Statistical methods for the study of computer experiments failures: Application to a fuel-coolant interaction simulation code

Faouzi Hakimi, Claude Brayer, Amandine Marrel, Fabrice Gamboa, Benoît Habert

Statistical methods for the study of computer experiments failures: Application to a fuel-coolant interaction simulation code

Faouzi Hakimi,[*,a,c] Claude Brayer,[a] Amandine Marrel,[b,c] Fabrice Gamboa,[c] and Benoît Habert[a]

[a] *CEA, DES, IRESNE, DTN, Cadarache F-13108 Saint-Paul-Lez-Durance, France*

[b] *CEA, DES, IRESNE, DER, Cadarache F-13108 Saint-Paul-Lez-Durance, France*

[c] *Institut Mathématique de Toulouse, 31062 Toulouse, France*

[*]Email: faouzi.hakimi@cea.fr

Number of pages: 21
Number of tables: 1
Number of figures: 6

Abstract

In the framework of risk assessment in nuclear accident, simulation tools are widely used to understand and model physical phenomena. These simulation tools take into account a large number of uncertain input parameters. We often use Monte-Carlo type methods to explore their range of variation: the inputs space is randomly sampled and a code run is performed on each sampled point. However, some of these code runs may fail to converge. Analyzing these code failures to understand which of the inputs have the most influence on them lead to a better understanding of how the code works. It also intends to improve the robustness of the simulation software and code computations. For this purpose, we propose two complementary approaches performing a statistical analysis of the code failures. The first approach is based on Goodness-of-fit tests and compares conditional probability distributions according to code failures to a reference one. A second approach, based on a dependence measure named the Hilbert-Schmidt Independence Criterion (HSIC), provides another way to measure the global dependence between the inputs and the code failures. The development of this methodology is carried out in the context of severe nuclear accidents. More especially, the presented methods are applied for the study of a simulation code that simulates the fuel-coolant interaction in severe nuclear accident context, MC3D.

Keywords — Code failure analysis, Sensitivity analysis, Hilbert-Schmidt Independence Criterion, Severe nuclear accidents, Code robustness

I. INTRODUCTION

In the framework of risk assessment in nuclear accident, some calculation tools are now widely used to understand, model and predict physical phenomena. In particular, the characterization of severe accident involves the use of a large amount of models linked together in a global numerical simulator. Each of these models and the way they are linked together require a large set of input parameters. These parameters may be subject to uncertainties due to the partial characterisation of the phenomenon, the state of knowledge or modelling uncertainties. It is therefore necessary to explore the whole range of variation of these inputs in order to understand the overall behavior of the simulation code.

To achieve this exploration, Monte-Carlo type methods are often used: the inputs space is randomly sampled. A code run is then performed on each sampled point. These type of exploration methods have the advantage of allowing the deployment of statistical inference tools: central limit theorem, estimation theory, statistic test and confidence interval theory, etc. Monte-Carlo methods thus enable, at the end of the exploration, to get quantitative results to analyze with associated degrees of confidence (e.g via confidence intervals and statistical tests). However, the simulation codes considered here are generally expensive in terms of computing time. This means that we have access to only a limited number of simulations. This limit guides the choice of statistical tools and methods to be used.

In some cases, a portion of the code run does not manage to converge. This can be due to numerical problems or suitability of the models used by the simulation code. Analyzing these code failures to understand which inputs have the most influence on them will allow a better understanding of how the code works. This will also provide valuable information on the values of the input parameters to avoid code failures. More generally, this analysis intends to improve the robustness of the simulation software and code computations.

Ideally, the direct use of classification algorithms such as random forest [1], logistic regression [2] or support vector machine (SVM) [3] could predict code failures according to the inputs. Nevertheless, as previously mentioned, accident simulation codes are complex and take many variables as input. Their computational cost is also often high, which limits the number of available simulations for the training set. These drawbacks lead to some difficulties in obtaining satisfactory results using this kind of algorithms.

In this context, we propose statistic-based tools to analyze code failures. Failure occurrence can be considered as a binary output in its own right. It allows to consider code failures analysis in the general context of sensitivity analysis. Sensitivity Analysis methods aim at determining how the variability of model's inputs affects the fluctuation of its output. Many methods have been developed for this purpose [4]. Most of classic tools used for sensitivity analysis, such as Sobol' indices [5] or the Elementary Effect method [6, 7] 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 estimation of total Sobol' indices requires a large number of simulations. Recently, tools based on dependence measures have been proposed for global sensitivity analysis. These tools allow to remove some of these limitations [8]. Among them, the Hilbert-Schmidt Independence Criterion [9] denoted by HSIC, generalizes the notion of covariance between two random variables. These tools can be used with many different types of variables, including binary ones. Moreover, the HSIC allows to fully characterize the independence between two variables (with certain parametric choices). The HSIC estimation only requires only a limited budget of simulations. Last but not least, statistical tests of independence can be built upon HSIC measures. In this work, we propose two methods to perform a sensitivity analysis on

code failures:

- A first approach based on Goodness-of-fit tests that compares the conditional probability distributions knowing the code failures;
- Another approach, based on the HSIC that measures the global dependence between the inputs and the occurrence of code failures.

The development of these methods is carried out in the context of severe nuclear accidents. More precisely, the simulation code under study is MC3D [10, 11]. This code models the fuel-coolant interaction (FCI). This interaction can lead to the steam explosion phenomenon [12]. Further details about the application case will be given in a dedicated section. When exploring the input space of this code using a Monte-Carlo method, we found a fairly high failure ratio (around 35%). Understanding these code failures may help to reduce this failing ratio. This can be interesting knowing that several hours to few days are necessary for one run of this code.

Thus, this paper is organized as follows: Section 2 aims at introducing the methodological framework and the notations used often. Section 3 presents the application context. Sections 4 and Section 5 present the two different aforementioned approaches proposed to perform a sensitivity analysis on code failures. An example use of the results provided by the two methods is given in Section 6. This application stands on the code modeling of severe nuclear accident. Finally, Section 7 provides some conclusions and perspectives. More technical material is postponed in the appendices.

II. MATHEMATICAL FORMALISM

A simulation code can be viewed as a model function $\mathcal{M}$ linking the inputs to one (or several) output(s):

$$\mathcal{M}: \begin{array}{l} \mathcal{X} \to \mathcal{Y} \\ \mathbf{X} \longmapsto \mathcal{M}(\mathbf{X}) = \mathbf{Y}. \end{array} \tag{1}$$

Here,

- $\mathbf{X} = \{\mathrm{X}_1, ...\mathrm{X}_d\}$ is the vector containing the d uncertain inputs evolving in a measurable space denoted by $\mathcal{X} \in \mathbb{R}^d$;
- $\mathcal{M}$ is the model function that links the inputs to the outputs;
- $\mathbf{Y}$ is the vector of the code outputs and $\mathcal{Y}$ its range of variations.

We are studying here the quantity of interest Z representing the code failures, and defined by the function $f_Z(\mathbf{X}) : \mathcal{X} \to \{0, 1\}$:

$$\mathrm{Z} = f_{\mathrm{Z}}(\mathbf{X}) = \begin{cases} 1 & \text{if the model } \mathcal{M}(\mathbf{X}) \text{ fails} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Figure 1 summarizes the proposed formalism.

As the model $\mathcal{M}$ is not known analytically, a direct computation of the distribution of Z is not straightforward. Indeed, only observations (corresponding to code runs) of $\mathcal{M}$ are available. It is therefore assumed in the following that we have a n-size input sample and its associated output sample. This sample is denoted by ($\boldsymbol{X}$,$\boldsymbol{Z}$), where:

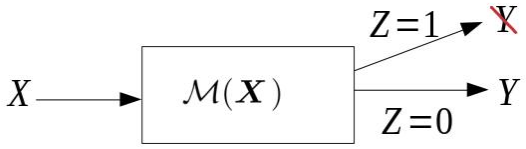

Fig. 1. *Diagram summarizing the proposed mathematical formalism.* Z = 1 *means that the code failed and there is nothing else to observe. If* Z = 0*, the code has no error and we have access to the output* $\mathbf{Y}$*.*

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

In the context of code failures, we decided to organize the DoE in two sub-designs: the first part is constituted by the realisations for which the code fails while the second part corresponds to the successful simulations. Thus, we define them by:

- $A = \{l_1, \ldots, l_{|A|}\}$: subset of the indices taken from $\{1, \ldots, n\}$ such that the corresponding simulations fail ($Z = 1$);
- $\bar{A} = \{1, \ldots, n\} \backslash A$ the subset of the indices taken from $\{1, \ldots, n\}$ such that the corresponding simulations do not fail ($Z = 0$);
- $\boldsymbol{X}_A = \left(\boldsymbol{x}^{(l_1)}, \ldots, \boldsymbol{x}^{(l_{|A|})}\right)^T$ the matrix containing the elements of X such that the code fails, $\boldsymbol{X}_A \in M_{|A|,d}(\mathbb{R})$;
- $\boldsymbol{x_{A,j}} = \left(x_j^{(l_1)}, \ldots, x_j^{(l_{|A|})}\right)$ the values of the j^{th} variable for the failing part of the sample;
- $\boldsymbol{X}_{\bar{A}}$ the matrix containing the elements of $\boldsymbol{X}$ such that the code does not fail.

III. APPLICATION CONTEXT

The work proposed here is motivated by the study of the failures of a severe nuclear accident simulation code. An accident in a nuclear reactor becomes severe when the core starts to melt and lose its integrity. The molten materials coming from the core is called corium. Severe accident simulation codes help to improve the safety of nuclear reactors by providing information to predict failure of barriers or safety systems. These codes are generally meshed (or at least partially meshed) with physical modeling being done in each mesh in the form of conservation laws.

The code on which we apply the proposed tools is the simulator MC3D [10, 11]. This calculation code simulates the Fuel-Coolant Interaction (FCI) in nuclear severe accident context. The fuel-coolant interaction is a complex, non-linear phenomenon in which many mechanisms interact. Indeed, under certain conditions, these interactions result in 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. The code MC3D that simulates this phenomenon is composed of two chained models, related to the main phases of the FCI [12]:

- the premixing, corresponding to the fragmentation in the water of the molten corium jet into large droplets;
- the steam explosion, corresponding to the fine fragmentation of the droplets generated during the premixing phase. This leads to an increase of the heat transfers between these fragments and the generation a pressure wave that propagates in the fuel coolant mixture.

The application case here corresponds to the study of the failures of the code modeling the premixing phase. The framework of the application case is the simulation by MC3D of corium interaction in the KROTOS experimental facility [13, 14]. Figure 2 depicts the spatial meshing used for the code runs (on the left) and the Krotos facility (on the right). The spatial mesh is 2D axisymmetric with 27 R axis meshes and 103 Z axis meshes. This 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 grey areas correspond to solid structures (crucible and jet formation device, or conical bottom of the test tube). The duration of each run of the code is between 8 hours and a day. Note that runs can be done in parallel (one run per calculation node).

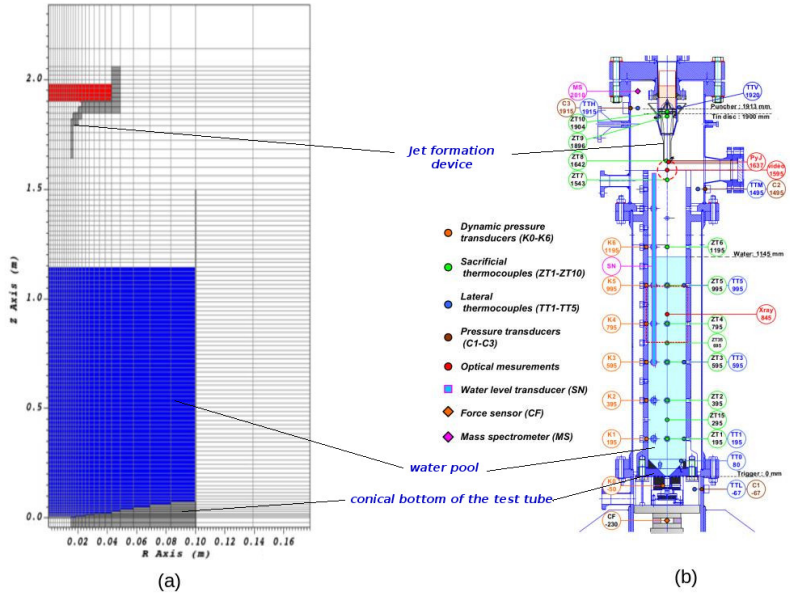

Fig. 2. *(a) Representation of the spatial mesh for the numerical experiment.The R-axis is expanded with respect to the Z-axis for better visibility.(b) Representation of the Krotos facility*

The number of input parameters of the MC3D considered here is $d = 50$. These parameters vary uniformly around their nominal value. 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 rescaling is performed before running the simulation. Hence, we will consider in the following that all the d inputs vary uniformly in $[0, 1]$: $\mathbf{X} \sim U_{[0,1]^d}$.

In our application, the design of experiment $\boldsymbol{X}$ has been obtained by a space-filling sampling method, named Latin Hypercube Sampling (denoted LHS [15]). We proceed so to improve the distribution of the realizations along the 1D-sub-projections of the design. Finally, we have in our study case $n = 2000$ simulations of the code. Among them, $l_{|A|} = 700$ failed. This represents 35% of the code runs. Considering the computational cost of each code simulation, this significant number of code failures justifies in practice the development of the methods presented here.

IV. ASSESSMENT OF THE IMPACT OF INPUTS ON CODE FAILURES BASED ON COMPARISONS OF CONDITIONAL DISTRIBUTIONS

In this section, we present a method to study the influence of each input parameter X_j on the code failures. For this aim, a study of the realizations $\boldsymbol{x_{A,j}}$ such that the code fails and their associated distribution function is performed. Indeed, if we suppose that an input X_j has no direct effect on code failures, then the subset A such that the code fails ($\mathrm{Z} = 1$) is independent of the values taken by X_j. That implies that the samplings $\boldsymbol{x_{A,j}}$ follows the same distribution as $\boldsymbol{x_j}$. Thus, since the realizations of $(\boldsymbol{x_j})_{j\in[\![1,d]\!]}$ are uniformly distributed, if an input X_j is not directly involved on code failures, its marginal distribution $\boldsymbol{x_{A,j}}$ should remain uniformly distributed. From this, a first solution to detect the inputs influencing the code failure is to compare the distribution of each $\boldsymbol{x_{A,j}}$ to a uniform law $U_{[0,1]}$ using a statistical goodness-of-fit test.

The classical Kolmogorov-Smirnov (KS) [16, 17] goodness-of-fit test is used here to perform this comparison. This statistical method is well suited to compare a sample distribution to a reference one. 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 an initial analysis and given the large number of variables involved in the application case, potentially missing few influential variables may be less of a problem than wrongly selecting some.

The next subsection gives a brief review on the Kolmogorov-Smirnov (KS) statistic and the associated test method. It also presents how we use it to study the influence of inputs on code failures. More theoretical details on the KS statistic and test are given in Appendix 1.

IV.A. Comparison of the marginal of each input to the uniform distribution using Kolmogorov-Smirnov test

Let X be a random variable and F_0 the reference cumulative distribution function. Let $\boldsymbol{x} = \left(x^{(1)}, \ldots, x^{(n)}\right)$ be an independent and identically distributed (i.i.d) sample of X. The Kolmogorov-Smirnov statistic corresponds to the re-scaled maximum difference between the cumulative distribution function of the data denoted F_n and the reference one F_0. Formally, it is defined by Eq. (3):

$$S_n^{KS}(F_n, F_0) = n^{1/2} \sup_t |F_n(t) - F_0(t)|. \tag{3}$$

$F_n(t)$ is the empirical estimation of the true cumulative distribution of X defined by $F_n(t) = \sum_{i=1}^{n} \mathbf{1}_{x^{(i)}<t}$. Here, $\mathbf{1}_A$ is the indicator function of a set A.

The Kolmogorov-Smirnov (KS) test is built upon this statistic S_n^{KS}. Consider the statistical test where the null hypothesis is "H_0: $F = F_0$" and the alternative hypothesis is "H_1: $F \neq F_0$".

The observed value $s^{KS}_{n,obs}(F_n, F_0) = n^{1/2} \sup_t |F_n(t) - F_0(t)|$ is computed on the data. It is then compared to the theoretical distribution of the statistic S^{KS}_n under the null hypothesis H_0. As usually in statistical inference, the p-value, defined here by $p^{KS} = \Pr(S^{KS}_n > s^{KS}_{n,obs}|H_0)$, is computed to carry out this comparison. More details on KS statistic and test are given in Appendix 1.

Thus, the p-value drives the decision process. The lower the p-value, the stronger the null hypothesis is rejected. A significance level α (also known as "Type I error"), corresponding to an error of rejecting wrongly the null hypothesis H_0, is classically associated to the test procedure. 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 arbitrarily chosen. In practice, it is usually set at $\alpha = 0.01$, $\alpha = 0.05$ or $\alpha = 0.10$, 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. Thus, the idea is to apply the Kolmogorov-Smirnov test to assess the marginal influence of each input. For each vector samplings $(\boldsymbol{x_{A,j}})_{j\in\{1,..d\}}$, the statistic $s^{KS}_{n,obs}(F_{j,n}, F_{U_{[0,1]}})$ is computed, $F_{j,n}(t)$ being the empirical distribution function of $\boldsymbol{x_{A,j}}$. The p-values $\{p^{KS}_1 ... p^{KS}_d\}$ associated with these tests are then calculated. The obtained p-values allow us to rank the inputs by likelihood of the null hypothesis. The p-values below the threshold α correspond to the inputs for which the hypothesis of a uniform distribution of $\boldsymbol{x_{A,j}}$ is highly unlikely. These inputs are therefore those likely to have a significant impact on the code failures.

IV.B. Results for the case study

The method described above is applied to the simulations of the MC3D premixing code. Kolmogorov-Smirnov test is applied on each of the $d = 50$ one-dimensional sub-projection realizations. As stated in Section II, the DoE $\boldsymbol{X}$ is obtained using a space-filling method named LHS. Therefore, the samplings of $\boldsymbol{x}_{A,j}$ are not independent, even under the null hypothesis. However, the majority of the theoretical convergence of the Kolmorov-Smirnov statistic (presented in Appendix 1) have been only demonstrated under the hypothesis of i.i.d samplings of the inputs. For this reason, we used for the case study an empirical distribution of the Kolmogorov statistic to estimate the d p-values $\{p^{KS}_1 ... p^{KS}_d\}$ instead of the theoretical asymptotic distribution. The process for the empirical estimation of a statistic distribution is detailed at the end of Appendix 1.

Figure 3 presents the estimated p-values and highlights the result of KS-test process.

In our study we choose the threshold $\alpha = 0.1$. This choice is supported by the jump in p-values observed around this threshold: the p-value of X_{24} ($p^{KS}_{24} = 0.075$) is just followed by the one of X_{18} with ($p^{KS}_{18} = 0.159$). Two groups of inputs are clearly identified: a first one made up of 13 variables for which the uniform distribution hypothesis is rejected ($p < 0.1$) and a second one with the 37 remaining inputs for which the test cannot reject the hypothesis of a uniform distribution.

In conclusion, a first method has been introduced here to study the direct implication of each input in code failures. It has been done by comparing the marginal empirical distribution of $\boldsymbol{x}_{A,j}$ to the theoretical distribution of $\boldsymbol{x}_j$ (uniform). This method allows to identify which variables were most likely to have an impact on code failure.

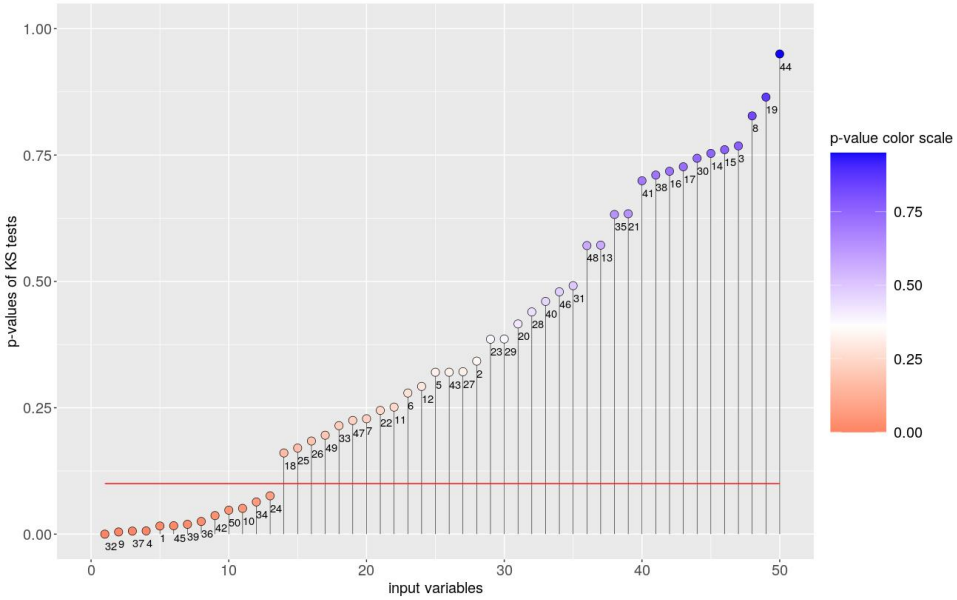

Fig. 3. *Ordered p-values of the KS-tests performed for each of the* $d = 50$ *inputs.*

V. SENSITIVITY ANALYSIS OF CODE FAILURES BASED ON DEPENDENCE MEASURES

In this section, we present another metric, based on an dependence measure, to study the global influence of each input on code failures. The idea is now to study the dependence between the random variable $(\mathrm{X}_j)_{j\in[\![1,d]\!]}$ and the binary random variable Z characterizing the code failures. This enables in particular the detection of variables that only have an influence on code failure, alone or in interaction with other variables.

More precisely, this method relies on the Hilbert-Schmidt Independence Criterion (HSIC) [18] between each of the inputs X_j and the output Z. It is inspired from the work on target HSIC proposed by [19]. The next subsection gives a quick presentation of this criterion and the associated test. It also discuss how it is applied to quantify the impact of the input parameters on the code failure. A more precise description of this criterion and its associated properties are given in the Appendix 2.

V.A. HSIC-based independence test between each inputs X_j and the code failures

The HSIC, proposed by [9], is a criterion that detects the dependence between two random variables. By applying it to the elements of the vector inputs $\mathbf{X}$ and the output characterizing the failures of the code Z, we may detect on which inputs the code failures depend. This criterion is based on a kernel approach. More precisely, it is based on cross-covariance operators in Reproducing Kernel Hilbert Spaces (RKHS) [9]. RKHS are composed of mapping functions (features) and characterized by positive definite kernel function. For a random variable X and the output Z, the HSIC is defined by:

$$HSIC(\mathrm{X},\mathrm{Z})_{\mathcal{F},\mathcal{G}} = \sum_{p,q} \mathbb{C}\mathrm{ov}(u_p(\mathrm{X}), v_q(\mathrm{Z}))^2. \tag{4}$$

Here, $\mathcal{F}$ and $\mathcal{G}$ are the two RKHS associated to X and Z respectively. The functions $(u_p), p \geq 0$ and $(v_q), q \geq 0$ are some orthonormal base associated to $\mathcal{F}$ and $\mathcal{G}$, respectively. $\mathbb{C}$ov denotes the covariance operator. Assuming specific conditions detailed in Appendix 2, we have the following property:

$$HSIC(\mathrm{X}, \mathrm{Z})_{\mathcal{F},\mathcal{G}} = 0 \Leftrightarrow \mathrm{X} \perp\!\!\!\perp \mathrm{Z}. \tag{5}$$

Here, $\perp\!\!\!\perp$ means independence. Thus, the HSIC allows to fully characterize the dependence between X and Z.

Notice that we only have at hand a n realizations of (X, Z). So that, we can only compute an estimation of the HSIC. This estimation is denoted by $\widehat{HSIC}(\mathrm{X}, \mathrm{Z})$. An asymptotically unbiased estimator proposed by [9] is given by Eq. (10) in Appendix 2. From this estimation, an independence test based on the statistic $S_n^{HSIC}(\mathrm{X}, \mathrm{Z}) = n \times \widehat{HSIC}(\mathrm{X}, \mathrm{Z})$ can be suited. Using this statistic and the equivalence (5), the following statistical hypothesis is settled:

$$\text{“}H_0\text{: } HSIC(\mathrm{X}, \mathrm{Z})_{\mathcal{F},\mathcal{G}} = 0\text{” against “}H_1\text{: } HSIC(\mathrm{X}, \mathrm{Z})_{\mathcal{F},\mathcal{G}} > 0\text{”}.$$

As previously, the idea is to process the HSIC test for each inputs $(\mathrm{X}_j)_{j \in [\![1,d]\!]}$ through the comparison of the observed quantity $s_{n,obs}^{HSIC}(\boldsymbol{x}_j, \boldsymbol{Z})$ and the probability distribution of the reference statistic under the null hypothesis $S_n^{HSIC}(\mathrm{X_j}, \mathrm{Z})$. For this, we compute the p-value : $p^{HSIC} = \Pr(S_n^{HSIC} > s_{n,obs}^{HSIC} | H_0)$. 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.

V.B. Results for the case study

We apply now the HSIC-based independence test method to the case study. Recall that in our application, the design of experiment $\boldsymbol{X}$ has been performed using the Latin hypercube sampling method. However, as for the KS test, the HSIC test procedure theory (presented in details in Appendix 2) has been initially built upon the hypothesis of i.i.d samplings of the inputs. Nevertheless, a recent numerical study have shown that HSIC test results remain relevant under Latin Hypercube Sampling. See [20] for more details. The classic procedure can then be reasonably applied.

Figure 4 shows the obtained results. For reasons similar to those for the KS test, a threshold $\alpha = 0.1$ is chosen, to screen the significantly lowest p-values for which the independence hypothesis is rejected. Thus, 12 inputs are selected. These are the inputs most likely to have a direct impact on code failures regarding the HSIC tests.

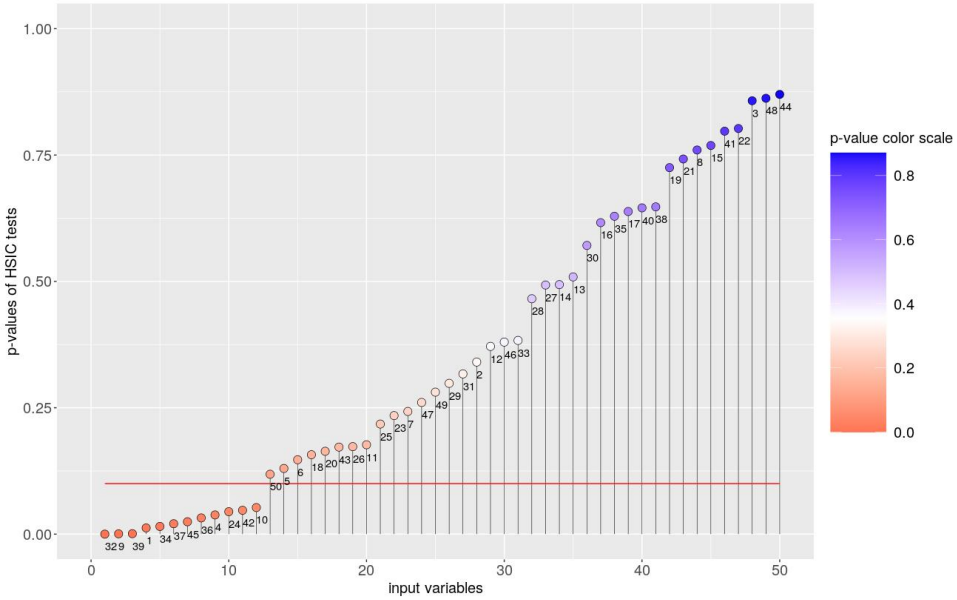

Fig. 4. *Ordered p-values associated to HSIC tests of independence performed for each of the* $d = 50$ *inputs.*

VI. APPLICATION TO OUR CASE STUDY: GRAPHICAL ANALYSIS AND PHYSICAL INTERPRETATION OF THE RESULTS

In this section, the use of the results from the presented methods for the study of the MC3D code is given. Table I summarizes these results. It presents the set of variables selected by at least one of the two test methods. We notice that 12 of the 13 variables presented have been selected by both methods. Only the input X_{50} is significant only for the KS test method but not the HSIC one. This input is just above the threshold for the HSIC test method. Confidence in this variable selection is thus enhanced by the consistency between the results of the two test procedures.

TABLE I
Summary of tests results.

input	p-value KS test	p-value HSIC test
X_{32}	0.000	0.000
X_9	0.004	0.001
X_{37}	0.006	0.021
X_4	0.007	0.038
X_1	0.016	0.012
X_{45}	0.017	0.024
X_{39}	0.020	0.001
X_{36}	0.025	0.032
X_{42}	0.036	0.047
X_{50}	0.047	0.12
X_{10}	0.051	0.052
X_{34}	0.064	0.015
X_{24}	0.075	0.234

To go further in the analysis, the estimated density of $\boldsymbol{x}_{A,j}$ (marginal sample such as the code fails) in comparison to the $\boldsymbol{x}_j$ estimated densities is plotted in Figure 5 for the variables with the lowest p-values. The densities are estimated with a kernel-based non parametric approach. For more information about this method of estimation, one can refer to [21, 22]. Note that since the input DoE is a LHS, the estimated density of the $\boldsymbol{x}_j$ look very close to the theoretical uniform distribution. Figure 5 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.

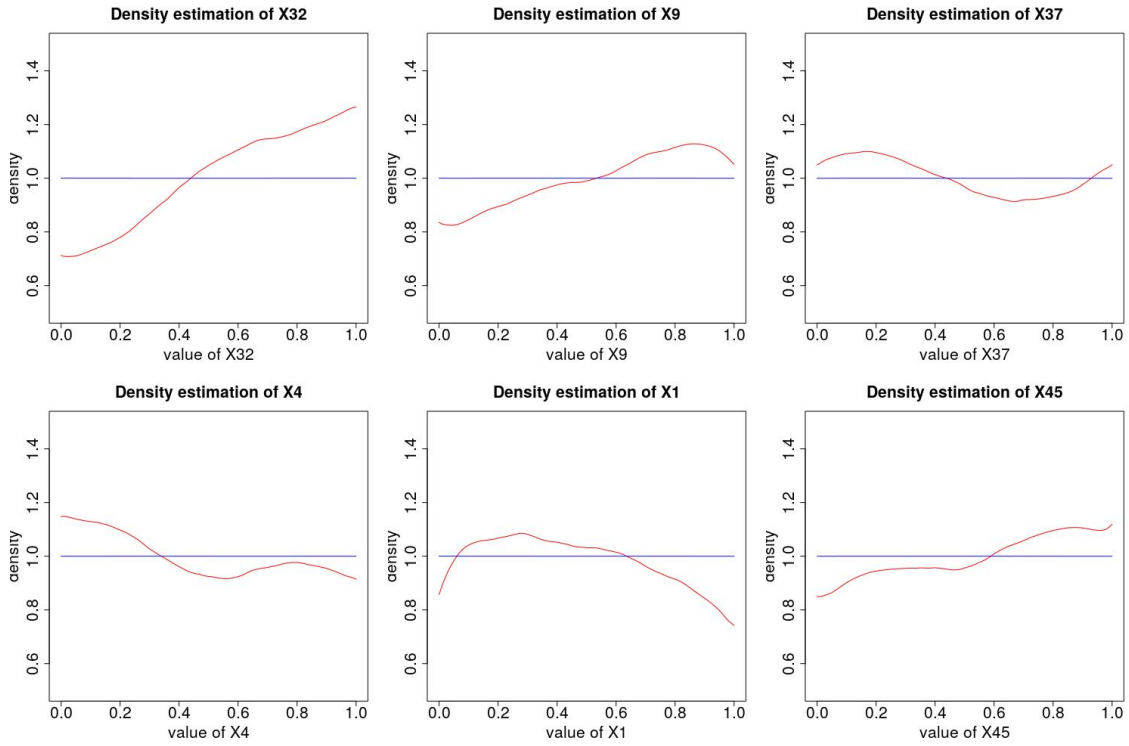

Fig. 5. *Empirical density plots of the six inputs having the lowest p-values regarding the KS test.*

Let focus on the input X_{32}, corresponding to the variable MULTEH in the simulation code MC3D. This is the most likely to have an impact on code failure regarding both test methods (cf. Table I). MULTEH is a multiplicative parameter of the heat transfer coefficient calculated by the Epstein-Hauser correlation [23], between the fuel droplets and the interface between the vapor film surrounding the droplets and the coolant (Figure 6). It has a direct influence on the heat transfer from the droplets to the film interface. Therefore, the vaporization rate, which is determined from the energy balance of the interface, directly depends on the parameter MULTEH. The bigger this parameter is, the bigger vaporization rate of the water is. 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 with the parameter MULTEH, as can be seen on Figure 5.

X_1, named KELMHOLTZ_FRAGNUM in MC3D, is also an input that has a high probability to have an impact on code failures (cf Table I). It is a multiplicative coefficient of the fragmentation rate of the corium jet. Among the two jet fragmentation models available in MC3D premixing

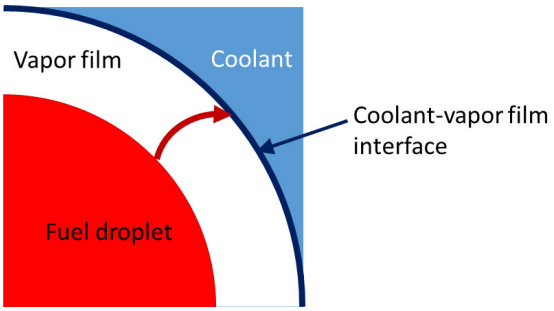

Fig. 6. *Model of the heat exchange from the fuel droplet to the interface between the coolant and the vapor film, (red arrow).*

application, we have used for this study the one based on the Kelvin-Helmholtz fragmentation mechanism. In practice, this model is an extension of the Kelvin-Helmholtz instability model to multiphase flows [24]. In this model, the jet fragmentation rate and the diameter of the generated fuel droplets are respectively proportional to the Kelvin-Helmholtz instability growth velocity and to most amplified wavelength. KELMHOLTZ_FRAGNUM is the multiplicative coefficient between the Kelvin-Helmholtz instability growth velocity and the jet fragmentation rate. The larger this parameter, the higher the fragmentation rate and the higher the mass of drops created. A larger amount of fuel droplets means a higher vaporization of the water, as for MULTEH. But, contrary to MULTEH, the probability of code failure decreases for the highest values of KELMHOLTZ_FRAGNUM. This study thus highlight an unexpected result, inconsistent with physical interpretation. This point will need to be further investigated.

VII. CONCLUSION AND PROSPECTS

Code failures are common problems in the context of numerical simulation of complex physical phenomena such as severe accidents in a nuclear reactor. To improve the robustness of these simulation codes, it is useful to understand which inputs are involved in code failures and how they are involved. In this context and 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 (code failures) 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 computationally expensive simulation codes, the number of possible simulations is limited. This is also a major drawback to the application of usual sensitivity analysis methods. The large number of inputs to be taken into account for our application case (around 50) makes the use of methods whose cost depends on the dimension of the inputs even more difficult.

Within this framework, we introduced two methods to detect which input variables of a code 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 a goodness-of-fit test based on the Kolmogorov statistic. This test method allowed to detect the variables for which the distribution as code fails is different from the initial distribution. 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 to detect all the possible dependency between the input variables and the failures of the code. These methods have the advantage of being usable for general design of experiments. They allow, with a small additional cost, to have a first idea of the variables involved

in the code failure.

The development of these methods was motivated by the study of the simulation code MC3D. This code models the fuel-coolant interaction that can lead to the steam explosion phenomenon. In the context of the exploration of the input space of the code, several code run (2000) have been carried out. Among these, more than a third of them failed. We aimed to understand the origin of these code failures. Thus, Section 6 presents an example of the use of the presented tools. Indeed the two presented methods were used to select a set of variables which were considered significant regarding code failures of MC3D. A graphical analysis of the selected densities such as the code fails has been proceeded. A physical interpretation of the role of some of the significant variables regarding these failures has also been given.

Some extensions of this work can be considered. For instance, it might be interesting to perform a more detailed analysis of code failures focusing the analysis on the 12 selected inputs. We could for example extend our work in the general case of categorical output, in order to distinguish the different causes of code failures in the analysis. Another option could be to study which inputs interact in order to cause code failures. To do so, a good idea could be to use kernel based method in a similar way to the second method presented.

Acknowledgments

Support from the ANR-3IA Artificial and Natural Intelligence Toulouse Institute is gratefully acknowledged.

APPENDIX 1: KOLMOGOROV-SMIRNOV STATISTIC AND ITS ASSOCIATED TEST

This section is devoted to the formal description of the Kolmogorov statistic and the associated Kolmogorov-Smirnov test procedure. These tools are used for the method presented in section IV.

The Kolmogorov statistic

Let F be the distribution function associated the 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 characteristic function of A. Thus, the empirical distribution of X is $F_n(t) = \sum_{i=1}^{n} \mathbf{1}_{x^{(i)}<t | t\in\mathbb{R}}$.

A distance between the empirical distribution F_n and F can be then defined as follow:

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

Note that $D_n^{KS}(F_n, F) \rightarrow_{n\rightarrow\infty} 0$ almost surely, according to the Glivenko-Cantelli theorem.

Let $\left(x^{[1]}, \ldots, x^{[n]}\right)$ be the ordered realizations of $\boldsymbol{x}$. We have the following equality:

$$D_n^{KS}(F_n, F) = \max_{1\leq i\leq n}[\max(|i/n - F(x^{[i]})|, |F(x^{[i]}) - (i-1)/n|)] \tag{6}$$

This property allow to compute easily the quantity $D_n^{KS}(F_n, F)$ in practice.

The Kolmogorov-Smirnov (KS) statistic corresponds to the quantity $S_n^{KS} = n^{1/2} D_n^{KS}(F_n, F)$. Two important properties about the KS statistic S_n^{KS} has been shown (see [16] for instance):

(i) $\left(x^{(1)}, \ldots, x^{(n)}\right)$, S_n^{KS} converges in law to K when $n \to \infty$, K being a known law such with a distribution function defined by

$$F_K(t) = P(K \leq t) = 1 + 2\sum_{k=1}^{+\infty} (-1)^k e^{-2k^2 t^2}.$$

Notice that the distribution $F_K(t)$ does not depend neither in F_n nor in F.

(ii) If F_0 is continuous distribution function such as $F \neq F_0$, the quantity $S_n^{KS}(F_n, F_0) \to_{n\to\infty} \infty$ in probability.

The Kolmogorov-Smirnov test procedure

The objective of Kolmogorov-Smirnov test is to compare the cumulative distribution F associated to $\boldsymbol{x}$ with a reference probability distribution F_0.

The test hypotheses are:

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

Given the results presented above, the statistic considered for this test is $S_n^{KS}(F_n, F_0)$. 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. It is then compared to the theoretical distribution K associated to the Kolmogorov statistic under H_0 . Since we have S_n^{KS} converging in law to K under H_0 and $S_n^{KS}(F_n, F_0) \to_{n\to\infty} +\infty$ in probability under H_1, the critical region $\{K > s_n^{KS}\}$ can be associated to the test. Thus, the corresponding p-value is defined by the quantity $P(K > s_n^{KS} | H_0)$.

Estimating a statistic under the null hypothesis using non-iid samples

In many practical cases, one use quasi-Monte Carlo methods to sample our input space (instead of pure Monte Carlo sampling). This is useful when the input space dimension is high and the code is computationally expensive. These methods have the advantage of better covering the input space, and thus generally give a better idea of the response of the code. This is why we used a Latin Hyper Cube Sampling (LHS) method to build our numerical design in our practical case. Nevertheless, most of the theoretical results concerning the convergence of known statistics (such as Kolmogorov statistic) are only true for *independent and identically distributed* (i.i.d) samples. The quasi-Monte Carlo samples in general (and LHS samples in particular) are not i.i.d. Thus, one cannot use these theoretical results tailored for i.i.d variables directly in practice, unless theoretical results have been specifically demonstrated for these sampling methods and/or the estimated statistic.

The natural solution to address this problem is to simulate samples under the null hypothesis H_0 in order to get the empirical cumulative distribution of the statistic under H_0. Then we use it to estimate the p-value associated to the test we want to proceed.

To do so in our case, we simulate L times independently the samples we could have under the null hypothesis: a LHS of size n and dimension 1 that corresponds to $x_{j,s}$ and an n size vector of independent Bernoulli draws following the distributions $\mathcal{B}(l_{|A|}/n)$. This vector corresponds

to a size n vector of outputs 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 the formula (6). This allows us to get the empirical cumulative distribution $F_{S^{KS|H_0}}(t) = \sum_{l=1}^{L} \mathbf{1}_{S_n^{KS|H_0,(l)}<t}$. Note that the higher is L, the better is the estimation of the cumulative distribution. We then estimate, for each vector samplings $(\boldsymbol{x_{A,j}})_{j\in\{1,..d\}}$, the statistic $s_{n,obs}^{KS}(F_{j,n}, F_{U_{[0,1]}})$. Here, $F_{j,n}(t)$ is the empirical distribution function of $\boldsymbol{x_{A,j}}$ and $F_{U_{[0,1]}}$ is the uniform cumulative distribution function. Finally and for each of the d statistic, we estimate the p-value associated to the test $p^{KS} = 1 - F_{S_n^{KS|H_0}}(s_{n,obs}^{KS}(F_{j,n}, F_{U_{[0,1]}}))$. The Algorithm 1 summarizes the whole process.

Algorithm 1 conditional to LHS KS P-values

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

1: **for** $l = 1$ to L **do**
2: Generate an LHS sample of size n and dimension 1 to simulate a sample of type x_j
3: A sample of size n is then independently drawn according to Bernoulli distribution $\mathcal{B}(l_{|A|}/n)$ that corresponds to a sample of the output Z under the null hypothesis
4: A replica under the null hypothesis of $x_A|H_0$ is deduced form this drawn
5: Compute the observed Kolmogorov statistic $S_n^{KS|H_0} = n^{1/2} D_n^{KS}(F_{x_A|H_0}, F_{U_{[0,1]}})$, $F_{x_A|H_0}$ being the empirical cumulative distribution of $x_A|H_0$
6: **end for**
7: An empirical distribution $F_{S_n^{KS|H_0}}(t) = \sum_{l=1}^{L} \mathbf{1}_{S_n^{KS|H_0,(l)}<t}$ of the KS statistic under H_0 over the obtained L observed Kolmogorov statistics.
8: Compute the d observed Kolmogorov statistic on the $\boldsymbol{x_{A,j}}$ 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 $F_{s_{n,obs}^{KS}}$

APPENDIX 2: HILBERT-SCHMIDT INDEPENDENCE CRITERION AND ITS ASSOCIATED TEST

This section is devoted to a formal description of the context related to the Hilbert-Schmidt independence criterion (HSIC). We focus on its main properties and the associated test procedure. These tools are used for method presented in section V.

Presentation of the criterion the associated key concepts

The Hilbert-Schmidt Independence Criterion (HSIC), proposed in [9], aims to detects the dependence between two random variables. This criterion is based on a kernel approach. More precisely, it is based on cross-covariance operators that generalizes the notion of the classical covariance [9]

First, let define the covariance. For any real-valued random variable X and Y with finite second moments, the covariance is the expected value of the product of their deviations from their individual expected values: $\mathbb{C}\text{ov}(X,Y) = \mathbb{E}[(X - \mathbb{E}(X))(Y - \mathbb{E}(Y))]$. An important property about the covariance is that if two random variables X and Y are independent, then $\mathbb{C}\text{ov}(X,Y) = 0$. The reciprocal statement is not true in general. The main idea of the HSIC is to generalize the concept of covariance in order to get the reciprocal statement.

Before defining the HSIC, we need to introduce the concepts of Reproducing *Kernel Hilbert Spaces* (RKHS), *characteristic kernels* and *generalized cross-covariance operator*. Let $\mathcal{F}$ be an Hilbert space (a complete vector space equipped with an inner product $\langle , \rangle_{\mathcal{F}}$) of functions of $\mathcal{X}$ in $\mathbb{R}$. $\mathcal{F}$ 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. $\forall f \in \mathcal{F}, x \in \mathcal{X}, \langle f, k(x, .)\rangle = f(x)$ (reproducing property).

This function k is called the reproducing kernel of $\mathcal{G}$. Roughly speaking, we can consider the function k as a function that quantifies the similarity between two elements of $\mathcal{F}$.

Let $\mathcal{F}$ (resp. $\mathcal{G}$) be a RKHS associated to X (resp. 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 be defined as the operator mapping from $\mathcal{F}$ to $\mathcal{G}$ and verifying for all $(f, g) \in \mathcal{F} \times \mathcal{G}$:

$$\langle f, C_{X,Y}(g)\rangle_{\mathcal{F}} = \mathbb{C}\text{ov}(f(X), g(Y)) \tag{7}$$

It generalizes the notion of the covariance between X and Y so it can detect a larger amount of dependence between them.

To summarize the information provided by $C_{X,Y}$, we consider its Hilbert-Schmidt norm the HSIC , as stated by Eq. (8).

$$HSIC(X,Y)_{\mathcal{F},\mathcal{G}} = \|C_{X,Y}\|^2 = \sum_{p,q} \langle u_p, C_{X,Y}(v_q)\rangle_{\mathcal{F}} = \sum_{p,q} \mathbb{C}\text{ov}(u_p(X), v_q(Y))^2. \tag{8}$$

Here, $(u_p), p \geq 0$ and $(v_q), q \geq 0$ are somme orthonormal bases of $\mathcal{F}$ and $\mathcal{G}$ respectively. An interesting fact about the HSIC is that we have an equivalence between these two propositions:

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

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 [25] for instance).

Thus, for suitable RHKS $\mathcal{F}$ and $\mathcal{G}$, we get the equivalence between the nullity of $HSIC(X, Z)_{\mathcal{F},\mathcal{G}}$ and the independence between X and Y. In this case, the HSIC allows to generalize the notion of covariance in the sense presented at the beginning of this section.

A sufficient condition so that nullity of the HSIC characterizes the independence is the use of characteristic kernels. A kernel k is characteristic of $\mathcal{F}$ if, for all probability measures $\mathbb{P}$ defined on $\mathcal{F}$, the function $\mathbb{P} \rightarrow \int k(., x)d\mathbb{P}(x)$ is injective. For more details about characteristic kernels, one refer to [26, 27].

For real variables, the most used characteristic kernel is the Gaussian kernel. It is defined, for $(x, x') \in \mathbb{R} \times \mathbb{R}$, by:

$k(x, x') = \exp(-\frac{\lambda}{2}(x - x')^2)$, and characterized by the bandwidth parameter λ. This parameter is often set at $\lambda = 1/\sigma^2$ with σ^2 being the empirical variance of the samples x (resp. x'). The inputs in our case study, $(X_j)_{j \in [\![1,d]\!]}$, being defined in $[0, 1]^d$, they have been associated to the Gaussian kernel.

For categorical variables, the Dirac kernel can be appropriate. Let $(y, y') \in \{0, ...K\}^2$ be two categorical variables and K the number of categories. The dirac kernel is defined by: $l(y, y') = \delta_{y,y'}/n_y$. Here, δ represents the Dirac measure and n_y the number of sample in the same category as y. In our application, it corresponds to the characteristic kernel used for the output Z. In our case, $K = 1$ and n_y corresponds either to the number of code failures ($n_A = 700$) or to successful calculations $n - n_A = 1300$.

Using RKHS properties, Gretton [9] has shown that HSIC can also be expressed in a convenient form using kernels (9):

$$HSIC(X, Y)_{\mathcal{F},\mathcal{G}} = \mathbb{E}[k(X, X')l(Y, Y')] + \mathbb{E}[k(X, X')]\mathbb{E}[l(Y, Y')] - 2\mathbb{E}[\mathbb{E}[k(X, X')]\mathbb{E}[l(Y, Y')]] \quad (9)$$

with (X', Y') being *independent and identically distributed* (i.i.d) copies of (X, Y).

Estimation of the Hilbert-Schmidt Independence Criterion

In practice, the computation of HSIC is proceed using Eq. (9). For instance, Gretton [9] proposed an estimator using this equation. Let $\{(x^{(1)}, y^{(1)}), ..., (x^{(n)}, y^{(n)})\} \subseteq \mathcal{X} \times \mathcal{Y}$ be a series of n independent copies of (X, Y). According to [9], an HSIC estimator is given by the following formula:

$$\widehat{HSIC}(X, Y) = \frac{1}{(n-1)^2}\text{tr}(\boldsymbol{KHLH}) \quad (10)$$

with:

- $\boldsymbol{H}, \boldsymbol{K}, \boldsymbol{L}$ three matrices in $\mathbb{R}^{n\times n}$, such as: $K^{(i',j')} = k(x^{(i')}, x^{(j')})$, $L^{(i'j)'} = l(y^{(i')}, y^{(j')})$ and $H^{(i',j')} = \delta^{(i',j')} - \frac{1}{n}$, $\delta^{(i',j')}$ being the dirac distribution.

- tr the trace application, such as for any $A \in \mathbb{R}^{n\times n}$, $\text{tr}(A) = \sum_{i'=1}^{n} A^{(i'i')}$.

This estimator is asymptotically unbiased [9].

HSIC statistic and test procedure

A statistical independence test can be built on the fundamental property of the HSIC to test the independence between two variables. For a given variable X and other variable Z, it aims at testing the null hypothesis H_0: "The input X and the output Z are independent" against the alternative hypothesis H_1: "There is a dependency structure between the input X and the output Z". 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 test. If X and Z are independent and under asymptotic convergence (i.e if n is large enough), it has been proved that the law of S_n^{HSIC} can be asymptotically approached by a *Gamma distribution* with a shape parameter and a scale parameter. The interested reader is refer of [18] for more details.

As usual, the p-value drives the decision process. 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:

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

REFERENCES

[1] T. K. Ho, Random decision forests, in: Proceedings of the Third International Conference on Document Analysis and Recognition - Volume 1, IEEE Computer Society, USA, 1995, p. 278.

[2] D. W. Hosmer, S. Lemeshow, Applied logistic regression, John Wiley and Sons, (2000).

[3] C. Cortes, V. Vapnik, Support-vector networks, Machine learning 20 (3) (1995) 273–297.

[4] B. Iooss, P. Lemaître, A review on global sensitivity analysis methods, in: Uncertainty Management in Simulation-Optimization of Complex Systems, Springer US, 2015, pp. 101–122.

[5] M. Sobol', Sensitivity Estimates for Nonlinear Mathematical Models(english translation), Mathematical Modeling and Computational Experiment (1993) 407–414.

[6] M. D. Morris, Factorial Sampling Plans for Preliminary Computational Experiments, Technometrics 33 (2) (1991) 161–174.

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

[8] S. Da Veiga, Global sensitivity analysis with dependence measures, Journal of Statistical Computation and Simulation 85 (7) (2015).

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

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

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

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

[13] C. Brayer, A. Charton, D. Grishchenko, P. Fouquart, Y. Bullado, F. compagnon, P. Correggio, N. Cassiaut-Louis, P. Piluso, Analysis of the KROTOS KFC test by coupling x-ray image analysis and MC3d calculations, june 24, in: Proceedings of ICAPP '12, American Nuclear Society (ANS), Chicago, IL, USA, (2012), pp. 1854–1863.

[14] V. Bouyer, N. Cassiaut-Louis, P. Fouquart, P. Piluso, Plinius prototypic corium experimental platform, in: Proceedings of Nureth-16, Chicago, IL, USA,September 30, (2015), pp. 5327–5340.

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

[16] A. L. Kolmogorov, Sulla determinazione empirica di una legge di distribuzione, G. Ist. Ital. Attuari 4 (1933) 83–91.

[17] N. Smirnov, Table for Estimating the Goodness of Fit of Empirical Distributions, The Annals of Mathematical Statistics 19 (2) (1948) 279 – 281.

[18] A. Gretton, K. Fukumizu, C. Teo, L. Song, B. Schölkopf, A. Smola, A kernel statistical test of independence, in: J. Platt, D. Koller, Y. Singer, S. Roweis (Eds.), Advances in Neural Information Processing Systems, Vol. 20, Curran Associates, Inc., 2008, pp. 585–592.

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

[20] M. R. El Amri, A. Marrel, More powerful HSIC-based independence tests, extension to space-filling designs and functional data, preprint (Oct. 2021).
URL https://hal-cea.archives-ouvertes.fr/cea-03406956/file/More_robust_powerful_HSIC_based_independence_tests_and_extension_to_Space_Filling_Designs.pdf

[21] M. Rosenblatt, Remarks on Some Nonparametric Estimates of a Density Function, The Annals of Mathematical Statistics 27 (3) (1956) 832 – 837.

[22] E. Parzen, On estimation of a probability density function and mode, The Annals of Mathematical Statistics 33 (3) (1962) 1065–1076.

[23] M. Epstein, G. M. Hauser, Subcooled forced-convection film boiling in the forward stagnation region of a sphere or cylinder, International Journal of Heat and Mass Transfer 23 (2) (1980) 179–189.

[24] D. Fletcher, A review of hydrodynamic instabilities and their relevance to mixing in molten fuel coolant interactions (1984).

[25] J. Jacod, P. Protter, Probability Essentials, 1st Edition, Springer Berlin Heidelberg, 2004.

[26] K. Fukumizu, A. Gretton, X. Sun, B. Schölkopf, Kernel measures of conditional dependence, in: J. Platt, D. Koller, Y. Singer, S. Roweis (Eds.), Advances in Neural Information Processing Systems, Vol. 20, Curran Associates, Inc., Red Hook, NY, USA, 2008, p. 489–496.

[27] B. Sriperumbudur, K. Fukumizu, G. Lanckriet, On the relation between universality, characteristic kernels and rkhs embedding of measures, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, PMLR, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010, pp. 773–780.