Research report

Field of Study: Mechanical Engineering
Scholar Year : 2023-2024

Non-Intrusive Polynomial Chaos Expansion Applied to Full-Order Stochastic CFD

Authors: Raphael ALVES HAILER, Yani AÏT AMMAR

École Polytechnique Tutor: Pascal CHABERT
Ariane Group Tutor: Mohamed BOUARFA

Abstract

In the present work, we delve into the application of the Polynomial Chaos Expansion (PCE) method for undertaking uncertainty quantification in stochastic Computational Fluid Dynamics (CFD) simulations. These simulations are performed using OpenFOAM, a comprehensive open-source solver. The study specifically concentrates on two well-established benchmark problems within the CFD domain: the lid-driven cavity problem and the dynamic airflow analysis around a NACA0012 airfoil. Both problems were chosen due to the amount of available literature for results comparison and discussion. By integrating the PCE method into stochastic CFD analyses, we aim to showcase its effectiveness as an uncertainty quantification tool. This approach stands out for its potential to bypass the high computational costs associated with traditional Monte Carlo methods in statistical evaluations.

Keywords— Uncertainty quantification, Surrogate model, Predictive model, Polynomial chaos, Stochastic, CFD, OpenFOAM, Full-Order model, NACA0012, Lid-driven cavity problem.

Résumé

Dans ce travail, nous explorons l'application de la méthode d'Expansion du Chaos Polynomial pour réaliser une quantification d'incertitudes dans des simulations numériques de mécanique des fluides. Ces simulations sont effectuées à l'aide d'OpenFOAM, un solveur open-source complet. L'étude se concentre spécifiquement sur deu problèmes de référence bien établis dans le domaine de la CFD : celui de la cavité entraînée par un couvercle, et la caractérisation dynamique de l'écoulement de l'air autour d'un profil NACA0012. Ces deux problèmes ont été choisis en raison de la quantité importante de littérature disponible pour la comparaison et la discussion des résultats. En intégrant cette méthode dans les analyses stochastiques de CFD, nous visons à démontrer son efficacité en tant qu'outil de quantification d'incertitudes. Cette approche se distingue par ses faibles coûts computationnels par rapport aux méthodes traditionnelles de Monte Carlo dans les évaluations statistiques.

Mots clés— Quantification d'incertitudes, Modèle de substitution, Modèle prédictif, Chaos polynomial, Stochastique, CFD, OpenFOAM, Modèle d'ordre complet, NACA0012, Cavité entraînée

Contents

List of Figures

List of Tables

1 Introduction

The use of **Computational Fluid Dynamics** (CFD) in the scientific community and the industry is in growing demand, as it has become a critical piece of every major engineering project. In fact, CFD has morphed from a mere analytical tool into an indispensable instrument for not only validating qualitative outcomes, but also quantifying results. This is particularly the case when traditional experimental studies prove either impractical or prohibitively costly.

However, most of the CFD studies are conducted using a deterministic approach, thereby providing a single solution for a given set of input parameters. This poses a significant problem when attempting to evaluate situations where the results are highly sensitive to input data, which may be uncertain in practice. This is particularly relevant in the field of aerospace engineering, where dynamic, non-stationary factors play a critical role. In scenarios like the launch of a spacecraft, the system encounters a variety of complex aerodynamic phenomena, such as buffeting, shock waves and boundary layer separation. These events are highly unpredictable, making accurate assessment and prediction a formidable task that demands precise and sensitive analysis of the input data. Consequently, it is necessary to use **Uncertainty Quantification** (UQ) methods in order to characterize the effects of input uncertainties on the output variables of interest, obtained by the CFD simulations. Consider, for example, the importance of accounting for input parameters uncertainties in simulations used for modeling critical mechanical system components. This practice is essential for establishing confidence intervals that accurately reflect potential variations in output quantities of interest. However, conducting numerous simulations with varied input values to conduct a statistical study, as one would with the Monte-Carlo method, can be challenging as complex problems often require substantial computational resources. For instance, one single simulation of a space launch vehicle take-off can take several days, even with hundreds of CPUs working in parallel. This is where UQ methods prove valuable, as they offer cost-effective approximations of the relationship between uncertain input variables and resulting outcomes.

Among the several methods developed for uncertainty propagation in computational simulations, there are the deterministic ones (interval analysis, propagation of error using sensitivity derivatives) and the probabilistic ones (Monte-Carlo, moment methods, polynomial chaos expansion) [3]. In this work, we focus on the **Polynomial Chaos Expansion** (PCE) method, and more specifically on the **Non-Intrusive** (NIPCE) method.

The primary objective of the present work is to implement the NIPCE method using Python, as there is a vast collection of libraries that are able to implement UQ methods, including NIPCE. Some noteworthy examples include ChaosPy [4], OpenTurns [5], UQpy [6] and Dakota [7]. For this project, the method's implementation will be carried out using the Chaospy library. This choice is attributed to its specialization in the PCE method, providing a more user-friendly environment for ease of use and understanding. Initially, the project seeks to validate its efficiency through testing on benchmark problems in CFD. Two of them are going to be addressed under the UQ scope, using the NIPCE method, and the results will be compared against the deterministic outcomes obtained using a **full-order Model** (FOM) solver. In this work, the FOM solver is OpenFOAM [8] as it is a greatly reliable open-source software.

Another option for validation could involve the use of Monte-Carlo methods, appreciated for their ease of implementation and result quality. However, their feasibility is hindered by the substantial computation cost they require, especially when dealing with CFD simulations.

In the following section, we will discuss the basics of NIPCE, focusing on the Point-Collocation technique [9, 10], which will be the one used in this study for propagating uncertainty in stochastic simulations. Furthermore, in Section 3, two widely known benchmark problems are going to be tackled using the NIPCE method, and the results will be compared with the ones obtained in OpenFOAM. The problems to be studied are (1) the lid-driven cavity flow problem and (2) the dynamical characterization of airflow around a NACA0012 airfoil.

2 Introduction to the Non-Intrusive Polynomial Chaos Expansion method

2.1 Uncertainty Quantification

When dealing with uncertainty quantification, two sources of input parameter uncertainty can be distinguished: aleatoric and epistemic uncertainties. While the latter refer to uncertainties caused by a lack of knowledge, and therefore can in principle be reduced on the basis of additional information; aleatoric ones relate to the inherent randomness within a data set. For example, when studying an airfoil in a flow, one should take into account the operational environment, as variations in wind speed and direction, especially sudden gusts or turbulent flow conditions, can introduce aleatoric uncertainties. As a result, the Mach number and angle of attack of the flow become uncertain input parameters to the problem, whose impact on the ouput parameters such as the aerodynamic coefficients of the airfoil, must be assessed. In this study, we focus on the aleatoric -also referred to as statistical- uncertainties. For this particular type of uncertainty, typically, there is enough data to use probabilistic methods in order to compute response distribution statistics based on the **Probability Density Function** (PDF) of each input [2].

In the realm of uncertainty quantification, a variety of methods have been developed and refined to address the complexities inherent in various fields, particularly in engineering and scientific research. Among these, Monte Carlo simulations are widely recognized for their accuracy and straightforward implementation, involving random sampling to approximate probability distributions of uncertain parameters. However, due to their high computational cost, alternative methods have gained popularity. One such method is the Polynomial Chaos Expansion (PCE).It presents a more computationally efficient alternative, and is founded on a spectral representation of the uncertainty, as the output variable of interest is expanded on a stochastic orthogonal polynomial basis. We end up with a sum of separable deterministic coefficients and orthogonal polynomials that depend on the uncertain input parameters. Stochastic post-processing can then be performed easily, and at reduced cost, on the expansion obtained once the deterministic coefficients have been calculated. This provides statistical information on the output of interest, and its dependence on input uncertainties.

The PCE method [9] has been extensively explored and applied across various domains. An application of the PCE method in stochastic finite elements used for structural problems can be read in [11], while studies of the PCE method being used in the fluid dynamics community can be read in [12] and [13].

The generalized polynomial chaos expansion, also known as the Wiener-Askey polynomial chaos [14], is an intrusive approach that computes the unknown deterministic coefficients by projecting the fluid dynamics equations onto the orthogonal polynomial basis. A study on the application of generalized PCE in stochastic modeling of flow-structure interactions and fluid diffusion can be found in [15] and [16], respectively. However, it is important to note that the intrusive technique requires modifications to the deterministic code of the CFD solver, which may not always be possible, particularly when using commercial CFD software which operate as a black box. In addition, modification of the source code can be a labor-intensive and time-consuming process, especially when dealing with complex computational problems, such as a 3D simulations of viscous and turbulent flows around real vehicles, or multidisciplinary codes for modeling mechanical and chemical evolution of a system. To address these challenges, the NIPCE approach has been developed. It involves post-processing the output data acquired from the deterministic code, which greatly reduces the complexity of implementation. This is generally done using sampling, collocation or quadrature based methods [9] in order to compute the deterministic coefficients of the polynomial expansion.

2.2 Fundamentals

The generalized PCE method is designed to quantify the uncertainties related to a stochastic process, and is based on the spectral representation of the uncertainty. This means that the random output variable of interest obtained by the CFD simulation can be decomposed into a sum of separable

deterministic coefficients and a basis of orthogonal polynomials. These polynomials are the stochastic components, as they depend on the uncertain input parameters. Being the random desired output denoted $\Theta(\vec{x}, \vec{\xi})$, we can write the truncated expansion for approximating Θ as follows:

$$\Theta(\vec{x}, \vec{\xi}\,) \approx \sum_{i=0}^{P} \theta_i(\vec{x})\psi_i(\vec{\xi}\,) \tag{1}$$

In Equation 1, we have that Θ can be a scalar output (the aerodynamic coefficients for instance), but also a vector field, such as the velocity. Thus, in certain cases, dependence on $\vec{x}$, the position vector, is not systematic. $\vec{\xi} = (\xi_1, \xi_2, \ldots, \xi_n)$ is the n-dimensional random variable vector, containing all the random input parameters with a specific PDF to describe them. At last, we denote $\theta_i(\vec{x})$ and $\psi_i(\vec{\xi})$ as being the deterministic components and correspondent polynomial basis functions, which takes into account the randomness of the input parameters. When $n \geq 2$, the polynomials are obtained by the tensor product of $1D$ orthogonal bases. The number of terms in the expansion depends on the the accuracy required, and is given by the following equation:

$$N_t = P + 1 = \frac{(n+p)!}{n!p!} \tag{2}$$

In Equation 2, the total number of terms N_t is intimately linked to the number of random dimensions n and the order of the polynomial chaos p, representing the polynomial expansion's order. To provide a clearer understanding of this relationship, let us consider a simple example. We will focus on a $2D$ random variable vector $\vec{\xi} = (\xi_1, \xi_2)$ while setting the order of the polynomial chaos to $p = 2$. If we denote ϕ_i and ϕ'_i as the respective $1D$ polynomial bases associated with ξ_1 and ξ_2, and indexed by the polynomial order, we find that the expansion consists of six terms:

$$\begin{aligned}
\psi_0(\xi_1, \xi_2) &= \phi_0(\xi_1) \otimes \phi'_0(\xi_2) \\
\psi_1(\xi_1, \xi_2) &= \phi_1(\xi_1) \otimes \phi'_0(\xi_2) \\
\psi_2(\xi_1, \xi_2) &= \phi_0(\xi_1) \otimes \phi'_1(\xi_2) \\
\psi_3(\xi_1, \xi_2) &= \phi_2(\xi_1) \otimes \phi'_0(\xi_2) \\
\psi_4(\xi_1, \xi_2) &= \phi_0(\xi_1) \otimes \phi'_2(\xi_2) \\
\psi_5(\xi_1, \xi_2) &= \phi_1(\xi_1) \otimes \phi'_1(\xi_2)
\end{aligned}$$

In the previous example, distinct polynomial bases were chosen to represent each of the random input parameters. This choice was made based on the existence of an optimal basis for each probability density function associated with a random parameter. For instance, when random parameters follow normal distributions, the optimal polynomial basis consists of Hermite polynomials. This optimality arises from the definition of the orthogonality relatively to the following inner product, which itself depends on the PDF of the random vector $\vec{\xi}$.

For two functions $f(\vec{\xi})$ and $g(\vec{\xi})$ the inner product is defined as:

$$\left\langle f(\vec{\xi}), g(\vec{\xi}) \right\rangle = \int_{\Omega} f(\vec{\xi})g(\vec{\xi})w(\vec{\xi})d\vec{\xi} \tag{3}$$

In Equation 3, it is apparent that w is the weight function, which can be regarded as the PDF of $\vec{\xi}$. Additionally, Ω denotes the support range of the weight function, defined as the interval within which the PDF of $\vec{\xi}$ is non-zero. This insight allows us to deduce that, for each PDF function, a specific polynomial basis can be identified that ensures orthogonality according to the aforementioned inner product definition. Table 1 contains the correspondence between the PDF of the random input parameters and the optimal polynomial basis choice. In this table, the PDFs are in standard forms, which explains the simpler formulations of the PDF and weight function of each distribution.

Distribution	PDF	Polynomial family	Weight function	Support range
Normal	$\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$	Hermite $H_n(x)$	$e^{-\frac{x^2}{2}}$	$[-\infty, +\infty]$
Uniform	$\frac{1}{2}$	Legendre $P_n(x)$	1	$[-1, 1]$
Beta	$\frac{(1-x)^\alpha(1+x)^\beta}{2^{\alpha+\beta+1}B(\alpha+1,\beta+1)}$	Jacobi $P_n^{(\alpha,\beta)}(x)$	$(1-x)^\alpha(1+x)^\beta$	$[-1, 1]$
Exponential	e^{-x}	Laguerre $L_n(x)$	e^{-x}	$[0, +\infty]$
Gamma	$\frac{x^\alpha e^{-x}}{\Gamma(\alpha+1)}$	Generalized Laguerre $L_n^\alpha(x)$	$x^\alpha e^{-x}$	$[0, +\infty]$

Table 1: Correlation between standard forms of continuous probability distributions and the optimal continuous orthogonal polynomial basis. Credits to [2]

In Table 1, one can note the presence of the Gamma function, denoted as Γ, and of the Beta function, represented as $\beta(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$. Additionally, it is important to note that the density and weight functions differ only by a constant factor, a consequence of ensuring that the integral of the PDF over the support range equals one [2]. In practice, we must employ multivariate polynomials for the expansion, since more often than not $n > 1$, as it was illustrated in the example above.

The idea behind the NIPCE method is then to use this orthogonality to obtain numerical approximations of the deterministic coefficients θ_i, $\forall i \in [\![0, P]\!]$, without having to alter the original deterministic solver code. Considering the inner-product defined, we can project Equation 1 onto the k-th basis function and make use of the orthogonality. Thus, after projecting we have:

$$\left\langle \Theta(\vec{x}, \vec{\xi}), \psi_k(\vec{\xi}) \right\rangle = \sum_{i=0}^{P} \left\langle \theta_i(\vec{x})\psi_i(\vec{\xi}), \psi_k(\vec{\xi}) \right\rangle \tag{4}$$

By considering the orthogonality of the polynomial basis, we have the following:

$$\left\langle \Theta(\vec{x}, \vec{\xi}), \psi_k(\vec{\xi}) \right\rangle = \theta_k(\vec{x}) \left\langle \psi_k(\vec{\xi}), \psi_k(\vec{\xi}) \right\rangle \tag{5}$$

Thus, we obtain

$$\theta_k(\vec{x}) = \frac{\left\langle \Theta(\vec{x}, \vec{\xi}), \psi_k(\vec{\xi}) \right\rangle}{\left\langle \psi_k(\vec{\xi}), \psi_k(\vec{\xi}) \right\rangle} \tag{6}$$

2.3 Random input parameters with normal distributions

Although several correspondences have been presented, it is common to consider that the uncertain parameters follow a normal distribution, which facilitates the construction of the optimal basis. It is this particular case that we will describe in detail in this section. Under this assumption, the polynomial chaos expansion can be performed using only multivariate Hermite polynomials. The multivariate Hermite polynomial family is defined as follows, where p indicates the order of the polynomial:

$$H_p(\vec{\xi}) = e^{\frac{1}{2}\vec{\xi}\cdot\vec{\xi}}(-1)^p \nabla^p e^{-\frac{1}{2}\vec{\xi}\cdot\vec{\xi}} \tag{7}$$

For instance, when considering $\vec{\xi} = (\xi_1, \xi_2)$ with an expansion order of two, we have $(n, p) = (2, 2)$, and utilizing Equation 2 we deduce that our truncated expansion comprises a total of six terms. As seen before, there is one zeroth order polynomial, two first order polynomials and three second order polynomials. The computation of all these Hermite polynomials is presented below.

Hermite polynomial with p=0:

$$H_0(\xi_1, \xi_2) = \psi_0(\xi_1, \xi_2) = e^{\frac{1}{2}(\xi_1^2+\xi_2^2)} e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)} = 1 \tag{8}$$

Hermite polynomials with p=1:

$$
\begin{aligned}
H_1^{(1)}(\xi_1,\xi_2) = \psi_1(\xi_1,\xi_2) &= -e^{\frac{1}{2}(\xi_1^2+\xi_2^2)}\frac{\partial}{\partial\xi_1}\left(e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)}\right) = \xi_1 \\
H_1^{(2)}(\xi_1,\xi_2) = \psi_2(\xi_1,\xi_2) &= -e^{\frac{1}{2}(\xi_1^2+\xi_2^2)}\frac{\partial}{\partial\xi_2}\left(e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)}\right) = \xi_2
\end{aligned}
\tag{9}
$$

Hermite polynomials with p=2:

$$
\begin{aligned}
H_2^{(1,1)}(\xi_1,\xi_2) = \psi_3(\xi_1,\xi_2) &= e^{\frac{1}{2}(\xi_1^2+\xi_2^2)}\frac{\partial^2}{\partial\xi_1^2}\left(e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)}\right) = \xi_1^2 - 1 \\
H_2^{(1,2)}(\xi_1,\xi_2) = \psi_4(\xi_1,\xi_2) &= e^{\frac{1}{2}(\xi_1^2+\xi_2^2)}\frac{\partial^2}{\partial\xi_1\partial\xi_2}\left(e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)}\right) = \xi_1\xi_2 \\
H_2^{(2,1)}(\xi_1,\xi_2) = \psi_4(\xi_1,\xi_2) &= e^{\frac{1}{2}(\xi_1^2+\xi_2^2)}\frac{\partial^2}{\partial\xi_2\xi_1}\left(e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)}\right) = \xi_1\xi_2 \\
H_2^{(2,2)}(\xi_1,\xi_2) = \psi_5(\xi_1,\xi_2) &= e^{\frac{1}{2}(\xi_1^2+\xi_2^2)}\frac{\partial^2}{\partial\xi_2^2}\left(e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)}\right) = \xi_2^2 - 1
\end{aligned}
\tag{10}
$$

From the set of Hermite polynomials of second order in 10, it is clear that $H_2^{(1,2)} = H_2^{(2,1)}$ due to the fact that $\frac{\partial^2}{\partial\xi_1\partial\xi_2}\left(e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)}\right) = \frac{\partial^2}{\partial\xi_2\partial\xi_1}\left(e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)}\right)$. This is a direct implication of the Schwarz's Theorem, since $e^{-\frac{1}{2}(\xi_1^2+\xi_2^2)} \in \mathcal{C}^2$, meaning that all the second partial derivatives exist and are continuous. Furthermore, in the polynomial expansion, there is no need to account two times for this polynomial, as the deterministic coefficients of the expansion would just combine to obtain a single term multiplying ψ_4.

If desired, a higher order of Hermite polynomials can be used to improve the accuracy of the approximation. However, this comes at a computational cost. The truncation of the polynomial expansion approximation is a way to reduce the number of terms to a finite amount, and the truncation error, denoted as ϵ and defined by $\epsilon(n,p) = \Theta(\vec{x},\vec{\xi}\,) - \sum_{i=0}^{P}\theta_i(\vec{x})\psi_i(\vec{\xi}\,)$, depends on both n and p. This error itself is a random variable, and the truncated expansion converges in the mean square sense as $(n,p) \to (+\infty,+\infty)$ [17, 18], meaning that

$$
\lim_{\substack{n\to+\infty \\ p\to+\infty}} \left\langle \epsilon^2(n,p) \right\rangle = 0 \tag{11}
$$

Hence, since most stochastic CFD problems involve a fixed number of random dimensions n, the Polynomial Chaos expansion becomes efficient when a relatively small value of p is sufficient to achieve high accuracy for Θ.

When it comes to sampling-based methods, the objective is to compute $\theta(\vec{x})\psi_k(\vec{\xi}\,)$ for all $k \in [\![0,P]\!]$ for a fixed number of samples and averaging it out to estimate the inner product $\left\langle \theta(\vec{x}), \psi_k(\vec{\xi}\,) \right\rangle$. In contrast, quadrature methods compute the inner product (Equation 3) using numerical quadrature. After evaluating this term, each deterministic coefficient can be determined by varying the value of k. However, it is most often very complicated to acquire enough data to perform an accurate averaging of $\Theta(\vec{x})\psi_k(\vec{\xi}\,)$. Additionally, while the quadrature method simplifies the integral computation, it still demands a substantial number of simulations and computations to achieve desirable convergence. An alternative, more efficient approach exists, based on the sampling method —the Point-Collocation Non-Intrusive Polynomial Chaos Expansion method.

2.4 Point-collocation method

For the point-collocation NIPCE approach, Equation 1 will be considered for $N+1$ different random input scenarios, resulting in a linear system that will provide the desired deterministic coefficients. Consider $\vec{\xi_j} = (\xi_1,\xi_2,\ldots,\xi_n)_j$, $\forall j \in [\![0,N]\!]$ to be the j-th random input vector, and $\Theta_j = \Theta(\vec{x},\vec{\xi_j})$, $\forall j \in [\![0,N]\!]$ to be the correspondent deterministic simulation result. Thus, the linear system to solve writes as follows:

$$\begin{bmatrix} \psi_0(\vec{\xi_0}) & \psi_1(\vec{\xi_0}) & \psi_2(\vec{\xi_0}) & \cdots & \psi_P(\vec{\xi_0}) \\ \psi_0(\vec{\xi_1}) & \psi_1(\vec{\xi_1}) & \psi_2(\vec{\xi_1}) & \cdots & \psi_P(\vec{\xi_1}) \\ \psi_0(\vec{\xi_2}) & \psi_1(\vec{\xi_2}) & \psi_2(\vec{\xi_2}) & \cdots & \psi_P(\vec{\xi_2}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \psi_0(\vec{\xi_N}) & \psi_1(\vec{\xi_N}) & \psi_2(\vec{\xi_N}) & \cdots & \psi_P(\vec{\xi_N}) \end{bmatrix} \begin{Bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_N \end{Bmatrix} = \begin{Bmatrix} \Theta_0 \\ \Theta_1 \\ \Theta_2 \\ \vdots \\ \Theta_N \end{Bmatrix} \tag{12}$$

If the number of samples taken N coincides with the number of polynomials $P+1$ needed for the expansion, then we obtain a square linear system in which the number of equations is equal to the number of unknown variables. Thereby, it can be solved numerically to compute the deterministic coefficients needed. However, it is of common practice to use a redundant number of sampling evaluations, which is larger than the number of unknown coefficients, and to solve the system obtained using a regression approach, such as the least squares minimization. Let the system in Equation 12 be written as follows:

$$[\psi]\{\theta\} = \{\Theta\} \tag{13}$$

Here, $[\psi]$ denotes a rectangular matrix. The following manipulation allows us to obtain another linear system to solve, but now with a square matrix.

$$\left([\psi]^T[\psi]\right)\{\theta\} = [\psi]^T\{\Theta\} \tag{14}$$

And one can solve this by inverting $\left([\psi]^T[\psi]\right)$. In [19], an investigation of different sampling techniques (Random, Latin Hypercube and Hammersley) takes place in order to optimize the total number of collocation points N. The authors concluded that all sampling techniques exhibited similar performances, but that the **Latin Hypercube Sampling** (LHS) and the Hammersley sampling demonstrated a much smoother convergence than with random sampling. In addition to this, they have studied the effects of chosing $N > P$. They defined the oversampling ratio N_p as:

$$N_p = \frac{\text{number of samples}}{P+1} \tag{15}$$

Thus, in a regression scenario of NIPCE, the total number of function evaluations (or samples collected) can be computed as $N_p(P+1)$. It will be interesting to investigate the impact of the selected sampling technique (Random, Latin Hypercube, Hammersley, etc.) and the total number of chosen collocation points on result accuracy for our benchmark problems.

2.5 Stochastic post-processing

Once the spectral modes θ_j have been obtained by solving the linear system presented in Equation 12, we can achieve stochastic post-processing for the output variable of interest Θ, using the computed polynomial expansion . The orthogonality of the polynomial basis simplifies the calculation of various stochastic quantities. For example, the mean μ_Θ and the variance σ^2_Θ can be expressed as follow (assuming that the basis has been normalized, which is generally the case):

$$\mu_\Theta = \int \left(\sum_{i=0}^{P} \theta_i \psi_i(\vec{\xi}) \right) w(\vec{\xi})\, \mathrm{d}\vec{\xi} = \theta_0 \tag{16}$$

In Equation 16, the orthogonality of the $\psi_i, i \geq 0$ to ψ_0 or any other scalar factor allows us to determine that the mean is given by the first coefficient of the expansion. In the same way, we obtain the variance as:

$$\sigma^2_\Theta = \int \left(\sum_{i=0}^{P} \theta_i \psi_i(\vec{\xi}) \right)^2 w(\vec{\xi})\, \mathrm{d}\vec{\xi} = \sum_{i=1}^{P} \theta_i^2 \tag{17}$$

Using the same approach, any other stochastic quantity can be estimated with the help of the expansion, which acts as a surrogate for the real variable of interest.

2.6 Computational implementation

When it comes to performing an UQ study, there is a vast range of computational libraries available, specially in Python. As mentioned in the introduction, while this same work could be conducted using Python libraries like OpenTurns, UQpy or Dakota, Chaospy has been chosen due to the specialization of its code in the PCE method, and its straightforward implementation process. In addition, it is important to note that each of these libraries offers a range of statistical tools to complement the PCE method, such as sensitivity analysis and Monte Carlo methods.

3 Numerical applications using OpenFOAM

3.1 OpenFOAM

Our decision to utilize OpenFOAM for CFD in this context is driven by several compelling factors. Firstly, OpenFOAM's status as an open-source software is a significant advantage. This aspect not only makes it accessible without the constraints of licensing fees, but it also benefits from a vast, collaborative community. This community contributes to an extensive array of online resources, tutorials, and forums, facilitating problem-solving and innovation. Such a rich repository of knowledge and tools enhances its adaptability and applicability to diverse CFD problems.

Secondly, OpenFOAM's architecture is particularly conducive to integration with Python scripts, a feature that greatly enhances its flexibility and capability for automation. This compatibility is essential for streamlining the simulation process, allowing for more efficient iteration and analysis. The ease of automating simulations via Python scripts is especially beneficial in the context of UQ, where numerous simulations are often required to adequately capture the range of uncertainties. Furthermore, OpenFOAM's compatibility with Python opens the door to seamless integration with specialized libraries such as ChaosPy. Finally, it is important to highlight that the software's version used is OpenFOAM v2306 for the Lid driven cavity problem, and OpenFOAM v10 for the NACA0012.

3.2 Lid-Driven Cavity Problem

The first benchmark problem selected to be analyzed is the lid-driven cavity problem. It consists of a square cavity of length L and unit depth, with fluid filling it completely. The flow is induced by the movement of the top of the square, which is called the lid. A diagram of the problem is presented in Figure 1. In this diagram, we have denoted the horizontal and vertical velocities as u and v, respectively.

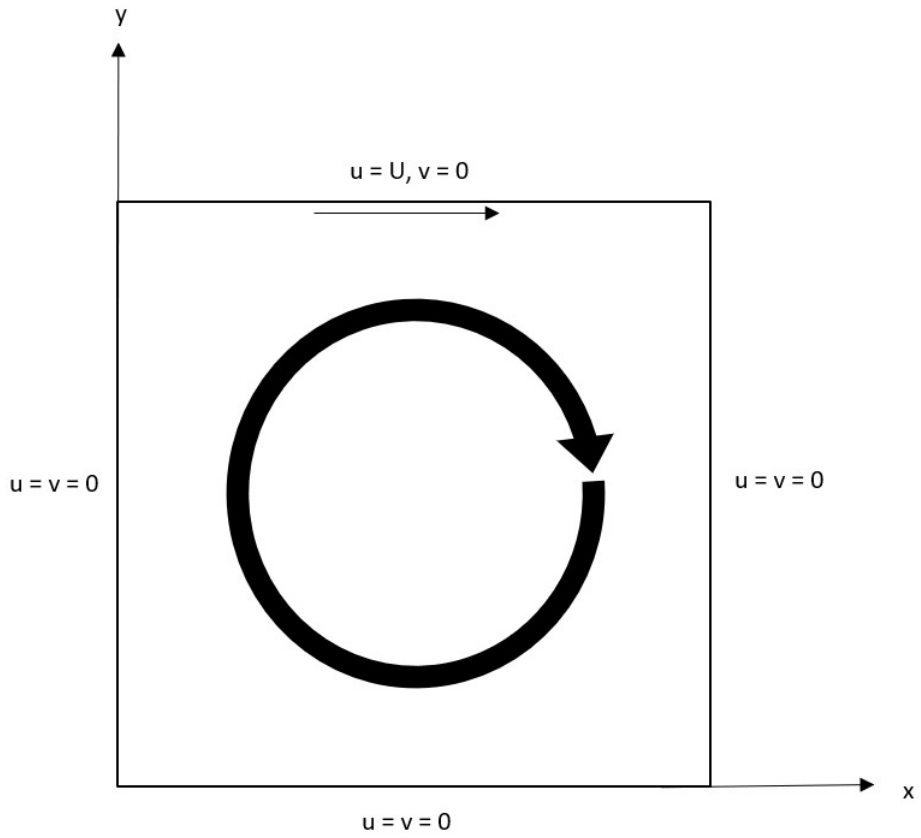


Figure 1: Geometry of the lid-driven cavity problem.

The lid-driven cavity problem has been widely used in the CFD community as a validation case for

new codes or new methodologies, as this is a very simple problem in terms of geometry and boundary conditions. It has been solved by many researchers, both for laminar and turbulent flowsdee. For our study, we will focus on analyzing the laminar steady solution, and we are going to compare the obtained results with the reference article from Ghia et al. [1]. The decision to focus exclusively on the laminar flow solution aims to simplify the initial implementation in OpenFOAM, given that this serves merely as a validation case.

3.2.1 Formulation of The Problem

At first, it is very important to understand what are the equations that we are going to solve using OpenFOAM, to truly comprehend the physics behind the lid-driven cavity problem. We recall that we consider a square cavity with side length L and a horizontal lid velocity equal to U, at the top of the domain. It is also considered that the left, bottom and right walls of the cavity are fixed. Considering that the speed vector is denoted as $\vec{u} = (u, v)$, being u and v the horizontal and vertical velocities respectively, we have $\vec{u}_{\text{lid}} = (U, 0)$, while $\vec{u}_{\text{wall}} = (0, 0)$. These are Dirichlet boundary conditions of the problem.

We aim to solve the incompressible Navier-Stokes equations assuming constant flow properties (viscosity, thermal conductivity, specific heat). The equations to be solved are the following:

$$\nabla \cdot \vec{u} = 0 \tag{18}$$

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} = -\frac{\nabla p}{\rho} + \nu \Delta \vec{u} \tag{19}$$

Where ν is the Kinematic viscosity. The reference article deals with the nondimensional Navier-Stokes equations, therefore we must perform a non-dimensionalization of the governing equations. In order to do so, we consider the following non-dimensional variables:

$$\tilde{x} = \frac{x}{L},\ \tilde{y} = \frac{y}{L},\ \tilde{t} = \frac{t}{L/U},\ \tilde{\vec{u}} = \frac{\vec{u}}{U},\ \text{and } \tilde{p} = \frac{p}{\rho U^2}$$

Note that by using the new non-dimensional variables, we obtain the non-dimensional nabla and laplacian operators:

$$\tilde{\nabla} = \frac{1}{L}\left(\frac{\partial}{\partial \tilde{x}}, \frac{\partial}{\partial \tilde{y}}\right) \text{ and } \tilde{\Delta} = \frac{1}{L^2}\left(\frac{\partial^2}{\partial \tilde{x}^2}, \frac{\partial^2}{\partial \tilde{y}^2}\right)$$

In addition, consider the following definition of the Reynolds number in the context of the cavity problem:

$$R_e = \frac{UL}{\nu} \tag{20}$$

By inserting the new variables in the Navier-Stokes equations, we can nondimensionalize the equations. Proceeding with Equation 18, we obtain

$$\nabla \cdot \vec{u} = \frac{U}{L}\tilde{\nabla} \cdot \tilde{\vec{u}} = 0 \Rightarrow \tilde{\nabla} \cdot \tilde{\vec{u}} = 0$$

When considering Equation 19, it results in the following:

$$\begin{aligned}\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} = -\frac{\nabla p}{\rho} + \nu \Delta \vec{u} &\Leftrightarrow \frac{U^2}{L}\frac{\partial \tilde{\vec{u}}}{\partial \tilde{t}} + \frac{U^2}{L}(\tilde{\vec{u}} \cdot \tilde{\nabla})\tilde{\vec{u}} = -\frac{U^2}{L}\tilde{\nabla}\tilde{p} + \frac{\nu}{L^2}\tilde{\Delta}\tilde{\vec{u}} \\ &\Leftrightarrow \frac{\partial \tilde{\vec{u}}}{\partial \tilde{t}} + (\tilde{\vec{u}} \cdot \tilde{\nabla})\tilde{\vec{u}} = -\tilde{\nabla}\tilde{p} + \frac{1}{R_e}\tilde{\Delta}\tilde{\vec{u}}\end{aligned}$$

By dropping the tilde notation, without loss of generality, we find ourselves with the non-dimensional Navier-Stokes equations to solve:

$$\nabla \cdot \vec{u} = 0 \tag{21}$$

$$\frac{\partial \vec{u}}{\partial t} = -(\vec{u} \cdot \nabla)\vec{u} - \nabla p + \frac{1}{R_e}\Delta \vec{u} \tag{22}$$

3.2.2 Implementation Using OpenFOAM

The primary goal is to ascertain the reliability of the OpenFOAM solver across different regimes, ensuring confidence in subsequent simulations when employing the NIPCE method. This will enable us to use OpenFOAM's outcomes as a benchmark for future comparisons across various flow regimes. Since we must solve the nondimensional Navier-Stokes equations, it is then necessary to consider the square to be of length $L = 1$ m and the lid velocity to be equal to $U = 1$ m/s. To characterize the different regimes of flow.

The main objective of examining the lid-driven cavity problem is to become proficient with OpenFOAM's implementation, enabling the generation of multiple parallel simulations. This facilitates subsequent data post-processing for conducting UQ analysis on the problem. Hence, the initial task is to generate results for $R_e \in \{100, 400, 1000, 3200, 5000, 7500, 10000\}$, comparing these with the findings of Ghia et al. by examining the horizontal velocity at the mid-line as a function of the vertical position y. This comparison aims to identify the most suitable range of R_e values for conducting the UQ analysis.

For this problem, the only parameter that can be changed in order to introduce a stochastic behaviour in the output results is the Reynolds number. Since OpenFOAM solves the dimensionalized Navier-Stokes equations, it is better to maintain the lid velocity fixed, so that by altering the kinematic viscosity numerically we can manipulate the Reynolds number systematically and generate results to compare with the reference article. Although in real problems the kinematic viscosity must be fixed depending on the fluid analysed, the cavity problem is only being solved to compare results with literature and to validate the NIPCE method, so it is not problematic to alter the Reynolds number this way.

In order to solve the cavity problem using OpenFOAM, the icoFoam solver [20] was selected. This solver is designed to solve the incompressible laminar Navier-Stokes equations using the PISO (Pressure-Implicit with Splitting of Operators) algorithm [21]. Since the code is transient, it is mandatory to include initial conditions as well as boundary conditions.

The initial conditions considered are the relative pressure as $p = 0$ and $U = 0$ at all points. The pressure in OpenFOAM is given as $P = p_{ref} + p$, where p_{ref} is ambient pressure. The boundary conditions are $\nabla p = 0$ on all boundaries, $u = v = 0$ on the bottom, right and left walls and $u = 1$ and $v = 0$ at the lid. Since in [1] the results were obtained using 129 grid points in x and y, the OpenFoam mesh will consider 128 square cells in the x and y directions.

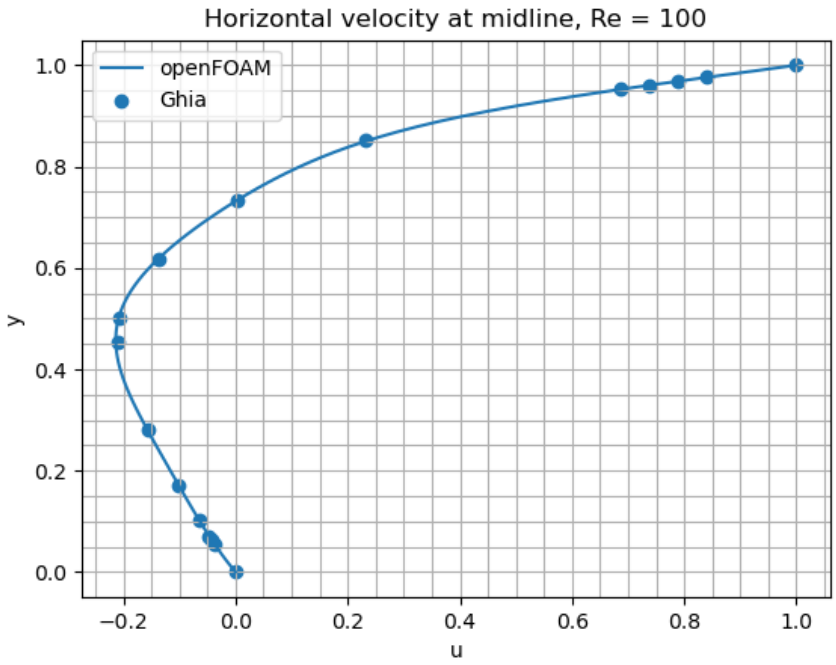
Before running any simulations, it is important to select the appropriate time step Δt to avoid instabilities. To achieve temporal accuracy, a Courant number $C_o < 1$ is necessary [22]. The Courant number for one cell is defined as:

$$C_o = \frac{\Delta t |\vec{u}|}{\Delta x} \tag{23}$$

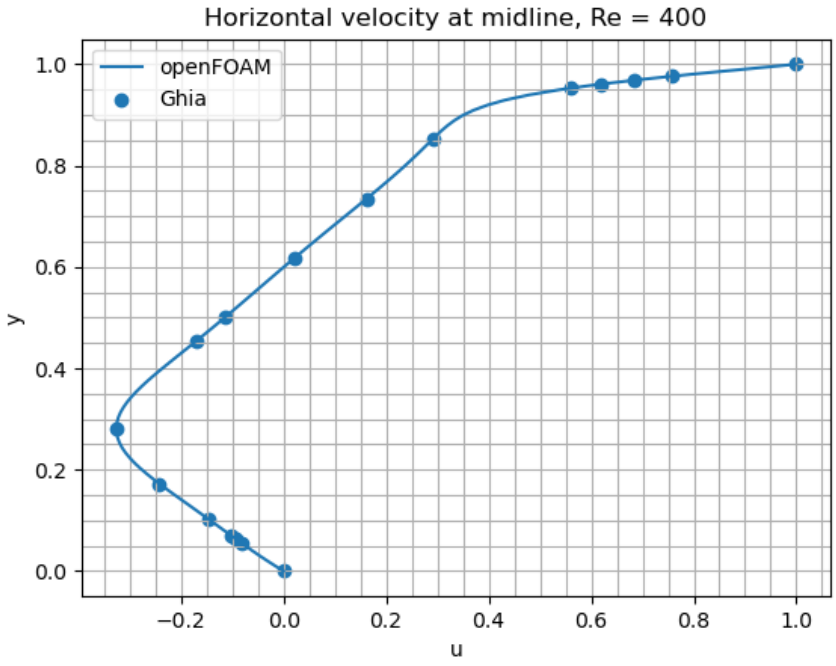
Where Δx and $|\vec{u}|$ denote the side of the square cell and the magnitude of the velocity vector in the cell. We are considering $n = 128$ cells in the initial simulations, and since the square side measures $L = 1$ m, we have $\Delta x = L/n = 0.00781$ m. In order to guarantee stability, the velocity magnitude was set equal to 1, which is the maximum speed found in the problem, and the Courant number was set to 0.8. This way, $\Delta t = 0.00625$ s.

With all these parameters fixed, the solver was set to run until a final time of 300 s, to ensure that we reach the steady-state solution from the non-stationary solver, and the horizontal velocity at the vertical mid-line was obtained. The steady-state solution is desired in order to conduct an honest comparison with the reference article, and the final time needed for convergence increased as the Reynolds number gets higher, so picking a very high value for the final time would assure that all results are indeed in the steady-state regime. The comparison with Ghia et al. is displayed in

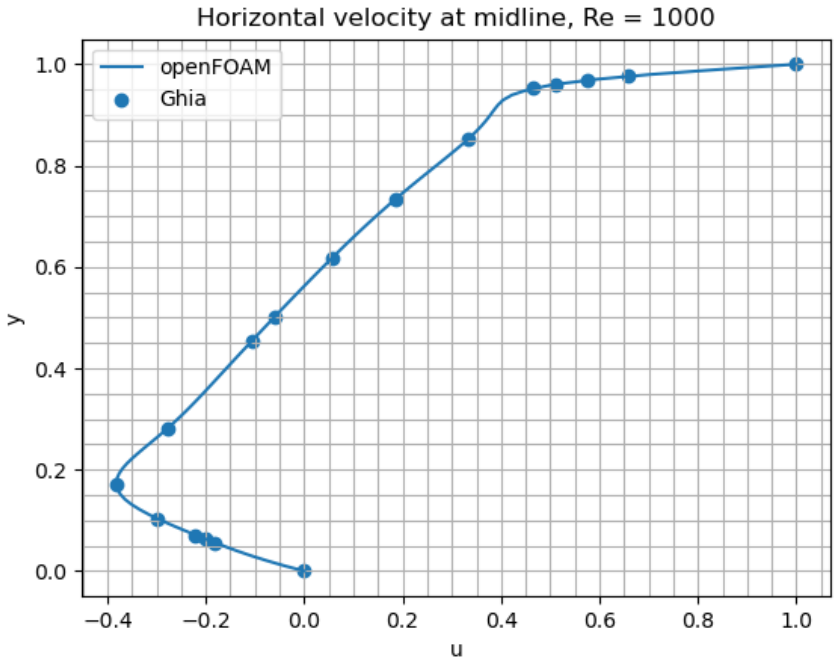
Figure 2. Finally, it is good to note that the horizontal velocity at the vertical mid-line or the vertical velocity at the horizontal mid-line are two possible output results that can be used to compare with [1]. A work that does the same kind of comparison to test numerical codes on the cavity flow problem is [23].



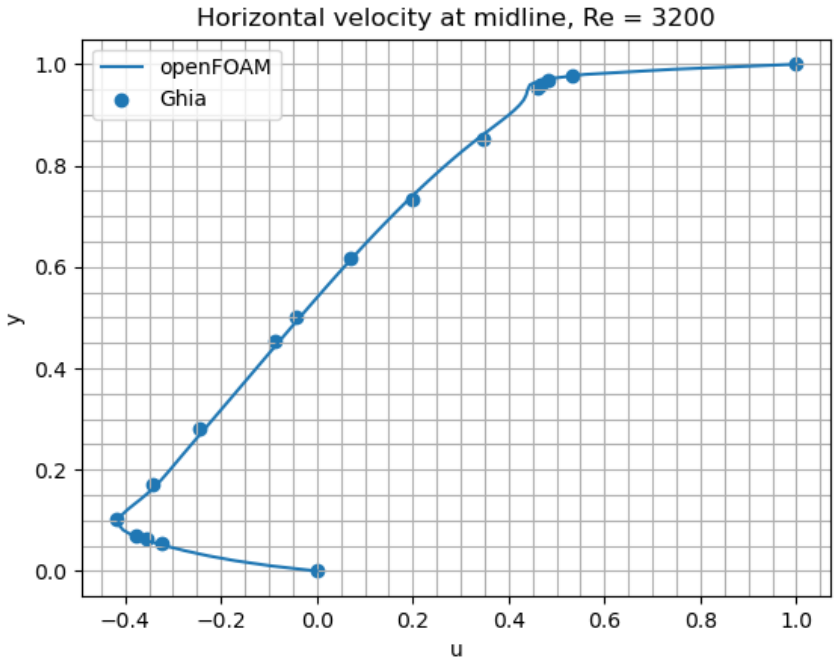
(a) Comparison between steady-state OpenFOAM results and Ghia et al. [1] for the horizontal velocity u at the vertical mid-line of the cavity, $R_e = 100$.



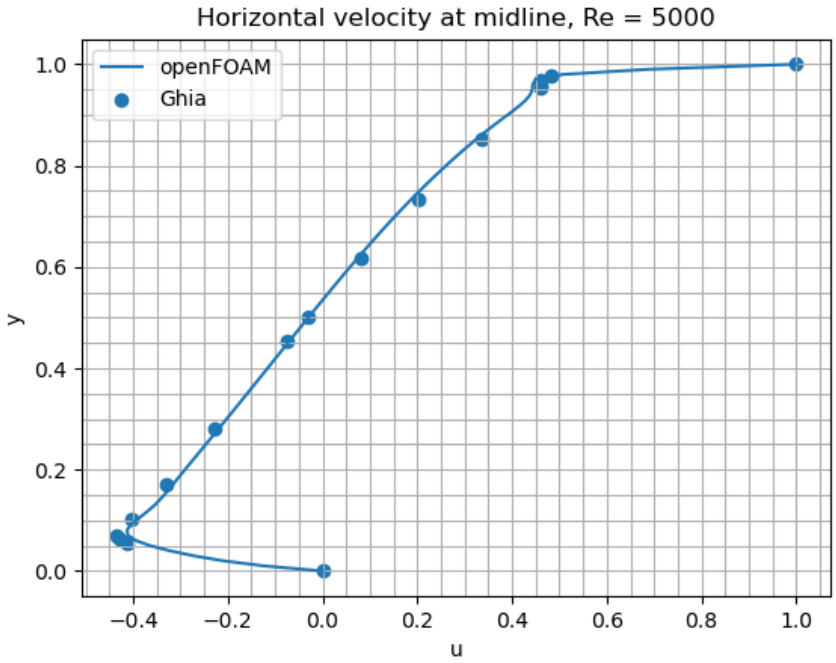
(b) Comparison between steady-state OpenFOAM results and Ghia et al. [1] for the horizontal velocity u at the vertical mid-line of the cavity, $R_e = 400$.



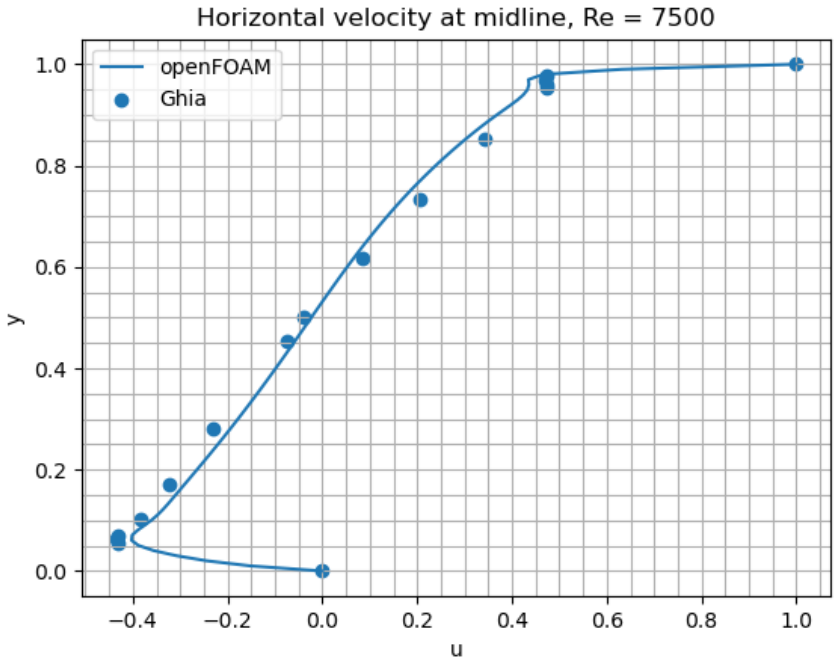
(c) Comparison between steady-state OpenFOAM results and Ghia et al. [1] for the horizontal velocity u at the vertical mid-line of the cavity, $R_e = 1000$.



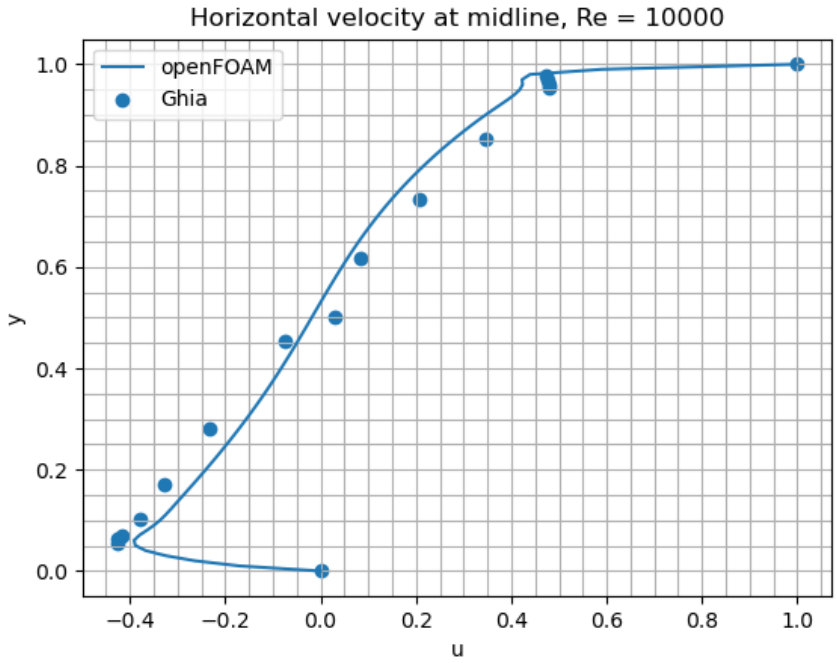
(d) Comparison between steady-state OpenFOAM results and Ghia et al. [1] for the horizontal velocity u at the vertical mid-line of the cavity, $R_e = 3200$.



(e) Comparison between steady-state OpenFOAM results and Ghia et al. [1] for the horizontal velocity u at the vertical midline of the cavity, $R_e = 5000$.



(f) Comparison between steady-state OpenFOAM results and Ghia et al. [1] for the horizontal velocity u at the vertical midline of the cavity, $R_e = 7500$.



(g) Comparison between steady-state OpenFOAM results and Ghia et al. [1] for the horizontal velocity u at the vertical midline of the cavity, $R_e = 10000$.

Figure 2: Comparison between OpenFOAM and Ghia et al. [1] results for the horizontal velocity u at the mid-line of the cavity, for various Reynolds numbers.

Although all simulation results obtained using OpenFOAM display the expected behaviour, it is clear that OpenFOAM does not appear to be so reliable for higher Reynolds numbers. In order to compare the results in quantitative terms, linear interpolation of the data obtained from the OpenFOAM simulations was used to directly compare with Ghia et al., since the y coordinates of the horizontal velocities computed with OpenFOAM were not the same as the ones in the reference article. This allowed us to compute the **Root Mean Square Error** (RMSE) and directly compare the simulation outputs with Ghia et al. The results are displayed figure 3.

As observed in Figure 3 we can rely with great confidence in the OpenFOAM results up to $R_e = 1000$, as for these Reynolds numbers the RMSE calculated are the lowest before a sudden increase. Thus, this will be the domain of study for the UQ analysis.

Before proceeding to the statistical study using the NIPCE method, a convergence analysis must be conducted. Such analysis is of pivotal importance to assure that neither Δx nor the final time of the OpenFOAM simulations will influence the final velocity field obtained. Indeed, Reynolds number must be the only parameter introducing uncertainty in the case, making it possible to assess the quality of the NIPCE analysis. In addition, it is highly important to conduct a convergence analysis in order to seek the minimal number of cells and final time required to obtain numerical convergence in the OpenFOAM simulations, leading to minimizing the required computational time to run all the CFD simulations needed for the NIPCE study.

By taking into account that the statistical study will focus on varying the Reynolds number of the CFD simulations around an average Reynolds number $R_{e,avg} = 1000$, the convergence analysis was conducted with $R_e = 1000$. Being `ncells` the number of cells used for each side of the square domain in the simulations, it has been considered `ncells`$\in \{20, 40, 60, 80, 100, 120, 140\}$. The results are displayed in Figure 4

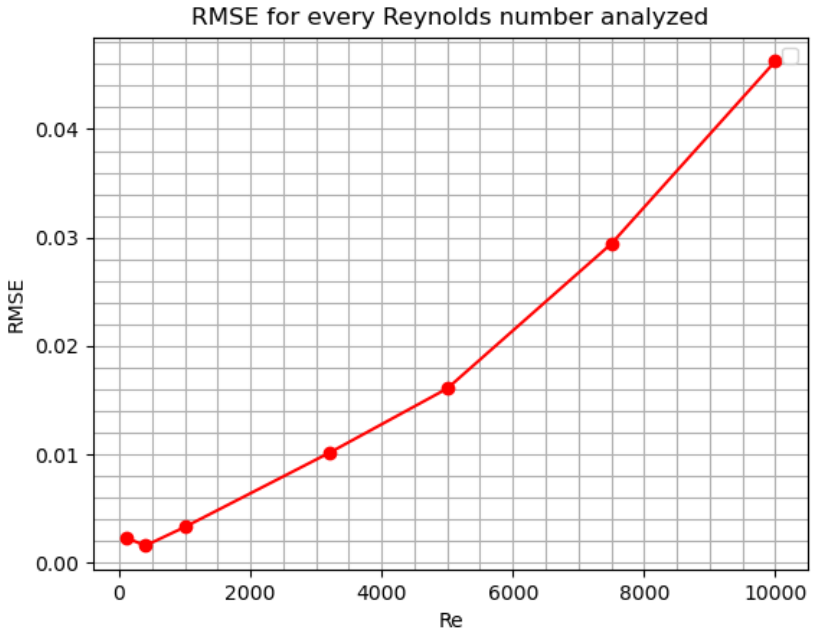
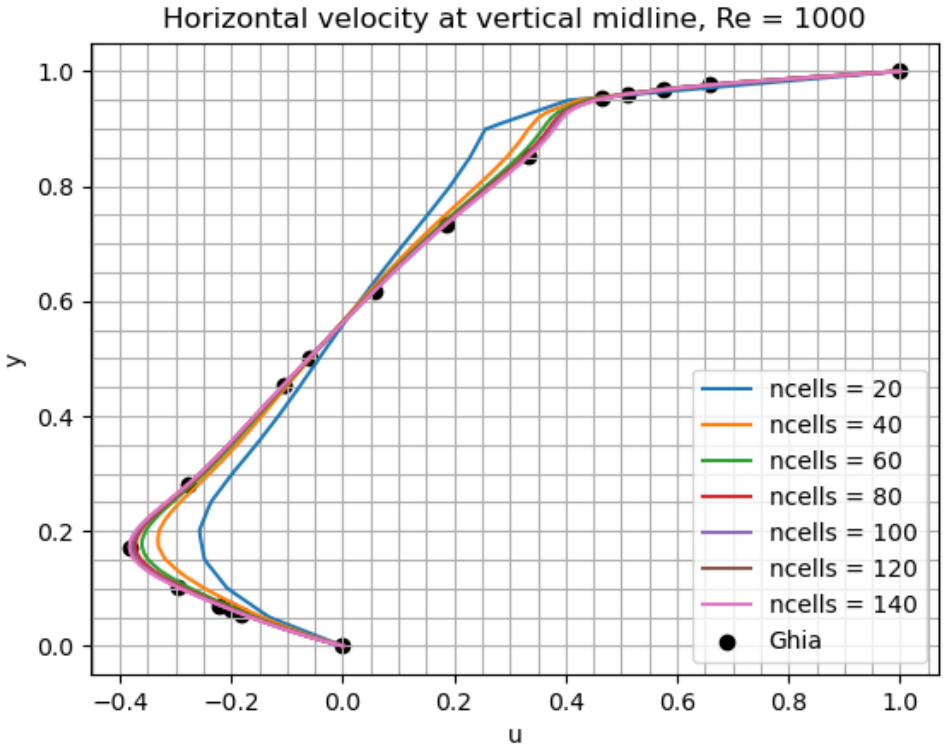
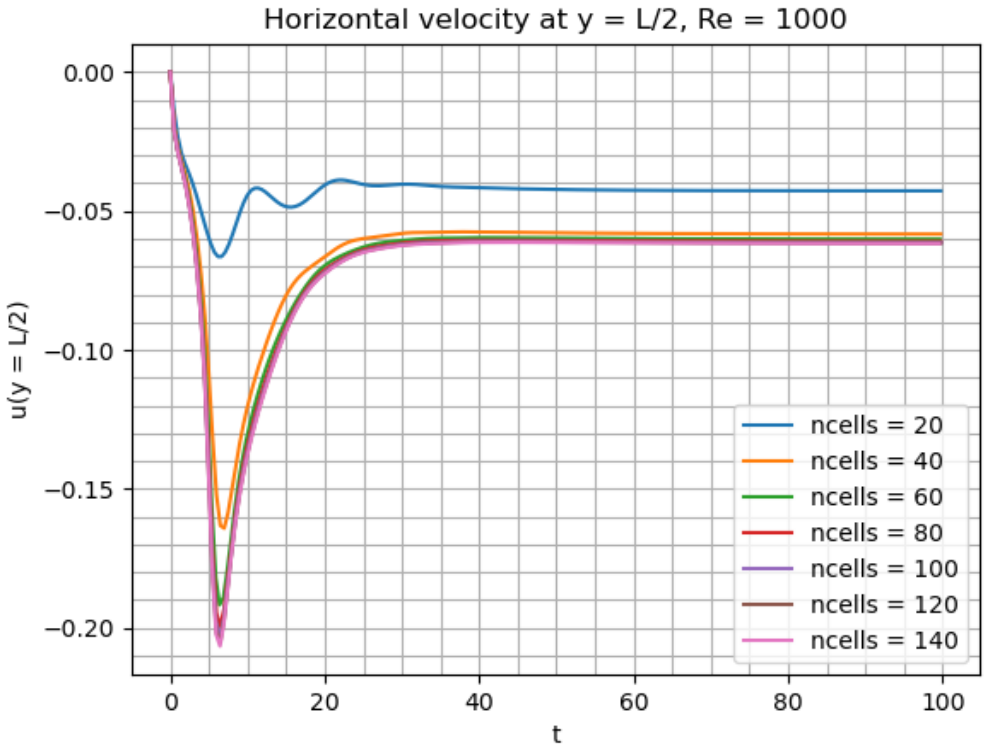


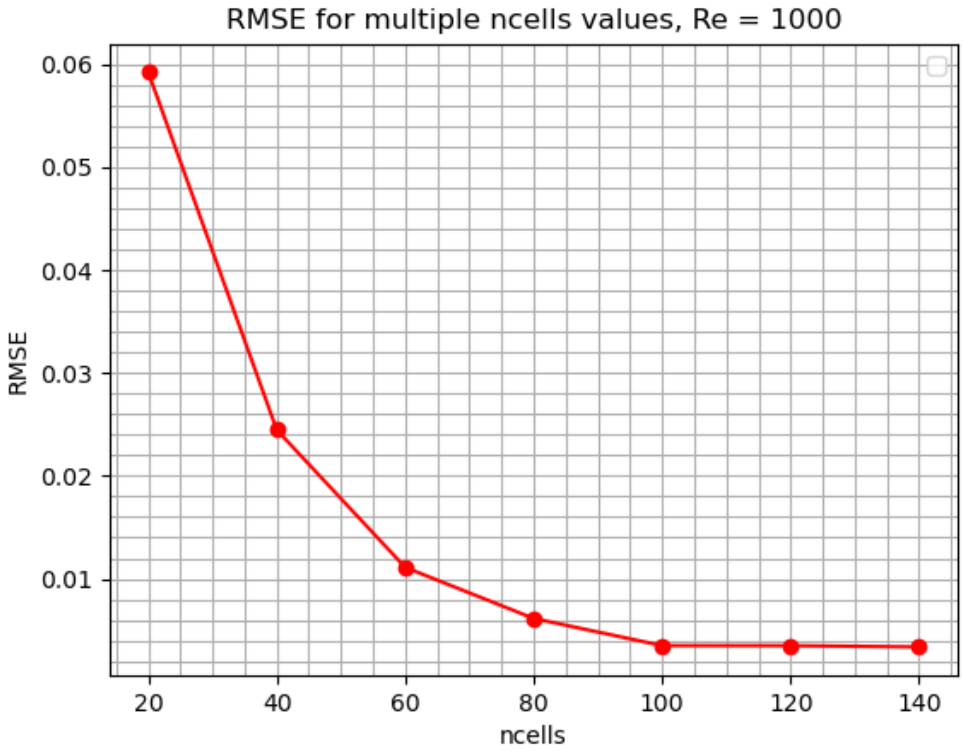
Figure 3: RMSE computed for all the Reynolds numbers analyzed in the comparison with the reference article [1].



(a) Comparison between steady-state OpenFOAM results and Ghia et al. [1] for the horizontal velocity u at the vertical mid-line of the cavity, systematically varying `ncells` with $R_e = 1000$.



(b) Comparison between OpenFOAM and Ghia et al. [1] results for horizontal velocity u at $(x, y) = (L/2, L/2)$, systematically varying `ncells` with $R_e = 1000$.



(c) RMSE computed for all simulations and analyzed in the comparison with the reference article, systematically varying `ncells` with $R_e = 1000$.

Figure 4: Convergence analysis results, systematically changing `ncells` with $R_e = 1000$.

It is possible to perceive from the convergence study that by considering `ncells` = 100 the RMSE values stop decreasing and achieve an asymptotic behavior. Additionally, for a final time $t_f = 100$ the CFD simulations converge nicely. Since we are interested in varying the Reynolds number during the statistical study, it is important to use a value of t_f that is higher than necessary, to always guarantee we obtain steady-state solutions from OpenFOAM simulations.

By fixing the mesh size and final step of the simulation, it is possible to finally build the surrogate model to predict the velocity field at the vertical mid-line in the cavity problem using the NIPCE method.

3.2.3 Statistical Study

A Python script was developed to facilitate the statistical study, enabling the parallel execution of all CFD simulations using OpenFOAM and integrating seamlessly with the Chaospy [4]library. Chaospy is a Python-based numerical toolbox that is designed to aid researches in the field of UQ. It is specialized in the use of PCE and advanced Monte Carlo methods, in addition to encompassing an extensive set of utilities for low-discrepancy sampling, quadrature generation, polynomial manipulations and much more.

In the context of the lid-driven cavity problem, Chaospy was used in order to generate Reynolds numbers that follow a desired probability distribution. It is possible to generate a set of Reynolds numbers following a diverse range of PDFs, such as the ones presented in Table 1, but we focused on implementing the analysis only by considering that $R_e \sim \mathcal{N}(R_{e,avg}, R_{e,std})$, being $\mathcal{N}(\mu, \sigma)$ the standard notation for the normal distribution with average μ and standard deviation σ. As this is a test case for the implementation of Chaospy, the values $R_{e,avg} = 1000$ and $R_{e,std} = 100$ were selected, thus $R_e \sim \mathcal{N}(1000, 100)$.

Each CFD simulation in OpenFOAM returns values for the horizontal velocity u for multiple coordinates in y, the vertical axis of reference. Bearing in mind that the vector containing all m y coordinates is written as $\{y\} = \{y_1, y_2, \ldots, y_m\}^T$, $\forall m \in \mathbb{N}$, it is necessary to address the fact that for a fixed coordinate y_j, $\forall j \in [\![0, m]\!]$, there will be multiple velocities computed for different R_e values. Consequently, it is necessary to build m polynomial expansions to serve as surrogates for u in all m y coordinates obtained in the collection of OpenFOAM outputs. Afterwards, each one of the m polynomial expansion will calculate the respective approximation for the horizontal velocity for $R_e = R_{e,avg}$ and all approximations will be superposed in order to build an approximate velocity profile for the vertical mid-line, further enabling the comparison with a deterministic OpenFOAM simulation and the reference article when $R_e = R_{e,avg}$.

Continuing the theory behind the statistical study, not only must the PDF of the R_e values be defined, but a set of values for the order of the polynomial chaos p must also be fixed in order to check the influence in the output PCE results. Moreover, since the only uncertain input parameter is R_e, we have a one-dimensional random variable vector $\vec{\xi}$, thus $n = 1$ in this problem. With Equation 2 we can obtain the total number of terms for all expansions computed during the analysis, and thus we can select the total number of samples used to compute the deterministic coefficients of each expansion by manipulating the oversampling ratio N_p, described in Equation 15. Therefore, this initial analysis was conducted by taking $p \in [\![3, 6]\!]$ and $N_p \in [\![1, 3]\!]$.

Given the p and N_p values, then the total number of samples can be written as

$$N_{samples} = N_t N_p \Leftrightarrow N_{samp} = N_p \frac{(n+p)!}{n!p!}$$

It is thus necessary to sample $N_{samples}$ values of R_e, which follows a normal distribution. There is a vast collection of methods to obtain pseudo-random samples given the variable's PDF, but in this work we have used the random and the **Latin Hypercube Sampling** (LHS) [24, 25]. The random method generates classical pseudo-random samples, in which new sample points are generated without considering the ones generated previously. It is not necessary to have any prior knowledge of the total number of sample points desired. In contrast, with LHS there is the need to know how many sample points are going to be used, because this sampling method divides the sample space in equally probable intervals, and then the sample points are selected in a way that guarantees a representative

and stratified sampling across the entire space. The greatest benefit of using LHS is that this method ensures a more balanced and comprehensive exploration of the sampling space.

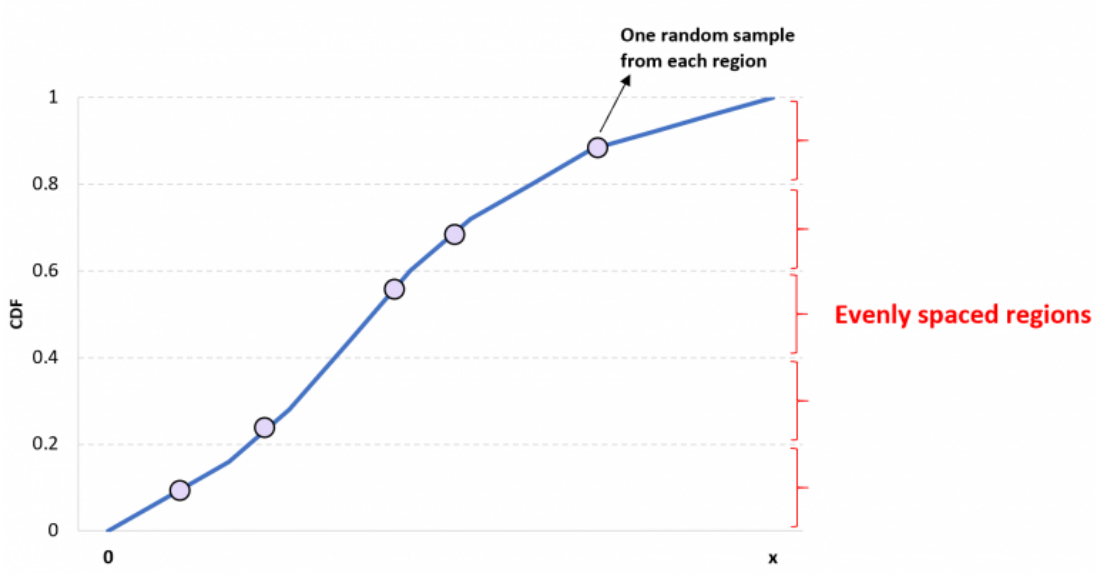


Figure 5: Example of stratified domain for the selection of sample points using the LHS method. Here, CDF stands for Cumulative Density Function.

In the context of this initial implementation of the Chaospy library, only the LHS sampling is considered. After running all necessary CFD simulations for all combinations of values for p and N_p, it was observed that the results were pretty close to each other, so the velocity profile approximation could not be distinguished. This way, in order to visualize the NIPCE outcomes, the following figure illustrates the results obtained for the velocity profile at the vertical mid-line considering $p = 3$ and $N_p = 2$.

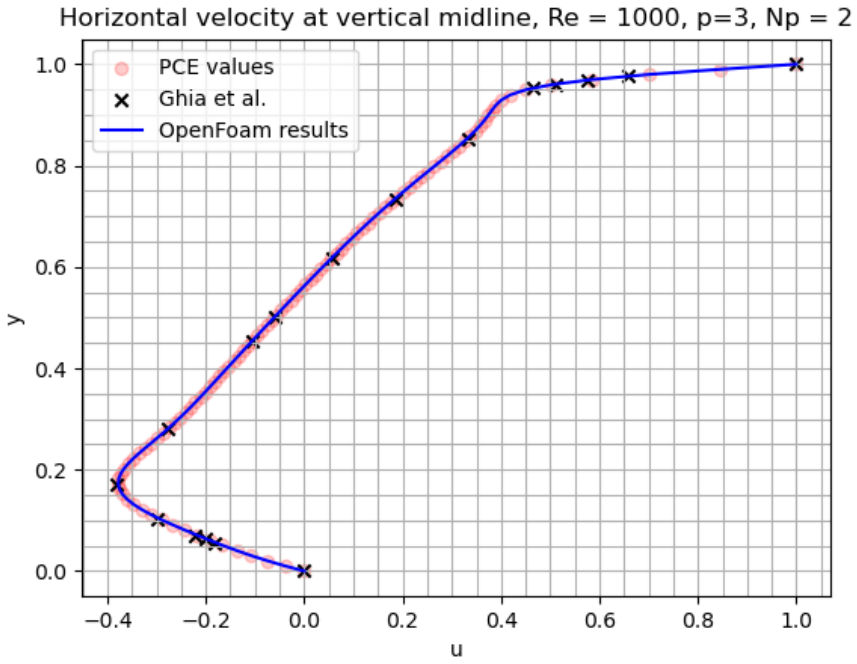
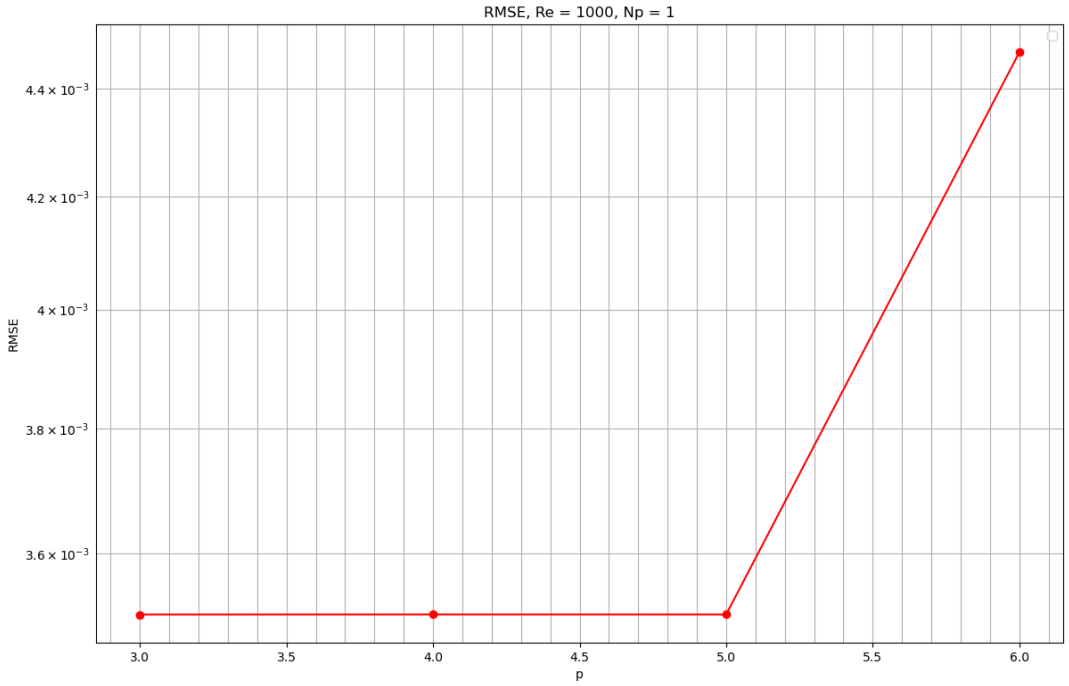
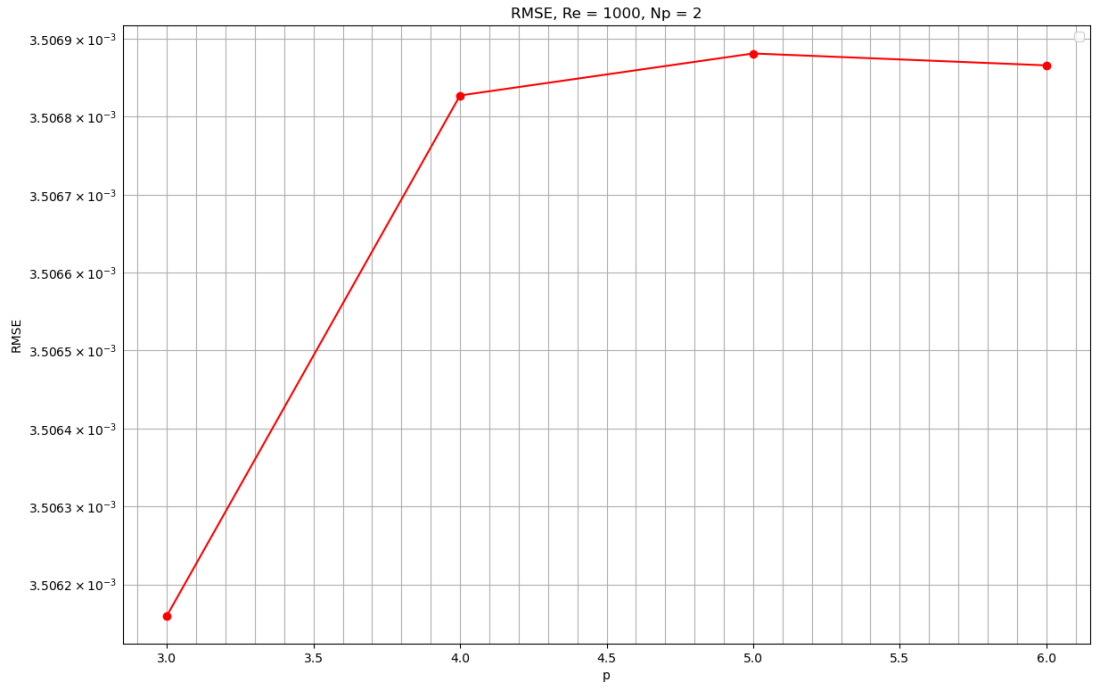


Figure 6: Comparison between steady-state OpenFOAM results, Ghia et al. [1] and the PCE approximations for the horizontal velocity u at the vertical mid-line of the cavity, for $p = 3$, $N_p = 2$ and $R_e = 1000$.

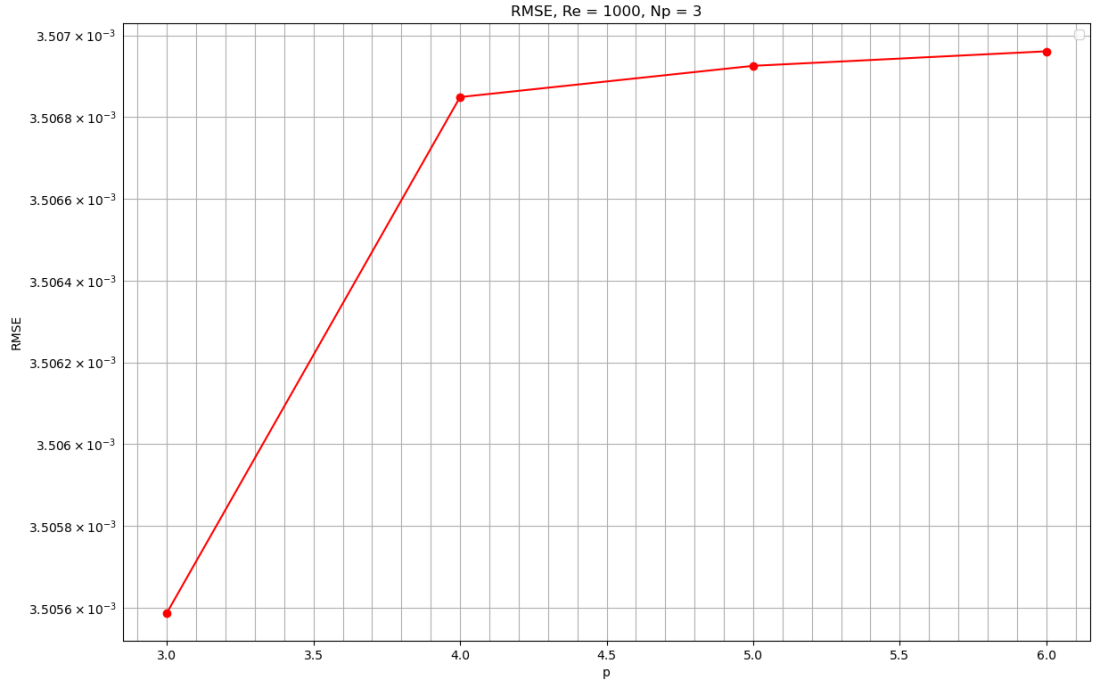
The way to compare all PCE results is to use the RMSE values for each value of N_p by varying the order of the polynomial chaos p. The results are displayed in Figure 7.



(a) RMSE of the PCE approximation when compared with Ghia et al. [1] for various values of p, with $N_p = 1$ and $R_e = 1000$.



(b) RMSE of the PCE approximation when compared with Ghia et al. [1] for various values of p, with $N_p = 2$ and $R_e = 1000$.



(c) RMSE of the PCE approximation when compared with Ghia et al. [1] for various values of p, with $N_p = 3$ and $R_e = 1000$.

Figure 7: RMSE of the PCE approximation when compared with Ghia et al. [1] for various values of p, with $N_p \in [\![1, 3]\!]$ and $R_e = 1000$.

Comparing the obtained results, we can conclude that the PCE was successfully implemented and with great results. For every combination of p and N_p used, there was a very small RMSE computed, and with the increase of N_p we can see the decrease of the overall RMSE. Although there is an apparent increase in the RMSE computed as p increases, it is very important to notice that they are all still of the same order of magnitude, thus this error can be associated to machine error in the calculations performed as well.

When compared with classical Monte Carlo methods, the NIPCE method implemented using Chaospy required less computational time and allowed for results that are very close to the reference article used.

3.3 Airflow around NACA0012 airfoil

The second benchmark problem studied in this work is the two-dimensional, steady state, compressible flow around a NACA0012 airfoil.

3.3.1 Presentation of the case

The NACA0012 is a symmetric airfoil shape commonly used in aerodynamic studies and applications, and is part of the NACA airfoil series. The "0012" denotes that the airfoil has no camber, a maximum thickness of 12% of the chord length, and a simple, streamlined profile. The focus of this simulation is on manipulating two critical parameters which will play the role of the uncertain input parameters: the upstream Mach number, and the airfoil's angle of attack, defined on Figure 8. The choice of these two parameters is explained by the fact that they are essential for influencing the aerodynamic forces acting on the profile, and that in practice they can be uncertain and vary due, for example, to wind gusts or turbulence phenomena. This simulation therefore aims to compute the lift (C_l) and drag

(C_d) coefficients, in order to study their sensitivity to variation in input parameters via the NIPCE method. The definition of these two coefficients is as follows:

- **Lift Coefficient (C_L):** The lift coefficient is a dimensionless quantity that represents the ratio of the lift force to the dynamic pressure and the reference wing area. It is expressed mathematically as:

$$C_L = \frac{L}{\frac{1}{2}\rho V^2 S}$$

  where:

  - L is the lift force,
  - ρ is the air density,
  - V is the airspeed,
  - S is the reference wing area.

- **Drag Coefficient (C_D):** The drag coefficient is also a dimensionless parameter, representing the ratio of the drag force to the dynamic pressure and the reference wing area. It is given by the formula:

$$C_D = \frac{D}{\frac{1}{2}\rho V^2 S}$$

  where D is the drag force.

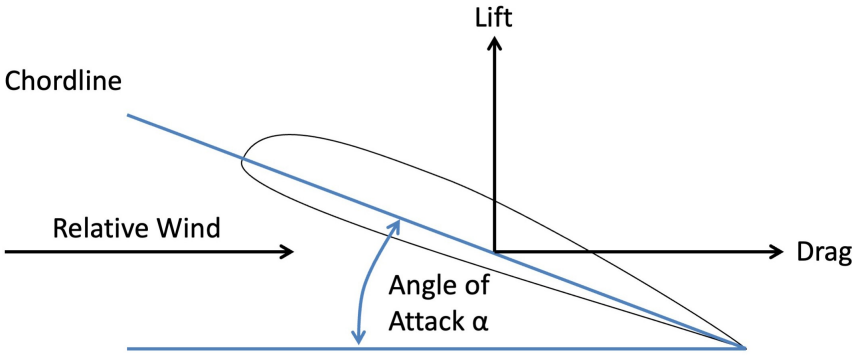


Figure 8: Flow around an airfoil

3.3.2 Numerical implementation on OpenFOAM

The starting point for this study was an existing OpenFOAM case from the tutorials. The fixed parameters of the problem were set according to the following table :

Parameter	Value
Pressure (p)	10^5 Pa
Temperature (T)	293 K
Gas Model	Air at room temperature (polytropic perfect gas)
Dynamic Viscosity (μ)	1.82×10^{-5}
Prandtl Number (Pr)	0.71
Turbulence Model	k-omega SST
Solver	rhoSimpleFoam

Table 2: Fixed Parameters of the CFD Simulation

The mesh used is the following :

```
domain
{
    xMax  100;
    zMax  50;

    xMin #neg $xMax;
    zMin #neg $zMax;

    // Number of cells
    zCells  60; // aerofoil to far field
    xUCells 60; // upstream
    xMCells 25; // middle
    xDCells 50; // downstream

    // Mesh grading
    zGrading    30000;  // aerofoil to far field
    xUGrading   10;     // towards centre upstream
    leadGrading 0.005;  // towards leading edge
    xDGrading   400;    // downstream
}
```

```
aerofoil
{
    xLead   0;
    zLead   0;
    xTrail  1;
    zTrail  0;
    xUpper  0.3;
    zUpper  0.06;

    xLower  $xUpper;
    zLower  #neg $zUpper;
}
```

Figure 9: Mesh parameters for the NACA0012 case

The airfoil under study was thus positioned at the center of a domain one hundred times larger in length. Besides, the specified number of cells and mesh grading allowed for the calculation and verification (using Paraview) of the first cell size, in contact with the airfoil, which was around $0.2mm$. Initially, the simulation faced numerical instabilities and convergence issues at a Mach number of 0.72 (transonic regime), likely due to the first cell size being too large for the initial velocity. Refining the mesh would have been too costly in terms of memory and computing time, knowing that the study was made using our own personal computers. Consequently, the focus shifted to studying the flow at a lower Mach number, around 0.3, to ensure the flow remained within the subsonic regime. With these values, the calculations began to converge. However, it was observed that at angles of attack greater than approximately 13°, a stall phenomenon emerged. This led to oscillations and instabilities in the simulation, as illustrated in the residuals plot (Figure 10) of a case with an angle of attack of 15° and a Mach number of 0.3. These residuals represent the error in the numerical solution of the governing equations of the fluid dynamics at each iteration: as the simulation progresses, we expect the residuals to decrease, indicating that the solution is converging towards a steady state. On the contrary, if the residuals increase, it may suggest that the simulation is not converging.

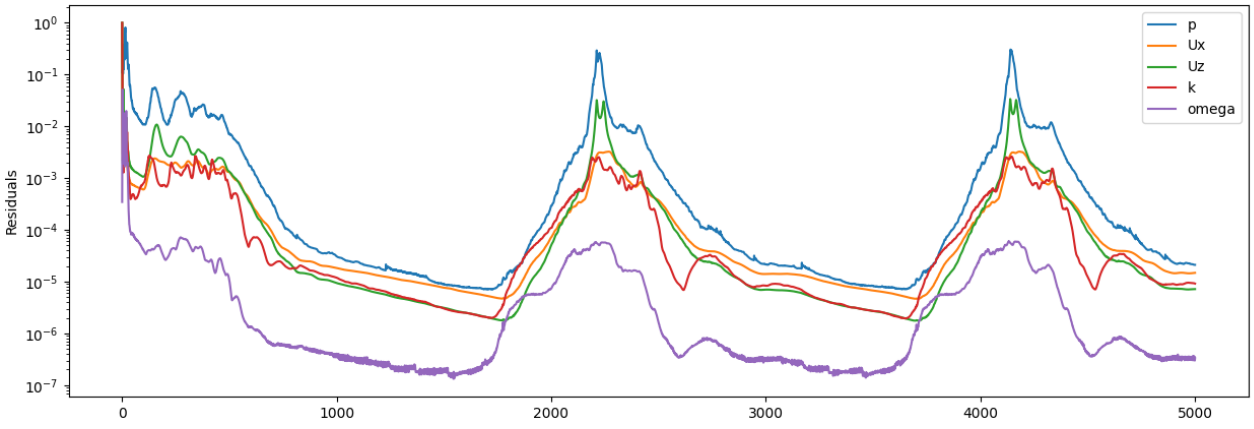


Figure 10: Iteration residuals for various fluid properties throughout the simulation

To streamline the process and validate the simulation results, we developed a Python script that automated the generation and the running of simulation cases based on a set of input parameters (angles of attack and Mach numbers), along with post-processing and computation of the aerodynamic coefficients. Initially, we only varied the angle of attack, keeping the Mach number fixed at 0.3. The outcomes of these simulations displayed good convergence. Further, these computational results were compared against experimental data from NASA ([26]), ensuring similarity in Reynolds number, Mach

number, and angle of attack conditions. This comparison is illustrated in Table 3 and Figure 11, highlighting the relative errors between the computational and experimental data.

Angle of Attack(°)	Cd (NASA)	Computed Cd	Cd Relative Error(%)	Cl (NASA)	Computed Cl	Cl Relative Error(%)
-4.37	0.0089	0.0065	37.0	-0.4943	-0.49	1.0
-2.02	0.0084	0.0063	32.9	-0.2284	-0.2316	1.4
-0.01	0.0083	0.0063	30.1	-0.0012	-0.0079	85.2
0.03	0.0083	0.006559	25.3	0.0035	0.0026	233.0
4.03	0.0088	0.00728	20.9	0.4557	0.4452	2.4
6.08	0.0095	0.00711	34.2	0.6859	0.6762	1.4
8.13	0.0113	0.00822	37.8	0.9044	0.9039	0.1
10.12	0.0137	0.012	13.9	1.1073	1.1251	1.6
11.13	0.0154	0.0132	24.9	1.2018	1.2273	2.1
12.12	0.0176	0.01397	26.3	1.2845	1.3172	2.5
13.32	0.0208	0.01484	18.2	1.3336	1.3939	4.3
14.27	0.0248	0.01827	41.6	1.9117	1.9377	1.4
15.14	0.0421	0.03423	35.9	1.9211	1.9213	0.01
16.25	0.0624	0.20698	69.9	1.2510	1.1132	12.4

Table 3: Comparison Table

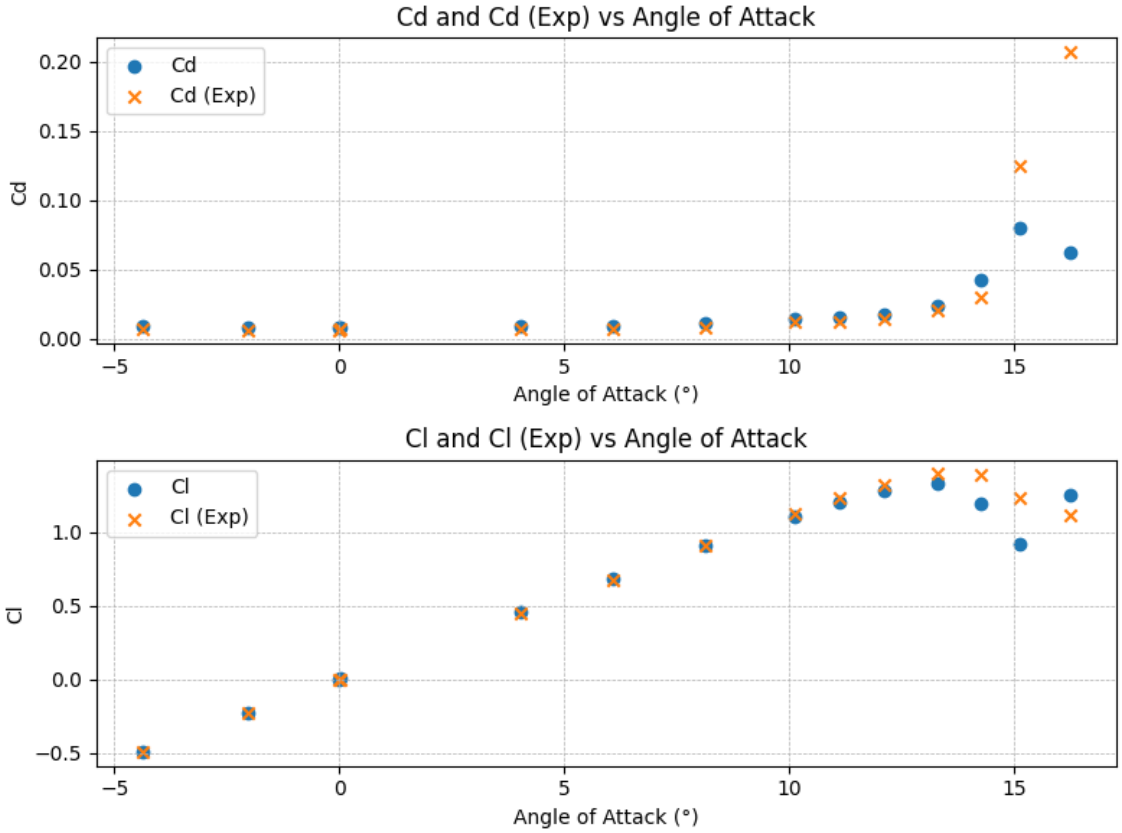


Figure 11: Comparison between the aerodynamic coefficients obtained experimentally and the ones computed using OpenFOAM

The comparison between the experimental and computational data reveals some insightful trends. It is observed that the computed lift coefficient (C_l) aligns relatively well with the experimental data, especially for angles of attack below 13°. Below this angle, the relative error remains under 5%. Notably, at angles near 0, a discrepancy arises due to the minuscule magnitude of the lift coefficient; however, both experimental and computational values of C_l are nearly zero, as illustrated in the plot. Beyond 13°, numerical instabilities associated with the stall phenomenon cause divergence, with the relative error escalating to 25%. In contrast, the drag coefficient (C_d) displays a higher relative error, around 25%, for angles below the stall angle. This can be attributed to the insufficient mesh resolution, and to the inherent limitations of the RANS turbulence model to accurately capture stall

phenomena around the airfoil, which significantly contribute to the observed drag. Though mesh refinement would enhance accuracy, it is prohibitively resource-intensive. However, considering the primary focus on Uncertainty Quantification (UQ) rather than precision in CFD calculations, we opt to continue with the current simulation setup. Consequently, the lift coefficient is designated as the sole output variable of interest for angles of attack between $-12°$ and $12°$, where the data correlation is satisfactory. Initially, the study will treat the angle of attack as the sole uncertain parameter, followed by an independent examination of the Mach number's effects. Finally, the Non-Intrusive Polynomial Chaos Expansion (NIPCE) method will be implemented considering both the angle of attack and Mach number as uncertain inputs.

3.3.3 First study : Angle of Attack as the only uncertain parameter

In this first application of the NIPCE method to the airfoil scenario, we arbitrarily set the Mach number at 0.3, identifying the angle of attack as the sole uncertain input variable. In view of previous discussions concerning the convergence of our results, the lift coefficient will be the only output variable of interest for this study. Besides, we also limit our examination to angles of attack between $-12°$ and $12°$. We therefore opted for a normal distribution of the angle of attack, centered on $0°$ with a standard deviation of $5°$. This selection was made arbitrarily, largely due to the constraints posed by the full-order model employed in this study. However, it should be noted that, in practice, the distribution function of the uncertain parameter is usually determined by empirical means.

For this specific scenario, we used Python to implement the NIPCE method. We designed a function using the Chaospy library, which accepts several parameters: the full-order model, the angle of attack distribution, the expansion order, the number of samples, the sampling method and the fitting method. This function calculates the polynomial chaos expansion by computing the deterministic coefficients using the point collocation technique. In addition, we have streamlined the analysis of the results by automating the process. The function also produces an Excel file that records all the parameters used, visualizes the PCE obtained, compiles descriptive statistics and facilitates comparison with the full-order model, improving the completeness and efficiency of our research methodology.

Beyond simply demonstrating the effectiveness of the NIPCE method, our objective extended to examining the influence of various parameters on the accuracy of the results obtained. The parameters examined in this investigation include sampling technique, fitting method, expansion order and oversampling ratio. To do so, we established a reference database using the full-order model. It includes lift coefficient values derived from around 160 different angles of attack, serving as a basic data set for assessing the impact of each parameter on the accuracy of NIPCE results. The results are plotted in Figure 12.

To compare the different approximate models derived by varying these parameters, we used the L^2 relative error defined as follows :

$$\epsilon = 100 \times \frac{\sqrt{\sum_{t=1}^{n} (C_{l_t}^{\mathrm{FOM}} - C_{l_t}^{\mathrm{PCE}})^2}}{\sqrt{\sum_{t=1}^{n} (C_{l_t}^{\mathrm{FOM}})^2}} \% \tag{24}$$

The parameter ϵ quantifies the discrepancy between the lift coefficients as predicted by the full-order model (FOM), obtained through CFD simulations using OpenFOAM, and the polynomial chaos expansion (PCE), which is derived from the approximate model constructed via the NIPCE method. The summation extends over the lift coefficients computed at the angles of attack within the reference dataset. A lower ϵ value signifies a closer alignment of the PCE results with the FOM data, thus indicating a more accurate model. However, it is crucial to acknowledge that the reference database against which ϵ is measured was compiled based on arbitrary selection criteria. Therefore, the error metric specifically reflects the model's capacity to replicate these selectively chosen data points and may not necessarily translate to the model's effectiveness or precision in more general or varied applications.

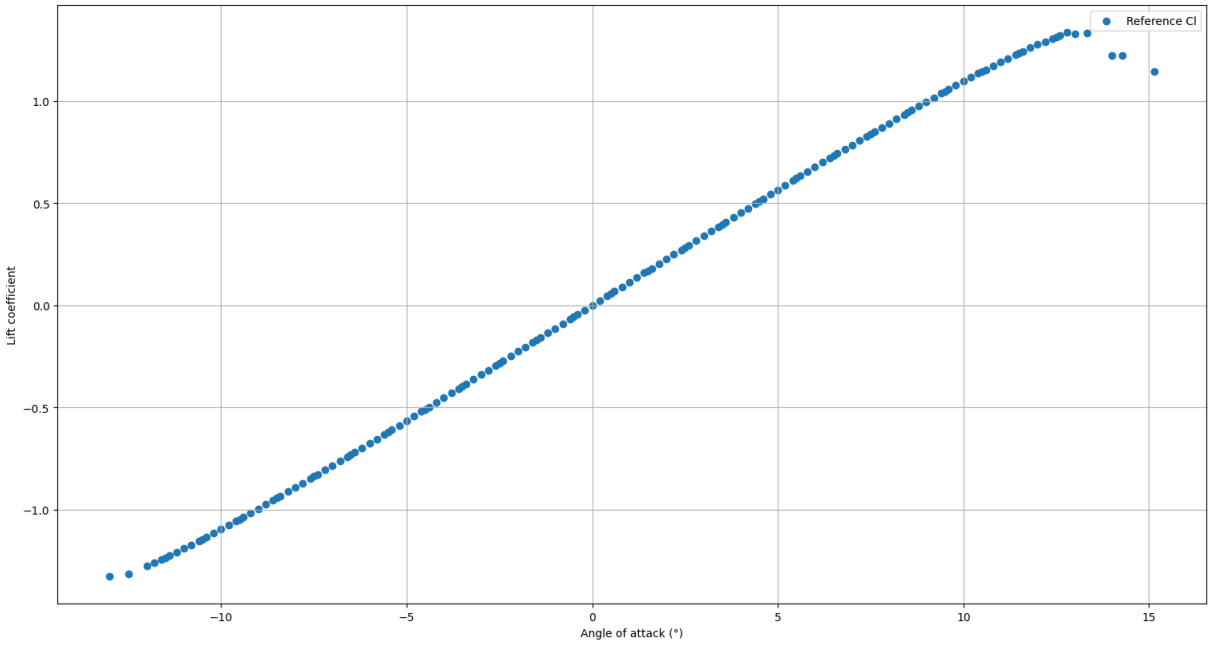


Figure 12: Reference values for the lift coefficient

Impact of sampling method

The first parameter we studied was the sampling method. Although the Chaospy function's default sampling method generates random samples, it is possible to opt for other methods such as Latin Hypercube Sampling (LHS), as described in the previous section. We anticipate that the LHS method will give better results, as it generates a more evenly distributed set of points and mitigates the clustering that can occur with a pseudo-random method. For the study, we held all other parameters constant and only varied the sampling method in each case in order to compare the different PCEs obtained using the L^2 relative error described above. It should be noted that the fitting method employed throughout the study, until the last subsection, will be Chaospy's default method, the least squares. The summarized results are presented in the following table:

Expansion order	Number of Samples	Sampling method	Cl L2 relative error (%)
3	6	Random	8.7
3	6	LHS	6.1
3	9	Random	6.2
3	9	LHS	6.1
4	8	Random	7.5
4	8	LHS	7.3
4	12	Random	7.5
4	12	LHS	5.6
5	10	Random	3.4
5	10	LHS	7.3
2	4	Random	8.9
2	4	LHS	8.1
5	15	Random	3.34
5	15	LHS	2.7

As expected, we observe that LHS sampling gives superior results in the majority of cases examined here, as shown by the lower relative L^2 error of the models obtained. It is worth noting one particular case, (expansion order of 5 and number of samples of 10), where random sampling gave a better

result. However, there is no apparent explanation other than that, in this particular scenario, the sample points obtained were more evenly distributed. Henceforth, we will adopt the LHS method as the default sampling method in all subsequent analyses.

Impact of expansion order and oversampling ratio

Next, we investigated the impact of expansion order and oversampling ratio on the accuracy of the results. It is pertinent to recall that in this 1D case, where the angle of attack is the sole uncertain parameter, the oversampling ratio is determined by $\frac{number\ of\ samples}{p+1}$, where p represents the polynomial chaos expansion order. While maintaining all other parameters constant, we varied one of the two parameters and derived a set of results. Subsequently, we plotted the L^2 relative error ϵ of C_l as a function of both the expansion order and the oversampling ratio. We anticipate that augmenting both parameters will lead to more precise results. The resulting plots are presented below:

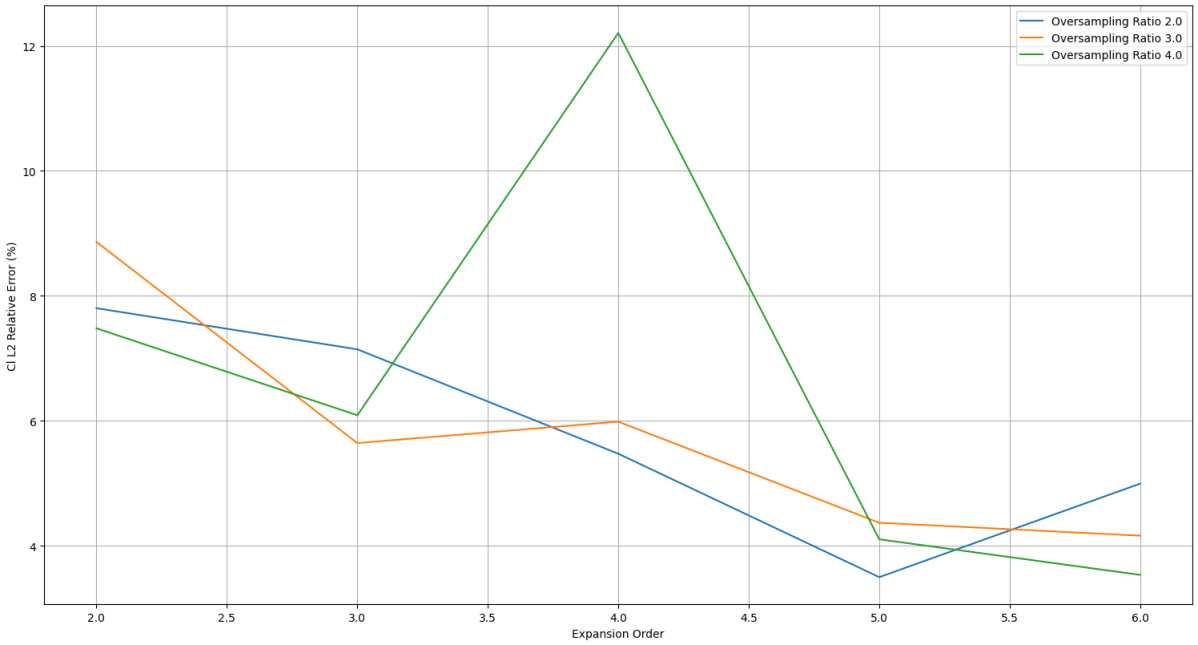


Figure 13: Cl L2 relative error vs Expansion Order

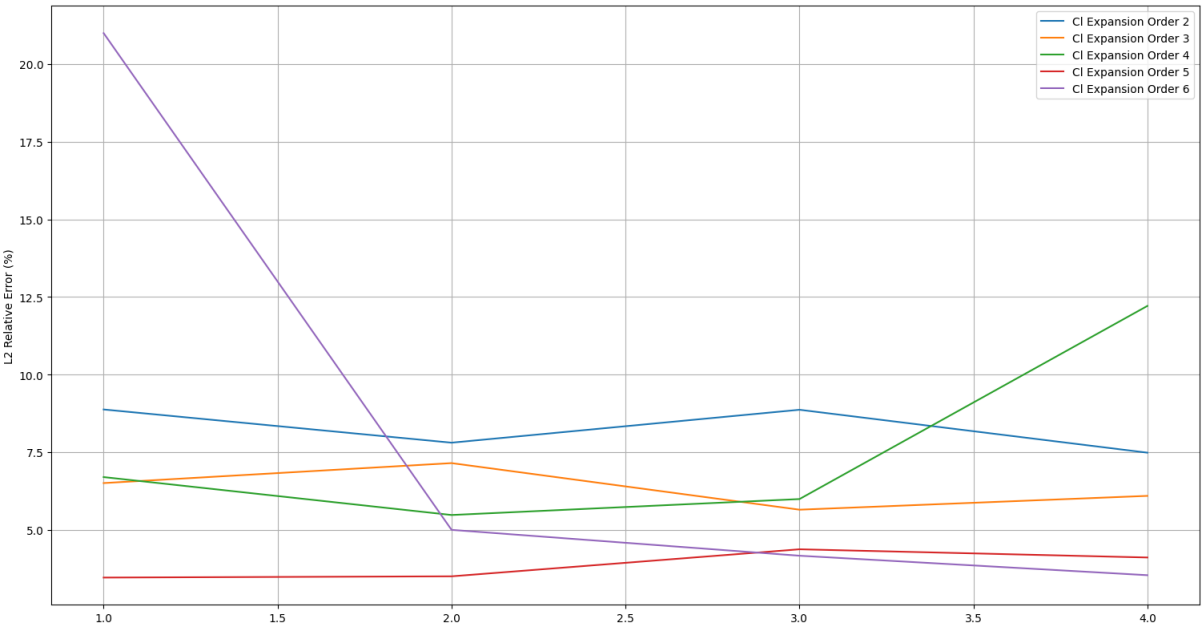


Figure 14: Cl L2 relative error vs Oversampling Ratio

In Figure 13, which illustrates the L^2 relative error as a function of the expansion order for a fixed oversampling ratio ranging from 2 to 4, a general trend emerges: as the expansion order increases, the accuracy of the approximate models improves. For instance, with an oversampling ratio of 3, the L^2 relative error decreases from 8.5%for an expansion order of 2 to approximately 4% for an expansion order of 6. However, anomalies are observed, particularly for the cases where the expansion order is 4 and the oversampling ratio is 4, and where the expansion order is 6 and the oversampling ratio is 2. In these instances, contrary to expectations, the error increases instead of decreasing.

The explanation for this phenomenon is as follows: when the expansion order is increased for a fixed oversampling ratio, the number of samples required for evaluation using the point collocation technique also increases. Consequently, even if the distribution—such as the normal distribution centered at 0° with a standard deviation of 5° in this case—is within the intervals of angles where our FOM converges (here, [-12°, 12°]), there remains a non-negligible probability that one of the sampled points will fall outside of these convergence intervals. This probability increases with the number of samples. When such occurrences arise, the evaluated points yield incorrect values of C_l, which confound the model. This phenomenon is depicted in Figure 15.

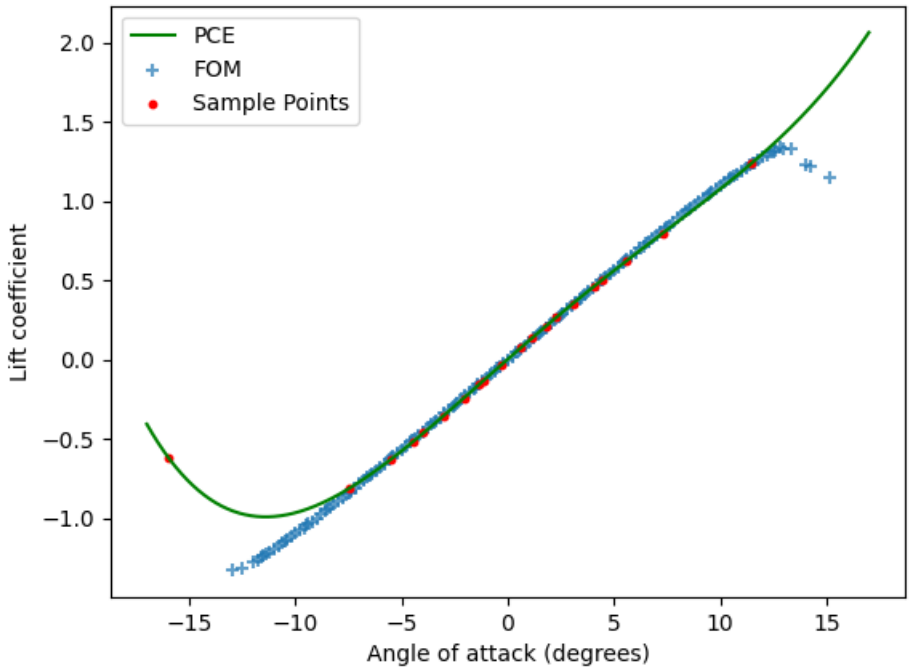


Figure 15: PCE obtained with an expansion order of 4 and an oversampling ratio of 4

The green line represents the evaluations of the lift coefficient using the polynomial expansion acquired through the NIPCE method, juxtaposed with the blue dots, which denote the reference data derived from the FOM. The red dots illustrate the samples utilized to compute the deterministic coefficients of the expansion employing the point collocation technique. A total of 20 samples are depicted, and it is notable that one sample corresponds to an angle of attack of -16°, which lies outside the previously identified convergence domain. This outlier leads the model to deviate from the reference points, consequently resulting in a poor approximation, and an increase of the error.

In Figure 14, which illustrates the L^2 relative error as a function of the oversampling ratio for fixed expansion orders ranging from 2 to 6, a general trend also appears: the error tends to decrease as the oversampling ratio increases. Similar anomalies, attributed to the reasons elucidated earlier, are also observed. Additionally, it is notable that beyond an oversampling ratio of 2, the reduction in error becomes less pronounced, while the computational time required significantly escalates due to the increased number of points at which the FOM must be evaluated. Furthermore, the expansion order appears to exert a more substantial influence in reducing the error compared to the oversampling ratio. Therefore, it would preferable to opt for a higher expansion order with a smaller oversampling ratio rather than the reverse.

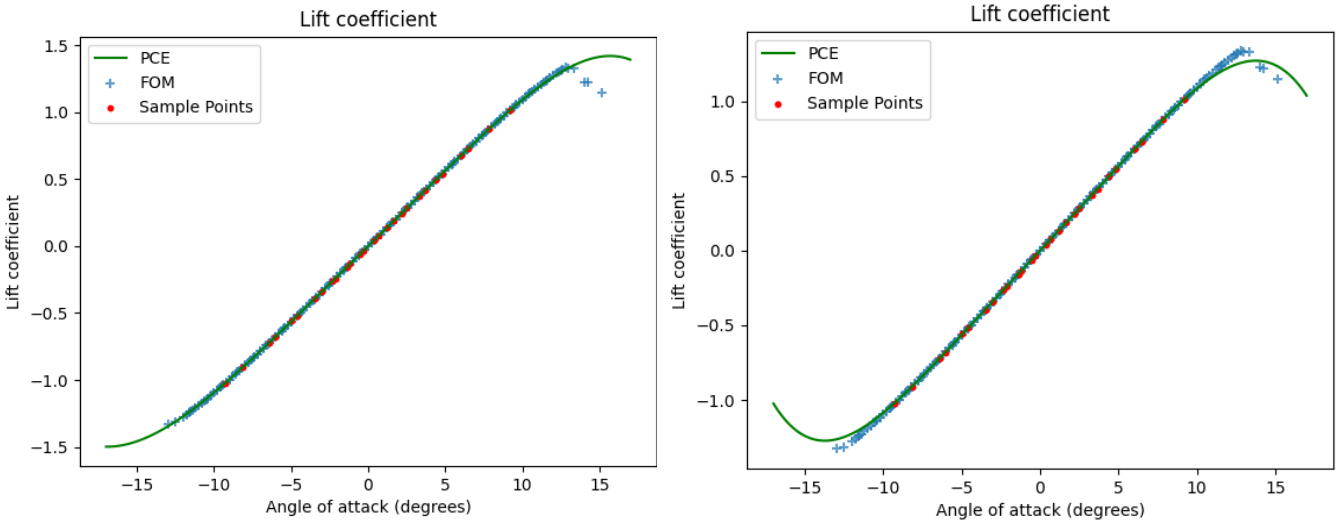
We also observe that for this particular case, an expansion order of 5 give very good results, even for low oversampling ratios. It should be noted that we have not attempted to obtain results with higher orders or oversampling ratios, due to the computational resources required for this purpose.

Impact of fitting Method

The last parameter studied is the fitting method. Throughout the previous analyses, the regression approach used to solve the system presented in equation 12 was the least squares method, which is the default method when using Chaospy functions. However, alternative regression approaches exist, that can be used, and we chose to explore the LASSO (Least Absolute Shrinkage and Selection Operator) regression, as it is one of the easiest to understand and implement.

Compared to the Least squares method, which minimizes the sum of the squared residuals, Lasso regression adds a penalty term to the standard cost function, specifically the sum of the absolute values of the coefficients. This tends to produce simpler models that select only the most important features and shrink the coefficients of less important features towards zero. This helps prevent over fitting and can also lead to sparse models where some coefficients are exactly zero, effectively performing variable selection. The key parameter in Lasso regression is the regularization parameter, often denoted by α, which controls the strength of the penalty term. A larger α leads to more shrinkage of the coefficients and a sparser model, while a smaller α allows the coefficients to be less restricted. This parameter α must be tuned according to each of the case studied.

To assess whether Lasso regression could provide better results, we revisited some of the scenarios studied previously and used Lasso regression to perform the fit while adjusting the α parameter. We observed that, in the absence of adjusted parameter, the results obtained were really poor. However, in each case, there was a value of alpha that improved the approximation over that derived from least squares minimization with the same set of parameters. For example, with an expansion order of 6 and an oversampling ratio of 4, we obtained the following two models (for Lasso regression, we set $\alpha = 0.001$):



(a) Fitting method : Least Squares

(b) Fitting method : Lasso with $\alpha = 0.001$

Figure 16: Results for an expansion order of 6 and an oversampling ratio of 4

With the Lasso method, the relative error is 2.7%, compared to 3.5% with the Least Squares minimization. Furthermore, we observe that the PCE derived from the Lasso regression better captures the behavior at the stall, even though it was not trained with any sampling points from this region. We thus observe that Lasso regression can indeed outperform LMS. However, it necessitates tuning the α parameter differently for each case to ensure consistent results. It is noteworthy that this tuning process does not demand significant computational time and resources. It can be implemented efficiently using a simple optimization algorithm, considering that the most resource-intensive task is the evaluation of the FOM on the sampling points.

Discussion

From this initial investigation, the NIPCE method appears capable of delivering accurate approximate models with a minimal number of FOM evaluations. It also demonstrates an ability to capture intricate relationships and behaviors of the output variable, even when not explicitly trained on such points (refer to Figure 16b). However, to offer a critical perspective on the undertaken study, it is essential to consider the following points:

- The selection of the L2 relative error as a metric, and the construction of the reference data set for comparing different models is somewhat arbitrary. While some models may exhibit larger errors, they could potentially excel in predicting phenomena such as stall, which occurs at higher angles of attack.

- The efficacy of the NIPCE method heavily relies on the robustness of the FOM model employed. As observed here, limitations emerged from OpenFOAM calculations diverging at high angles of attack, resulting sometimes in poor results that were not expected.

3.3.4 Second study : Mach number as the only uncertain parameter

In this second application of the NIPCE method to the airfoil scenario, we arbitrarily set the angle of attack at 5°, identifying this time the Mach number as the only uncertain input variable. The lift coefficient will still be the only output variable of interest for this study. After conducting a convergence study of the OpenFOAM simulations, we decided to limit our examination to Mach numbers between 0.15 and 0.5. We therefore opted for a normal distribution of the Mach number, centered on 0.325 with a standard deviation of 0.85. This selection was once again made arbitrarily according to the constraints posed by the full-order model employed in this study.

The implementation of the NIPCE method in this case remained largely unchanged. The code underwent modifications solely to incorporate the parallelization of numerical simulations, which facilitated a reduction in computational time. To perform a similar parametric study as before, we compiled a reference database using the FOM, encompassing lift coefficient values derived from 70 distinct Mach numbers (Figure 17). As in the previous analysis, we will employ the L^2 relative error to compare the various approximate models derived.

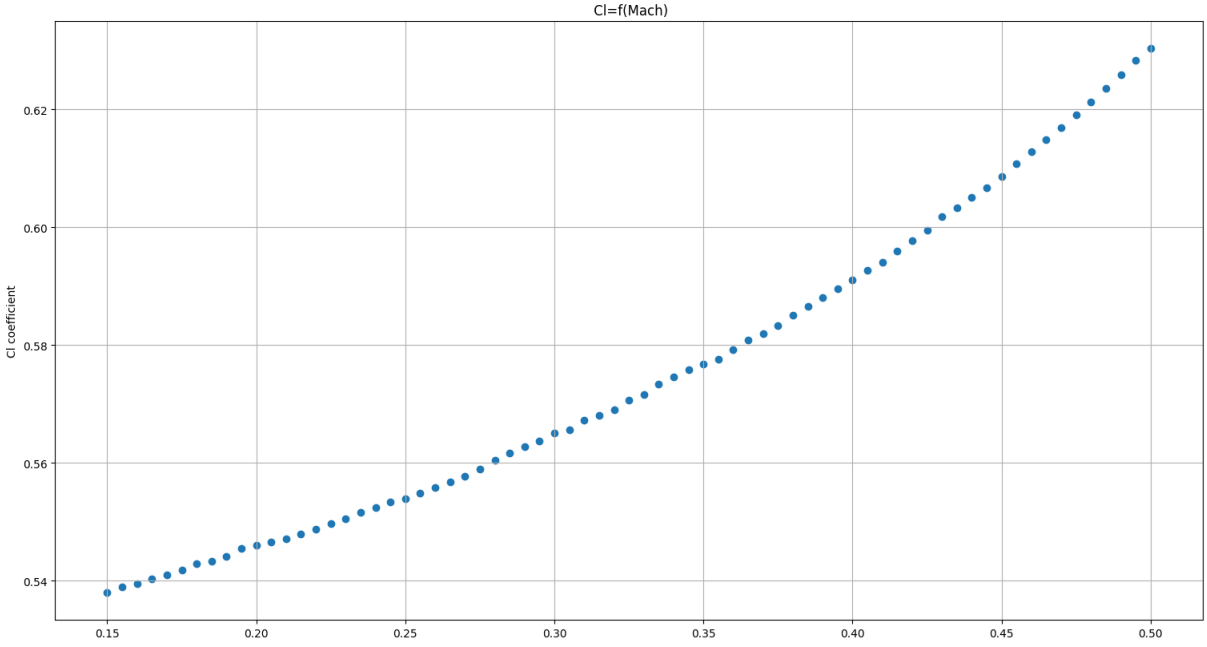


Figure 17: Reference values for the lift coefficient

Impact of sampling method

We noted analogous trends as observed for the angle of attack parameter. Specifically, Latin Hypercube Sampling outperformed random sampling in the majority of the scenarios examined. Consequently, we will adopt it as the default sampling method in all subsequent analyses.

Impact of expansion order and oversampling ratio

As before, we computed the L^2 relative error of C_l as a function of both the expansion order and the oversampling ration. We obtained the following plots :

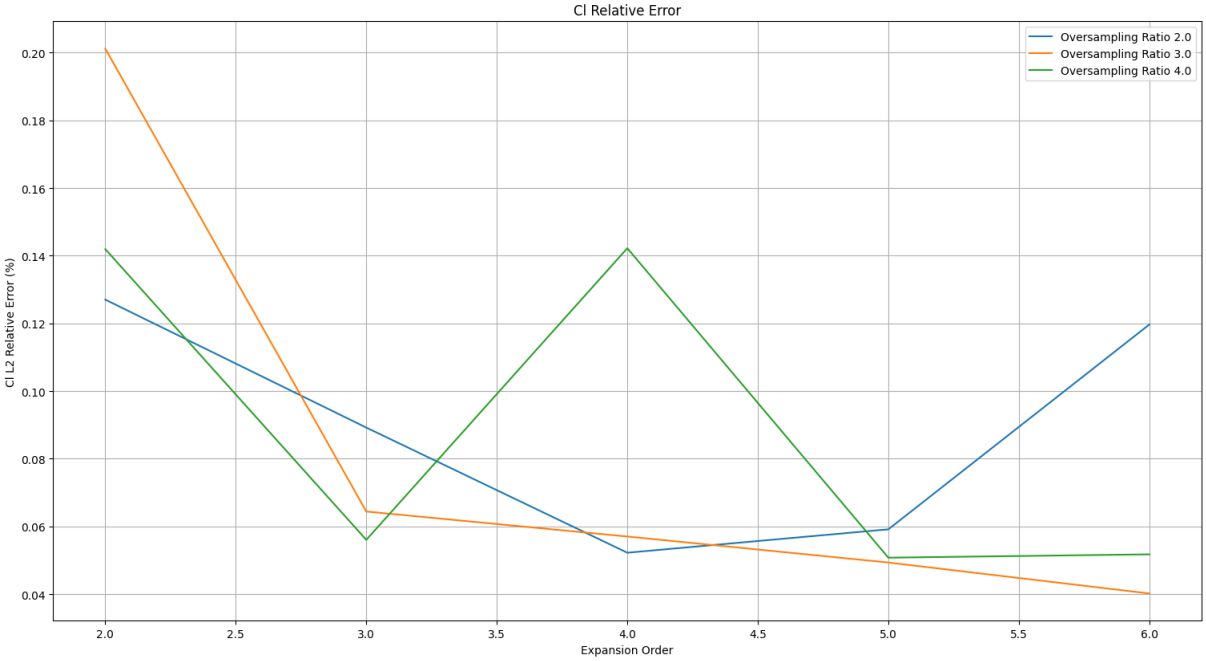


Figure 18: Cl L2 relative error vs Expansion Order

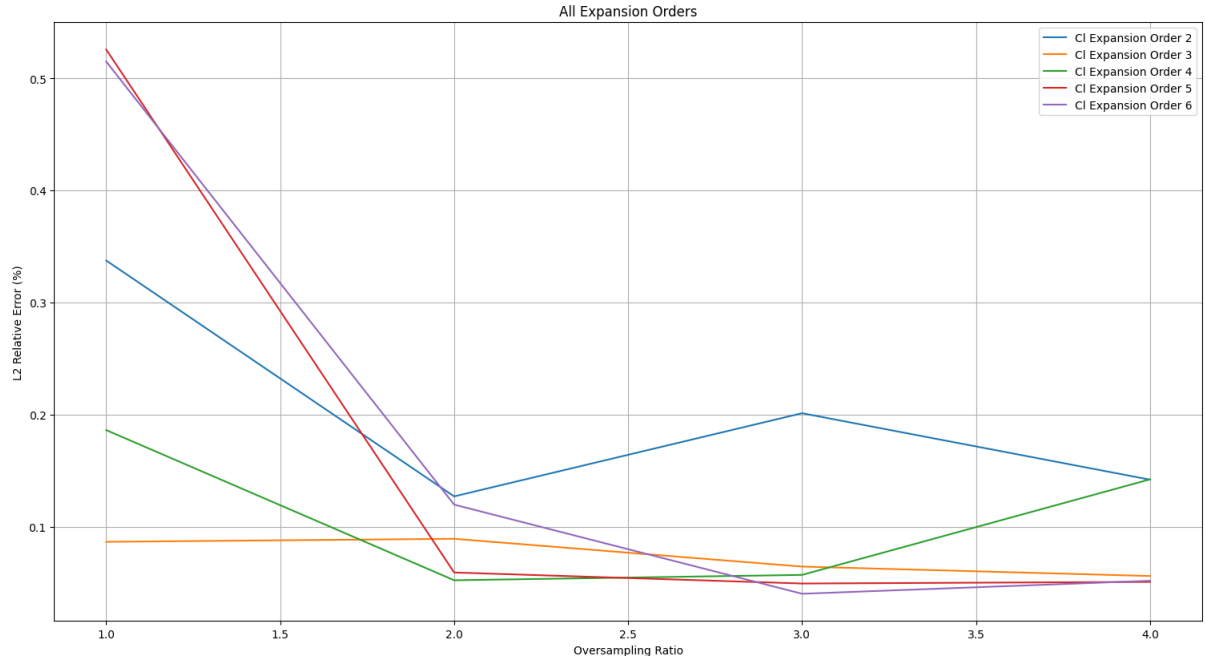


Figure 19: Cl L2 relative error vs Oversampling Ratio

Similar observations to those made previously can be made here. In Figure 18, a general trend is apparent where the error decreases as the expansion order increases for a given oversampling ratio. However, we also observe anomalies, which stem from the sampling of Mach numbers outside the domain of study, as illustrated in Figure 20.

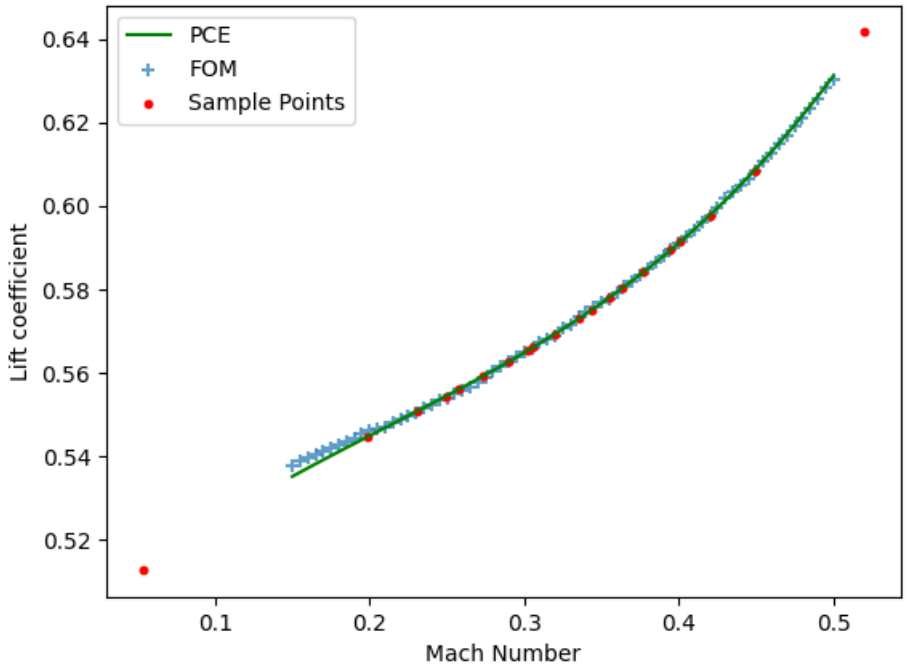


Figure 20: PCE obtained with an expansion order of 4 and an oversampling ratio of 4

In Figure 19, a decrease in error is also evident with the oversampling ratio. It is worth noting that beyond an oversampling ratio of 2, the reduction in error becomes less pronounced. Additionally, the computational time required significantly increases due to the higher number of points at which the FOM must be evaluated. Moreover, the expansion order appears to have a more significant impact on reducing the error compared to the oversampling ratio.

Impact of fitting method

Following a similar approach as before, we concluded that when properly tuned, the Lasso regression method enabled us to achieve better results than the least squares minimization method.

3.3.5 Two uncertain parameters : Angle of Attack and Mach number

Finally, we attempted to implement the NIPCE method while considering two uncertain input parameters: the angle of attack and the Mach number. Once again, we examined the convergence of the full-order model for various pairs of values and decided to restrict our analysis to angles between -8° and 8° and Mach numbers between 0.15 and 0.4. The lift coefficient remains the sole output variable of interest in this case. We opted for two normal distributions: the angle of attack will follow a normal distribution centered at 0° with a standard deviation of 3°, while the Mach number will follow a normal distribution centered at 0.275 with a standard deviation of 0.06.

The implementation of the NIPCE method in Python remains largely unchanged, utilizing the Chaospy library. However, now we must handle joint probability distributions, which Chaospy manages automatically. Additionally, we established a reference dataset containing 200 values of lift coefficients, represented below:

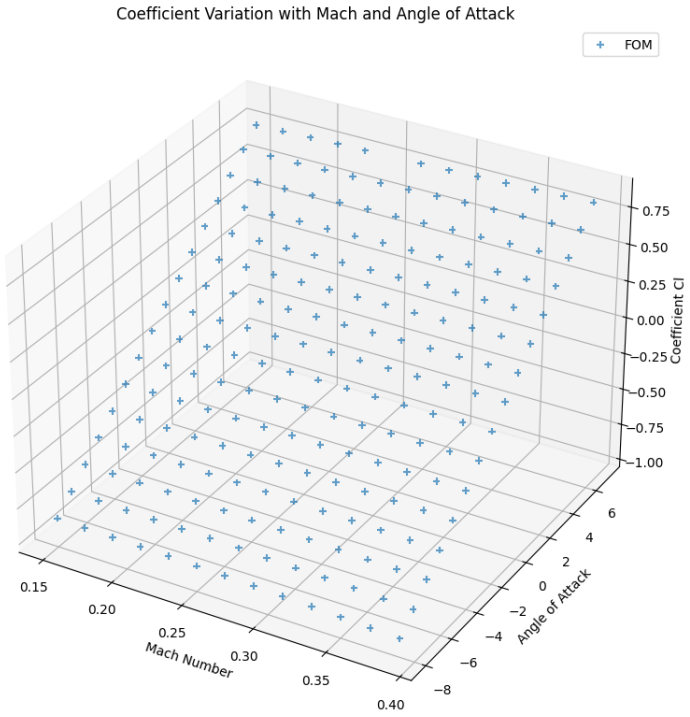


Figure 21: Reference values of the lift coefficient

From Figure 22, it appears that the angle of attack exerts a significantly greater influence on the lift coefficient compared to the Mach number.

Unfortunately, we did not have the time to conduct the same extensive study as before. However, we were able to demonstrate once again the efficiency of the NIPCE method in approximating the FOM with a minimal number of sample evaluations. Below is the PCE obtained with an expansion order of 3 and an oversampling ratio of 1, utilizing Latin Hypercube Sampling and the least squares minimization:

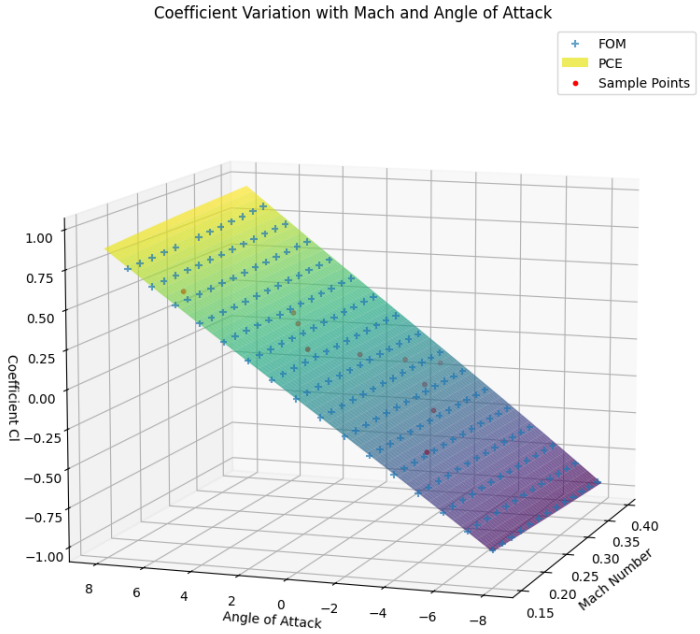


Figure 22: PCE obtained for an expansion order of 3 and an oversampling ratio of 1

It is worth noting that with two uncertain input parameters, the number of sample evaluations required for the point collocation technique is significantly higher than before, resulting in additional

computational costs. The oversampling ratio can be calculated as follows,

$$\text{Oversampling ratio} = \frac{\text{number of samples}}{P+1}$$

$$\text{where } P+1 = \frac{(n+p)!}{n!p!}$$

where p is the polynomial chaos expansion order and $n = 2$ the dimension of the space of uncertain parameters. Hence, in the previous example, with $p = 3$ and an oversampling ratio of 1, the number of samples is 10.

4 Conclusion

This work explored the theoretical underpinnings and practical applications of the Polynomial Chaos Expansion (PCE) method for Uncertainty Quantification (UQ) within stochastic Computational Fluid Dynamics (CFD) simulations, leveraging OpenFOAM software in conjunction with Python. This investigation is crucial due to the inherent variability of real-world input parameters and measurements, highlighting the indispensable role of UQ in ensuring the reliability and robustness of engineering projects.

At first, the study focused on the Non-Intrusive Polynomial Chaos Expansion (NIPCE) method applied to the lid-driven cavity problem. The promising results from this initial foray underscored the potential of further analysis. Notably, the integration of the Chaospy library with OpenFOAM for CFD simulations demonstrated both low error margins and efficient implementation.

Subsequently, we applied the Non-Intrusive Polynomial Chaos Expansion (NIPCE) method to the dynamic airflow characterization around a NACA0012 airfoil. Having once again confirmed the effectiveness of this method in building solid approximate models, we conducted a parametric investigation aimed to evaluate the method's parameters influence on the accuracy of the surrogate model obtained in comparison to the Full Order Model (FOM) outcomes. Our findings indicate that Latin Hypercube sampling consistently outperforms random sampling in achieving superior results. Furthermore, with appropriate calibration, Lasso regression demonstrated enhanced performance over least square minimization. As anticipated, an increase in the expansion number or oversampling ratio necessitated greater computational resources but generally resulted in improved outcomes. Notably, the polynomial order exerted a more significant influence than the oversampling ratio on the quality of results.

However, it became evident that the effectiveness of our approach heavily relies on the chosen FOM. Specifically, the OpenFOAM solver encountered convergence difficulties at high Mach numbers or angles of attack, impacting the reliability of our findings. This critical insight underscores the importance of considering the limitations of the FOM in our analysis.

In the final phase of our research, we extended the NIPCE method to accommodate two uncertain parameters simultaneously. Potential directions for future work include deepening the investigation into dual-parameter uncertainties and assessing whether similar trends persist. Moreover, employing more robust FOM solvers with enhanced computational capabilities could mitigate the convergence challenges encountered, enabling a broader scope of analysis. While the current implementation already provides valuable insights, transitioning to object-oriented programming could further streamline the process, enhancing both clarity and usability—a step not reached due to time constraints.

In validating the NIPCE method's efficacy, we compared deterministic outcomes from surrogate models against those from the FOM, constrained by computational expenses. An alternative strategy would leverage the NIPCE method's capability for statistical post-processing, comparing these results against those derived from Monte Carlo simulations. Exploring the application of other libraries prevalent in the industry, such as OpenTurns, could also offer additional perspectives. Beyond the current methodologies, integrating machine learning techniques like Gaussian Process Modeling into UQ holds promising prospects. This approach could herald new avenues for precision and efficiency in stochastic analysis, meriting further investigation.

5 References

[1] Ghia, UKNG, Kirti N Ghia, and CT Shin: *High-re solutions for incompressible flow using the navier-stokes equations and a multigrid method.* Journal of computational physics, 48(3):387–411, 1982.

[2] Eldred, Michael and John Burkardt: *Comparison of non-intrusive polynomial chaos and stochastic collocation methods for uncertainty quantification.* In *47th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition*, page 976, 2009.

[3] Walters, Robert W and Luc Huyse: *Uncertainty analysis for fluid mechanics with applications.* 2002.

[4] Jonathan Feinberg, Hans Petter Langtangen: *Chaospy: An open source tool for designing methods of uncertainty quantification.* Journal of Computational Science, 11:46–57, 2015.

[5] Michaël Baudin, Anne Dutfoy, Bertrand Iooss and Anne Laure Popelin: *Openturns: An industrial software for uncertainty quantification in simulation.* 2017.

[6] Olivier, Audrey, Dimitris G Giovanis, BS Aakash, Mohit Chauhan, Lohit Vandanapu, and Michael D Shields: *Uqpy: A general purpose python package and development environment for uncertainty quantification.* Journal of Computational Science, 47:101204, 2020.

[7] Dalbey, Keith R., Michael S. Eldred, Gianluca Geraci, John D. Jakeman, Kathryn A. Maupin, Jason A. Monschke, Daniel Thomas Seidl, Anh Tran, Friedrich Menhorn, and Xiaoshu Zeng: *Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 6.16 theory manual.* `https://www.osti.gov/biblio/1868423`.

[8] Weller, Henry G, Gavin Tabor, Hrvoje Jasak, and Christer Fureby: *A tensorial approach to computational continuum mechanics using object-oriented techniques.* Computers in physics, 12(6):620–631, 1998.

[9] Hosder, Serhat and Robert W Walters: *Non-intrusive polynomial chaos methods for stochastic cfd-theory and applications.* In *RTO Meeting Proceedings. Computational Uncertainty in Military Vehicle Design*, 2007.

[10] Hosder, Serhat, Robert Walters, and Rafael Perez: *A non-intrusive polynomial chaos method for uncertainty propagation in cfd simulations.* In *44th AIAA aerospace sciences meeting and exhibit*, page 891, 2006.

[11] Ghanem, Roger and Pol D Spanos: *Polynomial chaos in stochastic finite elements.* 1990.

[12] Xiu, Dongbin and George Em Karniadakis: *Modeling uncertainty in flow simulations via generalized polynomial chaos.* Journal of computational physics, 187(1):137–167, 2003.

[13] Mathelin, Lionel, M Yousuff Hussaini, Thomas A Zang, and Francoise Bataille: *Uncertainty propagation for a turbulent, compressible nozzle flow using stochastic methods.* AIAA journal, 42(8):1669–1676, 2004.

[14] Xiu, Dongbin and George Em Karniadakis: *The wiener–askey polynomial chaos for stochastic differential equations.* SIAM journal on scientific computing, 24(2):619–644, 2002.

[15] Xiu, Dongbin, Didier Lucor, C H Su, and George Em Karniadakis: *Stochastic modeling of flow-structure interactions using generalized polynomial chaos.* J. Fluids Eng., 124(1):51–59, 2002.

[16] Xiu, Dongbin and George Em Karniadakis: *Modeling uncertainty in steady state diffusion problems via generalized polynomial chaos.* Computer methods in applied mechanics and engineering, 191(43):4927–4948, 2002.

[17] Le Maître, Olivier and Omar M Knio: *Spectral methods for uncertainty quantification: with applications to computational fluid dynamics*. Springer Science & Business Media, 2010.

[18] Cameron, Robert H and William T Martin: *The orthogonal development of non-linear functionals in series of fourier-hermite functionals*. Annals of Mathematics, pages 385–392, 1947.

[19] Hosder, Serhat, Robert W Walters, and Michael Balch: *Efficient sampling for non-intrusive polynomial chaos applications with multiple uncertain input variables*. 2007.

[20] Foundation, The OpenFOAM: *User Guide: icoFoam*, 2018. `https://openfoamwiki.net/index.php/IcoFoam`, visited on December 19, 2023.

[21] Issa, Raad I: *Solution of the implicitly discretised fluid flow equations by operator-splitting*. Journal of computational physics, 62(1):40–65, 1986.

[22] Foundation, The OpenFOAM: *OpenFOAM lid-driven cavity flow tutorial*. `https://www.openfoam.com/documentation/tutorial-guide/2-incompressible-flow/2.1-lid-driven-cavity-flow`, visited on December 19, 2023.

[23] Cortes, AB and JD Miller: *Numerical experiments with the lid driven cavity flow problem*. Computers & fluids, 23(8):1005–1027, 1994.

[24] McKay, M. D., R. J. Beckman, and W. J. Conover: *A comparison of three methods for selecting values of input variables in the analysis of output from a computer code*. Technometrics, 21(2):239–245, 1979, ISSN 00401706. `http://www.jstor.org/stable/1268522`, visited on 2024-02-28.

[25] Shields, Michael D and Jiaxin Zhang: *The generalization of latin hypercube sampling*. Reliability Engineering & System Safety, 148:96–108, 2016.

[26] Ladson, Charles L.: *Nasa technical memorandum 4074 : Effects of independent variation of mach and reynolds numbers on the low-speed aerodynamic characteristics of the naca 0012 airfoil section*. 1988.

[27] Walters, Robert: *Towards stochastic fluid mechanics via polynomial chaos*. In *41 st AIAA Aerospace Sciences Meeting & Exhibit, Reno, NV*, 2003.

[28] Hijazi, Saddam, Giovanni Stabile, Andrea Mola, and Gianluigi Rozza: *Non-intrusive polynomial chaos method applied to full-order and reduced problems in computational fluid dynamics: A comparison and perspectives*. Quantification of Uncertainty: Improving Efficiency and Technology: QUIET selected contributions, pages 217–240, 2020.

[29] Debusschere, Bert J, Habib N Najm, Philippe P Pébay, Omar M Knio, Roger G Ghanem, and Olivier P Le Maı tre: *Numerical challenges in the use of polynomial chaos representations for stochastic processes*. SIAM journal on scientific computing, 26(2):698–719, 2004.

[30] Preece, Robin and Jovica V Milanović: *Efficient estimation of the probability of small-disturbance instability of large uncertain power systems*. IEEE Transactions on Power Systems, 31(2):1063–1072, 2015.