KS}(F_n, F) = \sqrt{n} D_n(F_n, F)$. Here,

$D_n(F_n, F) = \max_{1 \leq i \leq n}[\max(|\frac{i}{n} - F(x^{(\sigma(i))})|, |F(x^{(\sigma(i))}) - \frac{i-1}{n}|)]$.

Most of the theoretical results concerning the convergence of D_n and the Kolmogorov statistic are only true if $\boldsymbol{x} = (x^{(1)}, \dots, x^{(n)})^T$ is an $i.i.d$ sample of X. A summary of these results is given in Section 3.1. The goal here is to find the distribution function of $D_n^{LHS}(F_n, F)$ under LHS.

Let $\boldsymbol{x}^{LHS} = \left(x^{(1)}, \dots, x^{(n)}\right)$ be realizations of X generated by a LHS and $\tilde{\boldsymbol{x}}^{LHS}$ be the ordered realizations of $\boldsymbol{x}^{LHS}$. We have, by definition of LHS, that $(F(x^{(\sigma(1))}), \dots F(x^{(\sigma(n))}))^T = (\frac{i-u^{(\sigma(1))}}{n}, \dots, \frac{i-u^{(\sigma(n))}}{n})^T$. Here, $(u^{(\sigma(1))}, \dots, u^{(\sigma(n))})^T$ is a $i.i.d$ sample of the uniform distribution $U_{[0,1]}$.

Thus the distance D_n^{LHS} can be written as follow:

$$D_n^{LHS}(F_n, F) = \max_{1 \leq i \leq n}[\max(|\frac{i}{n} - \frac{i - u^{(\sigma(i))}}{n}|, |\frac{i - u^{(\sigma(i))}}{n} - \frac{i-1}{n}|)]$$

$$= \max_{1\leq i\leq n}[\max(|\frac{u^{(\sigma(i))}}{n}|, |\frac{1-u^{(\sigma(i))}}{n}|)].$$

Moreover, since $u^{(\sigma(i))} \sim U_{[0,1]}$ for all $i \in [\![1, n]\!]$, $w^{(i)} = \max(|\frac{u^{(\sigma(i))}}{n}|, |\frac{1-u^{(\sigma(i))}}{n}|)$ follows uniform distribution $U_{[1/2n,1/n]}$.

Let us now find the cumulative distribution function of $D_n^{LHS}(F_n, F)$. By definition, $F_{D_n^{LHS}}(t) = \mathbb{P}(D_n^{LHS} \leq t), t \in \mathbb{R}$. Thus, $F_{D_n^{LHS}}(t) = \mathbb{P}(D_n^{LHS} \leq t) = \mathbb{P}(\max_{1\leq i\leq n}(w^{(i)}) \leq t)$. Since the $(w^{(1)}, \ldots, w^{(n)})$ are *i.i.d*, we have: $F_{D_n^{LHS}}(t) = \prod_{i=1}^{n} \mathbb{P}(w^{(i)} \leq t)$.

The cumulative distribution function of $D_n^{LHS}(F_n, F)$ can therefore be written: $F_{D_n^{LHS}}(t) = \prod_{i=1}^{n} \frac{t-1/2n}{1/2n} \mathbf{1}_{[1/2n,1/n]}(t) = (2nt-1)^n \mathbf{1}_{[1/2n,1/n]}(t)$ (with $F_{D_n^{LHS}}(t) = 1$ for $t > 1/n$)

Finally, the corresponding density is $f_{D_n^{LHS}}(t) = 2n^2(2nt-1)^{n-1}\mathbf{1}_{[1/2n,1/n]}(t), t \in \mathbb{R}$. we can note that the values taken by D_n^{LHS} are bounded between $1/2n$ and $1/n$. This is due to the stratified character of LHS samples.

A plot of this density function for $n = 10$ is given on Figure 2.5. We also plot the histogram of D_n^{SRS} for comparison. It was estimated using $L = 1000$ SRS designs. It is interesting to observe that the values taken by D_n^{LHS} are overall much smaller than the D_n^{SRS} ones.

One can build a Kolmogorov test under LHS using these results by replacing the cumulative distribution of the statistics under SRS by $F_{D_n^{LHS}}$. This work was done because the Kolmogorov-Smirnov statistic is used in Chapter 3 of this thesis and the experimental design used in our industrial case is an LHS. Nevertheless, our case is a bit different from the one presented here (see Section 3.1 for more details).

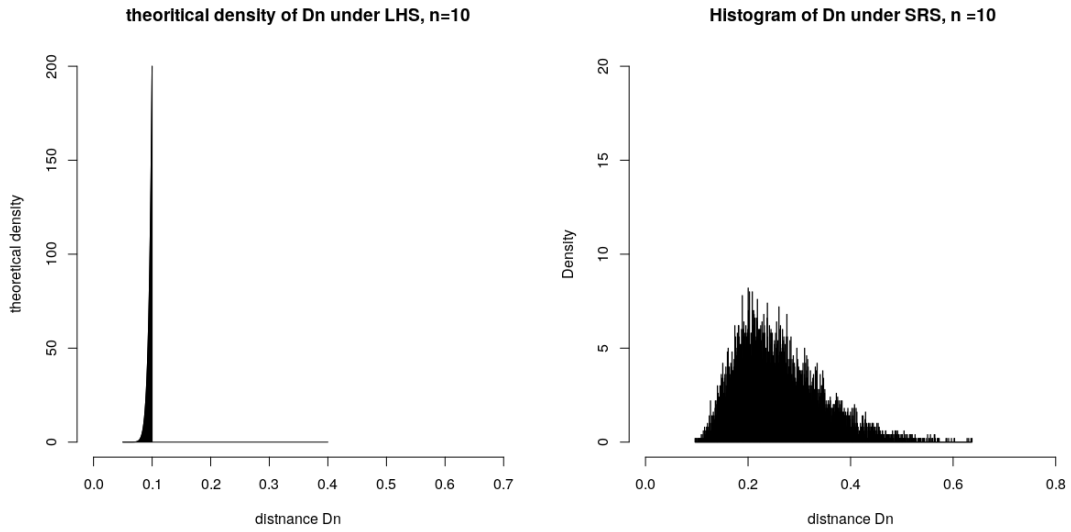

Figure 2.5: *Theoretical density of* D_n^{LHS} *for* $n = 10$ *on the left. Empirical histogram of* D_n^{SRS} *for* $n = 10$ *on the right.*

2.3.2 Sobol' first order indices: Chatterjee estimator under LHS

Variance-based indices are often used for sensitivity analysis in the framework of uncertainty quantification. These tools are used in the screening step and sensitivity analysis step of our methodology. For a complete overview, see Section 4.1 of this manuscript. Here we focus on the convergence of an estimator of some of these indices, called first-order Sobol' indices, under LHS. Introduced by F.Gamboa (Gamboa et al., 2022) following the work of Chatterjee (Chatterjee, 2020), this estimator is based on Rank Statistic (RS). This estimator is used in Chapter 4 to evaluate the first order indices of MC3D inputs in the screening step of our methodology.

Let us consider again a vector of independent inputs $\boldsymbol{X} = (X_1, \ldots, X_d)^T$ evolving in a measurable space $\mathcal{X} \subset \mathbb{R}^d$ and a measurable function f such that $Y = f(\boldsymbol{X})$, $Y \in \mathcal{Y}$, with $\mathcal{Y}$ being a measurable space of dimension 1. We also suppose that $\mathbb{E}(f(\boldsymbol{X})^2) < +\infty$. For an input $X_j, j \in [\![1, d]\!]$, the first order Sobol' index S_j is defined as follow:

$$S_j = \frac{\mathbb{V}\mathrm{ar}\Big(\mathbb{E}(Y|X_j)\Big)}{\mathbb{V}\mathrm{ar}(Y)}. \tag{2.5}$$

Let $((x_j^{(1)}, y^{(1)}), \ldots, (x_j^{(n)}, y^{(n)}))$ be a series of n *i.i.d* observations of $(X_j, Y), j \in [\![1, d]\!]$. If we note $\sigma_j(i)$ the rank of the realization $x_j^{(i)}$ such that $x_j^{(\sigma_j(1))} \leq \ldots \leq x_j^{(\sigma_j(n))}$, we can define $N_j(i)$ by (2.6):

$$N_j(i) = \begin{cases} \sigma_j^{-1}(\sigma_j(i)+1) \text{ if } \sigma_j(i) < n; \\ \sigma_j^{-1}(1) \text{ if } \sigma_j(i) = n. \end{cases} \tag{2.6}$$

We denote by $(y^{(N_j(1))}, \ldots, y^{(N_j(n))})$ the permuted sample through the permutation function N_j. From this, Gamboa et al. (2022) proposed an estimator of the first order Sobol' index S_j defined by the following formula (2.7):

$$\widehat{S_j^n}^{RS} = \frac{\frac{1}{n}\sum_{i=1}^n y^{(i)} y^{(N_j(i))} - (\frac{1}{n}\sum_{i=1}^n y^{(i)})^2}{\frac{1}{n}\sum_{i=1}^n (y^{(i)})^2 - (\frac{1}{n}\sum_{i=1}^n y^{(i)})^2}. \tag{2.7}$$

The aim of this subsection is to investigate the convergence of the statistic $\widehat{S_j^n}^{RS}$ under LHS and other sampling methods. For this, two numerical tests are processed with analytical functions classically used in GSA. Three types of DoE are compared:

- Simple random sampling (SRS);
- Classic Latin Hypercube Sampling (LHS) ;
- LHS designs optimizing a space-filling criterion, the star discrepancy (Damblin et al., 2013).

Note that even if the observations are not *i.i.d* under LHS and optimized LHS, we always consider the same estimator $\widehat{S_j^n}^{RS}$.

2.3.2.1 Test 1: Design comparison based on usual analytical functions in GSA

For the first numerical study, we compare the performances of the estimation procedure depending on the DoE type on three classical analytic functions used for GSA: the Ishigami function (Saltelli, 2008), the Sobol' function (Saltelli, 2008) and the Morris function (Morris, 1991).

Here, we only display the results of the Sobol' function. The results for the Ishigami function and the Morris function can be found in Appendix A. The Sobol' function (Saltelli, 2008) is defined in dimension $d = 8$ as follow:

$$f_S(X_1, \ldots, X_8) = \prod_{j=1}^{8} \frac{|4X_j - 2| + a_j}{1 + a_j}$$

with:

- $X_j, j \in [\![1, 8]\!]$ following uniform distributions in $[0, 1]$;
- $a_j \in \boldsymbol{a} = [0, 1, 4.5, 9, 99, 99, 99, 99]$ with $j \in [\![1, 8]\!]$.

The true Sobol' first order indices values for the Sobol' function are: $\boldsymbol{S} = [0.759, 0.146, 0.025, 0.003, 0, 0, 0, 0]$

For each input $X_j, j \in [\![1, 8]\!]$ of the test function three metrics are used to compare the convergence behavior of the estimator regarding the different sampling methods. The first one is the square bias, $\text{Bias}^2(\widehat{S_j^n}^{METHOD}) = (\mathbb{E}(\widehat{S_j^n}^{METHOD} - S_j))^2$ with S_j being the true value of the Sobol' index $j \in [\![1, 8]\!]$. The second one is the variance of estimation, $\mathbb{V}\text{ar}(\widehat{S_j^n}^{METHOD})$. The last one is the sum of the other two, the Mean Square Error (MSE): $MSE(\widehat{S_j^n}^{METHOD}) = \text{Bias}^2(\widehat{S_j^n}^{METHOD}) + \mathbb{V}\text{ar}(\widehat{S_j^n}^{METHOD})$. For each sampling method, the average value of these three metrics is computed over $L = 500$ independent designs with sample sizes n being from 20 to 100 (by steps of 20).

Figure 2.6 shows the evolution of the bias², the variance and the MSE for the Sobol' index estimator of the inputs of the Sobol' function. For sake of brevity, results are only displayed for X_1, X_4 and X_8 which corresponds respectively to the min, median and max values of the true Sobol' indices.

As expected, the estimators converge for all three designs. We observe a slightly lower average variance for the classic LHS design compared to SRS, for a similar square bias. Globally, the variance of estimation is even lower under optimized LHS, at a cost of a higher square bias. Optimal LHS-type structured designs are by their nature more deterministic and therefore have a lower variance of estimation, so their structure may induce a potentially significant bias depending on the test function. The MSE is equivalent for the three designs on average. The results are similar for the three functions tested.

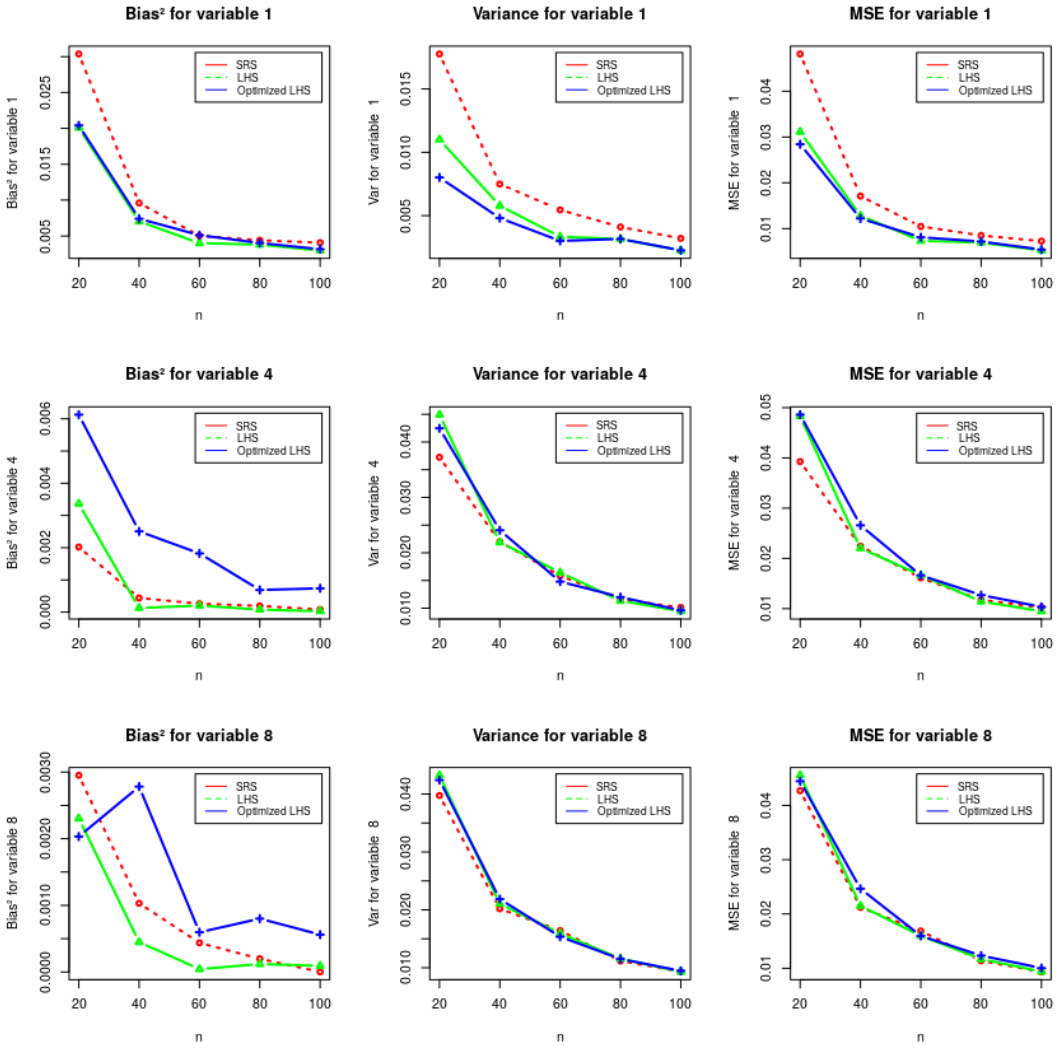

Figure 2.6: *Sobol' model function: square bias, variances and MSE of the estimators of first Sobol' indices, according to the numerical experiment design for 3 inputs (those with max, median and min indices values), according to the design experiment type.*

2.3.2.2 Test 2: comparison based on a randomized version of the Ishigami function

To broaden the base of test functions tested, we propose to consider the Ishigami function with a randomization of its parameters. Let us consider the Ishighami's function:

$$f_I(X_1, X_2, X_3) = \sin(X_1) + A\sin^2(X_2) + BX_3^4\sin(X_1)$$

where A and B are given parameters of the function, and the three random variables X_1, X_2 and X_3 are independent and uniformly distributed in $[-\pi, \pi]$.

The behavior of this function highly depends on its two parameters A and B. A reparametrization is considered here with B and the ratio $C = B/A$. The idea is therefore to draw randomly a large number of pairs (B,C) in order to observe the behavior of the estimator $\widehat{S_j^n}^{RS}$ in a large number of different configurations.

For each comparison between estimators from structured designs and pure Monte Carlo ones, a set of parameters (B,C) is randomly drawn. The choice of the distribution followed by (B,C) relies on the variance of the Ishigami function and its variance decomposition. More precisely, variance and terms of the decomposition are given by:

- $\mathbb{V}\text{ar}(f_I(X_1, X_2, X_3)) = \frac{1}{2} + \frac{A^2}{8} + \frac{B^2\pi^8}{18} + \frac{B\pi^4}{5}$;
- $V_1 = \mathbb{V}\text{ar}\Big(\mathbb{E}(f_I(X_1, X_2, X_3)|X_1)\Big) = \frac{1}{2}\left(1 + \frac{B\pi^4}{5}\right)$;
- $V_2 = \mathbb{V}\text{ar}\Big(\mathbb{E}(f_I(X_1, X_2, X_3)|X_2)\Big) = \frac{A^2}{2}$;
- $V_{13} = \mathbb{V}\text{ar}\Big(\mathbb{E}(f_I(X_1, X_2, X_3)|X_1, X_3)\Big) = \frac{8B^2\pi^8}{225}$;
- $V_3 = V_{12} = V_{123} = 0$.

The corresponding Sobol' indices are easily computable by dividing them by the total variance of the function. We notice that the value $V_1 = 0$ is possible when B is equal to $\frac{-5}{\pi^4} \approx -0.05$. We also remark that $A >> B$ leads to high values of V_2 while $B >> A$ leads to low values of V_2. Therefore, we have chosen the following distributions for B and C:

$$f : \left\{ \begin{array}{l} B \sim U_{[-1,1]} \\ C = B/A = 10^D \text{ with } D \sim U([-2, 2]). \end{array} \right. \tag{2.8}$$

The density of C is $f_C(t) = \frac{1}{4ln(10)t}\mathbf{1}(t)_{[10^{-2},10^2]}$ with $t \in \mathbb{R}$ and $A = B \times C$ so $\mathbb{E}(A) = 0$ and $\mathbb{V}\text{ar}(A) = \frac{10^4-10^{-4}}{24\ ln(10)}$.

With these ranges of variations for A and $C = B/A$ in the Ishigami function, there is a great amount of different configurations taken into account. As in test 1, we compare over $L = 500$ repetitions of the *average* square bias, variance and the mean square error (MSE) with the sampling sizes n growing ($20 \leq n \leq 100$ in steps of 20). The complete test procedure is described in detail in Algorithm 1.

Algorithm 1 Procedure of Test 2: Comparison based on a randomized version of the Ishigami function

Require: number of repetitions L

for $n = 20$ to $n = 100$ in steps of 20 **do**
  for $l = 1$ to L **do**
    Draw a couple (A,B) according to the previously established distributions (2.8).
    for each sampling method (SRS, LHS, optimized LHS) **do**
      Generate a design of size n and dimension $d = 3$ and run the model function on each sample.

      For each of the $d = 3$ inputs, compute $\widehat{S_j^n}^{METHOD}$ using equation (2.7).

    end for
  end for
  Compute the bias, the variance and the MSE of estimation
end for
For each of the $d = 3$ inputs and each sampling method, plot the *average* square bias, variance and MSE as a function of n

Note that the term *average* square bias (resp variance and MSE) is a bit imprecise. For this to be accurate, it would be necessary, for each value of (A, C), to compute square bias (resp variance and MSE) and then average the results. However, the obtained results would be similar to those presented, thanks to the law of total expectancy.

Figure 2.7 shows the evolution of the *average* square bias, variance and MSE of the Sobol' index estimator of the three inputs of the Ishigami function. The results of this test are similar to Test 1. The MSE appears to be equivalent for the three design types. If we look closely at the graphs, we can observe that the variance of estimation under optimized LHS tends to be better than under SRS and LHS for high valued Sobol' first order indices. However, they have more bias[2]. Therefore,

they are less efficient for low valued indices where bias^2 error is more detrimental (in terms of index value). In summary, classic LHS designs again offer a good compromise, regardless of the theoretical value of the Sobol' indices.

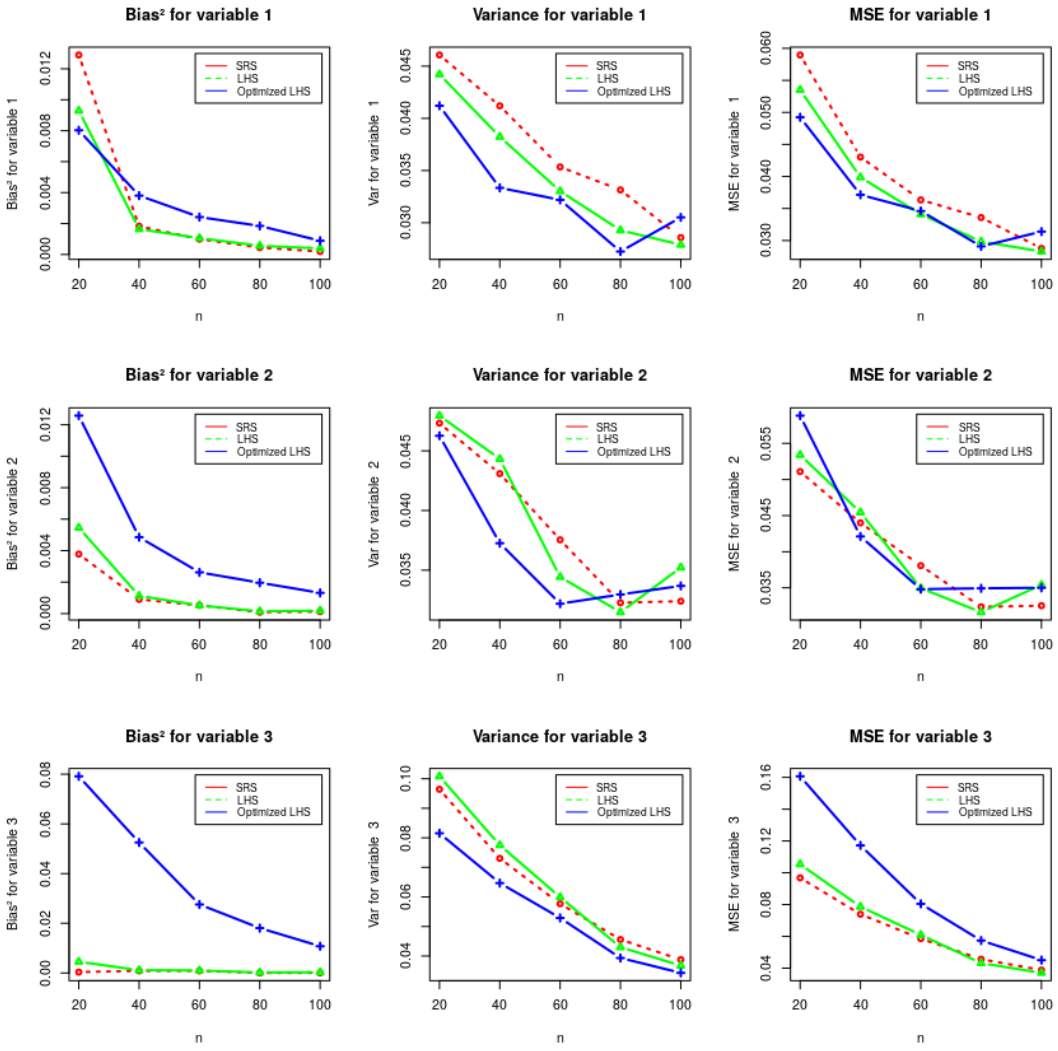

Figure 2.7: *Randomized version of the Ishigami function: square bias, variances and MSE of the estimators of 1rst Sobol' indices, for the three inputs, according to the design experiment type.*

2.3.3 Discussion on the convergence of other statistics used in the methodology

In addition to the Kolmogorov statistic and the first-order Sobol' indices, two other statistics are estimated under LHS in this work. First, there is the Hilbert-Schmidt Independence Criterion (HSIC). In the context of a sensitivity analysis, this criterion is used as a dependence measure to distinguish the inputs that influence the outputs of the code. For this, independence test built upon HSIC statistics are used. A complete presentation of this criterion and associated independence test can be found in Section 3.2. As it is often the case, most of the convergence properties of this statistic were established under the $i.i.d$ sampling assumption. However, a numerical study proposed in El Amri and Marrel (2021) seems to indicate that using an LHS instead of an SRS does not change the distribution of the HSIC statistic under the hypothesis of independence.

Gaussian Process (GP) regression is also performed. This regression method is introduced in detail in Chapter 4. The optimization of GP regression parameters is usually performed by maximum likelihood. These parameters are also estimated under LHS in this work. As we saw in Section 2.2, maximum likelihood estimators can be written as Z-estimators. An interesting perspective is to identify whether Theorem 2.3 presented in Section 2.2 can be applied to GP parameters estimation, and if so, under what conditions.

2.4 Conclusion of the chapter and prospects

In this chapter, sampling methods were discussed. In the framework of uncertainties quantification, they constitute an essential preliminary step to the analysis. Indeed, it is through the use of these sampling methods that one establishes the input-output data set. This data set therefore allows the exploration of the inputs space of the studied code but also often constitute a learning sample for metamodeling step. In Section 2.1, an overview of classical sampling methods used to explore the inputs space of a simulation code was given. We also justified our choice to use one of these methods, the Latin Hypercube Sampling (LHS). Indeed, it presents a good compromise between space-filling and theoretical convergence properties.

The asymptotic convergence of most statistics has been established under Simple Ramdom Samplings (SRS), which are $i.i.d$ (independent and identically distributed). The other sections of this chapter have been dedicated to the study of the convergence of different estimators under LHS. Thus, in Section 2.2, a work done on the theoretical convergence of Z-estimators under LHS has been presented. A reduction of the asymptotic estimation variance as well as a central limit theorem

for this class of estimators under LHS has been established. Nevertheless, some restrictive regularity conditions have been imposed on these estimators to establish this convergence. A possible perspective on this work could be to lift some of these restrictions, such as the existence of the $Z-$function second derivative for instance.

In Section 2.3, we discussed the convergence under LHS of the other statistics used to carry out the work presented in this manuscript. Some theoretical work on Kolmogorov statistic was presented. This work constitutes an important step in the theoretical construction of a Kolmogorov test under LHS. Numerical experiments have also been performed to study the convergence of a rank based estimator of Sobol' first order indices, recently proposed. They allowed us to empirically compare the asymptotic behavior of this estimator under Simple Random Sampling (SRS), under LHS and under optimized LHS. According to this work, classical LHS seems to be a good compromise for an efficient rank based estimation of first order Sobol' indices.

Finally, the convergence under LHS of two other statistics used in this work are discussed, namely the estimator of the Hilbert Schmidt Independence Criterion (HSIC) and of the parameters estimators of Gaussian Process (GP) regression models. No theoretical results have been established concerning these statistics under LHS. However, promising empirical studies have been proposed regarding HSIC estimator convergence. Moreover, the estimation of GP regression parameters is performed through maximum likelihood, which is a special case of Z-estimation. Another perspective of this work could be therefore to study theoretically the convergence of these statistics under LHS.

Chapter 3

Understanding code failures using sensitivity analysis tools

After defining the design of numerical experiments, the next step of code analysis in the Uncertainty Quantification (UQ) framework is classically to run the corresponding simulations with the code. The objective of this step is to explore the code behavior and more especially the inputs/outputs relationship. However, sometimes some of the code runs fail to converge.

These code failures can be due to numerical problems or suitability of the models used by the simulation code. Analyzing the occurrence of failures to understand which inputs have the most influence on these unexpected events allows a better understanding of how the code works. This will also provide valuable information on the values of the input parameters that can prevent code failures. Moreover, these failures may have an influence on the rest of the code study. It is therefore important to process this analysis in order to adapt the rest of the methodology. More generally, analysis of code failures intends to improve the robustness of simulation software and code computations. The objective of this chapter is thus to propose a method to analyze code failures and apply it to our industrial case.

This work was motivated by the results obtained during the exploration of the MC3D inputs space. Indeed, when we launched a first design of experiments, we realized that nearly a third of the simulations did not converge. These code failures on MC3D has motivated the study presented in this chapter. A preprint on this topic has been accepted by the Nuclear Science and Engineering Journal (Hakimi et al., 2022). This article gathers most of the methods presented here.

To perform the code failure analysis, we place ourselves in the framework of global sensitivity analysis (GSA). For this, we can consider failure occurrence as a

binary output coding the failure occurrence and apply on it GSA methods. The purpose of the latter is to determine how the variability of model's inputs affects the fluctuation of its output (see Chapter 4 for more details). Many methods have been developed for this purpose (Iooss and Lemaître, 2015 ; Da Veiga et al., 2021). However, most of classic tools used for sensitivity analysis, such as Sobol' indices (Sobol', 1993) or the Elementary Effect method (Brayer et al., 2020; Morris, 1991) are not well tailored for the case of code failures. Indeed, in this case the studied output is binary while most sensitivity analysis tools are designed to study continuous outputs. Furthermore, the use of some of these methods require a large number of simulations.

Recently, tools based on dependence measures have been proposed for global sensitivity analysis. These tools remove some of these limitations (Da Veiga, 2015). Among them, the Hilbert-Schmidt Independence Criterion (Gretton et al., 2005) denoted by HSIC, generalizes the notion of covariance between two random variables. The HSIC can be used with many different types of variables, including binary ones. Moreover, it fully characterizes the independence between two variables and its estimation only requires a limited budget of simulations. Last but not least, statistical tests of independence can be built upon HSIC measures.

In this context, we explored two methods to perform a variable selection (screening) by sensitivity analysis on code failures:

- a first approach is based on goodness-of-fit tests and compares the conditional probability distributions knowing the code failures to the initial one;
- a second approach, based on the HSIC, measures the global dependence between the inputs and the occurrence of code failures.

Then, a graphical analysis and a physical interpretation of the results is given. After this study, the sample considered in the rest of the methodology is conditioned by the absence of code failures. It is therefore important to have an idea of the eventual dependence between the inputs sampling induced by this conditioning. The final section of this chapter is devoted to the study of the the interdependence between the inputs samples knowing the code failures through the HSIC.

Since in our industrial case code runs are very computationally expensive, the proposed method only relies on the inputs/outputs sample used for the whole methodology.

Let us now introduce the notations used in this chapter. As a reminder, a simulation code like MC3D can be viewed as a measurable model function f linking the inputs to one (or more) output(s):

$$f : \begin{array}{l} \mathcal{X} \to \mathcal{Y} \\ \mathbf{X} \longmapsto f(\boldsymbol{X}) = Y. \end{array}$$

Here,

- $\boldsymbol{X} = (X_1, ...X_d)$ is the vector containing the d random variables of the uncertain inputs evolving in a measurable space denoted by $\mathcal{X}$;
- Y is the vector of the code outputs evolving in a measurable space denoted by $\mathcal{Y}$.

We are studying in this chapter the quantity of interest Z representing the occurrence of code failures. It is defined by the function $g_Z(\boldsymbol{X}) : \mathcal{X} \to \{0, 1\}$:

$$Z = g_Z(\mathbf{X}) = \begin{cases} 1 & \text{if the model } f(\mathbf{X}) \text{ fails} \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

In this section, the observed sample is $(\mathbf{X}, \mathbf{Z})$, where:

- $\mathbf{X} = \left(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}\right)^T$ with $\boldsymbol{x}^{(i)} = \left(x_1^{(i)}, \ldots, x_d^{(i)}\right)$ denotes, as mentioned in the first chapter of the manuscript, the matrix containing a sample of size n also called the Design of Experiments (DoE), $\mathbf{X} \in M_{n,d}(\mathbb{R})$;
- $\boldsymbol{x}_j = \left(x_j^{(1)}, \ldots, x_j^{(n)}\right)^T$ with $j \in [\![1, d]\!]$ denotes the observed samples for the input X_j;
- $\mathbf{Z} = \left(z^{(1)}, \ldots, z^{(n)}\right)^T$ is the vector of outputs corresponding to the DoE $\mathbf{X}$ with $z^{(i)} = g_Z(\boldsymbol{x}^{(i)})$.

In the context of code failures, the DoE is divided into two sub-designs. The first sub-design contains the simulations for which the code fails while the second sub-design contains the successful simulations. Thus, we define:

- $A = \{l_1, \ldots, l_{|A|}\}$: subset of the indices taken from $\{1, \ldots, n\}$ such that the corresponding simulations fail ($Z = 1$);
- $\bar{A} = \{1, \ldots, n\} \setminus A$ the subset of the indices taken from $\{1, \ldots, n\}$ such that the corresponding simulations do not fail ($Z = 0$);
- $\mathbf{X}_A = \left(\boldsymbol{x}^{(l_1)}, \ldots, \boldsymbol{x}^{(l_{|A|})}\right)^T$ the matrix containing the elements of $\boldsymbol{X}$ such that the code fails, $\mathbf{X}_A \in M_{|A|,d}(\mathbb{R})$;

- $\boldsymbol{x_{A,j}} = \left(x_j^{(l_1)}, \ldots, x_j^{(l_{|A|})}\right)$ the values of the j^{th} variable for the failing part of the sample;
- $\mathbf{X}_{\bar{A}}$ the matrix containing the elements of $\mathbf{X}$ such that the code does not fail.

As a reminder, the number of input parameters of the MC3D code considered here is $d = 57$. We also consider that all the d inputs are independent and vary uniformly in $[0, 1]$: $\mathbf{X} \sim U_{[0,1]^d}$. A DoE of $n = 2000$ simulations of the code has been performed. Among them, $l_{|A|} = 744$ failed. This represents 37% of the code runs. Considering the computational cost of each code simulation, this significant number of code failures justifies in practice this in-depth study.

The chapter is divided in five sections. The two first sections are dedicated to the presentation of the tools used for the screening by sensitivity analysis regarding code failures occurrences. Section 3.3 aims at presenting the graphical and the physical analysis of the results. In Section 3.4, we present how we used these results in order to detect groups of interdependence regarding code failures. Finally, Section 3.5 aims at presenting conclusion and perspectives regarding the presented work.

3.1 Assessment of the inputs impact on code failures based on comparisons of conditional distributions

A first way to measure the influence of the inputs X_j on code failures can be the study of the realizations $\boldsymbol{x_{A,j}}$ such that the code fails. Indeed, if we suppose that an input X_j has no effect on code failures, then X_j and Z are independent. Thus the random variables X_j and $X_j|Z$ (X_j conditioned by Z) should be identically distributed. This implies that the realizations such as the code fails $\boldsymbol{x_{A,j}}$ follow the same distribution as the realizations $\boldsymbol{x_j}$.

The first proposed approach to detect the inputs influencing the code failure is thus to compare the distributions of each $\boldsymbol{x_{A,j}}$ to the theoretical distribution of $\boldsymbol{x_j}$ using a statistical goodness-of-fit test.

Several methods can be used to perform this comparison. We can cite the Wasserstein metric (Villani, 2003) or the Cramér-von Mises criterion (Cramér, 1928) for instance. Here, the classical Kolmogorov-Smirnov (KS) (Shiryayev, 1992; Smirnov, 1948) goodness-of-fit test is used. However, our method can be directly extended to the two other metrics presented above.

The Kolmogorov-Smirnov statistic is well suited to compare a sample distribution to a reference continuous one. It is sensitive to differences in both location and shape of the cumulative distributions functions. The conservative aspect of this test method is not an issue here. Indeed, the tests will only discriminate the input parameters whose distribution is really different from the uniform distribution and thus the most likely to have a significant effect on code failures. For a first analysis and given the large number of variables involved in the application case, potentially missing a few influential variables may be less of a problem than wrongly selecting one.

The next subsection gives a review on the Kolmogorov-Smirnov (KS) statistic and the associated test method. Then, the methodology is adapted to sample generated by the Latin Hypercube Sampling (LHS) method. Finally, the results obtained with this approach for our application case are briefly presented.

3.1.1 Comparison of the marginal of each input to the uniform distribution using Kolmogorov-Smirnov test

The Kolmogorov–Smirnov statistic compares the cumulative distribution function of a known reference distribution with the empirical distribution of sample [1] using a distance measure.

Thus, let F be the distribution function associated to a continuous random variable X. We define $\boldsymbol{x} = \left(x^{(1)}, \ldots, x^{(n)}\right)$ as an independent and identically distributed (i.i.d) sampling of X. For any set A, we define $\mathbf{1}_A$ as the indicator function of A. Thus, the empirical distribution of X is $F_n(t) = \sum_{i=1}^{n} \mathbf{1}_{x^{(i)} \leq t (t \in \mathbb{R})}$.

To measure the difference between the empirical distribution F_n and F, we can use the L^∞ distance (already introduced in Chapter 2):

$$D_n(F_n, F) = \|F_n(t) - F(t)\|_\infty = \sup_t |F_n(t) - F(t)|. \tag{3.2}$$

The Glivenko-Cantelli theorem ensures that we have $D_n(F_n, F) \xrightarrow[n\to\infty]{} 0$ almost surely. In practice, D_n can be easily computed using the following formula:

$$D_n(F_n, F) = \max_{1 \leq i \leq n} [\max(|i/n - F(x^{(i)})|, |F(x^{(i)}) - (i-1)/n|)] \tag{3.3}$$

Here, $\left(x^{(1)}, \ldots, x^{(n)}\right)$ corresponds to the ordered realizations of $\boldsymbol{x}$.

[1] Note that the KS statistic can also be used to compare the empirical distribution of two samples. This will not be discussed here.

The Kolmogorov-Smirnov (KS) statistic is defined using this distance. Indeed, it corresponds to the rescaled distance $S_n^{KS} = n^{1/2} D_n(F_n, F)$.

There is two important properties on the asymptotic convergence of S_n^{KS} (Shiryayev (1992)):

1. S_n^{KS} converges in distribution to the random variable K when $n \to \infty$ with K having a cumulative distribution function defined by $F_K(t) = \mathbb{P}(K \leq t) = 1 + 2\sum_{k=1}^{+\infty}(-1)^k e^{-2k^2t^2}$. Note that $F_K(t)$ does not depend on F.
2. If F_0 is a continuous distribution function such as $F \neq F_0$, the quantity $S_n^{KS}(F_n, F_0) \xrightarrow[n\to\infty]{} +\infty$ in probability.

Thanks to these two properties, one can associate a test procedure to the Kolmogorov-Smirnov statistic $S_n^{KS}(F_n, F_0)$. This test answers the question: does the cumulative distribution of the tested samples F is the same as a reference cumulative distribution F_0. The test hypotheses are:

- the null hypothesis 'H_0: $F = F_0$',
- the alternative hypothesis 'H_1: $F \neq F_0$'.

To process the test, the observed value $s_{n,obs}^{KS}(F_n, F_0) = n^{1/2} \sup_t |F_n(t) - F_0(t)|$ is computed. This observed value is then compared to the theoretical distribution K associated to the Kolmogorov statistic under H_0.

Under H_0, we have that S_n^{KS} converges in law to K. Further, under H_1, $S_n^{KS}(F_n, F_0) \xrightarrow[n\to\infty]{} +\infty$ in probability. So that, the critical region $\{K > s_{n,obs}^{KS}\}$ can be associated to the test when n is large enough.Thus, the p-value associated to the test corresponds to the quantity $\mathbb{P}(K > s_{n,obs}^{KS} | H_0)$.

The p-value drives the decision process. The lower the p-value, the stronger the null hypothesis is rejected. A significance level $\alpha \in [0, 1]$ is classically associated to the test procedure. This significance level corresponds to the probability of rejecting wrongly the null hypothesis H_0. For a given sample, if the observed p-value is lower than the chosen significance threshold α, then the null hypothesis is rejected at the chosen level of significance α. It is important to note that this significance level is chosen arbitrarily. In practice, it is usually set depending on the application and the consequences of wrongly rejecting H_0.

In our case, we wish to compare each of the d samples $(\boldsymbol{x_{A,j}})_{j\in\{1,..d\}}$ to the uniform distribution $U_{[0,1]}$. Thus, the idea is to apply the Kolmogorov-Smirnov test to assess the marginal influence of each input. For each input sample $(\boldsymbol{x_{A,j}})_{j\in\{1,..d\}}$, the statistic $s_{n,obs}^{KS}(F_{j,n}, F_{U_{[0,1]}})$ is computed. Here, $F_{j,n}(t)$ is the empirical distribution function of $\boldsymbol{x_{A,j}}$. The p-values $\{p_1^{KS}...p_d^{KS}\}$ associated with these tests are then

calculated. The obtained p-values allow us to select the inputs regarding the likelihood of the null hypothesis. The p-values below the chosen threshold α correspond to the inputs for which the hypothesis of a uniform distribution of $\boldsymbol{x}_{\boldsymbol{A},\boldsymbol{j}}$ is rejected. These inputs are therefore considered to have a significant impact on the code failures.

3.1.2 Impact of the use of a Latin Hypercube Sampling

As discussed in Chapter 2, the LHS method has been used to sample the inputs space. We saw in Chapter 2 what shape the Kolmorogov statistic takes if the sample under study was generated using the LHS design. Unfortunately, we are not exactly in this case here. Indeed, if the initial $\boldsymbol{x}_j$ samples were effectively generated by an LHS, the samples $\boldsymbol{x}_{\boldsymbol{A},\boldsymbol{j}}$ are not a LHS anymore because of the code failures, even under the null hypothesis H_0. To illustrate this, Figure 3.1 shows the empirical distribution of the distance $D_n(F_n, F)$, estimated using $L = 100000$ classic Simple Random Sampling (SRS) designs, LHS designs, and finally LHS designs from which we have randomly selected 1/3 and 2/3 of the realizations. The sampling size used for the illustration is relatively small ($n = 10$) to better visualize the differences between the different distributions. In particular, we can observe that the distributions of $D_n(F_n, F)$ for the truncated LHS are multimodal. The number of modes seems to depend on the proportion of samples selected. The theoretical expression of these particular statistics have not been studied here.

Hence, we do not have access to the theoretical distribution of the KS statistic associated with truncated LHS. To overcome this issue, the natural solution to process the test is to simulate samples under the null hypothesis H_0. By doing so we get the empirical cumulative distribution of the statistic under H_0. Then we use it to estimate the p-value associated to the observed statistic.

Let us describe more precisely the procedure. We generate L LHS of size n and dimension 1 corresponding to samples $\boldsymbol{x}_j, j \in [\![1, d]\!]$ and L vectors of n independent Bernoulli realizations $\mathcal{B}(l_{|A|}/n)$ representing samples of Z under the null hypothesis. We then estimate the L associated statistics $\{S_n^{KS|H_0,(1)}, ..., S_n^{KS|H_0,(L)}\}$ using Formula (3.3). From the resulting L-size sample of KS statistics, the empirical cumulative distribution $F_{n,S^{KS|H_0}}(t) = \frac{1}{n}\sum_{l=1}^{L} \mathbf{1}_{S_n^{KS|H_0,(l)} \leq t}$. Note that the higher is L, the better is the estimation of the cumulative distribution. In our case, we took $L = 10000$. We then estimate, for each sample $(\boldsymbol{x}_{\boldsymbol{A},\boldsymbol{j}})_{j\in\{1,..d\}}$, the statistic $s_{n,obs}^{KS}(F_{j,n}, F_{U_{[0,1]}})$. Here, $F_{j,n}(t)$ is the empirical distribution function of $\boldsymbol{x}_{\boldsymbol{A},\boldsymbol{j}}$ and $F_{U_{[0,1]}}$ is the uniform cumulative distribution function. Finally and for each of the d statistics, we estimate the p-value associated to the test p^{KS}. Algorithm 2 summarizes the whole procedure.

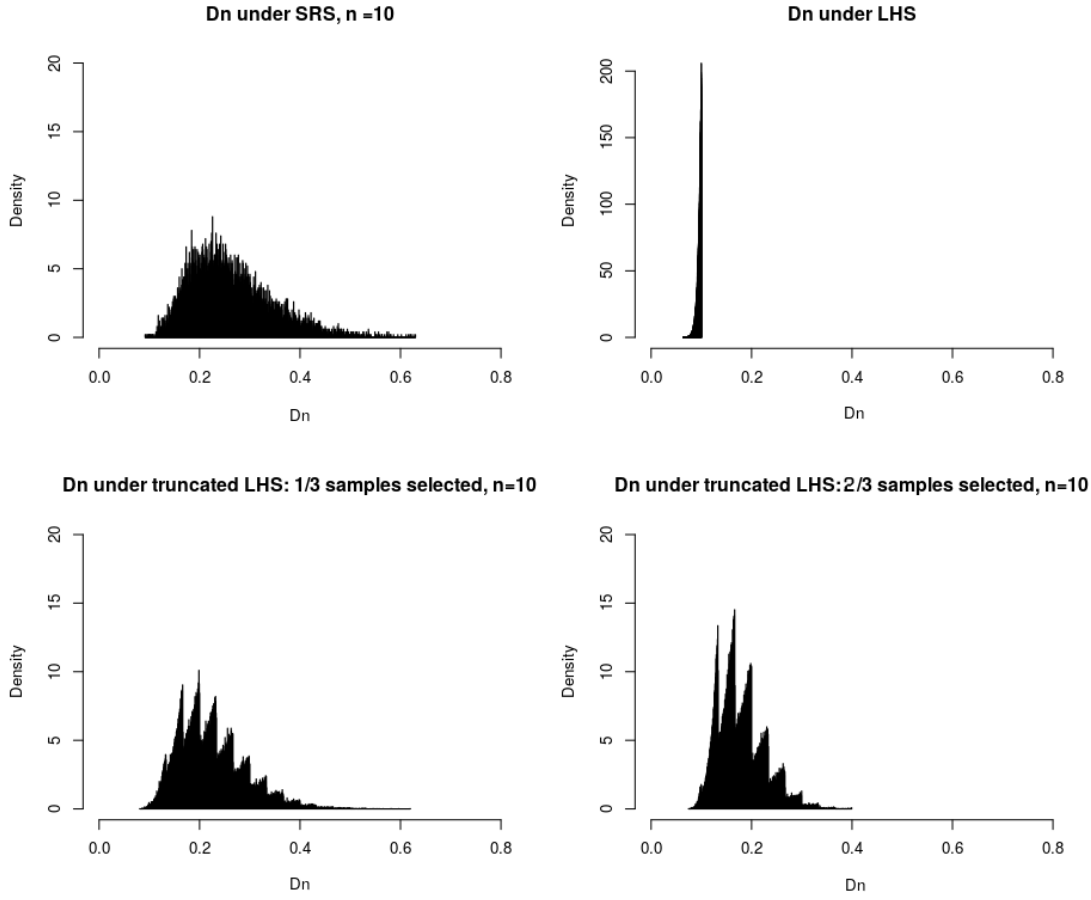

Figure 3.1: *Histograms of the distance D_n for different sampling methods: Simple Random Sampling (SRS), LHS, and two truncated LHS.*

Algorithm 2 Corrected K-S test for truncated LHS designs

Require: Sample size n, number of repetitions L and design $\boldsymbol{X}_A$

1: **for** $l = 1$ to L **do**
2: Generate an LHS sample of size n and dimension 1 to simulate a sample of type $\boldsymbol{x}_j$.
3: Generate an $i.i.d$ sample of size n following Bernoulli distribution of parameter $(l_{|A|}/n)$ that corresponds to a sample of the output Z under the null hypothesis
4: Deduce a replica under the null hypothesis of $x_A|H_0$ form this drawn
5: Compute the observed Kolmogorov statistic $S_n^{KS|H_0} = \sqrt{n}D_n(F_{x_A|H_0}, F_{U_{[0,1]}})$. Here, $F_{x_A|H_0}$ is the empirical cumulative distribution of $x_A|H_0$
6: **end for**
7: Estimate the empirical distribution $F_{S_n^{KS|H_0}}(t) = \sum_{l=1}^{L} \mathbf{1}_{S_n^{KS|H_0,(l)} \leq t}$ of the KS statistic under H_0 from the L observed Kolmogorov statistics.
8: Compute the d Kolmogorov statistic on $\boldsymbol{x}_{\boldsymbol{A},\boldsymbol{j}}$ and the corresponding statistics $\{s_{n,obs}^{KS,(1)}, ...s_{n,obs}^{KS,(d)}\}$
9: Get the d p-values $\{p_1^{KS}...p_d^{KS}\}$, using the empirical distribution of the KS statistic under the null hypothesis $F_{s_{n,obs}^{KS}}$

3.1.3 Results for the case study

Figure 3.2 presents the result of KS-test process procedure given by Algorithm 2 and applied to the $n = 2000$-size sample of the MC3D code. The obtained p-values are given by increasing order. The number of the corresponding input is indicated right next the colored circle.

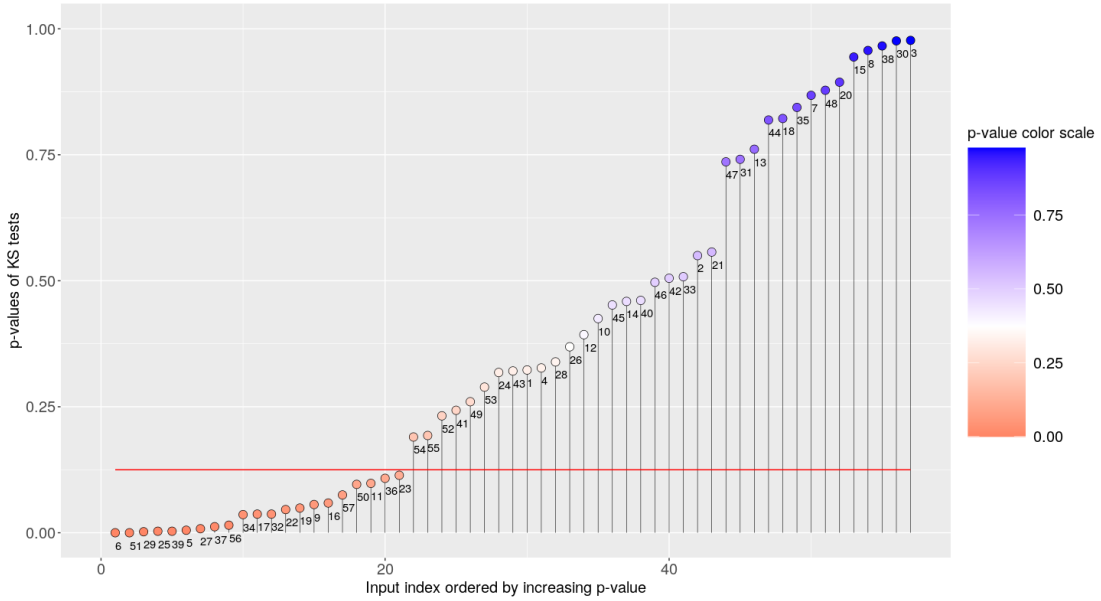

Figure 3.2: *Ordered p-values of the KS-tests performed for each of the $d = 57$ inputs.*

In our study we choose the threshold $\alpha = 0.125$ (red line on Figure 3.2). This choice is supported by the jump in p-values observed around this threshold. The p-value of X_{36} ($p_{23}^{KS} = 0.114$) is indeed just followed by the one of X_{18} such as $p_{54}^{KS} = 0.19$. Two groups of inputs are clearly identified: a first one made up of 21 variables for which the uniform distribution hypothesis is rejected ($p < \alpha$) and a second one with the 36 remaining inputs for which the test cannot reject the hypothesis of a uniform distribution.

In conclusion, a first method based on KS test and adapted for truncated LHS has been introduced to study the implication of each input in code failures. This method allowed us to identify which variables are most likely to have an impact on code failure.

3.2 Sensitivity analysis of code failures based on dependence measures

In this section, we present another way to study the global influence of each input on code failures. This metric is based on a dependence measure. The idea is to study the probabilistic dependence between the random variable $(\mathrm{X}_j)_{j\in[\![1,d]\!]}$ and the binary random variable Z characterizing the code failures. The key point is to select a dependence measure that detects the variables whether they are influential on Z alone or in interaction with other variables.

A probabilistic dependence corresponds to any relationship between two random variables. More formally, two random variables are designated as dependent if the realization of one affects the distribution of the other. To measure the dependence between two variables, several tools have been proposed. The simplest of them is the correlation coefficient (Pearson, 1895). It measures the normalized covariance between two variables. Formally, for two random variables X and Y, the correlation $\rho_{X,Y}$ is defined by:

$$\rho_{X,Y} = \frac{\mathbb{C}\mathrm{ov}(X,Y)}{\sigma_X \sigma_Y} = \frac{\mathbb{E}\Big((X-\mathbb{E}(X))(Y-\mathbb{E}(Y))\Big)}{\sqrt{\mathbb{V}\mathrm{ar}(X)\mathbb{V}\mathrm{ar}(Y)}} \tag{3.4}$$

Unfortunately, there is no equivalence between correlation and independence. In fact, if two variables X and Y are correlated ($\rho_{X,Y} \neq 0$) , they are dependent, but the reciprocal is false. Thus, the use of this tool is limited. To address this issue, several measures of global dependence have been proposed. We can cite for instance those based on the f-divergence (Csiszár, 1972) or the ones based on the notion of entropy (Shannon, 1948). Among these dependence measures, the ones based on Reproducing Kernel Hilbert Space (RKHS) distinguish themselves (Aronszajn, 1950). Indeed, they can be used for any type of random variable (continuous, categorical, graphs, etc.). These methods can also be easily adapted to the multidimensional case. In addition, the size of the sample required to accurately estimate this dependence measure is relatively low.

One of the most interesting kernel based dependence measure is the Hilbert Schmidt Independence Criterion (HSIC). The HSIC yields efficient results even on small sample sizes and seems to numerically outperform other RKHS measures (Gretton et al., 2005). This explains why it has been widely used in the context of sensitivity analysis recently (Da Veiga, 2015 ; De Lozzo and Marrel, 2017). HSIC are also inherently good at measuring dependencies between continuous variables and a binary variable, such as code failures (Marrel and Chabridon, 2021).

The next subsection gives a presentation of this criterion and the associated test. We will also discuss how this criterion might be used to measure the impact of the input parameters on the code failures.

3.2.1 HSIC-based independence test between each input and the code failures

The Hilbert-Schmidt Independence Criterion (HSIC) aims to detects the dependence between two random variables. This criterion is based on the cross-covariance operator that generalizes the notion of the classical covariance (Gretton et al., 2005). In fact, under certain conditions, the nullity of the HSIC is equivalent to the independence. It thus corrects the main default of the Pearson coefficient presented above.

Before defining the HSIC, we need to introduce the concepts of Reproducing Kernel Hilbert Spaces (RKHS), characteristic kernels and generalized cross-covariance operator.

3.2.1.1 Reproducing Kernel Hilbert Spaces (RKHS) and characteristic kernels.

Let $\mathcal{F}$ be an Hilbert space (a complete vector space equipped with an inner product $\langle , \rangle_{\mathcal{F}}$) of real-valued functions on a set $\mathcal{X}$. $\mathcal{F}$ is a Reproducing Kernel Hilbert Space (RKHS), if there is a unique symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that:

1. $\forall x \in \mathcal{X}, k(x, .) \in \mathcal{F}$;
2. $\forall f \in \mathcal{F}, \forall x \in \mathcal{X}, \langle f, k(x, .)\rangle = f(x)$ (reproducing property).

This function k is called the reproducing kernel of $\mathcal{F}$. Moreover, a kernel k is said *characteristic* for $\mathcal{F}$ if, for all probability measures $\mathbb{P}$ defined on $\mathcal{F}$, the function $\mathbb{P} \to \int k(., x)d\mathbb{P}(x)$ is injective. See (Fukumizu et al., 2008) or (Sriperumbudur et al., 2010) for details.

3.2.1.2 Cross-covariance operator

Now we can define the cross-covariance operator. Let $X \in \mathcal{X}$ (resp $Y \in \mathcal{Y}$) be a random variable and $\mathcal{F}$ (resp $\mathcal{G}$) be the RKHS associated to $\mathcal{X}$ (resp $\mathcal{Y}$). Let $k(., .)$ (resp. $l(., .)$) be the characteristic kernel of $\mathcal{F}$ (resp. $\mathcal{G}$). We also define $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{G}}$ as their inner products.

The generalized cross-covariance operator is defined as the operator mapping from $\mathcal{F}$ to $\mathcal{G}$ and verifying for all $(f, g) \in \mathcal{F} \times \mathcal{G}$:

$$\langle f, C_{X,Y}(g)\rangle_{\mathcal{F}} = \mathbb{C}\text{ov}(f(X), g(Y)). \tag{3.5}$$

This operator generalizes the notion of the covariance between X and Y.

3.2.1.3 HSIC

To summarize the information provided by $C_{X,Y}$, we consider its squared Hilbert-Schmidt norm, called HSIC:

$$HSIC(X,Y)_{\mathcal{F},\mathcal{G}} = \|C_{X,Y}\|^2 = \sum_{p,q}\langle u_p, C_{X,Y}(v_q)\rangle^2_{\mathcal{F}} = \sum_{p,q}\mathbb{C}\text{ov}(u_p(X), v_q(Y))^2, \tag{3.6}$$

where $(u_p), p \geq 0$ and $(v_q), q \geq 0$ are some orthonormal bases of $\mathcal{F}$ and $\mathcal{G}$ respectively. Using RKHS properties, Gretton et al. (2005) has shown that HSIC can also be expressed in a convenient form using kernels, as shown in Eq. (3.7).

$$HSIC(X,Y)_{\mathcal{F},\mathcal{G}} = \mathbb{E}(k(X,X')l(Y,Y')) + \mathbb{E}(k(X,X'))\mathbb{E}(l(Y,Y'))$$

$$-2\mathbb{E}\Big(\mathbb{E}(k(X,X')|X)\mathbb{E}(l(Y,Y')|Y)\ \Big). \tag{3.7}$$

Here, (X', Y') is an independent copy of (X, Y).

The use of *characteristic kernels* enable us to to have an equivalence between these two propositions:

1. $HSIC(X,Y) = 0$;
2. $\mathbb{C}\text{ov}(f(X), g(Y)) = 0\ \forall (f,g) \in \mathcal{F}\times\mathcal{G}$.

Besides, it has been shown that two variables X and Y are independent if and only if $\mathbb{C}\text{ov}(f(X), g(Y)) = 0$ for all continuous functions $(f,g) \in \mathcal{F}\times\mathcal{G}$ (see Jacod and Protter (2004) for instance).

Thus, for characteristic kernels k and l, we get the equivalence between the nullity of $HSIC(X,Y)_{\mathcal{F},\mathcal{G}}$ and the independence between X and Y. In this case, the HSIC generalizes the notion of covariance in the sense presented at the beginning of this section.

3.2.1.4 Two examples of characteristic kernels

For a real variable X, one of the most useful characteristic kernel is the Gaussian kernel. It is defined, for $(x, x') \in \mathbb{R} \times \mathbb{R}$, by: $k(x, x') = \exp(-\frac{\lambda}{2}(x - x')^2)$ with a bandwidth parameter $\lambda > 0$. This parameter is often set at $\lambda = 1/\sigma^2$ with σ^2 being the empirical variance.

For categorical variables, the Dirac kernel can be appropriate. Let Y be a categorical variable with $K + 1$ modalities $\{0, \ldots, K\}$. The Dirac kernel is defined, for $(y, y') \in \{0, ...K\}^2$, by $l(y, y') = \mathbf{1}_{y=y'}$. Here,$\mathbf{1}$ is the indicator function.

These two characteristic kernels are used in our application. Note that it is possible to use a kernel similar to k for categorical variables by defining a distance between the different modalities. We also notice that the kernel l is related to the kernel k with $\lambda \to +\infty$. For more information about kernels and categorical inputs, the interested reader can refer to (Roustant et al., 2020).

3.2.1.5 Estimation of the HSIC

In practice, the computation of HSIC is processed using Eq. (3.7). For instance, Gretton et al. (2005) proposed to estimate the quantities written in this equation. Let $\{(x^{(1)}, y^{(1)}), ..., (x^{(n)}, y^{(n)})\} \subseteq \mathcal{X} \times \mathcal{Y}$ be a series of n independent realizations of (X, Y).

According to (Gretton et al., 2005), an HSIC estimator is given by the following formula:

$$\widehat{HSIC}(X, Y) = \frac{1}{n^2}\text{tr}(\boldsymbol{KHLH}) \tag{3.8}$$

with:

- $\boldsymbol{H}, \boldsymbol{K}, \boldsymbol{L}$ three matrices in $\mathbb{R}^{n\times n}$, such as: $K_{i,i'} = k(x^{(i)}, x^{(i')})$, $L_{i,i'} = l(y^{(i)}, y^{(i')})$ and $H_{i,i'} = \delta(i, i') - \frac{1}{n}$, $\delta(i, i')$ being the dirac distribution. Here, $(i, i') \in [\![1, n]\!] \times [\![1, n]\!]$.
- tr the trace operator, for $A \in \mathbb{R}^{n\times n}$, $\text{tr}(A) = \sum_{i=1}^{n} A_{i,i}$.

Gretton et al. (2005) showed that this estimator is asymptotically unbiased.

3.2.1.6 HSIC-based statistical independence test

A statistical independence test can be built on the fundamental property of the HSIC to test the independence between two variables (Gretton et al., 2008). For a given variable X and another variable Y, it aims at testing the null hypothesis H_0: 'The input X and the output Y are independent' against the alternative hypothesis H_1: 'There is a dependency structure between X and Y'. Since there

is an equivalence between the independence of variables and the nullity of HSIC (with characteristic kernels), the two hypotheses can be reformulated as follows:

- H_0: '$HSIC(X,Y)_{\mathcal{F},\mathcal{G}} = 0$';
- H_1: '$HSIC(X,Y)_{\mathcal{F},\mathcal{G}} > 0$'.

The quantity $S_n^{HSIC}(X,Y) = n \times \widehat{HSIC}(X,Y)$ is a natural statistic for this independence test. If X and Y are independent and under asymptotic convergence (i.e if n is large enough), it has been proven that the law of S_n^{HSIC} can be asymptotically approached by a *Gamma distribution* with shape and scale parameters (γ, β) (Gretton et al., 2008). These parameters can be estimated by:

$$\hat{\gamma} = \frac{\mathbb{E}\Big(\widehat{HSIC}(X,Y)\Big)^2}{\mathbb{V}\mathrm{ar}\Big(\widehat{HSIC}(X,Y)\Big)}, \ \hat{\beta} = \frac{n.\mathbb{V}\mathrm{ar}\Big(\widehat{HSIC}(X,Y)\Big)}{\mathbb{E}\Big(\widehat{HSIC}(X,Y)\Big)}$$

The interested reader can refer to (Gretton et al., 2008) for more details.

As for the Kolmogorov-Smirnov test, the decision process is driven by the p-value. Here, the p-value corresponds to the probability, under H_0, that the statistic S_n^{HSIC} becomes greater or equal to the value observed on the studied data (here $s_{n,obs}^{HSIC}$). Formally, the p-value is defined by:

$$p^{HSIC} = \mathbb{P}(S_n^{HSIC} > s_{n,obs}^{HSIC} | H_0)$$

As for the previous method, the idea in our case is to process the HSIC test for each inputs $(X_j)_{j \in [\![1,d]\!]}$. We measure the dependence between each input and Z through the comparison of the observed quantity $s_{n,obs}^{HSIC}$ and the approximated Gamma probability distribution. To do this, a kernel must be chosen for each input X_j and the binary output Z. As usual, a Gaussian kernel is associated to X_j since this is a continuous variable. For Z, a Dirac kernel is used. After computing the statistic $s_{n,obs}^{HSIC}$, the associated p-value p^{HSIC} is estimated. The inputs with the lowest p-values regarding this procedure are the ones for which the assumption of independence with Z is the least compatible with the observed data. These inputs are therefore the most likely to explain the failures of the code.

3.2.2 Results for the case study

We have applied the HSIC-based independence test method to our case study. Recall that in our application, the design of experiment for $\boldsymbol{X}$ has been performed using the Latin hypercube sampling method. As discussed in the previous chapter, the convergence of the HSIC statistic under H_0 is well ensured using a LHS method (El Amri and Marrel, 2021). The classic procedure can then be reasonably applied.

Figure 3.3 shows the obtained results. For reasons similar to those for the KS test, a threshold $\alpha = 0.125$ is chosen, to screen the significantly lowest p-values for which the independence hypothesis is rejected. Thus, 18 inputs are selected. These are the inputs most likely to have a direct impact on code failures regarding the HSIC tests.

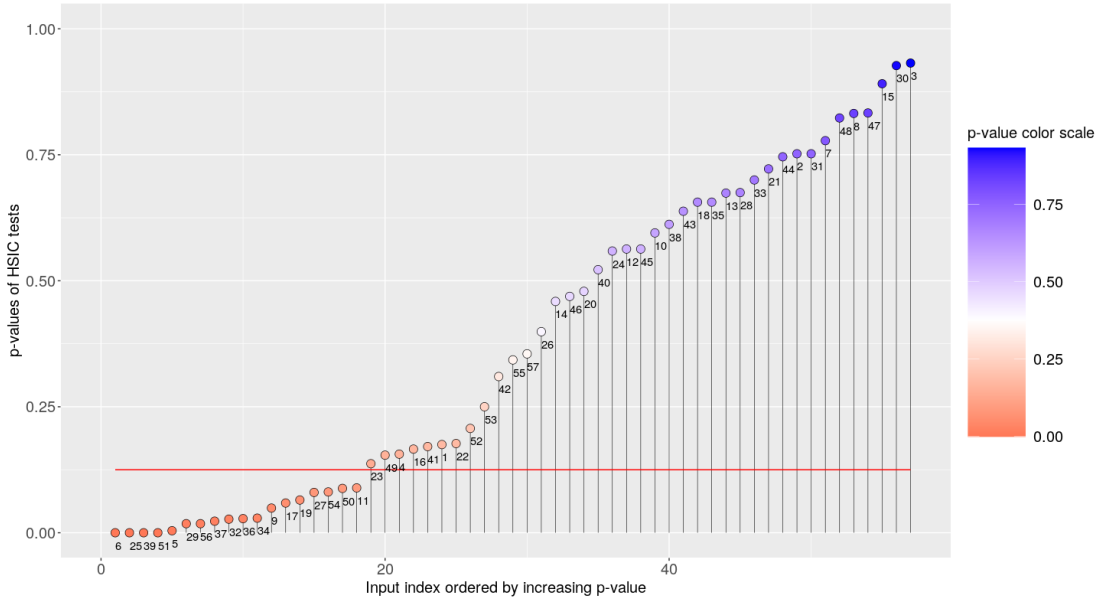

Figure 3.3: *Ordered p-values of the HSIC tests performed for each of the $d = 57$ inputs, from the $n = 200$-size sample of MC3D code.*

3.3 Graphical and physical analysis of the results

Table 3.1 summarizes the results obtained with the KS and the HSIC methods of Section 3.1 and 3.2, respectively. It presents the set of variables selected by at least one of the two test methods. We find that the 18 variables selected by the HSIC test method are also selected by the KS test method. Confidence in this variable selection is thus increased by the consistency between the results of the two test procedures.

Table 3.1: *Summary of the results obtained with the KS and the HSIC tests.*

Input number	Pvalues KS	Pvalues HSIC
6	0.000	0.000
51	0.000	0.000
29	0.002	0.018
25	0.003	0.000
39	0.003	0.000
5	0.005	0.004
27	0.008	0.08
37	0.012	0.023
56	0.015	0.018
34	0.036	0.029
17	0.037	0.059
32	0.037	0.027
22	0.046	0.177
19	0.049	0.065
9	0.056	0.049
16	0.059	0.166
57	0.075	0.355
50	0.096	0.088
11	0.098	0.089
36	0.108	0.028
23	0.114	0.137
54	0.19	0.081

To go further in the analysis, the estimated density of $\boldsymbol{x}_{A,j}$ (marginal sample such as the code fails) is compared to the density of $\boldsymbol{x}_j$ for the variables with the lowest p-values regarding both tests. The results are given in figure 3.4. Both densities are estimated with a kernel-based non parametric approach. For more information

about this method of estimation, one can refer to M. Rosenblatt (1956) or Parzen (1962).

Figure 3.4 provides information on how the occurrence of failure impacts the density of each input. Therefore, thanks to Bayes formula, it directly informs on the probability distribution of code failure occurrence depending on the values of each input. Without being exhaustive, we give bellow a physical interpretation of these graphics.

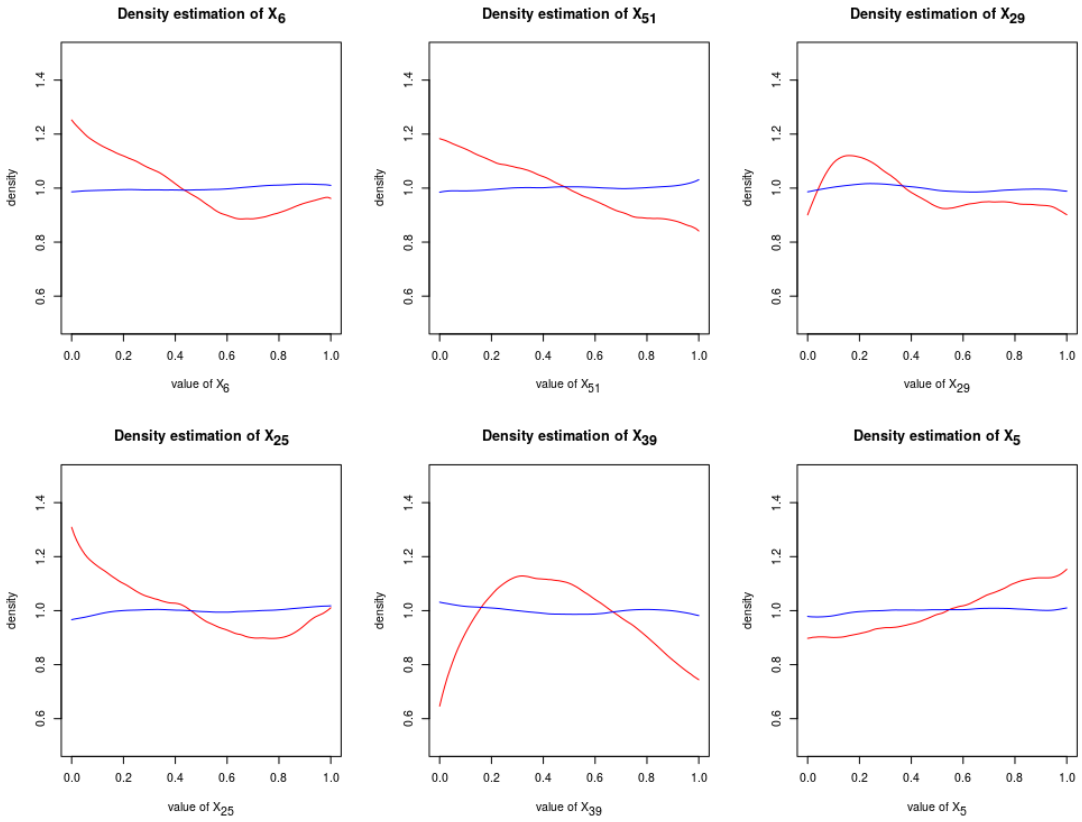

Figure 3.4: *Plots of empirical densities of* $\boldsymbol{x}_{A,j}$ *(in red) and* $\boldsymbol{x}_j$ *(in blue) for* 6 *inputs having the lowest p-values regarding the two test methods.*

Among these 6 inputs, let us first focus on the two implicated in the jet fragmentation during the premixing step (modeled by the model function f_1): the input X_5 (KELMHOLTZ_C_VELFRA) and the input X_6 (KELMHOLTZ_RADIUS). As a reminder, the model used for the jet fragmentation in our case is based on a Kelvin-Helmoltz instabilities model. In this model, the drop ejection velocity V_e is assumed to be proportional to the fragmentation rate Γ_f:

$$V_e \approx C_{vit}\Gamma_f.$$

Here, C_{vit} corresponds to the modeling parameter KELMHOLTZ_C_VELFRA (X_5). The fragmentation Γ_f is a function of the densities and mean velocities of the jet and the surrounding water (liquid and vapor). To prevent an impact of the calculation grid size on the jet fragmentation, the liquid water, the steam water densities and velocities are averaged on a parametric radius KELMHOLTZ_RADIUS (X_6).

We can see on the corresponding plot of Figure 3.4 that the sample distribution of X_5 such as the code fails to converge is a growing function of X_5. This is consistent with the modeled physics. Indeed, the higher KELMHOLTZ_C_VELFRA, the higher the drop velocity and heat exchange between the drops and surrounding water. As the premixing model of MC3D is not developed to handle violent physics, which is more the role of the explosion model, this may be a reason why the number of failures increases.

The sampling distribution of X_6 such as the code fail is a decreasing function of X_6. Again, this is consistent with the physics modeled. A high value of KELMHOLTZ_RADIUS corresponds to an averaging of the velocities over a larger space. This makes the fragmentation rate less sensitive to local variations and thus makes the code output more stable. The slight increase in the density function for the highest values of $X_6|Z = 1$ is more difficult to explain.

Finally, we can examine the input X_{51}. Among the 6 inputs presented, this is the only input of the explosion application $f_{[2]}$. It corresponds to the artificial viscosity VISCART. This pseudo-viscosity has been added in order to limit numerical instabilities inherent to the modeling explosion shock wave. Naturally, the higher this pseudo-viscosity, the more stable the code. This is what we observe on the second plot of Figure 3.4.

3.4 Measure of interdependence between the inputs conditioned by failure occurrence

As seen previously, the inputs are assumed to be independent and initially sampled according to this assumption. However, the analysis that will be carried out after will be performed only on the simulations that have converged. Therefore, the considered samples of inputs will be conditioned by the absence of code failures. This may induce dependences between input variables. It is therefore interesting to measure the eventual interdependences between inputs conditioned by failure occurrence.

For this, we propose to analyze the pairs of inputs such as the code fails to converge. Interdependent variables could be then grouped together for the remaining steps of our methodology. The screening step performed in the previous sections can be useful in order to reduce the number of interactions to be studied. Indeed, if we consider the $d = 57$ inputs, we would have to study a total of $\binom{d}{2} = 1596$ pairs of inputs, compared to only $\binom{d'}{2} = 231$ if we limit ourselves to the $d' = 22$ inputs which have been identified as significantly influential on failures. To study the dependence between a couple of inputs $(X_j, X'_j), j, j' \in [\![1, d']\!]$ and the occurrence of failures Z, we proposed again HSIC based tests.

Let $X_j|Z$ and $X_{j'}|Z$ be two input variables conditioned by failure occurrence in the pool of the d' selected inputs. To assess if an interdependence has been induced by the conditioning, we consider the HSIC statistic applied on $X_j|Z = 1$ and $X_{j'}|Z = 1$. More precisely, we estimate the quantity $s^{HSIC}_{n,obs}(\boldsymbol{x}_{\boldsymbol{A},\boldsymbol{j}}, \boldsymbol{x}_{\boldsymbol{A},\boldsymbol{j'}})$. For each of the $\binom{d'}{2} = 231$ pairs of inputs and associated statistic $s^{HSIC}_{n,obs}$, a p-value is estimated using the approximated Gamma probability distribution. As before, we associate a significance level to these tests. Considering the high number of tests performed here, the level is set to a quite low value, $\alpha = 0.01$. It allows us to only focus on the main interdependencies regarding the test procedure. Selecting too many interdependencies (with a higher level α) would make the results difficult to analyze.

Figure 3.5 shows the results obtained in the form of a graph. Each node j represents the variable $X_j|Z$. The pairs of variables for which the independence is rejected are linked by a line. This representation enables to easily locate the groups of interdependent variables.

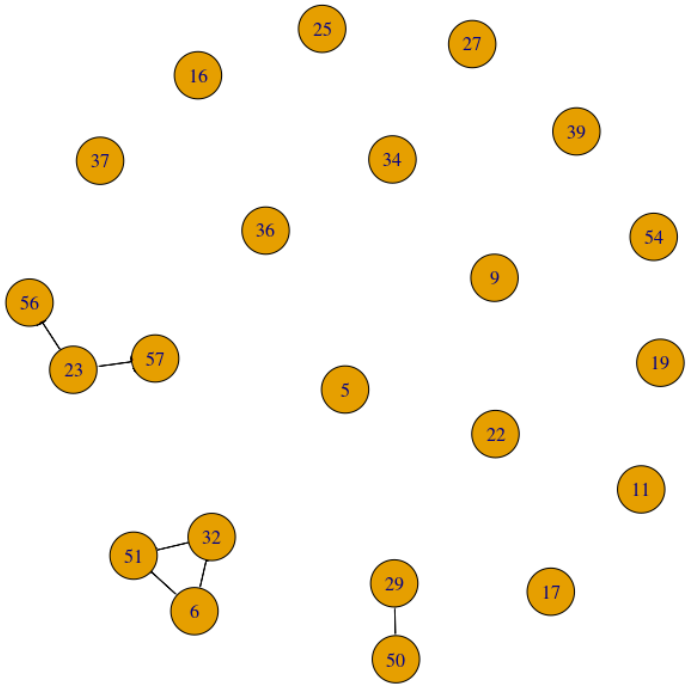

Figure 3.5: *Graph representing the interdependence between pairs of inputs regarding the results of the HSIC tests for a level* $\alpha = 0.01$.

We observe on the graph three groups of interdependent inputs. In the first group, dependences have been detected between X_{23}, X_{56} and X_{57}. They corresponds respectively to NU_BUL, DMAXEXP and HFRAGFB. The second group of interdependent inputs is composed of the inputs X_6,X_{51} (KELMHOLTZ_RADIUS and VISCART) and the input X_{32} (MULTEH). Finally, a last group of interdependent inputs composed of the pair X_{29} and X_{50} (WECRBUL and VISCART) can be observed. From a physical point of view, these results are difficult to analyze. This would require in-depth investigations outside the scope of this thesis.

For the screening step of our global methodology, we will consider these variables as a component of their groups of dependences. It means that if one of them is considered as "significant" regarding a studied output, the whole sub-group will be.

3.5 Conclusion of the chapter and prospects

In the context of the exploration of the MC3D input space, a sample of $n = 2000$ code runs have been performed. Among them, more than a third failed to converge. We aimed to understand the origin of these code failures. More generally, the existence of code failures is a common problem in the context of the numerical simulation of complex physical phenomena. To improve the robustness of these simulation codes, it is useful to understand which inputs are involved in these code failures and how they are involved. Considering code failures as a binary output in its own right, the principles of global sensitivity analysis can be used to identify the variables that have the greatest effect on the failures. Since the output under study is a categorical variable, most classic tools of sensitivity analysis such as Sobol' indices or Morris method are not adapted. Moreover, in the case of a computationally expensive simulation code like MC3D, the number of possible simulations is limited. This also prevents the direct use of common methods of sensitivity analysis. In addition, the number of studied inputs is large in our application case ($d = 57$). This makes the use of methods whose cost depends on the dimension even more difficult to apply.

Within this framework, we first introduced two methods to detect which input variables might be involved in code failures. The first method consisted in comparing the initial distribution of each input against its distribution such as the code fails. It has been done using a goodness-of-fit test based on the Kolmogorov-Smirnov (KS) statistic. This test method allowed to detect the variables for which the distribution such as code fails is different from the initial one. Since our original test design is an LHS, the test procedure has been adjusted accordingly. A second method, based on the Hilbert Schmidt Independence Criterion (HSIC), has been then proposed. This criterion measures the dependence between two variables. This method allows the detection of the probabilistic dependence between each input variable and occurrence of a code failure. The latter is therefore represented as a categorical output variable. The parametrization of HSIC (i.e kernels) was adapted in consequence. These methods have the advantage of being usable for general design of experiments and perform well even in small sample sizes. The two methods gave consistent results and allowed the selection of 22 variables.

After that, we exploited the results of the KS test and HSIC based test in two different ways. First, a graphical analysis of the conditional densities of the inputs selected as influential on code failures has been done. This allowed a physical analysis of the role of some of the significant variables regarding the failures. Then, we used the previously obtained results to facilitate the detection of possible dependences between inputs conditioned by code failures. To measure these dependences, we reused the Hilbert Schmidt Independence Criterion (HSIC).

This criterion is fully adapted to this task. We were thus able to detect groups of interdependent inputs if we only keep the converged simulations.

Some extensions of this work can be considered. A first interesting perspective could be to distinguish the different causes of code failures in the analysis. The use of classification algorithms to learn the occurrence of failures as a function of the selected inputs could also provide valuable additional information.

The work presented in this section is part of the general methodology of simulation code analysis developed in this thesis. Thus, in addition to the contribution in terms of understanding the code, the tools developed and used in this section will be useful in the continuation of our work. Indeed, as we have seen, the dependences between inputs due to code failures will be partially taken into account in the rest of the methodology.

Chapter 4

Sensitivity analysis in high dimension

In the context of uncertainty quantification, sensitivity analysis aims to determine how variations in the inputs of a computational code contribute to the variations of its outputs. Sensitivity analysis is therefore useful for validating, simplifying, or better understanding models. It can also guide efforts to characterize the input parameters in order to reduce the uncertainties the most efficiently. Many reviews of sensitivity analysis methods have been conducted (Da Veiga et al., 2021; Iooss and Lemaître, 2015; Saltelli et al., 2004). Two types of methods are often distinguished: Local Sensitivity Analysis (LSA) and Global Sensitivity Analysis (GSA).

Local Sensitivity Analysis studies the variability of the output for small variations of the inputs around their nominal values. Among the approaches for LSA, the main methods are based on partial derivatives (Pujol, 2009). The idea behind these methods is to estimate the partial derivatives of the numerical model relative to each input at the nominal point. These partial derivatives represent the effect of the perturbation of each input on the total perturbation of the output. Thus, they are directly interpreted as indices of local sensitivity relative to each input. These quantities can be estimated by the code itself (when physical equations are solved) or by using sampling methods based on One-At-a-Time (OAT) designs. OAT methods consist of disturbing a single input at a time while setting the other inputs to their nominal values. See Morris (1991) for more details.

However, these LSA methods do not take into account the uncertainties of the inputs in their full range of variation. In order to quantify the overall impact of each input on the output and characterize their probabilistic dependence, statistical

methods of Global Sensitivity Analysis (GSA) have been developed. In contrast to LSA, the global approach requires the characterization of inputs uncertainty over their range of variations. For this purpose, a probabilistic framework is usually considered and probability distributions are associated to the uncertain inputs. In this context, a classic GSA method is based on a decomposition of the variance of the output, where each term of the decomposition represents the fractional contribution of an input or a group of inputs to the variance of the output (Sobol', 1993). The sensitivity indices obtained by this decomposition are called Sobol' indices. They have the advantage of being easily interpretable. However, they are computationally expensive to evaluate, even if efforts have been made to reduce this cost (Gamboa et al., 2022, 2016).

As mentioned in Chapter 3, tools based on dependence measures also have been proposed for Global sensitivity analysis (Da Veiga, 2015). The objective of these measures is to evaluate, from a probabilistic point of view, the dependence between the random variable representing the output value and the random variables representing the input parameters. In this frame, the Hilbert-Schmidt independence criterion (HSIC) presented in Chapter 3 distinguished itself by its capacity to capture a very broad spectrum of dependence forms between the variables while being relatively computationally inexpensive compared to other methods. Nevertheless, it has the disadvantage of being less easily interpretable from a quantitative point of view.

The objective of this last chapter is to present the methods used to perform a quantitative and global sensitivity analysis of the MC3D code from our initial experimental design. After having dealt with the first two steps in Chapter 2 and 3, the current chapter now tackles the last three steps of the methodology presented in Chapter 1. Thus, given the large number of input parameters of the MC3D code ($d = 57$), a screening step is first performed in Section 4.1 to reduce the number of inputs considered. The outputs are then approximated by regression models. This is discussed in Section 4.2. Finally, the resulting metamodels are used to perform a more refined sensitivity analysis of the outputs in Section 4.3. This work is processed on the $n = 1137$ MC3D converged simulations.

4.1 Screening by sensitivity analysis

The objective of screening is to distinguish inputs that have a significant impact on the outputs of the code from the others. This step is essential in the context of the presented work. As we have seen, the MC3D code actually takes into account a very large number of modeling parameters which can interact with each other. This screening step allows to focus on the most important parameters. This simplifies the following steps of our methodology, namely the metamodeling and the quantitative sensitivity analysis steps.

To perform this screening, two complementary GSA approaches from the Uncertainty Quantification (UQ) framework are used. The first one is the HSIC-based GSA (Gretton et al., 2005). As a reminder, the underlying idea of HSIC is to measure the statistical dependence between two random variables. This criterion can therefore capture the global influence of an input X_j on an output Y. Nevertheless, the information given by the HSIC is difficult to interpret. In other words, we know whether there is dependence between the variables tested or not, but we have limited information on the nature of this dependence. For a full description of HSIC and its use in sensitivity analysis, see Section 3.2 of this manuscript.

The second approach relies on variance-based sensitivity indices. These indices give a more complete and easy to interpret information about the dependency between the inputs and an output. Nevertheless, as mentioned before, their estimation is computationally expensive. In fact, only the causal relationship between the output and each input alone (without its interaction with the others) can be estimated using our MC3D input/output sample through the so called first order Sobol' indices. A full description of these Sobol' indices is given in the next subsection (subsection 4.1.1). Note that the first order Sobol' indices do not allow a screening as such since their nullity is not equivalent to independence. However, they provide additional information to the HSIC one, as it will be detailed later.

Finally, subsection 4.1.2 aims at presenting our approach, based on these two tools, to perform a variable selection on the MC3D modeling parameters. It describes how important inputs were distinguished from the others using HSIC and the complementary information on the additive part of the model of first order Sobol' indices. The obtained results are then analyzed.

4.1.1 Variance-based sensitivity analysis

Variance-based sensitivity analysis, proposed in (Sobol', 1993), gives a general framework to study the relationship between each input X_j and an output Y. To achieve this, the idea is to decompose the variance of the model function output into fractions which can be attributed to inputs or sets of inputs.

4.1.1.1 Sobol' indices: definition and properties

Let us consider a vector of independent inputs $\boldsymbol{X} = (X_1, \ldots, X_d)^T$ and a measurable model function $f : \mathbb{R}^d \to \mathbb{R}$ such that $\mathbb{E}(f(X_1, \ldots, X_d)^2) < +\infty$. This function can be decomposed into a sum of functions of increasing dimensions:

$$Y = f(X_1, \ldots, X_d) = f_0 + \sum_{j=1}^{d} f_j(X_j) + \sum_{1 \leq j < j' \leq d} f_{jj'}(X_j, X_{j'}) + \ldots + f_{1,\ldots,d}(\boldsymbol{X}). \quad (4.1)$$

Here:

- $f_0 = \mathbb{E}(Y)$;
- $f_j(X_j) = \mathbb{E}(Y|X_j) - \mathbb{E}(Y)$;
- $f_{j,j'}(X_j, X_{j'}) = \mathbb{E}(Y|X_{j'}, X_j) - \mathbb{E}(Y|X_j) - \mathbb{E}(Y|X_{j'}) + \mathbb{E}(Y)$ and so on.

This decomposition was first introduced in Hoeffding (1948) and taken over in Efron and C. (1981). To simplify writing, we define $\boldsymbol{u}$ as a subset $[\![1, d]\!]$. The decomposition (4.1) can therefore be written as a sum over all the subsets of $[\![1, d]\!]$:

$$Y = \sum_{\boldsymbol{u} \subset [\![1,d]\!]} f_{\boldsymbol{u}}(\boldsymbol{X}_{\boldsymbol{u}}). \quad (4.2)$$

Here, $\boldsymbol{X}_{\boldsymbol{u}}$ is the vector of the inputs $\{X_j\}_{j \in \boldsymbol{u}}$.

Using the decomposition 4.2, the variance of Y can be then written as follow:

$$\mathbb{V}\mathrm{ar}(Y) = \sum_{\boldsymbol{u} \subset [\![1,d]\!]} \mathbb{V}\mathrm{ar}\Big(f_{\boldsymbol{u}}(\boldsymbol{X}_{\boldsymbol{u}})\Big). \quad (4.3)$$

This decomposition leads to variance-based sensitivity indices. For a vector of inputs $\boldsymbol{X}_{\boldsymbol{u}}$, $u \subset [\![1, d]\!]$, the so-called Sobol' index (Sobol', 1993) is thus defined as follow:

$$S_{\boldsymbol{u}} = \frac{\mathbb{V}\mathrm{ar}\Big(f_{\boldsymbol{u}}(\boldsymbol{X}_{\boldsymbol{u}})\Big)}{\mathbb{V}\mathrm{ar}(Y)}. \tag{4.4}$$

In particular, if we take $u = \{j\}$ with $j \in [\![1, d]\!]$, we obtain the first order Sobol' indices of the input X_j briefly introduced in Chapter 2,

$$S_j = \frac{\mathbb{V}\mathrm{ar}\Big(f_j(X_j)\Big)}{\mathbb{V}\mathrm{ar}(Y)} = \frac{\mathbb{V}\mathrm{ar}\Big(\mathbb{E}(Y|X_j)\Big)}{\mathbb{V}\mathrm{ar}(Y)}.$$

The interpretation of these indices is straightforward. Indeed, all the indices are positive and, thanks to eq. (4.3), their sum is equal to 1. The index $S_{\boldsymbol{u}}$ quantifies the influence of the vector $\boldsymbol{u}$ on the output, independently from the other variables. An index $S_{\boldsymbol{u}}$ close to 1 indicates a high influence of $\boldsymbol{X}_{\boldsymbol{u}}$ on the output. Conversely, an index close to 0 indicates a low influence of $X_{\boldsymbol{u}}$ alone on the output. Note that the nullity of $S_{\boldsymbol{u}}$ is not equivalent to the independence between $X_{\boldsymbol{u}}$ and Y since it does not include the interaction effects between $X_{\boldsymbol{u}}$ and the other inputs.

For a function with d inputs, there is a total of $2^d - 1$ Sobol' indices. Thus in practice, when the dimension of the input space is large, the estimation of all the indices can be very tedious. To address this problem, Homma and Saltelli (1996) introduced the so called *total Sobol' indices*. These indices quantify the global influence of each input on the output, including their interactions with the other inputs. Formally, the total sensitivity index S_{T_j} associated with the input X_j is defined as the sum of all sensitivity indices related to the variable X_j:

$$S_{T_j} = \sum_{\boldsymbol{u} \subset [\![1,d]\!], j \in \boldsymbol{u}} S_{\boldsymbol{u}} \tag{4.5}$$

Let $\boldsymbol{X}_{-j} = (X_1 \dots X_{j-1}, X_{j+1}, \dots X_d)$ be the vector of all the inputs except the input X_j. Using the variance decomposition presented above, one can also define the Sobol' total index of the input X_j as follow:

$$S_{T_j} = 1 - \frac{\mathbb{V}\mathrm{ar}(\mathbb{E}(Y|\boldsymbol{X}_{-j}))}{\mathbb{V}\mathrm{ar}(Y)}. \tag{4.6}$$

These total indices summarize the global impact of an input on an output, including its interactions with other parameters. As for the other indices, their interpretation is direct. A high (resp. low) value of S_{T_j} indicates that, in total, the impact of X_j on the studied output is important (resp. negligible). The nullity of S_{T_j} is equivalent to the independence between X_j and and Y.

4.1.1.2 Monte-Carlo estimation of the Sobol' indices

Many Monte-Carlo methods, named Monte-Carlo Pick-freeze methods or simply Pick-freeze methods, have been proposed to estimate the Sobol' indices (Saltelli, 2002). These estimators are based on the following equality:

$$\mathbb{V}\mathrm{ar}\Big(f_{\boldsymbol{u}}(\boldsymbol{X}_{\boldsymbol{u}})\Big) = \mathbb{C}\mathrm{ov}(Y, Y'_{\boldsymbol{u}}), \tag{4.7}$$

with $Y = f(\boldsymbol{X}) = f(\boldsymbol{X}_{\boldsymbol{u}}, \boldsymbol{X}_{-\boldsymbol{u}})$ and $Y'_{\boldsymbol{u}} = f(\boldsymbol{X}_{\boldsymbol{u}}, \boldsymbol{X}_{-\boldsymbol{u}}{}')$.

Here, $\boldsymbol{X}_{-\boldsymbol{u}}$ is the vector of all inputs except the inputs $\boldsymbol{X}_{\boldsymbol{u}}$ and $\boldsymbol{X}_{-\boldsymbol{u}}{}'$ is an independent copy of $\boldsymbol{X}_{-\boldsymbol{u}}$. The name of the method, *Pick-Freeze* (PF), comes from the way the input parameters are used to compute $Y'_{\boldsymbol{u}}$ from those used to compute Y. In fact, the *picked* inputs $\boldsymbol{X}_{-\boldsymbol{u}}{}'$ are *frozen* and all the others are replaced by independent copies with the same distributions. An underlying hypothesis for this decomposition is that Y and ${Y_{\boldsymbol{u}}}'$ are following the same distribution and are independent conditionally to $\boldsymbol{X}_{\boldsymbol{u}}$. This is the case if the inputs are independent.

Thanks to the decomposition (4.7), we have $S_{\boldsymbol{u}} = \frac{\mathbb{C}\mathrm{ov}(Y, Y'_{\boldsymbol{u}})}{\mathbb{V}\mathrm{ar}(Y)}$ under the assumptions described previously. By using the empirical estimator of the covariance, an estimator for $S_{\boldsymbol{u}}$ can be built. For instance, let $\{(y^{(1)}, y_j'^{(1)}), \ldots, (y^{(n)}, y_j'^{(n)})\}$ be n independent realizations of $(Y, {Y_j}')$. The statistic $\widehat{S_j^n}^{PF}$ defined by (4.8) is an estimator of the Sobol' first order index associated to the input X_j, $j \in [\![1, d]\!]$:

$$\widehat{S_j^n}^{PF} = \frac{\frac{1}{n}\sum_{i=1}^{n} y^{(i)} y_j'^{(i)} - (\frac{1}{n}\sum_{i=1}^{n} y^{(i)})(\frac{1}{n}\sum_{j=1}^{n} y_j'^{(i)})}{\frac{1}{n}\sum_{i=1}^{n} (y^{(i)})^2 - (\frac{1}{n}\sum_{i=1}^{n} y^{(i)})^2}. \tag{4.8}$$

Higher order indices as well as total indices S_{T_j} can be estimated similarly using equation (4.7).

The total estimation cost of the first order and total indices is $O(n \times (d+1))$. With this estimation method, the number of code calls increases linearly with the number of input variables. This is a practical problem for domains with large input dimensions and CPU-expensive simulations. Another limitation of this estimation scheme arises from the need for the special pick-freeze design, which is not always available. For instance, regarding the computational cost of MC3D and the number of inputs considered, the direct use of the pick-freeze method is not an option for our application case. In fact, only one MC3D input/output sample is available for the whole presented methodology so the design used has to more generic (like Monte Carlo Sampling or Latin Hypercube Sampling for instance).

To address this problem, we have turned to a new estimator based on a rank statistic recently proposed by (Gamboa et al., 2022). This method enables the estimation of all the first-order indices from a Monte Carlo simple inputs/outputs sample. This estimator has been previously introduced (in Chapter 2 of this manuscript). Naturally, we will consider this estimation method for the screening step. The first order and the total indices of the built metamodels will be then computed using the classic Pick-Freeze method.

4.1.2 Application on MC3D

The HSIC statistics and the first-order Sobol' indices are estimated from the learning sample of the MC3D case. As mentioned before, Latin Hypercube Sampling method has been used to built this learning sample. The convergence of these statistics under LHS have been discussed in Chapter 2. Numerical studies indeed show good convergence capabilities of these statistics estimated from LHS design.

Regarding the values taken by each of these two indicators, a subset of inputs will be selected. An input selected by at least one of these two selection criteria is considered as influential in the rest of the study. In addition, as stated in Chapter 3, MC3D code failures caused dependencies between the inputs of the code. If at least one of the inputs selected by the procedure belongs to a group of dependencies, the whole set of variables belonging to this group is also considered as influential. Note that it is not possible to estimate directly the Sobol' indices of variables associated with each group. Indeed, the rank based estimation method only allows to estimate the first order indices.

For the HSIC test, the classic screening procedure based on p-values is processed again: variables whose p-values are below the level $\alpha = 0.05$ correspond to influential variables. For the Sobol' first-order indices, the selection process is quite similar. As a reminder, the higher the Sobol' index of an input, the higher the sensitivity of the output to that input (taken individually). Therefore, we consider inputs with a Sobol' index greater than $S_{min} = 0.05$ are influential. As mentioned in Chapter 3, these thresholds are fixed arbitrarily. Here they have been empirically established to facilitate metamodeling while maintaining a sufficient number of explanatory variables. On one hand, letting too many explanatory variables can lead to overfitting (in linear regression for instance) and/or optimization problems (in Gaussian process modeling notably) in the metamodeling. On the other hand, if we use too few explanatory variables, the part of the variance explained by the unaccounted for variables can become too important, which can lead to poor predictivity of the metamodel.

The results, obtained for the four outputs of interest, are summarized in Table 4.1. The mention *Total (Sobol' + HSIC)* corresponds to the number of inputs selected by at least one of the two approaches. The last line includes the group of dependencies. For each output, around fifteen inputs are initially identified as influential by the variable selection procedure. Taking into account the dependencies brings this total to about twenty inputs.

Table 4.1: *Number of inputs selected either by the HSIC based method or the first order Sobol' indices based method for the four studied outputs.*

	Y_1	Y_2	Y_3	Y_4
HSIC	10	16	17	15
Sobol'	3	6	5	5
Total (Sobol' + HSIC)	12	19	18	16
Total with dependent variables	17	23	20	17

4.2 Metamodeling

The term *regression analysis* gathers all statistical processes for estimating the relationships between a set of variables (often called *features* or *inputs*) and a target variable (often called *response variable* or *output*). These methods address two main objectives. First, regression analysis is widely used for prediction, especially in machine learning. Second, regression analysis can be used to infer causal relationships between the inputs and the outputs. Many regression methods have been proposed. For example, Generalized Linear Models briefly introduced in Chapter 2 (Nelder and Wedderburn, 1972), Support Vector Regression (Cortes and Vapnik, 1995), Neural Networks (F. Rosenblatt, 1962) or Gaussian processes (Rasmussen and Williams, 2005) are all regression methods. Each regression method has specific constraints, related to the amount and type of data we have access to and what we want to do with it.

In the context of uncertainty quantification, we use regression models, often called surrogate models or metamodels, to approximate or *emulate* expensive simulation codes. Simulation codes such as MC3D are indeed very computationally expensive. It is therefore difficult to perform analysis that require many simulation runs, such as sensitivity analysis (Da Veiga et al., 2021) or risk assessment (Marrel et al., 2022) for instance, directly using the code. To address this, surrogate models are trained on a reasonable number of simulations to emulate the outputs of interest. Once trained and validated, these metamodels can produce a large amounts of predictions of the code output with negligible computational effort. They can

therefore be interestingly used to perform the aforementioned analyzes (GSA and risk assessment).

In this work, surrogate models are used to perform a quantitative and global sensitivity analysis of the MC3D code outputs. This section aims to introduce the tools used for metamodeling in our work. An application on our industrial case and an analysis of the results is then provided.

Regression models are associated with a learning base. In our industrial context, it is $\boldsymbol{D}^n = (\mathbf{X}, \mathbf{Y}) = \left((\boldsymbol{x}^{(1)}, y^{(1)}) \dots, (\boldsymbol{x}^{(n)}, y^{(n)})\right)$ with a sample size $n = 1137$ and an input space dimension $d = 57$.

Here, $\mathbf{Y} = f(\mathbf{X}) \in \mathbb{R}^n$ with f the model function and $\mathbf{X} \in M_{n,d}(\mathbb{R})$ the input data set. Formally, the objective of the metamodeling is to approximate the function f by a regression model denoted $\hat{f} : \mathbb{R}^d \to \mathbb{R}$. The next subsection is dedicated to the simplest and the most common regression method, the linear regression. In our industrial application, it is used as a reference (in sense of basic) metamodel for benchmark.

4.2.1 Linear regression

4.2.1.1 Classic linear regression

Linear regression is a very common tool to approximate a response variable. The idea is to suppose that the response variable and the features are linked through a linear relationship. The model function f is approximated by a linear function of the d inputs $\hat{f}_{LR} : \mathbb{R}^d \to \mathbb{R}$ defined by the equation (4.9):

$$\hat{f}_{LR}(\boldsymbol{X}) = \boldsymbol{X}\boldsymbol{\beta} = \sum_{j=0}^{d} X_j \beta_j, \tag{4.9}$$

with $X_0 = 1$ [1]. Here, $\boldsymbol{\beta} = (\beta_0, \dots, \beta_d)^T \in \mathbb{R}^{d+1}$ is the vector of regression coefficients.

The values of $\boldsymbol{\beta}$ is estimated using the data set. For instance, a classic estimation method consist in minimizing the quadratic loss function [2] (4.10):

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{x}^{(i)} \boldsymbol{\beta} \right)^2 \right). \tag{4.10}$$

[1] To simplify the notations, the vector $(X_0, X_1, \dots, X_d)^T$ is also written $\boldsymbol{X}$.

[2] As mentioned in Owen (1992), the convergence of $\hat{\boldsymbol{\beta}}$ is ensured under LHS (Latin Hypercube Sampling) and its variance of estimation is lower than under pure Monte Carlo sampling.

Then, the least squares estimate of this regression can be computed using the formula (4.11):

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\boldsymbol{Y} \tag{4.11}$$

To improve the prediction accuracy and interpretability of linear regression models, parameter regularization can be performed. We present here three methods classically used for this: the ridge, the LASSO and the Elastic net regularization (which is a mix of the LASSO and the ridge). All three methods rely on a penalization of the coefficients of the regression $\boldsymbol{\beta}$ to reduce the overfitting risk.

4.2.1.2 Linear regression with LASSO regularization

Introduced in Tibshirani (1996), the LASSO (Least Absolute Shrinkage and Selection Operator) regularization is based on a L_1-penalization of $\boldsymbol{\beta}$ which is written, for $\lambda_1 \geq 0$:

$$\hat{\boldsymbol{\beta}}_{LASSO} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{d+1}}{\operatorname{argmin}}\left(\sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\beta}^T\boldsymbol{x}^{(i)}\right)^2 + \lambda_1||\boldsymbol{\beta}||_1 \right). \tag{4.12}$$

Here, $||.||_1$ is the L_1 norm. The higher the λ, the more coefficients are set to zero and therefore the fewer inputs are used in the regression. The LASSO regularization therefore allows to perform a variable selection. In practice, the parameter λ_1 is often estimated by cross-validation (Hastie et al., 2009). Cross validation is a re-sampling method that uses different portions of the data to test and train a model over different iterations.

4.2.1.3 Linear regression with Ridge regularization

The ridge regularization, introduced in Hoerl and Kennard (1970), relies on a L_2-penalization of $\boldsymbol{\beta}$ which is written, for $\lambda_2 \geq 0$:

$$\hat{\boldsymbol{\beta}}_{Ridge} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{d+1}}{\operatorname{argmin}}\left(\sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\beta}^T\boldsymbol{x}^{(i)}\right)^2 + \lambda_2||\boldsymbol{\beta}||_2^2 \right)$$

.{#eq:LossBeta_Ridge}

Here, $||.||_2$ is the L_2 norm (the Euclidean norm). The higher is λ_2, the stronger the regularization and thus the lower are the absolute values of the coefficients of the regression. In practice, the ridge regularization lowers the coefficients of

correlated inputs towards each other. As for the LASSO method, the parameter λ_2 is also estimated by cross-validation.

4.2.1.4 Linear regression with Elastic-net regularization

Finally, the Elastic-net regularization, introduced in Trevor Hastie (2005), combines the LASSO and the ridge regularization. The elastic-net minimization equation is written as follow:

$$\hat{\boldsymbol{\beta}}_{Elastic-net} = \underset{\boldsymbol{\beta}\in\mathbb{R}^{d+1}}{\operatorname{argmin}}\left(\sum_{i=1}^{n} \left(y^{(i)} - \boldsymbol{\beta}^T \boldsymbol{x}^{(i)}\right)^2 + \lambda_1 ||\boldsymbol{\beta}||_1 + \lambda_2 ||\boldsymbol{\beta}||_2^2 \right) \tag{4.13}$$

with $\lambda_1, \lambda_2 \geq 0$.

Linear regression, particularly with parameters regularization, is widely used as regression model because it offers very good results when the input/output relationship is linear. However, for industrial code like MC3D, this is rarely the case. More complex models such as Gaussian process regression might be better suited.

4.2.2 Gaussian process regression

Gaussian process regression (Rasmussen and Williams, 2005), sometimes called kriging, is widely used in the uncertainty quantification framework to emulate code outputs (Santner et al., 2003). The idea behind Gaussian process (GP) regression is to consider the (unkown) model function as a realization of a Gaussian process. Before defining more formally GP regression, let us first introduce the key concepts necessary to understand it, namely Gaussian vectors and Gaussian processes.

4.2.2.1 Gaussian vectors

Gaussian vectors, sometimes called multivariate Gaussian distribution or joint normal distribution, are a generalization of the univariate normal distributions to higher dimensions. More formally, a real valued random vector $\boldsymbol{W} = (W_1, \ldots, W_m)$ with $m \in \mathbb{N}^*$ is a Gaussian vector if, for any $\boldsymbol{\lambda} = (\lambda_1, .., \lambda_m) \in \mathbb{R}^m$, $\boldsymbol{\lambda^T W}$ has a Gaussian distribution. A Gaussian vector is fully characterized by its mean $\mathbb{E}(\boldsymbol{W}) = \boldsymbol{\mu} \in \mathbb{R}^m$ and its covariance matrix $\mathbb{C}\text{ov}(\boldsymbol{W}) = \boldsymbol{\Sigma} \in M_{m,m}(\mathbb{R})$.

An interesting property about Gaussian vectors concerns the conditioning. Let us consider a size m Gaussian vector $\boldsymbol{W} = (\boldsymbol{W}_A, \boldsymbol{W}_B)^T$ such that: $\boldsymbol{W} = \binom{\boldsymbol{W}_A}{\boldsymbol{W}_B} \sim \mathcal{N}_m\left(\binom{\boldsymbol{\mu}_A}{\boldsymbol{\mu}_B}, \begin{pmatrix} \boldsymbol{\Sigma}_{A,A} & \boldsymbol{\Sigma}_{A,B} \\ \boldsymbol{\Sigma}_{A,B}{}^T & \boldsymbol{\Sigma}_{B,B} \end{pmatrix}\right)$. Here,

- $\boldsymbol{W}_A$ and $\boldsymbol{W}_B$ are Gaussian vectors of size m_A and m_B respectively, such that $m_A + m_B = m$;
- $\boldsymbol{\mu}_A$ and $\boldsymbol{\mu}_B$ are vectors of size m_A and m_B respectively;
- $\boldsymbol{\Sigma}_{A,A} \in M_{m_A,m_A}(\mathbb{R})$, $\boldsymbol{\Sigma}_{A,B} \in M_{m_A,m_B}(\mathbb{R})$ and $\boldsymbol{\Sigma}_{B,B} \in M_{m_B,m_B}(\mathbb{R})$.

Thus, the conditional random vector $\boldsymbol{W}_A|\boldsymbol{W}_B$ is also a Gaussian vector. Supposing that $\boldsymbol{\Sigma}_{B,B}$ is invertible, the mean of this Gaussian vector is equal to $\boldsymbol{\mu}_A + \boldsymbol{\Sigma}_{A,B}\boldsymbol{\Sigma}_{B,B}^{-1}(\boldsymbol{W_B} - \boldsymbol{\mu_B})$ and its covariance matrix is equal to $\boldsymbol{\Sigma}_{A,A} - \boldsymbol{\Sigma}_{A,B}\boldsymbol{\Sigma}_{B,B}^{-1}\boldsymbol{\Sigma}_{A,B}^T$. This result is called the conditioning property of Gaussian vectors.

4.2.2.2 Gaussian processes

Let us now consider a probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Here, Ω is a sample space, $\mathcal{F}$ a σ-algebra on Ω and $\mathbb{P}$ a probability measure on $\mathcal{F}$. A stochastic process is a sequence $\{V(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathbb{R}^p}$, $p \in \mathbb{N}^*$. The random process $\{V(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathbb{R}^p}$ is said to be indexed by $\boldsymbol{x}$. At $\boldsymbol{x}$ fixed, the application $V : \Omega \to \mathbb{R}$ is a random variable. At $\omega \in \Omega$ fixed, the application $\boldsymbol{x} \to V(\boldsymbol{x})$ is a trajectory of the stochastic process.

A random process $\{V(\boldsymbol{x})\}_{\boldsymbol{x}\in\mathbb{R}^p}$ is a Gaussian process if and only if for every finite set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ in $\mathbb{R}^p$, $(V(\boldsymbol{x}_1), \ldots V(\boldsymbol{x}_m))$ is a Gaussian vector. A Gaussian process is characterized by a mean and a covariance function (also called kernel). They are defined, for $\boldsymbol{x}, \boldsymbol{x'} \in \mathbb{R}^p$, as follow:

- $m(\boldsymbol{x}) = \mathbb{E}(V(\boldsymbol{x}))$,
- $c(\boldsymbol{x}, \boldsymbol{x'}) = \mathbb{C}\,\mathrm{ov}(V(\boldsymbol{x}), V(\boldsymbol{x'}))$.

4.2.2.3 Covariance function in Gaussian processes

The covariance function mostly defines the Gaussian process as it controls its smoothness and its scale. Classically in emulation of computer experiments, chosen covariance functions are stationary, which means that they only depends on $(\boldsymbol{x}-\boldsymbol{x'})$, for $\boldsymbol{x}, \boldsymbol{x'} \in \mathbb{R}^p$, $p \in \mathbb{N}^*$.

For $x, x' \in \mathbb{R}$ they are of the form:

$$c(x, x') = \sigma^2 k_l(x - x'). \tag{4.14}$$

Here, $\sigma^2 \in]0, +\infty[$ is the variance parameter of the function and k_l is a symmetric and positive semi-definite function parametrized by the correlation length $l \in \mathbb{R}^*$. k_l is called *correlation function*. A convenient way to extend this definition to the multidimensional case ($p > 1$) is to take the tensor product of 1D correlation functions and a global variance parameter $\sigma^2 > 0$. It is defined, for $\boldsymbol{l} = (l_1, \ldots, l_p) \in \mathbb{R}^{*p}$, as follow:

$$c(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 k_{\boldsymbol{l}}(\boldsymbol{x} - \boldsymbol{x}') = \sigma^2 \prod_{j=1}^{p} k_{l_j}(x_j - x'_j). \tag{4.15}$$

This is the method used in this work to handle the multidimensional case. Let us now give some examples of classic correlation functions.

The nugget correlation function

This correlation function, which yields to the white Gaussian noise, is defined by:

$$k_{\boldsymbol{l}}(x, x') = \mathbf{1}_{x-x'=0}, \text{ for } x, x' \in \mathbb{R}. \tag{4.16}$$

Here, $\mathbf{1}$ is the indicator function. Note that the nugget correlation function does not depend on any correlation length. This kernel is rarely used alone, but it is often added to other covariance functions in GP regression. More details are given below.

The Gaussian correlation function

This function, for a correlation length $l \in \mathbb{R}^*$, is defined by:

$$k_l(x, x') = \exp\Big(- \frac{(x - x')^2}{l^2}\Big), \text{ for } x, x' \in \mathbb{R} \tag{4.17}$$

This covariance function is infinitely differentiable. This means that a GP with this covariance function has trajectories that have mean square derivatives of all orders. This strong regularity yields to very smooth trajectories which can be unrealistic in practice.

The ν-Matérn correlation function

The ν-Matérn correlation function is defined by:

$$k_l(x, x') = \frac{(\sqrt{2\nu}\frac{(\boldsymbol{x}-\boldsymbol{x}')}{l})^\nu}{\Gamma(\nu)2^{\nu-1}} K_v\Big(\sqrt{2\nu}\frac{(x-x)'}{l}\Big), \text{ for } x, x' \in \mathbb{R} \tag{4.18}$$

Here,

- $l \in \mathbb{R}^*$ is the correlation length ;
- $\nu \in \mathbb{R}^{+*}$ is the regularity parameter;
- $\Gamma : \mathbb{R}^{+*} \to \mathbb{R}^+$ the gamma function;
- $K_v : \mathbb{R} \to \mathbb{R}$ is a modified Bessel function (see Abramowitz and Stegun (1964) for details).

The parameter ν controls the smoothness of the GP. In fact, a GP with a ν-Matérn correlation function is continuous and q-time differentiable if and only if $\nu > q$ (Rasmussen and Williams, 2005). Besides, for $\nu \to +\infty$, we obtain the aforementioned Gaussian covariance function. In metamodeling, two popular choices for the regularity parameter are $\nu = \frac{3}{2}$ and $\nu = \frac{5}{2}$. They respectively yield to GP trajectories once and twice differentiable. Another special case is the one with $\nu = \frac{1}{2}$, also referred to as the exponential covariance function. This last covariance function only provides continuous GP trajectories.

4.2.2.4 GP regression

We have now all the material necessary to define Gaussian process regression. The idea behind GP regression is to consider the sample $\mathbf{Y} = (y^{(1)}, \dots, y^{(n)})$ we want to model as realizations of a Gaussian process indexed by the inputs sample $\mathbf{X} = \left(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(n)}\right)^T$. This GP is denoted $\{Y_{GP}(\boldsymbol{x})\}_{\boldsymbol{x} \in \mathbb{R}^d}$, d being the dimension of the input space. This GP is completely specified by its mean function $m(\boldsymbol{x})$ and its covariance function $c(\boldsymbol{x}, \boldsymbol{x}')$. The conditional distribution for any $\boldsymbol{x} = (x_1 \dots x_d) \in \mathbb{R}^d$ of $Y_{GP}(\boldsymbol{x})$ can be obtained analytically from the following joint distribution:

$$\begin{pmatrix} Y_{GP}(\boldsymbol{x}) \\ \boldsymbol{Y}_{GP}(\mathbf{X}) \end{pmatrix} \sim \mathcal{N}_{n+1}\left(\begin{pmatrix} m(\boldsymbol{x}) \\ \boldsymbol{m}(\boldsymbol{X}) \end{pmatrix}, \begin{pmatrix} c(\boldsymbol{x}, \boldsymbol{x}) & \boldsymbol{c}(\boldsymbol{x}, \mathbf{X}) \\ \boldsymbol{c}(\boldsymbol{x}, \mathbf{X})^T & \boldsymbol{C}(\mathbf{X}) \end{pmatrix}\right). \tag{4.19}$$

Here,

- $\boldsymbol{Y}_{GP}(\mathbf{X}) = \left(Y_{GP}(x^{(1)}), \dots Y_{GP}(x^{(n)})\right)^T$;
- $\boldsymbol{m}(\mathbf{X}) = (m(\boldsymbol{x}^{(1)}), \dots, m(\boldsymbol{x}^{(n)}))^T$;
- $\boldsymbol{c}(\boldsymbol{x}, \mathbf{X}) = (c(\boldsymbol{x}, \boldsymbol{x}^{(1)}), \dots c(\boldsymbol{x}, \boldsymbol{x}^{(n)}))$;
- $\boldsymbol{C}(\mathbf{X})$ is a matrix of $M_{n,n}(\mathbb{R})$ such that $C_{i,i'} = c(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(i')}), (i, i') \in [\![1, n]\!]^2$.

By applying the aforementioned conditioning property of Gaussian vectors to the joint distribution (4.19), we obtain that the conditional random process $Y_{GP}(\boldsymbol{x}) | \boldsymbol{Y}_{GP}(\mathbf{X}) = \mathbf{Y}$ is still a GP characterized by its mean $\hat{f}_{GP} : \mathbb{R}^d \to \mathbb{R}$ and its covariance function $\hat{c} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. They are defined as follows:

- $\hat{f}_{GP}(\boldsymbol{x}) = \mathbb{E}\Big[Y_{GP}(\boldsymbol{x}) | \boldsymbol{Y}_{GP}(\boldsymbol{x}) = \mathbf{Y}\Big] = m(\boldsymbol{x}) + \boldsymbol{c}(\boldsymbol{x}, \mathbf{X})\mathbf{C}^{-1}(\mathbf{X})(\mathbf{Y} - \boldsymbol{m}(\mathbf{X}))$
- $\hat{c}(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{C}\mathrm{ov}\Big[Y_{GP}(\boldsymbol{x}), Y_{GP}(\boldsymbol{x}')\Big] = c(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{c}(\boldsymbol{x}, \mathbf{X})\boldsymbol{C}^{-1}(\mathbf{X})\boldsymbol{c}(\boldsymbol{x}', \mathbf{X})^T$

Therefore, the predictive distribution for a new (unobserved) point $\boldsymbol{x}$ is the Gaussian distribution $\mathcal{N}(\hat{f}_{GP}(\boldsymbol{x}), \hat{c}(\boldsymbol{x}, \boldsymbol{x}))$. Moreover, from the Gaussian predictive distribution and the explicit formulations of $\hat{f}_{GP}(\boldsymbol{x})$ and $\hat{c}(\boldsymbol{x}, \boldsymbol{x})$, confidence inter-

vals (CI) in prediction can be built for any given α-level. This can be useful for uncertainty quantification framework.

To completely define the model, the mean function (also called trend) as well as the correlation function need to be specified. It is common to suppose that the trend is constant or linear: $m(\boldsymbol{x}) = \beta_0, \beta_0 \in \mathbb{R}$ or $m(\boldsymbol{x}) = \boldsymbol{X}\boldsymbol{\beta} = \sum_{j=0}^{n} X_j\beta_j$ with $\boldsymbol{\beta} = (\beta_0, \dots \beta_d)^T \in \mathbb{R}^{d+1}$ and $X_0 = 1$. Here, $\boldsymbol{\beta}$ (resp β_0) is the parameter used to control the learning process of the linear part of the model. The covariance function is often parameterized by a correlation length $\boldsymbol{l} = (l_1, \dots, l_d) \in \mathbb{R}^{*d}$ and a variance parameter $\sigma^2 > 0$. These parameters are also supposed unknown and need to be estimated. The choice of the type of correlation function is also crucial. Indeed, as mentioned above, it controls the smoothness and the scale of the approximation. In practice, several covariance functions (Gaussian, 5/2-Matérn, 3/2 Matérn, etc.) are tested and the one with the best predictivity is chosen.

4.2.2.5 Parameters estimation

Several approaches are possible to estimate GP regression parameters $(\boldsymbol{\beta}, \sigma^2, \boldsymbol{l})$, including the minimization of the model error with cross-validation strategy (Bachoc, 2013) or more classically the maximization of the model likelihood (Rasmussen and Williams, 2005). In our application case, the parameters will be estimated using the second method. In the GP regression framework, the observations are considered as realizations of a Gaussian process, $\boldsymbol{Y}_{GP}(\mathbf{X}) = \mathbf{Y}$. The distribution of the vector of observations Y can thus be written:

$$\mathbf{Y}|\boldsymbol{\beta}, \boldsymbol{l}, \sigma^2 \sim \mathcal{N}_n(\boldsymbol{m}(\mathbf{X}), \boldsymbol{C}(\mathbf{X})) \tag{4.20}$$

Here,

- $\boldsymbol{m}(\mathbf{X}) = (m(\boldsymbol{x}^{(1)}), \dots, m(\boldsymbol{x}^{(n)}))^T$ with m being parametrized by $\boldsymbol{\beta}$;
- $\boldsymbol{C}(\mathbf{X})$ is a matrix of $M_{n,n}(\mathbb{R})$ such that $C_{i,i'} = c(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(i')}), (i, i') \in [\![1, n]\!]^2$ and $c(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 k_{\boldsymbol{l}}(\boldsymbol{x} - \boldsymbol{x}'), (\boldsymbol{x}, \boldsymbol{x}') \in (\mathbb{R}^d)^2$.

The corresponding log-likelihood is therefore a function of the parameters and the observations. Suppose that the GP trend is linear such $m(\boldsymbol{x}) = \boldsymbol{X}\boldsymbol{\beta}$ with $\boldsymbol{\beta} = (\beta_0, \dots \beta_d) \in \mathbb{R}^{d+1}$ and $X_0 = 1$. Let also $\boldsymbol{K}_{\boldsymbol{l}}(\mathbf{X})$ be a matrix of $M_{n,n}(\mathbb{R})$ such that $\boldsymbol{C}(\mathbf{X}) = \sigma^2 \boldsymbol{K}_{\boldsymbol{l}}(\mathbf{X})$. To simplify the notations, the matrix $\boldsymbol{K}_{\boldsymbol{l}}(\mathbf{X})$ is written in the following $\boldsymbol{K}_{\boldsymbol{l}}$. Note that, since the function $k_{\boldsymbol{l}}$ is symmetric and positive-definite, $\boldsymbol{K}_{\boldsymbol{l}}$ is invertible. The log-likelihood can be thus written as follow:

$$\mathcal{L}(\mathbf{Y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{l}, \sigma^2) = -\frac{1}{2}\Big[\log|\sigma^2 \boldsymbol{K}_{\boldsymbol{l}}| - \frac{1}{\sigma^2}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})^T \boldsymbol{K}_{\boldsymbol{l}}^{-1}(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})\Big] \tag{4.21}$$

The Maximum Likelihood (ML) estimator of these parameters corresponds the values of the model parameters that maximize the function $\mathcal{L}$ over the parameters $\boldsymbol{\beta}$, σ^2 and $\boldsymbol{l}$. The maximum values of $\mathcal{L}$, for $\boldsymbol{\beta}$ and σ^2, can be found by canceling the partial derivatives of $\mathcal{L}$, given the other parameters. The derivatives of the function $\mathcal{L}$ regarding $\boldsymbol{\beta}$ and σ^2 can indeed be explicitly expressed:

$$\frac{\partial \mathcal{L}(\mathbf{Y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{l}, \sigma^2)}{\partial \boldsymbol{\beta}} = \frac{1}{2\sigma^2} \mathbf{X}^T \boldsymbol{K}_{\boldsymbol{l}}^{-1} (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) \tag{4.22}$$

$$\frac{\partial \mathcal{L}(\mathbf{Y}, \mathbf{X}, \boldsymbol{\beta}, \boldsymbol{l}, \sigma^2)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4} (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})^T \boldsymbol{K}_{\boldsymbol{l}}^{-1} (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}) \tag{4.23}$$

Using equation (4.23) and (4.22) to solve $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}} = 0$ and $\frac{\partial \mathcal{L}}{\partial \sigma^2} = 0$ allows the ML estimators $\hat{\boldsymbol{\beta}}_n$ and $\hat{\sigma}_n^2$ to be explicitly computed, given the parameter $\boldsymbol{l}$:

$$\hat{\boldsymbol{\beta}}_n(\boldsymbol{l}) = (\mathbf{X}^T \boldsymbol{K}_{\boldsymbol{l}} \mathbf{X})^{-1} \mathbf{X}^T \boldsymbol{K}_{\boldsymbol{l}}^{-1} \mathbf{Y} \tag{4.24}$$

$$\hat{\sigma}_n^2(\boldsymbol{l}) = \frac{1}{n} (\mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}}_n)^T \boldsymbol{K}_{\boldsymbol{l}}^{-1} (\mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}}_n) \tag{4.25}$$

By injecting $\hat{\boldsymbol{\beta}}_n(\boldsymbol{l})$ and $\hat{\sigma}_n^2(\boldsymbol{l})$ in the log-likelihood expression (4.21), the parameter $\boldsymbol{l}$ can be estimated by solving numerically:

$$\hat{\boldsymbol{l}}_n = \underset{\boldsymbol{l} \in \mathbb{R}^d}{\operatorname{argmin}} |\hat{\sigma}_n^2(\boldsymbol{l}) \boldsymbol{K}_{\boldsymbol{l}}| \tag{4.26}$$

Finally, the values of $\hat{\boldsymbol{\beta}}_n$ and $\hat{\sigma}_n^2$ are deduced using the estimated value of $\hat{\boldsymbol{l}}_n$ with Eq. 4.25 and Eq. 4.26.

4.2.2.6 Additional nugget effect

An additional nugget effect can also be considered in the kernel. This means that one assumes an additive, independent, constant white noise (such as defined in Eq (4.16)). This noise is different from the noise which can affect the observations. In most cases, it is assumed to be independent of the input: its variance is constant. This nugget effect both relaxes the interpolation properties of the GP regression and improves the conditioning number of the covariance matrix (GP regularization). This additional noise is parametrized by a scalar variance parameter τ. This parameter can be fixed or estimated (by maximum-likelihood for instance).

4.2.3 Metamodeling validation criteria

4.2.3.1 Predictivity coefficient Q^2 and $y/\hat{y}$ plot

To evaluate the accuracy of a regression model $\hat{f}$, a common method consists in calculating, on a test sample $\boldsymbol{D}_{TEST}^{n_{TEST}} = \left((\boldsymbol{x}^{(1)}, y^{(1)}), \ldots, (\boldsymbol{x}^{(n_{TEST})}, y^{(n_{TEST})})\right)$ independent from the learning sample $\boldsymbol{D}^n$, the coefficient of determination in prediction, denoted Q^2_{TEST} in this case:

$$Q^2_{TEST} = 1 - \sum_{i=1}^{n_{TEST}} \frac{(y^{(i)} - \hat{f}(\boldsymbol{x}^{(i)}))^2}{(y^{(i)} - \bar{y}_{n_{TEST}})^2}, (\boldsymbol{x}^{(i)}, y^{(i)})_{j \in [\![1, n_{TEST}]\!]} \in \boldsymbol{D}_{TEST}^{n_{TEST}}, \tag{4.27}$$

with $\bar{y}_{n_{TEST}} = \frac{1}{n_{TEST}} \sum_{i=1}^{n_{TEST}} y^{(i)}$.

The interpretation of this coefficient is straightforward. The closer to 1 this coefficient, the better the accuracy of the metamodel. In practice, a validation sample is not always available and the predictivity coefficient can then be calculated by a cross-validation procedure (Hastie et al., 2009). In particular, a Leave-One-Out (LOO) approach can be used. The idea of LOO is to estimate, for each point i at our disposal, the regression model $\hat{f}^*(\boldsymbol{x}^{(i)})$, where $\hat{f}^*$ represents the model trained on $\boldsymbol{D}^n$ without the ith sample. By repeating this n times, one can obtain another estimation of this predictivity coefficient Q^2_{LOO}:

$$Q^2_{LOO} = 1 - \sum_{i=1}^{n} \frac{(y^{(i)} - \hat{f}^*(\boldsymbol{x}^{(i)})^2}{(y^{(i)} - \bar{y}_n)^2}, (y^{(i)}, \boldsymbol{x}^{(i)})_{i \in [\![1, n]\!]} \in \mathbf{D^n}. \tag{4.28}$$

In the following, we will only use Q^2_{LOO}, which we will write Q^2 to simplify the notations.

Another way to evaluate the quality of a regression model is to plot the true values of the model output $\boldsymbol{Y}$ as a function of the values predicted by your model by LOO $\hat{\boldsymbol{Y}}^* = \hat{f}^*(\mathbf{X})$. This plot gives us information on the prediction quality of the metamodel according to the value taken by the output. The interpretation of this graphic is also straightforward. The more points there are around the identity line (first bisector), the more points are such that $\hat{y}^* \approx y$ and so the better is the predictivity of the metamodel. This graph is called in this document the $y/\hat{y}$ plot.

4.2.3.2 α-CI plot for GP regression models

GP regression has the specificity of not only giving a predicted value, but also associating it with a probability distribution. It can therefore be interesting to compare the predicted confidence intervals (CI) of GP regression models with the proportions of observations that actually fall within these intervals. This was suggested in Demay et al. (2021). In practice, for a given theoretical confidence level $\alpha \in [0, 1]$, the observed CI level $\hat{\Delta}(\alpha)$ is calculated by Leave-One-Out (LOO) as follows:

$$\hat{\Delta}(\alpha) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{y^{(i)} \in CI_{\alpha,-i}}$$

Here, the notation $\mathbf{1}_A$ denotes, for any set A, the characteristic function of A. $CI_{\alpha,-i}$ is the α-level of confidence interval for the point $\boldsymbol{x}^{(i)}$ built from the Gaussian distribution $\mathcal{N}(\hat{f}^*_{GP}(\boldsymbol{x}^{(i)}), \hat{\sigma}^{*(i)})$. $\hat{f}^*_{GP}(\boldsymbol{x}^{(i)})$ and $\hat{\sigma}^{*(i)}$ are the mean and the variance of the predicted distribution when $(x^{(i)}, y^{(i)})$ is removed from the observations. Roughly speaking, the quantity $\hat{\Delta}(\alpha)$ corresponds to the rate of realizations which fall within the α-predicted intervals.

This *observed* rate $\hat{\Delta}(\alpha)$ can be plotted as a function of the theoretical levels of prediction α. In Demay et al. (2021), this plot is called the α-CI plot. The more points there are around the identity line (meaning that the intervals $\hat{\Delta}(\alpha)$ are close to the α), the more accurate the estimated GP intervals are. If the $\hat{\Delta}(\alpha)$ are above the identity line, it means that the confidence intervals predicted are too large (the model is *underconfident*). At the contrary, if the $\hat{\Delta}(\alpha)$ are under the identity line can mean, if the predictivity Q^2 is good, that the confidence intervals predicted are too small (the model is *overconfident*). It can also mean, if the Q^2 is low, that the GP mean is badly predicted. More generally, poorly predicted confidence intervals can also mean that the modeling assumptions (covariance and mean functions) or the Gaussian hypothesis are invalid for the output we are trying to approximate.

4.2.4 Application to our case study

Due to the large number of inputs and the CPU time for running MC3D, the use of metamodels is required to perform a complete quantitative sensitivity analysis of the code. They will also be very useful for the calibration of the code with experimental results, this step being the continuation of this thesis work. In this subsection, we detail the procedure followed to build the metamodels on the four MC3D outputs of interest. This process is performed on each output independently.

Thanks to the screening step, the input variables are divided into two groups for each output:

- the explanatory inputs, which corresponds to the inputs identified as significantly influential by the screening and denoted $\boldsymbol{X_{exp}}$,
- the other inputs, denoted $\boldsymbol{X}_\epsilon$ with $\boldsymbol{X}_\epsilon = \{\boldsymbol{X} \setminus \boldsymbol{X_{exp}}\}$.

The metamodeling process is focused on fitting each output only considering the explanatory inputs $\boldsymbol{X_{exp}}$. The method used to perform this metamodeling is a Gaussian process regression. In theory, the GP regression could be made directly on the $d = 57$ inputs. However in practice, building the GP in such large dimension without preliminary variable selection often leads to issues in the optimization procedure required for the estimation of the GP hyperparameters (resolution of Eq. 4.21). This most often yields to a GP with poor predictivity.

Furthermore and as discussed before, the choice of the covariance function can greatly influence the accuracy of regression. Several covariance functions have been therefore tested: the exponential, the Gaussian, the 3/2-Matérn and the 5/2-Matérn. A linear trend function is considered (with a preliminary Elastic-net based selection for the trend). The constant trend have also been tested, but the GP models obtained gave less good results. All the GP hyperparameters are estimated by maximum likelihood on the learning sample. The residual effect of the inputs $\boldsymbol{X}_\epsilon$ is captured using an additional *nugget effect*, also estimated by maximum likelihood.

The quality of the regression models is then assessed. The main quantitative tool used for this is the predictivity coefficient Q^2, estimated by Leave-One-Out. The performances of GP regressions with different covariance functions are compared through this index. A simple linear regression (with Elastic net variable selection) is also performed for benchmarking.

Table 4.2 summarizes the results of this comparison. The first thing we can observe is that linear regression does not fit well three of the four outputs. GP models provide better results. Indeed, we see for example that the predictivity of the GP models of the maximum impulsion (Y_3) is at least 31% better than that of the linear regression. These results underline the highly non-linear behavior of the MC3D code.

Let us now examine the influence of the choice of the covariance function on the results. First of all, we denote that the exponential kernel gives significantly worse prediction results than the other covariance functions. The other three seem to be equivalent with respect to the Q^2. However, the 5/2-Matérn covariance appear to be slightly better in general.

Table 4.2: *Predictivity coefficient Q^2 estimated by LOO for each GP regression and the four MC3D studied outputs.*

	Y_1(vm)	Y_2(dsauter_p)	Y_3(im_k)	Y_4(mf)
Linear regression	0.36	0.68	0.38	0.54
GP with exponential kernel	0.59	0.79	0.50	0.63
GP with Gaussian kernel	0.60	0.84	0.51	0.67
GP with 3/2-Matérn kernel	0.60	0.84	0.53	0.66
GP with 5/2-Matérn kernel	0.61	0.84	0.52	0.67

Let us go further in the analysis of the metamodeling performances. Figure 4.1 and Figure 4.2 provide the $y/\hat{y}$ and α-CI plots obtained for each of the approximated outputs, with a 5/2-Matérn GP. The results obtained for the other covariance functions, not displayed here for concision, are similar. Experimentally observed values and their uncertainties are shown in red on the $y/\hat{y}$ plots.

The best approximated output is Y_2. Its predictivity Q^2 is indeed over 80% and the prediction levels are also well approximated according to the $\alpha-$CI plot. For this output, linear effects seem to be important, as indicated by its good approximation by a linear regression. This can explain these good results. Y_4 is also correctly approximated, although not as well as Y_2. The GP has a predictivity close to 70% and the prediction levels are also well approximated (cf. Figure 4.2, bottom right).

The output Y_1 is not correctly predicted by the GP regression. First, we observe that only 60% of the variance of the output is explained by the GP metamodel ($Q^2 = 0.6$). Furthermore its α-CI plot shows that the model tends to overestimate the confidence intervals (especially around $\alpha = 0.5$). This may indicate that the modeling assumptions (the stationarity of the covariance for example) are not verified by the output Y_1. We also observe on the $y/\hat{y}$ plot, that the values taken by Y_1 are very unevenly distributed. In fact, we may remark that the majority of the realizations are located on very low values (around 0.005), corresponding to experimental results.

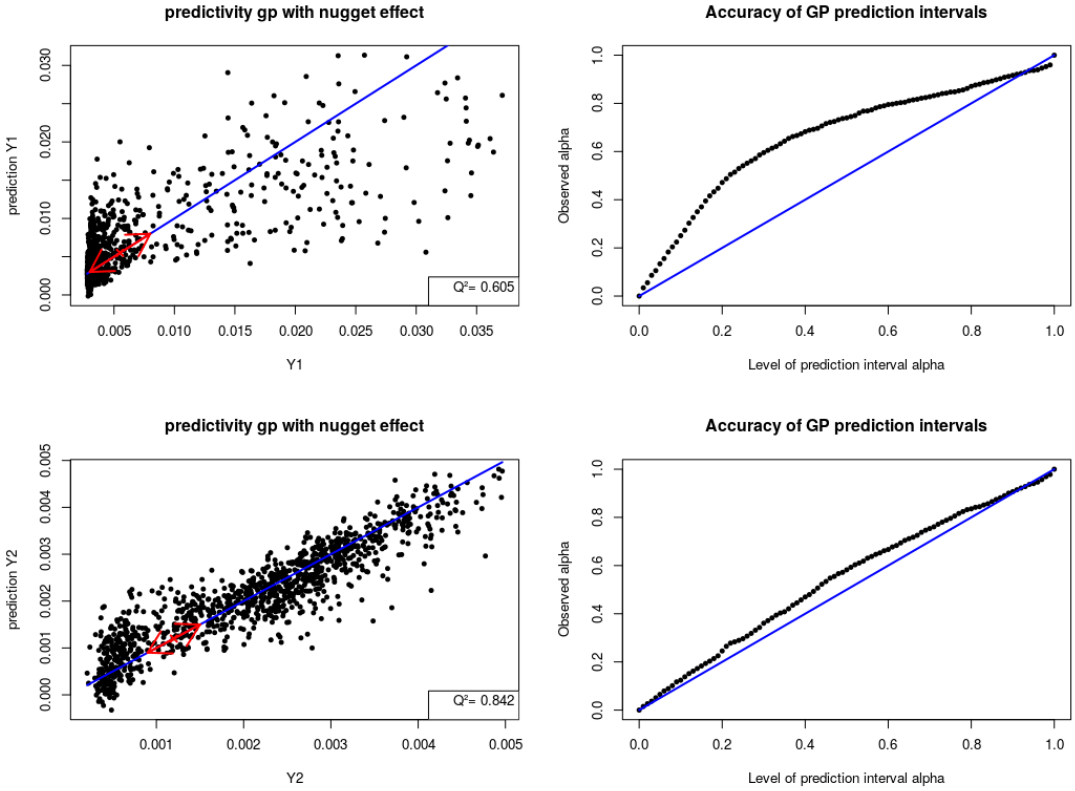

Figure 4.1: *Evaluations of GP performances of the two outputs Y_1 and Y_2. On the left: predicted vs. values simulated by MC3D. Experimentally observed values and their uncertainty are in red. On the right: α-CI plots.*

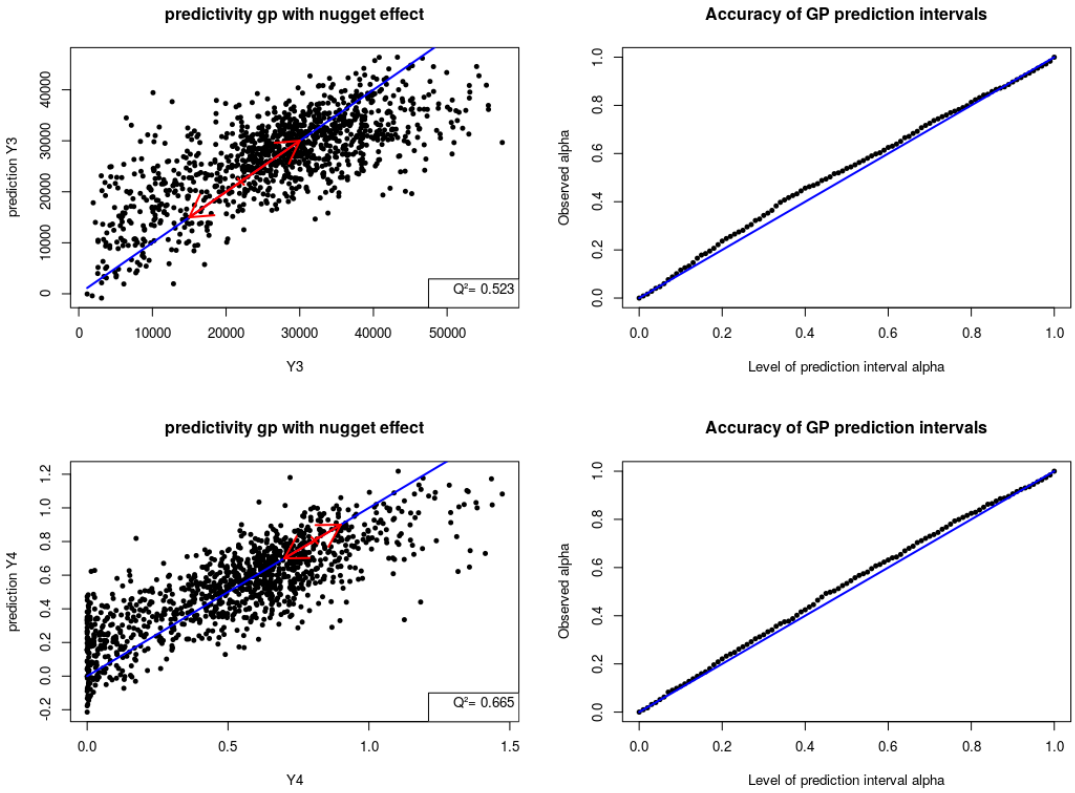

Figure 4.2: *Evaluations of GP performances of the two outputs Y_3 and Y_4. On the left: Predicted vs. values simulated by MC3D. Experimentally observed values and their uncertainty are in red. On the right: α-CI plots.*

Finally, the output Y_3 is the most poorly fitted by GP regression regarding the predictivity ($Q^2 = 0.53$). This means that almost half of the output variance is not explained by our GP metamodel. As mentioned in Chapter 1, the output Y_3 (named *im_K*) corresponds to the maximum impulsion measured on the different sensors of the installation. The steam explosion is a highly non linear and chaotic phenomenon. It is therefore understandable that this output is difficult to approximate. An interesting thing to notice is that the confidence intervals are well estimated by the GP. But as mentioned previously, it is not sufficient to ensure accurate prediction intervals since the Q^2 is low.

Now let us focus on the metamodeling around the experimental values. A first observation we can make is that the areas of interest, including the observed values and their associated uncertainties, are on average much smaller than the set of observed values. We also observe empirically on the $y/\hat{y}$ plots that current metamodeling is not better in these areas of interest, except perhaps for Y_1. An interesting perspective could be to start new MC3D simulations with reduced input variation ranges to better focus on around observed values in the metamodeling process.

In summary, the approximation of at least two of the four outputs studied (Y_1 and Y_3) could be significantly improved. As it is, the built metamodels are difficult to use for calibration. They remain interesting for sensitivity analysis. However, it should be kept in mind that a non-negligible part of the output variance is not taken into account by these surrogate models.

Several hypotheses can explain the obtained metamodeling results. The most obvious is the lack of training data. This explanation is explored in the subsection 4.2.5. Another hypothesis concerns a poor estimation of the hyperparameters. Even if dimension reduction was performed, the number of inputs considered remains important in our case (about 20 parameters per output). However, estimating GP hyperparameters by maximum likelihood in high dimension is a difficult problem. See (Basak et al., 2021) for more information. Another explanation could be that the GP modeling hypothesis are not well suited for the poorly approximated outputs. In particular, the stationarity hypothesis may be unadapted, at least over the whole input range of variation. As already mentioned, it might also be interesting to reduce the input space to focus more on the experimental values.

Several other possibilities have been explored to improve GP regression modeling. In particular, we tried to apply transformations to the output variables before metamodeling. We also considered joint metamodeling of the outputs using a PCA (Principal Component Analysis) of the vector of outputs before emulating the main PCA components by GP regression. Nevertheless, these different attempts did not significantly improve the accuracy of the GP metamodels.

4.2.5 Impact of the data size on metamodeling performances

As mentioned previously, an intuitive idea to improve metamodeling is to add new training data. Enrichment of learning base is often a good way to improve the accuracy of metamodels. Nevertheless and as we have seen, the MC3D code is very expensive in terms of computation time. Therefore, before launching new simulations, it is important to assess and evaluate the interest of adding data to the GP metamodels.

For this purpose, we measured the prediction quality of metamodels with Q^2 for data sets of different sizes. These data sets are obtained by re-sampling the database available. For each sub-sample and for each output, a GP metamodel is trained and its Q^2 prediction coefficient is evaluated. The trained GPs are the same as before: a 5/2-Matérn GP with a linear trend and an additional nugget effect. This process is repeated $L = 10$ times to get an average Q^2 for each sample size and each output. The number L of repetitions is quite low, but it already gives a first idea of the evolution of the Q^2 regarding sample size. Figure 4.4 depicts the results for the four studied outputs.

First, we can observe a stagnation of the Q^2 for all the four outputs from $n = 400$ or 600 according to the output. For example, the average Q^2 associated with Y_2 increases by 0.09 (a 12% gain) when the database increases from 200 to 600 points. Between 600 and 1000 points it is only 0.02 (2.5% gain).

Second, the sample size affects the Q^2 differently depending on the output. In fact, for the least well-fitted output Y_3, going from $n = 200$ to $n = 1000$ actually improves Q^2 by only 0.03 (+6%). Going from $n = 200$ to $n = 1000$ increases the Q^2 of Y_2 by 0.11 (+15%). We can therefore suppose that increasing the learning base affects the predictive ability of well-approximated outputs more than the poorly approximated ones. To conclude, it is reasonable to think that adding new simulations will not yield to significative improvements of GP performances.

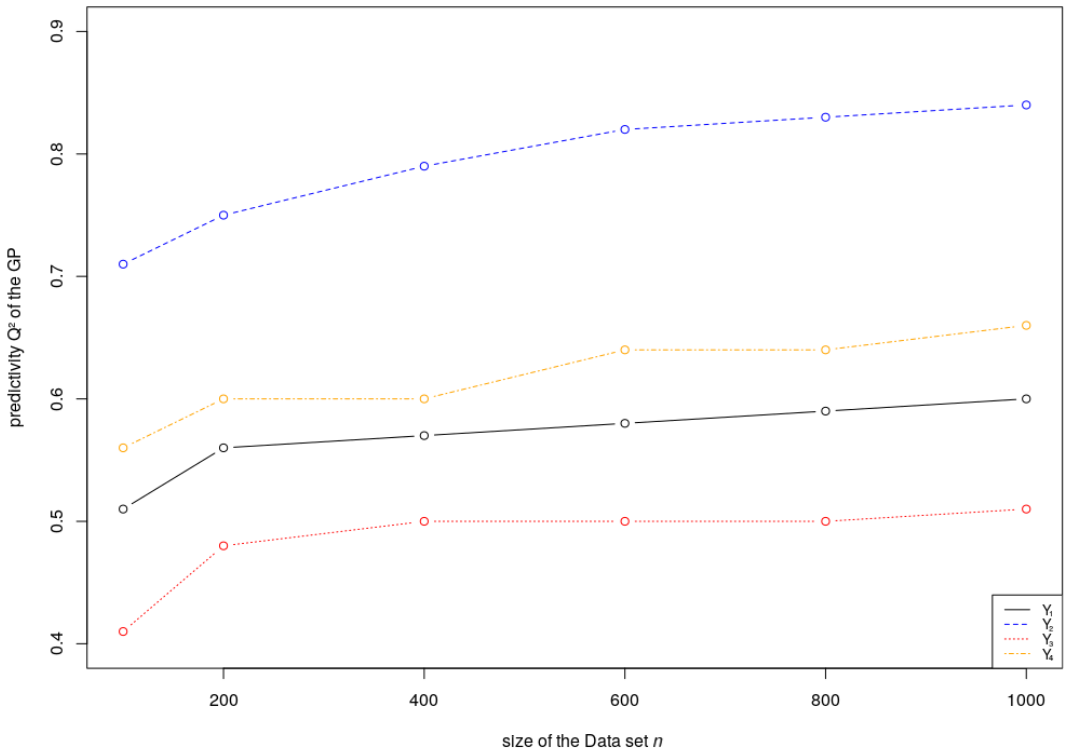

Figure 4.3: *Evolution of the GP's predictivity Q^2 for each output regarding the training set size.*

4.3 Quantitative sensitivity analysis on the built Gaussian process

The computation of total Sobol' indices in addition to first-order indices provides a more complete and quantitative sensitivity analysis since they take into account the effect of interactions. Moreover, their interpretation is easier than the HSIC. However, estimating Sobol' total indices, as we saw in Section 4.1, requires a large number of model evaluations. This is unmanageable for time-consuming computer codes like MC3D.

To address this issue, we can use surrogate models to have a good estimate of these indices. Note that, as mentioned before, a link can be made between the estimation error of the Sobol' indices estimated by a metamodel and the prediction coefficient Q^2 of this surrogate model. In fact, when the Q^2 is estimated under the probability distribution of the inputs, it provides an estimate of the part of the variance explained by the metamodel. Metamodel-based GSA provides sensitivity indices of the *true* model only on this explained part of variance. Thus, the more accurate a metamodel, the more relevant the metamodel-based sensitivity analysis. This must be taken into account when interpreting the Sobol' indices estimated with surrogate models.

Therefore, we estimated all first-order and total Sobol' indices using the GP regression models of the MC3D outputs built in Section 4.2. The GP considered is again with a 5/2-Matérn covariance, a linear trend and an additional nugget effect. Dependencies between inputs caused by code failures are neglected here for the sake of simplicity. Furthermore, the estimate of the Sobol' indices is only based on the trend of the GP metamodels [3]. Figure **??** summarizes the results obtained in the form of two colored tables. For a better understanding, the GP metamodels predictivity Q^2 of the four outputs is recalled on the first line, next to the output name.

A first thing to note is that first order effects represent an important part of the variance explained by the GP. Indeed, the sum of the first order indices represents, for each of the four outputs, between 65% and 90% of the total variance. We also observe that the least well-fitted outputs Y_1 and Y_3 have the lower part of their variances explained by first order effect (65% and 77% respectively).

Three inputs distinguish themselves regarding these results: X_{29} (corresponding to WECRDL), X_{13} (corresponding to CFRAG) and X_7 (corresponding to KTDROP).

[3] Several methods have been developed to associate an uncertainty to Sobol' estimates using GP metamodels (Marrel et al., 2009 ; Gratiet et al., 2014). They are not considered here.

	Y_1 (Q^2 = 0.6)	Y_2 (Q^2=0.84)	Y_3 (Q^2=0.53)	Y_4 (Q^2 = 0.67)
1	0	0,02	NA	0,01
2	NA	0,03	NA	0,03
5	NA	0	NA	0,01
6	NA	0,05	0,05	0,05
7	0,14	0,13	0,1	0,09
10	NA	0	0	NA
11	0,02	0,04	0,04	0,08
13	0,15	0,22	0,17	0,22
16	NA	NA	NA	0
18	NA	NA	0	NA
25	0,03	0,05	0,05	0,07
29	0,25	0,26	0,26	0,26
32	0,01	0,06	0,01	0,01
34	NA	0	NA	NA
36	0	NA	NA	NA
37	0,02	0,01	0	0,01
39	0,03	0	0	0,01
40	NA	0	NA	NA
41	NA	0	NA	0,01
44	NA	NA	0	NA
45	0	0,02	NA	NA
46	NA	0	NA	NA
51	NA	NA	NA	0
52	NA	NA	0,06	0,05
53	NA	NA	0	NA
55	NA	NA	0	NA
56	0	NA	0	0
57	NA	0	0,02	NA
Total	0,65	0,87	0,76	0,9

	Y_1 (Q^2 = 0.6)	Y_2 (Q^2=0.84)	Y_3 (Q^2=0.53)	Y_4 (Q^2 = 0.67)
1	0,01	0,03	NA	0,01
2	NA	0,04	NA	0,04
5	NA	0,01	NA	0
6	NA	0,07	0,06	0,05
7	0,32	0,16	0,19	0,13
10	NA	0	0,01	NA
11	0,05	0,05	0,06	0,09
13	0,31	0,26	0,23	0,27
16	NA	NA	NA	0
18	NA	NA	0	NA
25	0,06	0,06	0,07	0,09
29	0,48	0,3	0,34	0,3
32	0,06	0,09	0,04	0,05
34	NA	0	NA	NA
36	0	NA	NA	NA
37	0,02	0,02	0,03	0,02
39	0,06	0,01	0,01	0,01
40	NA	0	NA	NA
41	NA	0,01	NA	0,01
44	NA	NA	0,01	NA
45	0,01	0,02	NA	NA
46	NA	0	NA	NA
51	NA	NA	NA	0
52	NA	NA	0,06	0,06
53	NA	NA	0,01	NA
55	NA	NA	0	NA
56	0	NA	0,01	0
57	NA	0	0,08	NA

Figure 4.4: *Sobol' first order (on the left) and total (on the right) indices estimated on the GP metamodels. The line "Total" corresponds to the sum of all the first order indices for each output.*

In particular, the first order indices related to X_{29} are all around 0.25. This means that around 25% of the GP-explained variance of each output is due to this input alone. The Sobol' first order indices of the two other inputs are comprised between 0.09 and 0.22. This remains high regarding the number of inputs considered (between 17 and 23). For these inputs, the total indices are generally significantly larger than the first order indices. For X_{29}, we have $S_{T_{29}} - S_{29} = 0.23$. For X_{13} and X_7, this difference is equal to 0.16 and 0.18. This means that these three inputs not only influence the outputs individually, but also through their interactions with each other.

These three inputs are related in the code to the mechanism of drop fragmentation. Indeed, the input WECRDL (X_{29}) corresponds to the critical Weber number under which there is no fragmentation of the molten corium droplets surrounded by liquid water. The Weber number is an adimensionnal number representing the ratio of the inertial forces applied on the drops by the surrounding fluid to the forces of surface tension that stabilize the droplets. The input CFRAG (X_{13}) is the coefficient related to the Pilch's model of the drop fragmentation mentioned in Chapter 1. Finally, KTDROP (X_7) is related to the solidification temperature of the drops. Since only melted drops can be fragmented, it is also linked to the drops fragmentation. This indicates that the drop fragmentation model is one of the most influential mechanisms on the MC3D code output.

Let us go further in the analysis. Figure 4.5, 4.6 and 4.7 show scatterplots of the four outputs according to the three inputs X_{29}, X_{13} and X_7 respectively. An additional local regression using a first degree polynomial model (Wand and Jones, 1994) is added on each scatterplot. For each input $X_j, j \in [29, 13, 7]$ and each output $Y_k, k \in [\![1, 4]\!]$, this gives a rough estimate of $\mathbb{E}(Y_k|X_j)$ and therefore allows a first interpretation of the primary influence of each input. A general observation that can be made is that the outputs are, on average, monotonic functions of the three studied inputs.

Let us now analyze the relationship between WECRDL (X_{29}) and the four studied outputs (Figure 4.5). The Weber number is defined by the following formula:

$$We = \frac{\rho v^2 D}{\sigma}, \tag{4.29}$$

with:

- ρ is density of the surrounding fluid, here liquid water (kg/m^3);
- v is the difference of velocity between the drop and the surrounding fluid (m/s);
- D is the drop diameter (m);
- σ is the surface tension (N/m).

We can notice that the this number is proportional to the diameter of the droplets. This means that the higher the critical Weber, the higher the drop diameter can be. This explain why the output *dsauter_p* (Y_2), representing the average drop diameter in the premixing step, appears to be a growing function of WECRDL (X_{29}). A possible consequence of bigger drop diameter is that the interfacial area (the average drop surface area per unit volume of the drops) is smaller. It means that there is less thermal exchange between drops and liquid water and therefore less vapor is produced. This hypothesis seems to be confirmed by the first plot of Figure 4.5. We indeed observe that the vapor mass (Y_1) is a decreasing function of X_{29}. Finally, we observe that high values of X_{29} amplify the vapor explosion. Indeed, we observe that the maximum impulse (Y_3) and the fragment mass are growing functions of X_{29}. This can be explained from a physical point of view by the fact that only the molten drops of corium in liquid water are involved in the steam explosion phenomenon. However, the less steam there is, the more drops are in direct contact with the liquid water and can be fragmented by the explosion. The steam can also mitigate the pressure wave generated by the explosion.

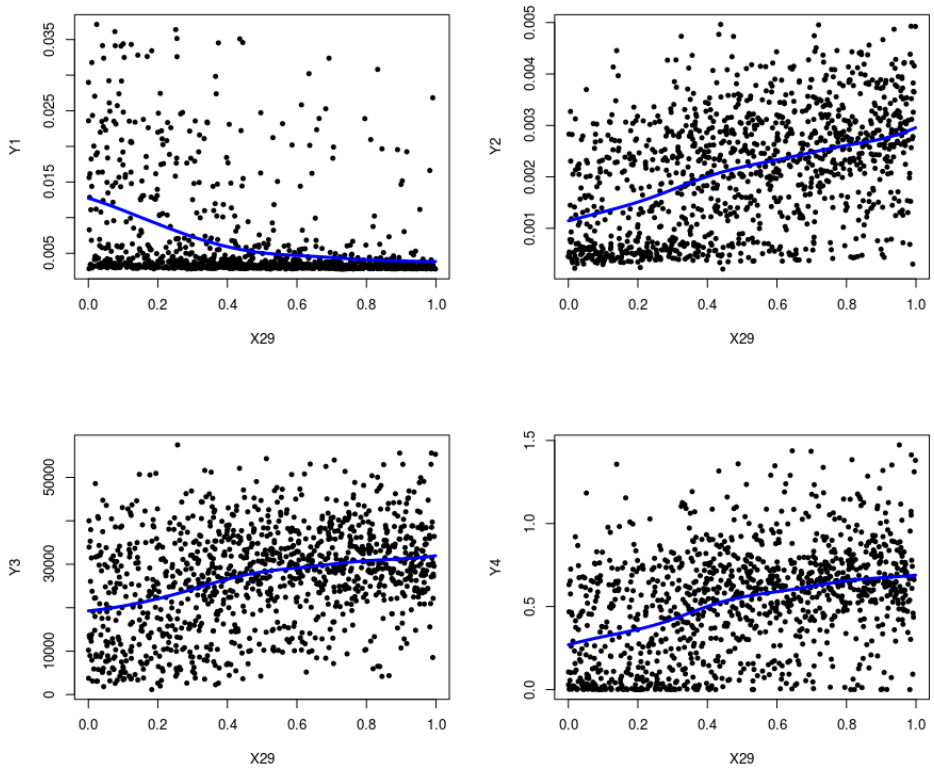

Figure 4.5: *Scatterplots with local polynomial regression (in blue) for the four outputs according to the input X_{29}, from the learning sample.*

Now let us examine Figure 4.6. As indicated above, the CFRAG coefficient (X_{13}) parameterizes the fragmentation of the drops. The higher it is, the more the drops are fragmented and therefore the smaller they are. This is what we observe on the second plot of the figure (Y_2 versus X_{13}). For the other plots, the previous physical analysis seems to apply. Thus, we see that the vapor mass is an increasing function of the fragmentation coefficient. The smaller the drops are, the more vapor there is. We also observe that the maximum impulse and the fragment mass reflecting the intensity of the vapor explosion phenomenon are decreasing functions of X_{13}.

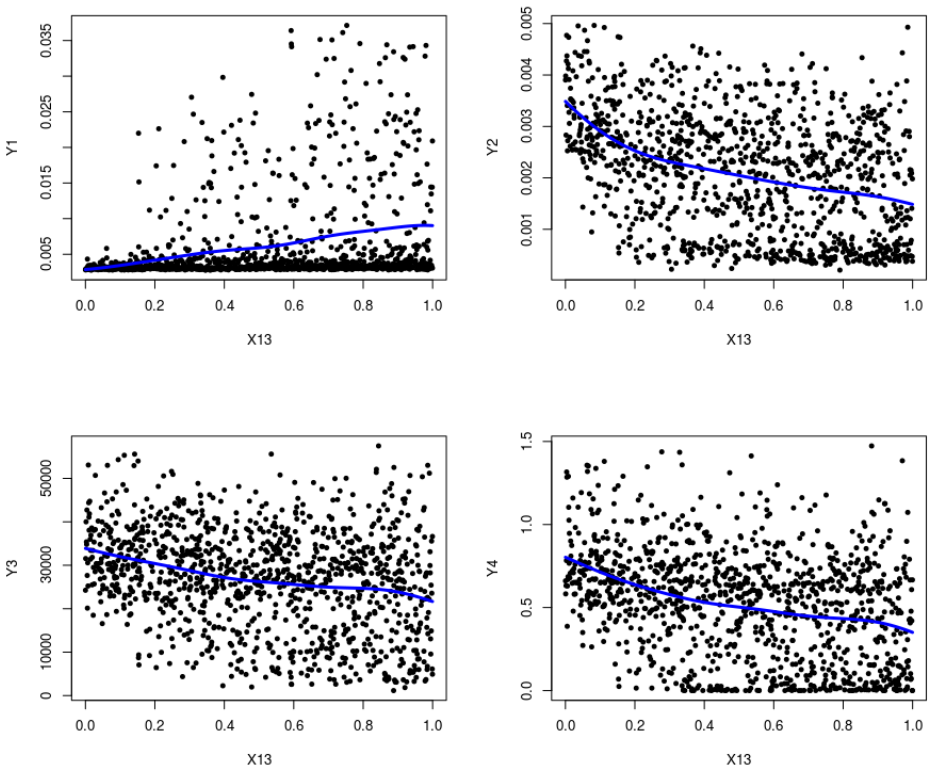

Figure 4.6: *Scatterplots with local polynomial regression (in blue) for the four outputs according to the input X_{13}, from the learning sample.*

Finally, Figure 4.7 depicts the scatterplots of the four outputs according to X_7 (KTDROP). The input X_7 corresponds to the solidification temperature of the drops. For $X_7 = 1$, the drops solidify directly after reaching the temperature $T_{liquidus}$ of the corium. For $X_7 = 0$, it must reach the temperature $T_{solidus}$ of the corium before solidifying. This means that it must have transferred all its latent heat ($3.10^5 J$) to the surrounding water before solidifying. We therefore understand that the closer X_7 is to 0, the more the drops will tend to stay liquid. Moreover, only liquid drops can be fragmented (and thus become smaller). This reasoning can explain what we observe on the second plot of Figure 4.7. Indeed, we observe that the diameter of the drops in the premixing step (Y_2) is a decreasing function of X_7. The analysis of the other graphs is similar to what we have seen before: the amount of vapor (Y_1) is an increasing function of X_7, and thus a decreasing function of the size of the drops. We also observe that the intensity of the explosion (through Y_3 and Y_4) is a decreasing function of X_7.

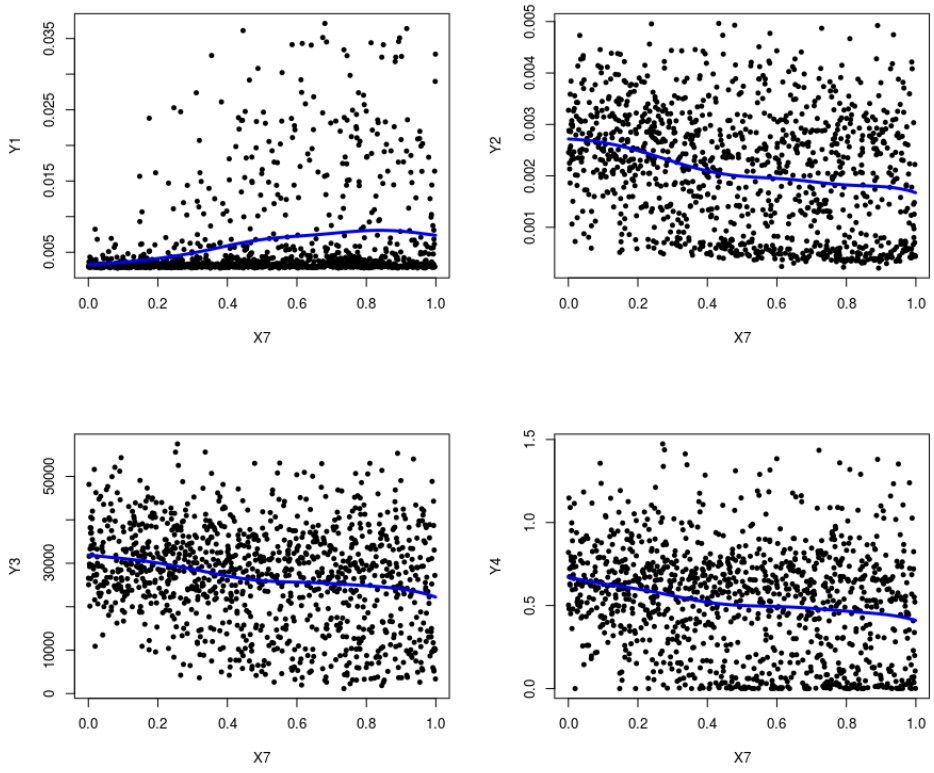

Figure 4.7: *Scatterplots with local polynomial regression (in blue) for the four outputs according to the input X_7, from the learning sample.*

4.4 Conclusion of the chapter and prospects

In this chapter, we performed the last steps of a modern and applicative approach for a quantitative sensitivity analysis of CPU-expensive simulation codes. This operational approach is well suited for complex industrial codes such as MC3D. A first screening step allowed the classification of the variables in two categories: the explanatory variables of the code and the others. This step is essential when the studied code has a large number of input parameters (several dozens for instance), as it is the case with MC3D. To perform this selection step, two advanced sensitivity analysis tools have been used: the HSIC and the first order Sobol' indices estimated with rank based statistics. These tools are well tailored to analyse non-linear model functions.

After this screening step, regression models have been built to approximate the code outputs. These metamodels only stand on the variable identified as explanatory by the screening step. Classic Gaussian process (GP) regression have been used to perform this metamodeling. As the choice of the kernel highly influences the behavior of GP models, several kernels have been tested. Among them, the 5/2-Matérn covariance was preferred. A complete analysis of prediction performances of these metamodels has been then carried out. This analysis showed that among the four studied outputs, an output is well fitted by GP metamodels, a second one is a little less well emulated and the GP metamodels are clearly inaccurate for the last two. Several hypothesis have been proposed to explain these results and some of them have been explored. Ways of improvements have been considered and are discussed further bellow.

Finally, we exploited the built metamodels to obtain a quantitative global sensitivity analysis (GSA) on the MC3D studied outputs. To process this GSA, the classic Sobol' first order and total indices have been estimated using the built metamodels. This second analysis allowed us to distinguish three inputs that are the most influential on the outputs of the code. All three are directly related to the fragmentation of the corium drops. Among the different models present in MC3D, this phenomenon seems to dominate in the modeling of the corium-water interaction. Scatterplots of the four outputs regarding these three inputs have been given. They allowed thus to understand more precisely the impact of these three inputs on the outputs.

The main developments foreseen in this work concern the improvement of the quality of the metamodeling. As mentioned at the end of Section 4.2, several possibilities can be envisaged to improve it. A first idea could be to *smartly* increase the size of the learning set to improve the GP learning, using the so-called *expected improvement* criterion for instance (Jones et al., 1998). However, the

significative probability of code failures could limit the interest of this approach for our application case. Relaxing some assumptions of GP models, such as the stationarity of GPs, is also a promising possibility. For instance, the use of Deep GP (Damianou, 2015) with a larger dataset and only taking into account the most influential inputs might provide interesting results. As the metamodeling are built in perspective and support of the calibration of the code, relaxing the GP interpolation constraints far from the values experimentally observed could be considered. It could lead to better predictivity results in the outputs ranges of interest. Recent approaches on relaxed Gaussian process (Petit et al., 2021) has shown promising results in this regard. Finally, considering completely different regression methods (SVR, neural networks) could also open new possibilities. However, the use of Gaussian processes offers advantages for calibration, in particular for Bayesian calbration since GP metamodels provides uncertainties in prediction. GP predictive distributions can be then included in Bayesian calibration framework (see for instance Carmassi (2018) or Damblin (2015) for details).

Another interesting perspective of this work is to better account for the interaction between inputs caused by code failures in sensitivity analysis. For example, the use of Generalized Sobol indices (Chastaing et al., 2014) is a promising approach for this task.

Conclusion and prospects

The industrial context of this PhD is the simulation of severe nuclear accidents in pressurized water reactors. More precisely, we focused on the modeling of the fuel-coolant interaction (FCI) that can lead to the steam explosion phenomenon. This explosion could damage the structures and compromise the integrity of the nuclear reactor. The calculation code used at the CEA to simulate this phenomenon is MC3D (MultiComponent Code 3D software). However, the methods proposed in this work are general enough to be applied to other simulation codes.

The FCI is a complex and non-linear phenomenon in which many mechanisms interact. This complexity induces a large number of model parameters to be managed and a relatively long computation time for each simulation with MC3D code. In this context, the main objective of this work is to investigate the simulation of the FCI by MC3D. In particular, we are interested in evaluating the impact of the uncertain modeling parameters on the outputs of the MC3D code in the general framework of uncertainty quantification. This study also aims to prepare the future calibration of these parameters on experimental data to improve the reliability of the code. To investigate the impact of uncertain modeling parameters, we proposed a methodology based on several advanced statistical methods. This methodology consists in five main steps:

1. Exploration of the input space using sampling methods.
2. Sensitivity analysis of code failures to understand which inputs have the most influence on them.
3. Selection of significant inputs and global sensitivity analysis of the code outputs of interest.
4. Emulation of the outputs by regression models (metamodeling).
5. Use of metamodels (built at Step 4) to perform a more complete sensitivity analysis.

In the first chapter of the thesis, we detailed the industrial context, the objectives of the thesis and the statistical methodology proposed to address them. We also specified the case study. In fact, we only focused on model parameters of the code,

which we considered as uncertain. The simulated conditions are those of a real experiment allowing the observation of the FCI, KROTOS KS4. We also focused only on a limited number of scalar outputs of the MC3D code. The other chapters of the thesis dealt with the different steps of our statistical methodology.

In Chapter 2, sampling methods were discussed. In the context of our work, choosing a sampling method is essential to perform the analysis of the code. A sampling method indeed determines the way in which the input space is explored and thus the properties of the input-output data set. In this chapter, an overview of the classical sampling methods was first given. In our application, we used the Latin hypercube sampling (LHS) because it has both good space-filling and theoretical convergence properties. An original work done on the convergence of Z-estimators under LHS was then presented. A reduction of the asymptotic estimation variance as well as a central limit theorem for this class of estimators under LHS were established. However, some restrictive regularity conditions were imposed on these estimators in order to obtain this convergence. Removing some of these limitations is an interesting prospect for future work.

After that, we discussed the convergence under LHS of other statistics used in our methodology. Some theoretical work about the Kolmogorov statistic was first presented. Numerical experiments on the convergence of a Sobol' first order indices rank based estimator have also been proposed. We empirically compared the asymptotic behavior of this estimator under pure Monte Carlo sampling, LHS and optimized LHS. According to this work, classical LHS appears to be a good compromise between bias and variance for rank based estimation of first order Sobol' indices. Finally, the convergence under LHS of two other statistics used in this work were discussed, namely the Hilbert Schmidt Independence Criterion (HSIC) and the hyperparameters of Gaussian Process (GP) regression models, the latter being estimated by maximum likelihood, which is a special case of Z-estimation. The theoretical study of their convergence is an interesting perspective.

The topic of Chapter 3 is the step 2 of our methodology, namely the analysis of code failures through the prism of sensitivity analysis. Code failures are common problems in the context of numerical simulation of complex physical phenomena. Detecting the inputs involved in these code failures and their implication is therefore helpful to improve the robustness of these simulation codes. Results obtained during the exploration of the input space of the MC3D code motivated this work. In fact, after defining our design of experiments, we ran the code on the sample points and observed that a third of the simulations did not converge. So, our aim was to investigate this issue.

Considering the occurrence of a code failure as a binary output in its own right, the principles of global sensitivity analysis can be used to identify the variables that have the greatest impact on the failures. Within this framework, we first introduced two methods to detect the inputs that might be involved in code failures. First, we studied the input distribution of the samples such as the code fails. It has been done using the Kolmogorov-Smirnov (K-S) goodness-of-fit test. This test method allows to detect the variables whose distribution conditional on code failures differs significantly from their initial distribution. Since our initial design of experiment is a LHS, the test procedure has been adapted for this frame. A second method, based on the Hilbert Schmidt Independence Criterion (HSIC), has been then proposed. This kernel-based method allows the detection of probabilistic dependencies between the input variables and code failures. Moreover, HSIC-based statistical tests of independence can be built for screening and ranking of inputs (by order of influence on code failures). From results of K-S and HSIC-based tests, a set of significantly influential inputs was selected. A graphical analysis of the densities conditional on code failures has been proceeded for these inputs. We then analyzed the obtained results from a physical point of view.

Then, a second sensitivity analysis was performed to detect and identify the interdependencies that appear between the inputs when considering the sample of only converged simulations. For this purpose, we reused the HSIC dependence measure, but this time between each pair of input variables conditioned on the occurrence of a code crash. Hence, three groups of interdependencies between the conditioned inputs have been detected with this method. A continuation of this work could be to distinguish the different causes of code failures in the analysis. The use of classification algorithms to identify the domains of input values leading to code crashes is also an interesting perspective. It would be indeed very useful in practice to predict code failures in order to avoid wasting computational resources.

Finally in Chapter 4, we discussed the three other steps of our global methodology: the screening, the metamodeling and the quantitative sensitivity analysis. Given the large number of inputs considered in our study, the selection (or screening) step is essential. It results in the classification of the input variables into two subgroups: the explanatory variables with respect to code outputs and the others. Given the limited number of available simulations, two advanced sensitivity analysis tools have been used to perform this preliminary selection of inputs: the HSIC and the first order Sobol' indices. Both tools are well tailored to analyze non-linear model functions and provide complementary information. HSIC (with associated independence test) is a screening method which allows to detect the non-significantly influential inputs. The first order Sobol' indices provide additional information on the proportion of variance explained by each of these variables individually and thus on the additive part of the input/output relationship.

Then, considering only the inputs selected as explicative, regression models (also called metamodels) were built in order to emulate the code outputs according to the inputs. Classical Gaussian process regression has been used to perform this metamodeling. Once trained on the set of available simulations, these metamodels can produce large amounts of predicted results with negligible computational effort. They thus allow an intensive exploration of the input variation domain. However, a validation of their predictive capacity, i.e. their capacity to correctly predict the results of the code with reliable predictive intervals, is necessary. This validation step revealed that some studied MC3D outputs were correctly reproduced while for others, only half of the output variance is explained by the metamodel. Despite these contrasting results, a complete, easily interpretable and quantitative sensitivity analysis of the outputs has been carried out using the metamodels. More precisely, the classic Sobol' first order and total indices have been estimated with the metamodels. As a result, three inputs were identified as the main and most influential on the code's outputs. All three are directly related to a specific phenomenon of the FCI, the fragmentation of the corium drops in water. In addition, scatterplots enabled a more exhaustive physical understanding of the impact of these three inputs on the outputs.

To summarize, we proposed in this work a modern and applicative approach to perform a quantitative sensitivity analysis on complex industrial codes (such as MC3D), in support of a thorough physical interpretation of the effects of uncertain inputs.

The perspectives regarding our methodology are numerous. The most prominent one is the calibration of the code on the KROTOS experiments. Our work allowed the identification of a restrained number of inputs that seem to have a significant impact on the global behavior. The next step is naturally to adjust the code, in a robust way, to the experimental data at our disposal. The major challenge concerning this perspective concerns the quantity of data on which we have to perform this calibration. In fact, little experimental data is available because KROTOS experiments are very expensive. To achieve this calibration, Bayesian methods are promising. They have indeed several advantages. First of all, the bayesian framework will allow us to consider the information we have a priori about the parameters and their uncertainty. This information is essential in our case given the small amount of experimental data available. Additionally, in this frame, the calibrated parameters are associated with a probability law that quantifies their post-calibration uncertainties.

However, many code runs are necessary to perform this Bayesian calibration. The use of metamodels thus appears to be essential given the high computing time of the MC3D code. In particular, GP regressions provide a good theoretical framework for the Bayesian calibration (see for instance (Carmassi, 2018) or (Damblin, 2015) for details). However, the currently built metamodels are not good enough for some outputs, to allow an accurate calibration of the code on experimental data. Thus, a preliminary step to the calibration is the improvement of the metamodels.

Increasing the learning base using *Expected Improvement* techniques (Jones et al., 1998) is a promising approach to do so, but its application in high dimension remains complicated. The results of sensitivity analysis might be used to reduce the search space. Moreover, the significative probability of MC3D code failure must also be taken into account. Another solution to improve the accuracy of GP metamodel could lies in relaxing the assumption of stationarity. The use of Deep GP (Damianou, 2015) with a larger dataset and focusing on the main inputs of the code for instance could be an interesting option. Another possibility is to relax the GP interpolation constraints far from the region of interest regarding the calibration, using relaxed Gaussian process (Petit et al., 2021) for example. Finally, running new calculations on a smaller input space may also be interesting. This can be done in two ways. First, one can reduce the number of inputs by taking only those that influence the FCI. Our sensitivity analysis will thus be very useful. The variation spaces of the considered inputs can also be reduced to better focus on obtained experimental values. Reducing the size of the input space could indeed improve the metamodeling in the areas of interest for calibration.

Bibliography

Abramowitz, M., and Stegun, I. A. (1964). *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Dover.

Achour, M. H. (2017). *Fragmentation de métal liquide dans l'eau* (PhD thesis). University of Lorraine.

Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American Mathematical Society 68*, 337–404.

Bachoc, F. (2013). Cross validation and maximum likelihood estimations of hyper-parameters of gaussian processes with model misspecification. *Computational Statistics & Data Analysis 66*, 55–69.

Basak, S., Petit, S. J., Bect, J., and Vazquez, E. (2021). Numerical issues in maximum likelihood parameter estimation for gaussian process regression. In *7th international conference on machine learning, optimization and data science*.

Berthoud, G. (1996). *Proposition d'amélioration de la "loi de Pilch" pour la fragmentation hydrodynamique lors de la phase de prémélange dans une explosion de vapeur.* (Technical note No. STI/LEF/96-143). Grenoble: CEA.

Berthoud, G., and Gros D'Aillon, L. (2009). Film boiling heat transfer around a very high temperature thin wire immersed into water at pressure from 1 to 210bar: Experimental results and analysis. *International Journal of Thermal Sciences 48*, 1728–1740.

Berthoud, G., and Valette, M. (1994). Development of a multidimensional model for the premixing phase of a fuel-coolant interaction. *Nuclear Engineering and Design 149*, 409–418.

Board, S. J., Hall, R. W., and Hall, R. S. (1975). Detonation of fuel coolant explosions. *Nature 5498*, 319–321.

Bouyer, V., Cassiaut-Louis, N., Fouquart, P., and Piluso, P. (2015). Plinius prototypic corium experimental platform. In *Proceedings of Nureth-16*, pages 5327–5340.

Brayer, C., and Berthoud, G. (1997). First Vapor Explosion Calculations Performed With MC3D Thermal-Hydraulic Code. In *Proceedings of the OECD/CSNI specialist meeting on fuel-coolant interactions*, pages 391–409.

Brayer, C., Charton, A., and Grishchenko, et al. (2012). Analysis of the KROTOS KFC test by coupling x-ray image analysis and MC3D calculations, june 24. In *Proceedings of ICAPP '12*, pages 1854–1863. American Nuclear Society (ANS).

Brayer, C., Le Monnier, A., and Chikhi, N. (2020). Impact of corium thermophysical properties on fuel-coolant interaction. *Annals of Nuclear Energy 147*, 107613.

Cannamela, C. (2007). *Apport des méthodes probabilistes dans la simulation du comportement sous irradiation du combustible à particules* (PhD thesis). University Paris 7.

Carmassi, M. (2018). *Uncertainty quantification and calibration of a photovoltaic plant model : Warranty of performance and robust estimation of the long-term production.* (PhD thesis). University Paris-Saclay.

Chastaing, G., Gamboa, F., and Prieur, C. (2014). Generalized sobol sensitivity indices for dependent variables: Numerical methods. *Journal of Statistical Computation and Simulation 85*, *85*, 1306–1333.

Chatterjee, S. (2020). A new coefficient of correlation. *Journal of the American Statistical Association 116*, 1–21.

Chi, H., Beerli, P., Evans, D. W., and Mascagni, M. (2005). On the scrambled soboĺ sequence. In *Computational science – ICCS 2005*, pages 775–782. Springer Berlin Heidelberg.

Ciccarelli, G., and Frost, D. L. (1988). Dynamics of explosive interactions between multiple drops of tin and water. *American Institute of Aeronautics and Astronautics*, *114*, 451–473,.

Corradini, M. L., Kim, B. J., and Oh, M. D. (1988). Vapor explosions in light water reactors: A review of theory and modeling. *Progress in Nuclear Energy 22*, 1–117.

Cortes, C., and Vapnik, V. (1995). Support-vector networks. *Machine Learning 20*, 273–297.

Cramér, H. (1928). On the composition of elementary errors. *Scandinavian Actuarial Journal 1*, 13–74.

Csiszár, I. (1972). A class of measures of informativity of observation channels. *Periodica Mathematica Hungarica 2*, 191–213.

Da Veiga, S. (2015). Global sensitivity analysis with dependence measures. *Journal of Statistical Computation and Simulation 85*, 1283–1305.

Da Veiga, S., Gamboa, F., Iooss, B., Prieur, C., Industrial, S. for, and Mathematics, A. (2021). *Basics and trends in sensitivity analysis: Theory and practice in R.* Society for Industrial; Applied Mathematics.

Dacunha-Castelle, D., and Duflo, M. (1986). *Probability and statistics volume II.* Springer-Verlag.

Damblin, G. (2015). *Contributions statistiques au calage et à la validation des codes de calcul* (PhD thesis). Université Paris Saclay.

Damblin, G., Couplet, M., and Iooss, B. (2013). Numerical studies of space filling designs: Optimization of Latin Hypercube Samples and subprojection properties. *Journal of Simulation 7*.

Damianou, A. (2015). *Deep gaussian processes and variational propagation of uncertainty* (PhD thesis,). University of Sheffield.

De Lozzo, M., and Marrel, A. (2017). Sensitivity analysis with dependence and variance-based measures for spatio-temporal numerical simulators 31. *Stoch Environ Res Risk Assess*, 1437–1453.

Demay, C., Iooss, B., Gratiet, L. L., and Marrel, A. (2021). Model selection based on validation criteria for gaussian process regression: An application with highlights on the predictive variance 38. *Quality and Reliability Engineering International*, 1482–1500.

Devroye, L. (1986). *Non-uniform random variate generation. Journal of the American Statistical Association*, page 28.

Dullforce, T. A., Buchanan, D. J., and Peckover, R. S. (1976). Self-triggering of small-scale fuel-coolant interactions: I. Experiments. *Journal of Physics D: Applied Physics 9*, (9), 1295.

Efron, B. et S., and C. (1981). The jackknife estimate of variance. *The Annals of Statistics*, *9*(3), 586–596.

El Amri, M. R., and Marrel, A. (2021, October). *More powerful HSIC-based independence tests, extension to space-filling designs and functional data*. preprint, available at https://hal-cea.archives-ouvertes.fr/cea-03406956.

Epstein, M., and Hauser, G. M. (1980). Subcooled forced-convection film boiling in the forward stagnation region of a sphere or cylinder. *International Journal of Heat and Mass Transfer 23*, 179–189.

Fang, K.-T., Li, R., and Sudjianto, A. (2005). *Design and modeling for computer experiments (computer science & data analysis)*. Chapman & Hall/CRC.

Fisher, R. A. (1922). On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London, Series A 222*, 309–368.

Fletcher, D. F. (1995). Steam explosion triggering: A review of theoretical and experimental investigations. *Nuclear Engineering and Design 1*, 27–36.

Fukumizu, K., Gretton, A., Sun, X., and Schölkopf, B. (2008). Kernel measures of conditional dependence. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in neural information processing systems*,Vol. 20, pages 489–496. Red Hook, NY, USA: Curran Associates, Inc.

Gamboa, F., Gremaud, P., Klein, T., and Lagnoux, A. (2022). Global sensitivity analysis: A novel generation of mighty estimators based on rank statistics. *Bernoulli 28*, 2345–2374,.

Gamboa, F., Janon, A., Klein, T., Lagnoux, A., and Prieur, C. (2016). Statistical inference for sobol pick-freeze monte carlo method. *Statistics 50*, 881–902.

Gratiet, L. L., Cannamela, C., and Iooss, B. (2014). A bayesian approach for global sensitivity analysis of (multifidelity) computer codes. *SIAM/ASA Journal on Uncertainty Quantification 2*, 336–363.

Gretton, A., Bousquet, O., Smola, A., and Schölkopf, B. (2005). Measuring Statistical Dependence with Hilbert-Schmidt Norms. *Algorithmic Learning Theory*, 63–77.

Gretton, A., Fukumizu, K., Teo, C., Song, L., Schölkopf, B., and Smola, A. (2008). A kernel statistical test of independence. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in neural information processing systems*,Vol. 20, pages 585–592. Curran Associates, Inc.

Grishchenko, D., Piluso, P., Fouquart, P., Cassiaut-Louis, N., Bullado, Y., compagnon, F., ... Payan, E. (2011). *KROTOS KS-4 test data report* (Technical note No. DEN/DTN/STRI/LMA/2011/006 OECD/SERENA-2011-TR08).

Hakimi, F., Brayer, C., Marrel, A., Gamboa, F., and Habert, B. (2022). *Statistical methods for the study of computer experiments failures: Application to a fuel-coolant interaction simulation code*. unpublished.

Halton, J. (1964). Algorithm 247: Radical-inverse quasi-random point sequence. *Communications of the ACM 7*, 701–701.

Hansson, R. C. (2007). *An experimental study on the dynamics of melt-water micro-interactions in a Vapor explosion* (PhD thesis). KTH University, Stockholm, Sweden.

Harlow, F. H., and Amsden, A. (1975). Numerical calculation of multiphase fluid flow. *Journal of Computational Physics 17*, 19–52.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer.

Helmholtz, H. (1868). On discontinuous movements of fluids. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 36*, 337–346.

Helton, J. C. (1997). Uncertainty and sensitivity analysis in the presence of stochastic and subjective uncertainty. *Journal of Statistical Computation and Simulation 57*, 3–76.

Hoeffding, W. (1948). A class of statistics with asymptotically normal distribution. *The Annals of Mathematical Statistics 19*, 293–325.

Hoerl, A. E., and Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics 12*, 55–67.

Homma, T., and Saltelli, A. (1996). Importance measures in global sensitivity analysis of non linear models. *Reliability Engineering and System Safety 52*, 1–17.

Huhtiniemi, I., and Magallon, D. (2001). Insight into steam explosions with corium melts in KROTOS. *Nuclear Engineering and Design 204*, (1), 391–400.

Huhtiniemi, I., Magallon, D., and Hohmann, H. (1999). Results of recent KROTOS

FCI tests: Alumina versus corium melts. *Nuclear Engineering and Design 189*, 379–389.

International Energy Agency. (2021). *Key world energy statistics*. (IEA, Ed.).

Iooss, B., and Lemaître, P. (2015). A review on global sensitivity analysis methods. In *Uncertainty management in simulation-optimization of complex systems*, pages 101–122. Springer US.

Jacod, J., and Protter, P. (2004). *Probability essentials*. Springer Berlin Heidelberg.

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions 13. *Journal of Global Optimization*, 455–492.

Kelvin, L. (1871). XLVI. Hydrokinetic solutions and observations. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 42*, 362–377.

Loh, W. L. (1996). On latin hypercube sampling. *Annals of Statistics 24*, 2058–2080.

Malmazet, E. de. (2009). *Etude de la fragmentation de gouttes chaudes en ébullition en film dans un écoulement d'eau* (PhD thesis). Grenoble Institute of Technology.

Marrel, A., and Chabridon, V. (2021). Statistical developments for target and conditional sensitivity analysis: Application on safety studies for nuclear reactor. *Reliability Engineering & System Safety 214*, 107711.

Marrel, A., Iooss, B., and Chabridon, V. (2022). The ICSCREAM methodology: Identification of penalizing configurations in computer experiments using screening and metamodel—applications in thermal hydraulics. *Nuclear Science and Engineering*, *196*, 301–321.

Marrel, A., Iooss, B., Dorpe, F., and Volkova, E. (2008). An efficient methodology for modeling complex computer codes with gaussian processes. *Computational Statistics and Data Analysis*, *52*, 4731.

Marrel, A., Iooss, B., Laurent, B., and Roustant, O. (2009). Calculations of sobol indices for the gaussian process metamodel. *Reliability Engineering & System Safety 94*, 742–751.

Mckay, M., Beckman, R., and Conover, W. (1979). A comparison of three methods for selecting vales of input variables in the analysis of output from a computer code. *Technometrics*, 239–245.

Meignen, R., and Magallon, D. (2005). Comparative review of FCI computer models used in the OECD-SERENA program. In *ICAAP*.

Meignen, R., Picchi, S., Lamome, J., Escobar, B. R. S. C., and Nicaise, G. (2014). The challenge of modeling fuel–coolant interaction: Part I – Premixing. *Nuclear Engineering and Design 280*, 511–527.

Meignen, R., Raverdy, B., Picchi, S., and Lamome, J. (2014). The challenge of modeling fuel–coolant interaction: Part II – Steam explosion. *Nuclear Engineering and Design 280*, 528–541.

Mercier, P. (1989). *Méthode numérique et développement du modèle TRIO-MC : Cas d'un problème adiabatique à deux champs de vitesses* (Technical note No. STT/LPML/89/O9/C).

Morris, M. D. (1991). Factorial Sampling Plans for Preliminary Computational Experiments. *Technometrics 33*, 161–174.

Nelder, J., and Wedderburn, R. (1972). Generalized linear models. *Journal of the Royal Statistical Society. Series A (General). Blackwell Publishing 135*, 370–384.

Nelson, L. S., and Duda, P. M. (1985). *Steam explosion experiments with single drops of iron oxide melted with a CO2 laser. Part II: Parametric studies.* (No. NUREG/CR-2718 SAND82-1105 R3), page 147.

Owen, A. B. (1992). A Central Limit Theorem for Latin Hypercube Sampling. *Journal of the Royal Statistical Society 54: Series B (Methodological)*, (2), 541–551.

Parzen, E. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics 33*, (3), 1065–1076.

Pearson, K. (1895). Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London 58*, *58*.

Petit, S. J., Bect, J., Feliot, P., and Vazquez, E. (2021). Model parameters in gaussian process interpolation: An empirical study of selection criteria. *Submitted to the SIAM/ASA Journal on Uncertainty Quantification*.

Pilch, M. (1981). *Acceleration induced fragmentation of liquid drops* (PhD thesis). University of Virginia.

Pujol, G. (2009). Simplex-based screening designs for estimating metamodels. *Reliability Engineering & System Safety 94*, 1156–1160.

Rasmussen, C. E., and Williams, C. K. I. (2005). *Gaussian processes for machine learning (adaptive computation and machine learning)*. The MIT Press.

Rayleigh, L. (1883). Investigation of the Character of the Equilibrium of an Incompressible Heavy Fluid of Variable Density. In *Proceedings of the london mathematical society*, pages 170–177.

Rocquigny, E. de. (2008). *Uncertainty in industrial practice : A guide to quantitative uncertainty management*. (J. W. & Sons, Ed.).

Rosenblatt, F. (1962). *Principles of neurodynamics : Perceptrons and the theory of brain mechanisms*. Report (Cornell Aeronautical Laboratory), Spartan Books.

Rosenblatt, M. (1956). Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics 27*, 832–837.

Rousseeuw, P. J., Hampel, F. R., Ronchetti, E. M., and Stahel, W. A. (1986). *Robust statistics: The approach based on influence functions*. Wiley.

Roustant, O., Padonou, E., Deville, Y., Clément, A., Perrin, G., Giorla, J., and Wynn, H. (2020). Group kernels for gaussian process metamodels with categorical inputs. *SIAM/ASA Journal on Uncertainty Quantification 8*, 775–806.

Saltelli, A. (2002). Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications 145*, 280–297.

Saltelli, A. (Ed.). (2008). *Global sensitivity analysis: The primer*. Chichester, England ; Hoboken, NJ: John Wiley.

Saltelli, A., Tarantola, S., Campolongo, F., and Ratto, M. (2004). *Sensitivity analysis in practice: A guide to assessing scientific models*. USA: Halsted Press.

Santner, T. J., Williams, B. J., Notz, W. I., and Williams, B. J. (2003). *The design and analysis of computer experiments. 1*. Springer.

Scott, E. (1978). *Contribution à l'étude de l'interaction thermique: Modélisation d'une détonation thermique en milieu multiphase* (PhD thesis). Grenoble Institute of Technology, Grenoble.

Shannon, C. E. (1948). A mathematical theory of communication 27. *The Bell System Technical Journal*, 379–423.

Shiryayev, A. N. (1992). 15. On the empirical determination of a distribution law. In *Selected works of a. N. Kolmogorov: Volume II probability theory and mathematical statistics*, pages 139–146. Springer Netherlands.

Smirnov, N. (1948). Table for Estimating the Goodness of Fit of Empirical Distributions. *The Annals of Mathematical Statistics 19*, (2), 279–281.

Sobol', M. (1967). Distribution of points in a cube and approximate evaluation of integrals". *U.S.S.R Comput. Maths. Math. Phys 7*, 86–112.

Sobol', M. (1993). Sensitivity Estimates for Nonlinear Mathematical Models (english translation). *Mathematical Modeling and Computational Experiment 4*, 407–414.

Sriperumbudur, B., Fukumizu, K., and Lanckriet, G. (2010). On the relation between universality, characteristic kernels and RKHS embedding of measures. In Y. W. Teh and M. Titterington, editors, *Proceedings of the thirteenth international inproceedings on artificial intelligence and statistics 9*, pages 773–780. Proceedings of Machine Learning Research.

Stein, M. (1987). Large Sample Properties of Simulations Using Latin Hypercube Sampling. *Technometrics 29*, (2), 143–151.

Thiémard, E. (2000). *Sur le calcul et la majoration de la discrépance à l'origine* (PhD thesis). University EPFL.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society 58: Series B*, 267–288.

Trevor Hastie, H. Z. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67*, 301–320.

Trucano, T. G., Swiler, L. P., Igusa, T., Oberkampf, W. L., and Pilch, M. (2006). Calibration, validation, and sensitivity analysis: What's what. *Reliability Engineering & System Safety 91*, 1331–1357.

Van der Vaart, A. W. (1998). *Asymptotic statistics*. Cambridge University Press.

Villani, C. (2003). *Topics in Optimal Transportation*. American Mathematical Society.
Wand, M. P., and Jones, M. C. (1994). *Kernel smoothing*. Chapman; Hall/CRC.

Appendix A

Complementary results on the first test of Subsection 2.3.2

In this appendix, we display the results of the first test performed to investigate the convergence of the Sobol' first order indice rank estimator $\widehat{S_j^n}^{RS}$ under different sampling methods. In the main document (Section 2.3.2), we shown the obtained results for the Sobol' function. The results for the Ishigami function (Saltelli, 2008) and the Morris (Morris, 1991) function are displayed here.

The classic Ishigami function (Saltelli, 2008) is defined as follow:

$$f_I(X_1, X_2, X_3) = \sin(X_1) + b\sin^2(X_2) + bX_3^4\sin(X_1)$$

with:

- X_j $j \in [1, 2, 3]$ following uniform distributions in $[-\pi, \pi]$;
- $a = 7$ and $b = 0.1$.

The theoretical values of the first Sobol' indices for the classic Ishigami function are $\boldsymbol{S} = [0.305, 0.4356, 0]$.

The Morris function (Morris, 1991) is defined as follow:

$$f_M(X_1, \ldots, X_{20}) = \beta_0 + \sum_{j=1}^{20}\beta_j w_j + \sum_{j<j'}^{20}\beta_{jj'}w_j w_{j'} + \sum_{j<j'<j''}^{20}\beta_{jj'j''}w_j w_{j'} w_{j''} + \sum_{j<j'<j''<j'''}^{20}\beta_{jj'j''j'''}w_j w_{j'} w_{j''} w_{j'''}$$

with:

- $w_j = 2(X_i - 1/2)$ except for $i = 3, 5$ and 7, where $w_j = 2(1.1X_i/(X_i + 0.1) - 1/2)$;
- the first order coefficients $\beta_j = 20$ for $j \in [\![1, 10]\!]$;
- the second order coefficients $\beta_{j,j''} = -15$ for $j, j' \in [\![1, 6]\!]$;
- the third order coefficients $\beta_{j,j',j''} = -10$ for $j, j', j'' \in [\![1, 5]\!]$;
- the fourth order coefficients $\beta_{j,j',j'',j'''} = 5$ for $j, j', j'', j''' \in [\![1, 4]\!]$;
- $X_j, j \in [\![1, 20]\!]$ following uniform distributions in $[0, 1]$.

The remainder of the first and second order coefficients are generated independently from a normal distribution $\mathcal{N}(0, 1)$. The remainder third and fourth order coefficients are set to 0.

The theoretical values of the first Sobol' indices for the Morris function are equal to:

$\boldsymbol{S} = [0.012, 0.008, 0.019, 0.015, 0.021, 0.005, 0.07, 0.138, 0.138,$

$0.129, 0.007, 0.007, 0.007, 0.007, 0.008, 0.009, 0.007, 0.009, 0.008, 0.008]$.

Figures A.1 and A.2 show the evolution of the bias^2, the variance and the mean square error (MSE) of the Sobol' index estimator of the inputs of the Ishigami function and the Morris function respectively. The estimation of these three criteria are done with $N_{tests} = 500$ for each value of n. For the Morris function, we only display here the errors of the inputs that have the maximum (input 9), the median (input 15) and the minimum (input 6) true S_1 values for concision.

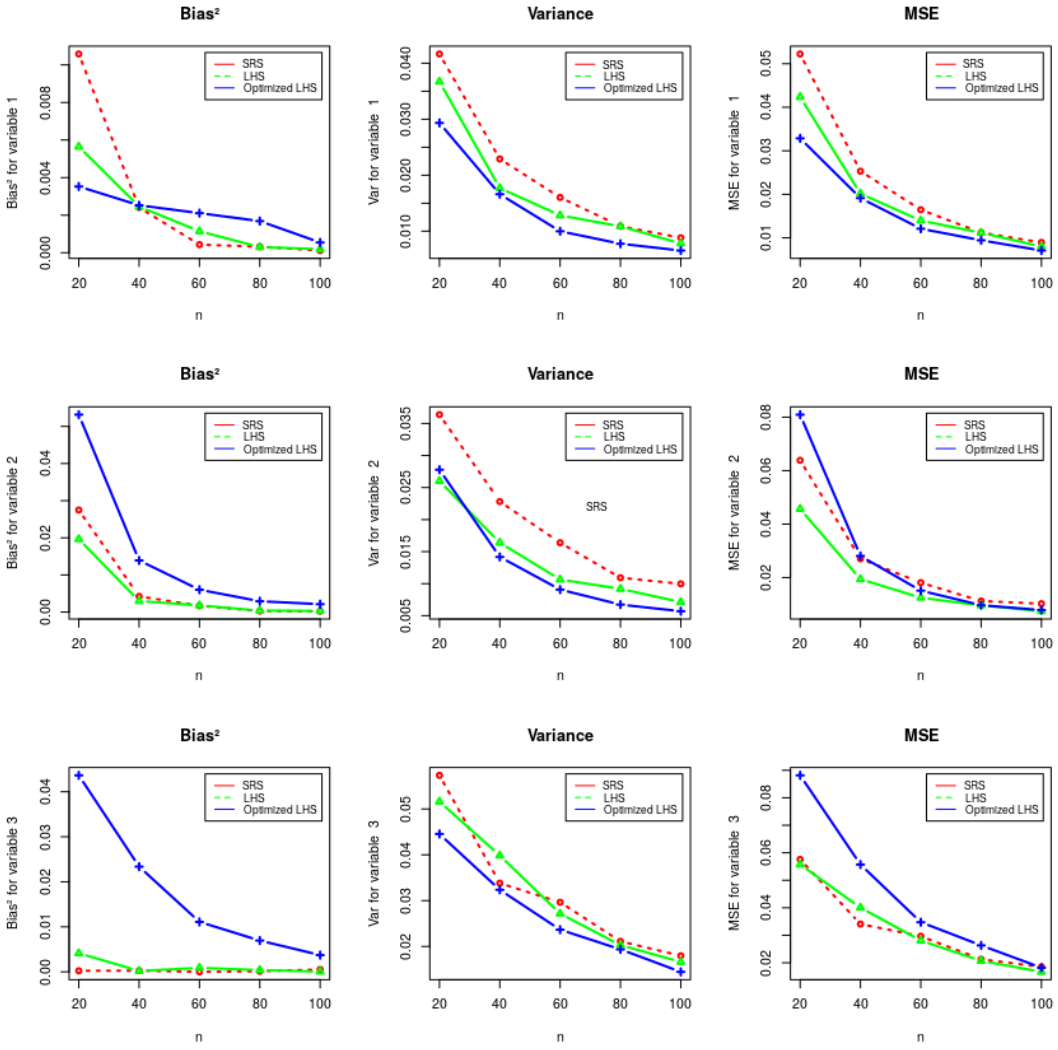

Figure A.1: *Ishigami model function: square bias, variances and MSE of the estimators of 1rst Sobol' indices, according to the numerical experiment design for the 3 inputs, according to the design experiment type.*

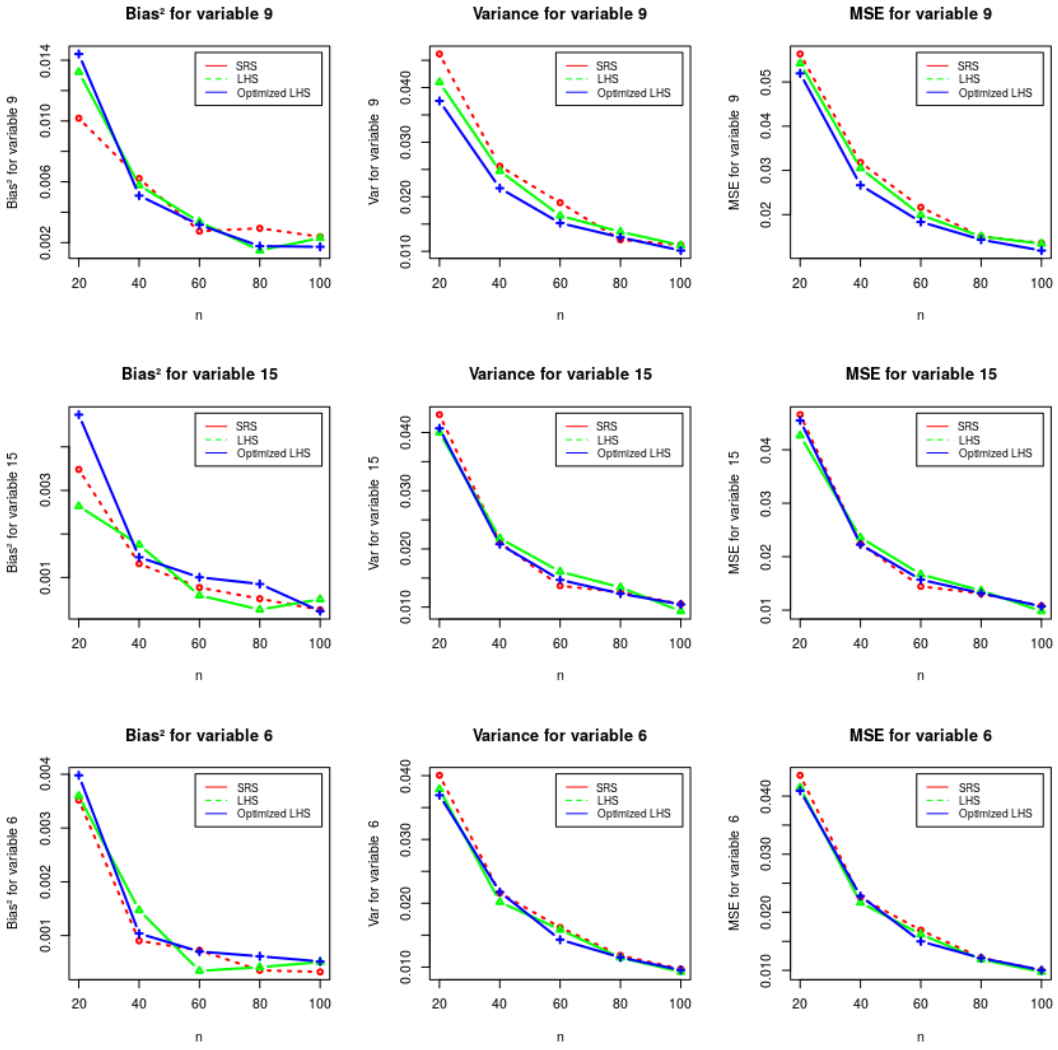

Figure A.2: *Morris model function: square bias, variances and MSE of the estimators of 1rst Sobol' indices, according to the numerical experiment design for 3 inputs (those with max, median and min indices values), according to the design experiment type.*

Appendix B

Synthèse du manuscrit en français

Cette synthèse en français est composée d'un résumé complet du Chapitre 1 (Contexte, objectifs et méthodologie) et d'une traduction de la conclusion.

B.1 Contexte, objectif et méthodologie

En France, environ 70 % de l'électricité consommée annuellement est produite par des réacteurs nucléaires (International Energy Agency, 2021) (56 réacteurs en fonctionnement). La construction de six nouveaux réacteurs a également été annoncée pour les années à venir. La recherche dans ce domaine est donc d'une importance majeure pour la production d'électricité en France. Une part significative de ce travail de recherche est assurée par le CEA (Commissariat à l'Énergie Atomique et aux énergies alternatives).

La quasi-totalité des réacteurs existants utilisent l'énergie libérée par la réaction de fission d'un atome lourd d'uranium 235 par un neutron. Cette réaction est fortement exothermique et produit de nouveaux neutrons, conduisant dans certaines conditions à une réaction en chaîne. L'énergie produite est transformée en chaleur par un liquide de refroidissement. En France, le caloporteur est de l'eau maintenue sous pression (155 bars) et ayant une température pouvant atteindre $350°C$. On parle ainsi de Réacteurs à Eau Pressurisée (REP). Cette eau chaude circule dans le cœur du réacteur dans un circuit fermé, appelé circuit primaire, et échange de l'énergie thermique avec l'eau d'un autre circuit, appelé circuit secondaire. Ce deuxième circuit génère de la vapeur qui est utilisée pour actionner des turbines reliées à des générateurs qui produisent de l'électricité. Enfin, un circuit de refroidissement permet de refroidir et condenser la vapeur après son passage dans les turbines. La Figure B.1 illustre le fonctionnement d'un réacteur à eau pressurisée ainsi décrit.

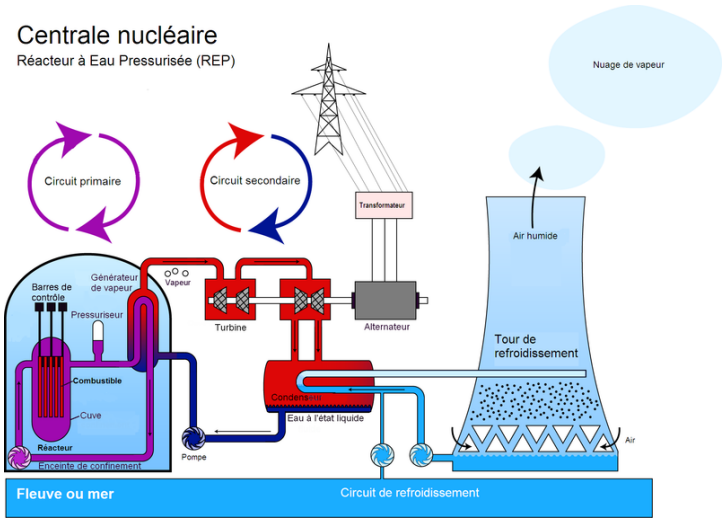

Figure B.1: *Schéma de principe du fonctionnement d'un réacteur nucléaire à eau pressurisée. Tiré de wikipedia : Réacteur à eau pressurisée*

De nombreux systèmes de sécurité ont été développés et mis en œuvre pour permettre le bon fonctionnement de ces réacteurs nucléaires et ainsi éviter les situations accidentelles ou limiter leurs effets. Ainsi, en plus des composants de base mentionnés ci-dessus, les réacteurs nucléaires sont équipés de plusieurs systèmes de sécurité indépendants et redondants. Ces systèmes opèrent en parallèle pour atténuer les risques liés aux anomalies de fonctionnement ou aux accidents d'exploitation. Ils empêchent également ces situations accidentelles de dégénérer.

Cependant, l'efficacité de tout système de sécurité dépend de l'accident pour lequel il est conçu. Des séquences accidentelles sortant du cadre des dispositifs de sécurité existants peuvent se produire. En effet, plusieurs évènements de ce genre ont eu lieu dans l'histoire. On peut notamment citer Three Mile Island (USA, 1979), Tchernobyl (Ukraine, 1986) et plus récemment Fukushima Daiichi (Japon, 2011). Ces événements ont souligné la nécessité d'adopter des mesures pour atténuer l'occurrence de scénarios pouvant conduire à ce que l'on appelle des *accidents graves*.

Un accident est considéré comme *grave* lorsque l'énergie libérée par le cœur du réacteur est supérieure à l'énergie dissipée par le fluide de refroidissement, et ce malgré tous les systèmes de sécurité. Dans ce cas, la température du cœur augmente et peut atteindre sa température de fusion, entraînant la perte de l'intégrité du cœur. Le magma chaud (environ 3000 K) composé d'éléments fondus du cœur et des structures ainsi formé est appelé par le nom générique *corium*. Ce magma

peut se propager hors du cœur et former un bain au fond de la cuve du réacteur. En raison de sa température élevée, le corium ainsi accumulé peut ablater la paroi de la cuve. Si celle-ci est percée, le corium peut alors se propager dans le reste du réacteur, atteindre le radier en béton et interagir avec celui-ci. Le corium peut également interagir avec l'eau environnante étant utilisée pour refroidir le réacteur. C'est ce qu'on appelle l'Interaction Corium-Eau (ICE). La figure B.2 illustre les différentes étapes de la progression du corium dans le réacteur.

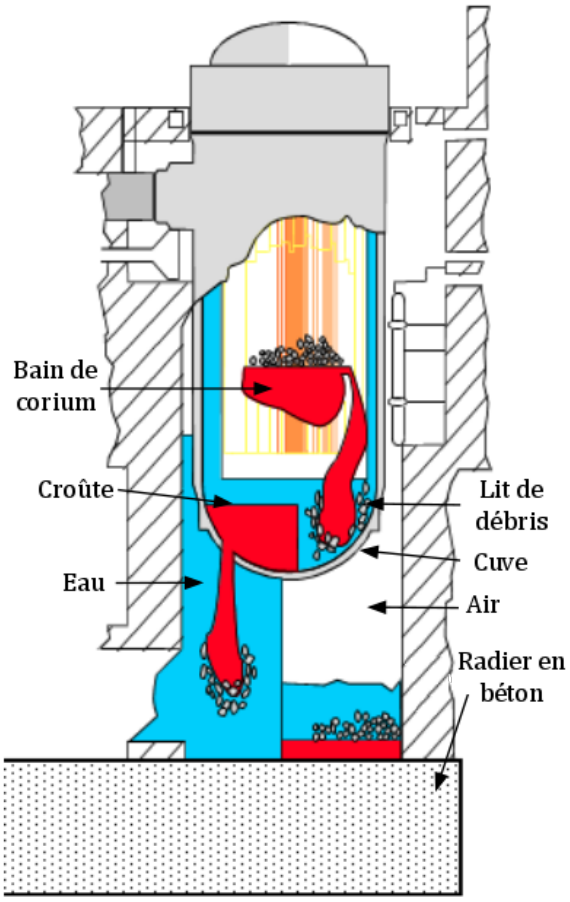

Figure B.2: *Progression du corium lors d'un accident nucléaire grave dans un réacteur à eau pressurisée.*

Cette thèse s'inscrit dans le contexte industriel de l'étude de l'ICE. Ce chapitre vise ainsi à présenter ce contexte, ainsi que la problématique de travail relative à celui-ci. Nous présenterons également la méthodologie, basée sur des outils statistiques, qui a été proposée pour y répondre.

B.1.1 Contexte industriel : étude de l'Interaction Corium Eau (ICE)

Comme pour la majorité des phénomènes physiques, deux types d'outils sont disponibles pour étudier l'interaction corium eau. D'une part, il y a les installations expérimentales. Elles permettent une observation directe du phénomène et de ses conséquences. Cependant, ces installations ne sont généralement pas entièrement représentatives du cas réacteur. En effet, elles sont souvent plus petites qu'un réacteur nucléaire réel et les matériaux utilisés expérimentalement ne sont pas les matériaux réels. Les expériences effectuées sur ces installations peuvent également être très coûteuses : jusqu'à plusieurs millions d'euros pour une expérience simulant l'interaction corium-eau avec des matériaux prototypiques (même composition chimique que le corium réel mais avec de l'uranium appauvri). Enfin, le type et la quantité de données expérimentales qui peuvent être recueillies sont limités. D'autre part, nous avons à notre disposition des logiciels de simulation pour modéliser et prédire les phénomènes physiques comme l'ICE. Le faible coût d'exploitation de ces outils de simulations comparativement aux expériences réelles permet de traiter beaucoup plus de cas différents. Cependant, il est important de s'assurer que la modélisation du phénomène étudié par ces codes de calcul est fidèle à la réalité.

L'objectif de cette section est de décrire l'interaction corium-eau. L'installation expérimentale (KROTOS) et l'outil numérique (MC3D : Multicomponent Code 3D) utilisés pour analyser le phénomène au CEA seront également présentés.

B.1.1.1 Description du phénomène

Comme indiqué précédemment, l'ICE dans le contexte d'un accident grave (en REP) correspond au contact entre le corium chaud et l'eau de refroidissement environnante. Cette interaction peut conduire, sous certaines conditions, à une fragmentation fine du corium entraînant une vaporisation violente du liquide de refroidissement et la propagation d'une onde de pression. Ce phénomène, appelé explosion de vapeur, peut menacer l'intégrité du réacteur et conduire à la dispersion d'éléments radioactifs dans l'environnement.

Les principales étapes de l'ICE, décrites en détail par Corradini (1988), sont les suivantes (cf. Figure B.3) :

1. **Prémélange.** Cette étape correspond au premier contact entre l'eau et le corium fondu, qui se fragmente alors en gouttes. La température du corium est telle que le réfrigérant est en régime d'ébullition en film à son contact.

2. **Déclenchement.** Le déclenchement de l'explosion résulte d'une déstabilisation locale des films de vapeur entourant les gouttes de corium. Cette déstabilisation induit une fragmentation fine de ces gouttes, augmentant la surface d'échange et donc les échanges thermiques entre le corium et l'eau.

3. **Propagation.** Les transferts de chaleur entre le corium et l'eau sont si rapides qu'une onde de pression est générée et se propage, déstabilisant les gouttes voisines. Celles-ci se fragmentent à leur tour et génèrent des ondes de pressions s'additionnant entre elles. On a alors une réaction en chaîne : une onde de pression se propage dans le mélange et s'amplifie, fragmentant rapidement l'ensemble des gouttes de corium.

4. **Détente.** La chute de pression qui suit le passage de l'explosion permet la génération et la détente de la vapeur d'eau. De plus, lorsque l'onde de pression atteint la surface libre, une onde de détente se propage dans la direction opposée, diminuant la pression. Celle-ci permet la génération et la détente de la vapeur.

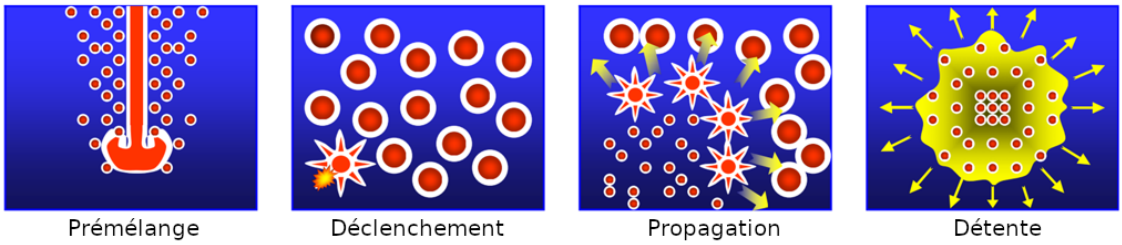

Figure B.3: *Principales étapes de l'ICE conduisant à une explosion vapeur.* (Hansson, 2007).

B.1.1.2 L'installation expérimentale KROTOS

L'installation KROTOS permet d'observer expérimentalement l'interaction entre un corium prototypique et de l'eau servant de liquide de refroidissement. Anciennement située au CCR d'Ispra (Huhtiniemi et al., 1999 ; Huhtiniemi and Magallon, 2001), l'installation a été déplacée au CEA de Cararache au début des années 2000. À Cadarache, un système de radioscopie à rayons X a été ajouté pour caractériser le mélange corium-eau pendant la phase de prémélange.

Cette installation fait sept mètres de haut et est construit sur deux niveaux de la plateforme expérimentale plinius située au CEA Cadarache (Brayer et al., 2012).

Table B.1: *Principales quantités mesurées sur l'installation KROTOS.*

Quantité mesurée	Méthode	Commentaires
Forme du jet (continu/fragmenté), diamètre du jet et Vitesse	Camera	Uniquement sur un champ de vision réduit
Vue d'ensemble du prémélange	Radioscopie par rayons X	Uniquement sur un champ de vision réduit
Pression explosion (local)	Transducteurs de pression dynamique	7 capteurs répartis le long de la paroi du tube à essai, dont 1 dédié au déclencheur
Détection du front de jet	Thermocouples sacrificiels	5 capteurs sur l'axe de la section d'essai
Propriétés des fragments (masse, taille, etc.)	Collecté à la fin de l'expérience	Aucun
Niveau d'eau	Transducteur de niveau d'eau	Aucun
Température de l'eau	Thermocouple	Aucun

Elle est composée de 2 éléments principaux :

- un four résistif pour chauffer le combustible ;
- une cuve abritant le dispositif de formation du jet de corium, l'éprouvette, le déclencheur et les outils de légende.

Les principales grandeurs mesurées par l'installation et les capteurs associés sont présentés dans le Tableau B.1.

D'autres quantités peuvent être déduites des observables précitées. Par exemple, l'impulsion de l'explosion correspond à l'intégrale de la pression sur le temps d'explosion. Cette quantité peut donc être calculée à partir de la pression mesurée lors de l'explosion. Les propriétés des gouttes (masse, taille, etc.) pendant le prémélange et l'explosion peuvent également être estimées à partir des fragments de corium collectés. Pour plus de détails sur l'installation KROTOS et les expériences KROTOS, on peut se référer à (Brayer et al., 2012 ; Bouyer et al., 2015). Dans le cadre de ce travail, nous nous focalisons essentiellement sur l'expérience KROTOS KS4.

B.1.1.3 Le code de simulation MC3D

MC3D est l'un des codes ayant été développés dans le monde pour modéliser l'interaction corium-eau (Meignen and Magallon, 2005). Initialement, celui-ci a été créé par le CEA (Berthoud and Valette, 1994 ; Brayer and Berthoud, 1997). Le code est maintenant la propriété de l'IRSN qui en poursuit sa maintenance et son développement (Meignen, Picchi, et al., 2014 ; Meignen, Raverdy, et al., 2014). Le développement des dernières versions a été effectué dans le cadre du projet pluriannuel RSNR-ICE mené par l'IRSN, répondant au projet dit *post-Fukushima*. Il implique le CEA et est cofinancé par EDF, Framatome et l'ANR (Agence Nationale de la Recherche). La version 3.09 de MC3D a été utilisée pour ce travail. Une nouvelle version du code (la 3.10) a été publiée en 2022.

MC3D est un code multiphasique multi-composants eulérien dédié à la simulation de l'interaction corium-eau. Il utilise un maillage cartésien ou cylindrique en 2D ou en 3D. MC3D est organisé autour d'un noyau commun et de modules pour modéliser chaque composant. Le noyau gère le schéma numérique, le solveur et le problème entrées/sorties. Chaque module décrit, pour les différents composants, les équations de conservation de masse, de quantité de mouvement et d'énergie ainsi que les équations de fermeture nécessaires à la description du modèle.

MC3D modélise l'ICE à l'aide de deux applications chainées. La première, PREMIX, modélise l'étape de prémélange. Cependant, cette application est suffisamment générale pour être utilisée pour d'autres applications. La seconde application, EXPLO, est limitée dans son utilisation à l'étude de la phase d'explosion de vapeur. Un calcul complet est donc effectué en deux étapes, un calcul de prémélange suivi d'une explosion utilisant en entrée les données de sortie du calcul de prémélange. Toutes les entrées et sorties du code sont définies dans le fichier de données de MC3D.

Les entrées

Dans MC3D, les entrées sont scalaires ou catégorielles. De plus, nous pouvons distinguer plusieurs types d'entrées en fonction de leur rôle dans le processus de simulation. Tout d'abord, nous avons les conditions géométriques et physiques. Elles permettent de décrire la situation que l'on souhaite simuler.

Ensuite, nous avons les paramètres numériques : pas de temps, taille de la grille, etc. Ces paramètres influent sur la précision de la simulation, mais aussi sur sa durée et sa stabilité. Ensuite, nous avons les paramètres de modélisation. Ils influencent le comportement des modèles implémentés dans MC3D. Dans ce travail de thèse, nous nous intéressons à ce type d'entrées.

Les sorties

On distingue deux types de sorties dans MC3D :

- les sorties dites de *type 1* correspondant à l'évolution temporelle de quantités physiques moyennées sur l'ensemble du maillage ou localisées à un endroit spécifique du domaine de calcul (pour émuler un capteur par exemple).
- les sorties dites de *type 2* correspondant à la représentation des quantités physiques dans chaque maille du domaine à différents instants de la simulation.

La comparaison directe des sorties de *type 2* aux résultats expérimentaux est difficile parce que ces résultats sont donnés par des capteurs localisés (sauf pour le film vidéo et la radioscopie). Nous n'avons donc pas accès à l'évolution de toutes les quantités observées à chaque endroit du domaine.

Modèles utilisés dans MC3D

Pour rappel, MC3D utilise pour modéliser l'ICE la méthode classique eulérienne multicomposants où chaque composant/mélange est décrit par ses équations de conservation de masse, de quantité de mouvement et d'énergie. Ces équations sont résolues avec une version adaptée de la méthode numérique eulérienne implicite à fluide continu (Harlow and Amsden, 1975 ; Mercier, 1989). Un maillage cartésien ou cylindrique est utilisé où les vitesses sont exprimées aux faces et les autres variables au centre des mailles. En outre, plusieurs modèles sont utilisés pour décrire des phénomènes spécifiques de l'interaction corium-eau.

Les principaux phénomènes spécifiques modélisés dans MC3D sont la description des flux de réfrigérant et de combustible, la fragmentation de la masse fondue, les transferts thermiques et la solidification des gouttes.

L'application prémélange

Dans l'application de prémélange, MC3D se focalise sur la modélisation de la fragmentation et du mélange du jet du corium. Les équations de conservation de masse, de quantité de mouvement et d'énergie sont résolues pour les composants suivants : eau liquide, mélange de vapeur et de gaz non condensables, corium en phase continue (jet de corium) et le corium dispersé. De plus, des modèles physiques sont associés aux différents phénomènes impliqués dans l'ICE : fragmentation du jet, fragmentation et solidification des gouttes, comportement et transitions de phase de l'eau, oxydation du corium, etc.

Modélisation de l'explosion de vapeur

Les aspects fondamentaux des schémas d'écoulement numériques établis pour l'application prémélange sont conservés pour la phase d'explosion. Cependant, il existe des différences importantes :

- le corium dispersé est traité par deux champs distincts, les gouttes/débris issus du prémélange et les fragments fins créés par l'explosion ;
- il n'y a pas de jet de corium continu : l'éventuel jet restant à la fin de l'étape de prémélange est transformé en gouttes de combustible.

De plus, de nouveaux modèles sont ajoutés pour traiter les phénomènes spécifiques qui se produisent pendant l'explosion de vapeur. Le plus important d'entre eux est la fragmentation fine des gouttes de corium.

B.1.2 Objectifs et méthodologie proposée

Nous avons décrit précédemment l'interaction corium-eau, un phénomène qui peut se produire lors d'un accident nucléaire grave. Nous avons ensuite présenté l'installation expérimentale utilisée au CEA pour observer ce phénomène. Enfin, un aperçu global du logiciel de l'IRSN qui simule l'ICE, MC3D, a été donné.

Ainsi, on comprend que de nombreux modèles physiques interagissent entre eux dans le code MC3D pour simuler l'interaction corium-eau. Un ensemble de paramètres est nécessaire pour construire chacun de ces modèles. Ces paramètres de modélisation sont calibrés indépendamment via des expériences dédiées. On peut citer par exemple l'expérience TREPAM pour l'étude de l'ébullition en films (Berthoud and Gros D'Aillon, 2009) et DROPSG ou GALAD pour la fragmentation des gouttes (Achour, 2017 ; Malmazet, 2009). Cependant, ces études ne prennent pas en compte les interactions entre modèles. Il existe donc une incertitude épistémique importante concernant ces paramètres. Pour prendre en compte ces interactions entre modèles, des programmes expérimentaux tels que KROTOS ont été développés. Ils permettent d'observer l'ensemble de l'interaction entre le combustible et le liquide de refroidissement.

Dans ce contexte, le principal objectif de ce travail de thèse est d'étudier la simulation de l'ICE dans l'installation KROTOS par MC3D. En particulier, on s'intéresse à l'évaluation de l'influence des différents paramètres de modélisation et de leurs interactions sur les sorties du code MC3D. Ce travail a également pour but de préparer la calibration de ces paramètres sur les expériences KROTOS afin d'améliorer la fiabilité du code.

B.1.2.1 Le contexte de la quantification des incertitudes

Comme on l'a vu précédemment, les modèles physiques qui décrivent la phénomènologie d'un accident nucléaire grave tel que l'ICE sont complexes et représentés par des équations déterministes. Ils conduisent, lors de la phase de modélisation numérique, au développement de codes de simulation.

Ces outils de simulation prennent en compte de nombreux paramètres d'entrée caractérisant le phénomène étudié. Ces paramètres peuvent être sujets à des incertitudes en fonction du degré de connaissance et de caractérisation du phénomène. De manière générale, deux sources principales composent l'incertitude :

- l'une due à un manque d'information (incertitude épistémique) ;
- l'autre inhérente à la nature aléatoire des paramètres de modélisation (incertitude stochastique).

Quelle que soit leur nature, il est important de quantifier ces incertitudes et d'étudier leur impact sur les sorties du code. Cette analyse permet de valider le modèle mathématique, physique ou numérique, de guider les efforts de caractérisation sur les paramètres les plus influents et plus généralement de mieux comprendre le phénomène modélisé.

Pour prendre en compte les incertitudes des entrées des simulateurs numériques, une approche générale a été développée depuis plus de dix ans (Rocquigny, 2008) et reste un sujet de recherche actif. On peut notamment citer le Groupement De Recherche CNRS MASCOT-Num (acronyme de Méthodes d'Analyse Stochastique pour les COdes et Traitements Numériques)[1] qui regroupe la plupart des acteurs académiques et technologiques autour du développement d'approches stochastiques pour l'analyse des simulateurs numériques.

Dans ce contexte, la démarche méthodologique habituelle de modélisation peut être divisée en plusieurs grandes étapes, comme le montre la Figure B.4.

L'étape A de spécification du problème consiste à définir le système à étudier (modèle, simulateur ou processus de mesure), à identifier les entrées incertaines ou fixes, ainsi que les quantités d'intérêt à étudier. Ces quantités sont dérivées des variables de sortie du modèle. Le simulateur peut être assimilé à une fonction reliant les entrées aux sorties. Voir l'équation (B.1) ci-dessous.

$$f : \begin{array}{l} \mathcal{X} \to \mathcal{Y} \\ \boldsymbol{X} \longmapsto f(\boldsymbol{X}, \boldsymbol{c}) = \boldsymbol{Y} \end{array} \tag{B.1}$$

Ici,

[1] site web : http://www.gdr-mascotnum.fr/

- $\boldsymbol{X}$ est le vecteur aléatoire des entrées incertaines évoluant dans un espace mesurable $\mathcal{X}$, $d \in \mathbb{N}^*$ sa dimension ;
- $\boldsymbol{c} \in \mathbb{R}^{d_c}$ est le vecteur des entrées fixes considérées comme déterministes et $d_c \in \mathbb{N}^*$ sa dimension ;
- f est une fonction mesurable représentant le modèle reliant les entrées aux sorties ;
- $\boldsymbol{Y}$ est la sortie d'intérêt évoluant dans un espace mesurable $\mathcal{Y} \subset \mathbb{R}^p, p \in \mathbb{N}^*$.

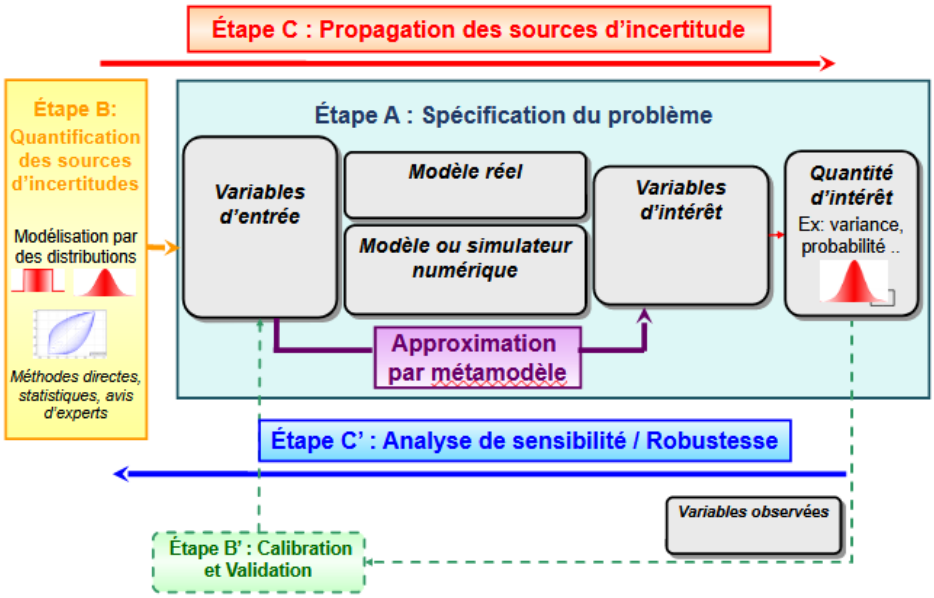

Figure B.4: *Schéma général de la méthodologie de traitement des incertitudes, adapté de* Rocquigny (2008).

La deuxième étape B de cette méthodologie consiste à quantifier l'incertitude des entrées. Dans le cadre probabiliste, les incertitudes des variables d'entrée aléatoires sont modélisées par des distributions de probabilité entièrement ou partiellement connues (Helton, 1997). Ces distributions de probabilité peuvent être choisies en utilisant toutes les données disponibles et en estimant les paramètres des distributions sur ces données, en formalisant l'opinion de l'expert (élicitation) ou en utilisant des données bibliographiques.

La quantification des incertitudes est accompagnée par une étape de calibration, l'étape B'. Dans ce contexte, cette étape vise à caractériser l'incertitude sur les paramètres estimés définissant le modèle physique. En effet, certains paramètres ne sont pas directement observables, mais doivent être estimés. Cette étape peut

également intégrer la détection et l'estimation d'un terme de biais de modèle. Ce terme de biais rend compte du fait que le modèle n'est pas une représentation parfaitement fidèle de la réalité. Classiquement, des méthodes de régression basées sur le maximum de vraisemblance sont utilisées pour effectuer cette étape de calibration (Trucano et al., 2006). Plus récemment, des méthodes de calibration bayésienne par assimilation de données ont également été proposées (Carmassi, 2018). Ce type de méthodes permet de prendre en compte les informations *a priori* sur les paramètres à calibrer et leurs incertitudes. De plus, les paramètres calibrés par les méthodes bayésiennes sont associés à une distribution de probabilité qui quantifie leurs incertitudes post-calibration.

L'objectif de l'étape C de la méthodologie est de mesurer comment l'incertitude des variables d'entrée se propage dans le modèle étudié. L'objectif de cette étape est de quantifier l'influence des incertitudes d'entrée sur les sorties prédites par le modèle. Plus précisément, elle mesure l'effet sur les quantités d'intérêt associées aux sorties. Ces quantités d'intérêt dépendent des objectifs de l'étude. Il peut s'agir par exemple d'une mesure de la moyenne ou de la dispersion des sorties, d'un quantile ou d'une probabilité de dépasser une valeur critique (Cannamela, 2007).

En parallèle et en complément de l'étape de propagation de l'incertitude, il est également important de procéder à une Analyse de Sensibilité Globale (ASG), l'étape C' de la méthodologie. L'analyse de sensibilité vise à déterminer comment la variabilité des paramètres d'entrée affecte la valeur de la sortie ou la quantité d'intérêt (Saltelli et al., 2004 ; Iooss and Lemaître, 2015 ; Da Veiga et al., 2021). Elle permet ainsi d'identifier voire de quantifier, pour chaque entrée ou groupe d'entrées, sa contribution à la variabilité de la sortie. En support de ces grandes étapes, deux thèmes transversaux sont abordés dans le cadre du traitement des incertitudes.

- **La planification d'expériences numériques** : il s'agit de choisir où réaliser les simulations pour échantillonner les distributions modélisées en entrée afin de maximiser l'information collectée en un minimum d'expériences (numériques). Pour ce faire, de nombreuses approches sont possibles, depuis de simples tirages aléatoires $i.i.d$ (type Monte Carlo) jusqu'à des constructions plus élaborées (Fang et al., 2005), telles que les *Latin Hypercube Sampling* (Mckay et al., 1979) par exemple.
- **L'approximation du simulateur par un modèle de régression** : à partir d'observations simulées définies par un plan d'expériences, un modèle d'apprentissage statistique est construit. Ces modèles simplifiés sont parfois appelés modèles de substitution ou métamodèles. Ils se rapprochent le plus possible du simulateur étudié et nécessitent un temps de calcul négligeable.

Ces métamodèles sont ensuite utilisés pour prédire les sorties du code de simulation pour tout ensemble de valeurs d'entrée et permettent ainsi une exploration très intensive du modèle.

Ce contexte méthodologique semble bien répondre aux objectifs de la thèse. En effet, les outils d'analyse de sensibilité globale peuvent être utilisés pour comprendre l'influence de différents paramètres de modélisation sur les sorties du code MC3D. De plus, les méthodes de calibration présentées ci-dessus permettront d'ajuster de manière robuste les paramètres de modélisation du code aux données expérimentales dans un travail ultérieur.

B.1.2.2 Le cas d'application

Paramètres d'entrée incertains considérés

Dans ce travail, nous nous concentrons sur les paramètres de modèles d'MC3D présentés précédemment. Ces paramètres peuvent fortement influencer le comportement général du code.

Les paramètres de fonctionnement (températures, pressions, tailles) ont été fixés sur les mesures de l'expérience KS4. Les incertitudes concernant ces paramètres ne sont pas prises en compte dans ce travail. De plus, certains de ces paramètres sont connus pour avoir une grande influence sur la stabilité du code MC3D et donc sur sa propension à converger correctement. Les $d = 57$ paramètres restants considérés comme incertains ici sont répartis comme suit :

- 39 entrées de l'application prémélange (code 1) ;
- 11 entrées des deux codes ;
- 7 entrées de l'application explosion (code 2).

Le tableau B.2 présente les paramètres associés à chaque modèle du code MC3D.

On a très peu d'informations concernant l'incertitude de ces entrées. On suppose donc une distribution uniforme centrée autour de leur valeur nominale pour caractériser leur incertitude. Leurs domaines de variations ont été choisis en fonction de la plausibilité des valeurs prises d'un point de vue physique et des limites d'utilisation des modèles numériques. La stabilité du code MC3D a également été prise en compte.

Table B.2: *Liste des entrées incertaines associées à chaque sous-modèle MC3D.*

Modèle de MC3D	Entrées concernées
Fragmentation du jet	1-6
solidification des gouttes	7-9
Fragmentation des gouttes et transferts thermiques	10-33
Compressibilité du corium	34
Paramètres équation de conservation des moments des gouttes	35-40
Propriétés physiques du corium	41-50
viscosité artificielle	51
Paramètres généraux de l'explosion vapeur	51-57

B.1.2.3 Objectifs de thèse et méthodologie

Le travail proposé dans ce doctorat vise à répondre aux deux besoins opérationnels suivants :

1. Comprendre l'influence des paramètres de modélisation sur les sorties du code.
2. Préparer une calibration de ces paramètres de modélisation sur des données expérimentales.

Ainsi, l'objectif de ce travail est de développer et d'appliquer une méthodologie dans le cadre de la quantification d'incertitude pour répondre à ces besoins. Pour ce faire, il est nécessaire de prendre en compte les spécificités du contexte industriel décrit précédemment. En effet, la modélisation de l'interaction corium eau requiert l'utilisation d'un grand nombre de modèles reliés entre eux dans un simulateur de modèle global. Par conséquent, le code MC3D est très coûteux en termes de calcul (nombre limité de simulations possibles). De plus, le nombre de paramètres d'entrée considérés est assez important (57). Ces deux spécificités limitent l'exploration de l'espace d'entrée. L'imbrication des phénomènes modélisés avec des effets parfois opposés rend en outre le comportement global du code difficile à anticiper d'un point de vue physique. En effet, la relation entre les entrées et les sorties est fortement non-linéaire. Cette complexité peut même conduire, pour un ensemble de données d'entrée plausible à première vue, à des échecs de code.

Pour répondre aux objectifs mentionnés ci-dessus en tenant compte du contexte particulier de notre problème industriel, une méthodologie séquentielle est proposée. Adaptée de la méthodologie ICSREAM (acronyme de Identification of penalizing Configurations using SCREening And Metamodel) de Marrel et al. (2022), elle consiste en cinq grandes étapes :

1. **Échantillonnage de l'espace d'entrée.** Connaissant le domaine de variation des variables d'entrée, l'espace des entrées est échantillonné de manière aléatoire ou pseudo aléatoire. Une exécution du code est ensuite effectuée sur chaque point échantillonné du plan d'expérience. Le jeux de données entrées/sorties obtenu constitue l'*échantillon d'apprentissage*. Le but est ici d'explorer avec cet échantillon l'espace d'entrée et d'obtenir un maximum d'informations sur les sorties du code. Ne disposant que d'un budget limité pour les simulations, nous utilisons une méthode d'échantillonnage quasi aléatoire, appelée *Latin Hypercube Sampling* (LHS) (Mckay et al., 1979), au lieu d'une méthode d'échantillonnage aléatoire classique. Cette méthode assure une meilleure couverture spatiale des distributions marginales tout en conservant certaines propriétés de l'échantillonnage Monte-Carlo classique.

2. **Analyse de sensibilité des échecs de code.** Pendant l'exploration de l'espace d'entrée, il a été constaté qu'une proportion significative des simulations d'MC3D n'a pas convergé. L'analyse de ces échecs de code pour comprendre quelles entrées ont le plus d'influence sur eux permet de mieux comprendre le fonctionnement du code. Cette étape a donc été intégrée à cette méthodologie. Cela pourrait également aider à mieux identifier le domaine d'utilisation possible des entrées.

3. **Sélection des entrées à partir de l'analyse de sensibilité des sorties.** À partir de l'ensemble des données, une sélection est effectuée en utilisant deux outils d'analyse de sensibilité. Le premier est l'indice de Sobol' de premier ordre. Cet indice est basé sur la variance conditionnelle des sorties par rapport aux entrées (Sobol', 1993). L'autre est le critère d'indépendance de Hilbert Schmidt nommé HSIC (acronyme de *Hilbert Schmidt Independence Criterion*). Cet outil est basé sur la mesure des dépendances entre les entrées et les sorties (Gretton et al., 2005). Un groupe d'entrées considérées comme *significatives* est ensuite sélectionné en utilisant ces outils.

4. **Approximation des sorties par des modèles de régression (appelés métamodèles).** À partir du jeux de données entrées/sorties et en utilisant la sélection réalisée à l'étape 3, des modèles de régression sont construits pour ajuster les sorties du simulateur. Ces métamodèles sont destinés à remplacer le code initial. Pour cela, la méthode de régression par processus gaussien (GP) est utilisée (Rasmussen and Williams, 2005) dans ce travail. Cette méthode présente plusieurs avantages par rapport aux autres méthodes de régression : prédiction probabiliste et propriété d'interpolation exacte, ainsi que l'évaluation des capacités de modélisation du simulateur numérique (Santner et al., 2003 ; Marrel et al., 2008).

5. **Utilisation des modèles de régression (construits à l'étape 4) pour effectuer une analyse de sensibilité plus précise.** Une analyse de sensibilité quantitative basé sur une exploration intensive de l'espace d'entrée est effectuée en utilisant les modèles de régression au lieu du code de simulation. Ceci est rendu possible par le temps d'évaluation négligeable des métamodèles. Nous sommes focalisés sur l'estimation des indices de sensibilité basés sur la variance, les indices de Sobol' de premier ordre (comme à l'étape 3) et, beaucoup plus informatif, les indices de Sobol' totaux. Cette analyse de sensibilité quantitative prépare la calibration en classant les entrées par niveau d'influence sur les sorties.

B.2 Conclusion du manuscrit et perspectives

Le contexte industriel de cette thèse est la simulation d'accidents nucléaires graves dans les réacteurs à eau pressurisée. Plus précisément, nous nous sommes concentrés sur la modélisation de l'Interaction Corium-Eau (ICE) pouvant conduire au phénomène d'explosion de vapeur. Cette explosion peut endommager les structures et compromettre l'intégrité du réacteur nucléaire. Le code de calcul utilisé au CEA pour simuler ce phénomène est le logiciel MC3D (MultiComponent Code 3D). Cependant, les méthodes proposées dans ce travail sont suffisamment générales pour être appliquées à d'autres codes de simulation.

L'ICE est un phénomène physique complexe et non-linéaire faisant intervenir de nombreux mécanismes. Pour simuler ce phénomène, le code MC3D utilise un grand nombre de paramètres modélisant ces différents mécanismes. Un temps de calcul relativement long est également nécessaire pour chaque simulation. Dans ce contexte, l'objectif principal de ce travail a été d'étudier la simulation de l'ICE par MC3D. En particulier, nous nous sommes intéressés à l'évaluation de l'impact des paramètres de modèles considérés comme incertain sur les sorties du code MC3D dans le cadre général de la quantification des incertitudes. Cette étude vise également à préparer la calibration future de ces paramètres sur des données expérimentales afin d'améliorer la fiabilité du code. Pour étudier l'impact de ces paramètres de modèles, nous avons proposé une méthodologie basée sur des outils statistiques avancées, décomposée en cinq grandes étapes :

1. Échantillonnage de l'espace d'entrée.
2. Analyse de sensibilité des échecs de code.
3. Sélection des entrées à partir de l'analyse de sensibilité des sorties.
4. Approximation des sorties par des modèles de régression (appelés métamodèles).
5. Utilisation des modèles de régression de l'étape 4 pour effectuer une analyse de sensibilité plus précise.

Dans le premier chapitre de la thèse, nous avons présenté en détail le contexte industriel, les objectifs de la thèse et la méthodologie proposée pour y répondre. Nous avons également décrit le cas d'application. En effet, nous nous sommes focalisés uniquement sur paramètres de modèles du code, que nous avons considérés comme incertains. Les conditions physiques des simulations ont ainsi été fixées et correspondent à une expérience réelle permettant l'observation de l'interaction corium-eau, KROTOS KS4. Nous nous sommes également concentrés uniquement sur un nombre limité de sorties scalaires du code MC3D. Les autres chapitres de la thèse ont traité des différentes étapes de notre méthodologie.

Le thème du chapitre 2 a été la planification d'expériences numériques (l'échantillonage), correspondant à la première étape de la méthodologie présentée. Une méthode d'échantillonnage caractérise la manière dont l'espace d'entrée est exploré et donc les propriétés de l'ensemble des données entrée-sortie. Pour analyser un code comme MC3D, il est donc crucial de bien la choisir. Dans ce chapitre, nous avons d'abord donné un aperçu des méthodes de planification d'expériences numériques classiques. Dans notre application, nous avons utilisé la méthode *Latin Hypercube Sampling* (LHS) car elle présente à la fois de bonnes propriétés de couverture d'espace et des propriétés de convergence asymptotique. Un travail théorique original réalisé sur la convergence des Z-estimateurs sous LHS a ensuite été présenté. Une réduction de la variance asymptotique ainsi qu'un théorème central limite pour cette classe d'estimateurs sous LHS ont été établis. Cependant, d'importantes conditions de régularité sont supposées pour obtenir ces résultats. Généraliser ces travaux en levant certaines de ces restrictions est une perspective intéressante.

Nous avons ensuite discuté de la convergence sous LHS d'autres statistiques utilisées dans notre méthodologie. Des travaux théoriques sur la statistique de Kolmogorov ont d'abord été présentés. Des expériences numériques sur la convergence d'un estimateur des indices de Sobol' de premier ordre basé sur une statistique de rang ont également été proposées. Nous avons comparé empiriquement le comportement asymptotique de cet estimateur sous échantillonnage pur Monte Carlo, LHS et LHS optimisé. D'après ces travaux, le LHS classique semble être un bon compromis entre le biais et la variance pour l'estimation des indices de Sobol' du premier ordre.

Enfin, la convergence sous LHS de deux autres statistiques utilisées dans ce travail a été discutée, à savoir le *Hilbert Schmidt Independence Criterion* (HSIC) et les hyperparamètres des modèles de régression par Processus Gaussien (PG), ces derniers étant estimés par maximum de vraisemblance, qui est un cas particulier de la Z-estimation. L'étude théorique de leur convergence asymptotique est également une perspective à considérer.

Le chapitre 3 a porté sur l'étape 2 de notre méthodologie, à savoir l'analyse des échecs de code par analyse de sensibilité. Les échecs de code sont des problèmes courants dans le contexte de la simulation numérique de phénomènes physiques complexes. Détecter les entrées impliquées dans ces défaillances et leur niveau d'implication est donc utile pour améliorer la robustesse de ces codes de simulation. Les résultats obtenus lors de l'exploration de l'espace d'entrée du code MC3D ont motivé ce travail. En effet, après avoir déterminé notre plan d'expériences, nous avons exécuté le code sur les points d'échantillonnage et observé qu'un tiers des simulations ne convergeaient pas. Notre objectif était donc de comprendre ce phénomène.

En considérant l'occurrence d'un échec de code comme une sortie binaire à part entière, les principes de l'analyse de sensibilité globale peuvent être utilisés pour identifier les variables qui ont le plus d'impact sur les défaillances. Dans ce contexte, nous avons d'abord introduit deux méthodes pour détecter les entrées qui pourraient être impliquées dans les défaillances du code. Premièrement, nous avons étudié la distribution des entrées des échantillons tels que le code échoue à converger. Cela a été fait en utilisant le test d'adéquation de Kolmogorov-Smirnov (K-S). Cette méthode de test permet de détecter les variables dont la distribution conditionnelle aux échecs du code diffère significativement de leur distribution initiale. Comme notre plan d'expérience initial est un LHS, la procédure de test a été adaptée à ce cas.

Une deuxième méthode, basée sur le HSIC, a ensuite été proposée. Cet outil permet de détecter les dépendances probabilistes entre les variables d'entrée et les échecs de code. De plus, un test statistique d'indépendance peut être construit à partir de ce critère. Celui-ci permet d'effectuer une sélection et un classement des entrées par ordre d'influence sur la sortie (ici les échecs de code). À partir des résultats des tests K-S et HSIC, un ensemble d'entrées significativement influentes a été sélectionné. Une analyse graphique des densités conditionnelles aux échecs de code a été effectuée pour ces entrées. Nous avons ensuite analysé les résultats obtenus d'un point de vue physique.

Une deuxième analyse de sensibilité a ensuite été réalisée pour détecter et identifier les interdépendances qui apparaissent entre les entrées lorsque l'on considère l'échantillon des simulations qui ont convergé. En effet, seules ces données-là sont utilisées dans la suite de la méthodologie. Pour cela, la mesure de dépendance HSIC a été à nouveau utilisée, mais cette fois entre chaque paire de variables d'entrée conditionnées à l'occurrence d'un échec de code. Ainsi, trois groupes d'interdépendances entre les entrées conditionnées ont été détectés avec cette méthode.

Une suite à ce travail pourrait être de distinguer les différentes causes de défaillances du code dans l'analyse. L'utilisation d'algorithmes de classification pour identifier les domaines de valeurs d'entrées conduisant à des échecs de code est également une perspective intéressante. Il serait en effet très utile en pratique de pouvoir prédire les échecs de code afin d'éviter le gaspillage des ressources de calcul.

Enfin, dans le chapitre 4, nous avons abordé les trois autres étapes de notre méthodologie : la sélection de variables d'entrée importantes, la métamodélisation des sorties et l'analyse de sensibilité sur ces métamodèles. Étant donné le grand nombre de paramètres considérés dans notre étude, l'étape de sélection (appelé aussi criblage) est essentielle. Elle aboutit à la classification des variables d'entrée en deux sous-groupes : les variables explicatives par rapport aux sorties du code et les autres. Étant donné le nombre limité de simulations disponibles, deux outils avancés d'analyse de sensibilité ont été utilisés pour effectuer cette sélection des entrées : le HSIC et les indices de Sobol' de premier ordre. Ces deux outils sont bien adaptés à l'analyse des fonctions de modèles non linéaires et fournissent des informations complémentaires. Le HSIC (avec le test d'indépendance associé) est une méthode de sélection qui permet de détecter les entrées dont l'influence n'est pas significative. Les indices de Sobol' de premier ordre fournissent des informations supplémentaires sur la proportion de variance expliquée par chacune de ces variables individuellement et donc sur la partie additive de la relation entrée/sortie.

Ensuite, en ne considérant que les entrées considérées comme explicatives, des modèles de régression (également appelés métamodèles) ont été construits afin d'approximer les relations entre les entrées et les sorties étudiées. La méthode classique de régression par processus gaussien a été utilisée pour effectuer cette métamodélisation. Une fois entraînés sur l'ensemble des simulations disponibles, ces métamodèles peuvent produire de grandes quantités de résultats prédits avec un coût de calcul négligeable. Ils permettent donc une exploration intensive du domaine de variation des entrées. Cependant, une validation de leur capacité de prédiction, c'est-à-dire de leur capacité à prédire correctement les résultats du code avec des intervalles de prédiction fiables, est nécessaire.

Cette étape de validation a révélé que certaines sorties de MC3D étudiées ont été correctement reproduites alors que pour d'autres, seule la moitié de la variance de la sortie est expliquée par le métamodèle ainsi construit. Malgré ces résultats contrastés, une analyse de sensibilité complète, facilement interprétable et quantitative des sorties a été réalisée à l'aide de ces métamodèles. Plus précisément, les indices classiques de premier ordre et totaux de Sobol' ont été estimés avec les métamodèles. Cela a permis d'identifier trois entrées ayant une influence majeure sur les différentes sorties du code. Toutes trois sont directement liées à un phénomène spécifique de l'ICE, la fragmentation des gouttes de corium dans l'eau. En outre, des diagrammes de dispersion ont permis une compréhension physique plus exhaustive de l'impact de ces trois entrées sur les sorties.

En résumé, nous avons proposé dans ce travail une approche moderne et applicative pour effectuer une analyse de sensibilité quantitative sur des codes industriels complexes (tels que MC3D) avec une interprétation physique approfondie des effets des entrées incertaines.

Les perspectives concernant notre méthodologie sont nombreuses. La plus importante est la calibration du code sur les expériences KROTOS. Notre travail a permis d'identifier un nombre restreint d'entrées qui semblent avoir un impact significatif sur l'ICE simulée par MC3D. L'étape suivante consiste naturellement à ajuster le code MC3D, de manière robuste, aux données expérimentales dont nous disposons. Le défi majeur concernant cette perspective concerne la quantité de données sur lesquelles nous devons effectuer cette calibration. En effet, peu de données expérimentales sont disponibles car les expériences KROTOS sont très coûteuses (plusieurs millions d'euros par expérience). Pour réaliser cette calibration, les méthodes bayésiennes peuvent ainsi être prometteuses. Elles présentent en effet plusieurs avantages. Tout d'abord, le cadre bayésien permet de prendre en compte de façon probabiliste les informations dont nous disposons *a priori* sur les paramètres et leur incertitude. Cette information est essentielle dans notre cas étant donné la faible quantité de données expérimentales disponibles. De plus, dans ce cadre, les paramètres calibrés sont associés à une loi de probabilité qui quantifie leurs incertitudes post-calibration.

Cependant, de nombreuses simulations sont nécessaires pour effectuer cette calibration bayésienne. L'utilisation de métamodèles apparaît donc essentielle compte tenu du temps de calcul élevé du code MC3D. En particulier, les régressions par processus gaussiens fournissent un bon cadre théorique pour la calibration bayésienne (voir par exemple (Carmassi, 2018) ou (Damblin, 2015) pour plus de détails). Hors, les métamodèles actuellement construits ne sont pas assez bons pour permettre une calibration précise du code sur des données expérimentales. Ainsi, une étape préliminaire à la calibration est l'amélioration des métamodèles.

L'augmentation de la base d'apprentissage à l'aide des techniques d'*Expected Improvement* (Jones et al., 1998) est une approche prometteuse, mais son application en dimension élevée reste compliquée. Les résultats de l'analyse de sensibilité pourraient être utilisés pour réduire l'espace des entrées sur lequel travailler. De plus, la probabilité non-négligeable d'échec de code doit également être prise en compte. Une autre possibilité pour améliorer la précision de la métamodélisation pourrait consister à alléger l'hypothèse de stationnarité. L'utilisation des *Deep Gaussian Process* (Damianou, 2015) avec un ensemble de données plus important et en se concentrant sur les principales entrées du code par exemple pourrait être une option intéressante. Une autre possibilité consiste à alléger les contraintes d'interpolation des PG loin de la région d'intérêt concernant la calibration, avec les *relaxed Gaussian process* introduit dans (Petit et al., 2021) par exemple. Enfin, il peut être intéressant d'effectuer de nouveaux calculs sur un espace des entrées plus restreint. La réduction de l'espace à considérer peut être fait de deux manières. Premièrement, on pourrait réduire le nombre d'entrées considéré en ne prenant que celles qui influencent l'ICE d'après l'analyse sensibilité effectuée. Les domaines de variations des entrées considérées peuvent également être réduits pour mieux se focaliser sur les valeurs expérimentales observées. Une réduction de la taille de l'espace d'entrée pourrait en effet améliorer la métamodélisation dans les domaines d'intérêt pour la calibration.

- Did you do it?
- Yes.
- What did it cost?
- Everything.