University of Verona

DEPARTMENT OF COMPUTER SCIENCE
Master Degree in Mathematics

Polynomial Chaos Expansion with applications to PDEs

Candidate:
Gregorio Pellegrini
VR368398

Thesis advisor:
Marco Caliari PhD

Thesis co-advisor:
Manolo Venturin PhD
EnginSoft S.p.A.

a.y. 2013/2014

Personal Information

For any question, suggestion or information please contact Gregorio Pellegrini at

gregorio.pellegrini@gmail.com

Contents

Abstract **vi**

1 Orthogonal Polynomials **1**
1.1 Settings . . . 1
1.2 Legendre polynomials . . . 3
1.2.1 Definitions . . . 3
1.2.2 Properties . . . 6
1.3 Hermite polynomials . . . 8
1.3.1 Definitions . . . 8
1.3.2 Properties . . . 10
1.4 Properties of orthogonal maximal systems . . . 14

2 Polynomial Chaos Expansion **17**
2.1 One dimensional Polynomial Chaos Expansion . . . 17
2.2 Properties of gPC basis and PCE . . . 20
2.2.1 Example: decomposition of a Lognomal random variable . . . 21
2.2.2 Weak convergence . . . 22
2.3 Multidimensional gPC basis and PCE . . . 23
2.4 PCE of random vectors . . . 27
2.5 A multi-element approach . . . 27
2.5.1 Decomposing the range of a scalar random variable . . . 27
2.5.2 A Multi-Element Polynomial Chaos . . . 30

3 Non-intrusive Spectral Projection **33**
3.1 Motivation: Uncertainty Quantification . . . 33
3.2 Univariate NISP approach . . . 34
3.3 Univariate quadrature formula . . . 35
3.4 Scilab's NISP toolbox . . . 35
3.5 NISP toolbox features . . . 36
3.5.1 Definition of the model . . . 36
3.5.2 Problem data definition . . . 36
3.5.3 Design of Experiment (DOE) . . . 36
3.5.4 Polynomial Chaos definition and computation . . . 37
3.5.5 Post process analysis . . . 38
3.6 Multivariate NISP approach . . . 40
3.7 NISP approach for output random vector . . . 42

4 Application of NISP toolbox **43**
4.1 Bimodal . . . 43
4.1.1 Arcsine distribution . . . 43
4.1.2 PCE decomposition of an arcsine distribution . . . 43
4.1.3 Mixture of random variables . . . 45
4.1.4 PCE for Mixture of two normal random variables . . . 46
4.2 Non-linear ODE . . . 48
4.2.1 Analytical solution . . . 49

4.2.2 First order ODE . . . 50
4.2.3 PCE of the output . . . 51
4.2.4 A null . . . 52
4.2.5 A negative . . . 54
4.2.6 A positive . . . 56
4.3 Lid-Driven Cavity . . . 59
4.3.1 Freefem++ solver . . . 59
4.3.2 PCE for lid-driven cavity . . . 60
4.4 Pyclaw . . . 62
4.5 Conclusions . . . 68

A General topics: random variables, histograms and softwares 69
A.1 Equally distributed random variables . . . 69
A.2 Histogram of multi-element PCE . . . 69
A.3 Scilab's shell (sh) command execution . . . 70
A.4 Installation of Pycalw . . . 72
A.5 On vector ordering: Fortran, C++ and Scilab . . . 73

Abstract

In this thesis polynomial chaos expansion (PCE) is studied for both univariate and multivariate cases. This technique recovers a finite second order random variable by means of a linear combination of orthogonal polynomials, whose entries are a selected class of random variables called *germs* or *basic random variables*. The choice of these entries characterizes the orthogonality property achieved by these functional polynomials.

This approach approximates either a random variable of interest or the output of a process in presence of uncertain inputs. Moreover PCE belongs to Uncertainty Quantification (UQ) methods, which aim to define suitable theoretical background and reliable numerical methods in order to consider the effects of uncertainties in simulations. The classical Monte Carlo approach constitutes the first example of such methodology, but it requires lots of simulations to get reliable results, while polynomial chaos expansion is an efficient alternative to overwhelm low convergence rate.

The coefficients of PCE are detected via Non-Intrusive Spectral Projection (NISP), that uses suitable simulations of the process to compute such values. While Monte Carlo requires lots of evaluations to have appropriate convergence, NISP achieves spectral convergence with few simulations. Since the process is run for fixed realizations of the input parameters, it is not recast into a probabilistic framework. This motivates the non-intrusive nature of such approach.

Such theory is applied to four examples which concern: the decomposition of two bimodal random variables, the analysis of the solution for a non-linear ordinary differential equation (ODE), when random physical parameter is considered, and the description of the solutions of two partial differential equations (PDEs) in presence of uncertain parameters.

Their study and analysis were made during an internship at EnginSoft S.p.A., a consulting firm in the field of Simulation Based Engineering Science (SBES). This was the starting point for investigating the theoretical properties of PCE in order to describe the features and advantages of NISP approach.

The presented routines are implemented in Scilab environment, which is a free and open source software for numerical computation providing powerful computing tools for engineering and scientific applications.

On the other hand the data, used for detecting the spectral projection, are computed via functions implemented within Scilab and with two open source softwares: FreeFem++ a partial Differential Equation (PDE) solver based on the Finite Element Method and Pyclaw, a hyperbolic PDE solver based on Finite Volume Method. These situations cover most of the possible type of data sources.

Scilab functions represent the possibility to define all features of the solver considered, while FreeFem++ scripts allow to define only the basic properties of the solver, such as the type space on which the solution is recovered and the boundary conditions; but it does not explicitly show how the solution is computed. Pyclaw is an example of a black-box: the solver and the problem are already implemented, only marginal data can be customized by the user, such as physical quantities and the size of the mesh.

The final results of this thesis concern both theoretical issues and applications. The most important theoretical features of PCE are deeply analyzed, keeping particular attention on the

relation between the basic random variables and the quantity that one wishes to recover. Moreover the multi-element approach described makes use of the already defined PCE (it is not followed the technique developed by Wan and Karniadakis in [17], that implies the definition of new orthogonal polynomials).

The presented examples prove the flexibility of NISP technique, indeed it is applied both to decomposition of random variables and to approximation of the solution of the two most common classes of differential equations, ODEs and PDEs. Moreover NISP interfaces with three possible data sources, described above.

As last remark it is pointed out, especially in the last example, how NISP interacts with respect to deterministic simulation, highlighting the importance of taking into account the uncertainties into model simulations.

Chapter 1

Orthogonal Polynomials

In this chapter definitions, properties and approximation results of Legendre and Hermite orthogonal polynomials are discussed. Moreover only univariate setting is considered. The detection of multivariate orthogonal polynomials is based on section II in [11], that makes use of tensor products of univariate orthogonal families.

1.1 Settings

Orthogonal polynomials are defined in Hilbert spaces, since the notion of orthogonality is well defined by means of scalar product. The univariate real-valued polynomials are defined on a real interval D. Moreover they belong to $L^2(D, w(x)dx)$, where $w(x)$ is the weight function that characterizes the orthogonality condition. The class of Legendre polynomials is defined on $D = [-1, 1]$ and for

$$w(x) = 1/2$$

while for Hermite class $D = \mathbb{R}$ and the weight function is of kind

$$w(x) = e^{-\alpha^2 x^2}$$

where α is a positive scaling factor. Their definition in terms of this parameter α is useful for some applications as described in Tang [5]. With suitable values of α the two most common classes of Hermite polynomials are achieved, namely the physicists' and probabilists' Hermite polynomials. They are respectively characterized by

$$w(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \qquad w(x) = \frac{1}{\sqrt{\pi}} e^{-x^2}$$

For any $f, g \in L^2(D, w(x)dx)$, the scalar product is characterized by the weight function $w(x)$

$$\langle f, g \rangle = \int_D f(x)g(x)w(x)dx \tag{1.1}$$

usually it is referred as $\langle \cdot, \cdot \rangle_w$, but to shorten the notation, it would be the contest that characterizes the weight function involved.

Once the scalar product is set, the orthogonal polynomials are computed via Gram-Schmidt orthogonalization procedure.

Definition 1.1 *The Gram-Schmidt orthogonalization procedure in $L^2(D, w(x)dx)$ applied to the basis B of polynomials of degree at most n*

$$B = \{1, x, \ldots, x^n\}$$

defines an orthogonal family of monic polynomials $\mathcal{H}_n = \{\Psi_i\}_{i=0}^n$ which are linear independent elements in $L^2(D, w(x)dx)$.

The Gram-Schmidt algorithm is defined recursively, by setting $\Psi_0 = 1$, and then, for each natural index $i > 0$

$$\Psi_1 = x - \alpha_{10}\Psi_0$$
$$\vdots$$
$$\Psi_i = x^i - \sum_{j=0}^{i-1} \alpha_{ij}\Psi_j$$

where the coefficients α_{ij} are defined as

$$\alpha_{ij} = \frac{\langle x^i, \Psi_j \rangle}{\langle \Psi_j, \Psi_j \rangle} \tag{1.2}$$

Let us denote with $\mathcal{H}$ the basis defined via Gram-Schmidt procedure. Following the approach presents by Gautschi in [3], some properties of general orthogonal system of polynomials can be stated. First let us define $\mathbb{P}$ as the space of real-valued univariate polynomials.

Theorem 1.1 *If the inner product is positive definite on $\mathbb{P}$, there exists a unique sequence $\{\Psi_j\}_{j\in\mathbb{N}}$ of monic orthogonal polynomials.*

Proof. For each $i \in \mathbb{N}$ *Gram-Schmidt orthogonalization procedure* defines

$$\Psi_i = x^i - \sum_{j=0}^{i-1} \alpha_{ij}\Psi_j$$

By equation (1.2) the coefficients are always defined, and the algorithm detects uniquely these orthogonal polynomials. □

Since the scalar product of $L^2(D, w(x)dx)$ is positive definite for all functions in $L^2(D, w(x)dx)$, the assumption of the previous theorem is always fulfilled, thus unique class of orthogonal and monic polynomials are defined for each weight function.

Lemma 1.1 *If the weight function $w(x)$ is symmetric about y-axis and it is defined on a symmetric domain D, then the j-th orthogonal polynomial Ψ_j satisfies*

$$\Psi_j(-x) = (-1)^j \Psi_j(x)$$

Thus being even or odd polynomial depends on the parity of j.

Proof. Let $\hat{\Psi}_j := (-1)^j \Psi_j$ for all indexes $j \in \mathbb{N}$, then by computing the inner product

$$\left\langle \hat{\Psi}_j, \hat{\Psi}_i \right\rangle = (-1)^{i+j} \langle \Psi_j, \Psi_i \rangle = \delta_{ij} \|\Psi_i\|^2$$

hence $\{\hat{\Psi}_j\}_{j\in\mathbb{N}}$ is another family of polynomials orthogonal in $L^2(D, w(x)dx)$. By Theorem 1.1 for all $j \in \mathbb{N}$, $\hat{\Psi}_j = \Psi_j$, hence

$$\Psi_j(-x) = (-1)^j \Psi_j(x)$$

□

Definition 1.2 *If $\{u_n\}_{n\in\mathbb{N}}$ is a sequence of real values, then its generating function is a real function $u(t)$ such that*

$$u(t) = \sum_{n=0}^{\infty} u_n t^n$$

for $|t| < R$, where R is the radius of convergence of the series.

Example. Let us consider a geometric sequence $u_n = a^n$, then

$$\sum_{n=0}^{\infty} a^n t^n = \frac{1}{1-at}$$

with radius of convergence

$$R = \frac{1}{|a|}$$

Proposition 1.1 *Let $\{u_k\}_{k\in\mathbb{N}}$ be a sequence whose generating function is $u(t)$ and let m be a positive integer, then for $t \neq 0$ and $|t| < R$*

$$\sum_{k=0}^{\infty} u_{k+m} t^k = \frac{1}{t^m}\left[u(t) - \sum_{j=0}^{m-1} u_j t^j\right]$$

Proof. By the very definition of *generating function* we get

$$u(t) = \sum_{j=0}^{\infty} u_j t^j = \sum_{j=0}^{m-1} u_j t^j + \sum_{j=m}^{\infty} u_j t^j$$

By substituting $k = j - m$

$$u(t) - \sum_{j=0}^{m-1} u_j t^j = t^m \sum_{k=0}^{\infty} u_{k+m} t^k$$

and by dividing both term of the equation by t^m, we get

$$\frac{1}{t^m}\left[u(t) - \sum_{j=0}^{m-1} u_j t^j\right] = \sum_{k=0}^{\infty} u_{k+m} t^k$$

□

1.2 Legendre polynomials

Univariate Legendre polynomials are usually defined on the compact set $D = [-1, 1]$, and they fulfill the orthogonal condition with respect to scalar product defined in (1.1), where the weight function is

$$w(x) = \frac{1}{2}$$

1.2.1 Definitions

By Definition 1.1 Legendre polynomials are detected via Gram-Schmidt orthogonalization procedure. Notice that the weight function as well as the domain satisfies the assumptions of Lemma 1.1, thus it is possible to simplify the computations of coefficients α_{ij} in Gram-Schmidt algorithm. Indeed

$$\langle x^i, P_j \rangle = \int_{-1}^{1} x^i P_j \frac{1}{2} dx$$

it is null if i and j have different parities. Moreover by equation (1.2) for the same indexes i and j, the coefficients α_{ij} are null. Applying these strategies, the first four monic Legendre

polynomials are easily detected

$$
\begin{aligned}
&P_0 = 1\\
&P_1 = x - \alpha_{10}P_0 = x\\
&P_2 = x^2 - \alpha_{21}P_1 - \alpha_{20}P_0 = x^2 - \alpha_{20}P_0 = x^2 - \frac{1}{3}\\
&P_3 = x^3 - \alpha_{32}P_2 - \alpha_{31}P_1 - \alpha_{30}P_0 = x^2 - \alpha_{31}P_1 = x^3 - \frac{3}{5}x\\
&P_4 = x^4 - \alpha_{43}P_3 - \alpha_{42}P_2 - \alpha_{41}P_1 - \alpha_{40}P_0 = x^4 - \alpha_{42}P_2 - \alpha_{40}P_0 = x^4 - \frac{6}{7}x^2 + \frac{3}{35}
\end{aligned}
$$

This definition of Legendre polynomials is not the common one, usually orthogonality is described as a property of such family, not the main issue that allows to define them. Therefore let us investigate the equivalences between the most common definitions

Theorem 1.2 *Let $\{P_n\}_{n\in\mathbb{N}}$ be a class of polynomials in $L^2([-1,1], w(x)dx)$. The following statements are equivalent*

(i) The family satisfies the recurrence relation

$$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x) \tag{1.3}$$

with $P_0 = 1$ and $P_1 = x$

(ii) the generating function is

$$\frac{1}{\sqrt{1-2xt+t^2}} = \sum_{n=0}^{\infty} P_n(x)t^n \tag{1.4}$$

for $x \in [-1,1]$ and $|t| \leq 1$.

Such family $\{P_n\}_{n\in\mathbb{N}}$ is the collection of Legendre polynomials.

Proof.

$(i) \implies (ii)$

Let $x \in [-1,1]$ be a fixed value. Then $Q_x(t)$ denotes the *generating function* of the sequence of real values $\{P_n(x)\}_{n\in\mathbb{N}}$. By Definition 1.2

$$Q_x(t) = \sum_{n=0}^{\infty} P_n(x)t^n$$

then it is derived term by term, with respect to t and the result is multiplied by t, getting

$$tQ'_x(t) = \sum_{n=1}^{\infty} nP_n(x)t^n$$

Let us multiply the recurrence relation (1.3) by t^{n+1} and sum it from $n = 1$ to $n = \infty$

$$\sum_{n=1}^{\infty}(n+1)P_{n+1}(x)t^{n+1} = \sum_{n=1}^{\infty}\Big[(2n+1)xP_n(x) - nP_{n-1}(x)\Big]t^{n+1}$$

By expanding the summands on the right-hand side

$$\sum_{n=1}^{\infty}(n+1)P_{n+1}(x)t^{n+1} = 2xt\sum_{n=1}^{\infty}nP_n(x)t^n + \sum_{n=1}^{\infty}xP_n(x)t^{n+1} - \sum_{n=1}^{\infty}nP_{n-1}(x)t^{n+1}$$

After relabeling $m = n + 1$ on left-hand side and by setting $k = n - 1$ to the last two summands of the right-hand side, the equation can be recast as

$$\begin{aligned}\sum_{m=2}^{\infty} mP_m(x)t^m &= 2xt\sum_{n=1}^{\infty} nP_n(x)t^n + \sum_{k=0}^{\infty} xP_{k+1}(x)t^{k+2} - \sum_{k=0}^{\infty}(k+1)P_k(x)t^{k+2} = \\ &= 2xt\sum_{n=1}^{\infty} nP_n(x)t^n + t^2\left[\sum_{k=0}^{\infty} xP_{k+1}(x)t^k - \sum_{k=0}^{\infty}(k+1)P_k(x)t^k\right] = \\ &= 2xt\sum_{n=1}^{\infty} nP_n(x)t^n + t^2\sum_{k=0}^{\infty}\left[xP_{k+1}(x)t^k - (k+1)P_k(x)t^k\right]\end{aligned}$$

The equation achieved is

$$\sum_{m=2}^{\infty} mP_m(x)t^m = 2xt\sum_{n=1}^{\infty} nP_n(x)t^n + t^2\sum_{k=0}^{\infty}\left[xP_{k+1}(x)t^k - (k+1)P_k(x)t^k\right] \tag{1.5}$$

The summand of previous equation can be defined in terms of $Q_x(t)$ and its derivative

$$\sum_{m=2}^{\infty} mP_m(x)t^m = tQ'_x(t) - P_1(t)x = tQ'_x(t) - xt$$

and

$$\sum_{k=0}^{\infty}(k+1)P_k(x)t^k = tQ'_x(t) + Q_x(t)$$

while

$$t\sum_{k=0}^{\infty} P_{k+1}(x)t^k = Q_x(t) - P_0(t) = Q_x(t) - 1$$

by applying Proposition 1.1. Thus let us plug these results into equation (1.5)

$$tQ'_x(t) - tx = 2xt^2Q'_x(t) + tx(Q_x(t) - 1) - t^2\big(tQ'_x(t) + Q_x(t)\big)$$

that yields to

$$Q'_x(t) = -\frac{t-x}{1-2tx+t^2}Q_x(t)$$

By integrating the differential equation with $Q_x(0) = P_0(x) = 1$ as initial condition, the explicit formula for $Q_x(t)$ is achieved

$$\frac{1}{\sqrt{1-2xt-t^2}} = Q_x(t) = \sum_{n=0}^{\infty} P_n(x)t^n$$

which converges whenever $|t| \leq 1$.

$(ii) \implies (i)$

Let us differentiate the equation (1.4) with respect to t

$$\frac{x-t}{(1-2xt-t^2)^{3/2}} = \sum_{n=0}^{\infty} nP_n(x)t^{n-1}$$

It can be done term by term, since $|t| \leq 1$. Multiplying both sides by $1 - 2xt + t^2$ and recognizing the generating function times $(x - t)$ in the left-hand side, it yields to

$$(x-t)\sum_{n=0}^{\infty} P_n(x)t^n = (1-2xt+t^2)\sum_{n=0}^{\infty} nP_n(x)t^{n-1}$$

expanding all products

$$\sum_{n=0}^{\infty} xP_n(x)t^n - \sum_{n=0}^{\infty} P_n(x)t^{n+1} = \sum_{n=0}^{\infty} nP_n(x)t^{n-1} - \sum_{n=0}^{\infty} 2nxP_n(x)t^n + \sum_{n=0}^{\infty} nP_n(x)t^{n+1}$$

Then by equating the coefficients of each t^n term and by algebraic simplifications we get

$$(2n+1)xP_n(x) - nP_{n-1}(x) = (n+1)P_{n+1}(x)$$

□

The easiest way to get the class of Legendre polynomials is using the recurrence formula in (1.3), hence

$$\begin{aligned} P_0(x) &= 1 \\ P_1(x) &= x \\ P_2(x) &= \frac{1}{2}\Big(3x^2 - 1\Big) \\ P_3(x) &= \frac{1}{2}\Big(5x^3 - 3x\Big) \\ P_4(x) &= \frac{1}{8}\Big(35x^4 - 30x^2 + 3\Big) \end{aligned}$$

At first Legendre polynomials were defined as the unique class of polynomials that is orthogonal with respect the the scalar product (1.1). Hence the family $\{P_n\}_{n\in\mathbb{N}}$ defined by Theorem 1.2 coincides, up to a scalar factor, with the one coming from Definition 1.1, if orthogonality property is proven.

1.2.2 Properties

Proposition 1.2 *Let us consider the class of Legendre polynomials $\{P_n\}_{n\in\mathbb{N}}$, then for all $n, m \in \mathbb{N}$*

$$\int_{-1}^{1} P_n(x)P_m(x)\frac{1}{2}dx = \delta_{nm}\frac{1}{2n+1}$$

Proof. The square of the equation (1.4) is

$$\frac{1}{1-2xt+t^2} = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} t^m t^n P_m(x)P_n(x)$$

Let us integrate from -1 to 1 both sides of previous equation. Notice that the right-hand side can be integrate term by term, by uniform convergence of power series for $x \in [-1, 1]$. Then developing the integral on left-hand side

$$-\frac{1}{2t}\Big[\ln(1-2tx+t^2)\Big]_{-1}^{1} = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} t^{m+n}\int_{-1}^{1} P_n(x)P_m(x)dx$$

Hence evaluating the left-hand side

$$-\frac{1}{t}\ln\left(\frac{1-t}{1+t}\right) = \sum_{m=0}^{\infty}\sum_{n=0}^{\infty} t^{m+n}\int_{-1}^{1} P_n(x)P_m(x)dx \tag{1.6}$$

Moreover since $|t| \leq 1$ the left-hand side of the previous equation can be replaced with the difference of the two Taylor expansions of $\ln(1+t)$ and $\ln(1-t)$.Thus the left-hand side becomes

$$-\frac{1}{t}\ln\left(\frac{1-t}{1+t}\right) = \frac{1}{t}\sum_{k=0}^{\infty}\frac{2t^{2n+1}}{2n+1} = \sum_{k=0}^{\infty}\frac{2t^{2n}}{2n+1} \tag{1.7}$$

By equating the coefficients of the right-hand sides of (1.6) and (1.7), two possibilities arise: if $n \neq m$

$$0 = \int_{-1}^{1} P_n(x) P_m(x) dx$$

If $m = n$

$$\frac{2}{2n+1} = \int_{-1}^{1} (P_n(x))^2 \, dx$$

then the lemma is proven by dividing both sides of the previous equations by two. □

The next step proves that the class of orthonormal Legendre polynomials constitutes a Hilbert basis for $L^2([-1,1], w(x)dx)$. The proof is based on the Theorem at page 74 of [1], where the maximality of a basis in Hilbert spaces is equivalent to density of the linear subspace spanned by the orthonormal family considered.

The proof requires two well known results. The first one is the Weierstrass approximation theorem.

Theorem 1.3 (Weierstrass) *Every continuous function on a closed interval* $[-1,1]$ *can be uniformly approximated by polynomials on* $[-1,1]$.

The uniform convergence ensures convergence in $L^2([-1,1], w(x)dx)$ for continuous functions. Indeed let $f \in \mathcal{C}^0 \cap L^2([-1,1], w(x)dx)$, by Weierstrass theorem for every real $\epsilon > 0$, there exists a polynomial p such that $\|f-p\|_\infty^2 < \epsilon/2$. Hence

$$\|f-p\|^2_{L^2([-1,1],w(x)dx)} = \int_{-1}^{1} (f-p)^2 \frac{1}{2} dx \leq \|f-p\|_\infty^2 < \frac{\epsilon}{2} \tag{1.8}$$

The second step is based on density theorem for Lebesgue spaces.

Theorem 1.4 *Let* $1 \leq p < +\infty$ *and* Ω *be an open set in* $\mathbb{R}$. *The set of continuous functions with compact support is dense in* $L^p(\Omega)$

Let us consider an open interval Ω in $\mathbb{R}$, such that $[-1,1] \subset \Omega$. Then any function $f \in L^2([-1,1], w(x)dx)$ can be extended to zero on Ω, defining $\overline{f} \in L^2(\Omega)$, indeed

$$\left\|\overline{f}\right\|^2_{L^2(\Omega)} = \|f\|^2_{L^2([-1,1])} = 2\|f\|^2_{L^2([-1,1],w(x)dx)} \tag{1.9}$$

By Theorem 1.4 for every real $\epsilon > 0$, there exists a continuous function $\overline{g}$ on Ω with compact support, such that

$$\left\|\overline{f} - \overline{g}\right\|^2_{L^2(\Omega)} < \epsilon \tag{1.10}$$

Let us define the continuous function $g(x)$ as the restriction of $\overline{g}$ to $[-1,1]$. By equation (1.9) the relation between the two norms in $L^2([-1,1])$ and $L^2([-1,1], w(x)dx)$ implies that

$$\|f-g\|^2_{L^2([-1,1])} = 2\|f-g\|^2_{L^2([-1,1],w(x)dx)} \tag{1.11}$$

Moreover $\left\|\overline{f} - \overline{g}\right\|^2_{L^2(\Omega)} \geq \|f-g\|^2_{L^2([-1,1])}$, thus combining this observation with results in (1.10) and (1.11) for a continuous function g

$$\|f-g\|^2_{L^2([-1,1],w(x)dx)} < \frac{\epsilon}{2} \tag{1.12}$$

Let us consider the polynomial p, that comes from Weierstrass theorem upon setting the same real $\epsilon/2 > 0$ of equation (1.12). By triangular inequality

$$\|f-p\|^2_{L^2([-1,1],w(x)dx)} \leq \|f-g\|^2_{L^2([-1,1],w(x)dx)} + \|g-p\|^2_{L^2([-1,1],w(x)dx)} \tag{1.13}$$

the first summand of right-hand side is already estimated in (1.12), as well as the second one by means of observation in (1.8). Therefore

$$\|f - p\|^2_{L^2([-1,1],w(x)dx)} < \epsilon \tag{1.14}$$

Furthermore let us call n the degree of p. The Gram-Schmidt orthogonalization procedure ensures that the linear spaces on $\mathbb{R}$ generated by $\{1, x, \dots, x^n\}$ and $\{P_0, \dots, P_n\}$ are the same. Therefore p can be expressed as a linear combination of first n Legendre polynomials.

1.3 Hermite polynomials

In literature two classes of orthogonal Hermite polynomials are defined: probabilists' and physicists'. They are defined on whole real line $D = \mathbb{R}$, while their weight function $w(x)$, for scalar product definition (see (1.1)), is respectively

$$w(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \qquad\qquad w(x) = \frac{1}{\sqrt{\pi}} e^{-x^2}$$

In both cases the system of Hermite polynomials is a maximal orthogonal family in $L^2(\mathbb{R}, w(x)dx)$. Since in applications only the class of physicists' Hermite polynomials is used, the discussion is restricted to this family. Nevertheless the same arguments apply to prove properties of the other class in the corresponding Hilbert space.

1.3.1 Definitions

Let us define the family of physicists' Hermite polynomials.

Theorem 1.5 *The following definitions are equivalent.*

(i) For every $n \in \mathbb{N}$

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{-x^2} \tag{1.15}$$

this is also called Rodrigues's formula.

(ii) For every fixed $x \in \mathbb{R}$

$$e^{2xt - t^2} = \sum_{n=0}^{\infty} \frac{H_n(x)}{n!} t^n \tag{1.16}$$

which is the generating function *of $\{H_n(x)\}_{n\in\mathbb{N}}$.*

(iii) If $H_0(x) = 1$ and $H_1(x) = 2x$, then for every natural index $n = 2, 3, \dots$

$$H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x)$$

Moreover, for all $n \in \mathbb{N}$

$$H_n'(x) = 2nH_{n-1}(x)$$

The family $\{H_n\}_{n\in\mathbb{N}}$ is called the class of physicists' Hermite polynomials.

Proof.

$(i) \implies (ii)$

The results of (1.15) are polynomials, since the n-th derivative of e^{-x^2} has at least n summands, each of them formed by a polynomial multiplied by e^{-x^2}. The transcendent function simplifies when multiplied by e^{x^2}, leaving only summation of monomials.
The function e^{-x^2} is a fixed point of Fourier transform, therefore

$$e^{-x^2} = \frac{1}{\sqrt{\pi}} \int_{\mathbb{R}} e^{-t^2} e^{2xit} dt$$

moreover, since the derivatives with respect to x of the integrand are dominated by a summable function, the theorem of derivation under integration sign yields to

$$\frac{d^n}{dx^n}e^{-x^2} = \frac{(2i)^n}{\sqrt{\pi}}\int_{\mathbb{R}} e^{-t^2}t^n e^{2xit}dt$$

Thus

$$H_n(x) = (-1)^n e^{x^2}\frac{d^n}{dx^n}e^{-x^2} = e^{x^2}\frac{(-2i)^n}{\sqrt{\pi}}\int_{\mathbb{R}} e^{-t^2}t^n e^{2xit}dt$$

Then, multiplying both sides by $\frac{r^n}{n!}$ and summing up for all $n \in \mathbb{N}$

$$\sum_{n=0}^{\infty}\frac{H_n(x)}{n!}r^n = \sum_{n=0}^{\infty}\frac{1}{n!}r^n e^{x^2}\frac{(-2i)^n}{\sqrt{\pi}}\int_{\mathbb{R}} e^{-t^2}t^n e^{2xit}dt$$

since

$$\sum_{n=0}^{\infty}\frac{1}{n!}(-2irt)^n$$

converges uniformly to e^{-2irt}, the series pass under the integral sign, hence

$$\begin{aligned}\sum_{n=0}^{\infty}\frac{H_n(x)}{n!}r^n &= \frac{e^{x^2}}{\sqrt{\pi}}\int_{\mathbb{R}}\sum_{n=0}^{\infty}\frac{(-2irt)^n}{n!}e^{-t^2}e^{2xit}dt = \\ &= \frac{e^{x^2}}{\sqrt{\pi}}\int_{\mathbb{R}} e^{-t^2}e^{-2irt}e^{2xit}dt = \\ &= \frac{e^{x^2}}{\sqrt{\pi}}\int_{\mathbb{R}} e^{-t^2}e^{2it(x-r)}dt = \\ &= e^{x^2}e^{-(x-r)^2} = \\ &= e^{2xr-r^2}\end{aligned}$$

$(ii) \implies (iii)$

Let us set

$$F(x,r) = e^{2xr-r^2}$$

then by trivial computations

$$\frac{\partial F}{\partial r} - (2x-2r)F = 0$$

Hence by plugging the serial definition of the function F into the previous differential equation, we get by property of series of powers

$$\sum_{n=1}^{\infty}\frac{H_n(x)}{n!}nr^{n-1} = 2x\sum_{n=0}^{\infty}\frac{H_n(x)}{n!}r^n - 2\sum_{n=0}^{\infty}\frac{H_n(x)}{n!}r^{n+1}$$

Then by recasting all the powers and equating terms of kind r^n, for $n = 2, \dots$

$$\frac{H_{n+1}(x)}{n!} = 2x\frac{H_n(x)}{n!} - 2\frac{H_{n-1}(x)}{(n-1)!}$$

and hence

$$H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x)$$

To prove the other recurrence formula, the function F accomplishes

$$\frac{\partial F}{\partial x} - 2rF = 0$$

Hence following the same procedure above, by differentiating the series term by term

$$\sum_{n=0}^{\infty}\frac{H_n'(x)}{n!}r^n - 2\sum_{n=0}^{\infty}\frac{H_n(x)}{n!}r^{n+1} = 0$$

and by equating the coefficients of all terms r^n for every fixed $n \in \mathbb{N}$

$$\frac{H_n'(x)}{n!} - 2\frac{H_{n-1}(x)}{(n-1)!} = 0$$

eventually

$$H_n'(x) = 2nH_{n-1}(x)$$

$(iii) \implies (i)$
Let us prove by induction that the polynomials achieved by Rodigues's formula are the same of the polynomials defined via recurrence formula. The basic step of induction is trivially proven by tabled Hermite polynomials.
Then let us consider the property of being equal true for all $k \leq n$, $k \in \mathbb{N}$. By exploiting the recurrence in (iii):

$$H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x) = 2xH_n(x) - H_n'(x)$$

Thus the inductive hypothesis applies to H_n and H_n'

$$\begin{aligned} H_{n+1}(x) &= 2xH_n(x) - H_n'(x) = \\ &= 2x(-1)^n e^{x^2}\frac{d^n}{dx^n}e^{-x^2} - (-1)^n(2x)e^{x^2}\frac{d^n}{dx^n}e^{-x^2} - (-1)^n e^{x^2}\frac{d^{n+1}}{dx^{n+1}}e^{-x^2} = \\ &= (-1)^{n+1}e^{x^2}\frac{d^{n+1}}{dx^{n+1}}e^{-x^2} \end{aligned}$$

Hence the formula (1.15) is achieved for $n+1$.

□

These equivalent definitions are used in different settings: the recurrence formula is employed for computing orthogonal polynomial, while the suitable one for describing properties of this class is (1.16). The first four Hermite polynomials are

$$\begin{aligned} H_0(x) &= 1 \\ H_1(x) &= 2x \\ H_2(x) &= 4x^2 - 2 \\ H_3(x) &= 8x^3 - 12x \\ H_4(x) &= 16x^4 - 48x^2 + 12 \end{aligned}$$

1.3.2 Properties

Orthogonality of $\{H_n\}_{n\in N}$ in $L^2(\mathbb{R}, w(x)dx)$ is the first property proven, where

$$w(x) = \frac{1}{\sqrt{\pi}}e^{-x^2}$$

is the weight function.

Proposition 1.3 *Given the system of Hermite polynomials $\{H_n\}_{n\in\mathbb{N}}$. Then for $n, m, \in \mathbb{N}$*

$$\int_{\mathbb{R}} H_n(x)H_m(x)w(x)dx = \delta_{nm}2^n n!$$

Proof. By equation (1.16) for all $|t| \leq 1$ and $|s| \leq 1$

$$e^{2xt-t^2} = \sum_{n=0}^{\infty}\frac{H_n(x)}{n!}t^n \qquad e^{2xs-s^2} = \sum_{m=0}^{\infty}\frac{H_m(x)}{m!}s^m$$

Therefore

$$e^{2xt-t^2+2xs-s^2} = \sum_{n=0}^{\infty}\sum_{m=0}^{\infty} \frac{H_n(x)H_m(x)}{m!n!} t^n s^m$$

Let us multiply both sides by e^{-x^2} and integrate on the real line

$$\int_{\mathbb{R}} e^{2xt-t^2+2xs-s^2} e^{-x^2} dx = \sum_{n=0}^{\infty}\sum_{m=0}^{\infty} \frac{t^n s^m}{m!n!} \int_{\mathbb{R}} H_n(x)H_m(x)e^{-x^2} dx \tag{1.17}$$

The series passes out the integral sign since it is a uniform convergent power series. Therefore

$$\int_{\mathbb{R}} e^{-(x-s-t)^2+2ts} dx = e^{2st} \int_{\mathbb{R}} e^{-(x-s-t)^2} dx = e^{2st} \int_{\mathbb{R}} e^{-u^2} du = e^{2st}\sqrt{\pi} = \sqrt{\pi} \sum_{n=0}^{\infty} \frac{2^n s^n t^n}{n!} \tag{1.18}$$

by equating the coefficients of the right-hand side of equations (1.17) and (1.18)

$$\int_{\mathbb{R}} H_n(x)H_m(x)e^{-x^2} dx = \delta_{nm}\sqrt{\pi}2^n n!$$

the thesis is achieved by dividing both sides by $\sqrt{\pi}$.

□

Since the system of polynomials defined by Theorem 1.5 is orthogonal, it coincides up to a constant, with the one defined via Gram-Schmidt orthogonalization procedure, see Theorem 1.1.

Let us prove that the set of physicists' Hermite polynomials is a maximal system in $L^2(\mathbb{R}, w(x)dx)$. A family $\mathcal{H} = \{H_n\}_{n\in\mathbb{N}}$ is said to be maximal if any function in $L^2(\mathbb{R}, w(x)dx)$ such that $f \notin \mathcal{H}$ and orthogonal to each H_n for $n \in \mathbb{N}$ is the zero function.

Lemma 1.2 *If f is an integrable function on $\mathbb{R}$ and*

$$\hat{f}(x) = \int_{\mathbb{R}} f(t)e^{ixt} dt \equiv 0 \tag{1.19}$$

then $f = 0$ almost everywhere

Proof. Let us multiply the left-hand side of (1.19) by e^{-ixa}, where a is a fixed real number.

$$\int_{\mathbb{R}} f(t)e^{ix(t-a)} dt = 0$$

By splitting the integration domain by means of a, and bringing on left-hand side one of the two resulting integrals, the previous equation becomes

$$\int_{-\infty}^{a} f(t)e^{ix(t-a)} dt = -\int_{a}^{+\infty} f(t)e^{ix(t-a)} dt$$

Let us define two functions for $z = x + iy$:

$$L(z) := \int_{-\infty}^{a} f(t)e^{iz(t-a)} dt \qquad R(z) = -\int_{a}^{+\infty} f(t)e^{iz(t-a)} dt$$

These complex functions are defined respectively for $\{\Im(z) \leq 0\}$ and $\{\Im(z) \geq 0\}$. Indeed let us consider $L(z)$, then the integrand function is

$$f(t)e^{i(x+iy)(t-a)} = f(t)e^{ix(t-a)}e^{-y(t-a)}$$

since $t \in (-\infty, a]$ it is summable whenever $y \leq 0$, therefore if $\{\Im(z) \leq 0\}$. The same argument applies to $R(z)$.

Moreover $L(z)$ is analytical on $Z^- = \{z \ : \ \Im(z) < 0\}$. Indeed the integral, that defines $L(z)$, is absolute convergent if $\Im(z) = y < 0$. Then for every triangle T that lies in Z^-

$$\oint_T L(z)dz = \oint_T \int_{-\infty}^{a} f(t)e^{itz}dtdz$$

Fubini-Tonelli Theorem applies, since $f(t)e^{itz}$ is absolutely integrable on Z^-, thus

$$\oint_T L(z)dz = \int_{-\infty}^{a} f(t) \oint_T e^{itz}dzdt = 0$$

the line integral over the triangle is null since e^{itz} is an analytic function. Therefore by Morena theorem $L(z)$ is analytical whenever $\{z \ : \ z \in \Im(z) < 0\}$. By the same argument the analyticity of $R(z)$ on $Z^+ = \{z \ : \ z \in \Im(z) > 0\}$ is proven.

Let us define $F(z)$ as

$$F(z) = \begin{cases} L(z) & \Im(z) \leq 0 \\ R(z) & \Im(z) > 0 \end{cases}$$

it is continuous at $\{z \in \mathbb{C} \ : \ \Im(z) = 0\}$, due to dominated convergence theorem:

$$\lim_{y \to 0} L(x+iy) = \int_{-\infty}^{a} f(t)e^{ix(t-a)}dt = -\int_{a}^{+\infty} f(t)e^{ix(t-a)}dt = \lim_{y \to 0} R(x+iy)$$

Theorem 7.7 in [7] ensures that $F(z)$ is entire on whole complex plane, moreover it is bounded for every $z \in \mathbb{C}$, by exploiting the integrability of $f(t)$

$$|L(z)| \leq \int_{-\infty}^{a} \left| f(t)e^{iz(t-a)} \right| dt \leq \int_{-\infty}^{a} e^{-y(t-a)} \left| f(t) \right| dt \leq \int_{\mathbb{R}} \left| f(t) \right| dt \leq C$$
$$|R(z)| \leq \int_{a}^{+\infty} \left| f(t)e^{iz(t-a)} \right| dt \leq \int_{a}^{+\infty} e^{-y(t-a)} \left| f(t) \right| dt \leq \int_{\mathbb{R}} \left| f(t) \right| dt \leq C$$

$F(z)$ satisfies the assumptions of Liouville's theorem, thus $F(z)$ is constant. In particular

$$\lim_{y \to +\infty} F(iy) = \lim_{y \to +\infty} -\int_{a}^{+\infty} f(t)e^{-y(t-a)}dt = 0$$

the limit passes under the integral sign by dominated convergence theorem. Therefore $F(z) = 0$ for all $z \in \mathbb{C}$ and in particular $F(0) = 0$, that yields to

$$\int_{-\infty}^{a} f(t) = 0$$

for every $a \in \mathbb{R}$, then $f = 0$ almost everywhere by Titchmarsh [8].

□

Lemma 1.3 *Let $f \in L^2(\mathbb{R}, w(x)dx)$, if for every $n = 0, 1, 2, \ldots$*

$$\int_{\mathbb{R}} f(x)x^n w(x)dx = 0$$

then $f(x) = 0$ almost everywhere in $L^2(\mathbb{R}, w(x)dx)$.

Proof.
For any complex number $z = t + is$, such that $s \in (-M, M)$, the function

$$F(z) = \int_{\mathbb{R}} e^{izx} f(x)w(x)dx$$

is entire, therefore it admits a series expansion of type

$$F(z) = \sum_{i=0}^{\infty} c_n z^n$$

where the coefficients are computed via derivations of F. The assumption for derivation, with respect to z variable, under integral sign are satisfied, thus

$$c_n = \frac{1}{n!} F^{(n)}(0) = \frac{1}{n!} i^n \int_{\mathbb{R}} x^n w(x) f(x) dx$$

by assumption the right-hand side is zero, for all $n \in \mathbb{N}$, therefore $F(z) = 0$, in particular, restricting the function to $\Re(z) = t$

$$0 = F(t) = \int_{\mathbb{R}} e^{itx} f(x) w(x) dx$$

$f(x)w(x)$ is integrable in $\mathbb{R}$ thus it accomplishes assumption of Lemma 1.2, implying that $f(x)w(x) = 0$ almost everywhere with respect to Lebesgue measure. By positiveness of $w(x)$, the function $f(x)\sqrt{w(x)}$ is zero almost everywhere. In other words

$$0 = \left\| f\sqrt{w} \right\|^2_{L^2(\mathbb{R})} = \int_{\mathbb{R}} f^2(x) w(x) dx = \|f\|^2_{L^2(\mathbb{R}, w(x)dx)}$$

proving that is zero in $L^2(\mathbb{R}, w(x)dx)$.

□

The two Lemmas above are used to prove the maximality of the set of physicists' Hermite polynomials in $L^2(\mathbb{R}, w(x)dx)$, where the weight function is $w(x) = \frac{1}{\sqrt{\pi}} e^{-x^2}$.

Theorem 1.6 *The set of physicists' Hermite polynomials $\{H_n\}_{n\in\mathbb{N}}$ is a Hilbert basis for $L^2(\mathbb{R}, w(x)dx)$*

Proof.

Let us choose a function $f \in L^2(\mathbb{R}, w(x)dx)$ such that for every $n \in \mathbb{N}$

$$\langle f, H_n \rangle = \int_{\mathbb{R}} f(x) H_n(x) w(x) dx = 0$$

by Gram-Schmidt orthogonalization procedure, for every $n \in \mathbb{N}$

$$H_n = x^n - \sum_{j=0}^{n-1} \alpha_{nj} H_j$$

applying $\langle f, \cdot \rangle$ to both sides of the previous equation, it becomes

$$\langle f, H_n \rangle = \langle f, x^n \rangle - \sum_{j=0}^{n-1} \alpha_{nj} \langle f, H_j \rangle$$

by hypothesis $\langle f, H_n \rangle = 0$ for all $n \in \mathbb{N}$, hence

$$\int_{\mathbb{R}} f(x) x^n w(x) dx = 0$$

by Lemma (1.3) $f = 0$ in $L^2(\mathbb{R}, w(x)dx)$, namely the maximality of $\{H_n\}_{n\in\mathbb{N}}$ basis is proven.

□

1.4 Properties of orthogonal maximal systems

Other useful properties of Legendre and Hermite polynomials follow by maximality of the systems considered. Since these properties are true for both classes let us use the notation developed in section 1.1. $L^2(D, w(x)dx)$ denotes the Hilbert space where the polynomials are defined. Moreover let

$$\mathcal{H} = \{\Psi_n(x)\}_{n\in\mathbb{N}} \tag{1.20}$$

be the maximal orthonormal family of polynomials.

From now on let U be an element of $L^2(D, w(x)dx)$. The orthogonal projection of U in the closure of the linear space generated by $\{\Psi_0, \ldots, \Psi_N\}$ for a fixed N is defined as

$$P_N(U) = U_N = \sum_{i=0}^{N} c_i \Psi_i(x) \tag{1.21}$$

where for each $i = 1, 2, \ldots, N$ the coefficients are

$$c_i = \langle U, \Psi_i \rangle = \int_D U(x)\Psi_i(x)w(x)dx$$

Definition 1.3 *If $U \in L^2(D, w(x)dx)$ its Fourier coefficients with respect to $\mathcal{H}$ are*

$$c_i = \langle U, \Psi_i \rangle \tag{1.22}$$

where $i \in \mathbb{N}$

Notice that the first N generalized Fourier coefficients coincide with the one defined in (1.21). Moreover they satisfy the Parceval identity.

Theorem 1.7 *(Parceval Identity) Let U be an element of $L^2(D, w(x)dx)$ and let $\mathcal{H}$ (1.20) be a maximal and orthonormal family in $L^2(D, w(x)dx)$. If $\{c_i\}_{i\in\mathbb{N}}$ are its Fourier coefficients, then*

$$\sum_{i=0}^{\infty} |c_i|^2 = \|U\|_{L^2(D,w(x)dx)}$$

The orthogonal projection in (1.21) allows to recover the best approximation of U on the finite dimensional subspace onto which is projected. Moreover $P_N(U)$ converges to U as $N \to +\infty$.

Theorem 1.8 *Given a function $U \in L^2(D, w(x)dx)$. If U_N is defined as in (1.21), then*

$$\lim_{N\to\infty} \|U - U_N\|_{L^2(\mathbb{R},w(x)dx)} = 0$$

Proof. The elements of $\mathcal{H}$ are orthonormal vectors in $L^2(D, w(x)dx)$, thus for a fixed $h \in \mathbb{N}$

$$\|U_N - U_{N+h}\|^2_{L^2(D,w(x)dx)} = \sum_{i=N+1}^{N+h} c_i^2$$

because of the convergence of $\sum_{k=0}^{\infty} c_k^2$ (Parceval identity), the sequence $\{U_N\}_{N\in\mathbb{N}}$ is Cauchy and it converges to an element $G \in L^2(D, w(x)dx)$, since $L^2(D, w(x)dx)$ is a complete space.

Then it is enough to prove U and G are the same element in $L^2(D, w(x)dx)$. Let us consider a fixed index $k \in \mathbb{N}$ such that $N > k$, moreover due to (1.21), $c_k = \langle U_N, \Psi_k \rangle$. Then

$$\begin{aligned}\int_D G(x)\Psi_k(x)w(x)dx - c_k &= \int_D G(x)\Psi_k(x)w(x)dx - \int_D U_N(x)\Psi_k(x)w(x)dx = \\ &= \int_D \big[G(x) - U_N(x)\big]\Psi_k(x)w(x)dx \leq \\ &\leq C\int_D \big[G(x) - U_N(x)\big]^2 w(x)dx\end{aligned}$$

the last step follows by Cauchy-Schwartz inequality and by $\Psi_k \in L^2(D, w(x)dx)$. Due to convergence in norm of $U_N \to G$ as $N \to +\infty$, for all $k \in \mathbb{N}$

$$\int_D G(x)\Psi_k(x)w(x)dx - c_k = 0$$

by plugging the very definition of c_k (Definition (1.3)) in the previous equation, we get

$$\int_D \big[G(x) - U(x)\big]\Psi_k(x)w(x)dx = 0 \qquad k = 0, 1, 2 \dots$$

by maximality of $\mathcal{H}$ system $G - U = 0$ in $L^2(D, w(x)dx)$. Therefore they are the same element in $L^2(D, w(x)dx)$.

□

Chapter 2

Polynomial Chaos Expansion

Generalized Polynomial Chaos (gPC) is a particular set of polynomials in a given random variable, usually denoted by ξ, with which an approximation of a finite second order random variable is computed. This procedure is named Polynomial Chaos Expansion (PCE).

From a theoretical point of view the elements of gPC are functionals, since their entries are the random variables ξ, moreover these mesurable functions characterize the measure of probability used for computing the expansion. Not every random variable ξ allows to define a gPC basis, this topic is far from the purposes of this document but it is related with Hamburger moment problem (see Ernst et al. in [14]).

The discussion is focused on a couple of measurable functions ξ: the gaussian distribution $\mathcal{N}(0, 1/2)$ and the uniform $\mathcal{U}(-1, 1)$ random variable. It is also analyzed the link between gPC basis and Legendre or Hermite polynomials, showing how the former inherits properties of the latter.

This technique exploits orthogonal properties of polynomials involved, to detect a representation of random variables as series of functionals. From a computational point of view it is an advantage, since polynomials are efficiently implemented by softwares for numerical computation.

In this chapter the polynomial chaos expansion of real-valued random variables is investigated starting from univariate case and passing to multivariate one.

2.1 One dimensional Polynomial Chaos Expansion

Let $(\Omega, \Sigma, \mathsf{P})$ be a probability space, where Ω is the abstract set of elementary events, Σ is a σ-algebra of subsets of Ω and P is a probability measure on Σ.

Definition 2.1 *The space $L^2(\Omega, \Sigma, \mathsf{P})$ is the Hilbert space of scalar real-valued random variables X defined on $(\Omega, \Sigma, \mathsf{P})$ such that*

$$\mathbb{E}[X^2] = \int_\Omega \big(X(\omega)\big)^2 d\mathsf{P}(\omega) < +\infty$$

Notice that, as for Lebesgue spaces, the elements $X \in L^2(\Omega, \Sigma, \mathsf{P})$ are equivalent classes of random variables.

This is a Hilbert space endowed with the following scalar product

$$\mathbb{E}[XY] = \langle X, Y \rangle_{\mathsf{P}} = \int_\Omega X(\omega)Y(\omega) d\mathsf{P}(\omega)$$

Therefore the norm is

$$\|X\|^2_{L^2(\Omega,\Sigma,\mathsf{P})} = \mathbb{E}[X^2] = \int_\Omega \big(X(\omega)\big)^2 d\mathsf{P}(\omega)$$

To shorten the notation $\|X\|_{\mathsf{P}}^2 := \|X\|_{L^2(\Omega,\Sigma,\mathsf{P})}^2$; the convergence in this norm is always referred as *mean square convergence* or *strong convergence*.

Among elements in $L^2(\Omega,\Sigma,\mathsf{P})$ there is the class of *basic random variables*, which is used to decompose, as entries of functionals, the quantity of interest Y. Not all functions $\xi : \Omega \to D$ are admissible, at least they have to accomplish two properties

- ξ has finite raw moments of all orders.
- The distribution function $F_\xi(x) := \mathsf{P}(\xi \leq x)$ of the basic random variables is continuous.

In particular only uniform $\mathcal{U}(-1,1)$ or normal $\mathcal{N}(0,1/2)$ are used as basic random variables.

The elements in $L^2(\Omega,\Sigma,\mathsf{P})$ can be gathered in two groups: the basic random variable that rules the decomposition and generic elements Y that one wish to decompose.

This coexistence of these two random variables Y and ξ yields to some subtleties on defining the correct measurable Hilbert space on which the decomposition holds. It is only a matter of matching σ-algebras.

Let us denote by $\sigma(\xi)$ the σ-algebra generated by the basic random variable ξ. Of course $\sigma(\xi) \subset \Sigma$. In order to express the random variable Y in terms of ξ, via a polynomial decomposition, it has at least to be measurable with respect to σ-algebra $\sigma(\xi)$.

A sufficient and necessary condition is given by Doob-Dynkin Lemma (as shown in Kallenberg [9], Lemma 1.13). It ensures that Y is $\sigma(\xi)$-measurable, by detecting a Borel measurable function $g : \mathbb{R} \to \mathbb{R}$, such that $Y = g(\xi)$. As consequence let us restrict the discussion to compute the decomposition in $L^2(\Omega,\sigma(\xi),\mathsf{P})$.

The basic random variable ξ induces, via its cumulative density function, a measure dF_ξ on $(D,\mathcal{B}(D))$, endowed with Borel σ-algebra on D.

Since the measure on $\mathbb{R}$ is absolutely continuous, there are two possible situations:

- if ξ is a Uniform $\mathcal{U}(-1,1)$, then $D = [-1,1]$, while $dF_\xi = w(x)dx$ where $w(x) = 1/2$.
- if ξ is a Normal $\mathcal{N}(0,1/2)$, then $D = \mathbb{R}$, while $dF_\xi = w(x)dx$ where $w(x) = \frac{1}{\sqrt{\pi}} e^{-x^2}$.

In the previous section the classes of Legendre and Hermite polynomials were studied. These families $\{\Psi_n(x)\}_{n\in\mathbb{N}}$ are orthogonal with respect to the weight function $w(x)$ defined above, now understood as probability density function of ξ.

Moreover the family $\{\Psi_n(x)\}_{n\in\mathbb{N}}$ can be interpreted as orthogonal with respect to measure dF_ξ, where the Hilbert space is $L^2(D,\mathcal{B}(D),dF_\xi)$ in this new framework.

By the very definition of random variable ξ, the sequence $\{\Psi_n(\xi)\}_{n\in\mathbb{N}}$, is an orthogonal system of functional polynomials in Hilbert space $L^2(\Omega,\sigma(\xi),\mathsf{P})$, which inherits all properties of $\{\Psi_n(x)\}_{n\in\mathbb{N}}$ in $L^2(D,\mathcal{B}(D),dF_\xi)$.

In particular Ψ_0 is considered as a degenerate random variable, also referred as *almost surely* constant random variable

$$\mathsf{P}(Y = 1) = 1$$

or equivalently Ψ_0 is P-almost equal to one.

Definition 2.2 *Given a basic random variable ξ, the associated orthogonal system $\{\Psi_n(\xi)\}_{n\in\mathbb{N}}$ is called generalized polynomial chaos (gPC) basis for the space $L^2(\Omega,\sigma(\xi),\mathsf{P})$. Moreover if $Y \in L^2(\Omega,\sigma(\xi),\mathsf{P})$ its polynomial chaos expansion (PCE) is*

$$Y = g(\xi) = \sum_{i=0}^{\infty} c_i \Psi_i(\xi) \tag{2.1}$$

and for all $i \in \mathbb{N}$

$$c_i = \frac{\mathbb{E}[Y\Psi_i]}{\mathbb{E}[\Psi_i^2]} = \frac{1}{\mathbb{E}[\Psi_i^2]} \int_\Omega g\big(\xi(\omega)\big)\Psi_i\big(\xi(\omega)\big) d\mathsf{P}(\omega) \tag{2.2}$$

By means of the scalar product in $L^2(\Omega, \sigma(\xi), \mathsf{P})$, the coefficients can be defined equivalently as

$$c_i = \frac{\langle Y, \Psi_i \rangle_{\mathsf{P}}}{\|\Psi_i\|^2_{\mathsf{P}}}$$

for all $i \in \mathbb{N}$.

Theorem 2.1 *Let ξ be a basic random variable with associated gPC basis $\{\Psi_n(\xi)\}_{n\in\mathbb{N}}$. If $Y \in L^2(\Omega, \sigma(\xi), \mathsf{P})$, then the truncated series of PCE (2.1)*

$$Y^{(N)}(\xi) = \sum_{i=0}^{N} c_i \Psi_i(\xi)$$

where the coefficients are defined as in equation (2.2), is a sequence of random variable $\{Y^{(N)}\}_{N\in\mathbb{N}}$ that converges in mean square sense to Y as $N \to +\infty$.

Proof. Since $Y \in L^2(\Omega, \sigma(\xi), \mathsf{P})$, there exists a Borel measurable function g, such that $Y = g(\xi)$. Let us consider a fixed degree N

$$\left\| g(\xi) - Y^{(N)} \right\|^2_{\mathsf{P}} = \int_\Omega \left(g\big(\xi(\omega)\big) - Y^{(N)}\big(\xi(\omega)\big) \right)^2 d\mathsf{P}(\omega) = \int_D \left(g(x) - Y^{(N)}(x) \right)^2 dF_\xi(x)$$

the right-hand side is meant as Lebesgue-Stietjes integral, therefore

$$\left\| g(\xi) - Y^{(N)} \right\|^2_{\mathsf{P}} = \left\| g(x) - Y^{(N)}(x) \right\|^2_{L^2(D, \mathcal{B}(D), dF_\xi)}$$

The set of orthogonal polynomials $\{\Psi_n(x)\}_{n\in\mathbb{N}}$ is a maximal system in $L^2(D, \mathcal{B}(D), dF_\xi)$, therefore for every real $\epsilon > 0$ there exists a N such that

$$\left\| g(x) - Y^{(N)}(x) \right\|^2_{L^2(D, \mathcal{B}(D), dF_\xi)} < \epsilon$$

Hence $\{Y^{(N)}(\xi)\}_{N\in\mathbb{N}}$ converges in mean square sense to $g(\xi) = Y$.

□

Doob-Dynking lemma restricts to $L^2(\Omega, \sigma(\xi), \mathsf{P})$ the class of random variables that can be decomposed into the gPC basis. Let us give an example where the PCE does not make sense even if the decomposed random variable has finite second raw moment.

Example. Let ξ be a $\mathcal{N}(0, 1/2)$ and let Y be an arbitrary and independent (with respect to ξ) random variable with finite second order. The coefficients of its decomposition into gPC basis are: for $i = 0$

$$c_0 = \mathbb{E}[\Psi_0 Y] = \mathbb{E}[Y]$$

while for each $i \in \mathbb{N}$ and $i > 0$, by independence property

$$c_i = \frac{1}{\mathbb{E}\left[\Psi_i^2\right]} \mathbb{E}[Y \Psi_i] = \frac{1}{\mathbb{E}\left[\Psi_i^2\right]} \mathbb{E}[Y] \cdot \mathbb{E}[\Psi_i] = 0$$

since $\mathbb{E}[\Psi_i] = 0$ for all $i > 0$, see Section 2.2, hence the decomposition in gPC basis is

$$Y^{(N)} = c_0$$

while the second order random variable is not almost constant. Moreover the approximation error in $L^2(\Omega, \Sigma, \mathsf{P})$ is the variance of Y, indeed

$$\left\| Y - Y^{(N)} \right\|^2_{\mathsf{P}} = \int_\Omega (Y - c_0)^2 d\mathsf{P}(\omega) = \int_\Omega Y^2(\omega) d\mathsf{P}(\omega) - c_0^2 = \mathsf{Var}[Y]$$

therefore no convergence occurs.

2.2 Properties of gPC basis and PCE

By means of induced measure dF_ξ on $(D, \mathcal{B}(D))$ lots of properties of the family of Legendre and Hermite polynomials in $L^2(D, w(x)dx)$ transfer to gPC basis in abstract space $L^2(\Omega, \sigma(\xi), \mathsf{P})$.

Property 1

$$\mathbb{E}\big[\Psi_i\big] = \begin{cases} 1 & i = 0 \\ 0 & i > 0 \end{cases}$$

Indeed, when $i = 0$

$$\mathbb{E}\big[\Psi_0\big] = \int_\Omega \Psi_0 d\mathsf{P}(\omega) = \int_\Omega d\mathsf{P}(\omega) = 1$$

while for $i > 0$, by orthogonality with respect to Ψ_0 polynomial

$$\mathbb{E}\big[\Psi_i\big] = \mathbb{E}[\Psi_0\Psi_i] = 0$$

Property 2

For each $i \in \mathbb{N}$, let us consider

$$\gamma_i = \frac{1}{\|\Psi_i\|_\mathsf{P}}$$

then $\{\gamma_i\Psi_i(\xi)\}_{i\in\,\mathbb{N}}$ is an orthonormal and maximal system in $L^2(\Omega, \sigma(\xi), \mathsf{P})$, thus Parceval identity applies, hence

$$\sum_{i\in\mathbb{N}} d_i^2 = \|g(\xi)\|_\mathsf{P}^2$$

where d_i is the i-th Fourier coefficient, namely

$$d_i = \langle g(\xi), \gamma_i\Psi_i\rangle$$

Notice that

$$d_i = \langle g(\xi), \gamma_i\Psi_i\rangle = \frac{\langle g(\xi), \Psi_i\rangle}{\|\Psi_i\|_\mathsf{P}^2}\|\Psi_i\|_\mathsf{P} = c_i\|\Psi_i\|_\mathsf{P}$$

where c_i is i-th coefficient of the PCE decomposition, see (2.2). Then

$$\|g(\xi)\|_\mathsf{P}^2 = \sum_{i\in\mathbb{N}} d_i^2 = \sum_{i\in\mathbb{N}} c_i^2\|\Psi_i\|_\mathsf{P}^2$$

This gives a precise formula to compute the second raw moment of $Y = g(\xi)$. Moreover the estimation of the mean square error is achieved, indeed

$$\epsilon^{(N)} = \left\|Y^{(N)} - Y\right\|_\mathsf{P} = \|Y\|_\mathsf{P} - \left\|Y^{(N)}\right\|_\mathsf{P}$$

this follows from basic properties of orthogonal projection in Hilbert space setting (see [1]). Then exploiting both Parceval identity and relation between d_i and c_i coefficients

$$\epsilon^{(N)} = \sum_{i=N+1}^{+\infty} c_i^2\|\Psi_i\|_\mathsf{P}^2$$

From now on $\epsilon^{(N)}$ is the mean square error of the truncated expansion of degree N.

Property 3 The polynomial chaos expansion allows to get statistics of the random variable $Y \in L^2(\Omega, \sigma(\xi), \mathsf{P})$ by means of coefficients of the decomposition, indeed

$$\mathbb{E}[Y] = \mathbb{E}[g(\xi)] = \mathbb{E}[g(\xi)\Psi_0] = \int_\Omega g\big(\xi(\omega)\big)\Psi_0\big(\xi(\omega)\big)d\mathsf{P}(\omega) = c_0$$

While the variance can be computed as $\mathsf{Var}[Y] = \mathbb{E}[Y^2] - \big(\mathbb{E}[Y]\big)^2$. The first summand of the right-hand side is understood as the square norm in $L^2(\Omega, \sigma(\xi), \mathsf{P})$ of Y. Thus by the previous property

$$\mathbb{E}[Y^2] = \mathbb{E}[g(\xi)^2] = \sum_{i=0}^{+\infty} c_i^2 \|\Psi_i\|_{\mathsf{P}}^2$$

Hence by plugging this result into $\mathsf{Var}[Y]$ formula

$$\mathsf{Var}[Y] = \mathsf{Var}[g(\xi)] = \left(\sum_{i=0}^{+\infty} c_i^2 \|\Psi_i\|_{\mathsf{P}}^2 \right) - c_0^2 = \sum_{i=1}^{+\infty} c_i^2 \|\Psi_i\|_{\mathsf{P}}^2$$

2.2.1 Example: decomposition of a Lognomal random variable

Let X be a $\mathcal{N}(\mu, \sigma^2)$, then the *Lognormal distribution* is defined as $Y = e^X$. Let us compute its gPC expansion of degree N, using the basic random variable $\xi \sim \mathcal{N}(0, 1/2)$, where $w(x)$ is its probability density function.
By plugging $X = \mu + a\xi$, where $a = \sqrt{2}\sigma$, into the definition of Y

$$Y = e^{\mu} e^{a\xi} =: g(\xi)$$

It is clear that this example fits into requirements of Definition 2.2. Hence its gPC expansion converges in mean square sense. Moreover let $\{H_i\}_{i \in \mathbb{N}}$ be the family of physicists' Hermite polynomials.

Let $i \in \mathbb{N}$ be fixed

$$c_i = \frac{1}{\|H_i\|_{\mathsf{P}}^2} \int_{\mathbb{R}} e^{\mu} e^{ax} H_i(x) \frac{1}{\sqrt{\pi}} e^{-x^2} dx$$

using trivial transformation the integrand becomes

$$c_i = \frac{e^{\mu} e^{a^2/4}}{2^i i!} \int_{\mathbb{R}} H_i(x) \frac{1}{\sqrt{\pi}} e^{-\left(x - \frac{a}{2}\right)^2} dx$$

then by setting $y = x - \frac{a}{2}$

$$c_i = \frac{e^{\mu} e^{a^2/4}}{2^i i!} \int_{\mathbb{R}} H_i\left(y + \frac{a}{2}\right) \frac{1}{\sqrt{\pi}} e^{-y^2} dy$$

which suggests to use the formula for translated Hermite polynomials, that is $H_i(y + t) = \sum_{k=0}^{i} \binom{i}{k} H_k(y)(2t)^{i-k}$. Hence by substituting it in the previous integral and using its linearity

$$c_i = \frac{e^{\mu} e^{a^2/4}}{2^i i!} \sum_{k=0}^{i} \binom{i}{k} a^{i-k} \int_{\mathbb{R}} H_k(y) \frac{1}{\sqrt{\pi}} e^{-y^2} dy$$

By orthogonality of H_i polynomials the only integral that is not trivial is for $k = 0$, whose value is one. Thus

$$c_i = e^{\mu} e^{a^2/4} \frac{a^i}{2^i i!}$$

Hence the truncated decomposition is

$$Y^{(N)}(\xi) = e^{\mu} e^{a^2/4} \sum_{i=0}^{N} \frac{a^i}{2^i i!} H_i(\xi)$$

2.2.2 Weak convergence

As required in Definition 2.2 the decomposition of a second order random variable Y, in terms of polynomials in basic random variable ξ, holds whenever there exists a Borel measurable function g, such that $Y = g(\xi)$.

Unfortunately in practical cases the detection of such g either it is not trivial or it is not possible. On the other hand Y is usually known by means of its distribution, this information is enough to define a kind of polynomial chaos expansion with weaker convergence property to Y.

The space, in which the decomposition is defined, is not changed: $L^2(\Omega, \sigma(\xi), \mathsf{P})$ as well as the family of orthogonal polynomials in ξ variable $\{\Psi_n(\xi)\}_{n\in\mathbb{N}}$. The new issue concerns the definition of a random variable $\overline{Y} = T(\xi)$, by means of

$$T = F_Y^{-1}\big(F_\xi(\xi)\big) \tag{2.3}$$

where F_Y, F_ξ are respectively the cumulative density functions of Y and ξ. As stated in [15] $\overline{Y}$ and Y have the same distribution.

If the transformation T defined in (2.3)

$$T \ : \ (\mathbb{R}, \mathcal{B}(\mathbb{R})) \to (\mathbb{R}, \mathcal{B}(\mathbb{R}))$$

is Borel measurable, $\overline{Y}$ is $\sigma(\xi)$-measurable by Doob-Dynkin Lemma. This is always the case since F_Y is always continuous function as well as ξ is a continuous random variable. Moreover by Baldi [10] having the same distribution implies equality of moment generating function and thus Y and $\overline{Y}$ have the same second raw moments, ensuring that $\overline{Y} \in L^2(\Omega, \sigma(\xi), \mathsf{P})$.

Hence the procedure in Definition 2.2 and Theorem 2.1 holds for $\overline{Y}$: the polynomial chaos expansion can be detected through the gPC basis, ensuring strong convergence of the truncated series to $\overline{Y}$.

In Appendix A it is shown that having the same cumulative density function does not imply to be the same measurable function. Therefore in general settings the function $\overline{Y}$ and Y are not the same element in $L^2(\Omega, \sigma(\xi), \mathsf{P})$. This justifies to handle $\overline{Y}$ and Y as different quantities.

Definition 2.3 *Let $Y \in L^2(\Omega, \Sigma, \mathsf{P})$ be a random variable with distribution F_Y, let ξ be a basic random variable with gPC basis $\{\Psi_n\}_{n\in\mathbb{N}}$. If $\overline{Y} = T(\xi)$, where T is as in (2.3), then*

$$\overline{Y} = T(\xi) = \sum_{i=0}^{+\infty} c_i \Psi_i(\xi) \tag{2.4}$$

is called weak approximation of Y. The coefficients are

$$c_i = \frac{1}{\mathbb{E}\left[\Psi_i^2\right]} \mathbb{E}\left[\overline{Y}\Psi_i\right]$$

for all indexes $i \in \mathbb{N}$.

Since $\overline{Y} \in L^2(\Omega, \sigma(\xi), \mathsf{P})$ the series in (2.4) is strongly convergent. Moreover it preserves a convergence property with respect to Y.

Theorem 2.2 *Let $Y \in L^2(\Omega, \Sigma, \mathsf{P})$ be a random variable with cumulative density function F_Y. Let ξ be a basic random variable with associated gPC $\{\Psi_n(\xi)\}_{n\in\mathbb{N}}$. Then the partial sums of the series (2.4)*

$$Y^{(N)} = \sum_{i=0}^{N} c_i \Psi_i(\xi)$$

is a sequence of random variables, that converges in probability to Y.

Proof. By assumption $\overline{Y}$ admits a PCE that converges to it in mean square sense, therefore the truncated expansion converges in probability: for each real $\epsilon > 0$

$$\mathsf{P}\left(\left|\overline{Y} - Y^{(N)}\right| > \epsilon\right) \to 0 \qquad N \to +\infty$$

Let us consider two sets

$$A = \left\{\omega \in \Omega \ : \ \left|\overline{Y}(\omega) - Y^{(N)}(\omega)\right| > \epsilon\right\}$$
$$B = \left\{\omega \in \Omega \ : \ \left|Y(\omega) - Y^{(N)}(\omega)\right| > \epsilon\right\}$$

they belong to this σ-algebra Σ by the very definition of random variable. Their images through functions Y and $\overline{Y}$ are the same set, namely $(\epsilon, +\infty) \cap D$. Moreover being equally distributed implies

$$1 - \mathsf{P}\big(\overline{Y}^{-1}(A)\big) = 1 - F_{\overline{Y}}(\epsilon) = 1 - F_Y(\epsilon) = 1 - \mathsf{P}\left(Y^{-1}(B)\right)$$

therefore

$$\mathsf{P}\left(\left|Y - Y^{(N)}\right| > \epsilon\right) \to 0 \qquad N \to +\infty$$

□

2.3 Multidimensional gPC basis and PCE

Multivariate polynomial chaos theory is described using the univariate case. Actually there are two theories for multivariate PCE: regarding to finitely many basic random variables and the infinitely many. The theoretical approach is very different, but, from a computational point of view, it is important to describe the first one, since inputs of the processes considered are always finitely many.

Let us consider a finite number of independent random variables $\xi_1, \xi_2, \ldots, \xi_M$ defined on $(\Omega, \Sigma, \mathsf{P})$. They can be collected into a random vector $\boldsymbol{\xi} = \boldsymbol{\xi}(\omega) \in \mathbb{R}^M$.

Denoting by

$$\left\{\Psi_i^{(m)}(\xi_m)\right\}_{i \in \mathbb{N}} \qquad m = 1, 2, \ldots, M$$

the sequence of orthogonal polynomials with respect to the m-th distribution, that belongs to $L^2(D_m, \sigma(\xi_m), \mathsf{P})$. The set of multivariate (tensor product) polynomials is given by

$$\Psi_{\mathbf{i}}(\boldsymbol{\xi}) = \prod_{m=1}^{M} \Psi_{i_m}^{(m)}(\xi_m) \qquad \mathbf{i} = (i_1, \ldots, i_M) \in \mathbb{N}^M$$

it constitutes a system of random variables in the space $L^2(\Omega, \sigma(\boldsymbol{\xi}), \mathsf{P})$. The random vector $\boldsymbol{\xi}$ induces a measure $dF_{\boldsymbol{\xi}}$ on the image space $(\mathbf{D}, \mathcal{B}(\mathbf{D}))$, were $D \subset \mathbb{R}^M$ and $\mathbf{D} = D_1 \times D_2 \times \cdots \times D_M$

$$dF_{\boldsymbol{\xi}} = dF_{\xi_1} \times \cdots \times dF_{\xi_M}.$$

Thus from properties of tensor products of Hilbert spaces (see [11] section II.4) the Hilbert basis of $L^2(\mathbf{D}, \mathcal{B}(\mathbf{D}), dF_{\boldsymbol{\xi}})$ is the tensor product of each basis of $L^2(D_m, \mathcal{B}(D_m), dF_{\xi_m})$, for $m = 1, \ldots, M$. Therefore its elements are

$$\Psi_{\mathbf{i}}(\boldsymbol{x}) = \prod_{m=1}^{M} \Psi_{i_m}^{(m)}(x_m) \qquad \mathbf{i} = (i_1, \ldots, i_M) \in \mathbb{N}^M$$

for $\mathbf{x} \in \mathbb{R}^M$. By the very definition of random vector $\boldsymbol{\xi}$, the sequence $\{\Psi_{\mathbf{i}}(\boldsymbol{\xi})\}_{\mathbf{i} \in \mathbb{N}^M}$ is an orthogonal system of functional polynomials in the Hilbert space $L^2(\Omega, \sigma(\boldsymbol{\xi}), \mathsf{P})$, which inherits properties of $\{\Psi_{\mathbf{i}}(\mathbf{x})\}_{\mathbf{i} \in \mathbb{N}^M}$ in $L^2(\mathbf{D}, \mathcal{B}(\mathbf{D}), dF_{\boldsymbol{\xi}})$.

This implies that actually the family of polynomials

$$\{\Psi_{\mathbf{i}}(\boldsymbol{\xi})\}_{\mathbf{i}\in\mathbb{R}^M}$$

is a maximal system in $L^2(\Omega, \sigma(\boldsymbol{\xi}), \mathsf{P})$.

In order to decomposed Y into the vector of independent basic random variables $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_M)$, Y has to be $\sigma(\boldsymbol{\xi})$-measurable. This is ensured by the "multidimensional" Doob-Dynkin lemma, namely if there exists a Borel measurable function g such that $Y = g(\boldsymbol{\xi})$.

Definition 2.4 *Let $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_M)$ be a finite vector of $M \in \mathbb{N}$ independent random variables and let $\left\{\Psi_i^{(m)}(\xi_m)\right\}_{i\in\mathbb{N}}$, $m = 1, \ldots, M$ be the associated orthogonal sequence of polynomials. The system of random variables*

$$\Psi_{\mathbf{i}}(\boldsymbol{\xi}) = \prod_{m=1}^{M} \Psi_{i_m}^{(m)}(\boldsymbol{\xi}) \qquad \mathbf{i} = (i_1, \ldots, i_M) \in \mathbb{N}^M$$

is an orthogonal basis of the space $L^2(\Omega, \sigma(\boldsymbol{\xi}), \mathsf{P})$ called generalized polynomial chaos (gPC) basis. Moreover the multidimensional polynomial chaos expansion (PCE) is

$$Y = g(\boldsymbol{\xi}) = \sum_{\mathbf{i}\in\mathbb{N}^M} c_{\mathbf{i}} \Psi_{\mathbf{i}}(\boldsymbol{\xi}) \tag{2.5}$$

where

$$c_{\mathbf{i}} = \frac{1}{\|\Psi_{\mathbf{i}}\|_{\mathsf{P}}^2} \langle g(\boldsymbol{\xi}), \Psi_{\mathbf{i}}(\boldsymbol{\xi})\rangle_{\mathsf{P}}$$

for all multi-indexes $\mathbf{i} \in \mathbb{N}^M$.

Theorem 2.3 *Let $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_M)$ be a vector of $M \in \mathbb{N}$ independent random variables. The series defined in equation (2.5) converges in mean square sense in $L^2(\Omega, \sigma(\boldsymbol{\xi}), \mathsf{P})$.*

Proof.

The random vector $\boldsymbol{\xi}$ induces a measure $dF_{\boldsymbol{\xi}}$ on image space $(\mathbf{D}, \mathcal{B}(\mathbf{D}))$. Moreover let us set $Z(\boldsymbol{\xi}) = \sum_{\mathbf{i}\in\mathbb{N}^M} c_{\mathbf{i}} \Psi_{\mathbf{i}}(\boldsymbol{\xi})$ as the PCE of $Y = g(\boldsymbol{\xi})$. Thus the mean square error is

$$\|g(\boldsymbol{\xi}) - Z(\boldsymbol{\xi})\|_{\mathsf{P}}^2 = \int_\Omega \left(g(\boldsymbol{\xi}(\omega)) - Z(\boldsymbol{\xi}(\omega))\right)^2 d\mathsf{P}(\omega) = \int_{\mathbf{D}} \left(g(\mathbf{x}) - Z(\mathbf{x})\right)^2 dF_{\boldsymbol{\xi}}(\mathbf{x})$$

Since the family of polynomials $\{\Psi_{\mathbf{i}}(\mathbf{x})\}$ is a Hilbert basis for $L^2(\mathbf{D}, \mathcal{B}(\mathbf{D}), dF_{\boldsymbol{\xi}})$ (see [11] section II.4), the previous quantity is zero. Therefore the PCE converges in mean square sense to Y.

□

The multivariate decomposition is defined using the univariate one, simply by considering the tensor products between the M-bases of orthogonal polynomials. Moreover truncation is not so straightforward. It is clear that this difficulty in truncation is due to the nonexistence of trivial order in multi-indexes representation of multivariate polynomials.

Before describing a possible solution let us detect the finite dimensional subspace where the truncated series lies.

Given a tuple $(n_1, \ldots, n_M) \in \mathbb{N}^M$, the univariate finite bases are

$$B_m = \left\{\Psi_i^{(m)}(\xi_m)\right\}_{i=0,\ldots,n_m} \qquad m = 1, \ldots, M$$

each of them generates a linear subspace of $L^2(\Omega, \sigma(\xi_m), \mathsf{P})$. Then let us consider $\mathbf{B}$ a finite dimensional subspace of $L^2(\Omega, \sigma(\boldsymbol{\xi}), \mathsf{P})$ generated by

$$\bigotimes_{m=1}^{M} B_m$$

Then it is clear that $dim\ \mathbf{B} = (n_1 + 1) \cdot (n_2 + 1) \cdots (n_M + 1)$.

A possibility is to define the *graded lexicographic order* for multi-indexes where $\mathbf{i} > \mathbf{j}$ if and only if $|\mathbf{i}| > |\mathbf{j}|$ and the first non zero entry in the difference $\mathbf{i} - \mathbf{j}$ is positive.
It allows to sort the multi-indexes in an ascending order following a single index. Table (2.1) shows a two dimensional example of such ordering. Therefore the truncation of (2.5) is

$$Y^{(N)} = \sum_{|\mathbf{i}| \leq N} c_{\mathbf{i}} \Psi_{\mathbf{i}}(\boldsymbol{\xi})$$

By construction it lies in the linear subspace of $L^2(\Omega, \sigma(\boldsymbol{\xi}), \mathsf{P})$ spanned by the multivariate polynomials of total degree at most N. This space is called $\mathbb{P}_N^M$, and its dimension is

$$dim\ \mathbb{P}_N^M = \binom{N+M}{N}$$

In implementation framework this is the most used ordering for multidimensional polynomials since the truncated series is directly defined by means of total degree N.

$\lvert\mathbf{i}\rvert$	Multi-index $\mathbf{i}$	Single index k
0	(0 0)	1
1	(1 0)	2
	(0 1)	3
2	(2 0)	4
	(1 1)	5
	(0 2)	6
3	(3 0)	7
	(2 1)	8
	(1 2)	9
	(0 3)	10

Table 2.1: An example of graded lexicographic ordering for a two dimensional multi-index $\mathbf{i}$

The main properties of PCE is convergence in mean square sense of (2.5). This is ensured by theory of tensor products of Hilbert spaces. But actually the final aim is to get convergence for increasing total degree N. It is enough to prove that for every fixed tuple $(n_1, \ldots, n_M) \in \mathbb{N}^M$ there exists a total index N, such that the linear space spanned by polynomials of total degree at most N contains the one spanned by tensor product basis.

The discussion is made using a practical example of a 2-dimensional finite basis $\boldsymbol{\xi} = (\xi_1, \xi_2)$, characterized by a couple $(n_1, n_2) = (2, 3)$. Therefore the associated univariate finite bases are

$$B_1 = \{\Psi_0^{(1)}, \Psi_1^{(1)}, \Psi_2^{(1)}\} \qquad B_2 = \{\Psi_0^{(2)}, \Psi_1^{(2)}, \Psi_2^{(2)}, \Psi_3^{(2)}\}$$

The tensor product consists in all the possible products of these elements.

$$\begin{array}{cccc} \Psi_0^{(1)}\Psi_0^{(2)} & \Psi_0^{(1)}\Psi_1^{(2)} & \Psi_0^{(1)}\Psi_2^{(2)} & \Psi_0^{(1)}\Psi_3^{(2)} \\ \Psi_1^{(1)}\Psi_0^{(2)} & \Psi_1^{(1)}\Psi_1^{(2)} & \Psi_1^{(1)}\Psi_2^{(2)} & \Psi_1^{(1)}\Psi_3^{(2)} \\ \Psi_2^{(1)}\Psi_0^{(2)} & \Psi_2^{(1)}\Psi_1^{(2)} & \Psi_2^{(1)}\Psi_2^{(2)} & \Psi_2^{(1)}\Psi_3^{(2)} \end{array}$$

On the other hand the lexicographical order, allows to consider finite basis via detecting all multi-index $|\mathbf{i}| \leq N$, for a fixed N.

These approaches can be represented graphically on a plane, see Figure (2.1). The entries of each couple $(n_1, n_2) \in \mathbb{N}^2$ are respectively the degree of the first and the second polynomial involved. Therefore the x-axis represents the degree of polynomials in first variable, while y-axis is the degree of polynomials in second variable.

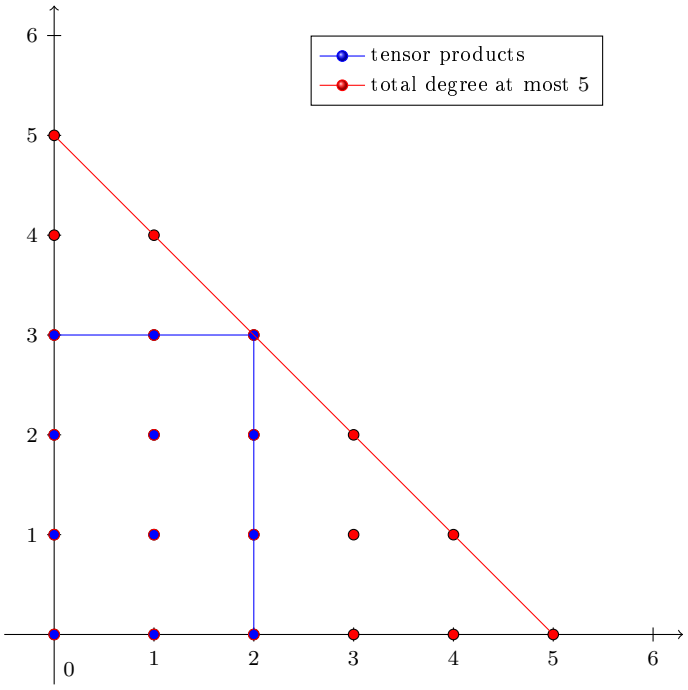

Figure 2.1: Degrees of 2D basis polynomials coming from either tensor product of polynomial bases either considering the polynomials of total degree at most N

For instance the couple $(2, 3)$ corresponds to $\Psi_2^{(1)}\Psi_3^{(2)}$.

The red circles represent the multivariate polynomials that follow a graded lexicographical ordering, whose total degree is at most N. They can be detected as all the couples of natural numbers that lies below the red line, which represent the bound $|\mathbf{i}| \leq N$.

While the tensor product elements are the one inside the rectangle whose sides are of length 2 and 3 respectively. If the total degree is

$$N = m_1 + n_2 = 2 + 3$$

all polynomials, coming from tensor product computations, have total degree less or equal to N.

In general setting it is enough to sum the components of the entries in the tuple. This is a direct consequence of the definition of total degree of the polynomial.

With this choice of N, from Figure 2.1 it is clear that the linear space generated by the tensor product basis is a subspace of $span\{\Psi_\mathbf{i}\}_{|\mathbf{i}|\leq N}$, thus the convergence in mean square sense for $(n_1, n_2) \to +\infty$ implies convergence as $N \to +\infty$ of

$$Y^{(N)} = \sum_{|\mathbf{i}|\leq N} c_\mathbf{i}\Psi_\mathbf{i}(\boldsymbol{\xi})$$

in $L^2(\Omega, \sigma(\boldsymbol{\xi}), \mathsf{P})$.

2.4 PCE of random vectors

Let $\mathbf{Y} = (Y_1, \dots, Y_J)$ be a random vector, whose components are random variables on $(\Omega, \Sigma, \mathsf{P})$, thus for each $j = 1, \dots, J$

$$Y_j : \Omega \to \mathbb{R}$$

For simplicity let us consider the univariate decomposition. The polynomial chaos expansion of $\mathbf{Y}$ is the collection of the PCE of each component, thus for a fixed $j \in \{1, \dots, J\}$

$$Y_j = \sum_{i=0}^{\infty} c_i^{(j)} \Psi_i(\xi)$$

where

$$c_i^{(j)} = \frac{\mathbb{E}\left[Y_j, \Psi_i\right]}{\mathbb{E}\left[\Psi_i^2\right]}$$

The strong convergence is achieved whenever Doob-Dynking lemma applies component wise, if this is not the case weak convergence is accomplished by means of section (2.2.2).

The vectorial PCE can be recast as

$$\mathbf{Y} = \sum_{i=0}^{\infty} \mathbf{c}_i \Psi_i(\xi)$$

where $\mathbf{c}_i = \left(c_i^{(1)}, \dots, c_i^{(J)}\right)$.

Whether $\boldsymbol{\xi}$ is a basic random vector, the multivariate decomposition is achieved by applying the multivariate polynomial chaos expansion component by component.

2.5 A multi-element approach

The described PCE suffers of low convergence rate if the random variable $Y = g(\xi)$, both in univariate and multivariate cases, has high curvature or is highly oscillating (see [19]). The usual polynomial chaos decomposition can be referred in terms of *global*, since a unique random variable is used to describe Y on the whole sampling space.

In order to overwhelm this low convergent rate, a multi-element polynomial chaos can be developed. The basic idea is breaking the domain into sub-elements and define local PCE that may be more efficient to reduce the error.

Multi-Element Generalized Polynomial Chaos (ME-gPC) bases were introduced by Wan and Karniadakis in [17]. This approach provides a decomposition of the quantity of interest Y on each element and it defines a new classes of orthogonal polynomials, in which the expansion is computed. The orthogonality is detected with respect to a weight function specific for each element.

The idea described in this section differs: the local decompositions are made in term of the usual basic random variable, then the polynomial chaos expansions are combined in order to get the *global* behavior of the random quantity of interest Y.

The discussion made below is restricted to univariate expansions, but it can be extended to multivariate settings exploiting the same idea described in the previous section.

2.5.1 Decomposing the range of a scalar random variable

The decomposition of the sample space into elements is required as first step. Then for each element a collection of local random variables, which are compatible with the global one Y, is detected.

Let $(\Omega, \Sigma, \mathsf{P})$ be a probability space. Suppose to have a real-valued random variable Y defined on it

$$Y \ : \ \Omega \longrightarrow D \subset \mathbb{R}$$

where D is a closed interval. It induces a measure dF_Y on $(D, \mathcal{B}(D))$, where $\mathcal{B}(D)$ is the Borel σ-algebra, for simplicity let us assume that the distribution F_Y is an absolute continuous function, thus an integrable probability density function f_Y always exists.

Let $\mathcal{A} = \{A_k\}_{k=1}^{N_e}$ be a N_e-element partition of D. These elements are pairwise disjoint intervals in $\mathbb{R}$, thus

$$D = \bigcup_{k=1}^{N_e} A_k$$

Then for every fixed index $k = 1, 2, \ldots, N_e$ let us define the indicator random variable $I_{A_k} : \Omega \to \mathbb{R}$ as

$$I_{A_k}(\omega) = \begin{cases} 1 & \xi(\omega) \in A_k \\ 0 & \text{otherwise} \end{cases}$$

By construction it is a measurable function from Ω to $\mathbb{R}$. Moreover $I_{A_k}^{-1}(1) \cap I_{A_h}^{-1}(1) = \varnothing$ if $k \neq h$ where $h, k \in \{1, 2 \ldots, N_e\}$, by means of the partition $\mathcal{A}$. Hence

$$\Omega = \bigcup_{i=1}^{N_e} I_{A_h}^{-1}(1)$$

This proves that actually the partition on the image space of Y reflects to the sample space Ω, thus no distinction will be done, since one implies the other.

The main goal is defining, for each element A_k, a random variable Y_k such that it is consistent with Y restricted to A_k. Technically the requirements are about a random variable defined of $\big(I_{A_k}^{-1}(1), \Sigma \cap I_{A_k}^{-1}, \mathsf{P}(\cdot | I_{A_k} = 1)\big)$, for simplicity $\mathsf{P}_k := \mathsf{P}(\cdot | I_{A_k} = 1)$, where

$$Y_k \ : \ I_{A_k}^{-1}(1) \longrightarrow A_k$$

that induces a measure on the image space $(A_k, \mathcal{B} \cap A_k, dF_{Y_k})$, such that

$$\mathsf{P}_k \left(Y_k \in B \cap A_k \right) = \frac{\mathsf{P} \left(Y \in B \cap A_k \right)}{\mathsf{P} \left(Y \in A_k \right)} \tag{2.6}$$

where the sets $\{Y \in A_k\}$ represent $\{\omega \in \Omega \ : \ Y(\omega) \in A_k\}$ for all $k = 1, \ldots, N_e$.

In order to define Y_k such that (2.6) holds, it is needed that $\mathsf{P}(Y \in A_k) > 0$ for all the elements A_k. Indeed if there exists an index k, such that $\mathsf{P}(Y \in A_k) = 0$, the set A_k can be put together with either A_{k-1} or A_{k+1}. For simplicity let us suppose to select the first. Since A_k and A_{k-1} are disjoint

$$\mathsf{P}\big(Y \in (A_{k-1} \cup A_k)\big) = \mathsf{P}(Y \in A_{k-1}) + \mathsf{P}(Y \in A_k) = \mathsf{P}(Y \in A_{k-1})$$

Therefore without loss of generality the partition $\mathcal{A}$ is made of non-zero length intervals.

By multiplying both sides of equation (2.6) by $P(Y \in A)$ and by summing it up with respect to all elements

$$\sum_{k=1}^{N_e} \mathsf{P} \left(Y \in B \cap A_k \right) = \sum_{k=1}^{N_e} \mathsf{P}_k \left(Y_k \in B \cap A_k \right) \cdot \mathsf{P}(Y \in A_k)$$

since the sets $B \cap A_k$ for $k = 1, \ldots, N_e$ are a partition of B

$$\mathsf{P}(Y \in B) = \sum_{k=1}^{N_e} \mathsf{P}_k \left(Y_k \in B \cap A_k \right) \cdot \mathsf{P}(Y \in A_k) \tag{2.7}$$

For a fixed index $k \in \{1, 2, \ldots, N_e\}$ a possible choice of Y_k on A_k is a random variable subjected to the following probability density function

$$f_{Y_k} = \begin{cases} \dfrac{f_Y(x)}{p_k} & x \in A_k \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

where $p_k = \int_{A_k} f_Y(x)dx$. The probability density function f_{Y_k} can be understood as the normalization to one of $f_Y|_{A_k}$.

Proposition 2.1 *Let $Y : \Omega \to D$ be a random variable with density function f_Y, and let $\mathcal{A} = \{A_k\}_{k=1}^{N_e}$ be the partition defined on D. If for each $k = 1, 2, \ldots, N_e$, Y_k is a random variable subjected to f_{Y_k}(2.8), then*

$$\mathsf{P}_k\left(Y_k \in B \cap A_k\right) = \frac{\mathsf{P}\left(Y \in B \cap A_k\right)}{\mathsf{P}\left(Y \in A_k\right)}$$

Proof. Let us consider a fixed index $k \in \{1, 2, \ldots, N_e\}$. From direct computations, exploiting definition of f_{Y_k}

$$\begin{aligned} \mathsf{P}_k\left(Y_k \in B \cap A_k\right) &= \int_{B \cap A_k} f_{Y_k}(x)dx = \\ &= \int_{B \cap A_k} \frac{f_Y(x)}{p_k} dx = \\ &= \frac{1}{p_k} \int_{B \in A_k} f_Y(x)dx = \\ &= \frac{\mathsf{P}\left(Y \in B \cap A_k\right)}{\mathsf{P}\left(Y \in A_k\right)} \end{aligned}$$

□

Let us compute the statistics of Y by means of Y_k moments.

Proposition 2.2 *In the hypothesis of the previous proposition, if $g : \mathbb{R} \to \mathbb{R}$ is a Borel measurable and integrable function, then*

$$\mathbb{E}\left[g(Y)\right] = \sum_{k=1}^{N_e} \mathbb{E}\left[g(Y_k)\right] \cdot \mathsf{P}\left(Y \in A_k\right)$$

Proof. Exploiting the definition of the expectation and the property of the partition $\mathcal{A}$

$$\mathbb{E}\left[g(Y)\right] = \int_D g(x) f_Y(x)dx = \sum_{k=1}^{N_e} \int_{A_k} g(x) f_Y(x)dx$$

then upon defining $p_k = \int_{A_k} f_Y(y)dy$ for each $k = 1, 2, \ldots, N_e$

$$\sum_{k=1}^{N_e} \int_{A_k} g(x) f_Y(x)dx = \sum_{k=1}^{N_e} \frac{p_k}{p_k} \int_D g(x) f_Y(x)dx$$

Keeping in mind that $p_k = \mathsf{P}\left(Y \in A_k\right)$ and f_{Y_k} then

$$\mathbb{E}\left[g(Y)\right] = \sum_{k=1}^{N_e} \mathsf{P}\left(Y \in A_k\right) \int_{A_k} g(x) f_{Y_k}(x)dx = \sum_{k=1}^{N_e} \mathbb{E}\left[g(Y_k)\right] \cdot \mathsf{P}\left(Y \in A_k\right)$$

□

The previous theorem detects directly the mean of Y, while in order to compute the variance of Y using $\mathsf{Var}[Y_k]$, some manipulations are needed. Let us consider as Borel measurable function $g(y) = (y-\mu)^2$, where $\mu = \mathbb{E}[Y]$ and let us call $\mu_k = \mathbb{E}[Y_k]$ for each $k = 1, 2, \ldots, N_e$.

$$\begin{aligned}
\mathbb{E}\left[(Y-\mu)^2\right] &= \sum_{k=1}^{N_e} \mathbb{E}\left[(Y_k-\mu)^2\right] \cdot \mathsf{P}\left(Y \in A_k\right) = \\
&= \sum_{k=1}^{N_e} \mathbb{E}\left[(Y_k-\mu_k+\mu_k-\mu)^2\right] \cdot \mathsf{P}\left(Y \in A_k\right) = \\
&= \sum_{k=1}^{N_e} \left[\mathbb{E}\left[(Y_k-\mu_k)^2\right] + (\mu_k-\mu)^2 - 2(\mu_k-\mu)\mathbb{E}\left[Y_k-\mu_k\right]\right] \cdot \mathsf{P}\left(Y \in A_k\right) = \\
&= \sum_{k=1}^{N_e} \left[\mathbb{E}\left[(Y_k-\mu_k)^2\right] + (\mu_k-\mu)^2\right] \cdot \mathsf{P}\left(Y \in A_k\right)
\end{aligned}$$

2.5.2 A Multi-Element Polynomial Chaos

In the previous section the random variables Y_k were defined in order to be consistent with Y on each element A_k for $k = 1, \ldots, N_e$. Next step provides to compute the PCE of each Y_k.

Wan and Karniadakis [17] proposed a multi-element polynomial chaos expansion, where for each element considered, a family of orthogonal polynomials is developed. The orthogonality condition is achieved with respect to the probability density function f_{Y_k}, thus it is different on each element.

This approach defines a gPC basis strictly related to the local behavior of the random variable Y, by the very definition of f_{Y_k}. A drawback is about the definition of new orthogonal families, one for each element.

In this section the approach proposed is slightly different. Despite developing new orthogonal families of polynomials, usual orthogonal classes are used for each element A_k. For simplicity the decomposition of each Y_k is made with the same basic random variables ξ.

Let us restrict the discussion to a generic element A_k of the partition $\mathcal{A}$, where the basic random variable ξ is already fixed (Normal or Uniform) as well as the associated gPC basis $\{\Psi_{k,i}(\xi)\}_{i\in\mathbb{N}}$. Therefore Y_k, defined by (2.8), can be decomposed as

$$Y_k = \sum_{i=0}^{\infty} c_{k,i}\Psi_{k,i}(\xi) \tag{2.9}$$

where the coefficients are defined as in (2.2). Moreover let us consider the truncated series of degree N

$$Y_k^{(N)} = \sum_{i=0}^{N} c_{k,i}\Psi_{k,i}(\xi) \tag{2.10}$$

By considering events $\{Y \leq x\}$ for some $x \in \mathbb{R}$, the equation (2.7) can be expressed in terms of cumulative density functions F_{Y_k}

$$F_Y(x) = \sum_{k=1}^{N_e} F_{Y_k}(x) \cdot \mathsf{P}\left(Y \in A_k\right)$$

The convergence in mean square sense of $Y_k^{(N)}$ to Y implies convergence in distribution of the same quantities, therefore by exploiting linearity of the limit

$$\sum_{k=1}^{N_e} F_{Y_k}^{(N)}(x) \cdot \mathsf{P}\left(Y \in A_k\right) \longrightarrow \sum_{k=1}^{N_e} F_{Y_k}(x) \cdot \mathsf{P}\left(Y \in A_k\right) = F_Y(x)$$

as $N \to +\infty$ for all continuity point x of $F_{Y_k}(x)$. Notice that $F_{Y_k}^{(N)}$ represents the cumulative density function of the truncated decomposition of degree N.

Let us prove that actually this approach defines a random variable that converges in probability to Y. First define

$$\eta(\omega) = \sum_{k=1}^{N_e} I_{A_k}(Y(\omega))Y_k(\omega)$$

which is a Σ-measurable function by considering Y_k extended to zero on Ω.

Since the elements of $\left\{I_{A_k}^{-1}(1)\right\}_{k=1}^{N_e}$ is a partition of Ω, $\{\eta \in B\} = \cup_{k=1}^{N_e}\{Y_k \in B \cap A_k\}$ for each Borel set $B \subset \mathbb{R}$. Then

$$\mathsf{P}(\eta \in B) = \mathsf{P}\left(\bigcup_{k=1}^{N_e}\{Y_k \in B \cap A_k\}\right) = \sum_{k=1}^{N_e}\mathsf{P}(Y_k \in B \cap A_k)$$

By the very definition of the distribution of each Y_k, $\mathsf{P}(Y_k \in B \cap A_k) = \mathsf{P}(Y \in B \cap A_k)$, since the measure induced on image space is given by Y. Thus

$$\mathsf{P}(\eta \in B) = \sum_{k=1}^{N_e}\mathsf{P}(Y \in B \cap A_k) = \sum_{k=1}^{N_e}\mathsf{P}(Y \in A_k)\frac{\mathsf{P}(Y \in B \cap A_k)}{\mathsf{P}(Y \in A_k)}$$

by exploiting the definition of $\mathsf{P}_k(Y_k \in B \cap A_k)$ in (2.6), and by equation (2.7)

$$\mathsf{P}(\eta \in B) = \sum_{k=1}^{N_e}\mathsf{P}(Y \in A_k)\cdot\mathsf{P}_k(Y_k \in B \cap A_k) = \mathsf{P}(Y \in B)$$

Therefore η and Y have the same distribution.

As second step let us prove that the sequence in N of random variables below

$$\sum_{k=1}^{N_e} I_{A_k}(Y(\omega))Y_k^{(N)}(\omega) \tag{2.11}$$

converges in probability to η. Thus let us consider the following event, for a fixed N

$$\begin{aligned}\left|\sum_{k=1}^{N_e} I_{A_k}(Y(\omega))Y^{(N)}(\omega) - \sum_{k=1}^{N_e} I_{A_k}(Y(\omega))Y_k(\omega)\right| &= \left|\sum_{k=1}^{N_e} I_{A_k}(\omega)\left(Y_k^{(N)} - Y_k\right)\right| \leq \\ &\leq \sum_{k=1}^{N_e} I_{A_k}(\omega)\left|Y_k^{(N)}(\omega) - Y_k(\omega)\right| \leq \\ &\leq \sum_{k=1}^{N_e}\left|Y_k^{(N)}(\omega) - Y_k(\omega)\right|\end{aligned}$$

for each $k \in \{1, \ldots, N_e\}$ the random variable Y_k and $Y_k^{(N)}$ are defined on disjoint sets, therefore the sum in last inequality becomes a union of sets. Moreover by considering a fixed real $\epsilon > 0$

$$\left\{\left|\sum_{k=1}^{N_e} I_{A_k}(Y(\omega))Y_k^{(N)}(\omega) - \sum_{k=1}^{N_e} I_{A_k}(Y(\omega))Y_k(\omega)\right| > \epsilon\right\} \subseteq \left\{\left(\bigcup_{k=1}^{N_e}\left|Y_k^{(N)}(\omega) - Y_k(\omega)\right|\right) > \epsilon\right\}$$

Again exploiting the definition of Y_k random variables

$$\left\{\left(\bigcup_{k=1}^{N_e}\left|Y_k^{(N)}(\omega) - Y_k(\omega)\right|\right) > \epsilon\right\} = \bigcup_{k=1}^{N_e}\left\{\left|Y_k^{(N)}(\omega) - Y_k(\omega)\right| > \epsilon\right\}$$

thus by monotonicity of the probability measure and keeping in mind that $Y_k^{(N)}$ converges in probability to Y_k

$$\mathsf{P}\left(\left|\sum_{k=1}^{N_e} I_{A_k}(Y(\omega))Y_k^{(N)}(\omega) - \sum_{k=1}^{N_e} I_{A_k}(Y(\omega))Y_k(\omega)\right| > \epsilon\right) \leq \mathsf{P}\left(\bigcup_{k=1}^{N_e}\left\{\left|Y_k^{(N)}(\omega) - Y_k(\omega)\right| > \epsilon\right\}\right)$$
$$= \sum_{k=1}^{N_e} \mathsf{P}\left(\left|Y_k^{(N)}(\omega) - Y_k(\omega)\right| > \epsilon\right) \to 0$$

That proves how $\sum_{k=1}^{N_e} I_{A_k}(\omega)Y_k^{(N)}(\omega)$ converges in probability to η, as $N \to +\infty$. Then, since η and Y are equally distributed,

$$\sum_{k=1}^{N_e} I_{A_k}(\omega)Y_k^{(N)}(\omega) \xrightarrow{P} Y(\omega) \qquad N \to +\infty$$

Convergence in $L^2(\Omega, \sigma(\xi), \mathsf{P})$ norm is not investigated since the global decomposition satisfies this requirement, and there is not other possibility in such Hilbert space.

In univariate settings lots of effort was made to distinguish the cases of weak convergence and strong convergence of the polynomial chaos expansion. This allowed to classify the random variables into two classes. In multi-element framework there is not any distinction among elements in $L^2(\Omega, \sigma(\xi), \mathsf{P})$, since the strongest convergence property is in probability.

As last remark notice that the connectivity property between two adjacent elements was never discussed, namely

$$Y_k^N(\xi) = Y_{k+1}^{(N)}(\xi) \qquad \xi \in \overline{A_k} \cap \overline{A_{k+1}}$$

for all $k = 1, \ldots, N_e - 1$, and where $\overline{A_k}$ and $\overline{A_{k+1}}$ represent the closure of the two elements. This continuity condition is not required since the Lebesgue measure of the interface between two random elements is zero, indeed statistics, convergence features are defined as a Lebesgue integration.

Chapter 3

Non-intrusive Spectral Projection

This chapter describes Non-Intrusive Spectral Projection (NISP) methods for approximation of the output of a process in presence of random inputs, parametrized by a finite number of random variables $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_m)$, defined on a probability space $(\Omega, \Sigma, \mathsf{P})$.

When a simulation of a real process occurs, uncertainties have to be taken into account. They have several origins: model discrepancy with respect to real process, lack of precise knowledge of physical parameters and inputs. NISP is a tool developed to describe how uncertainties propagate into a model.

Let be more precise by defining a model $\mathcal{M}$ of a process of interest with a finite number of random inputs $\mathbf{X} = (X_1, \ldots, X_J)$. The aim is to get features of the output random vector $\mathbf{Y} = (Y_1, \ldots, Y_H)$

$$\mathbf{Y} = \mathcal{M}(\mathbf{X})$$

The models considered admit a unique solution for almost all realization of the random inputs.

3.1 Motivation: Uncertainty Quantification

In many cases, the input data set may not be completely specified, for instance due to incomplete knowledge of the real system or because of intrinsic variability. The associated uncertainties may have different origins, for example, it may not be possible to determine precisely the boundary conditions of the system, or the forcing that it is subjected to. Furthermore, the physical properties of the system may not be exactly known.

Thus, though model equations may be deterministic, it may not be possible to rely on a single deterministic simulation because the input data are not precisely known, or they admit intrinsic variabilities. Consequently, one must associate an uncertainty resulting from incomplete knowledge of the input data with the simulation results.

Uncertainty Quantification (UQ) was introduced to get such a behavior . It is a fundamental step towards validation and certification of numerical methods to be used for critical decisions. Validation consists in comparing the simulations with respect to the real measurements, performed on real system, in order to achieve good prediction of reality.

Uncertainties can be classified as: epistemic or aleatory. The former is a potential deficiency, due to a lack of knowledge. It could be a consequence of assumptions introduced in the derivation of the mathematical model, such as limited accuracy in the measurement of the physical constant involved (it is also called reducible uncertainty or incertitude). Such kind of uncertainty source can be reduced by redefining the model and increasing its accuracy.

The latter is the intrinsic physical variability of the system. It is not linked with lack of knowledge and cannot be reduced (also referred as variability, stochastic uncertainty). The mathematical model is the noise and the probabilistic framework gets its behavior.

Aleatory uncertainties motivate the introduction of UQ methods. The most famous one is Monte-Carlo (MC), that relies on pseudo-random sampling of inputs parameter of a process. The model is simulated on these realizations, and the results allow to compute statistics, probability law, etc..., of the output. MC is a robust method, but it suffers of low convergence rate, thus many simulations are required for good approximation.

The spectral method studied in this chapter is based on radically different approach: it reconstructs the output Y in terms of a linear combination of functionals whose entries are known random variables called *germs* or *basic variables*. The functional representation is based on PCE of the output quantity, and the spectral method is characterized by the scheme used for the detection of the coefficients.

In this chapter only *Non-Intrusive* approach is presented. This technique computes the coefficients of output's PCE using a set of deterministic simulations of the process. This is the most attractive feature of NISP: it does not recast of the model into a probabilistic framework while at the same time it gets its random behavior.

3.2 Univariate NISP approach

Let us consider a model where both input and output are real-valued scalar random variables on $(\Omega, \Sigma, \mathsf{P})$

$$Y = \mathcal{M}(X)$$

Non-Intrusive Spectral Projection provides a decomposition of $\mathcal{M}(X)$ into a finite gPC basis $\{\Psi_0 \dots, \Psi_N\}$, hence it is the truncated PCE of Y. Moreover the basic random variable ξ inherits all assumptions made in the previous chapter. The spectral projection of the output is

$$Y^{(N)} = \sum_{i=0}^{N} c_i \Psi_i(\xi) \tag{3.1}$$

In order to achieve (3.1) the process has to be described in terms of ξ. Thus the key component is performing a transformation of variable from the original random input X to basic random variable ξ and then applying the PCE.

This step relies on inverse transform method (see Xiu [15], Proposition 2.11)

$$X = T(\xi) \qquad T(\xi) = F_X^{-1}\left(F_\xi(\xi)\right) \tag{3.2}$$

and the model becomes

$$\overline{\mathcal{M}}(\xi) = \left(\mathcal{M} \circ T\right)(\xi)$$

This ensures that actually the output depends on the basic random variable ξ.

Without loss of generality $Y \in L^2(\Omega, \sigma(\xi), \mathsf{P})$, thus it is always a finite second moment random variable. Indeed in real applications output random variables with infinite variance are not interesting.

Since $Y^{(N)}$ is the projection of the output onto $\overline{span_{\mathbb{R}}\{\Psi_0, \dots, \Psi_N\}}$, for each index $i \in \{0, \dots, N\}$ the coefficients in (3.1) are defined as

$$c_i = \frac{\left\langle \overline{\mathcal{M}}, \Psi_i \right\rangle_{\mathsf{P}}}{\|\Psi_i\|_{\mathsf{P}}^2} \tag{3.3}$$

When gPC basis is selected the values $\|\Psi_i\|_{\mathsf{P}}^2$ are tabled, thus only the computation of scalar product is left:

$$\left\langle \overline{\mathcal{M}}, \Psi_i \right\rangle_{\mathsf{P}} = \int_D \overline{\mathcal{M}}(x)\Psi_i(x) f_\xi(x) dx \tag{3.4}$$

where $D \subset \mathbb{R}$ is the support of probability density function f_ξ of the basic random variable ξ.

3.3 Univariate quadrature formula

In this section let us consider a fixed index $i \in \{0, \ldots, N\}$. The integral in (3.4) is computed via Gaussian quadrature formulas. This is the natural choice since the numerical scheme requires computation of orthogonal polynomials in $L^2(D, f_\xi(x)dx)$, that are elements of the gPC basis.

To shorten the notation let us define $g_i(x) = M(x)\Psi_i(x)$, then the right-hand side in (3.4) is approximated by

$$Q_N^{(1)}(g_i) = \sum_{k=1}^{N+1} g_i\left(\xi^{(k)}\right) w^{(k)} \tag{3.5}$$

where $\xi^{(k)} \in D$ and $w^{(k)} \in \mathbb{R}$, $k = 1, \ldots, N$ are nodes and weights of the chosen quadrature formula. It could be either Gauss-Legendre or Gauss-Hermite by means of non-negative weight function f_ξ.

In further sections the set $\left\{\xi^{(k)}\right\}_{k=1}^{N+1}$ is called a sampling of ξ due to the very definition of the integral (3.4): the quadratures nodes can be understood as particular realizations of basic random variable ξ.

Combining the equations (3.5), (3.4) and (3.3) it is clear how NISP relies on a set of deterministic model resolutions, corresponding to some specific realizations of ξ. Along this line, a deterministic simulation code can be used as a black-box, which associates to each realization of the parameters the corresponding model output.

3.4 Scilab's NISP toolbox

The NISP library is based on a set of three C++ classes that provide an object-oriented framework for uncertainty analysis. The Scilab toolbox implements a pseudo-object oriented interface to this library, so that the two approaches are consistent. The NISP library is characterized by three tools.

- The "randvar" class allows to manage random variables, specified by their distribution laws and their parameters. Once a random variable is created, one can generate random numbers from the associated law.
- The "setrandvar" class allows to manage a collection of random variables. This collection is associated with a sampling method, such as MonteCarlo, Quadrature, etc... It is possible to build the sampling and to get it back so that the experiments can be performed.
- The "polychaos" class handles a polynomial representation of the simulated model. Such object must be associated with a set of experiments which have been performed. This set may be read from a data file. The object is linked with a collection of random variables. Then the coefficients of the polynomial can be computed by integration (quadrature formulas). Moreover the mean, the variance and other feature of spectral decomposition can be directly computed from the coefficients.

Let us describe the general approach of a NISP codes. It is required that the user has a numerical solver (the model of the process) which has the form $Y = \mathcal{M}(X)$, where X is a realization of the uncertain input parameter and Y is the corresponding output of the simulation. The method is based on the following steps:

- As first step ξ random variable is defined, such as $\mathcal{N}(0,1)$ and $\mathcal{U}(0,1)$. This choice allows to define the basis for the polynomial chaos, denoted by $\{\Psi_k\}_{k\in\mathbb{N}}$. Depending on the type of random variable, the polynomials $\{\Psi_k\}_{k\in\mathbb{N}}$ are based on Hermite or Legendre polynomials. Laguerre polynomials are available but they are not used.
- Design Of Experiments (DOE) can be defined and the physical uncertain parameter X is detected with random variable transformation rules. These values are inputs of the numerical solver $\mathcal{M}$. Several types of DOE are available: *Monte-Carlo*, *Latin Hypercube Sampling*, etc...

- The simulation can be performed by evaluating the process on values defined in DOE, and then one can project the variable Y into the polynomial basis, by computing, for $i = 0, 1, \ldots, N$, its coefficients c_i.

3.5 NISP toolbox features

Let us discuss in detail features of NISP library by means of an example. Let

$$Y = e^X$$

be a process of interest, where $X \sim \mathcal{N}(\mu, \sigma^2)$ for $\mu = 1$ and $\sigma = 1/2$. The analytical solution is a lognormal distribution Y.

The code can be split into five sections: definition of the model, problem data, design of experiments, polynomial chaos definition and computation and post process analysis.

3.5.1 Definition of the model

The numerical solver, that represents the model $\mathcal{M}(X)$ is defined either as a Scialb function or an external one. The former is used in this example:

```
function y = RV(x)
      y = exp(x);
endfunction
```

the keywords `function` and `endfunction` are used to define a function within Scilab, while the inputs `x` is a realization of the input random variable.

3.5.2 Problem data definition

In this section features of the model are defined, such as parameters of random inputs, physical data involved in the model definition, maximum degree of the decomposition and number of output random variables.

For such example they are

```
N= 10;
nodeint = 10;
noutput = 1;
mu = 1;
sigma = 0.5;
```

where N is the degree of PCE truncation, `mu` and `sigma` represent the mean and the standard deviation of the input. Moreover it is set the number of integration nodes used for quadrature formula, it has to be at leas greater or equal to `N`.

3.5.3 Design of Experiment (DOE)

Design of experiment is one of the most important step in NISP approach. It consists in definition of gPC basis and computation of quadrature nodes for the detection of the coefficients.

In NISP library Legendre polynomials on $[-1, 1]$ and physicists' Hermite polynomials are implemented. In order to select one of the two classes it is enough to set a normalized random variable: either Uniform $\mathcal{U}(0, 1)$ or Normal $\mathcal{N}(0, 1)$.

In this setting the most natural decomposition is the second one, thus

```
xi = randvar_new('Normale');
srvxi = setrandvar_new();
setrandvar_addrandvar(srvxi,xi);
```

`xi` represents a normalized random variable that inhabits a container `srvxi` defined by `setrandvar_new()`. Then the random inputs are set

```
x = randvar_new('Normale',mu,sigma);
srvx = setrandvar_new();
setrandvar_addrandvar(srvx,x);
```

Since $X \sim \mathcal{N}(\mu, \sigma^2)$, the mean and the standard deviation of X are specified as inputs of `randvar_new()`.

The key part of DOE is the detection of quadrature nodes on which the process will be evaluated. This requires two steps

- The first one amounts to

  ```
  setrandvar_buildsample(srvxi,"Quadrature",nodeint);
  ```

  where `"Quadrature"` specifies that a Gaussian quadrature rule is used, it could be Gauss-Legendre or Gauss-Hermite by means of `srvxi`, while `nodeint` sets the number of quadrature points. The result is the grid of quadrature nodes, i.e. the zeros of (`nodeint` + 1)-th orthogonal polynomial.

- These points are transformed into a sampling of the input by

  ```
  setrandvar_buildsample(srvx,srvxi);
  ```

Notice that `xi` is not the basic random variable, indeed it should be $\mathcal{N}\big(0, 1/2\big)$ rather than $\mathcal{N}(0, 1)$. This is motivated by the algorithm used for computing quadratures nodes: it requires probabilists' Hermite polynomials, whose zeros and associated weights are

$$\left\{\xi_{\text{old}}^{(k)}\right\}_{k=1}^{N+1} \qquad \left\{w_{\text{old}}^{(k)}\right\}_{k=1}^{N+1}$$

Then these values are transformed into the correct ones (for the weight function $f_\xi(x) = \frac{1}{\sqrt{\pi}} e^{-x^2}$ in (3.4)) using a linear transformation

$$\xi^{(k)} = \sqrt{2} \cdot \xi_{\text{old}}^{(k)} \qquad w^{(k)} = \frac{w_{\text{old}}^{(k)}}{\sqrt{\pi}}$$

For Gauss-Legendre integration the quadrature nodes and weights are already set to be compatible with Legendre polynomials on $[-1, 1]$, thus no transformation occurs.

3.5.4 Polynomial Chaos definition and computation

Spectral projection of the output is defined by means of PCE:

```
pc = polychaos_new(srvxi,noutput);
np = setrandvar_getsize(srvxi);
polychaos_setsizetarget(pc,np);
```

The routine `polychaos_new` generates the object `pc` and it requires the basic random variable (`srvxi`) and the number of outputs (`noutput`) as inputs. Moreover the routine `polychaos_setsizetarget` reserves memory's space needed to save the evaluation of the process at quadrature nodes: it is related to the size of the sampling defined in DOE. Then the process is evaluated and the degree of the decomposition is set

```
inputdata = setrandvar_getsample(srvx);
outputdata = RV(inputdata);
polychaos_settarget(pc,outputdata);
polychaos_setdegree(pc,degre);
```

Al data needed for coefficients' computation are defined, thus

```
polychaos_computeexp(pc,srvxi,'Integration');
```

The underlying C++ method collects all information defined, such as quadrature nodes, evaluation of the process, and then it computes the gPC basis and PCE getting the coefficients via a numerical scheme compatible with the selected quadrature rule.

3.5.5 Post process analysis

Once PCE of the output is detected, NISP library provides lots of methods for post process analysis, such as mean, variance and sampling of $Y^{(N)}$.

```
average = polychaos_getmean(pc);
var = polychaos_getvaraince(pc);
polychaos_buildsample(pc,"Lhs",NMC);
sout = polychaos_getsample(pc);
```

The routine `polychaos_buildsample` requires: the PCE expansion, the sampling technique that will be used, and the number of realizations. `Lhs` stands for *Latin Hypercube Sampling*, it is achieved by forcing the sampler to draw a realization within equiprobable bins in the parameter range.

Other possibilities are `MonteCarlo`, that is the pseudo-random sampling and *Quasi Monte Carlo* (`QmcSobol`), defined as a low discrepancy sequence of points generated by Sobol algorithm, that maximizes the uniformity of the sample points. In Figure 3.1 are shown these three possibilities by simulating two independent $\mathcal{U}(0,1)$ for different sampling's size $N = 50, 150, 500$.

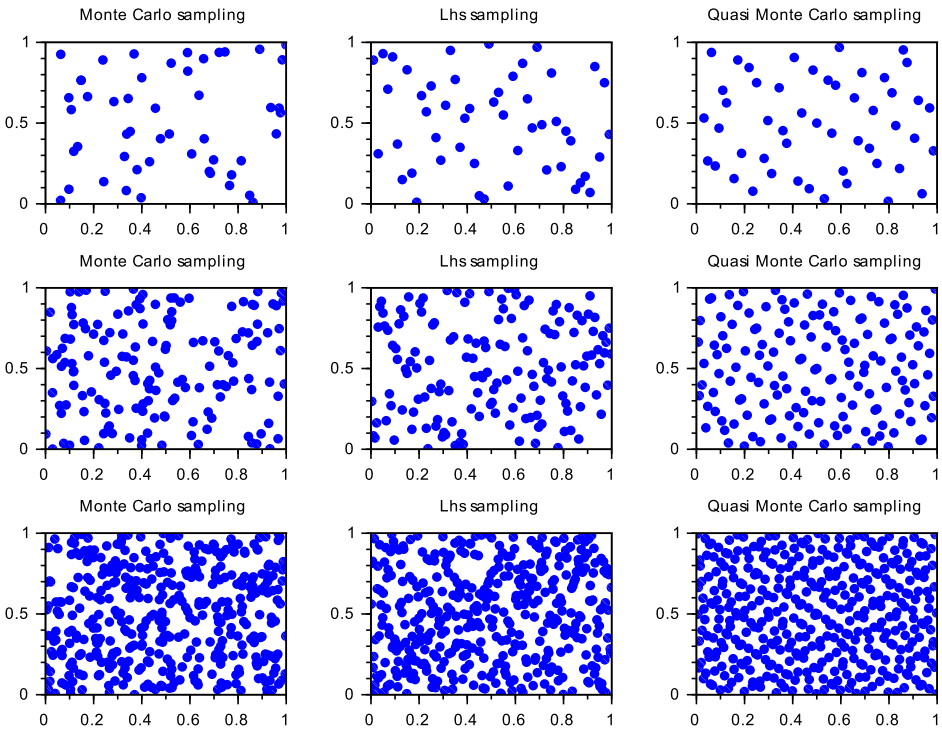

Figure 3.1: Comparison between three available sampling techniques for $N = 50, 150$ and 500 from top to bottom

Once the sampling of the PCE is detected, we can compare it with the analytical probability density function, by means of an histogram. (Figure 3.2).

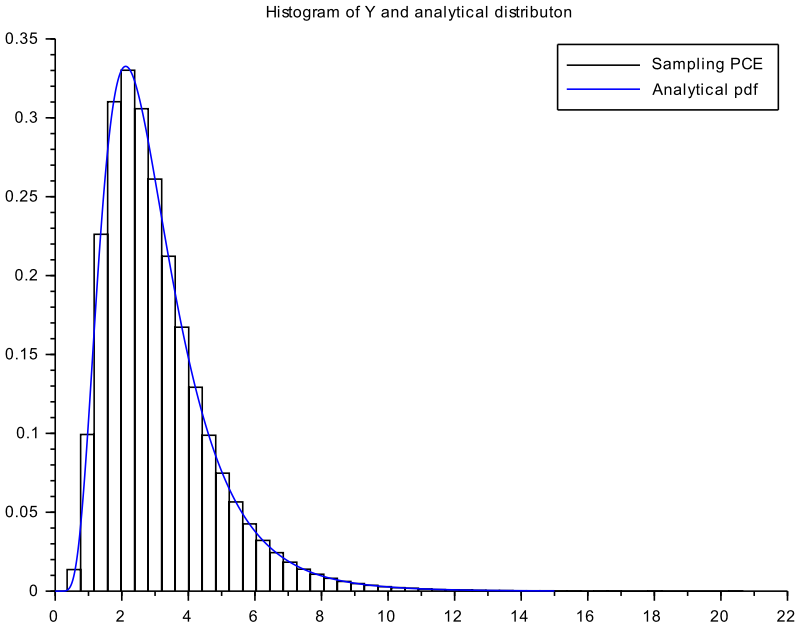

Figure 3.2: Comparison between sampling of size 20000 of PCE and the analytical probability density function (pdf) of lognormal distribution

Moreover the sampling of PCE allows to compute the mean square error of the approximation $Y^{(N)}$, that is

$$\epsilon^{(N)} = \int_{\Omega} E^{(N)}\big(\xi(\omega)\big) dP(\omega) \tag{3.6}$$

where

$$E^{(N)}\big(\xi(\omega)\big) = \left(\sum_{i=0}^{N} c_i \Psi(\xi(\omega)) - \mathcal{M} \circ T(\xi(\omega))\right)^2$$

Two strategies are used: either $\epsilon^{(N)}$ is computed with simulation (Monte Carlo) approach or via trapezoidal integration rule.

The former is characterized by approximating the integral with

$$\epsilon_{MC}^{(N)} = \frac{1}{NMC} \sum_{j=1}^{NMC} E^{(N)}\left(\xi^{(j)}\right)$$

where $\{\xi^{(j)}\}_{j=1}^{NMC}$ is a set of independent realizations of the basic random variable. This estimate suffers of low convergence rate, thus to achieve good approximation the size of the sampling has to be large enough. In further examples it is always set as $NMC = 20000$.

The second approach computes the integral using `inttrap` Scilab's routine, which requires: a set Ξ of values, compatible with the input of the process, and the evaluation of $E^{(N)}(\cdot)$ on this set.

Since $Y^{(N)}$ and $\overline{\mathcal{M}}$ have the basic random variable as input, the set Ξ can be selected as sampling of ξ. By construction the points in Ξ are not equally spaced, but the quadrature formula still holds.

This procedure is implemented as

```
setrandvar_buildsample(srvxi,"Lhs",NT);
setrandvar_buildsample(srvx,srvxi);
sample1 = setrandvar_getsample(srvxi);
sample2 = (sqrt(2)/2)*sample1;
sample3 = setrandvar_getsample(srvx);
```

The set Ξ is `sample2`, while `sample3` is required to evaluate the process (see (3.2)). Less points are needed to achieve good convergence rate, thus `NT` $= 200$. Then let us evaluate the process on `sample3`

```
val1 = RV(sample3);
```

while the evaluation of $Y^{(N)}$ on Ξ is a sampling `sout2` of the PCE of size `NT`. This is always true whenever `sample1` and `sout` share the sampling technique (in this case `Lhs`).

Then the approximation of the mean square error $\epsilon_{TP}^{(N)}$ is computed by

```
error = inttrap(sample2,((sout2-val1).^2).*((1/sqrt(%pi))*exp(-sample2.^2)));
```

In order to compare such approaches let us run the previous code for a range of degree $N = 1, 2, \ldots, 10$. The results are shown in Table 3.1,

N	$\epsilon_{MC}^{(N)}$	$\epsilon_{TP}^{(N)}$
1	3.2078e-01	2.4476e-01
2	2.1372e-02	1.2786e-02
3	2.1468e-03	5.7541e-04
4	1.2954e-03	2.9575e-05
5	1.3047e-03	1.1631e-06
6	1.3015e-03	3.4578e-08
7	1.2979e-03	1.0795e-09
8	1.2975e-03	2.8550e-11
9	1.2975e-03	6.5099e-13
10	1.2975e-03	1.4689e-14

Table 3.1: Table of mean square error computed with Monte Carlo and trapezoidal rule

Moreover Table 3.1 proves convergence in mean square sense of $Y^{(N)}$ to $\overline{\mathcal{M}}$.

3.6 Multivariate NISP approach

Let us consider a model $\mathcal{M}$ characterized by a random vector $\mathbf{X}$ of inputs, whose components are mutually independent, while the random output Y is a scalar quantity.

By multivariate PCE theory Y can be decomposed into the gPC basis, where the basic random vector $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_M)$ is defined on $(\Omega, \Sigma, \mathsf{P})$, moreover its components are mutually independent.

Also in this framework the input vector $\mathbf{X}$ has to be expressed in terms of $\boldsymbol{\xi}$, by extending inverse transform method to this setting, therefore

$$\mathbf{X} = \mathbf{T}(\boldsymbol{\xi})$$

hence the model becomes

$$\overline{\mathcal{M}}(\boldsymbol{\xi}) = \mathcal{M} \circ \mathbf{T}(\boldsymbol{\xi})$$

In Section 2.3 of Chapter 2 the spectral projection is defined using multi-index $\mathbf{i}$ notation, sorted by *graded lexicographical order* and truncated by means of total degree N, thus

$$Y^{(N)} = \sum_{|\mathbf{i}| \leq N} c_{\mathbf{i}} \Psi_{\mathbf{i}}(\boldsymbol{\xi})$$

where for each multi-index $|\mathbf{i}| \leq N$

$$c_{\mathbf{i}} = \frac{\langle \overline{\mathcal{M}}, \Psi_{\mathbf{i}} \rangle_P}{\|\Psi_{\mathbf{i}}\|_{\mathsf{P}}^2}$$

From now on let us consider a fixed multi-index $\mathbf{i}$ such that $|\mathbf{i}| \leq N$. As for univariate case, the main effort is computing the scalar product, that is

$$\langle \overline{\mathcal{M}}, \Psi_{\mathbf{i}} \rangle_{\mathsf{P}} = \int_{\mathbf{D}} \overline{\mathcal{M}}(\mathbf{x}) \Psi_{\mathbf{i}}(\mathbf{x}) f_{\boldsymbol{\xi}}(\mathbf{x}) d\mathbf{x} \tag{3.7}$$

where $\mathbf{x} = (x_1, \ldots, x_M) \in \mathbb{R}^M$ and $\mathbf{D} \subset \mathbb{R}^M$ is the support of the joint probability density function of the vector $\boldsymbol{\xi}$, defined as

$$f_{\boldsymbol{\xi}}(\mathbf{x}) = \prod_{m=1}^{M} f_{\xi_m}(x_m)$$

due to independence of the random variables $\xi_1, \ldots, \xi_M$.

In order to shorten the notation let us set $\mathbf{g}_{\mathbf{i}}(\mathbf{x}) = \overline{\mathcal{M}}(\mathbf{x}) \Psi_{\mathbf{i}}(\mathbf{x})$, then the integral in (3.7) is computed using multivariate Gaussian quadrature formula, defined by tensorization of 1D-formulas:

$$\left(Q_1^{(1)} \otimes \cdots \otimes Q_M^{(1)}\right) = \sum_{i_1=1}^{N_1} \cdots \sum_{i_M=1}^{N_M} \mathbf{g}_{\mathbf{i}} \left(\xi_1^{(i_1)}, \ldots, \xi_M^{(i_M)}\right) w_1^{(i_1)} \cdot w_M^{(i_M)}$$

where for each $m \in \{1, \ldots, M\}$ the sets

$$\left\{\xi_m^{(i_m)}\right\}_{i_m=1}^{N_m} \qquad \left\{w_m^{(i_m)}\right\}_{i_m=1}^{N_m}$$

represent the quadrature nodes and the weights of the selected quadrature formula along x_m-direction.

Clearly, the above formula requires $\prod_{m=1}^{M} N_m$ function evaluations. Therefore, when the number of input random variables is small, full tensor product quadrature is a very effective numerical tool. On the other hand the approximation based on tensor product grids suffers from the curse of dimensionality since the number of collocation points in a tensor grid grows exponentially fast in the number of input random variables.

The modification of previous Scialb's code for such setting concerns only the definition of correct basic random variables and random inputs. Thus

```
xi1 = randvar_new('Normale');
xi2 = randvar_new('Normale');
srvxi = setrandvar_new();
setrandvar_addrandvar(srvxi,xi1);
setrandvar_addrandvar(srvxi,xi2);

x1 = randvar_new('Normale',mu1,sigma1);
x2 = randvar_new('Normale',mu2,sigma2);
srvx = setrandvar_new();
setrandvar_addrandvar(srvx,x1);
setrandvar_addrandvar(srvx,x2);
```

The other code lines does not change.

3.7 NISP approach for output random vector

A further possibility is the application of NISP to vectorial random output $\mathbf{Y}$, for instance let us consider NISP procedure applied to resolution of ODEs and PDEs, whose output is the solution evaluated at spatial discretization points.

Te process considered is

$$\mathbf{Y} = \mathcal{M}(X)$$

where for simplicity X is considered scalar. Moreover let us assume that the input X and the basic random variables ξ, satisfies the assumption of previous sections. Then NISP for output random vector is the application of PCE to each component Y_j of $\mathbf{Y}$.

In particular when NISP is involved in description of the solution u of a generic ordinary differential equation integrated on D_M, each components Y_j can be understood as the solution restricted to the discretization point $x_j \in D_M$, thus the PCE computed describes the behavior of u on this specific point x_j. Moreover if the solution comes form a PDEs, thus it is time dependent, a vectorial PCE can be computed for each time step, keeping the same interpretation of each component Y_j.

The changes on code lines are about the definition of the model and the declaration of the number of outputs (`noutput`).

```
pc = polychaos_new(srvxi,noutput);
```

The extension to multivariate decomposition of vectorial NISP relies on computing multivariate PCE for each component of the output vector $\mathbf{Y}$.

Chapter 4

Application of NISP toolbox

This chapter deals with four applications of NISP toolbox for Scilab software. The first concerns the decomposition of two bimodal random variables, where a particular attention is dedicated to convergence properties of the PCE computed. The second describes the solution of a non-linear ODE, when a physical parameter is characterized by uncertainty. In the third and forth examples the behavior of two PDEs solutions in presence of uncertainty is analyzed.

The softwares used in this section are Scilab 5.5.1, FreeFem++ 3.3.2, and Pyclaw (based on Clawpack 5.2.0). The first provides the environment to employ tools of NISP toolbox. The second is a software that implements Finite Element Methods for solving PDEs and the last is a python interface of Clawpack that solves linear and non-linear hyperbolic systems of conservation laws.

4.1 Bimodal

Let us state the definition of Bimodal random variables

Definition 4.1 *Given a random variable Y, whose probability density function is $f(x)$. Y is called bimodal distribution if there exist only two distinct points x_1 and x_2 each of them is either a local maximum or such that* $\lim_{x\to x_i} f(x) = +\infty$.

4.1.1 Arcsine distribution

Let $[a, b]$ be a real interval, then a random variable Y follows the arcsine distribution if its cumulative density function is

$$F(x) = \frac{1}{\pi}\arcsin\left(2\frac{x-a}{b-a} - 1\right) + \frac{1}{2} \tag{4.1}$$

Moreover its probability density function (pdf) is

$$f(x) = \frac{1}{\pi}\cdot\frac{1}{\sqrt{(x-a)(b-x)}} \tag{4.2}$$

4.1.2 PCE decomposition of an arcsine distribution

In this section an arcsine random variable Y of support $D = [-3, 1]$ is considered. By equations (4.1) and (4.2) its probability and cumulative density functions are

$$F(x) = \frac{1}{\pi}\arcsin\left(\frac{x+1}{2}\right) + \frac{1}{2}$$
$$f(x) = \frac{1}{\pi}\frac{1}{\sqrt{(x+3)(1-x)}}$$

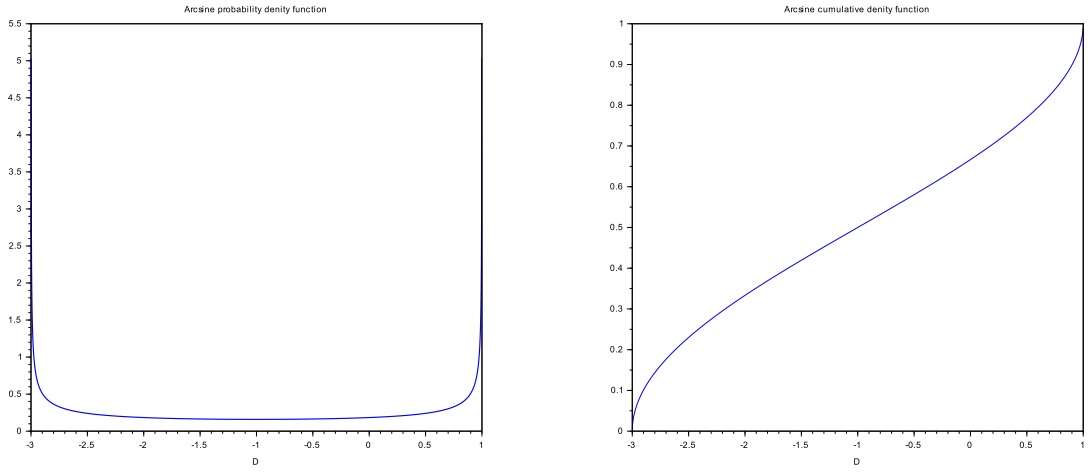

Figure 4.1: Probability and cumulative density functions for arcsine distribution of support $D = [-3, 1]$

whose graphs are shown in Figure 4.1.

The arcsine random variable can be written as

$$Y = T\big(\mathcal{U}(0,1)\big) = F^{-1}\big(\mathcal{U}(0,1)\big) \tag{4.3}$$

by inverse transform method (see Xiu [15]), where

$$F_Y^{-1}(x) = 2\sin\left(\pi x - \frac{\pi}{2}\right) - 1$$

Moreover it achieves the requirement to be decomposed using NISP toolbox, since Y is expressed as an output of a process T. The Scilab routine, that implements T, is

```
function y =RV(x,a,b)
      y = 0.5*(b-a)*(sin(%pi*(x-1/2))+1)+a;
endfunction
```

where the domain $D = [a, b]$ has to be specified as an input. Since the support of the random variable is compact the gPC basis chosen is the class of Legendre polynomials on $[-1, 1]$. Moreover by the very definition of the process (4.3), the input random variable is $\mathcal{U}(0, 1)$. Then the truncated expansion is

$$Y^{(N)} = \sum_{i=0}^{N} c_i \Psi_i(\xi) \tag{4.4}$$

where $\xi \sim \mathcal{U}(-1, 1)$.

The main goal is proving the convergence in mean square sense of (4.4) to Y as $N \to +\infty$. Thus for a set of degree $N = \{1, \ldots, 9\}$ the mean square error $\epsilon^{(N)}$ (3.6) is computed. The numerical integration is achieved using trapezoidal rule, these values are shown in Figure 4.2.

Such behavior of the mean square error is due to analytical expression of $\epsilon^{(N)}$, that is

$$\epsilon^{(N)} = \sum_{i=N+1}^{+\infty} c_i^2 \|\Psi_i\|_{\mathsf{P}}^2$$

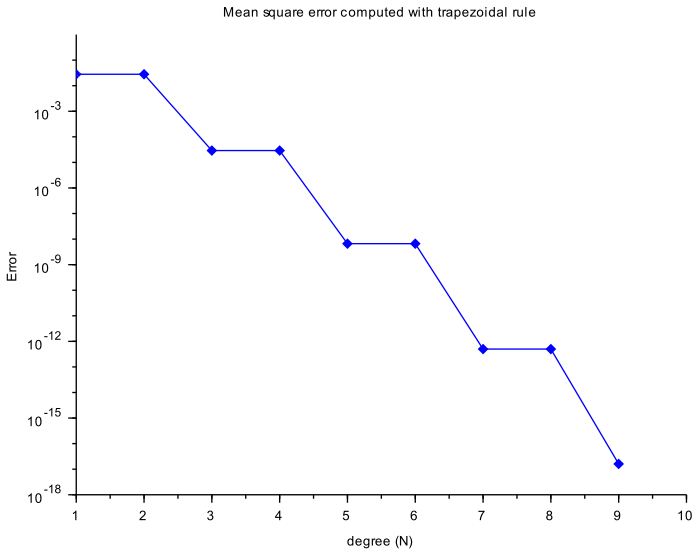

Figure 4.2: Plot of $\epsilon^{(N)}$ approximated by trapezoidal rule (`inttrap` routine)

this is motivated by property of orthogonal projection on a closed subset in the Hilbert space considered (see [1] for details). Upon fixing an admissible degree N, the local error is

$$\left|\epsilon^{(N+1)} - \epsilon^{(N)}\right| = c_{N+1}^2 \|\Psi_{N+1}\|_{\mathrm{P}}^2$$

Therefore in this case $\epsilon^{(N)}$, estimated by trapezoidal rule, behaves as in Figure 4.2 since the squares of the coefficients with even indexes are null.

4.1.3 Mixture of random variables

The *mixture* of random variable is a general technique that defines a new random variable by combining a finite set probability density functions. Let $j = 1, \ldots, d$ be a finite set of indexes and let $q_j \in (0, 1)$ be a collection of real values such that

$$\sum_{j=1}^{d} q_j = 1$$

Moreover let us consider a finite collection of random variables $\{X_j\}_{j=1}^d$ defined on the same probability space and with the same support D.

The mixture of $\{X_j\}_{j=1}^d$, weighted by $\{q_j\}_{j=1}^d$, is a random variable Y such that

$$f_Y(x) = \sum_{j=1}^{d} q_j f_{X_j}(x) \tag{4.5}$$

is its probability density function. Furthermore f_Y fulfills all properties of a probability density function: it is positive for all $x \in D$ and the integral on whole support is one, indeed

$$\int_D f_Y(x)\, dx = \int_D \sum_{j=1}^{d} q_j f_{X_j}(x) dx = \sum_{j=1}^{d} q_j \int_D f_{X_j}(x) dx = \sum_{j=1}^{d} q_j = 1$$

By Integrating (4.5) the cumulative density function of Y is achieved

$$F_Y(x) = \sum_{j=1}^{d} q_j F_{X_j}(x) \tag{4.6}$$

4.1.4 PCE for Mixture of two normal random variables

In this example Y is defined as a mixture of two normal random variables:

$$X_1 \sim \mathcal{N}\left(-\frac{3}{2}, \frac{1}{16}\right) \qquad X_2 \sim \mathcal{N}\left(1, \frac{1}{16}\right)$$

the weights are set as $q_1 = \frac{1}{10}$ and $q_2 = \frac{9}{10}$. Moreover the analytical expression of its probability density function is

$$f_Y(x) = \frac{1}{10}\frac{1}{\sqrt{2\sigma_1}} e^{\frac{(x-\mu_1)^2}{2\sigma_1^2}} + \frac{9}{10}\frac{1}{\sqrt{2\sigma_2}} e^{\frac{(x-\mu_2)^2}{2\sigma_2^2}}$$

where μ_1, σ_1 and μ_2, σ_2 are the means and standard deviations of the two random variables X_1, X_2. Such function is represented in Figure 4.3.

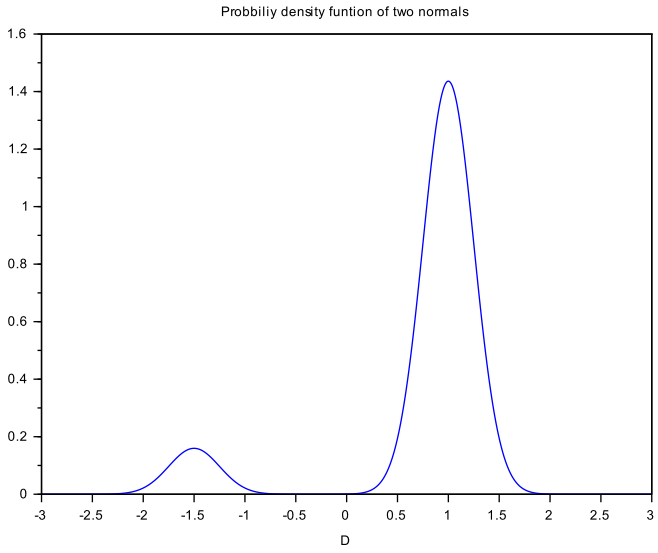

Figure 4.3: Mixture of two normal random variables

NISP approach requires the definition of the random variable Y as an output of a function, this is accomplished by the inverse transform method, thus

$$Y = F_Y^{-1}(\mathcal{U}(0,1)) \tag{4.7}$$

The analytical detection of the inverse is not an easy task, thus in computational framework it is achieved by linear interpolation

```
function y = RV(x,mu,sigma,xloc,ylox)
//
// INPUTS
//
// x = a realization of the input random variable U(0,1)
// mu = the vector of mean values mu(1) = E[X_1] and mu(2) = E[X_2]
// sigma = the standard deviations, sigma(1) = stdv(X_1) and sigma(2) =
stdv(X_2)
// xloc = the x-values for computing the inverse
// yloc = the evaluation at xloc of the cdf Y
//
// OUTPUT
//
```

```
// y = a realization of random variable Y

y = interp1([min(xloc);yloc;max(xloc)],[0;xloc;1],x,'linear');
endfunction
```

Since two random variables are used to define Y, it is enough to set q_1 while $q_2 = 1 - q_1$. Notice that the first and last entries of interpolation nodes (see in `interp1` routine) are changed in order to avoid infinity values coming from `distfun_normcdf` function.

By (4.7) the input random variables is a $\mathcal{U}(0,1)$, while the germ ξ is set as $\mathcal{U}(-1,1)$. Thus the truncated PCE of degree N is

$$Y^{(N)} = \sum_{i=0}^{N} c_i \Psi_i(\xi) \tag{4.8}$$

where $\{\Psi_i\}_{i=0}^{N}$ are the first N Legendre polynomials on $[-1,1]$. The polynomial chaos expansion is computed for a set $N = \{5, 10, 15, \ldots, 70\}$.

Then the mean square error of $Y^{(N)}$ is computed via Monte Carlo method, hence simulating the integral (3.6). These values are displayed in Figure 4.4.

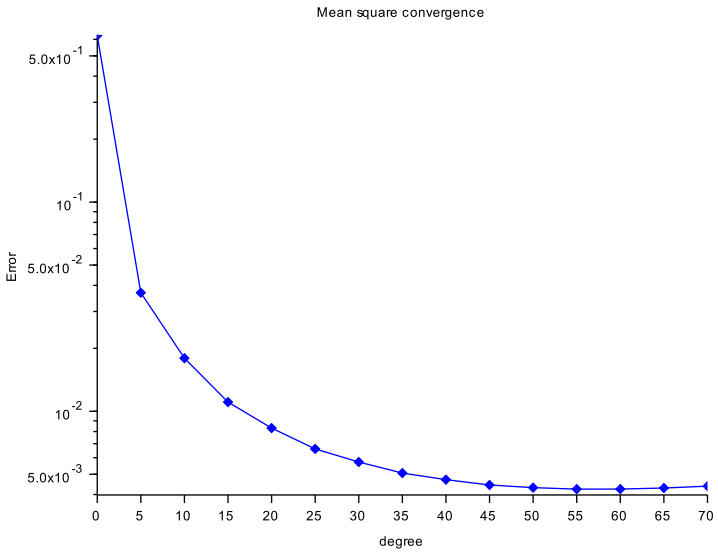

Figure 4.4: Semilogarithmic plot of $\epsilon^{(N)}$ approximated by Monte Carlo method for a global PCE of a mixture of two normal random variables

It turns out that $Y^{(N)}$ is affected by low convergence rate for $N \to +\infty$. Furthermore this is confirmed by Figure 4.5, where two samplings of size 20000, for the polynomials chaos expansions of degree $N = 30$ and $N = 70$, are computed.

A possibility to overwhelm such behavior is increasing the degree of the expansion, but this is proportional to computational costs for the PCE detection.

In order to get better approximation using lower degree expansions a multi-element decomposition is used. The approach is based on theory developed in section 2.5.2 of the second chapter. For simplicity only two elements (intervals) are considered A_1 and A_2, such that for every $k = 1, 2$ with fixed $p_k \in (0,1)$

$$p_k = \int_{A_k} f_Y(x)dx$$

In this example $p_1 = \frac{1}{10}$ and $p_2 = \frac{9}{10}$. Since $p_1 + p_2 = 1$, it is enough to set

$$p_1 = p \qquad p_2 = 1 - p$$

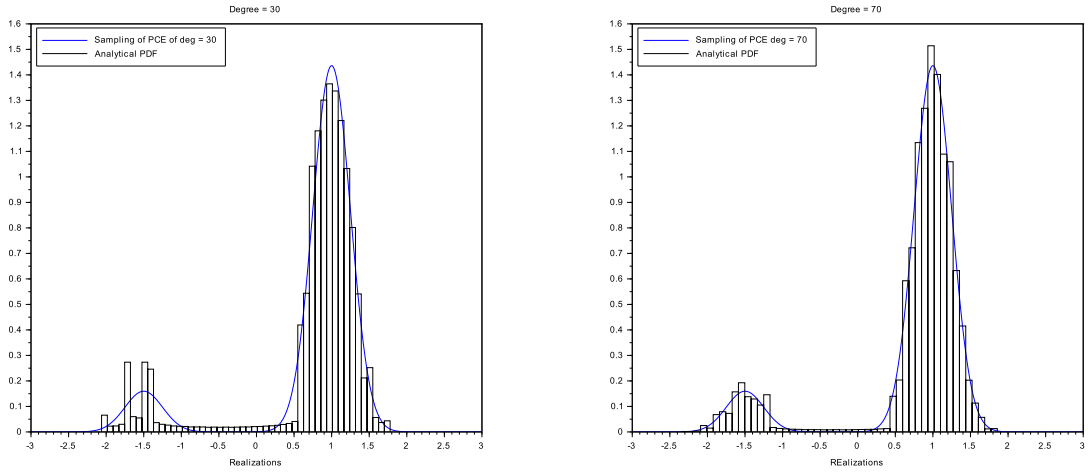

Figure 4.5: Two samplings of PCE for degree $N = 30$ (left) and degree $N = 70$ (right) relative to a mixture of two normally distributed random variables

When the partition of the support is detected, two random variables Y_1 and Y_2 are defined by means of (2.8). Moreover their probability density functions are

$$f_{Y_1}(x) = \frac{1}{p} f_Y(x) \mathbb{1}_{A_1}(x) \qquad f_{Y_2}(x) = \frac{1}{1-p} f_Y(x) \mathbb{1}_{A_2}(x)$$

where $\mathbb{1}_{A_k}(x)$ is the indicator function on A_k sets.

Then two polynomial chaos expansions are computed, one for each interval of the partition

$$Y_1^{(N)} = \sum_{i=0}^{N} c_{1,i} \Psi_{1,i}(\xi) \qquad Y_2^{(N)} = \sum_{i=0}^{N} c_{2,i} \Psi_{2,i}(\xi)$$

notice that they are truncated at the same degree and the basic random variable ξ is $\mathcal{U}(-1, 1)$ for both decompositions. Moreover the degree is set as $N = 15$.

In order to compare the analytical probability density function f_Y and this multi-element decomposition, two samplings of the two truncated decompositions are combined by means of (2.11). The comparison is based on two samplings of size 20000, whose realizations are set into two distinct histograms that are eventually gathered together since they lie on disjoint intervals (A_1 and A_2 respectively).

To make reasonable comparison the area under the histogram has to be one. For such discussion the equations (2.6) and (2.7) has to be taken into account. They state that the areas of the two histograms (one for $Y_1^{(N)}$ and the other for $Y_2^{(N)}$) have to be set to p and $1 - p$ respectively.

Due to this the routine `histogram_ME` is implemented and the two samplings are put together and compared with analytical probability density function in Figure 4.6.

4.2 Non-linear ODE

The one dimensional non-linear ordinary differential equation considered is

$$\left(\frac{1-2u}{2}\right) u_x - \mu u_{xx} = 0 \tag{4.9}$$

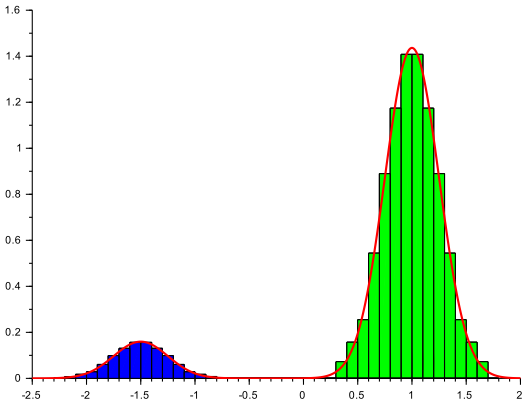

Figure 4.6: Multi-element PCE histogram with 50 bins for mixture of two normal random variable

where the viscosity μ is a strictly positive parameter and the function

$$f(u) = \frac{1-2u}{2}$$

is the the non-linear advection velocity. The equation (4.9) can be recast into conservative form

$$-\mu u_{xx} - \left(\frac{(1-2u)^2}{8}\right)_x = 0 \tag{4.10}$$

In this section the behavior of the solution $u(x)$ of (4.9) is studied, when the viscosity is a random variable space independent. Two different situations are studied: μ is either normal or lognormal distribution.

To avoid numerical instability of the solver, that occurs for small μ values, the viscosity is set as

$$\mu \sim \mathcal{N}\left(\frac{1}{4}, \frac{1}{2500}\right)$$

where the negative values have null probability. While the lognormal distribution is

$$\mu \sim e^X \qquad X = \mathcal{N}\left(\frac{1}{4}, \frac{1}{16}\right)$$

Their probability density functions are displayed in Figure 4.7.

4.2.1 Analytical solution

The analytical solution of (4.9) can be computed and it will be used to compare the results achieved by PCE. By linearity of the derivation the equation (4.10) becomes

$$\left[-\mu u_x - \frac{(1-2u)^2}{8}\right]_x = 0$$

thus

$$-\mu u_x - \frac{(1-2u)^2}{8} = A$$

where A is a real parameter, that does not depend on μ. Therefore

$$u_x = -\frac{(1-2u)^2}{8\mu} - \frac{A}{\mu} \tag{4.11}$$

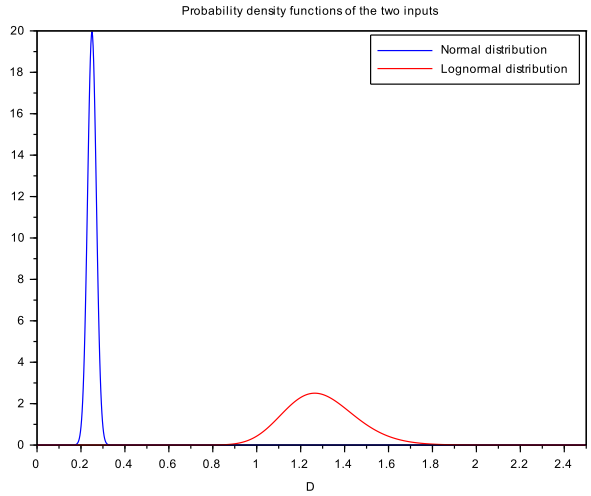

Figure 4.7: Probability density functions for μ distributed as normal (blue) and lognormal (red) random variables

Hence A identifies three cases by means of its sign: A null, negative and positive.

When A is null the solution is

$$u(x) = \frac{1}{2} + \frac{2\mu}{x+c} \tag{4.12}$$

If A is negative one achieves

$$u(x) = \frac{1}{2} + \sqrt{2\,|A|}\tanh\left(\sqrt{2\,|A|}\frac{x+c}{2\mu}\right) \tag{4.13}$$

Whether A is positive, the integration of (4.11) gives

$$u(x) = \frac{1}{2} - \sqrt{2A}\tan\left(\sqrt{2A}\ \frac{x+c}{2\mu}\right) \tag{4.14}$$

Three free parameters A and c are set in order to achieve boundary conditions.

4.2.2 First order ODE

For further discussion it is useful to recast (4.9) in a system of first order ODEs, thus for $u(x) = p$ and $u_x(x) = q$, we get

$$\begin{cases} \dot{p} = q \\ \dot{q} = \dfrac{1-2p}{2\mu}q \end{cases} \tag{4.15}$$

where the vector field is called $V(p, q)$,

Moreover let us consider in this setting the equation (4.11), which becomes

$$q = -\frac{(1-2p)^2}{8\mu} - \frac{A}{\mu} \tag{4.16}$$

Then let us define the following function:

$$H(p,q) = -\frac{(1-2p)^2}{8\mu} - \frac{A}{\mu} - q$$

this is a *first integral* of the system in (4.15), indeed $\nabla H(p, q)$ is orthogonal with respect to the vector filed $V(p, q)$.

4.2.3 PCE of the output

Since μ is a random variable, defined on a suitable probability space $(\Omega, \Sigma, \mathsf{P})$, the solution is an aleatory quantity too

$$u\big(x, \mu(\cdot)\big) : \Omega \to \mathbb{R}$$

where x is fixed value within the integration domain.

NISP technique requires the definition of the solver for computing the solution of the differential equation. In order to avoid the blow up of the solution, the differential equation (4.10) is solved as a boundary values problem (BVPs). The numerical solution is computed on an uniform discretization of the domain $D = [a, b]$, defined as

$$D_M = \Big\{ x_j \ : \ \ x_j = a + h(j-1) \ , \ 1 \leq j \leq M \Big\}$$

where $h = (b-a)/(M-1)$. In these computations $M = 101$. Thus for each x_j a polynomial chaos expansion of degree N is computed, hence for $j = 1, \ldots, M$

$$u_j^{(N)} = u^{(N)}(x_j, \xi) = \sum_{i=0}^{N} c_i(x_j)\Psi_i(\xi)$$

where u_j represents the j-th component of the discretized solution $\mathbf{u}$. Furthermore the decomposition is made using the set of physicists' Hermite polynomials, thus the basic random variable is set as $\xi \sim \mathcal{N}(0, 1/2)$.

It is analyzed the behavior of the PCE for A null, A negative and A positive, since three solutions of the ODE are available. Moreover for each case two polynomial chaos expansions are computed one for every input distribution.
Suitable Dirichlet boundary conditions are set in order to ensure convergence of the numerical solution to the selected analytical solution $u(x)$.

The discretization of (4.10) is computed by the second order centered finite differences method, thus

$$B \cdot F(\mathbf{u}) - \mu A \mathbf{u} = 0$$

where A is the stiffness matrix, B represents the matrix for first derivatives, $\mathbf{u}$ is the vector of discretized solution and $F(\mathbf{u})$ is the second term in (4.10) point wise evaluated. Since it is a non-linear system of equations the standard Newton's methods is implemented, where its tolerance is set as $tol = 10^{-8}$.

Then NISP method is applied to compute polynomial chaos expansion for an increasing set of degrees:

$$N = \{1, \ldots, 10\}$$

for each of them the mean and the variance of the PCE is computed. Moreover it is compared with values achieved using analytical solution. Thus for each $x_j \in D_M$

$$\bar{u}_j = \mathbb{E}\left[u(x_j, \mu(\cdot))\right] = \int_{\mathbb{R}} u(x_j, t) f_\mu(t) dt \tag{4.17}$$

$$v_j = \mathsf{Var}\big[u(x_j, \mu(\cdot))\big] = \int_{\mathbb{R}} \big(u(x_j, t) - \bar{u}\big)^2 f_\mu(t) dt \tag{4.18}$$

where f_μ is the cumulative density function of the random input. They are computed by means of Scilab routine `inttrap`. Moreover they are the components of two vectors $\bar{u}$ and v that will be compared with the vectors $\bar{u}^{(N)}$ and $v^{(N)}$, that represents the mean and variance of each polynomial chaos expansion computed.

The error of mean and variance of a truncated PCE of degree N is identified respectively by

$$\epsilon_{\text{Mean}}^{(N)} = \left\| \bar{u}^{(N)} - \bar{u} \right\|_{\infty}$$
$$\epsilon_{\text{Var}}^{(N)} = \left\| v^{(N)} - v \right\|_{\infty}$$

Notice that for all $j = 1, \dots, M$

$$\left| \mathbb{E}\left[u_j^{(N)} \right] - \bar{u} \right| = 0$$

Indeed for every fixed index j, the coefficient $c_0(x_j)$ in PCE is defined as

$$c_0(x_j) = \int_{\mathbb{R}} u\big(x_j, T(s)\big) f_\xi(s) ds =: \mathbb{E}\left[u_j^{(N)} \right]$$

The basic random variable has already transformed into the input one. This is made by means of inverse transform method, thus $T(s) = F_\mu^{-1}(F_\xi(s))$, where F_μ^{-1} and $F_\xi(s)$ are the cumulative density functions of μ and ξ respectively.

Then

$$\mathbb{E}\left[u_j^{(N)} \right] - \bar{u} = \int_{\mathbb{R}} u(x_j, t) f_\mu(t) dt - \int_{\mathbb{R}} u(x_j, T(s)) f_\xi(s) ds \tag{4.19}$$

Thus by setting

$$t = F_\mu^{-1}(F_\xi(s))$$

the integral in (4.19) becomes

$$\int_{\mathbb{R}} \big(u(x_j, t) f_\mu(t) - u(x_j, t) f_\mu(t) \big) dt = 0 \tag{4.20}$$

therefore the mean error is always null, and it is meaningless to compute it.

For each case considered let us select a subset of discretization points $\{x_i\}_{i \in J} \subset D_M$, where $J \subset \{1 \dots, M\}$ of size $d = 6$. For such points the Box and Whiskers plot is computed (see [20]), that gathers together many informations of the output random variable, such as InterQuartile Range, 25-th and 75-th quartiles, the mean and the median.

The Box and Whiskers plot requires a sampling of the quantity of interest, thus 5000 realizations of $u_j^{(N)}$ for each $j \in J$ are computed. In such computations the degree of the expansion is always set as $N = 10$.

Furthermore for these $\{x_i\}_{i \in J}$ their probability density functions are estimated by means of Kernel Density Estimation methods ([19] and [21]).

4.2.4 A null

The differential equation is integrate on $[-3, 8]$, with $u_a = 1$ and u_b set as the evaluation of the analytical solution at $b = 8$. Moreover the couple (a, u_a) allows to define free parameter c by means of (4.11).

In Table 4.1 the values of the variance error $\epsilon_{\text{Var}}^{(N)}$ are displayed for both random inputs and for the corresponding set of degrees. Moreover $\epsilon_{\text{Var}}^{(N)}$ has such stationary behavior since the Newton's method has a tolerance of $tol = 10^{-8}$.

Figure 4.8 represents the behavior of the mean and the variance of polynomial chaos expansion, whose degree is $N = 10$. For both cases (normal and lognormal inputs) the solution approaches the equilibrium ($u_{\text{eq}} = \frac{1}{2}$), see section 4.2.2. Thus for higher values of b, $u(x)$ is close to u_{eq}. Thus the advection term in (4.9) approaches zero, which implies that

$$\mu u_{xx} \approx 0$$

Degree N	Variance error normal	Variance error lognormal
1	1.1446e-07	1.4337e-06
2	1.2970e-06	1.7388e-07
3	1.3198e-06	1.8194e-07
4	1.3202e-06	1.8187e-07
5	1.3203e-06	1.8188e-07
6	1.3203e-06	1.8185e-07
7	1.3203e-06	1.8187e-07
8	1.3203e-06	1.8191e-07
9	1.3203e-06	1.8186e-07
10	1.3203e-06	1.8188e-07

Table 4.1: Values of variance errors for and $A = 0$

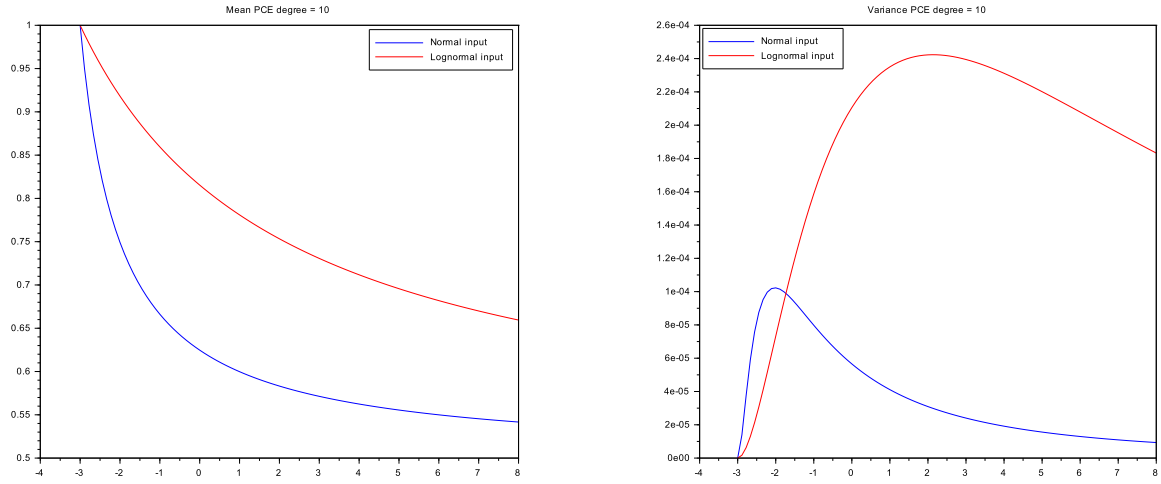

Figure 4.8: The mean (left) and variance (right) of PCE for $A = 0$ for each discretization points and for both random inputs: normal (blue) and lognormal (red)

hence the uncertainty of the input μ becomes less relevant. This motivates the decreasing values of variance in both situation. Moreover $\mathsf{Var}[u_1^{(N)}] = 0$ since u_a is a fixed values.

In Figure 4.9 the Box and Whisker plot and the estimate of the probability density function are shown for a subset of discretization points . Moreover the mean of the solution is displayed for each discretization points (magenta curve).

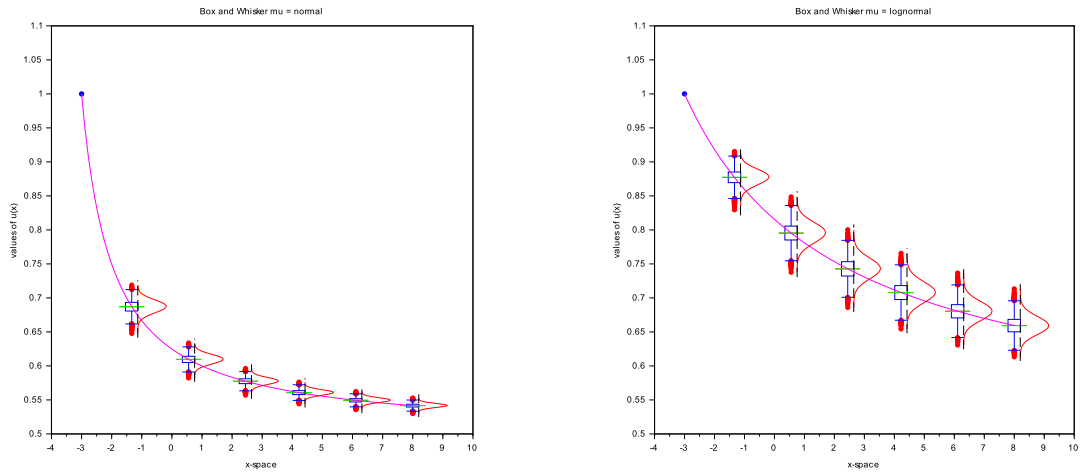

Figure 4.9: The mean (left) and the variance (right) of PCE expansion for $A = 0$

4.2.5 A negative

The domain of integration is set as $[-3, 3]$ and the free parameters in the analytical solution (see 4.13) are $A = -\frac{1}{8}$ and $c = 0$. Furthermore the boundary conditions u_a and u_b are set as the evaluation of the analytical solution at edges of the domain.

Table 4.2 gathers together the infinite norm of the variance errors $\epsilon_{\text{Var}}^{(N)}$.

Degree N	Variance error normal	Variance error lognormal
1	5.8108e-06	6.3609e-06
2	6.5412e-07	1.3968e-07
3	5.7909e-07	8.7907e-08
4	5.7916e-07	9.2244e-08
5	5.7926e-07	9.2145e-08
6	5.7924e-07	9.2145e-08
7	5.7921e-07	9.2145e-08
8	5.7925e-07	9.2145e-08
9	5.7925e-07	9.2145e-08
10	5.7924e-07	9.2145e-08

Table 4.2: Values or variance errors for both random inputs and $A < 0$

The errors do not converge since the tolerance for Newton's method is set to 10^{-8}.

In Figure 4.10 the mean and the variance of the PCE are shown. The PCE considered has degree $N = 10$.

When the viscosity is distributed as a normal random variable, it yields to a solution that is closed to equilibrium at edges of the domain. Thus by (4.15) $u_x = q \approx 0$ and hence the differential equation becomes

$$\mu u_{xx} \approx 0$$

proving that the viscosity does not influence the solution.

For μ that follows the lognormal distribution, the values $u_a = u(a)$ and $u_b = u(b)$ are far from the equilibrium, thus the variance has more influence, as Figure 4.10 displays.

Nevertheless in both cases the variance is null when the solution is evaluate at $x = 0$. Indeed $u(x) \approx \frac{1}{2}$, thus the ODE becomes $\mu u_{xx} \approx 0$ making the variance of the viscosity irrelevant.

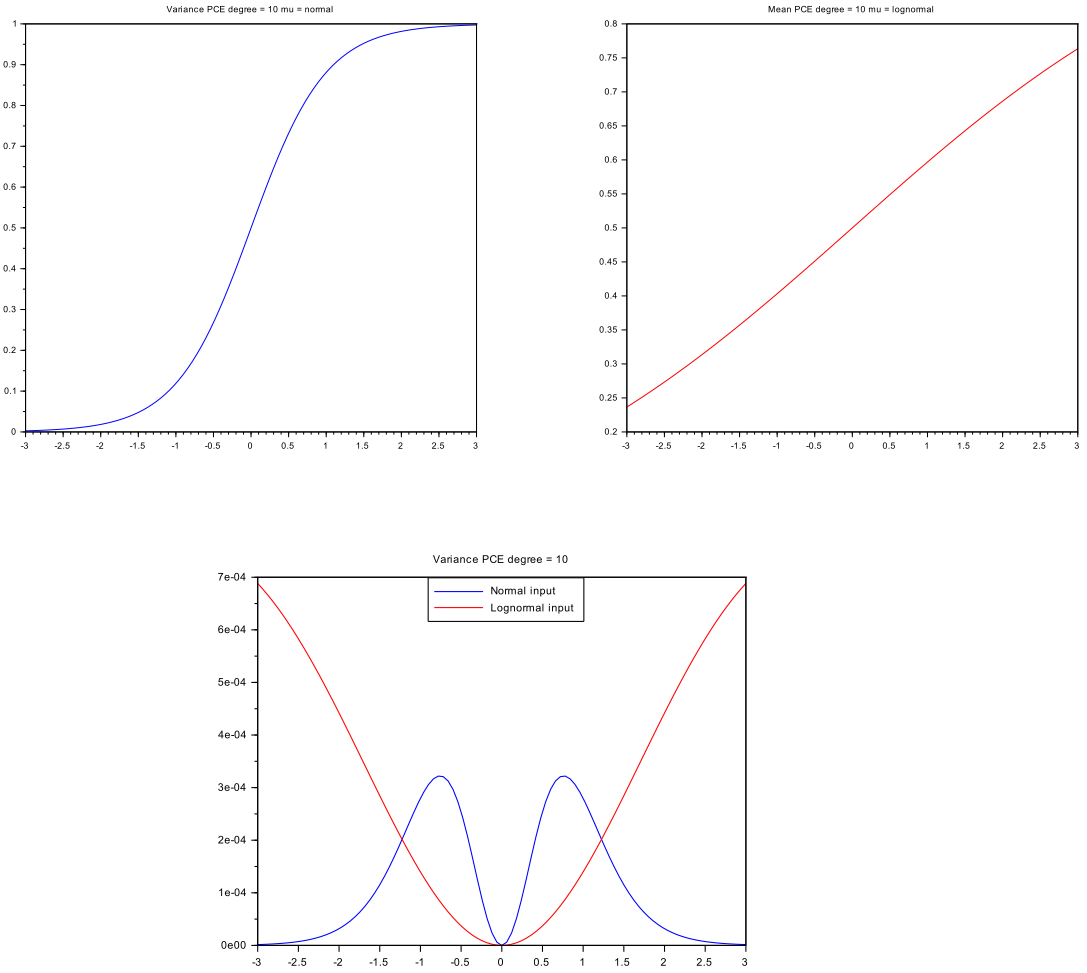

Figure 4.10: The mean and variance of PCE expansion for $A < 0$ for normal random variable (blue) and lognormal one (red)

In Figure 4.11 the Box and Whisker plot and estimate of the probability density function are shown for a subset of discretization point. Moreover the mean of the solution is shown for each discretization points (magenta curve).

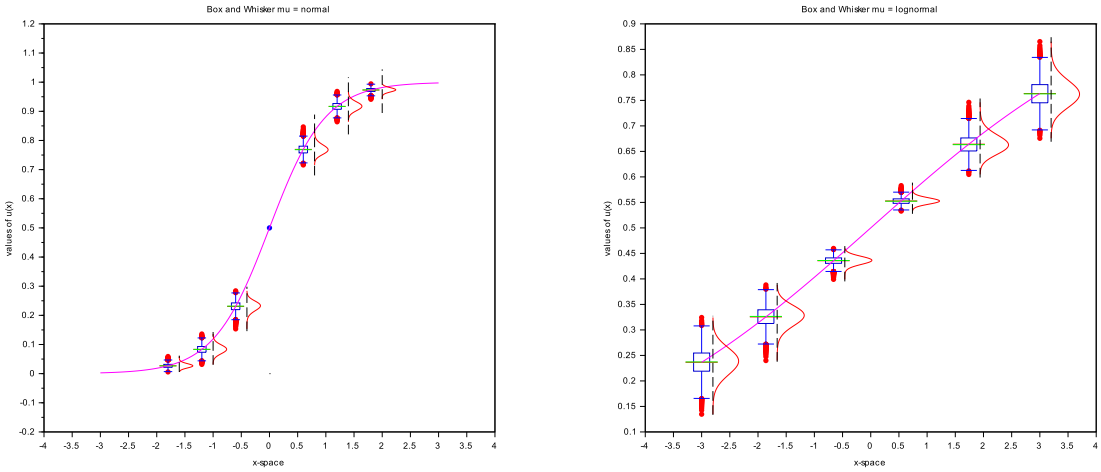

Figure 4.11: Box and Whisker for normal (left) and lognormal (right) for $A < 0$

4.2.6 A positive

The analytical solution in such case is (4.14), where the free parameters are set as $A = \frac{1}{8}$ and $c = 0$. In order to avoid blow up of the solution the domain of integration $[a, b]$ is designed as

$$a = -\frac{1}{10}\frac{\pi}{\sqrt{2A}} - c + k \qquad b = \frac{1}{10}\frac{\pi}{\sqrt{2A}} - c - k$$

where the constant is $k = \frac{1}{10}$. The boundary condition u_a and u_b are the evaluation of the analytical solution (4.14) at edges of the domain.

In Table 4.3 the error of the variance $\epsilon_{\text{Var}}^{(N)}$ are shown for both random inputs. As for the

Degree N	Variance error normal	Variance error lognormal
1	6.4410e-05	8.5067e-07
2	5.3142e-06	2.0574e-08
3	1.2240e-06	1.3083e-08
4	8.7715e-07	1.3011e-08
5	8.4110e-07	1.3010e-08
6	8.3658e-07	1.3010e-08
7	8.3591e-07	1.3010e-08
8	8.3580e-07	1.3010e-08
9	8.3577e-07	1.3010e-08
10	8.3577e-07	1.3010e-08

Table 4.3: Values or variance errors for and $A > 0$

previous cases the error is never less than the tolerance of Newton's method.

The mean of each $u_j^{(N)}$ for $j = 1, \ldots, M$ is shown in Figure 4.12. For simplicity only truncated expansions of degree $N = 10$ are considered. For the same degree the variance is plotted in Figure 4.13 for μ that follows both the Gaussian (blue) and lognormal (red) distribution.

In the two cases the variance is zero at $x = 0$, since the values of the solution $u(x)$ approaches $\frac{1}{2}$, that makes null the advection term in the differential equation. Therefore in a neighborhood of $x = 0$ the equation solved is

$$\mu u_{xx} \approx 0$$

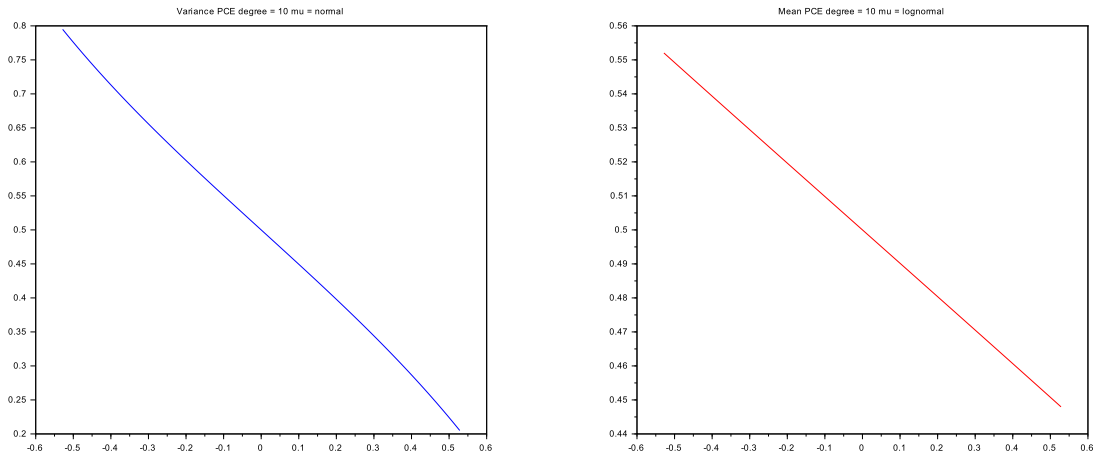

Figure 4.12: The mean f the polynomial chaos expansions for normal (left) and lognormal (right) random inputs $A > 0$

yielding to null impact of the viscosity's variance. The variance is higher at $x = a$ and $x = b$ indeed by their very definition thy are largely influenced by the viscosity.

Figure 4.14 represents the Box and Whisker plot and estimated probability density function for a subset of discretization point. Moreover the mean of the solution is shown for each discretization points (magenta curve).

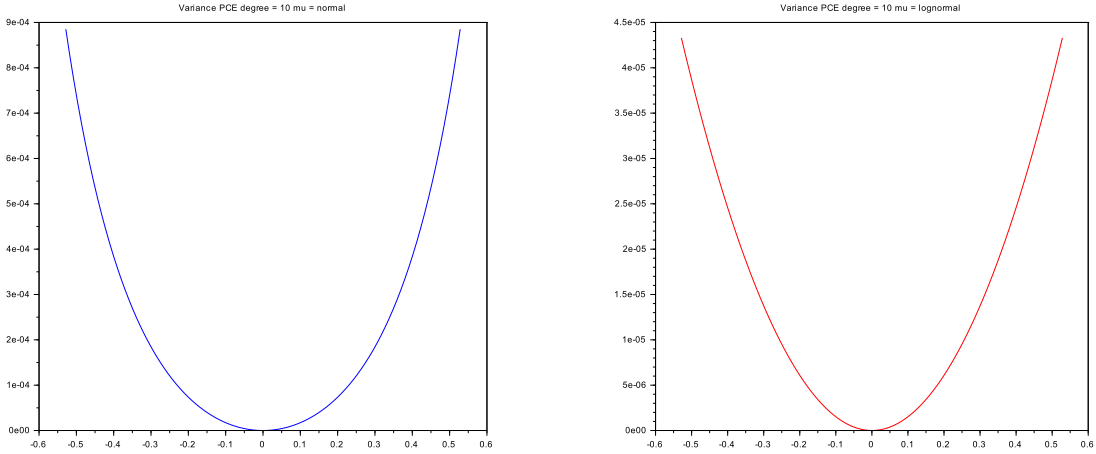

Figure 4.13: The mean of polynomial chaos expansions for normal (left) and lognormal (right) random inputs $A > 0$

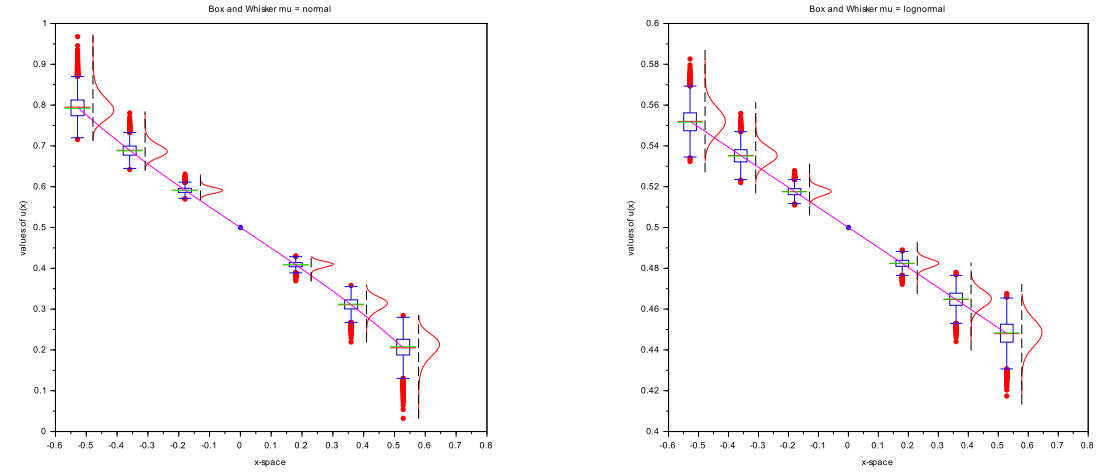

Figure 4.14: The Bow and Whiskers plot of PCE expansion for $A > 0$ normal (left) and lognormal (right) input

4.3 Lid-Driven Cavity

In this section the velocity field for a two-dimensional steady flow in a square domain with sides and bottom at rest is computed, while the lid is moving at uniform velocity U, this is sometimes referred as the *lid-driven cavity* problem.

The incompressible fluid is characterized by Reynolds number (Re) set to 100, moreover the domain is a square box $D = [0,1] \times [0,1]$. The governing equations are the incompressible and steady Navier-Stokes equations, whose non-dimensional formulation is

$$\begin{aligned} u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + \frac{\partial p}{\partial x} &= 0 \\ u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} - \frac{1}{\mathrm{Re}}\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + \frac{\partial p}{\partial y} &= 0 \\ \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0 \end{aligned}$$

where $u(x,y)$ and $v(x,y)$ are the two components of the vector field $\mathbf{u} : \mathbb{R}^2 \to \mathbb{R}^2$, and the scalar function $p(x,y)$ represents the pressure. These equations can be recast into the vectorial form

$$(\mathbf{u} \cdot \nabla)\mathbf{u} - \frac{1}{\mathrm{Re}}\Delta\mathbf{u} + \nabla p = 0 \tag{4.21}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{4.22}$$

Due to non-dimensionality of the variables, the velocity field is normalized with respect to the lid velocity U. Therefore for such equations the boundary conditions are

$$\mathbf{u} = \begin{cases} (1,0) & \text{on the lid} \\ 0 & \text{on the sides and on the bottom} \end{cases}$$

The usual weak formulation of the Navier-Stokes equations is find a $(\mathbf{u}, p) \in \mathbf{V} \times Q$ such that

$$\int_D (\mathbf{u} \cdot \nabla)\mathbf{u}\ \boldsymbol{v} + \frac{1}{\mathrm{Re}}\nabla\mathbf{u} \cdot \nabla\boldsymbol{v} + p\nabla \cdot \boldsymbol{v} = 0 \tag{4.23}$$

$$\int_D \nabla \cdot \mathbf{u}\ q = 0 \tag{4.24}$$

for every test function $\boldsymbol{v} = (v_1, v_2) \in \mathbf{V}$ and $q \in Q$. For the definition of these spaces see [22].

As suggest in [24] this weak formulation can be recast as

$$F(\mathbf{u}, p) = \int_D (\mathbf{u} \cdot \nabla)\mathbf{u}\ \boldsymbol{v} + \frac{1}{\mathrm{Re}}\nabla\mathbf{u} \cdot \nabla\boldsymbol{v} + p\nabla \cdot \boldsymbol{v} - \nabla \cdot \mathbf{u}q = 0 \tag{4.25}$$

The weak solution is the zero of the above non-linear function $F(\mathbf{u}, p)$. Hence let us compute its Jacobian applied to a vector $(\delta\mathbf{u}, \delta p) \in \mathbf{V} \times Q$

$$\int_D (\delta\mathbf{u} \cdot \nabla)\mathbf{u}\ \boldsymbol{v} + (\mathbf{u} \cdot \nabla)\delta\mathbf{u}\ \boldsymbol{v} + \frac{1}{\mathrm{Re}}\nabla\delta\mathbf{u} \cdot \nabla\boldsymbol{v} + \delta p\nabla \cdot \boldsymbol{v} - \nabla \cdot \delta\mathbf{u}\ q \tag{4.26}$$

4.3.1 Freefem++ solver

The solution is computed using Finite Element Method (FEM), thus the domain is triangulated using 20 points on each side of the domain.

As described in [23] Navier-Stokes equations require suitable definition of element spaces in order to ensure the existence of the discretized solution. In this example Taylor-Hood elements are used, where two different basis functions are defined for the unknowns: quadratic piecewise polynomials for the velocity components and linear piecewise polynomials for the pressure. The underlying requirement is the satisfiability of the Babuska-Brezzi (BB) condition (see [23]) that the elements have to fulfill.

Moreover a stabilization term is add to the left hand side of (4.25)

$$S_\epsilon(p) = -\int_D \epsilon pq$$

in this example $\epsilon = 10^{-8}$ (see [23] and [24] for details). This value is taken into account in computation of the Jacobian, by adding

$$-\int_D \epsilon \delta p q$$

to (4.26).

The key section of Freefem++ code is the Newton's algorithm to compute the solution. That is

```
solve Newton([du1,du2,dp],[v1,v2,q]) =
                        int2d(Th) ( invRe*(Grad(du1,du2)'*Grad(v1,v2) )
                        + UgradV(du1,du2, u1, u2)'*[v1,v2]
                        + UgradV( u1, u2,du1,du2)'*[v1,v2]
                        - div(du1,du2)*q - div(v1,v2)*dp+
                        - 1e-8*dp*q
                        )
                        + int2d(Th) (invRe*(Grad(u1,u2)'*Grad(v1,v2) )
                        + UgradV(u1,u2, u1, u2)'*[v1,v2]
                        - div(u1,u2)*q - div(v1,v2)*p
                         -1e-8*p*q
                        )
                        + on(1,2,3,4,du1=0,du2=0) ;
u1[] += du1[]; u2[] += du2[]; p[] += dp[];
err = du1[].linfty + du2[].linfty + dp[].linfty;
```

where `UgradV` and `Grad` represent the quantities $(\mathbf{u}\cdot\nabla)\mathbf{u}\boldsymbol{v}$ and $\mathbf{u}\cdot\nabla\boldsymbol{v}$ and `invRe`$= \frac{1}{\mathrm{Re}}$, moreover `v1`, `v2` and `q` are the test functions of weak formulation. The keyword `solve` is used to compute the solution of the linear system defined.

Newton's algorithm stops when the infinity norm of the error is less than 10^{-8}, moreover the values `du1`, `du1` and `dp` are set to zero on the boundary of the domain, since the first guess of the solution satisfies the boundary condition.

In Figure 4.15 the velocity field $\mathbf{u}$, that solves (4.21) and (4.22) for $U = 1$, is shown. This chart is achieved by saving the velocity field in visualization toolkit format (`.vtk`), and then these data are post processed by the software Paraview 4.3.1, an open-source, multi-platform data analysis and visualization application.

4.3.2 PCE for lid-driven cavity

The uncertainty occurs in the lid velocity, that is perturbed by a Gaussian noise

$$U = 1 + \eta \qquad \eta \sim \mathcal{N}\left(0, \frac{1}{16}\right)$$

The attention is focused on the first component $u(x, y)$ of the velocity field $\mathbf{u}$, restricted to a vertical segment S passing through the center of the cavity D

$$S = \left\{ (x, y) \ : \ x = \frac{1}{2} \ , \ y \in [0, 1] \right\}$$

The solution u is evaluated at $S_M \subset S$, whose elements are $M = 100$ equally spaced points. For simplicity let us call these points $\mathbf{x}_j$ for $j = 1, \ldots, M$.

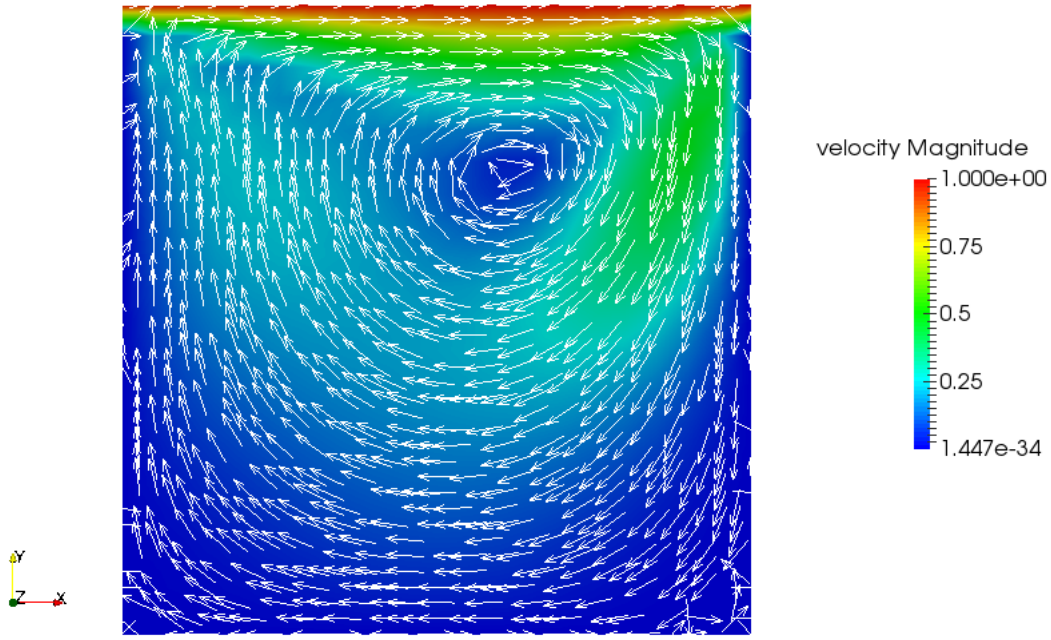

Figure 4.15: Solution of lid-driven cavity problem for $U = 1$ and Re$= 100$

Therefore for each S_M a PCE of degree $N = 5$ is computed, thus

$$u_j^{(N)} = u^{(N)}(\mathbf{x}_j, \xi) = \sum_{i=0}^{N} c_i(\mathbf{x}_j)\Psi(\xi)$$

where the basic random variable $\xi \sim \mathcal{U}(-1, 1)$, hence the gPC basis is constituted by Legendre polynomials on $[-1, 1]$.

The values of mean and of the variance of such approximation are shown in Figure (4.16). The variance is null in two points: one at the bottom of the cavity, since no-slip condition were set, and at another point within $[0.6, 0.7] \subset S$ interval. Moreover it increases up to reach the lid since U is the uncertain parameter.

For each PCE a sampling of size 5000 is computed in order to detect the 25-th (red curve) and 75-th quartile (green curve) which are displayed in Figure 4.17. Moreover in the same figure, for ten points in S_M, the Box and Whiskers plot is detected.

These computations are done in order to compare these values with the data available in Ghia, et. al. in [25], that are the blue points represented in the same figure. The magenta curve is the mean of each PCE $u_j^{(N)}$ for $j = 1, \ldots, M$.

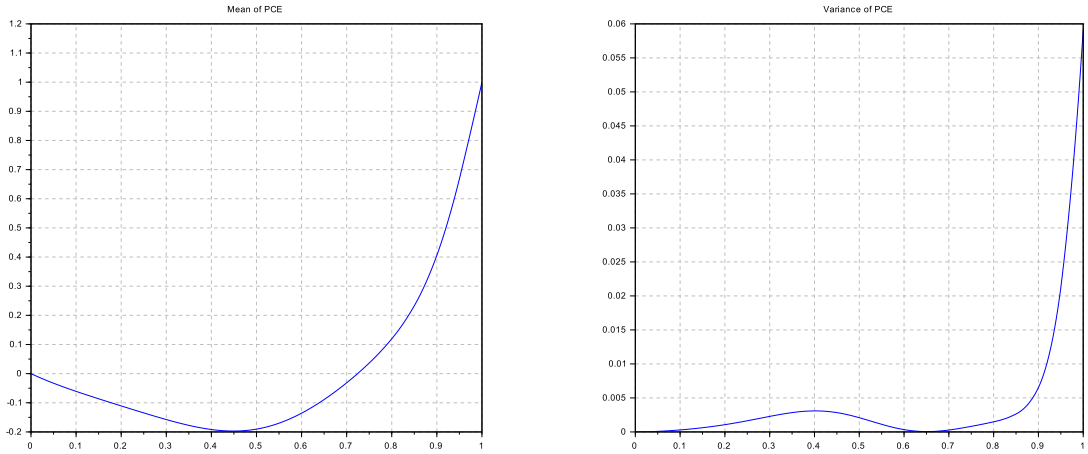

Figure 4.16: The mean and variance of PCE for lid-driven cavity with Re= 100

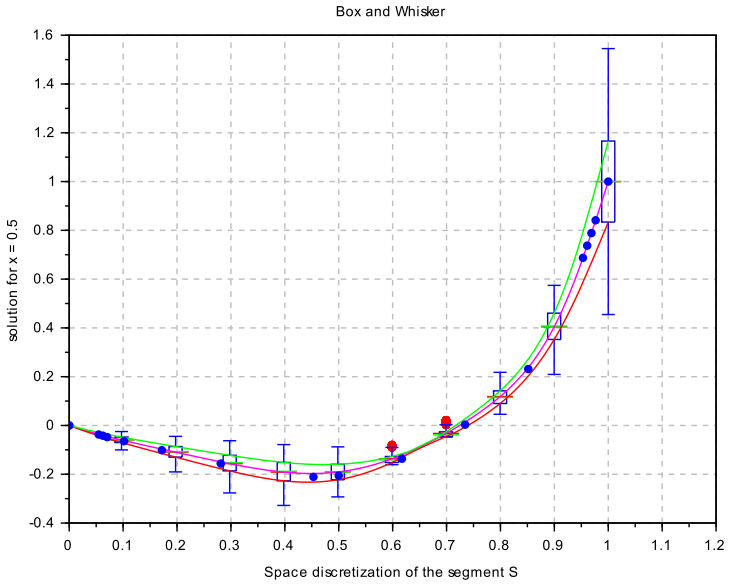

Figure 4.17: Box and Whisker plot for Lid-driven cavity with Re= 100

4.4 Pyclaw

The last example on which NISP technique is applied concerns the two dimensional transport equation

$$q_t + uq_x + vq_y = 0 \tag{4.27}$$

on the square domain $D = [0, 2] \times [0, 2]$, which is integrated, with respect to time, on $T = [0, 1]$. The function $q : T \times D \to \mathbb{R}$ is the conserved quantity, while $u \in \mathbb{R}$ and $v \in \mathbb{R}$ are the components of the constant velocity field that characterizes the advection.

The initial data is

$$q(0, x, y) = e^{-w \cdot [(x-0.3)^2 + (y-0.3)^2]}$$

where $w = 40$ and $(x, y) \in D$. To shorten the notation let us call the spatial coordinates

$\mathbf{x} = (x, y)$.

The software, used to compute the solution, is Pyclaw a python interface of Clawpack packages, that is an hyperbolic partial differential equation solver in 1D, 2D, and 3D, available for Linux distributions.

Clawpack stands for "Conservation Laws Package" and it was initially developed for linear and non-linear hyperbolic systems of conservation laws, with a focus on implementing high-resolution Godunov type methods using limiters in a general framework. These Finite Volume Methods require a *Riemann solver* to resolve the jump discontinuity at the interface between two grid cells into waves propagating into the neighboring cells. (For furhter details see LeVeque [26])

Since a Riemann solver is already set, the Pyclaw code is considered as a black-box: only marginals data of the problem can be customized, such as the domain of integration, the size of the mesh and the velocity field.

The domain D is discretized by an uniform rectangular grid of size $m_x \times m_y$, where $m_x = m_y = 50$. Thus for $k = k(i,j) \in \{1, \dots, m_x \cdot m_y\}$ for all $i = \{1, \dots, m_x\}$ and $j = \{1, \dots, m_y\}$, with a suitable sorting

$$\mathbf{x}_k = \mathbf{x}_{k(i,j)} = \left(x_i, y_j\right)$$

Moreover the time discretization is made by $m_T = 10$ equally spaced time steps $\{t_h\}_{h=1}^{m_T}$. Since the time domain is $T = [0, 1]$ the time step's length is $\delta t = \frac{1}{10}$.

The uncertainties are the two components of the velocity field, both of them is perturbed with a Gaussian noise, thus they are normally distributed random variables

$$u \sim \mathcal{N}\left(1, \frac{1}{20^2}\right) \qquad v \sim \mathcal{N}\left(\frac{1}{2}, \frac{1}{30^2}\right)$$

For each time step the PCE computed is a vectorial quantity, thus for each couple $(t_h, \mathbf{x}_k)$

$$q^{(N)}(t_h, \mathbf{x}_k, \boldsymbol{\xi}) = \sum_{\mathbf{i} \leq N} c_{\mathbf{i}}(t_h, \mathbf{x}_k) \Psi_{\mathbf{i}}(\boldsymbol{\xi})$$

The multivariate decomposition is truncated at total degree $N = 5$, and each components of the basic random vector $\boldsymbol{\xi} = (\xi_1, \xi_2)$ whose components are two $\mathcal{N}(0, 1/2)$.

In such setting, since $N = 5$, NISP library computes $(5+1)^2 = 36$ realizations of the two input random variables. These realizations are two dimensional points whose first component concerns u, while the second represents a realization of v. Then these values are plugged into the Pyclaw script, and then the simulations are run, getting the solution at each times steps. These values are imported within Scilab and used to compute the coefficients of the decomposition.

Let us point out that the data are saved by Pyclaw using Fortran ordering, for compatibility with Fortran routines, therefore, when imported in Scilab, they are set as row-wise vector to be compatible with C++ ordering that characterizes NISP library. Furthermore this issue has to be taken into account when the solution is visualized, as described in Appendix A.

In Figure 4.18 the mean of the polynomial chaos expansion at final time is displayed, moreover it is also shown the initial data.

The variance of the solution is shown in Figure 4.19, for each time step. It also points out how these values variate more along the x-direction than the on y-direction. This is perfectly coherent with higher variance that characterizes the first component u of the vector field than its second component v.

As last remark let us compare the effects of considering uncertainties into model simulation with respect to single deterministic simulation. Thus the Pyclaw code is run for the deterministic values of vector field

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$$

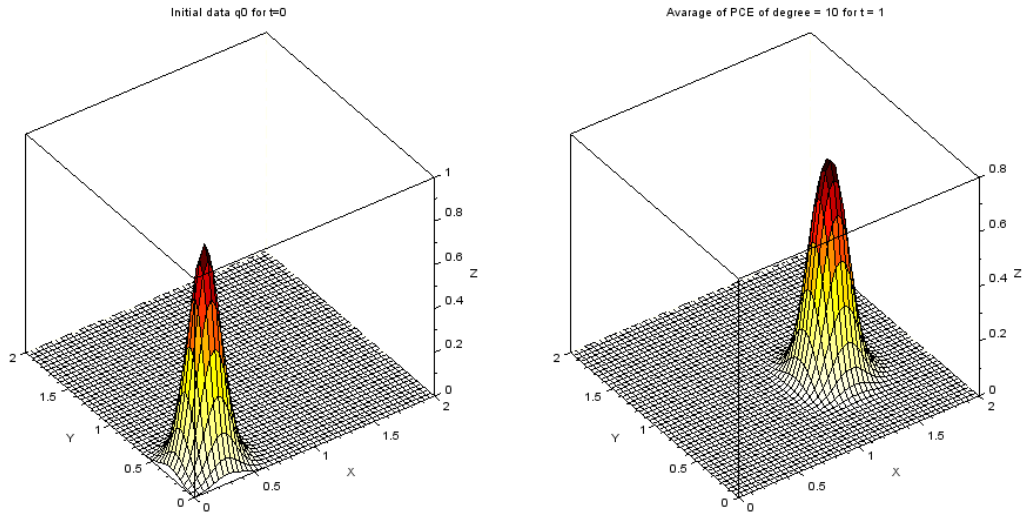

Figure 4.18: Initial data (left) and average of the polynomial chaos expansions for each point of the mesh at t=1 (right)

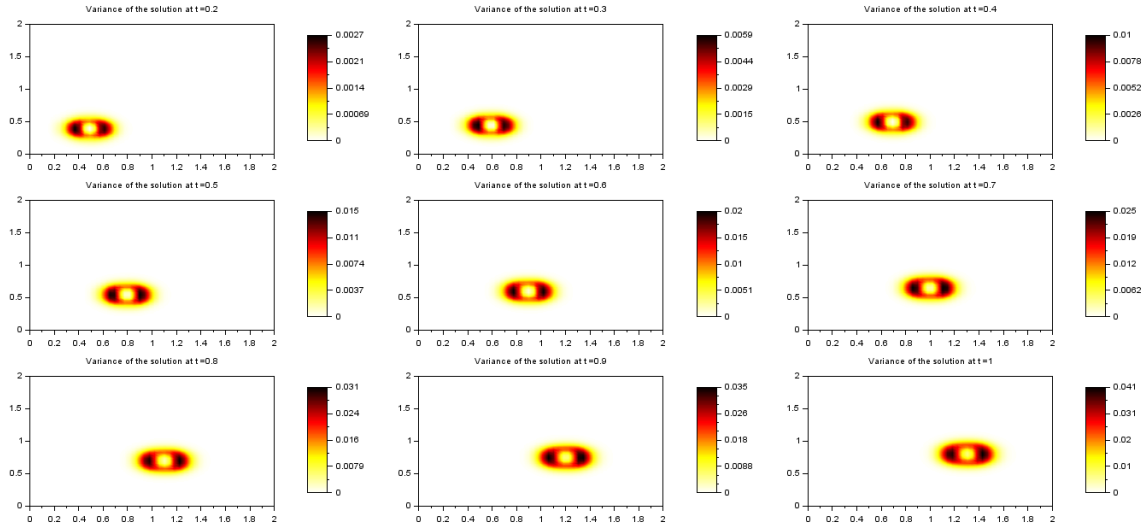

Figure 4.19: Values of the variance for all time steps

and then, for each time step t_h, the absolute value of the difference between the average of each $\{q^{(N)}(t_h, \mathbf{x}_k)\}_{k=1}^{m_x \cdot m_y}$ and the deterministic simulation. Then these values are plotted in Figure 4.20, Figure 4.21, Figure 4.22, Figure 4.23 and Figure 4.24.

These charts point out how the difference increases when the integration in time proceeds (see color bar), moreover these values highlight that the uncertainties, are not a marginal data in a model, their influence has to be taken into account for correct validation of a simulation.

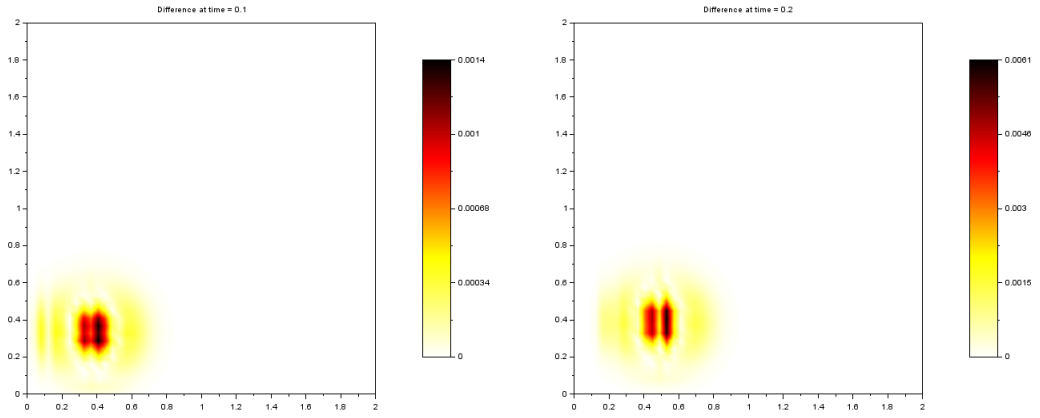

Figure 4.20: Values of the difference between deterministic and UQ simulations for the time steps $t = 0.2$ (left) and $t = 0.3$ (right)

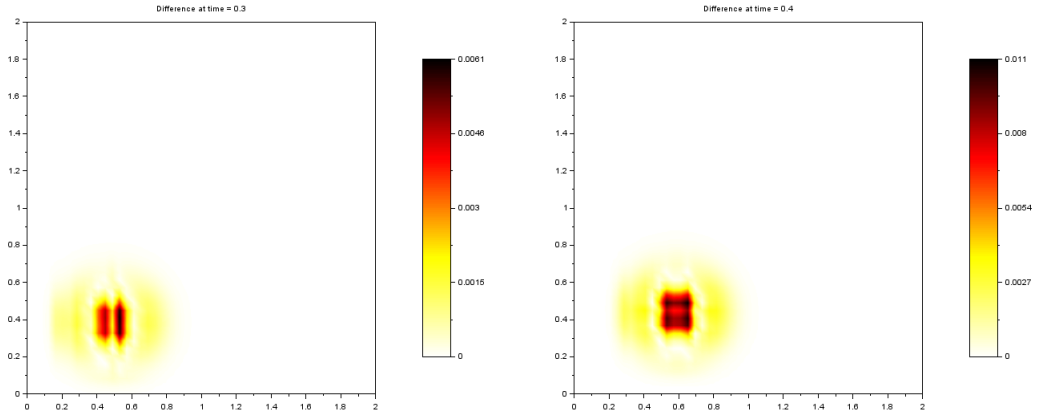

Figure 4.21: Values of the difference between deterministic and UQ simulations for the time steps $t = 0.4$ (left) and $t = 0.5$ (right)

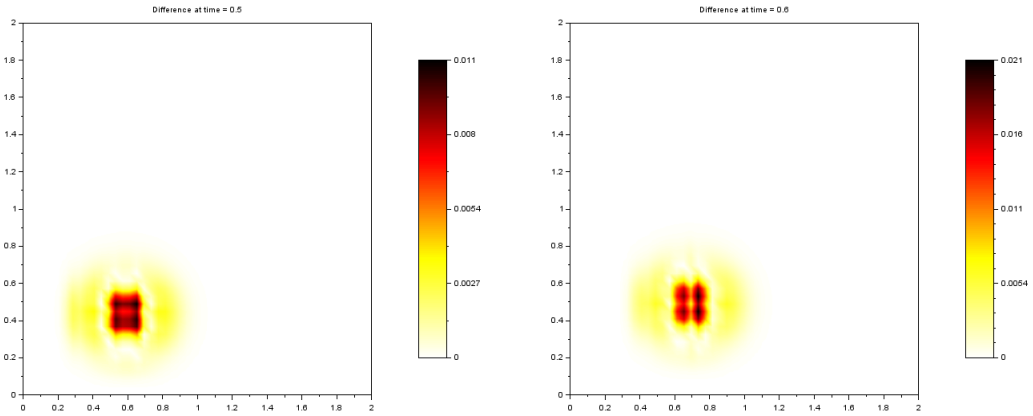

Figure 4.22: Values of the difference between deterministic and UQ simulations for the time steps $t = 0.6$ (left) and $t = 0.7$ (right)

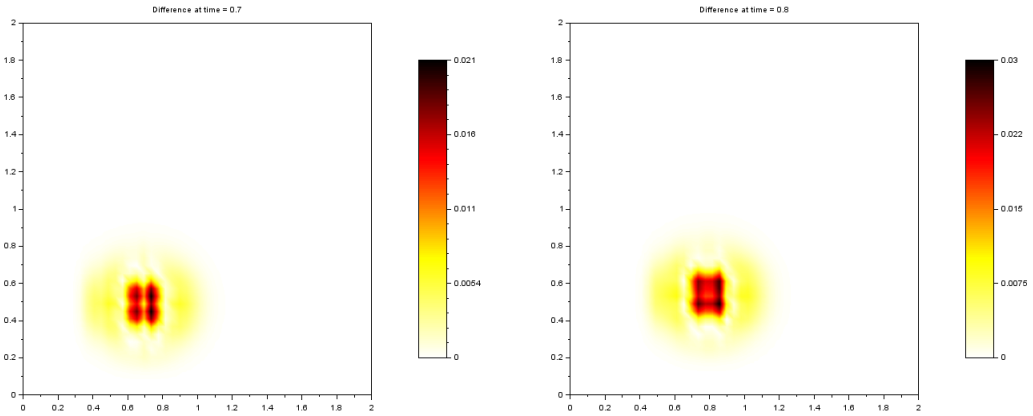

Figure 4.23: Values of the difference between deterministic and UQ simulations for the time steps $t = 0.8$ (left) and $t = 0.9$ (right)

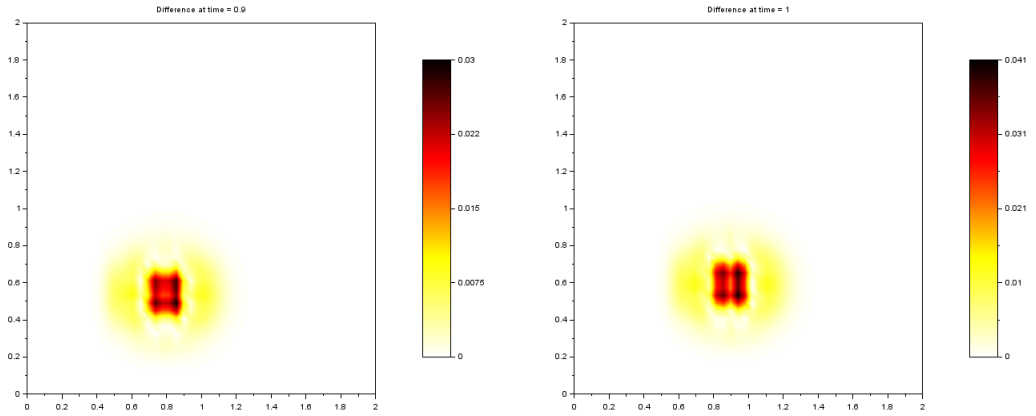

Figure 4.24: Values of the difference between deterministic and UQ simulations for the time steps $t = 0.9$ (left) and $t = 1$ (right)

4.5 Conclusions

These examples show that NISP technique is useful either to approximate with low number of polynomials random variables or to deal with the solution of a process in presence on uncertainties. Moreover the flexibility of NISP approach is proven since it is applied to several type of data sources, which are generated by completely tailored solvers, partially customizable ones and black-box codes.

As last remark let us point out how NISP is a very efficient tool for analyzing the aleatory solution of a process in presence of uncertainties even if the underlying model is deterministic. This is its most advantage, since it splits the approach to a given physical phenomena into two distinct steps: first one can focus its attention to detect the deterministic model of the physical phenomena of interest, then the uncertainties can be taken into account with NISP, exploiting the already detected model. Moreover NISP allows to consider by another point of view all the existing literature of deterministic model: they are still useful to extend the discussion in presence of uncertainties.

Appendix A

General topics: random variables, histograms and softwares

A.1 Equally distributed random variables

In this section it is proven, via a practical example, that there exist two equally distributed random variables X and Y, defined on the same probability space $(\Omega, \Sigma, \mathsf{P})$, which are not the same measurable function.

Let X be a standard normal random variable, whose probability density function is

$$f_X(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

Let us define a new random variable $Y = -X$. They are not the same Σ-measurable functions but actually $X \sim Y$. Indeed by symmetry of probability density function

$$\mathsf{P}(Y \leq t) = \mathsf{P}(-X \leq t) = \mathsf{P}(X \geq t) = \int_t^{+\infty} f_X(x)dx = \int_{-\infty}^{t} f_X(x)dx = \mathsf{P}(X \leq t)$$

A.2 Histogram of multi-element PCE

Let us discuss how to handle a sampling of a multi-element PCE of a random variable Y, in order to build a cumulative histogram, that allows to compare this decomposition with the probability density function of Y.

For simplicity let us suppose to be in aN univariate setting, and let us consider a partition $\mathcal{A} = \{A_k\}_{k=1}^{N_e}$ of the support D of the probability density function of Y. Without loss of generality they are set

$$A_k = (a^{(k-1)}, a^{(k)}]$$

for each $k \in \{1, 2, \ldots, N_e\}$.

Let S_k be the sampling of the truncated PCE of Y_k random variables. Let us call $\{C_j\}_{j \in J}$ the collection of bins of the histogram, for $J = \{1, 2, \ldots, M\}$. They are intevals of kind

$$C_j = (x_j, x_{j+1}] \qquad j = 1, \ldots, M$$

Moreover let $h_j^{(k)}$ be the number of data of S_k that lies in C_j. Two issues have to be discussed.

- Bins and partition of support.
  Two different partitions are defined on D: $\mathcal{A}$ and the collection of bins $\{C_j\}_{j \in H}$. In general

setting they do not coincide, therefore for each $k \in \{2 \dots, N_e - 1\}$ there exist an index j such that

$$a^{(k-1)} \in C_j$$

The values $a^{(0)}$ and $a^{(N_e)}$ are not considered since they conicide with the edges of the support D.

- Normalization.
  Usually the area of histogram $h(x)$ is defined as

$$\int_{x(1)}^{x(M+1)} h(x) = \sum_{j=1}^{M} h_j (x_{j+1} - x_j)$$

where $x(M+1)$ is the supremum edge of the M-th bin, while $x(1)$ is the infimum edge of the first bin. Moreover h_j represents the number of values within the j-th bin.

In order to compare a sampling with probability density function, the area of the histogram has to be normalized to one, thus for $j = 1, \dots, M$

$$h_j = \frac{h_j}{N}$$

where N is the size of the sampling considered.

For simplicity let us consider a fixed index $j \in J$, thus the attention is focused on a particular bin C_j.

In multi element PCE setting the normalization involves the weight associated to each element of the partition $\mathcal{A}$. This connection is expressed by equations (2.6) and (2.7) which suggest that the area of each sampling S_k has to be set equal to $p_k = \mathsf{P}(Y \in A_k)$. Therefore let us define

$$\overline{h}_j^{(k)} = \frac{h_j^{(k)}}{N} p_k$$

where N is the sum of the sizes of the samplings involved. Then, for each index $k \in \{1, 2, \dots, N_e\}$, let $\overline{I}_j \subset J$ be a set of indexes such that

$$A_k \cap C_j \neq \varnothing$$

Then the number of realizations in C_j is

$$h_j = \sum_{i \in \overline{I}_j} \overline{h}_j^{(i)}$$

Thus the histogram build with $\{C_j\}_{j \in J}$ bins and these values $\{h_j\}_{j \in J}$ has the area normalized to one.

A.3 Scilab's shell (sh) command execution

For the lid driven cavity problem and advection equation the solution is computed by an external software (Frefem++ and Pyclaw). NISP technique require the solution for a selected collection of realizations of the input random variables. Thus Scilab's shell command execution is needed to run the external scripts (The extension are `.edp` and `.py` respectively).

For simplicity let us consider the lid driven cavity example, where a univariate decomposition occurs. First is defined a model (`driven_cavity2D_model.edp`), in which the numerical quantities, that one wishes to customize during the execution, are replaced with special strings, defined by the user, such as `$$variable$$`.

Then the usual NISP code is implemented, where in place of evaluation of the model, a loop in each realization is made, such as

```
for i=1:size(inputdata,1)
     run_file(inputdata(j),lpath,j,M)
     save_data_path = fullfile(lpath,'run','data'+string(j)+'.txt');
     outputdata(:,j) = fscanfMat(save_data_path);
end
```

The three code lines create, by substituting the user's identifiers with numerical values, an executable `.edp` file and execute the scripts. Than the data, located in `save_data_path`, are import within Scilab due to `fscanfMat`. The Scilab routine `run_file` is shown below in detail. Moreover it can be divided in two section.

The former is about the substitution of all user's identifiers (such as `$$variable$$`) with numerical values. Namely

```
function run_file(U,lpath,counter,M)
//
// INPUT
//
// U = realizations of the inputs random variable
// lpath = directory in which data are saved and routines lie
// counter = counter to save the data of the i-th realization
// M = discretization points od the
//
// output (void) = cration of executible file and that is run

modelfile = fullfile(lpath,"driven_cavity2D_model.edp");

// reading the model
fd = mopen(modelfile,'r');
txt = mgetl(fd);
mclose(fd);

// Substitution of data for itxt=1:size(txt,1)
txtline = txt(itxt);
ind = strindex(txtline,"$$");
if isempty(ind) then
continue;
end
ind = strindex(txtline,"$$MUVAL$$");
if  isempty(ind) then
txtline = strsubst(txtline,"$$MUVAL$$",string(U));
end
ind = strindex(txtline,"$$COUNTER$$");
if  isempty(ind) then
Ustring = string(U);
txtline = strsubst(txtline,"$$COUNTER$$",string(counter));
end
ind = strindex(txtline,"$$NofPoints$$");
if  isempty(ind) then
Ustring = string(U);
txtline = strsubst(txtline,"$$NofPoints$$",string(M));
end

txt(itxt) = txtline;
end

// create the modified executible file
destfile = fullfile(lpath,"run","driven_cavity2D.edp");
```

```
fd = mopen(destfile,'wt');
mputl(txt,fd);
mclose(fd);
```

In this specific section, a realization of the random variable is substituted to `$$MURAL$$` special string. The others are used to customize the number of points on which the solution is evaluated (`$$NofPoints$$`) and to set a counter (`$$COUNTER$$`) to save in a text file all data computed.

The second part of the code is about the execution of the file

```
// Bat file
exestring = """C:\ Program Files (x86)\ FreeFem++\ FreeFem++.exe""...
-nowait -nw -ne -cd -f driven_cavity2D.edp";
batfile = fullfile(lpath,"run","test.bat");
fd = mopen(batfile,'wt');
mputl(exestring,fd);
mclose(fd);

// Run
exestring = "cmd.exe /C test.bat";
exestr = "pushd """ + fullfile(lpath,"run") + """&& " + exestring;
unix_w(exestr);
endfunction
```

Since the operative system is windows a `.bat` file is created specifying the location of the program `FreeFem++.exe` and the file that one wishes to run. Then is generated the executable string for windows prompts `exestr` where are set the correct directory, and the `.bat` file. Then using the Scilab's shell (sh) command `unix_w` the code is run, moreover the standard `cmd` output is redirected to Scilab window.

The situation is similar to execute a Pyclaw script. The file `run_data.sci` follows the same idea. The only changes are about the special characters used and about the shell command string. Indeed the operative system is Linux, thus to execute the Pyclaw file is enough to set

```
unix_w('python '+destfile);
```

in place of creation of bat file. Notice that `destfile` is the name `.py` document that one wishes to run.

A.4 Installation of Pycalw

Pyclaw is the python distribution based on Clawpack, a dedicated software for computing the solution of 1D, 2D, 3D hyperbolic differential equations, when a Riemann Solver is detected. See LeVeque [26] for details.

The installation were made on Ubuntu 14.04.02 LTS. Moreover the shell commands were

```
sudo apt-get install python-matplotlib
sudo apt-get install python-scipy
sudo apt-get install python-pip
sudo apt-get install gfortran
sudo pip install clawpak
```

The first two are libraries for visualization toolkit of Pyclaw, the Fortran compiler is needed since Clawpack uses Fortran codes and `pip` is a package management system used to install and manage software packages written in Python.

A.5 On vector ordering: Fortran, C++ and Scilab

This discussion is a consequence of the interaction among Pyclaw, C++ NISP and Scilab. When the solution is computed via Pyclaw solver, it is saved as (using meta-language)

```
for j=1:my
      for i=1:mx
            print q(i,j)
      end
end
```

where m_x are the number of discretization points along x-direction and m_y are the number of point in y-direction. It turns out that the solution is saved as a column array, for each time step. Thus, in order to be compatible with C++ routines of NISP library, is set as a row vector. This ordering is kept during computation of PCE for each discretization node.

For simplicity let consider the example of the advection equation. If for instance, for a chosen time step, one is interested in visualizing the behavior of the average of the vectorial PCE, the row vector has to be recast into a $m_x \times m_y$ matrix. First it is set again as column and then the Scilab routine

```
Av = matrix(average,mx,my)
```

is used which stacks column-wise the vector `average` into `Av` matrix.

The subtleties arise when `Av` interacts with the two visualization routines `surf` and `Sgrayplot` of Scilab. Indeed the first requires `X`, `Y` and `Z`, where $[X, Y]$ is the output for `meshgrid` routine. Moreover for each suitable indexes i and j

$$Z\big(X(i,j), Y(i,j)\big) = Z(x(j), y(i))$$

therefore `Av` has to be transposed in order to be compatible the previous relation, indeed the column of `Av` are evaluation of the solution for constant y-component. Thus the correct code line to visualize `Av` via `surf` routine is

```
surf(X,Y,Av')
```

Different situation happens for `Sgrayplot` which computes the smooth 2D plot of a surface using colors. It requires `x`, `y` which are the two row vectors that describe the discretization point on each side of the domain, and `Z`, that is

$$Z(i,j) = Z\big(x(i), y(j)\big)$$

therefore, despite in previous case, the routine `matrix` return compatible values for `Sgrayplot` visualization scripts.

Bibliography

[1] S. Baldo, *Lecture notes of functional analysis— Part 1*, a.a 2012/2013

[2] G. Andreaus, R. Askey, R.Ray, *Special Function*, Cambridge university press, 1999

[3] Walter Gautschi, *Orthogonal polynomials Computations and Approximations*, Oxford university press, 2004

[4] A. P. Sahanggamu, *Generating Function and their applications*, MIT 2006. URL http://ocw.mit.edu/courses/mathematics/18-104-seminar-in-analysis-applications-to-number-theory-fall-2006/projects/peter_s.pdf.

[5] T. Tang, *The Hermite spectral method for Gaussian-type functions*, SIAM J. Sci. Comput. 14(3) (1993), 594–606

[6] Spiegel M.R, *Fourier Analysis*, Schaum's outline series, McGraw-Hill

[7] Joseph Bak, Donald J. Newman, *Complex Analysis*, Springel, Third Edition 2010

[8] E.C. Titchmarsh, *The Theory of Functions*, 2nd edition, Oxford University Press, London, 1939.

[9] O.Kallenberg, *Foundation of Moderns Probability*, Springer, 1997

[10] Paolo Baldi, *Calcolo delle probabilità*, McGraw-Hill 2007, Milano

[11] M. Reed and B. Simon, *Methods of modern mathematical physics. 1. Functional analysis*, Academic press, New York, 1972

[12] O.P. Le Maître, O.M. Knio, *Spectral Methods for Uncertainty Quantification* Springer, 2010

[13] Bernt Oksendal *Stochastic differential equations* Springer, 2003

[14] O. G. Ernst, A. Mugler, H.-J. Starklff, E. Ullmann, *On the Convergence of Generalized Polynomial Chaos Expansions*, ESAIM: Mathematical Modelling and Numerical Analysis , 46(2), 317-339. 2011

[15] Dongbin Xiu, *Numerical Methods for Stochastic Computations* Priceton University Press, 2010

[16] R. Ghanem and P. Spanos, *Stochastic Finite Elements: a Spectral Approach*, Springer-Verlag, 1991

[17] X. Wan, G.E. Karniadakis, *Multi-element generalized polynomial chaos for arbitrary probability measures*, SIAM J. Sci. Comput.Vol. 28, No. 3, pp. 901–928 , 2006

[18] R Cannarsa, T.D'Aprile, *Lectures note on Measure Theory and Functional Analysis* Dipartimento di matematica, Università di Roma "Tor Vergata", a.a. 2006/07

[19] Motoi J. Namihira, *Probabilistic uncertainty analysis and its applications in option models*, The Florida State University Collage of Arts and Science, 2013

[20] G.Borzi, A.Bassi, *Data mining tutorial.* http://www.openeering.com/sites/default/files/Data_mining_0.pdf.

[21] B.W. Silverman, *Density Estimation for Statistics and Data Analysis.* Chapman & Hall/CRC, 1998.

[22] Alfio Quarteroni, *Modellistica Numerica per Problemi Differenziali*, Springer, 2012

[23] H.P.Langtange, R.Winther, *Numerical Methods for Incompressible Viscous Flow*, Advances in Water Resources. 25(8), Elsevier, 2002

[24] Hecht, F. *FreeFem++ Manual.* http://www.freefem.org/ff++/ftp/freefem++doc.pdf. 2012

[25] Ghia, U., Ghia, K. N. and C. T. Shin, *High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid Method.* Journal of Computational Physics. 48, 387-411. 1982

[26] R.J. LeVeque, *Numerical Methods for Conservation Laws.* Springer, 1992