National Technical University of Athens
School of Mechanical Engineering
Fluid Mechanics Department
Lab of Thermal Turbomachines
Parallel CFD & Optimization Unit

A Painless Intrusive Polynomial Chaos Expansion Approach to the CFD Analysis and the Adjoint-based Optimization

Diploma Thesis of

Michail Chatzimanolakis

Supervisor : K.C. Giannakoglou, Professor

Athens, 2018

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Μηχανολογων Μηχανικων
Τομεας Ρευστων
Εργαστηριο Θερμικων Στροβιλομηχανων
Παραλληλης ΥΡΔ & Βελτιστοποιησης

Επεμβατικό Ανάπτυγμα Πολυωνυμικού Χάους στην Υπολογιστική Ρευστοδυναμική και στη Βελτιστοποίηση Μορφής με τη Συζυγή Μέθοδο

Διπλωματική Εργασία

Μιχαήλ Χατζημανωλάκης

Επιβλέπων : Κ.Γ. Γιαννάκογλου, Καθηγητής

Περίληψη

Η διπλωματική αυτή εργασία προτείνει μία μέθοδο Ποσοτικοποίησης Αβεβαιότητας (Uncertainty Quantification – UQ) προς χρήση στην αεροδυναμική ανάλυση και βελτιστοποίηση υπό αβεβαιότητες, η οποία βασίζεται στη θεωρία του Αναπτύγματος Πολυωνυμικού Χάους (Polynomial Chaos Expansion – PCE), συγκεκριμένα στην επεμβατική (intrusive) εκδοχή της. Το Επεμβατικό Ανάπτυγμα Πολυωνυμικού Χάους (Intrusive Polynomial Chaos) θεωρείται μία υπολογιστικά αποδοτική μέθοδος UQ, που όμως απαιτεί μετατροπές στο λογισμικό επίλυσης των εξισώσεων ενός προβλήματος. Η εναλλακτική εκδοχή της μεθόδου είναι η μη–επεμβατική (non–intrusive PCE), που είναι απλούστερο να εφαρμοστεί αφού δεν συνοδεύεται από αλλαγές στο λογισμικό. Η εκδοχή αυτή είναι όμως πολύ χρονοβόρα για προβλήματα πολλών αβέβαιων μεταβλητών.

Η προτεινόμενη μέθοδος αποτελεί μία προσπάθεια συνδυασμού των πλεονεκτημάτων του Επεμβατικού και του μη–Επεμβατικού Αναπτύγματος Πολυωνυμικού Χάους. Παρουσιάζεται μία γενική προσέγγιση που απαιτεί πολύ λίγες αλλαγές στο λογισμικό. Αν και παρουσιάζεται για τις εξισώσεις Navier–Stokes για συμπιεστό ρευστό, γενικεύεται εύκολα σε άλλα προβλήματα. Η προτεινόμενη μέθοδος είναι υπολογιστικά αποδοτική και αξιόπιστη. Επιπλέον, αναπτύσσεται η συνεχής συζυγής διατύπωσή της για τον υπολογισμό των παραγώγων αντικειμενικών συναρτήσεων ως προς τις μεταβλητές σχεδιασμού στη βελτιστοποίηση με αβεβαιότητες. Και σε αυτήν την περίπτωση, στόχος είναι η προσέγγιση να είναι γενική και εύκολα εφαρμόσιμη. Τέλος, παρουσιάζονται εφαρμογές σε προβλήματα αεροδυναμικής ανάλυσης και βελτιστοποίησης στα οποία γίνεται σύγκριση της προτεινόμενης μεθόδου με υπολογισμούς που χρησιμοποιούν το μη–Επεμβατικό PCE.

National Technical University of Athens
School of Mechanical Engineering
Fluid Mechanics Department
Lab of Thermal Turbomachines
Parallel CFD & Optimization Unit

A Painless Intrusive Polynomial Chaos Expansion Approach to the CFD Analysis and the Adjoint-based Optimization

Diploma Thesis

Michail Chatzimanolakis

Supervisor : K.C. Giannakoglou, Professor

Abstract

This diploma thesis proposes a method of Uncertainty Quantification (UQ) for use in aerodynamic analysis and optimization under uncertainties, based on the Polynomial Chaos Expansion (PCE) theory, namely its intrusive variant. Intrusive PCE is considered to be a computationally efficient UQ method; however, it asks for changes in the software used to solve the governing equations. Thus, it is a problem–specific approach. The alternative PCE variant, the non–intrusive one, is easier to implement, as it does not require any software changes but is computationally expensive for problems with many uncertain variables.

The method proposed in this diploma thesis is an effort to combine the merits of the intrusive and non–intrusive PCE variants; a general approach is presented that requires very few software changes and is not specific to the equations governing a problem. At the same time, the proposed method is computationally efficient and robust. Though herein developed for the Navier–Stokes equations for compressible fluids, the proposed method can be extended to other disciplines governed by different systems of equations, in a straightforward manner. Over and above, the continuous adjoint formulation of the proposed method is developed, in order to compute the gradients of objective functions in aerodynamic shape optimization problems. Again, emphasis is laid on establishing a general approach that is easy to implement. Applications in aerodynamic analysis and optimization problems, that compare the method to its non–intrusive variant are presented.

Acknowledgements

First of all, I would like to express my sincere gratitude to my professor, Kyriakos Giannakoglou. His guidance has been invalueable; starting from teaching me the very basics of Fortran coding and numerical analysis during my first years in NTUA to trusting me to work as a part of his research group in the years after. His advice, eagerness to help and his availability truly contributed to the completion of this diploma thesis. Working alongside him has been a great experience that any student would want to have.

I would also like to thank his colleague, Varvara Asouti, first for tolerating my constant questions and answering them round the clock and second for her valuable contribution in the software used in this diploma thesis; without her help all the ideas of this diploma thesis would just be theoretical equations on paper, without real– world applications. Also, the help of other members of the Parallel CFD & Optimization Unit of NTUA has been of paramount importance for the completion of this work. All of them were readily available and keen to help, wherever an issue appeared and for that I am deeply thankful.

I would also like to thank my dear friend, Kyriakos Kantarakias, with whom this work started. Together we saw it grow, fighted more than a few times over it and sat next to each other for countless hours, in front of our laptops. Our joint work and his friendship were needed to keep me going, and for that I am sincerely grateful.

Lastly, I would like to thank my family for believing in me and putting up with my sleepless and mostly noisy night hours I spent working on this diploma thesis.

‘ I have not failed.
I’ve just found 10,000 ways that won’t work.’
- Thomas Edison

Acronyms

CFD	Computational Fluid Dynamics
NTUA	National Technical University of Athens
PCopt	Parallel CFD & Optimization unit
UQ	Uncertainty Quantification
PCE	Polynomial Chaos Expansion
PDF	Probability Density Function
iPCE	Intrusive Polynomial Chaos Expansion
niPCE	non–Intrusive Polynomial Chaos Expansion
QoI	Quantity of Interest
GQ	Gauss Quadrature
PDE	Partial Differential Equation
LHS	Left Hand Side
RHS	Right Hand Side
DDSP	Deterministic Derivatives Stochastic Primal

Contents

Contents

Chapter 1

Introduction

1.1 Uncertainty Quantification in Engineering

In applications where the stochastic nature of real–world fluid mechanics problems is neglected, Computational Fluid Dynamics (CFD) methods have an excellent record of predicting capabilities. CFD codes can predict flows subject to deterministic input parameters and accurately compute quantities of interest (QoI) to the engineer. For example, the drag coefficient of an airfoil can be computed for a given infinite flow angle and infinite Mach number.

However, there are many cases where uncertainties have a quantifiable and non–negligible effect on the behavior of systems; for instance, a slight change in a compressor's inlet flow angle may vastly affect its performance. In this case, the boundary condition of the compressor's inlet flow angle would follow a particular probability distribution and the engineer would be interested in finding the probability distribution followed by the QoI. In other words, the goal is to correctly **propagate** input uncertainties to some output, which is achieved through the process of **Uncertainty Quantification (UQ)**. Several UQ methods are mentioned below.

Stochastic Sampling

The most precise and exact UQ method is the **Monte–Carlo** technique. This simply involves sampling, i.e. solving the deterministic problem enough times, each time randomily choosing the stochastic inputs, so that these choices obey the inputs' probability distributions. Then, the distribution of the QoI can be determined. Although accurate, the standard Monte–Carlo method is simply too expensive in real–world applications, since a single CFD evaluation may take hours to complete and the convergence rate of the method is proportional to $1/\sqrt{N}$, for N samples [1].

To this end, more efficient stochastic sampling techniques have been developed. The **quasi–Monte Carlo** method uses quasi–random sequences of uncertain inputs, that share some properties of sequences of random inputs used in the standard

Monte–Carlo; this yields a convergence rate proportional to $(\log N)^8/N$, for N samples [2]. Another sampling technique was developed by McKay in [3] and is known as the **Latin Hypercube** sampling. In this case, the samples taken have to satisfy particular constraints, which make the sampling independant of the number of uncertain variables. Even with these improvements, stochastic sampling techniques are still not affordable for CFD applications and are mainly limited to other areas, such as computational finance.

Method of Moments

The **Method of Moments**, or Perturbation method, approximates the QoI with its Taylor Expansion in terms of the input uncertain variables, about their mean [4]. The expansion is usually truncated up to second–order and the moments of the QoI are directly approximated from the moments of the truncated expansion. The second–order truncation makes the method valid for small input and output variations; however, in [5] a higher order truncation scheme is applied and the statistical moments of outputs are expressed as functions of its derivatives with respect to the uncertain variables.

Stochastic Collocation

Stochastic Collocation methods are based on interpolation schemes, in order to compute stochastic quantities. Several types of interpolation schemes for the QoI have been adopted, such as piecewise linear of Lagrange interpolation [6],[7],[8]. The interpolation is constructed by sampling the QoI at a set of nodes in the stochastic space of the uncertain variables. In this case, the key issue is the selection of nodes, so that the obtained approximation is good enough, while the number of samples remains affordable.

Spectral Methods

In spectral methods, the QoI is expressed in terms of a series of basis functions which represent the spectrum of the uncertain inputs. The **Karhunen–Loève Expansion** [9] is a spectral method in which the stochastic QoI is expressed in terms of a series of orthogonal functions that are determined after solving an integral equation [10].

The **Polynomial Chaos Expansion (PCE)** is a another spectral method. The PCE relies upon the use of orthogonal polynomial bases to express the dependance of the evaluation model's outputs to the uncertain variables [11, 12, 13, 14]. This idea was originally proposed by Wiener in [15], for Gaussian processes, and was later on generalized by Xiu and Karniadakis for any probability distribution, in [16]. In numerical applications, PCE methods follow either an intrusive or a non–intrusive approach, depending on whether software programming is involved or not.

Non–intrusive PCE (niPCE) has the advantage of not altering the CFD code. Instead, the truncated spectral representation of the QoI is used and the coefficients of the basis functions of the PCE are found by using existing software as a 'black box'.

This is done by taking advantage of the orthogonal polynomial basis, which allows for the expression of every PCE coefficient in terms of integrals involving the QoI. Those integrals are computed by computing the values of the QoI at the so–called Gaussian nodes. This method's efficiency, in comparison with other UQ methods [17, 18] and its theoretical background [19] have been thoroughly studied, established and applied [20], the main issue being the so–called 'Curse of Dimensionality', which means that the number of samples increases exponentially, as the uncertain variables increase.

The niPCE method is, thus, very similar to the Stochastic Collocation, their difference being the chosen basis; in the Stochastic Collocation this choice depends on the interpolation scheme (which is often the Lagrange polynomials), while in the niPCE it depends on the PDFs of the stochastic inputs, since the chosen polynomial basis is orthogonal with respect to those PDFs. An interesting comparison between the two can be found in [21]. Regarding the main drawback of the niPCE (and the Stochastic Collocation method), the Curse of Dimensionality, several attempts have been made to reduce its computational cost. The involved integrals, which require the sampling of the QoI can be computed through Gauss Quadrature, using a sparse set of Smolyak nodes [22]. Alternatively, a least squares approach can be taken, in order to further reduce the number of samples required [23].

On the other hand, in **intrusive PCE (iPCE)** the uncertain variables are introduced into the governing equations and a new set of equations is derived through Galerkin projections, that are solved in order to compute the PCE of the flow variables [24],[25]. The iPCE method requires the derivation of the governing equations and the corresponding boundary conditions, their discretization, the formulation of the appropriate numerical solution scheme and extensive software development. The numerical solution of the new system of coupled PDEs provides the PCE coefficient fields of the flow variables. A detailed comparison between the iPCE and the niPCE is provided in [26].

1.2 Robust Design Optimization

Inherent uncertainties in the operating/environmental conditions of a system result in performance uncertainty which gives rise to the need of Robust Design, i.e. the art of designing systems the performance of which is not signlificantly affected by expected changes in the environment. Mathematically speaking, while a conventional design/optimization process aims to minimize an objective function F (in this case, the terms objective function and QoI can be used indifferently), the robust design optimization aims to minimize

$$\mu_F + k\sigma_F \ , \ k \in \mathbb{R}^+ \tag{1.1}$$

where μ_F stands for the mean value and σ_F for the variance of the QoI, while k is a user–defined weight.

In fig.1.1 an objective function of a single design variable b (which is though stochastic) is plotted in terms of b, along with the conventional optimization solution and

the robust design solution. Due to the presence of uncertainties, this system is expected to operate around the 'expected' (otherwise constant) value of the design variable. For this reason, it becomes clear that the robust solution is preferable to the conventional solution, although the latter may sometimes have lower values.

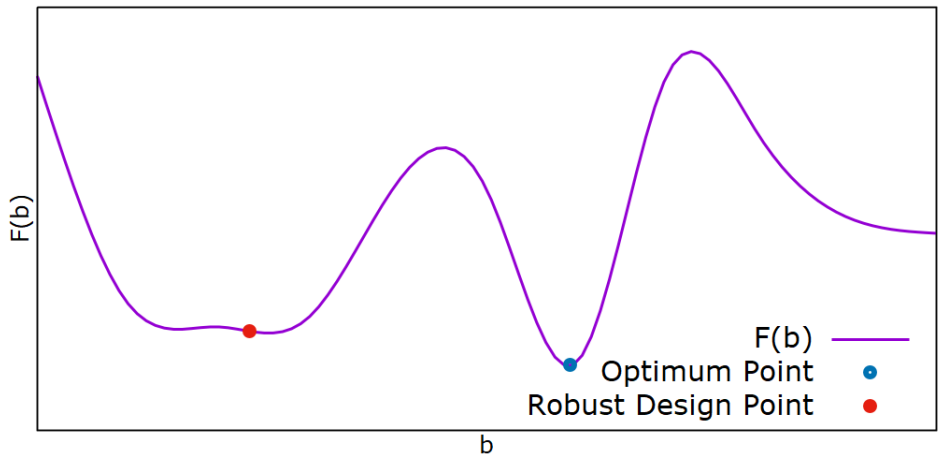


Figure 1.1: *Conventional optimization solutions and robust design solutions.*

UQ methods, such as the PCE, allow for the evaluation of functions like the one given in eq. 1.1, as they propagate the uncertainty from inputs to outputs.

A UQ method alone would suffice for a Robust Design optimization which is done through stochastic methods, such as evolutionary algorithms. This has been done in [27], where the niPCE was used as a UQ tool for the shape optimization of an airfoil. A main advantage of such an approach is that it requires absolutely no software development; the only requirement is the solver of the problem without uncertainties which is used for the samples of the niPCE. The niPCE is, then, used as a means to evaluate a function like the one in eq. 1.1, for the needs of the evolutionary algorithm. Similar approaches have been presented in [28],[29],[30],[31].

However, non–intrusive approaches combined with stochastic optimization methods may have an increased computational cost, when compared to optimization without uncertainties, since a single evaluation of the QoI/objective function in the first case is much more expensive; moreover, the cost of the stochastic optimization methods is usually much higher than that of their alternative, gradient–based methods. For this reason, adjoint–based techniques, that allow for the calculation of gradients need for optimization and/or UQ purposes are developed.

Regarding UQ methods, in [32], a discrete adjoint techinque is developed that allows for the calculation of gradients necessary for the implementation of the Method of Moments, in Nuclear Thermal–Fluids; the results are compared to those of the Monte–Carlo method. A similar approach is presented in [33], for nuclear energy problems. In this case though, the continuous adjoint of the problem is derived, in order to compute the necessary gradients.

For optimization purposes and Robust Design, adjoint methods are also implemented. In [34], a continuous adjoint method is developed, which is combined

with direct differentiation; this yields the sensitivities required by the Method of Moments and the ones needed for gradient–based optimization. A similar approach for compressible or incompressible industrial applications is presented in [35].

1.3 Structure of this Diploma Thesis

This diploma thesis proposes and alternative UQ approach that is based on the iPCE method. Emphasis is laid on making the proposed approach painless, so that the involved programming is as little as possible. Moreover, contrary to other problem–specific intrusive methods, the proposed one is more general and applicable to any problem that is governed by its own set of PDEs. Over and above, a continuous adjoint–based method that allows for the computation of the gradients required by robust design optimization is proposed. In summary, this diploma thesis contains the following chapters:

- **Chapter 2:** A brief introduction to the mathematical background of the PCE theory is outlined. Orthogonal polynomials are discussed here, akong with some of their properties.
- **Chapter 3:** The proposed iPCE method is described. Some propositions are given, along with the way the method was numerically set up and programmed.
- **Chapter 4:** Numerical applications of the method in 2D and 3D aerodynamic problems are presented, along with comparisons in terms of accuracy and computational time with the Monte–Carlo and the niPCE method.
- **Chapter 5:** The continuous adjoint method of the iPCE equations is presented.
- **Chapter 6:** A demonstration of the proposed continuous adjoint is shown, for the 2D Euler equations.
- **Chapter 7:** The continuous adjoint method is applied to the shape optimization under uncertainties, to a 2D airfoil.
- **Chapter 8:** An alternative the continuous adjoint of the iPCE equations is proposed, that aims to further reduce the computational cost.
- **Chapter 9:** Conclusions and future research ideas are summarized here.

Chapter 2

Orthogonal Polynomials and PCE

In this chapter, orthogonal polynomials are introduced, first in one and then in multiple dimensions. All theorems, properties and propositions presented in the chapter are thourougly analyzed and proven in [36]. Orthogonal polynomials are a key aspect of the PCE theory, as they are the basis used for the spectral expansion involved. Then, the PCE is discussed.

2.1 Univariate Orthogonal Polynomials

Let $w(\xi)$ denote a continuous and positive function, defined on the interval (a, b), such that the moments $\int_a^b \xi^n w(\xi) d\xi$ exist $\forall n \in \mathbb{N}$.
Then, the integral

$$\langle f, g \rangle_w := \int_a^b f(\xi) g(\xi) w(\xi) d\xi \tag{2.1}$$

is an inner product of the polynomials f and g, in (a, b). The function w is called the weight function for that inner product. The subscript w in the inner product will be sometimes ommited, when it can easily be implied.
Definition 2.1.1 (Orthogonal Polynomials)**.** A sequence of polynomials $\{p_n(\xi)\}_{n=0}^{\infty}$ with $degree[p_n] = n$ is called orthogonal with respect to the weight function $w(\xi)$ on the interval (a, b) if

$$\int_a^b p_n(\xi) p_m(\xi) w(\xi) d\xi = \delta_{mn} \langle p_n, p_n \rangle \tag{2.2}$$

where δ_{mn} is the Kronecker delta. If $< p_n, p_n >= 1 \ \forall n \in \mathbb{N}$, the sequence is called **orthonormal**. Also, (a, b) is the interval of orthogonality.

In order to obtain a sequence of orthogonal polynomials the following process can be followed, known as the **Gram–Schmidt orthogonalization**. First, p_0 is arbitrarily chosen. Then, each polynomial of the sequence can be obtained recursively, using

the formula

$$p_k(\xi) = \xi^k - \sum_{j=1}^{k-1} \frac{\langle \xi^k, p_j \rangle}{\langle p_j, p_j \rangle} p_j(\xi) \tag{2.3}$$

It is easy to see that eq. 2.3 defines a polynomial p_k that is orthogonal to all $p_j, \quad j = 0, \ldots, k-1$, since

$$\langle p_k, p_i \rangle = \langle \xi^k, p_i \rangle - \sum_{j=1}^{k-1} \frac{\langle \xi^k, p_j \rangle}{\langle p_j, p_j \rangle} \delta_{ij} = 0$$

Also, note that $degree[p_n] = n$ implies that the polynomials generated this way are linearly independent and, hence, form a basis of $\mathbb{R}$.

2.1.1 Common Univariate Orthogonal Polynomial Sequences

In what follows, two commonly used orthogonal polynomial sequences are discussed, the Hermite and the Legendre polynomials, which are orthogonal with respect to the normal and the uniform probability distributions, respectively. These distributions will be used later on, in the applications presented in this diploma thesis.

Hermite Polynomials

This section discusses the probabilists' Hermite polynomials, not to be confused with the physicists' Hermite polynomials.
The Hermite polynomials $\{He_n\}$ are orthogonal in $(-\infty, +\infty)$ with respect to the normal distribution $w(\xi) = \frac{1}{\sqrt{2\pi}} e^{-\xi^2/2}$. They satisfy the following recurrence formula

$$He_{n+1}(\xi) = \xi He_n(\xi) - nHe_{n-1}(\xi) \tag{2.4}$$

and their inner product is

$$\int_{\mathbb{R}} He_n(\xi) He_m(\xi) \frac{1}{\sqrt{2\pi}} e^{-\xi^2/2} d\xi = n! \delta_{mn} \tag{2.5}$$

Moreover, the Hermite polynomials are explicitly given by the following formula

$$He_n(\xi) = n! \sum_{m=0}^{\lfloor n/2 \rfloor} \frac{(-1)^m}{m!(n-2m)!} \frac{\xi^{n-2m}}{2^m} \tag{2.6}$$

where $\lfloor\rfloor$ denotes the floor function: $\lfloor x \rfloor = max\{m \in \mathbb{Z} | m \leq x\}$. Their triple product, is [37]

$$\begin{aligned}\langle He_l, He_m, He_n \rangle &:= \int_{-\infty}^{+\infty} He_l(\xi) He_m(\xi) He_n(\xi) \frac{e^{-\xi^2/2}}{\sqrt{2\pi}} d\xi \\ &= \frac{l!m!n!}{(\frac{l+m-n}{2})!(\frac{m+n-l}{2})!(\frac{n+l-m}{2})!}\end{aligned} \tag{2.7}$$

if $l+m+n$ is even and the sum of any two of l,m,n is not less than the third, and is zero otherwise. The first six Hermite polynomials are

$$\begin{aligned} He_0(\xi) &= 1 & He_3(\xi) &= \xi^3 - 3\xi \\ He_1(\xi) &= \xi & He_4(\xi) &= \xi^4 - 6\xi^2 + 3 \\ He_2(\xi) &= \xi^2 - 1 & He_5(\xi) &= \xi^5 - 10\xi^3 + 15 \end{aligned} \tag{2.8}$$

Legendre Polynomials

The Legendre polynomials $\{P_n\}$ are orthogonal in $(-1, 1)$ with respect to the uniform distribution $w(\xi) = \frac{1}{2}$. They satisfy the following recurrence formula

$$(n+1)P_{n+1}(\xi) = (2n+1)\xi P_n(\xi) - nP_{n-1}(\xi) \tag{2.9}$$

and their inner product is

$$\int_{-1}^{1} P_n(\xi) P_m(\xi) \frac{1}{2} d\xi = \frac{1}{2n+1} \delta_{mn} \tag{2.10}$$

They can also be expilicitly found by the formula

$$P_n(\xi) = \frac{1}{2^n} \sum_{k=0}^{n} \binom{n}{k} \binom{n+k}{k} \left(\frac{\xi - 1}{2} \right)^k \tag{2.11}$$

while their triple product is

$$\begin{aligned}\langle P_l, P_m, P_n \rangle &:= \int_{-1}^{1} P_l(\xi) P_m(\xi) P_n(\xi) \frac{1}{2} d\xi \\ &= (-1)^s \sqrt{\frac{(2s-2n)!(2s-2l)!(2s-2m)!}{(2s+1)!}} \frac{s!}{(s-n)!(s-l)!(s-m)!}\end{aligned} \tag{2.12}$$

when $2s = n + l + m$ is even, while it is zero otherwise. The first six Legendre polynomials are

$$\begin{aligned} P_0(\xi) &= 1 & P_3(\xi) &= \frac{1}{2}(5\xi^3 - 3\xi) \\ P_1(\xi) &= \xi & P_4(\xi) &= \frac{1}{8}(35\xi^4 - 30\xi^2 + 3) \\ P_2(\xi) &= \frac{1}{2}(3\xi^2 - 1) & P_5(\xi) &= \frac{1}{8}(63\xi^5 - 70\xi^3 + 15\xi) \end{aligned} \tag{2.13}$$

2.1.2 Some Properties of Orthogonal Polynomials

In this section, two propositions concerning univariate orthogonal polynomial sequences are given, [36].
Proposition 2.1.1. Every sequence of orthogonal polynomials $\{p_n(\xi)\}_{n=0}^{\infty}$ satisfies the recurrence relation

$$p_{n+1}(\xi) = (A_n\xi + B_n)p_n(\xi) + C_n p_{n-1}(\xi)$$

where $A_n = \frac{k_{n+1}}{k_n}$, $C_n = \frac{-A_n}{A_{n-1}} \frac{\langle p_n, p_n \rangle}{\langle p_{n-1}, p_{n-1} \rangle}$ and k_n is the coefficient of ξ^n in p_n.
Proposition 2.1.2. Each polynomial p_n of a sequence of orthogonal polynomials $\{p_n(\xi)\}_{n=0}^{\infty}$ has exactly n real simple roots in its interval of orthogonality. Also, the roots of $p_n(\xi)$ and of $p_{n+1}(\xi)$ alternate, i.e. between any two roots of p_{n+1} there is a root of p_n.

2.2 Multivariate Orthogonal Polynomials

Assume m sequences of univariate orthogonal polynomials $p^k \equiv \{p_n^k(\xi_k)\}_{n=0}^{\infty}, k = 1, \ldots, m$. Each sequence is orthogonal with respect to a weight function $w_k(\xi_k)$ with domain $\mathcal{E}_k$. Between any two of those sequences a tensor product can be defined as follows.
Definition 2.2.1. The tensor product of two sequences of functions $A = \{a_n(\xi_1)\}_{n=0}^{\infty}$ and $B = \{b_n(\xi_2)\}_{n=0}^{\infty}$ is defined as

$$A \otimes B := \{a_{n_1}(\xi_1)b_{n_2}(\xi_2)\}_{n_1,n_2=0}^{\infty} = \{a_0b_0, a_1b_0, a_0b_1, a_1b_1, a_2b_0, a_0b_2, \ldots\} \tag{2.14}$$

So, the following sequence of m–variate polynomials can be defined

$$Y \equiv \{Y_n\}_{n=0}^{\infty} := \otimes_{k=1}^{m} p^k = \{p_{n_1}^1(\xi_1)p_{n_2}^2(\xi_2)\ldots p_{n_m}^m(\xi_m)\}_{n_1,n_2,\ldots,n_m=0}^{\infty} \tag{2.15}$$

That is, the polynomials of this new sequence are formed as all possible combinations of products of m univariate polynomials. Therefore, in order to obtain all $m-$variate polynomials of a given degree p it is necessary to find all sets of integers $n_i \geq 0$, $i = 1, \ldots, m$ so that $n_1 + \cdots + n_m = p$. This can be achieved through an algorithm by Thomas Gerstner,[38], which can also be found in [39].

These polynomials are orthogonal with respect to the inner product given by

$$< f, g >_W = \int_{\mathcal{E}} f g W d\xi_1 \dots d\xi_m \quad , \quad W := \prod_{j=1}^{m} w_j(\xi_j) \tag{2.16}$$

which can be proven by writing

$$\int_{\mathcal{E}} Y_k Y_l W dx_1 \dots dx_m =$$

$$\int_{\mathcal{E}_1} p^1_{n_1}(x_1) p^1_{l_1}(x_1) w_1 dx_1 \cdots \int_{\mathcal{E}_m} p^m_{n_1}(x_m) p^m_{l_1}(x_m) w_m dx_m =$$

$$\delta_{n_1 l_1} < p_{n_1}, p_{n_1} >_{w_1} \dots \delta_{n_m l_m} < p_{n_m}, p_{n_m} >_{w_m} = \delta_{kl} < Y_k, Y_k >_W$$

The following combinatorics propositions may be useful.
Proposition 2.2.1. The total number of m–variate polynomials of degree d is $\binom{d+m-1}{d} = \frac{(d+m-1)!}{(m-1)!d!}$.
Proposition 2.2.2. The total number of m–variate polynomials of degree d or less is $\binom{d+m}{d} = \frac{(d+m)!}{m!d!}$.

2.3 PCE of a Function

Let $\boldsymbol{\xi} = (\xi_1, \dots, \xi_m)$ be a set of m uncorrelated uncertain variables, each associated with its own probability density function (PDF) $w_i(\xi_i)$ with domain $\mathcal{E}_i$. Also, let $\phi = \phi(\boldsymbol{\xi})$ be a function of $\boldsymbol{\xi}$.
Definition 2.3.1. The PCE of $\phi(\boldsymbol{\xi})$ is defined as the infinite series

$$\phi(\xi) = \sum_{j=0}^{\infty} \phi^j Y_j(\boldsymbol{\xi}) \tag{2.17}$$

where the polynomials Y_j are orthogonal with respect to $W(\boldsymbol{\xi}) := \prod_{j=1}^{m} w_j(\xi_j)$ and the spectral coefficients of the series are given by

$$\phi^j := \langle \phi(\xi), Y_j \rangle \tag{2.18}$$

which are the so–called Galerkin projections of ϕ to the polynomial Y_j.

This idea was originally proposed by Wiener in [15], for normally distributed variables and was later on generalized by Xiu and Karniadakis in [16]. An interesting property of the PCE of a function is stated in what follows.
Proposition 2.3.1. The spectral coefficients of the PCE of a function ϕ satisfy the

relations

$$E[\phi] \equiv \mu_\phi = \phi^0$$
$$Var[\phi] \equiv \sigma_\phi^2 = \sum_{j=1}^{\infty} (< Y_j, Y_j > \phi^j)^2 \tag{2.19}$$

This can be easily proven, as

$$\mu_\phi \equiv \int_{\mathcal{E}} \phi W d\boldsymbol{\xi} = \int_{\mathcal{E}} \phi Y_0 W d\boldsymbol{\xi} = \phi^0$$

$$\sigma_\phi^2 \equiv \int_{\mathcal{E}} (\phi - \mu_\phi)^2 W d\boldsymbol{\xi} = \int_{\mathcal{E}} \left(\sum_{j=0}^{\infty} \phi^j Y_j(\xi) - \phi^0 \right)^2 W d\boldsymbol{\xi} =$$

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \phi^j \phi^k \int_{\mathcal{E}} Y_j(\xi) Y_k(\xi) W d\boldsymbol{\xi} = \sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \phi^j \phi^k \delta_{jk} < Y_j, Y_k >= \sum_{j=1}^{\infty} (< Y_j, Y_j > \phi^j)^2$$

For a sequence of orthonormal polynomials, $< Y_k, Y_k >= 1$, eqs. 2.20 simplify to

$$E[\phi] \equiv \mu_\phi = \phi^0$$
$$Var[\phi] \equiv \sigma_\phi^2 = \sum_{j=1}^{\infty} (\phi^j)^2 \tag{2.20}$$

Therefore, the knowledge of the spectral coefficients is sufficient to fully determine the statistical behavior of a function of the uncertain variables $\boldsymbol{\xi}$. Higher statistical moments of a quantity can also be found by applying their definition to the PCE of that quantity. For example, the skewness is given by

$$\gamma_\phi \equiv E\left[\left(\frac{\phi - \mu_\phi}{\sigma_\phi} \right)^3 \right] = \frac{1}{\sigma_\phi^3} \int_{\mathcal{E}} (\phi - \phi^0)^3 W d\boldsymbol{\xi} = \frac{1}{\sigma_\phi^3} \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \langle Y_i, Y_j, Y_k \rangle \tag{2.21}$$

In what follows, two ways to implement the PCE of a function in UQ problems are discussed, the non–intrusive PCE (niPCE) and the intrusive PCE (iPCE). But first it is necessary to define what a UQ problem is.

2.4 Uncertainty Quantification & Propagation

Assume a set of n partial differential equations (PDEs), written as

$$\boldsymbol{R}(\boldsymbol{U}) = \boldsymbol{0} \quad , \quad \boldsymbol{U} \in \mathbb{R}^n \tag{2.22}$$

to be solved for the field variables $\boldsymbol{U}$; in fluid–mechanics applications, eq. 2.22 might be the Navier–Stokes equations. They are solved subject to some boundary conditions and many other input parameters, such as the heat capacity of a gas or

other fluid properties, which have fixed values. The solution of eq. 2.22 aims to compute the flow field and, then, the value of a Quantity of Interest (QoI), such as the drag or the lift coefficient of an aircraft, and is done numerically through some software. This can be thought of as a *deterministic* problem.

The UQ or *stochastic* problem is the case where the boundary conditions and/or the other input parameters needed to solve the equations are not known to have a fixed value, but to follow probability distributions. In this case, the goal is to find the probability distribution of the QoI, i.e. to **propagate** the uncertainty of the input parameters to some output.

More specifically, let us assume that the input parameters of the problem are functions of m uncorrelated uncertain variables $\boldsymbol{\xi} \in \mathbb{R}^m$, each with its own probability density function (PDF) $w_k(\xi_k)$ and domain $\mathcal{E}_k \ , \ k = 1, \ldots, m$. If that is the case, the field variables should also depend on $\boldsymbol{\xi}$; $\boldsymbol{U} = \boldsymbol{U}(\boldsymbol{\xi})$ which means that the QoI is also a function of the uncertain variables

$$F = F(\boldsymbol{U}) = F(\boldsymbol{U}(\boldsymbol{\xi})) \tag{2.23}$$

Thus, finding the function of $\boldsymbol{\xi}$ in eq. 2.23 is the solution of the stochastic problem.

2.5 Non–Intrusive PCE (niPCE)

In the non–intrusive PCE (niPCE), where the PCE is applied directly to the QoI, eq. 2.23 becomes

$$F = \sum_{j=0}^{\infty} F^j Y_j(\boldsymbol{\xi}) \tag{2.24}$$

The polynomial basis is chosen to be orthogonal to $W := \prod_{j=1}^{m} w_j(\xi_j)$. This choice, because of eq. 2.20, guarantees that

$$\begin{aligned} E[F] &= F^0 \\ Var[F] &= \sum_{j=1}^{\infty} (< Y_j, Y_j > F^j)^2 \end{aligned} \tag{2.25}$$

Therefore, in this case, the goal is to determine the spectral coefficients of the QoI, in eq. 2.24.

In order to do this numerically, eq. 2.24 must be truncated to a finite number of terms, denoted by $q + 1$. Several truncation schemes can be found in the literature [40], the most common being that of the so–called **chaos order**. In this case, a maximum degree of polynomials is chosen, which is called the chaos order C, and all the polynomials up to that degree are kept in the expansion. q is given by (recall proposition 2.2.1)

$$q + 1 = \frac{(C + m)!}{C! m!} \tag{2.26}$$

After a truncation scheme is applied, the result is

$$F = \sum_{j=0}^{q} F^j Y_j(\boldsymbol{\xi}) \tag{2.27}$$

and the spectral coefficients to be found are given by

$$F^j \equiv \langle F, Y_j \rangle \equiv \int_{\mathcal{E}} F Y_j W d\boldsymbol{\xi} \quad , j = 0, \ldots, q \tag{2.28}$$

As a result, the solution of the stochastic problem in the niPCE case comes down to evaluating the integrals in eq. 2.28.

Integral Evaluation through Gauss Quadrature

The computation of integrals appearing in eq. 2.28 is normally performed through Gauss Quadrature (GQ)

$$\int_{\mathcal{E}} F Y_j W d\boldsymbol{\xi} = \sum_{k=1}^{d} \omega_k F_(\boldsymbol{\xi}_k) Y_j(\boldsymbol{\xi}_k) \tag{2.29}$$

where ω_k and $\boldsymbol{\xi}_k$ are the quadrature weights and nodes.

The choice of d and its corresponding weights and nodes depends on the desired accuracy. For a tensorized grid of nodes, it is given by

$$d = (C+1)^m \tag{2.30}$$

Further discussion on the selection of nodes and weights is carried out in section 4.

It is concluded that the niPCE requires d evaluations of the function F, i.e. d numerical solutions of eq. 2.22 through some existing software, such as a Navier–Stokes equations solver. For each of these solutions, the input parameters that depend on $\boldsymbol{\xi}$ change, as $\boldsymbol{\xi}$ is equal to the value that corresponds to the GQ node each time.

2.6 Intrusive PCE (iPCE)

In the intrusive approach, the PCE is applied to the field variables (i.e. pressure, velocity components etc.) and not directly to the QoI

$$\boldsymbol{U} = \sum_{j=0}^{q} \boldsymbol{U}^j Y(\boldsymbol{\xi}) \tag{2.31}$$

and the fields $\boldsymbol{U}^j$, $j = 0, \ldots, q$ are the unknowns of the problem. Note that a particular truncation scheme (chaos order) is still necessary.

The field variables are then introduced in eq. 2.22

$$\boldsymbol{R}\left(\sum_{j=0}^{q}\boldsymbol{U}^{j}Y(\boldsymbol{\xi})\right)=\mathbf{0} \tag{2.32}$$

The necessary number of Galerkin projections are applied to eq. 2.32 afterwards, and the following new equations are derived

$$\int_{\mathcal{E}}\boldsymbol{R}\left(\sum_{j=0}^{q}\boldsymbol{U}^{j}Y(\boldsymbol{\xi})\right)Y_{k}W\,d\boldsymbol{\xi}=\mathbf{0}\ ,k=0,\ldots,q \tag{2.33}$$

which are numerically solved by altering or rewritting the original deterministic code. Finally, the QoI is computed at a post–processing level, as a function of the field variables.

2.7 A First Comparison of the niPCE and iPCE

The main advantage of the niPCE is that it can be applied in a straightforward manner, without any changes in the original code. However, as the number of uncertain variables grows, the niPCE can become computationally prohibitive. From eq. 2.30, it is deduced that the required evaluations (software runs) grow exponentially with the number of uncertain variables. This is known as the **curse of dimensionality** and can only be partially alleviated through the use of sparse quadrature grids (for instance, Smolyak grids [22]).

On the other hand, the iPCE is known to be computationally more efficient. Basically, it asks for a single solution of a larger set of equations. Unfortunatelly, the intrusive approach is specific to each problem; the equations usually have to be derived by hand seperately in each case. Also, the changes in the original code are often significant, while a change in the number of uncertain variables or chosen chaos order may result in a need for reprogramming.

Chapter 3

The Proposed iPCE Approach

In this chapter, an intrusive PCE approach is proposed which, contrary to conventional iPCE approaches, is more general and not specific to the set of governing PDEs. Several mathematical definitions are initially given, followed by some propositions and some ideas concerning the numerical application of the method. All definitions and propositions of this chapter were developed for the needs of the proposed method. Without loss of generality, all polynomial sequences used from now on will be considered **orthonormal**, i.e. $\langle Y_n, Y_n \rangle = 1$.

3.1 Some Definitions

In this section, we first define the Galerkin projection of a scalar and then extend the definition to define Galerkin projections of vectors and matrices. A property of these definitions is then shown. In all definitions, a set of m uncorrelated uncertain variables $\boldsymbol{\xi} \in \mathbb{R}^m$ are assumed, with PDFs $w_k(\xi_k)$ and domains $\mathcal{E}_k$. Also, a set of polynomials $Y = \{Y_n\}_{n=0}^{\infty}$ is assumed, that are orthogonal with respect to $W = \prod_{j=1}^{m} w_j$ in $\mathcal{E} = \prod_{j=1}^{m} \mathcal{E}_j$.

Definition 3.1.1 (Galerkin projection of scalar)**.** For any scalar $\phi(\boldsymbol{\xi})$, its Galerkin projection to the polynomial Y_j is defined as

$$\phi^j := \int_{\mathcal{E}} \phi Y_j W d\boldsymbol{\xi} \tag{3.1}$$

Definition 3.1.2 (Galerkin projection of vector)**.** For any vector $\boldsymbol{U}(\boldsymbol{\xi}) = [U_1(\boldsymbol{\xi}), ..., U_n(\boldsymbol{\xi})]^T \in \mathbb{R}^n$, its Galerkin projection of order q is defined as

$$\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}] := [\boldsymbol{U}^0, \boldsymbol{U}^1, ..., \boldsymbol{U}^q]^T \tag{3.2}$$

with $\boldsymbol{U}^k = [U_1^k, U_2^k, ..., U_n^k]^T \in \mathbb{R}^n,\ k = 0, \dots, q.$

Note that the application of the $\mathrm{G}^{\mathrm{q}}[]$ operator to a scalar is a special case of the previous definition, for $n = 1$; if $\phi = \phi(\boldsymbol{\xi}) \in \mathbb{R}$, then $\mathrm{G}^{\mathrm{q}}[\phi] = [\phi^0, \dots, \phi^q]^T$.

Definition 3.1.3 (Galerkin projection of matrix)**.** For any matrix $A \in \mathbb{R}^{n\times n}$ with components $A_{ij}{=}A_{ij}(\boldsymbol{\xi})$, its Galerkin projection of order q is defined as the block matrix

$$\mathrm{G}^{\mathrm{q}}[A] = \begin{bmatrix} A^{00} & A^{01} & \dots & A^{0q} \\ A^{10} & A^{11} & \dots & A^{1q} \\ \vdots & \vdots & \vdots & \vdots \\ A^{q0} & A^{q1} & \dots & A^{qq} \end{bmatrix} \tag{3.3}$$

where the (i, j) element of $A^{\lambda\mu} \in \mathbb{R}^{n\times n}$ is given by

$$A_{ij}^{\lambda\mu} := \int_{\mathcal{E}} A_{ij} Y_\lambda Y_\mu W d\boldsymbol{\xi} = \sum_{\rho=0}^{\infty} A_{ij}^{\rho} < Y_\rho, Y_\lambda, Y_\mu > \tag{3.4}$$

with $< Y_\rho, Y_\lambda, Y_\mu >:= \int_{\mathcal{E}} Y_\rho Y_\lambda Y_\mu W d\boldsymbol{\xi}$.

In numerical applications of the proposed method, all quantities that depend on $\boldsymbol{\xi}$ will have their own PCE and *the same truncation scheme will be applied to all of them* (in applications presented later in this diploma thesis, the chaos order truncation scheme is applied, in order to retain $q + 1$ terms in each expansion). In this case, the following propositions hold.

Proposition 3.1.1. If the expansions of the A and $\boldsymbol{U}$ components are truncated to $q + 1$ terms, namely

$$A_{ij}{=}\sum_{k=0}^{q} A_{ij}^k Y_k(\boldsymbol{\xi}) \ \text{ and } \ U_j{=}\sum_{k=0}^{q} U_j^k Y_k(\boldsymbol{\xi}) \ \text{ with } \ i, j = 1, \dots, n$$

then it can be shown that

$$\mathrm{G}^{\mathrm{q}}[A\boldsymbol{U}] = \mathrm{G}^{\mathrm{q}}[A]\,\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}] \tag{3.5}$$

Proof. Let $\boldsymbol{f}{=}A\boldsymbol{U}$ or $f_i{=}A_{ij}U_j$. Then, for any $0 \le p \le q$

$$f_i^p = (A_{ij}U_j)^p \equiv \int_{\mathcal{E}} A_{ij}U_j Y_p W d\boldsymbol{\xi} = U_j^\rho \int_{\mathcal{E}} A_{ij} Y_\rho Y_p W d\boldsymbol{\xi} = U_j^\rho A_{ij}^{\rho p}$$

which is nothing else but the p^{th} element of $\mathrm{G}^{\mathrm{q}}[A]\,\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}]$. □

Proposition 3.1.2. For two vectors $\boldsymbol{g} = (g_1(\boldsymbol{\xi}), \dots)$ and $\boldsymbol{h} = (h_1(\boldsymbol{\xi}), \dots)$ and a

constant $\lambda(\boldsymbol{\xi})$, the following property holds

$$\mathrm{G^q}\left[\boldsymbol{g}^T\right]\mathrm{G^q}\left[\lambda\boldsymbol{h}\right] = (\mathrm{G^q}\left[\boldsymbol{g}^T\boldsymbol{h}\right])^T\,\mathrm{G^q}\left[\lambda\right] \tag{3.6}$$

if their PCE are truncated to $q+1$ terms, i.e.

$$g_i = \sum_{j=0}^{q} g_i^j Y_j(\boldsymbol{\xi}) \quad , \quad h_i = \sum_{j=0}^{q} h_i^j Y_j(\boldsymbol{\xi}) \quad , \quad \lambda = \sum_{j=0}^{q} \lambda^j Y_j(\boldsymbol{\xi})$$

Proof.

$$\mathrm{G^q}\left[\boldsymbol{g}^T\right]\mathrm{G^q}\left[\lambda\boldsymbol{h}\right] = (\boldsymbol{g^j})^T(\lambda\boldsymbol{h})^j = (\boldsymbol{g^j})^T\lambda^k\boldsymbol{h}^i < Y_k, Y_i, Y_j >=$$
$$(\boldsymbol{g^j})^T\boldsymbol{h}^i < Y_k, Y_i, Y_j > \lambda^k = (\mathrm{G^q}\left[\boldsymbol{g}^T\boldsymbol{h}\right])^T\,\mathrm{G^q}\left[\lambda\right]$$

□

The aforementioned propositions are essential for the derivation of the numerical solution scheme of the iPCE equations, presented in the next subsection. Moreover, they facilitate the necessary mathematical work for the derivation of the continuous adjoint iPCE equations, as shown in chapter 5.

3.2 Proposed Numerical Solution of iPCE Equations

Let us consider a problem governed by a system of n PDEs (such as the Navier–Stokes equations, for instance), which can be written *in discrete form* as

$$\boldsymbol{R}(\boldsymbol{U}) = \boldsymbol{0} \tag{3.7}$$

with unknown variables $\boldsymbol{U} \in \mathbb{R}^n$ at each grid node. In the above system, uncertainty is introduced through the vector of uncertain variables $\boldsymbol{\xi} \in \mathbb{R}^m$, affecting the boundary conditions and/or other input parameters.

For non–linear problems, the system in eq. 3.7 can be solved by applying the iterative scheme

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)_{old} \Delta\boldsymbol{U} = -\left(\boldsymbol{R}\right)_{old} \quad , \quad \Delta\boldsymbol{U} = \boldsymbol{U}_{new} - \boldsymbol{U}_{old} \tag{3.8}$$

which is a linear system that separates numerics (the LHS Jacobian) from physics (the RHS). This system is solved for $\Delta\boldsymbol{U}$, followed by an updating step

$$\boldsymbol{U}_{new} = \boldsymbol{U}_{old} + \Delta\boldsymbol{U} \tag{3.9}$$

of the values of the field variables at each grid node. Then, the LHS and RHS are recalculated and the system is solved again, until convergence (sufficiently small $\boldsymbol{R}$) is reached.

The procedure described above will be applied to solve the iPCE equations, eq. 2.33. Merely by changing notation, eq. 2.33 can be written as

$$\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{R}\right] = \boldsymbol{0} \tag{3.10}$$

which is to be solved for the $q+1$ unknown fields $\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{U}\right]$. To this end, the $\mathrm{G}^{\mathrm{q}}\left[\right]$ operator is applied to eq.3.8, leading to (indices 'old' and 'new' are ommited from now on)

$$\mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\Delta\boldsymbol{U}\right] = \mathrm{G}^{\mathrm{q}}\left[-\boldsymbol{R}\right]$$

which, when combined with proposition 3.5, gives

$$\mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right]\mathrm{G}^{\mathrm{q}}\left[\Delta\boldsymbol{U}\right] = -\,\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{R}\right] \tag{3.11}$$

In eq. 3.11 $\mathrm{G}^{\mathrm{q}}\left[\Delta\boldsymbol{U}\right]$ are corrections to the unknowns $\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{U}\right]$ of the iPCE equations and $\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{R}\right]$ are the corresponding residuals. The latter can be computed by Gauss quadrature without explicitly deriving the iPCE equations. Instead, the involved integrals are found by evaluating $\boldsymbol{R}$ at specific values of $\boldsymbol{\xi}$ at the quadrature nodes. On the contrary, the conventional iPCE approach would require the explicit derivation of the equations, in order to calculate their residuals. A more detailed discussion on the evaluation of integrals using Gauss quadrature is made in section 3.5.

Regarding the LHS of eq. 3.11, it suffices to prove that $\mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right]$ is the exact Jacobian of the iPCE equations, i.e. the Jacobian we would get if we differentiated the discrete iPCE problem, eq. 3.10.

Proposition 3.2.1 (Exact Jacobian of discrete iPCE problem). Differentiation of the discrete iPCE problem given by eq.3.10 with respect to $\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{U}\right]$ is equivalent to the application of the $\mathrm{G}^{\mathrm{q}}\left[\right]$ operator to $\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}$, i.e.

$$\frac{\partial(\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{R}\right])}{\partial(\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{U}\right])} = \mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right] \tag{3.12}$$

Proof. Recall the PCE of $\boldsymbol{U}$ as $\boldsymbol{U} = \sum_{i=0}^{q} \boldsymbol{U}^i Y_i$ which yields $\frac{\partial \boldsymbol{U}}{\partial \boldsymbol{U}^i} = Y_i I$, with I the identity matrix. Therefore

$$\frac{\partial \phi}{\partial \boldsymbol{U}^\lambda} = Y_\lambda \, \frac{\partial \phi}{\partial \boldsymbol{U}} \tag{3.13}$$

for any scalar ϕ. Because of eq. 3.13, the (i,j) element of the (λ,μ) block of matrix $\mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right]$ is

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)_{ij}^{\lambda\mu} \equiv \int_{\mathcal{E}} Y_\lambda Y_\mu \left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)_{ij} W d\boldsymbol{\xi} = \int_{\mathcal{E}} Y_\mu \frac{\partial R_i}{\partial U_j^\lambda} W d\boldsymbol{\xi} = \left(\frac{\partial R_i}{\partial U_j^\lambda}\right)^{\mu}$$

which is equal to the corresponding element of $\frac{\partial(\mathrm{G}^{\mathrm{q}}[\boldsymbol{R}])}{\partial(\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}])}$, namely $\frac{\partial R_i^\mu}{\partial U_j^\lambda}$. □

Proposition 3.2.1 implies that the linearization of the discrete iPCE problem, eq.3.11, does not need to be explicitly derived. Instead of differentiating the iPCE equations, existing routines that compute the LHS and RHS of the deterministic problem are sufficient, in order to form and solve eq. 3.11. These codes evaluate the LHS and RHS at specific GQ nodes, for specific values of $\boldsymbol{\xi}$, which allows for the computation of the integrals involved in the Galerkin projections of the $\mathrm{G}^{\mathrm{q}}[]$ operator.

The procedure described above allows for the solution of the iPCE equations without the need to derive anything by hand, which is definitely a cumbersome task. In contrast to the standard approach which requires reprogramming for different chaos orders, the proposed method is flexible and the same software could handle any chaos order, number of uncertain variables or type of governing PDEs. Therefore, from this point of view, it enjoys the simplicity of the niPCE approach, with reduced computational cost though.

3.3 A First Comparison with the niPCE

The numerical stability and convergence rate of the proposed iPCE method is strongly related to the properties of the deterministic problem. The non–intrusive solution of eq. 3.10 would involve solving eq. 3.7 several times, one for each value of the uncertain variables at the current quadrature node. Each time, this would take n_j solutions (iterations) of eq.3.8, $j = 1, \ldots, d$, where d denotes the number of GQ nodes.

Proposition 3.3.1. The number of solutions of eq.3.11 needed to achieve convergence is equal to $\max(n_1, \ldots, n_b)$. Also, if all the non–intrusive runs and the intrusive one are each stopped after p iterations, they will produce the same $\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}]$.

Proof. The proof can be found in Appendix A. □

This essentially means that both methods will converge to the *same result in the same number of iterations*. Thus, a comparison between them can be made if the computational cost per iteration is compared.

Each iteration of the iPCE solver includes:

1. Computation of the LHS and RHS terms of eq. 3.11, through Gauss quadrature –based Galerkin projections of residuals and Jacobians.
2. Solution of the resulting system eq. 3.11.

On the other hand, the niPCE method requires d distinct numerical solutions of the standard PDEs. Per iteration this calls for:

1. d computations of the LHS and RHS terms of eq.3.8.
2. d solutions of the resulting systems, eq.3.8.

The cost to compute the LHS and RHS terms *within each iteration* is thus considered to be almost the same between the two variants; this is a key feature of the proposed iPCE method, in which the equations' residuals, for instance, are computed by the corresponding routine, used as a 'black box', at each $\boldsymbol{\xi}$ of the quadrature nodes. Essentially, we could say that the system given by eq.3.8 is formed via a non–intrusive approach.

The main difference can be found in the solution step. While the niPCE solves d systems of dimension say n, the iPCE solves a single system of dimension $(q+1)n$. If the assumption that the solution cost is proportional to n^2 is made, the iPCE is faster when

$$(q+1)^2 n^2 < d^2 n^2 \Rightarrow \frac{(C+m)!}{C!m!} < (C+1)^m \tag{3.14}$$

which is true when $m = 6$ or 7, depending on the choice of chaos order. For the sake of convenience, we the repeat that C is the chosen chaos order, m is the number of uncertain variables, $q+1$ denotes the number of retained terms each PCE and d is the number of GQ nodes.

3.4 Reducing Memory Requirements and Computational Cost of the iPCE

As was previously demonstrated, the proposed iPCE method does not seem to significantly outperform the niPCE, in terms of computational cost. Moreover, the involved matrices are now of much larger dimensions ($(q+1)^2$ larger), which results in important memory requirements. Efforts to reduce the computational and memory burden of the iPCE method have been made in [41], which was, however, a problem–specific approach applicable only to load–flows in power systems. This section will provide a more general way to remedy the aforementioned issues, that is applicable to any governing equations of a problem.

In order to handle the solution of eq. 3.11, this is rewritten, in matrix form, as

$$\begin{bmatrix} \mathcal{J}^{00} & \mathcal{J}^{01} & \dots & \mathcal{J}^{0q} \\ \mathcal{J}^{10} & \mathcal{J}^{11} & \dots & \mathcal{J}^{1q} \\ \vdots & \vdots & \vdots & \vdots \\ \mathcal{J}^{q0} & \mathcal{J}^{q1} & \dots & \mathcal{J}^{qq} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{U}^0 \\ \Delta \boldsymbol{U}^1 \\ \vdots \\ \Delta \boldsymbol{U}^q \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}^0 \\ \boldsymbol{R}^1 \\ \vdots \\ \boldsymbol{R}^q \end{bmatrix} \tag{3.15}$$

where $\mathcal{J} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}$. System 3.15 can be decoupled allowing the numerical solution of linear systems of smaller size. To this end, the $\boldsymbol{U}^0$ field which denotes mean flow variables' fields is approximated by the $\boldsymbol{U}$ field resulting from a single solution of the problem without uncertainties, eqs. 3.7. This solution is done for a given $\boldsymbol{\xi} = \boldsymbol{\xi}_z$, whose components are set equal to the zeros of all orthonormal polynomials of first

degree used in the PCE. The error in this approximation is then given by

$$\boldsymbol{U}(\boldsymbol{\xi}_z) - \boldsymbol{U}^0 = \sum_{i=1}^{q_1} \boldsymbol{U}^i Y_i(\boldsymbol{\xi}_z) + \sum_{i=q_1+1}^{\infty} \boldsymbol{U}^i Y_i(\boldsymbol{\xi}_z) = \sum_{i=q_1+1}^{\infty} \boldsymbol{U}^i Y_i(\boldsymbol{\xi} = \boldsymbol{\xi}_z) \tag{3.16}$$

where q_1 is given by setting $C = 1$ in eq. 2.26.

Moreover, for $C=1$,

$$\mathcal{J}_{ij}^{\lambda\mu} = \sum_{\rho=0}^{q_1} \mathcal{J}_{ij}^{\rho} < Y_\rho, Y_\lambda, Y_\mu >= \delta_{\lambda\mu} \mathcal{J}_{ij}^{00} \tag{3.17}$$

since

$$< Y_\rho, Y_\lambda, Y_\mu >= \delta_{0\rho}\delta_{\lambda\mu} \ , \quad 1 \leq \lambda, \ \mu \leq q_1 \ , \quad 0 \leq \rho \leq q_1 \tag{3.18}$$

Thus, eq. 3.15 takes the form

$$\begin{bmatrix} \mathcal{J}_{ij}^{00} & \mathcal{J}^{01} & \mathcal{J}^{02} & \dots & \mathcal{J}^{0q_1} \\ \mathcal{J}^{10} & \mathcal{J}_{ij}^{00} & \mathbf{0} & \dots & \mathbf{0} \\ \mathcal{J}^{20} & \mathbf{0} & \mathcal{J}_{ij}^{00} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathcal{J}^{q_1 0} & \mathbf{0} & \mathbf{0} & \dots & \mathcal{J}_{ij}^{00} \end{bmatrix} \begin{bmatrix} \Delta\boldsymbol{U}^0 \\ \Delta\boldsymbol{U}^1 \\ \Delta\boldsymbol{U}^2 \\ \vdots \\ \Delta\boldsymbol{U}^{q_1} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}^0 \\ \boldsymbol{R}^1 \\ \boldsymbol{R}^2 \\ \vdots \\ \boldsymbol{R}^{q_1} \end{bmatrix} \tag{3.19}$$

Assuming that $\boldsymbol{U}^0$ is well approximated, it is deduced that $\Delta\boldsymbol{U}^0 \approx \mathbf{0}$, which justifies the decision to keep only the diagonal blocks of the coefficient matrix in eq. 3.19. The simplified system can be solved efficiently as it consists of q_1+1 linear systems with the same LHS and different RHS terms. Moreover, there is no need to compute $\mathcal{J}_{ij}^{00}$, as this can be approximated by $\mathcal{J}$ as computed during the last iteration of the solution of the PDEs without uncertainties, i.e. those yielding $\boldsymbol{U}^0$.

The aforementioned steps compute the coefficients of the PCE of the field variables up to $C=1$. If $C>1$, the solution algorithm is similar. With $q=q(C)$ as in eq. 2.26, let us assume that the PCE coefficients corresponding to the first $q(C-1)$ terms of the expansions are available (computed as described above). To compute the next $q(C)-q(C-1)$ terms, eq. 3.15 is used. In such a case, the off–diagonal blocks are not zero and the diagonal blocks have some more terms, in addition to $\mathcal{J}_{ij}^{00}$. However, the previously computed $\boldsymbol{U}^i, i \leq q(C-1)$ result in $\Delta\boldsymbol{U}^i \approx \mathbf{0}$, meaning that most of the off–diagonal blocks can be neglected. Moreover (no summation for k),

$$\mathcal{J}_{ij}^{kk} = \mathcal{J}_{ij}^{\rho} < Y_\rho, Y_k, Y_k >\approx \mathcal{J}_{ij}^{00} \tag{3.20}$$

Given eq. 3.18, the first non–zero terms are those corresponding to the PCE coefficients of second (or greater) chaos order and are, thus, expected to be negligible compared to $\mathcal{J}_{ij}^{00}$. Therefore, the LHS of the system is again similar to that of eq. 3.19, yielding $q+1$ decoupled systems with the same LHS. To sum up, the proposed way to solve the iPCE equations consists of the following steps:

1. Solve the PDEs without uncertainties, eq. 3.7, to approximate $\boldsymbol{U}^0$.

2. Store the LHS of eq. 3.8, if possible, or re-compute it once in the beginning of the solutions of the iPCE equations.

3. Solve the iPCE equations for $C = 1$, to find the corresponding terms of the expansions of the field variables.

4. (If the user–defined C is greater than 1) Solve the iPCE equations, for $C = 2$ using the PCE coefficient fields for $C = 1$ as initialization, while keeping the same LHS approximation as in the previous step and so on and so forth. The RHS terms are always recomputed.

Note that, in the procedure described above, there is no need to build the LHS term of the linearized system in each and every iteration, since this remains constant. This also allows for an important reduction in memory requirements, as the space required to store the LHS of the iPCE equations, eq. 3.7, is no greater than that of a problem without uncertainties, eq. 3.7. Moreover, given the decoupling of the equations, the cost of the solution step is proportional to $q+1$ and can even be reduced to the cost of $q+1$ matrix–vector multiplications, if a method that computes the inverse of $\mathcal{J}^{00}$ was used. It is expected that the proposed initialization for the mean flow field $\boldsymbol{U}^0$ will facilitate the convergence of the iPCE equations by reducing the number of required iterations. Finally, in some cases where $C > 1$, it might be possible to skip step 3 and solve the iPCE equations directly for the chosen C by keeping only the approximation of the diagonal blocks of the LHS of eq. 3.11.

3.5 Workflow of the Proposed Method

This section describes the programming needed to apply the proposed iPCE approach, as well as an algorithm to implement it.

Firstly, the method requires a way to choose quadrature nodes and weights that correspond to the chosen chaos order, number of uncertain variables and probability distributions. The choice must be such that it guarantees the exact evaluation of integrals of polynomials with *degree up to* $2C$. In this way, Galerkin projections are exact. Note that, this way of choosing nodes and weights is exactly the same in the niPCE approach, which also requires the same accuracy in the computation of integrals involving polynomials.

For a single variable, an integral computed with Gauss quadrature is exact for a polynomial of degree $2d - 1$ if the nodes used correspond to the zeros of Y_d. Then, the weights are given by

$$d_i = \frac{k_d}{k_{d-1}} \frac{< Y_{d-1}, Y_{d-1} >}{Y'_d(\xi_i) Y_{d-1}(\xi_i)}, \; i = 1, \ldots, d \tag{3.21}$$

where k_d is the coefficient of ξ^d in $Y_d(\xi)$. So, in this case

$$2d - 1 \geq 2C \Rightarrow d = C + 1$$

is the number of required nodes that correspond to values of the uncertain variable ξ, for which the function to be integrated has to be evaluated.

For known distributions and their orthogonal polynomials, the nodes and weights can be found in the literature and they can be hardcoded. In the Appendix C, the nodes and weights that correspond to the Hermite and the Legendre polynomials are given. For an arbitrary probability distribution, its orthogonal polynomials can be found through the Gram–Schmidt orthogonalization (see chapter 2), so it is again possible to find their roots and then their weights, via eq.3.21. Note that proposition 2.1.2 guarantees the existence of the zeros in the interval of orthogonality.

For more than one variables, either a tensorized (full) or a sparse grid can be used. In the first case $d = (C+1)^m$, as the required multivariate nodes and weights are a result of a tensor product of m sets of $C+1$ nodes and weights each. In other words, $C+1$ nodes and weights per dimension are used. The sparse grid case will not be examined here, although it was used in some of the applications later demonstrated in this thesis (Smolyak sparse grid [22]). Table 3.1 shows the number of nodes required, if a tensorized or a sparse Smolyak quadrature grid is used.

Tensorized Grid / Sparse Smolyak Grid						
C/m	1	2	3	4	5	6
1	2/3	4/5	8/7	16/9	32/11	64/13
2	3/5	9/13	27/25	81/41	243/61	729/85
3	4/9	16/29	64/69	256/137	1024/241	4096/389
4	5/17	25/65	125/177	625/401	3125/801	15625/1457
5	6/33	36/145	216/441	1296/1105	7776/2433	46656/4865

Table 3.1: *Number of GQ nodes for a full and a Smolyak quadrature grid. C is the chosen chaos order and m is the number of uncertain variables.*

The result of this first step should be an array containing all the quadrature weights, as well as an array of all the orthogonal polynomials up to degree C, evaluated at each quadrature node.

Before describing the second step, an algorithm will be presented that allows for the computation of a Galerkin projected vector or matrix. More specifically, d quadrature nodes are assumed, along with arrays w and Ksi with the corresponding quadrature weights and values of $\boldsymbol{\xi}$ at the quadrature nodes. Also, an array $Y(i,j)$ with the value of the $j-th$ orthonormal polynomial at the $i-th$ quadrature node is assumed.

For a vector $\boldsymbol{V} = \boldsymbol{V}(\boldsymbol{\xi})$ and a process named 'vectorcalc' that, for a given value of $\boldsymbol{\xi}$, returns the value of $\boldsymbol{V}$, $\mathrm{G}^{\mathrm{q}}[\boldsymbol{V}]$ is found by the following algorithm

Algorithm 1 Galerkin projected vector calculation

1: $GqV(1:n*q) \leftarrow 0$ / initialize $\mathrm{G}^{\mathrm{q}}[\boldsymbol{V}]$ to zero
2: for $i = 1 : d$ / loop of d quadrature nodes
3: call vectorcalc(Ksi(i),V) / get $\boldsymbol{V}(\boldsymbol{\xi})$
4: for $j = 0 : q$
5: $GqV(q*j+1:q*j+n) \leftarrow GqV(q*j+1:q*j+n) + w(i)*V(1:n)*Y(j,i)$
6: endfor
7: endfor

Also, for a matrix $A = A(\boldsymbol{\xi})$ and a process 'matrixcalc' that, for a given value of $\boldsymbol{\xi}$, returns the value of A,$\mathrm{G}^{\mathrm{q}}[A]$ is found by the following algorithm

Algorithm 2 Galerkin projected matrix calculation

1: $GqA(1:n*q\ ,\ 1:n*q) \leftarrow 0$!initialize $\mathrm{G}^{\mathrm{q}}[A]$ to zero
2: for $i = 1 : d$! loop of d quadrature nodes
3: call matrixcalc(Ksi(i),A) ! get $A(\boldsymbol{\xi})$
4: for $j1 = 0 : q$! loop of block rows $A^{j1\ j2}$
5: for $j2 = 0 : q$! loop of block columns
6: $k1 \leftarrow j1*q$
7: $k2 \leftarrow j2*q$
8: $A(k1+1:k1+n\ ,\ k2+1:k2+n) \leftarrow A(k1+1:k1+n\ ,\ k2+1:k2+n) +$
9: $w(i)*A(1:n\ ,\ 1:n)*Y(j1,i)*Y(j2,i)$
10: endfor
11: endfor
12: endfor

Moving on to the second step, the field variables $\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}]$ must be initialized. The mean values $\boldsymbol{U}^0$ are initialized with the solution of the problem without uncertainties. The other coefficients can be set equal to zero, or be initialized by applying Galerkin projections to the routine used in the deterministic code that is responsible for the initialization of $\boldsymbol{U}$. In this case, algorithm 1 is used, so as to find $\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}]$ by using the routine that gives $\boldsymbol{U}$ its initial values, in place of 'vectorcalc'.

Afterwards, the code is ready to start iterating i.e. forming the system of equations and solving it.

To form the system of equations, eq. 3.11, the LHS and RHS have to be found. The term $\mathrm{G}^{\mathrm{q}}[\boldsymbol{R}]$ can again be found by applying algorithm 1, through the use a routine that returns the residuals $\boldsymbol{R}$ of the deterministic equations, when given the value of $\boldsymbol{\xi}$ at a GQ node as input. The LHS can be either found in a similar way, by applying algorithm 2, or, in order to save memory and speed up convergence, it can be replaced by a stored LHS matrix, found previously when a deterministic problem was solved to initialize the equations (see previous section).

Finally, the formed system has to be solved. In order to solve the whole system, without using a constant LHS to save memory, the user is free to choose a solution

method. However, if eq.3.19 is used instead (by keeping only the diagonal blocks and approximating them by the stored LHS), it is recommended to consider inverting the LHS matrix once, at the beginning of the code, so as to reduce the solution step to some matrix and vector multiplications. If this is not possible, the exact same solution algorithm used in the deterministic case can also be applied here.

3.6 Comparison with the niPCE and Conventional iPCE

It is evident that the proposed iPCE method involves minimum changes in the original deterministic code. Most subroutines do not have to be rewritten at all, while other require small changes, usually to write some of their inputs as functions of $\boldsymbol{\xi}$. This is a huge advantage over the conventional iPCE, which involves problem–specific approaches and major changes in existing software, while offering a limited choice of number of uncertain variables and chaos order, usually asking for reprogramming when they need to change. Thus, in terms of *complexity and flexibility*, this method seems undoubtedly superior to the conventional iPCE. The niPCE is, however, still more straightforward to apply and is definitely the way to go, when the number of uncertain variables is relatively small.

In terms of *computational cost*, it will be later on shown how this method can vastly outperform the non–intrusive approach, especially in complex problems, such as turbulent flows with many uncertain variables. Compared to the conventional iPCE approach, it should be expected that this method could be somewhat slower. The conventional iPCE approach involves problem–specific 'tricks', that could possibly result in slightly faster algorithms. Especially in the computation of the residual terms, $G^q[\boldsymbol{R}]$, the proposed method is essentially non–intrusive and is thus expected to be slower. But this is only a part of the proposed algorithm and seems like a small price to pay, if the involved level of complexity is to drop significantly.

To sum up, the proposed iPCE enjoys the benefits of the niPCE, with the computational cost if the iPCE. It is essentially a combination of the merits of both methods, without their most notorious disadvantages, the curse of dimensionality and the complexity involved, respectively.

Chapter 4

iPCE Applications in Aerodynamics

In this chapter aerodynamic problems studied using the proposed/programmed method are presented in which the accuracy and speed of the method are tested and compared to that of the niPCE. The flow model used for these cases is first given. Note that in all applications the computations were carried out on a single core of a Xeon CPU (E5-2630 at 2.20GHz) with 25MB cache and 128GB RAM.

4.1 Flow Equations

The 3D Reynolds–Averaged Navier–Stokes (RANS) equations for compressible fluid flows, in vector form,

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{f}_i^{inv}}{\partial x_i} - \frac{\partial \boldsymbol{f}_i^{vis}}{\partial x_i} = \boldsymbol{0} \tag{4.1}$$

are solved, with $\boldsymbol{U} = (\rho, \rho\ \boldsymbol{u}, E_t)^T$ the flow variable vector, ρ the density, $\boldsymbol{u} = [u_1, u_2, u_3]$ the velocity vector, $E_t = \frac{p}{\gamma-1} - \frac{1}{2}\rho\ \boldsymbol{u}^2$ the total energy per unit volume and p the pressure. The inviscid and viscous fluxes are given by

$$\boldsymbol{f}_i^{inv} = \begin{pmatrix} \rho u_i \\ \rho u_i \boldsymbol{u} + p\boldsymbol{\delta}_i \\ u_i(E_t + p) \end{pmatrix}\ ,\quad \boldsymbol{f}_i^{vis} = \begin{pmatrix} 0 \\ \boldsymbol{\tau}_i \\ u_j\tau_{ij} + q_i \end{pmatrix} \tag{4.2}$$

where $\boldsymbol{\delta}_i$ is the Kronecker symbol, q_i the thermal flux components and $\boldsymbol{\tau}_i = [\tau_{i1}, \tau_{i2}, \tau_{i3}]^T$ are viscous and turbulent stresses. The inviscid fluxes can be expressed in terms of the Jacobian matrices as $\boldsymbol{f}_i^{inv} = A_i\boldsymbol{U}$, $A_i = \frac{\partial \boldsymbol{f}_i^{inv}}{\partial \boldsymbol{U}}$.

Closure is effected by the state equation of perfect gases and the Spalart–Allmaras (SA) one–equation turbulence model,[42]. The compressible Spalart–Allmaras tur-

bulence model solves the following equation

$$\frac{\partial(\rho\tilde{\mu})}{\partial t}+\frac{\partial(\rho u_i\tilde{\mu})}{\partial x_i} = \frac{1}{\sigma}\left[\frac{\partial}{\partial x_i}\left((\mu+\tilde{\mu})\frac{\partial\tilde{\mu}}{\partial x_i}\right)+c_{b_2}\left(\frac{\partial\tilde{\mu}}{\partial x_i}\right)^2\right]+c_{b_1}(1-f_{t_2})\tilde{S}\rho\tilde{\mu}$$
$$- (c_{w_1}f_w-\frac{c_{b_1}}{\kappa^2}f_{t_2})\left(\frac{\tilde{\mu}}{d}\right)^2+\rho^2 f_{t_1}\Delta u^2 \quad (4.3)$$

where

$$\tilde{S}=|\boldsymbol{\omega}|+\frac{\tilde{\mu}}{y^2\kappa^2}f_{v_2},\quad f_{v_2}=1-\frac{\chi}{1+\chi f_{v_1}}$$
$$f_w=g\left(\frac{1+c_{w_3}^6}{g^6+c_{w_3}^6}\right)^{1/6},\quad g=r+c_{w_2}(r^6-r)\ ,\quad r=\frac{\tilde{\mu}}{\tilde{S}\rho\kappa^2y^2}$$

and $\boldsymbol{\omega}=\nabla\times\boldsymbol{u}$, μ is the fluid's dynamic viscosity, while y is the distance of a grid node from the closest solid wall. Eq. 4.3 is solved for $\tilde{\mu}$ and the eddy viscosity is then found from

$$\mu_t=\tilde{\mu}f_{v_1},\quad f_{v_1}=\frac{\chi^3}{\chi^3+c_{v_1}^3},\quad \chi=\frac{\tilde{\mu}}{\mu} \quad (4.4)$$

Regarding the transition coefficient (the last term of eq. 4.3)

$$f_{t_1}=c_{t_1}g_t\ exp\left(-c_{t_2}\frac{\omega_t^2}{\Delta u^2}[y^2+g_t^2y_t^2]\right),\quad g_t=min(0.1,\frac{\Delta u}{\omega_t}\Delta x) \quad (4.5)$$

where y_t denotes the distance of a point in the flow field from the transition point, which is located somewhere along the solid wall,ω_t is the vorticity at that point, Δu denotes the difference between the velocity of any point in the flow field and the transition point and Δx is the grid spacing along the wall at this point. The imposed inlet boundary conditions are of Dirichlet type, as $\tilde{\mu}$ has a fixed free–stream value there. At the solid walls, $\tilde{\mu}=0$ is imposed. Finally, the constants of the model have the following values

$$\sigma=\tfrac{2}{3},\quad \kappa=0.41,\quad Pr_t=0.9,\quad c_{v_1}=7.1,\quad c_{b_1}=0.1355,\quad c_{b_2}=0.622,$$
$$c_{w_1}=\tfrac{c_{b_1}}{\kappa^2}+\tfrac{1+c_{b_2}}{\sigma},\quad c_{w_2}=0.3\ ,\ c_{w_3}=2,\quad c_{t_1}=1,\quad c_{t_2}=2,\quad c_{t_3}=1.1,\quad c_{t_4}=2$$

The equations are solved on unstructured grids (in 2D, this comprises triangles and layers of quadrilaterals close to solid walls, whereas in 3D this comprises of pyramids, prisms and six–sided solids) using the finite-volume method and vertex-centered storage, with the application of the flux–vector splitting (FVS) upwind scheme, [43], while the linear systems that arise from the discretization are solved by point–implicit (involving internal sub–iterations) Jacobi solver.

4.2 Flow Around an Aircraft Model

The first application is considered with the inviscid flow around an aircraft model. Due to symmetric flow conditions the study is carried around half of the aircraft. The QoI is the lift coefficient, the mean value and standard deviation of which will be computed using the proposed iPCE approach, the niPCE and the Monte–Carlo sampling. In this application, the computational cost is not discussed, since all 'other' methods just serve to validate the accuracy of the proposed approach, and whether these are more/very expensive is not an issue at this point.

Uncertainties were introduced in the free–stream flow angle a_∞ and/or the free–stream Mach number M_∞. Four cases were studied and results are presented in table 4.1, where $N(\mu, \sigma)$ denotes the normal distribution with mean value μ and standard deviation σ. Also, $\mathcal{U}(a, b)$ denotes the uniform distribution in the interval $[a, b]$.

Flow Conditions		iPCE Chaos order $C=1$	niPCE Chaos order $C=1$	iPCE Chaos order $C=2$	niPCE Chaos order $C=2$	MC
$M_\infty = 0.7$	μ_{C_L}	0.1192	0.1191	0.1192	0.1191	0.1188
$a_\infty \sim N(5^o, 0.5^o)$	σ_{C_L}	0.00958	0.00966	0.00958	0.00964	0.00953
$M_\infty \sim N(0.7, 0.02)$	μ_{C_L}	0.1195	0.1193	0.1195	0.1193	–
$a_\infty = 5^o$	σ_{C_L}	0.00179	0.00189	0.00192	0.00194	–
$M_\infty \sim N(0.7, 0.02)$	μ_{C_L}	0.1193	0.1192	0.1194	0.1193	0.1191
$a_\infty \sim N(5^o, 0.5^o)$	σ_{C_L}	0.00932	0.00985	0.00959	0.00985	0.00971
$M_\infty \sim N(0.7, 0.02)$	μ_{C_L}	0.1193	–	0.1194	–	0.1191
$a_\infty \sim \mathcal{U}(4.5^o, 5.5^o)$	σ_{C_L}	0.0107	–	0.0110	–	0.0112

Table 4.1: *UQ for the flow around an aircraft model. Statistical moments of the lift coefficient values computed using iPCE, niPCE (with $C=1$ and $C=2$) and the Monte–Carlo method with 2000 replicates in each case.*

After the solution of the corresponding iPCE equations, the fields of the PCE coefficients of the flow variables are available. This allows for the computation of any quantity, such as the Mach number field around the aircraft. Figure 4.1 compares the Mach number's mean and standard deviation fields respectively, in the case uncertainty is only due to the Mach number ($a_\infty = 5^o$; second case in Table 4.1). It can be seen that the iPCE and niPCE results perfectly match each other all over the aircraft surface. One can also notice the increased variance after the supersonic area of the wing surface which, in the case of niPCE, is extended over a greater area along the wing. The pressure spectral coefficient fields are shown in fig. 4.2.

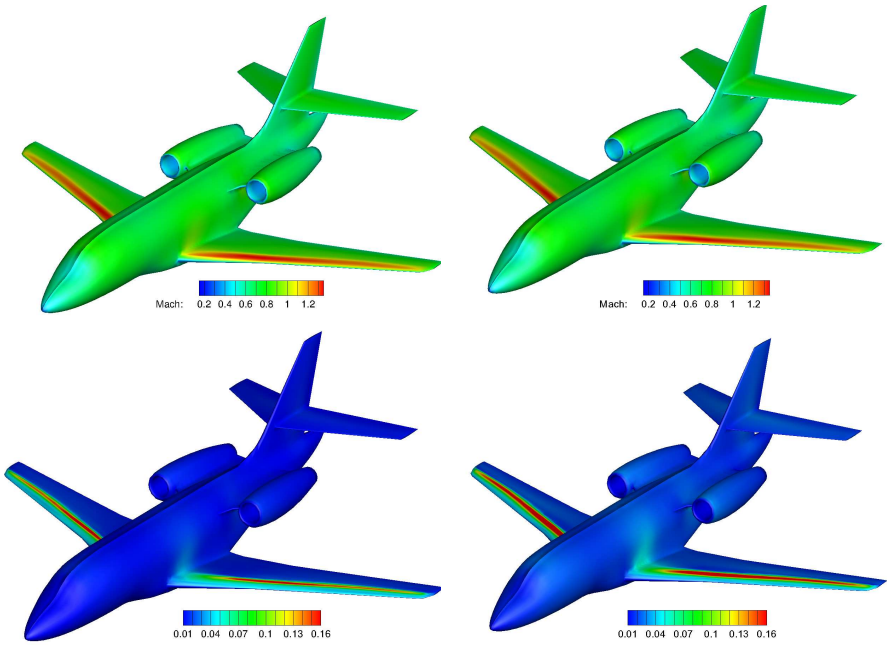


Figure 4.1: *UQ for the flow around an aircraft model ($M_\infty \sim N(0.7, 0.02)$, $a_\infty = 5^o$). Mean Mach number distribution (top) and standard deviation (bottom) on the aircraft surface, computed using the iPCE (left) and niPCE (right), with C=2.*

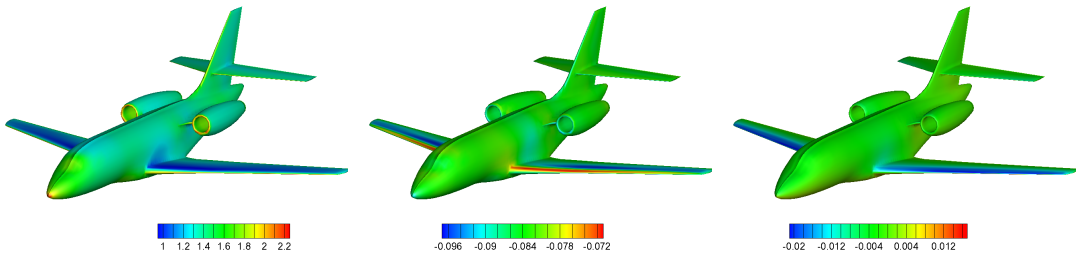


Figure 4.2: *UQ for the flow around an aircraft model ($M_\infty \sim \mathcal{N}(0.7, 0.02), a_\infty \sim \mathcal{U}(4.5^o, 5.5^o)$. Computed pressure spectral coefficients (iPCE, C=1). Mean pressure value (left), spectral coefficient corresponding to free–stream Mach number (middle) and spectral coefficient corresponding to free–stream flow angle (right).*

4.3 Isolated Airfoil Case

The second case deals with the turbulent flow around a 2D isolated airfoil. The two QoIs are the lift and drag coefficients and uncertainty was introduced in the flow conditions, namely a_∞, M_∞ and Re, yielding three uncertain variables in total. The chosen PDFs of the flow conditions are

$$a_\infty \sim \mathcal{U}(1.5^o, 2.5^o) \qquad M_\infty \sim N(0.3, 0.01) \qquad Re \sim N(10^6, 2.5 \cdot 10^4)$$

where it can be seen that different probability distributions can be used, not just the normal distribution.

The proposed iPCE is compared with the niPCE method and results are summarized in table 4.2, where the CPU time unit is defined as the computational cost of the iPCE method for $C = 1$. It is evident that the two methods are in good agreement, but the iPCE is significantly faster, which is explained below.

The iPCE method required about 1000 iterations to converge, which is small compared to the 4800 iterations that the CFD runs of the niPCE asked on average. This should be attributed to the simultaneously solved turbulence model PDE, which may introduce convergence difficulties when the uncertain flow conditions vary and the software should run for several, quite off–design values of them (in the niPCE). In contrast, in the iPCE method, the initialization is fairly close to the expected solution, which facilitates convergence. Note that, for a fair comparison, the non–intrusive CFD runs for $C = 2$ and $C = 3$, were initialized with the same mean field values as in the intrusive case, without though yielding any significant difference.

The mean Mach number field and its standard deviation that resulted from the iPCE are depicted in fig.4.3, for $C = 1$. It is interesting to notice that the standard deviation is higher in areas where the mean Mach number is high as well. Also, the pressure spectral coefficient fields are shown in fig. 4.4.

	iPCE	niPCE	iPCE	niPCE	iPCE	niPCE
	$C=1$		$C=2$		$C=3$	
μ_{C_L}	0.095598	0.095567	0.095591	0.095600	0.095611	0.095598
σ_{C_L}	0.013534	0.013546	0.013535	0.013438	0.013535	0.013512
μ_{C_D}	0.029460	0.029540	0.029426	0.029538	0.029319	0.029539
σ_{C_D}	0.000787	0.000764	0.000789	0.000768	0.000790	0.000768
CPU time units	*1*	*3.678*	*2.933*	*9.598*	*20.196*	*36.714*

Table 4.2: *Turbulent flow around an isolated airfoil, with three uncertain flow conditions. Statistical moments of C_L and C_D computed using iPCE and niPCE, for $C=1,2,3$ and computational cost.*

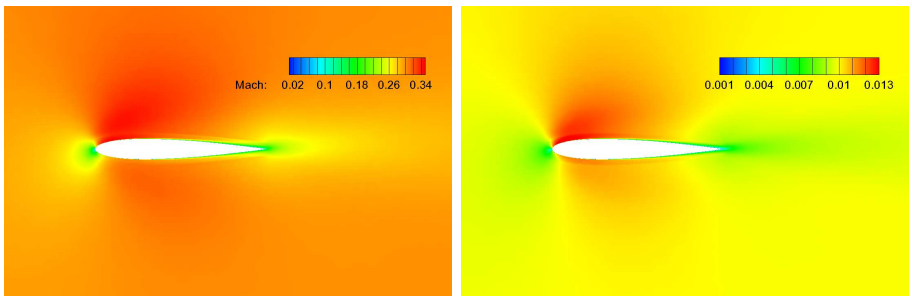


Figure 4.3: *Turbulent flow around an isolated airfoil, with uncertain flow conditions. Computed mean (left) and standard deviation (right) fields of the Mach number (iPCE, C=1).*

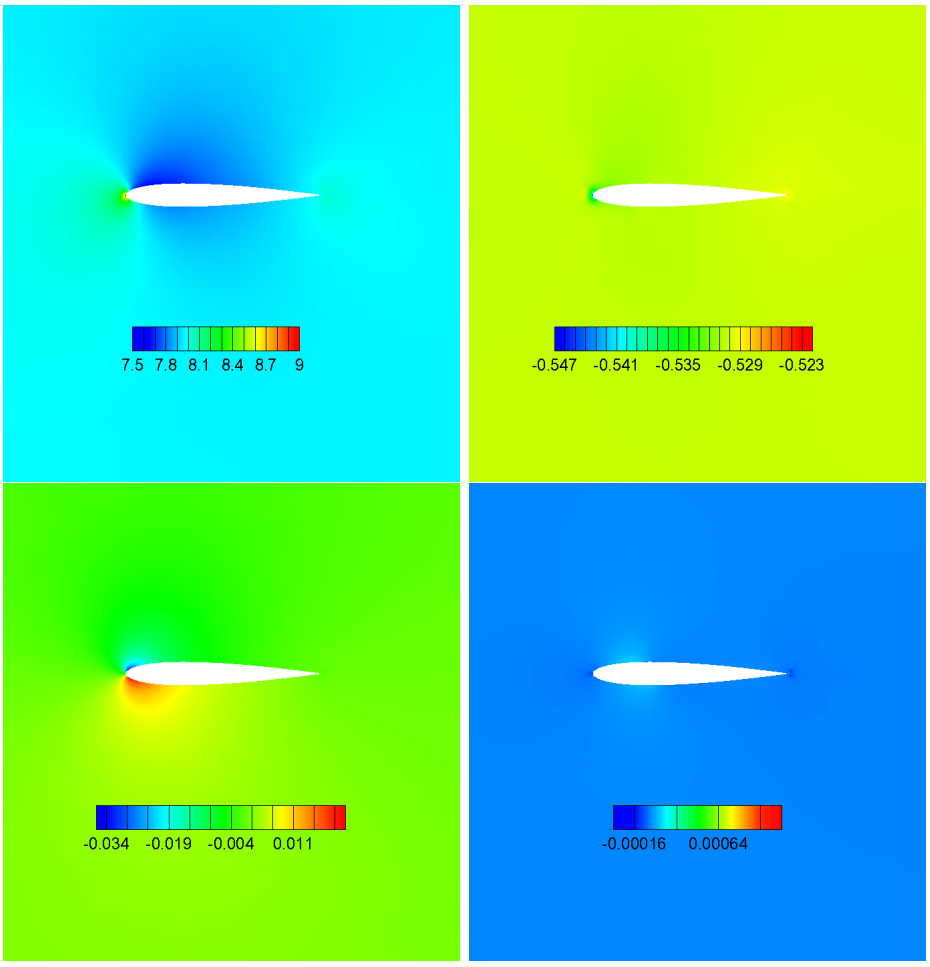


Figure 4.4: *Turbulent flow around an isolated airfoil, with uncertain flow conditions. Computed pressure spectral coefficients (iPCE, C=1). Mean pressure value (top left), spectral coefficient corresponding to free–stream Mach number (top right), spectral coefficient corresponding to free–stream flow angle (bottom left) and spectral coefficient corresponding to chord–based Reynolds number (bottom right)*

4.4 Flow around the DLR–F6 Aircraft

The last case is concerned with the inviscid transonic flow around an aircraft configuration (practically the wing–fuselage configuration of DLR–F6 of [44] though this is herein studied at inviscid flow conditions). The computational grid consists of about 1.5M nodes; a grid around the whole aircraft was used since one of the uncertain variables was the yaw angle.

Uncertainties are introduced in the flow conditions, as follows

$$a_\infty \sim \mathcal{U}(-0.5^o, +0.5^o) \ , \ \beta_\infty \sim \mathcal{U}(-0.5^o, +0.5^o) \text{ and } M_\infty \sim N(0.75, 0.02)$$

and the QoI is the lift coefficient.

Results are summarized in table 4.3, where it is again shown that the iPCE and the niPCE produce practically identical results. The iPCE method is, however, more efficient and the difference in cost increases with the chaos order. This should be attributed to the fact that the proposed method saves computational time in the solution step, recall the discussion in section 3.4. In this case, the solution step is the most costly one, since the computational grid is much larger than in the two previously presented cases. Therefore, the gain in computational time is greater.

Note that, for a fair comparison, the niPCE runs were initialized with the same flow field used to initialize $\boldsymbol{U}^0$ in the iPCE (which resulted from solving the equations without uncertainties once), which seemed to facilitate convergence, in either case. The convergence of the iPCE solver is, however, smoother and faster, see fig. 4.7 which shows the convergence of the PCE coefficients of the QoI as a function of computational time. In these plots, the wall clock time in thousands of seconds is used as a time unit.

Finally, fig.4.5 illustrates the mean value and the standard deviation of the Mach number over the aircraft surface, while fig. 4.6 shows the spectral coefficients of the pressure field.

	iPCE	niPCE	iPCE	niPCE	iPCE	niPCE
	$C=1$		$C=2$		$C=3$	
μ_{C_L}	0.10115	0.10115	0.10115	0.10115	0.10115	0.10115
σ_{C_L}	0.007603	0.007618	0.007653	007653	0.007650	0.007650
CPU time units	*1*	*1.248*	*3.121*	*6.695*	*6.050*	*15.603*

Table 4.3: *Inviscid flow around the DLR–F6 aircraft, with three uncertain flow conditions. Statistical moments of the lift coefficient computed using the iPCE and the niPCE (for $C=1,2,3$) and computational cost.*

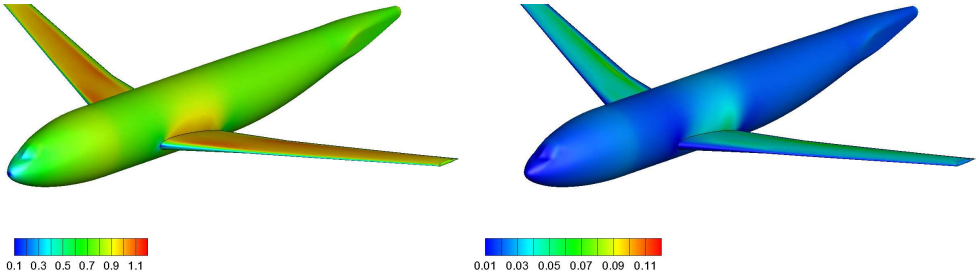


Figure 4.5: *Inviscid flow around the DLR–F6 aircraft, with three uncertain flow conditions. Mean field (left) and standard deviation field (right) of the Mach number (iPCE, $C=1$) over the aircraft surface.*

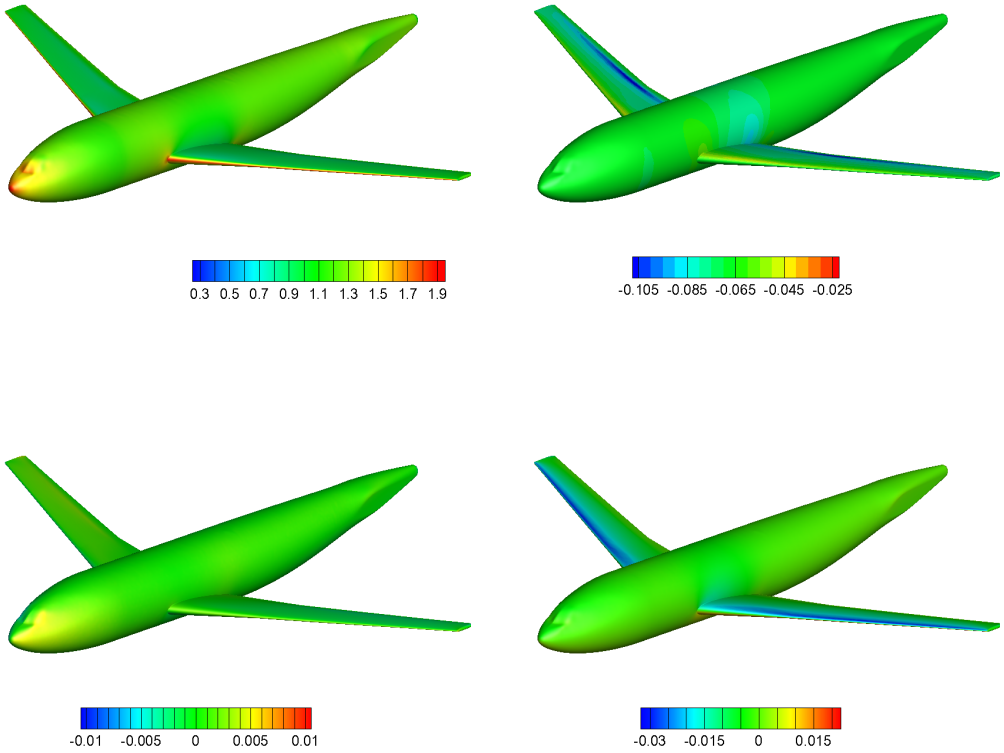


Figure 4.6: *Inviscid flow around the DLR–F6 aircraft, with three uncertain flow conditions. Computed pressure spectral coefficients (iPCE, $C = 1$). Mean pressure value (top left), spectral coefficient corresponding to free–stream Mach number (top right), spectral coefficient corresponding to free–stream flow angle (bottom left) and spectral coefficient corresponding to free–stream yaw angle (bottom right)*

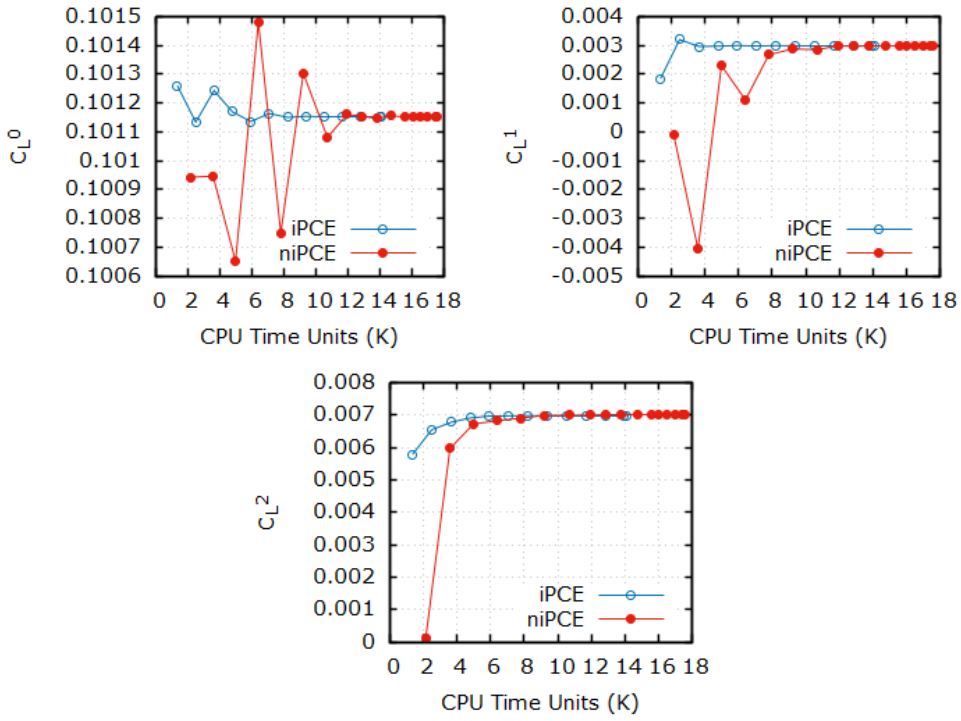


Figure 4.7: *Inviscid flow around the DLR–F6 aircraft, with three uncertain flow conditions. Comparison of the convergence of the PCE coefficients of C_L (iPCE and niPCE, $C=1$). Mean value of the lift coefficient C_L^0 (top – left), PCE coefficient C_L^1 corresponding to the free–stream Mach number (top – right) and PCE coefficient C_L^2 corresponding to the free–stream pitch angle (bottom). All runs were initialized with the converged mean flow field, i.e. the solution of the PDEs without uncertainties.*

Chapter 5

Continuous Adjoint of the iPCE

This chapter proposes a way to derive the continuous adjoint equations to the primal iPCE problem. Emphasis is laid on establishing a general method that is easy to program, with just a reasonable amount of interventions in an existing adjoint code for the problem without uncertainties. Before the continuous adjoint of the iPCE equations is developed, a brief reference to the deterministic continuous adjoint of a general set of PDEs is made, following the development found in [45]. Note that the approach presented in this chapter is the derivation of the adjoint equations to the primal iPCE equations. The opposite would be to apply the iPCE theory to the deterministic adjoint equations, which is not discussed in this diploma thesis.

5.1 Deterministic Continuous Adjoint

Let us consider a QoI given by

$$F = \int_{\Omega} \boldsymbol{g}^T \boldsymbol{U} d\Omega + \int_{S} \boldsymbol{h}^T C \boldsymbol{U} dS \tag{5.1}$$

$\boldsymbol{U}$ stands for the field variables, that are subject to the following equations

$$\begin{aligned} L\boldsymbol{U} - \boldsymbol{f} &= \boldsymbol{0} \ , \text{ in } \Omega \\ B\boldsymbol{U} - \boldsymbol{e} &= \boldsymbol{0} \ , \text{ in } S \end{aligned} \tag{5.2}$$

where $S \equiv \partial\Omega$ is the boundary of Ω, L,B and C are linear differential operators while $\boldsymbol{f}, \boldsymbol{e}, \boldsymbol{g}$ and $\boldsymbol{h}$ may depend on the spatial coordinates x_i but not on the field variables $\boldsymbol{U}$. Also, let $\boldsymbol{b}$ denote the array of design variables, $\delta \equiv \delta/\delta\boldsymbol{b}$ stands for the total derivative with respect to $\boldsymbol{b}$ and $\partial \equiv \partial/\partial\boldsymbol{b}$ is the partial derivative due to changes in the design variables while neglecting space deformations. Note that the volume integral of eq.5.2 does not include differential operators; if this were the case, such terms would become surface integrals, through Gauss' theorem.
In order to minimize (or maximize) F under the constraints posed by eq.5.2, the

following augmented function is defined

$$F_{aug} := F - \int_{\Omega} \boldsymbol{\Psi}^T (L\boldsymbol{U} - \boldsymbol{f}) d\Omega - \int_S (C^* \boldsymbol{\Psi})^T (B\boldsymbol{U} - \boldsymbol{e}) dS \tag{5.3}$$

where $\boldsymbol{\Psi}$ are the adjoint variables and the differential operator C^* will be defined later.
In general, we can write that

$$\delta(L\boldsymbol{U}) = L(\delta \boldsymbol{U}) + (\delta L)\boldsymbol{U} \tag{5.4}$$

For instance, if $L \equiv \frac{\partial}{\partial x_i}$, then $\delta(L\boldsymbol{U}) = \frac{\partial(\delta \boldsymbol{U})}{\partial x_i} + \frac{\partial \boldsymbol{U}}{\partial x_k} \frac{\partial(\delta x_k)}{\partial x_i}$, i.e. $\delta L = \frac{\partial(\delta x_k)}{\partial x_i} \frac{\partial}{\partial x_k}$.

Thus, because of eq.5.4, the total derivative of F_{aug} is

$$\delta F_{aug} = \delta F + \delta F_{SD}^{\Psi} - \int_{\Omega} \boldsymbol{\Psi}^T L(\delta \boldsymbol{U}) d\Omega - \int_S (C^* \boldsymbol{\Psi})^T B(\delta \boldsymbol{U}) dS \tag{5.5}$$

where

$$\delta F_{SD}^{\Psi} := \int_{\Omega} \boldsymbol{\Psi}^T [\delta \boldsymbol{f} - (\delta L)\boldsymbol{U}] d\Omega + \int_S (C^* \boldsymbol{\Psi})^T [\delta\ \boldsymbol{e} - (\delta B)\boldsymbol{U}] dS \tag{5.6}$$

Also, δF is given by

$$\delta F = \delta F_{SD} + \int_S \boldsymbol{h}^T C(\delta \boldsymbol{U}) dS + \int_{\Omega} \boldsymbol{g}^T \delta \boldsymbol{U} d\Omega \tag{5.7}$$

where

$$\begin{aligned} \delta F_{SD} := & \int_{\Omega} \delta \boldsymbol{g}^T \boldsymbol{U} d\Omega + \int_{\Omega} \boldsymbol{g}^T \boldsymbol{U} \delta(d\Omega) \\ & + \int_S \delta\ \boldsymbol{h}^T C \boldsymbol{U} dS + \int_S \boldsymbol{h}^T (\delta C) \boldsymbol{U} dS + \int_S \boldsymbol{h}^T C \boldsymbol{U} \delta(dS) \end{aligned} \tag{5.8}$$

In eqs. 5.6 and 5.8 'SD' stands for 'sensitivity derivatives'. The calculation of the variations present in eq. 5.8 depends on the parameterization of the problem and on the choice of design variables.

Then we apply integration by parts, as follows

$$\int_{\Omega} \boldsymbol{\Psi}^T L(\delta \boldsymbol{U}) d\Omega \equiv \int_{\Omega} (A\boldsymbol{\Psi})^T \delta \boldsymbol{U} d\Omega + \int_S (D\boldsymbol{\Psi})^T E(\delta \boldsymbol{U}) dS \tag{5.9}$$

In eq. 5.9, the differential operators A, D and E are known; they are essentially

defined so that eq. 5.9 holds. By inserting eq. 5.9 into eq. 5.5, we get

$$\begin{aligned}\delta F = \delta F_{SD} + \delta F_{SD}^{\Psi} + \int_{\Omega} (\boldsymbol{g} - A\boldsymbol{\Psi})^T \delta \boldsymbol{U} d\Omega + \int_{S} (\boldsymbol{h} - B^*\boldsymbol{\Psi})^T C(\delta \boldsymbol{U}) dS \\ + \underbrace{\int_{S} (B^*\boldsymbol{\Psi})^T C(\delta \boldsymbol{U}) dS - \int_{S} (D\boldsymbol{\Psi})^T E(\delta \boldsymbol{U}) dS - \int_{S} (C^*\boldsymbol{\Psi})^T B(\delta \boldsymbol{U}) dS}_{=:M}\end{aligned} \tag{5.10}$$

Notice that to derive eq. 5.10 the term $\int_S (B^*\boldsymbol{\Psi})^T C(\delta \boldsymbol{U}) dS$ was added and substracted from eq. 5.9, while B^* is a newly introduced differential operator that will be defined later.

If $\boldsymbol{\Psi}$ is chosen so that it satisfies the adjoint equation

$$\begin{aligned} A\boldsymbol{\Psi} - \boldsymbol{g} &= \boldsymbol{0} \ , \text{ in } \Omega \\ B^*\boldsymbol{\Psi} - \boldsymbol{h} &= \boldsymbol{0} \ , \text{ in } S \end{aligned} \tag{5.11}$$

and the operators B^* and C^* are such that M is zero, as shown in [45], then the variation of F will be given by

$$\delta F = \delta F_{SD} + \delta F_{SD}^{\Psi} \tag{5.12}$$

which is computed after the adjoint equations, eq. 5.11, are solved.

5.2 Continuous Adjoint iPCE Problem

In the presence of uncertainties, the QoI to be minimized/maximized is defined as

$$J := \sum_{i=0}^{q} \zeta_i |F^i| \tag{5.13}$$

where ζ_i are user–defined coefficients, q corresponds to a chosen chaos order and F^i are the PCE coefficients of F, which is given by eq. 5.1. The PCE coefficients of the field variables $\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}]$ are subject to the iPCE equations, namely

$$\begin{aligned} \mathrm{G}^{\mathrm{q}}[L\boldsymbol{U} - \boldsymbol{f}] &= \boldsymbol{0} \ , \text{ in } \Omega \\ \mathrm{G}^{\mathrm{q}}[B\boldsymbol{U} - \boldsymbol{e}] &= \boldsymbol{0} \ , \text{ in } S \end{aligned} \tag{5.14}$$

In this case, the augmented function is

$$J_{aug} := J - \int_{\Omega} \mathrm{G}^{\mathrm{q}}[\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[L\boldsymbol{U} - \boldsymbol{f}] \, d\Omega - \int_{S} \mathrm{G}^{\mathrm{q}}[C^*\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[B\boldsymbol{U} - \boldsymbol{e}] \, dS \tag{5.15}$$

Since operators δ and $\mathrm{G}^{\mathrm{q}}[]$ permute, the total derivative of J_{aug} is

$$\begin{aligned}\delta J_{aug} &= \delta J - \int_{\Omega} \mathrm{G}^{\mathrm{q}}[\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[L(\delta \boldsymbol{U})]\, d\Omega - \int_S \mathrm{G}^{\mathrm{q}}[C^*\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[B(\delta \boldsymbol{U})]\, dS \\ &+ \int_{\Omega} \mathrm{G}^{\mathrm{q}}[\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[\delta \boldsymbol{f} - (\delta L)\boldsymbol{U}]\, d\Omega + \int_S \mathrm{G}^{\mathrm{q}}[C^*\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[\delta\, \boldsymbol{e} - (\delta B)\boldsymbol{U}]\, dS\end{aligned} \tag{5.16}$$

where, if by defining

$$\zeta := \sum_{i=0}^{q} \zeta_i sign(F^i) Y_i(\boldsymbol{\xi}) \Rightarrow \mathrm{G}^{\mathrm{q}}[\zeta] = [\zeta_0 sign(F^0), \ldots, \zeta_q sign(F^q)] \tag{5.17}$$

the derivative of J is given by

$$\begin{aligned}\delta J &= \sum_{i=0}^{q} \zeta_i sign(F^i)\delta F^i = \mathrm{G}^{\mathrm{q}}[\zeta]^T \mathrm{G}^{\mathrm{q}}[\delta F] \\ &= \mathrm{G}^{\mathrm{q}}[\zeta]^T \left(\mathrm{G}^{\mathrm{q}}[\delta F_{SD}] + \int_S \mathrm{G}^{\mathrm{q}}\left[\, \boldsymbol{h}^T C(\delta \boldsymbol{U})\right] dS + \int_{\Omega} \mathrm{G}^{\mathrm{q}}\left[\, \boldsymbol{g}^T \delta \boldsymbol{U}\right] d\Omega \right) \\ &= \mathrm{G}^{\mathrm{q}}[\zeta]^T \mathrm{G}^{\mathrm{q}}[\delta F_{SD}] + \int_S \mathrm{G}^{\mathrm{q}}[\zeta\ \boldsymbol{h}]^T \mathrm{G}^{\mathrm{q}}[C(\delta \boldsymbol{U})]\, dS + \int_{\Omega} \mathrm{G}^{\mathrm{q}}[\zeta \boldsymbol{g}]^T \mathrm{G}^{\mathrm{q}}[\delta \boldsymbol{U}]\, d\Omega\end{aligned} \tag{5.18}$$

because of eq. 5.7 and proposition 3.6.
In this diploma thesis, uncertainties in the domain Ω are not examined (i.e. shape uncertainties of an airfoil, for instance). This allows for the permutation of integrals in Ω or S with the operator $\mathrm{G}^{\mathrm{q}}[]$. Therefore, because of eq. 5.6 and proposition 3.6, the last two terms of eq. 5.16 can be written as

$$\begin{aligned}&\int_{\Omega} \mathrm{G}^{\mathrm{q}}[\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[\delta \boldsymbol{f} - (\delta L)\boldsymbol{U}]\, d\Omega + \int_S \mathrm{G}^{\mathrm{q}}[C^*\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[\delta\, \boldsymbol{e} - (\delta B)\boldsymbol{U}]\, dS = \\ &\mathrm{G}^{\mathrm{q}}\left[\int_{\Omega} \boldsymbol{\Psi}^T[\delta \boldsymbol{f} - (\delta L)\boldsymbol{U}] d\Omega + \int_S (C^*\boldsymbol{\Psi})^T[\delta\, \boldsymbol{e} - (\delta B)\boldsymbol{U}] dS\right]^T \mathrm{G}^{\mathrm{q}}[1] = \\ &\mathrm{G}^{\mathrm{q}}\left[\delta F_{SD}^{\Psi}\right]^T \mathrm{G}^{\mathrm{q}}[1]\end{aligned} \tag{5.19}$$

where $\mathrm{G}^{\mathrm{q}}[1] = [1, 0, \ldots, 0]^T$.

By applying integration by parts in the second term of eq. 5.16, because of eq. 5.9 and proposition 3.5, we get

$$\begin{aligned}&\int_{\Omega} \mathrm{G}^{\mathrm{q}}[\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[L(\delta \boldsymbol{U})]\, d\Omega \equiv \\ &\int_{\Omega} \mathrm{G}^{\mathrm{q}}[A\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[\delta \boldsymbol{U}]\, d\Omega + \int_S \mathrm{G}^{\mathrm{q}}[D\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}}[E(\delta \boldsymbol{U})]\, dS\end{aligned} \tag{5.20}$$

Eq. 5.16, because of eqs. 5.19 and eq. 5.20, is now written as

$$\begin{aligned}\delta J_{aug} &= \mathrm{G}^{\mathrm{q}}\left[\zeta\right]^T \mathrm{G}^{\mathrm{q}}\left[\delta F_{SD}\right] + \mathrm{G}^{\mathrm{q}}\left[\delta F_{SD}^{\Psi}\right]^T \mathrm{G}^{\mathrm{q}}\left[1\right] + \int_{\Omega} \mathrm{G}^{\mathrm{q}}\left[\zeta \boldsymbol{g} - A\boldsymbol{\Psi}\right]^T \mathrm{G}^{\mathrm{q}}\left[\delta \boldsymbol{U}\right] d\Omega \\ &+ \int_S \mathrm{G}^{\mathrm{q}}\left[\zeta\ \boldsymbol{h} - B^*\boldsymbol{\Psi}\right]^T \mathrm{G}^{\mathrm{q}}\left[C(\delta \boldsymbol{U})\right] dS + \int_S \mathrm{G}^{\mathrm{q}}\left[B^*\boldsymbol{\Psi}\right]^T \mathrm{G}^{\mathrm{q}}\left[C(\delta \boldsymbol{U})\right] dS \\ &- \int_S \mathrm{G}^{\mathrm{q}}\left[C^*\boldsymbol{\Psi}\right]^T \mathrm{G}^{\mathrm{q}}\left[B(\delta \boldsymbol{U})\right] dS - \int_S \mathrm{G}^{\mathrm{q}}\left[D\boldsymbol{\Psi}\right]^T \mathrm{G}^{\mathrm{q}}\left[E(\delta \boldsymbol{U})\right] dS\end{aligned} \tag{5.21}$$

This means that the adjoint iPCE equation and boundary conditions are

$$\begin{aligned}\mathrm{G}^{\mathrm{q}}\left[A\boldsymbol{\Psi} - \zeta\ \boldsymbol{g}\right] &= \mathbf{0}\ ,\ \text{in } \Omega \\ \mathrm{G}^{\mathrm{q}}\left[B^*\boldsymbol{\Psi} - \zeta\ \boldsymbol{h}\right] &= \mathbf{0}\ ,\ \text{in } S\end{aligned} \tag{5.22}$$

Also, it can be shown that the last three terms of eq. 5.21 are equal to $\mathrm{G}^{\mathrm{q}}\left[M\right]^T \mathrm{G}^{\mathrm{q}}\left[1\right]$ (the proof is similar to eq. 5.19 and is omitted in the interest of space). This means that setting $\mathrm{G}^{\mathrm{q}}\left[M\right] = \mathbf{0}$, or $M = 0$, defines operators B^* and C^* so that they are equal to their counterparts in the case without uncertainties. Finally, the derivatives of J are found from

$$\delta J = \mathrm{G}^{\mathrm{q}}\left[\zeta\right]^T \mathrm{G}^{\mathrm{q}}\left[\delta F_{SD}\right] + \mathrm{G}^{\mathrm{q}}\left[\delta F_{SD}^{\Psi}\right]^T \mathrm{G}^{\mathrm{q}}\left[1\right] \tag{5.23}$$

5.2.1 How the Continuous Adjoint iPCE was Programmed

The continuous adjoint equations, boundary conditions, objective functions and sensitivity derivatives for both the deterministic case and the iPCE case are summarized

- Continuous Adjoint Equation

$$A\boldsymbol{\Psi} - \boldsymbol{g} = \mathbf{0} \;\; \text{(deterministic)} \;\; , \;\; \mathrm{G^q}\left[A\boldsymbol{\Psi} - \zeta\, \boldsymbol{g}\right] = \mathbf{0} \;\; \text{(iPCE)}$$

- Adjoint Boundary Conditions

$$B^*\boldsymbol{\Psi} - \boldsymbol{h} = \mathbf{0} \;\; \text{(deterministic)} \;\; , \;\; \mathrm{G^q}\left[B^*\boldsymbol{\Psi} - \zeta\, \boldsymbol{h}\right] = \mathbf{0} \;\; \text{(iPCE)}$$

  where $\zeta = \sum_{j=0}^{q} \zeta_j sign(F^j) Y_j(\boldsymbol{\xi})$

- Objective Function

$$F \;\; \text{(deterministic)} \;\; , \;\; J = \sum_{j=0}^{q} \zeta_j |F^j| \;\; \text{(iPCE)}$$

- Sensitivity Derivatives

$$\delta F = \delta F_{SD} + \delta F_{SD}^{\Psi} \;\; \text{(deterministic)}$$

$$\delta J = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\delta F_{SD}\right] + \mathrm{G^q}\left[\delta F_{SD}^{\Psi}\right]^T \mathrm{G^q}\left[1\right] \;\; \text{(iPCE)}$$

  where $\mathrm{G^q}\left[1\right] = [1, 0, \ldots, 0]$

The numerical solution of the adjoint iPCE equation is similar to that of the primal problem. An iterative scheme is chosen again, namely

$$\mathrm{G^q}\left[\frac{\partial \boldsymbol{R}^{adj}}{\partial \boldsymbol{\Psi}}\right] \mathrm{G^q}\left[\Delta\boldsymbol{\Psi}\right] = -\,\mathrm{G^q}\left[\boldsymbol{R}^{adj}\right] \tag{5.24}$$

where $\boldsymbol{R}^{adj} = A\boldsymbol{\Psi} - \zeta\, \boldsymbol{g}$ which is again solved by keeping only the diagonal blocks of the LHS, to save memory. Again, a routine that works as a 'black box' that evaluates $\boldsymbol{R}^{adj}$ is required. This routine can be the same one used in the problem without uncertainties, with only a slight change; in the problem without uncertainties $\zeta = 1$, while in the presence of uncertainties $\zeta = \zeta(\boldsymbol{\xi}) = \sum_{i=0}^{q} \zeta_i sign(F^i) Y_i(\boldsymbol{\xi})$.

Also, the initialization of $\boldsymbol{\Psi}$ is somewhat different to solving the deterministic adjoint problem. This time, the deterministic adjoint problem needs to be solved by setting

$$\zeta = \zeta_0 sign(F^0)$$

which corresponds to solving the adjoint iPCE equation for $C = 0$, or to finding $\Psi(\boldsymbol{\xi} = \boldsymbol{\xi}_z)$.

The solution of the iPCE adjoint equations is followed by the computation of the sensitivity derivatives given by eq. 5.20. Existing routines that evaluate F_{SD} and Ψ_{SD} can be used again.

Finally, in the choice of coefficients ζ_j, there are two things to consider. First, since

$$E[I] = I^0 \quad , \quad Var[I] = \sum_{j=1}^{q} (I^j)^2$$

the mangitude of $|\zeta_0|$, compared to $|\zeta_j|, j = 1, \ldots, q$ determines whether emphasis is laid on minimizing the mean value or variance. Second, the signs of the coefficients are important. Without loss of generality, J is to be minimized. Therefore $\zeta_j > 0, j = 1, \ldots, q$ since we are always interested in a minimum variance. If F is minimized in the deterministic problem, ζ_0 will also be positive and if I is maximized ζ_0 will be negative.

A Different Objective Function

An equivalent approach could be presented, if the chosen objective function is defined as

$$J = \sum_{j=0}^{q} \zeta_j (F^j)^2 \tag{5.25}$$

In this case, everything presented in this chapter will be the same, except for ζ, which will now be given by

$$\zeta = \sum_{j=0}^{q} 2\zeta_j F^j Y_j(\boldsymbol{\xi}) \tag{5.26}$$

As will be shown in chapter 8, the initial definition of the QoI is preferable to that of eq. 5.25 for the needs of the idea introduced there. However, the two definitions are equivalent, as minimizing either of them yields the desired results. Of course, the weights ζ_j should not be the same and their selection depends on each particular case.

Chapter 6

Demonstration of the Adjoint iPCE Method

In this chapter the previously proposed continuous adjoint method is applied to the 2D Euler equations for steady external flows. This set of equations is chosen because of its simplicity, although the method was programmed for the compressible Navier–Stokes equations.

6.1 Continuous Adjoint of the Deterministic Euler Equations

The Euler equations, written in conservative form, are

$$\begin{aligned} \frac{\partial \boldsymbol{f}_i}{\partial x_i} = A_i \, \frac{\partial \boldsymbol{U}}{\partial x_i} = \boldsymbol{0} \ , \text{ in } \Omega \\ u_i n_i = 0 \ , \text{ in } S \\ \boldsymbol{U} = \boldsymbol{U}_\infty \ , \text{ in } S_\infty \end{aligned} \tag{6.1}$$

where S is the airfoil's boundary and $\boldsymbol{n} = [n_1, n_2]^T$ is its normal unit vector. Also, S denotes the airfoil surface while S_∞ stands for the free–stream, which is far enough from the airfoil; $\boldsymbol{U}_\infty$ are the free–stream flow conditions.

The Jacobian matrices $A_i := \frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{U}}$ are given by

$$A_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{1}{2}[(\gamma-3)u_1^2+(\gamma-1)u_2^2] & (3-\gamma)u_1 & (1-\gamma)u_2 & \gamma-1 \\ -u_1u_2 & u_2 & u_1 & 0 \\ -\gamma u_1\frac{E_t}{\rho}+(\gamma-1)u_1\ \boldsymbol{u}^2 & \gamma\frac{E_t}{\rho}-\frac{\gamma-1}{2}(u_2^2+3u_1^2) & (1-\gamma)u_1u_2 & \gamma u_1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -u_1u_2 & u_2 & u_1 & 0 \\ \frac{1}{2}[(\gamma-3)u_2^2+(\gamma-1)u_1^2] & (1-\gamma)u_1 & (3-\gamma)u_2 & \gamma-1 \\ -\gamma u_2\frac{E_t}{\rho}+(\gamma-1)u_2\ \boldsymbol{u}^2 & (1-\gamma)u_1u_2 & \gamma\frac{E_t}{\rho}-\frac{\gamma-1}{2}(u_1^2+3u_2^2) & \gamma u_2 \end{bmatrix} \quad (6.2)$$

The QoI will be the lift of the airfoil

$$F = L \equiv \int_S p(n_2\cos a_\infty - n_1 \sin a_\infty)dS \quad (6.3)$$

where a_∞ is the free–stream flow angle. Following the discussion in the previous chapter, the augmented function is defined as

$$F_{aug} = F - \int_\Omega \boldsymbol{\Psi}^T\ \frac{\partial \boldsymbol{f}_i}{\partial x_i}d\Omega \quad (6.4)$$

Note that the constraint of the Euler equations' boundary conditions was not substracted from F, as was done in eq. 5.3. However, this constraint will be taken into account later on, making this approach equivalent to defining F_{aug} as in eq. 5.3.

The variation of F_{aug} is

$$\delta F_{aug} = \delta F - \int_\Omega \boldsymbol{\Psi}^T\ \frac{\partial(\delta \boldsymbol{f}_i)}{\partial x_i}d\Omega - \int_\Omega \boldsymbol{\Psi}^T\ \frac{\partial \boldsymbol{f}_i}{\partial x_k}\ \frac{\partial(\delta x_k)}{\partial x_i}d\Omega \quad (6.5)$$

where

$$\delta F = \underbrace{\int_S p(\delta n_2\cos a_\infty - \delta n_1 \sin a_\infty)dS + \int_S p(n_2\cos a_\infty - n_1\sin a_\infty)\delta(dS)}_{=\delta F_{SD}} + \int_S \delta p(n_2\cos a_\infty - n_1\sin a_\infty)dS \quad (6.6)$$

Application of integration by parts yields

$$\int_\Omega \boldsymbol{\Psi}^T\ \frac{\partial(\delta \boldsymbol{f}_i)}{\partial x_i}d\Omega = -\int_\Omega \frac{\partial \boldsymbol{\Psi}^T}{\partial x_i}A_i\delta\boldsymbol{U}d\Omega + \int_S \boldsymbol{\Psi}^T(\delta \boldsymbol{f}_i)n_i dS \quad (6.7)$$

Note that $\delta \boldsymbol{f}_i = A_i\delta\boldsymbol{U}$, which was used to derive eq. 6.7. Also, in eq. 6.7, surface integrals involving the variation of $\boldsymbol{f}_i$ in the far field were omitted, since $\boldsymbol{f}_i$ have constant values there, determined by the infinite flow conditions. Moreover, because

$u_i n_i = 0$ along S, or equivalently $\boldsymbol{f}_i n_i = [0, pn_1, pn_2, 0]^T$, it is

$$\int_S \boldsymbol{\Psi}^T (\delta \boldsymbol{f}_i) n_i dS = \int_S \Psi_{i+1} n_i \delta p dS + \int_S (\Psi_{i+1} p - \boldsymbol{\Psi}^T \boldsymbol{f}_i) \delta (n_i dS) \tag{6.8}$$

Eq. 6.5, because of eqs. 6.6,6.7 and 6.8 is written as

$$\delta F_{aug} = \delta F_{SD} + \underbrace{\int_S (\boldsymbol{\Psi}^T \boldsymbol{f}_i - \Psi_{i+1} p) \delta (n_i dS)}_{\delta F_{SD}^{\Psi}}$$
$$+ \int_S \delta p (n_2 \cos a_\infty - n_1 \sin a_\infty + \Psi_{i+1} n_i) dS + \int_\Omega \frac{\partial \boldsymbol{\Psi}^T}{\partial x_i} A_i \delta \boldsymbol{U} d\Omega \tag{6.9}$$

which results in the adjoint equation and its boundary conditions

$$\begin{aligned} A_i^T \frac{\partial \boldsymbol{\Psi}}{\partial x_i} &= \boldsymbol{0} \ , \text{ in } \Omega \\ n_2 \cos a_\infty - n_1 \sin a_\infty + \Psi_{i+1} n_i &= 0 \ , \text{ in } S \end{aligned} \tag{6.10}$$

the solution of which allows the computation of $\delta F = \delta F_{SD} + \delta F_{SD}^{\Psi}$.

6.2 Adjoint to the iPCE Euler Equations

By applying the $\mathrm{G}^{\mathrm{q}}[]$ operator to eq.6.1, the iPCE Euler equations are derived

$$\begin{aligned} \mathrm{G}^{\mathrm{q}} \left[\frac{\partial \boldsymbol{f}_i}{\partial x_i} \right] = \mathrm{G}^{\mathrm{q}} \left[A_i \frac{\partial \boldsymbol{U}}{\partial x_i} \right] &= \boldsymbol{0} \ , \text{ in } \Omega \\ \mathrm{G}^{\mathrm{q}} [u_i n_i] &= \boldsymbol{0} \ , \text{ in } S \\ \mathrm{G}^{\mathrm{q}} [\boldsymbol{U}] = \mathrm{G}^{\mathrm{q}} [\boldsymbol{U}_\infty] \ &, \text{ in } S_\infty \end{aligned} \tag{6.11}$$

Note that uncertainty may be introduced in the far–field boundary conditions, for example. It may not, however, affect the domain Ω or its boundary S, since then the procedure shown below is not valid. The QoI now is defined as

$$J = \sum_{j=0}^{q} \zeta_j |F^j| \tag{6.12}$$

The solution of eq. 6.11 through the process described in chapter 3 willl yield the spectral coefficients $F^j \equiv L^j$. The augmented function now is

$$J_{aug} = J - \int_\Omega \mathrm{G}^{\mathrm{q}} [\boldsymbol{\Psi}]^T \mathrm{G}^{\mathrm{q}} \left[\frac{\partial \boldsymbol{f}_i}{\partial x_i} \right] d\Omega \tag{6.13}$$

Therefore, because operators δ and $\mathrm{G^q}[]$ permute

$$\begin{aligned}\delta J_{aug} = \delta J - \int_\Omega \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\frac{\partial(\delta \boldsymbol{f}_i)}{\partial x_i}\right] d\Omega \\ - \int_\Omega \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\frac{\partial(\delta \boldsymbol{f}_i)}{\partial x_k}\,\frac{\partial(\delta x_k)}{\partial x_i}\right] d\Omega\end{aligned} \tag{6.14}$$

with

$$\begin{aligned}\delta J = \sum_{j=0}^{q} \zeta_j sign(F^j)\delta F^j = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\delta F\right] \\ = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\int_S \delta p(n_2 \cos a_\infty - n_1 \sin a_\infty) dS\right] + \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\delta F_{SD}\right]\end{aligned} \tag{6.15}$$

where $\zeta = \sum_{j=0}^{q} \zeta_j sign(F^j) Y_j(\boldsymbol{\xi})$, as in eq. 5.17, and F_{SD} is given from eq. 6.6. Then, integration by parts is applied. Because $\mathrm{G^q}[]$ permutes with $\partial/\partial x_i$ (since there is no uncertainty in the domain Ω) we write, as in eq. 6.7

$$\begin{aligned}\int_\Omega \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\frac{\partial(\delta \boldsymbol{f}_i)}{\partial x_i}\right] d\Omega = \\ - \int_\Omega \mathrm{G^q}\left[\frac{\partial \boldsymbol{\Psi}}{\partial x_i} A_i\right]^T \mathrm{G^q}\left[\delta \boldsymbol{U}\right] d\Omega + \int_S \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\delta \boldsymbol{f}_i\right] n_i dS\end{aligned} \tag{6.16}$$

Also, as in eq. 6.19

$$\begin{aligned}\int_S \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\delta \boldsymbol{f}_i\right] n_i dS = \\ \int_S \mathrm{G^q}\left[\Psi_{i+1}\right] n_i \,\mathrm{G^q}\left[\delta p\right] dS + \underbrace{\int_S (\mathrm{G^q}\left[\Psi_{i+1}\right] \mathrm{G^q}\left[p\right] - \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\boldsymbol{f}_i\right])\delta(n_i dS)}_{=\mathrm{G^q}\left[\delta F_{SD}^{\Psi}\right]^T \mathrm{G^q}[1]}\end{aligned} \tag{6.17}$$

where the last term is equal to $\mathrm{G^q}\left[\delta\Psi_{SD}\right]^T \mathrm{G^q}\left[1\right]$, because of proposition 3.6 and the fact that the involved integrals permute with $\mathrm{G^q}[].\delta F_{SD}^{\Psi}$ is given from eq.6.9. Therefore, eq. 6.14, through eqs. 6.15, 6.16 and 6.18, is written as

$$\begin{aligned}\delta J_{aug} = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[F_{SD}\right] + \mathrm{G^q}\left[\delta\Psi_{SD}\right]^T \mathrm{G^q}\left[1\right] + \int_\Omega \mathrm{G^q}\left[\frac{\partial \boldsymbol{\Psi}^T}{\partial x_i} A_i\right] \mathrm{G^q}\left[\delta \boldsymbol{U}\right] d\Omega \\ \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\int_S \delta p(n_2 \cos a_\infty - n_1 \sin a_\infty) dS\right] - \int_S \mathrm{G^q}\left[\Psi_{i+1}\right]^T \mathrm{G^q}\left[\delta p\right] n_i dS\end{aligned} \tag{6.18}$$

From which it is deduced that the iPCE Euler adjoint equation is

$$\mathrm{G^q}\left[A_i^T \,\frac{\partial \boldsymbol{\Psi}}{\partial x_i}\right] = \mathbf{0} \tag{6.19}$$

while its boundary conditions are found by setting

$$\mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\int_S -\delta p(n_2\cos a_\infty - n_1 \sin a_\infty)dS\right] - \int_S \mathrm{G^q}\left[\Psi_{i+1}\right]^T \mathrm{G^q}\left[\delta p\right] n_i dS = \mathbf{0} \Rightarrow$$

$$\mathrm{G^q}\left[\zeta\right]^T \int_S - \mathrm{G^q}\left[\delta p\right](n_2\cos a_\infty - n_1 \sin a_\infty)dS - \int_S \mathrm{G^q}\left[\Psi_{i+1}\right]^T \mathrm{G^q}\left[\delta p\right] n_i dS = \mathbf{0} \Rightarrow$$

$$\mathrm{G^q}\left[\zeta(n_2\cos a_\infty - n_1 \sin a_\infty) + \Psi_{i+1} n_i\right] = \mathbf{0} \tag{6.20}$$

Finally, the sensitivies of J are

$$\delta J = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\delta F_{SD}\right] + \mathrm{G^q}\left[\delta F_{SD}^{\Psi}\right]^T \mathrm{G^q}\left[1\right] \tag{6.21}$$

Regarding the choice of coefficients ζ_j, since we are interested in a maximum mean value and a minimum variance of the lift, we could choose a negative ζ_0 and positive ζ_j , $j > 0$, so that the minimization of J will yield the desired result.

Chapter 7

Numerical Implementation of the Adjoint iPCE Method

The aforementioned continuous adjoint method is applied to the laminar flow around a 2D isolated airfoil. The airfoil geometry is parameterized using two Bezier curves and the coordinates of their control points are the design variables. More specifically, nine control points are used for the suction side and nine for the pressure side, as shown in fig. 7.1. Note that the vertical displacement of each control point is bounded and that the first and last two control points are kept fixed. This is also shown in fig. 7.1. Two cases are tested.

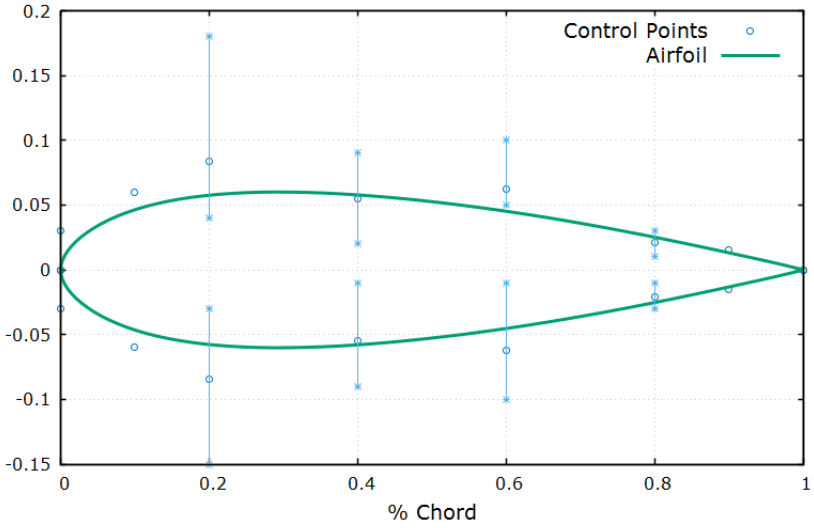


Figure 7.1: *Shape optimization of a 2D airfoil. Initial airfoil geometry and control points used to create the two Bezier curves.*

First case

In this first case, the objective function to be minimized in the absence of uncertainties is the airfoil's drag coefficient, i.e. $F = C_D$. This is in fact the QoI that will be used in the optimization under uncertainties that follows. Initially, a shape optimization without uncertainties is carried out, for which the flow conditions are

$$M_\infty = 0.5 \quad , \quad a_\infty = 2^o \quad , \quad Re = 6000 \tag{7.1}$$

where Re is the chord–based Reynolds number. The resulted optimized geometry yields a drag coefficient value of about 20% smaller than that of the initial symmetric airfoil, fig. 7.2, on the left.

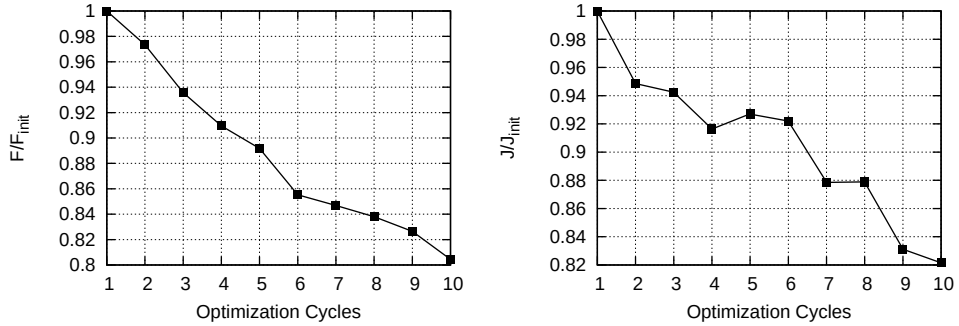


Figure 7.2: *Shape optimization of a 2D airfoil, drag minimization. History of the objective function value in the optimization without uncertainties (left) and in the optimization under uncertainties (right); laminar flow.*

The shape–optimization under uncertainties follows. Uncertainties are introduced in the flow conditions which have the following probability distributions

$$M_\infty \sim \mathcal{N}(0.5, 0.05) \quad , \quad a_\infty \sim \mathcal{U}(1.5^o, 2.5^o) \quad , \quad Re \sim \mathcal{N}(6000, 250)$$

In this case, the objective function is formulated as

$$J = \sum_{j=0}^{q} \zeta_j |F^j|$$

with $q = 19$ (for $m = 3$ uncertain variables and chaos order $C = 3$), $\zeta_0 = 1$ and $\zeta_j = 3 \ \forall \ j > 0$. A reduction of the QoI of about 18% is achieved, see fig. 7.2 on the right. The optimized airfoil geometries resulted from the two runs (without and under uncertainties) compared to the initial one are shown in fig. 7.4. Moreover, fig. 7.3 shows the coordinates of the control points for the optmized geometry. The fields of the adjoint velocity magnitude on the optimized geometries is illustrated in fig. 7.5. Table 7.2 compares the statistical moments of C_D from both optimization runs. The values on the left column are computed by an uncertainty quantification on the optimized geometry resulted from the run without uncertainties. As expected, the UQ that was applied on the geometry that resulted from the optimization without uncertainties results in lower mean value but has higher standard deviation of the

drag coefficient compared to the mean value and standard deviation that resulted from the optimization under uncertainties.

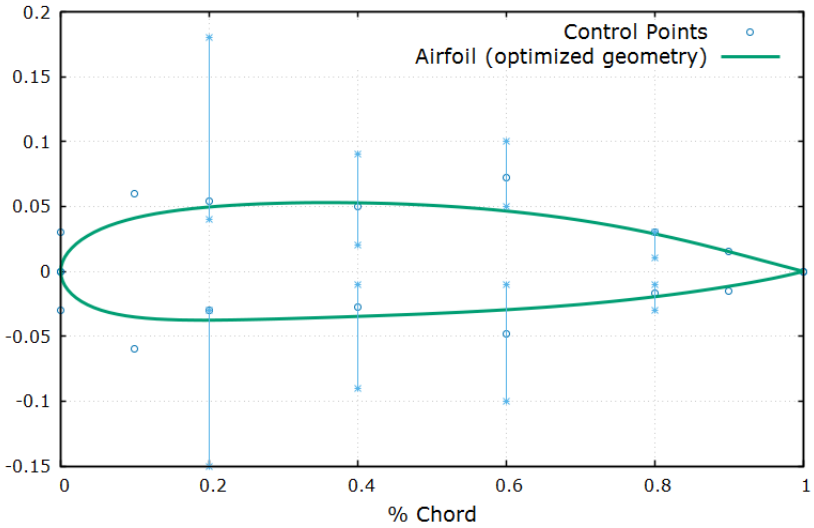


Figure 7.3: *Shape optimization of a 2D airfoil. Optimizedl airfoil geometry and control points used to create the two Bezier curves.*

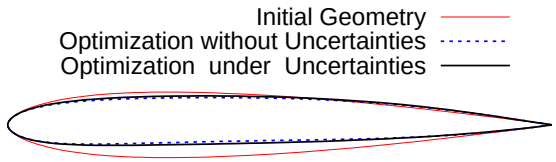


Figure 7.4: *Shape optimization of a 2D airfoil. Comparison of the optimized geometries with and without uncertainties with the initial one; laminar flow.*

	Without Uncertainties	Under Uncertainties
μ_{C_D}	$6.81 \cdot 10^{-2}$	$6.97 \cdot 10^{-2}$
σ_{C_D}	$1.11 \cdot 10^{-3}$	$1.05 \cdot 10^{-3}$

Table 7.1: *Shape optimization of a 2D airfoil. Comparison of the statistical moments of the drag coefficient of the optimized geometries resulted from the two optimization runs.*

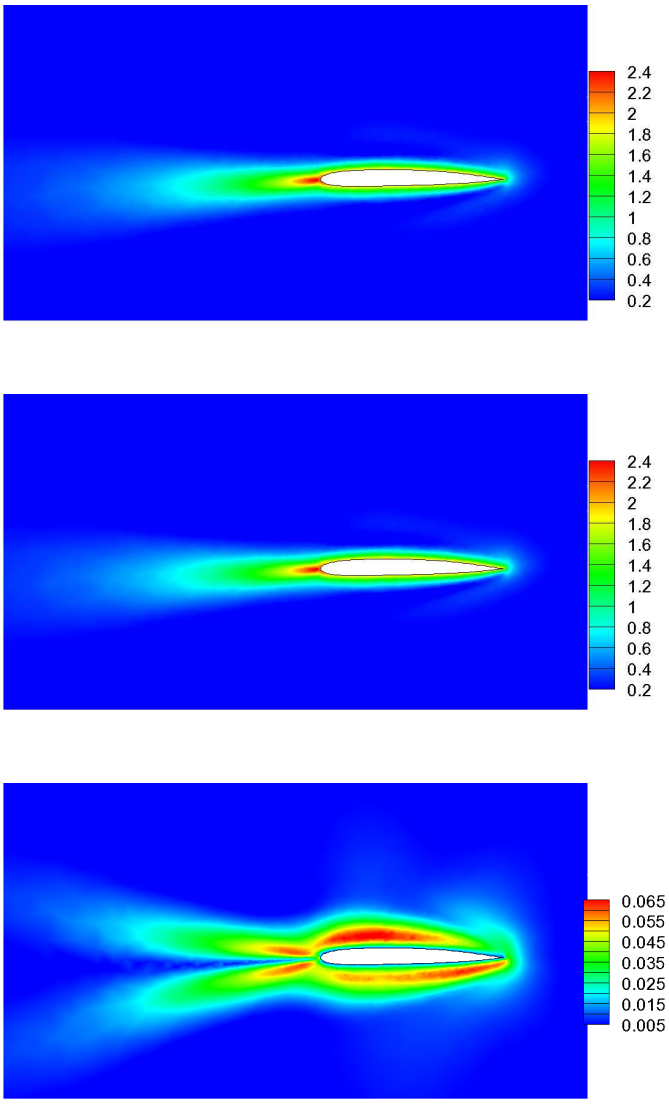


Figure 7.5: *Shape optimization of a 2D airfoil. Field of the non–dimensional adjoint velocity magnitude computed by the design without uncertainties (top) and mean (middle) and standard deviation (bottom) of the adjoint velocity from the design under uncertainties; laminar flow.*

Second case

In the second case, the boundary conditions have the following probability distributions

$$M_\infty \sim \mathcal{N}(0.5, 0.07) \quad , \quad a_\infty \sim \mathcal{U}(1.5^o, 2.5^o) \quad , \quad Re \sim \mathcal{N}(5000, 300) \tag{7.2}$$

while QoI for the case without uncertainties is defined as

$$I = \beta(C_L - C_{L_{tar}})^2 + C_D \tag{7.3}$$

with C_L and C_D being the lift and drag coefficients of the airfoil, while $C_{L_{tar}} = 0.18$ and $\beta = 0.1$. The QoI of the iPCE problem is then given by

$$J = \zeta_j |I^j| \ , \ \zeta_0 = 1 \ , \ \zeta_j = 10 \ , \ j > 0 \tag{7.4}$$

and the optimization is carried out for two different chaos orders, one for $C = 1$ and one for $C = 2$. Fig. 7.6 depicts the value of J as the optimization cycles increase. The two curves are similar, but the one corresponding to $C = 2$ has somewhat larger values. This should be attributed to the fact that more spectral coefficients I^j are kept for $C = 2$. It also depicts the initial and the optimal airfoil geometries. In fig. 7.7 the mean value and standard deviation of the drag coefficient are shown, as functions of the optimization cycles. In fig. 7.8 the initial mean value and standard deviation of the Mach number's field around the airfoil are shown, whereas fig. 7.9 shows the optimized airfoil geometry, along with the same fields, for $C = 1$.

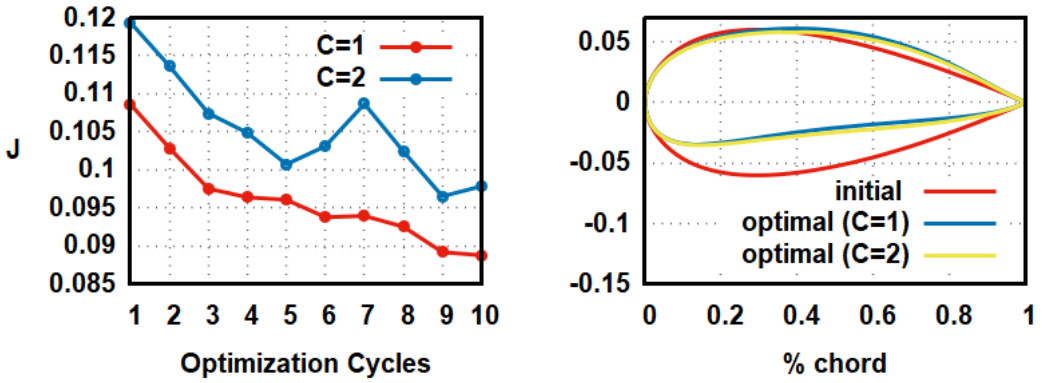


Figure 7.6: *QoI Minimization as a Function of the Optimization Cycles, for $C = 1, 2$ (left) and Initial (red) and Optimized (Blue) Airfoil Geometries for $C = 1$ (right).*

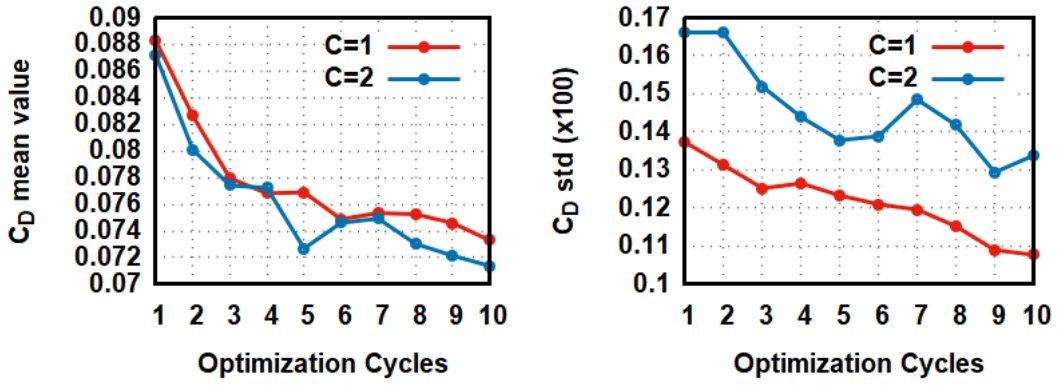


Figure 7.7: *Drag coefficient mean Value (left) and standard deviation (right) minimization as a function of the optimization cycles, for $C = 1, 2$*

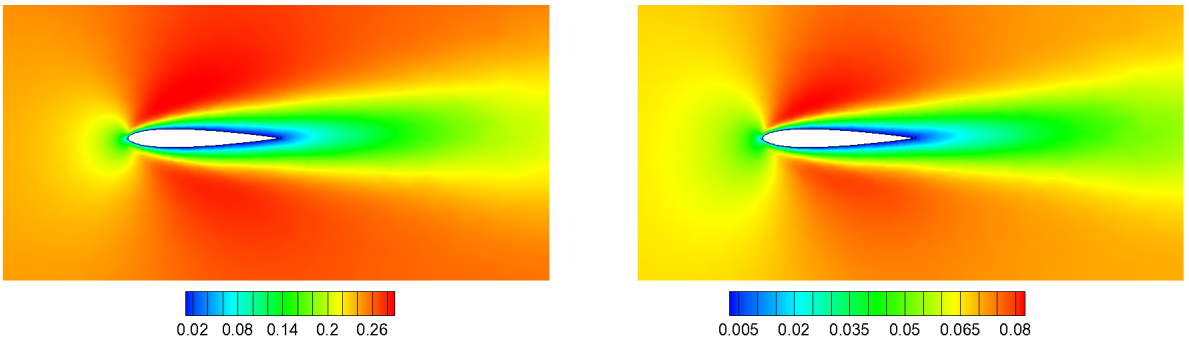


Figure 7.8: *Mean value (left) and standard deviation (right) of Mach number around the initial airfoil geometry.*

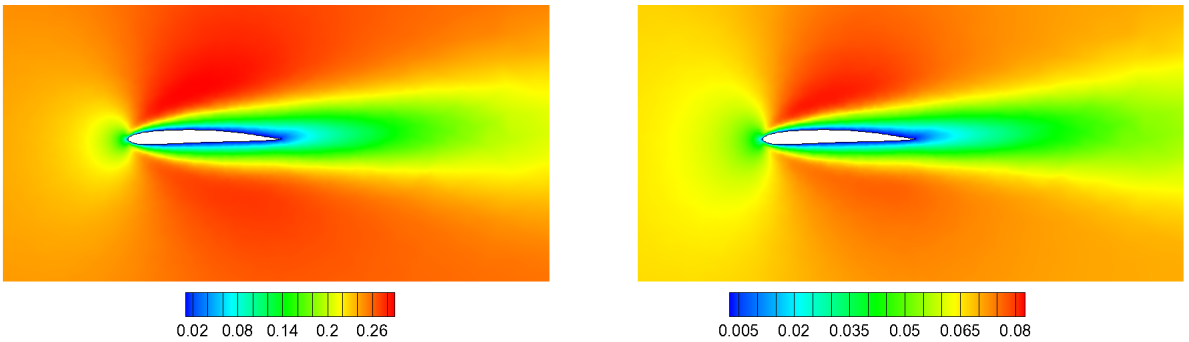


Figure 7.9: *Mean value (left) and standard deviation (right) of Mach number around the optimized airfoil geometry, for $C = 1$.*

In table 7.2 the results of the optimization that concern the drag coefficient are summarized. In the absence of uncertainties, another optimization was carried out for the following values of the free–stream boundary conditions

$$M_\infty = 0.5 \quad , \quad a_\infty = 2^o \quad , \quad Re = 5000 \tag{7.5}$$

that resulted in $C_D = 7.38 \cdot 10^{-1}$, which is practically equal to the predicted optimal mean value, in the presence of uncertainties. Then, the optimal values of the design variables that were found for the case without uncertainties were used as inputs for the iPCE solver, for $C = 1$ and $C = 2$. The resulting mean value and standard deviation of the drag coefficient are shown in table 7.3. Comparison of these values with the results of table 7.2 shows that the optimal airfoil in the case without uncertainties has a less robust performance than the one that resulted when uncertainties were taken into account.

	$C = 1$	$C = 2$
μ_{C_D}	$7.33 \cdot 10^{-2}$	$7.14 \cdot 10^{-2}$
σ_{C_D}	$1.08 \cdot 10^{-3}$	$1.34 \cdot 10^{-3}$

Table 7.2: *Optimization under uncertainties of a 2D airfoil in laminar flow conditions. Mean value and standard deviation of drag coefficent for optimal airfoil geometries in optimization under uncertainties for* $C = 1, 2$.

	$C = 1$	$C = 2$
μ_{C_D}	$7.44 \cdot 10^{-2}$	$7.32 \cdot 10^{-2}$
σ_{C_D}	$1.34 \cdot 10^{-3}$	$1.50 \cdot 10^{-3}$

Table 7.3: *Optimization under uncertainties of a 2D airfoil in laminar flow conditions. Mean Value and standard deviation of drag coefficent predicted by the iPCE solver with optimal design variables from optimization without uncertainties as inputs, for* $C = 1, 2$

Chapter 8

An Alternative to the Adjoint iPCE

In this chapter, a more efficient alternative to the adjoint iPCE method is proposed, programmed and assessed. This approach will be referred to as 'Deterministic Derivatives – Stochastic Primal', or as DDSP. The main idea is to replace the adjoint iPCE with a solution of the adjoint equations without uncertainties (at a specific value–set of the uncertain variables, determined through a process explained below), while computing the derivatives of the objective function accurately.

8.1 The DDSP method

Recall the definition of the objective function for the problem with uncertainties

$$J = \sum_{i=0}^{q} \zeta_i |F^i| = \sum_{i=0}^{q} \zeta_i \text{sign} F^i F^i \tag{8.1}$$

where ζ_i are user–defined coefficients and F^i stand for the spectral coefficients of the PCE of the QoI that may become known only after solving the (primal) iPCE equations. The gradient with respect to the design variables (let δ denote the gradient in this section) that the adjoint iPCE method should compute is

$$\delta J = \sum_{i=0}^{q} \zeta_i \text{sign}(F^i) \delta F^i \tag{8.2}$$

The adjoint run always follows the primal iPCE run, which means that the spectral coefficients F^i of the QoI at the current solution (current value–sets of the design variables) have been computed and are known. This means that $sign(F^i)$ are known quantities. On the other hand, the gradient computed by the numerical solution, for a given $\boldsymbol{\xi} = \boldsymbol{\xi}_s$ (this value–set will be defined later on), of the adjoint equations

without uncertainties is

$$\delta F(\boldsymbol{\xi}_s) \equiv \sum_{i=0}^{\infty} \delta F^i Y_i(\boldsymbol{\xi}_s) \tag{8.3}$$

Recall that giving the uncertain variables a fixed value–set $\boldsymbol{\xi} = \boldsymbol{\xi}_s$ means that the uncertain flow conditions of the stochastic problem (or any other input to it, should this be the case), take on values that can directly be derived from $\boldsymbol{\xi} = \boldsymbol{\xi}_s$ and the deterministic problem (i.e. the flow solver in the absence of uncertainties) should be solved for those values.

Since this new idea aims at replacing the solution of the iPCE adjoint equations with a single solution of the adjoint problem without uncertainties (being much less expensive, of course), without damaging the accuracy of computing δJ, we require that

$$\delta F = \delta J \Rightarrow \sum_{i=0}^{\infty} \delta F^i Y_i(\boldsymbol{\xi}_s) = \sum_{i=0}^{q} \zeta_i \text{sign}(F^i) \delta F^i \tag{8.4}$$

Note that the RHS of eq. 8.4 is not a truncated infinite sum, but the weighted sum of $q+1$ terms, exclusively depending upon the selected chaos order C and number of uncertain variables m. The LHS, however, is an infinite sum which must be truncated; thus, the first $q+1$ terms must be retained and terms of higher–order are neglected. The next step is to express the following equalities

$$Y_i(\boldsymbol{\xi}_s) = \zeta_i \text{sign}(F^i), i = 0, 1, \ldots, q \tag{8.5}$$

Eqs. 8.5 essentially imply that, by selecting $\boldsymbol{\xi}_s$ so as to satisfy eq. 8.5, the derivatives computed by a solution of the problem without uncertainties can be used, instead of solving the iPCE adjoint equations (if, of course, such a $\boldsymbol{\xi}_s$ can be found). Therefore, if the problem without uncertainties is solved for that $\boldsymbol{\xi}_s$ (along with its adjoint equations), the sensitivities computed this way can be used, instead of those resulting from solving the adjoint to the iPCE equations. However, it is very important to note that the sole source of error of the DDSP method spings from the truncation in eq. 8.4. It is expected that, if low–order truncation is performed (i.e. for a low value of q) the error in computing the gradient through the DDSP method should be higher; this is investigated, to a certain extent, in this section.

8.2 Solving eqs. 8.5

As it was previously mentioned, the DDSP method aims to determine a value–set $\boldsymbol{\xi}_s$ of the uncertain variables that satisfies eq. 8.5. Therefore, the unknowns in those equations are the components of $\boldsymbol{\xi}_s$, of course. However, for $j = 0$, the first out of the $q+1$ equations to be satisfied is

$$\zeta_0 = Y_0 \text{sign}(F^0) = \text{sign}(F^0) \tag{8.6}$$

since $Y_0 = 1$ for any set of orthonormal polynomials. Eq. 8.6 does not involve any component of $\boldsymbol{\xi}_s$, but must be satisfied, if the gradient of the objective function is to be accurately computed. For this reason, eq. 8.6 is considered to be an

additional constraint, which imposes a specific value to the otherwise user–defined ζ_0. However, this is not an issue. Without loss of generality, it would be possible to assume $|\zeta_0| = 1$ anyway, and then proceed to define the magnitude of the other weights ζ_j, since only the relative size ζ_j/ζ_0 , $j > 0$ is needed for the optimization to yield the desired results.

After defining ζ_0 through eq. 8.6 (and, of course, having already chosen the other weights ζ_j based on the user–defined ratios of ζ_j's), the number of remaining equations is reduced to q

$$Y_i(\boldsymbol{\xi}_s) = \zeta_i \text{sign}(F^i), i = 1, \ldots, q$$

which are handled differently, depending on the choice of chaos order. Note that defining ζ_0 through the constraint of eq. 8.6 will not result in ζ_0 changing signs between optimization cycles, as F^0 usually has a fixed sign (it may be the mean value of the drag/lift coefficient of an aircraft, for instance, which has a fixed sign)

Solving eqs. 8.5 for $C = 1$

The solution of eqs. 8.5 is straightforward, if the chosen chaos order is $C = 1$. In such a case, the number of unknowns is equal to the number of uncertain variables m, as $q = m$ for $C = 1$, recall eq. 2.26. Thus, they can be easily solved and $\boldsymbol{\xi}_s$ is determined this way.

Solving eqs. 8.5 for $C > 1$

For $C > 1$, the reader may notice that there are $q = (C + m)!/C!m! - 1$ equations and only m unknowns (ζ_0 is again given by eq. 8.6). For this reason, $\boldsymbol{\xi}_s$ will be chosen so that the following expression be minimized

$$M := \frac{1}{2} \sum_{i=1}^{q} [Y_i(\boldsymbol{\xi}) - \zeta_i \text{sign}(F^i)]^2 \tag{8.7}$$

Therefore, the system of equations to solve becomes

$$R_j := \frac{\partial M}{\partial \xi_j} = \sum_{i=1}^{q} [Y_i(\boldsymbol{\xi}) - \zeta_i \text{sign}(F^i)] \frac{\partial Y_i}{\partial \xi_j} = 0 \quad , \; j = 1, \ldots, m \tag{8.8}$$

Eq. 8.8 is solved iteratively through Newton's method

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\xi}} \right)_{old} \Delta \boldsymbol{\xi} = -\boldsymbol{R}_{old} \tag{8.9}$$

where $\boldsymbol{R} = (R_1, \ldots, R_m)$, $\boldsymbol{\xi}_{new} = \boldsymbol{\xi}_{old} + \Delta \boldsymbol{\xi}$ and

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\xi}} \right)_{jk} = \sum_{i=1}^{q} \left([Y_i - \zeta_i \text{sign}(F^i)] \frac{\partial^2 Y_i}{\partial \xi_k \partial \xi_j} + \frac{\partial Y_i}{\partial \xi_j} \frac{\partial Y_i}{\partial \xi_k} \right) \tag{8.10}$$

After the non–linear least–squares problem, eq. 8.8, is solved, a value–set $\boldsymbol{\xi}_s$ will be determined that does not, however, completely satisfy eq. 8.5. In order to completely satisfy eq. 8.5, the user would have to redefine the weights ζ_j , $j > 0$, so that they would be equal to the RHS of eq. 8.5, for the $\boldsymbol{\xi}_s$ that was just found. Equivalently, the solution of the least–squares problem will yield a value–set $\boldsymbol{\xi}_s$ (for which the gradient of the objective function J will be computed) that corresponds to the coefficients

$$c_j := \text{sign}(F^j) Y_j(\boldsymbol{\xi}_s) \ , \ j = 1, \ldots, q \tag{8.11}$$

and, as a result, the gradient of $\sum_{j=0}^{q} c_j |F^j|$ instead of that of $\sum_{j=0}^{q} \zeta_j |F^j|$ will be computed. This essentially means that the whole optimization will correspond to a somewhat different objective J, than the one initially chosen. This will be shown to be a minor issue and a small price to pay though, if a major gain in computational cost is to be achieved.

8.3 How to Apply the Idea - Discussion

Summing up, it has been proven that the solution of the adjoint equations without uncertainties can be sufficient, when the goal is to compute the gradient δJ. To achieve this, the solution must be made for a fixed $\boldsymbol{\xi}_s$, which is determined by solving the least squares problem discussed in the two previous sections.

However, one first thing to notice is the approximation of the infinite sum in eq. 8.4. Should the chosen chaos order be not high enough, the DDSP method is expected to have a non–negligible error in estimating δJ. More specifically, low chaos orders may not be sufficient to approximate the infinite sum of eq.8.4; in this case, the derivative δJ would be set equal to a sum containing not only δJ but also some other, non–negligible, terms, eventually leading to inaccuracies.

For example, assuming one uncertain variable $m = 1$ and a chaos order $C = 1$, eq. 2.26 yields $q = 1$. Then $J = \zeta_0 |F^0| + \zeta_1 |F^1|$. It is possible, though, that the term F^2, which corresponds to $C = 2$, is non–negligible compared to F^1 (terms that correspond to $C > 2$ are assumed negligible). Therefore, it would have been preferable to set $C = 2$. Having chosen $C = 1$, eq. 8.4 would be written as

$$\delta J = \delta F \Rightarrow \zeta_0 sign(F^0)\delta F^0 + \zeta_1 sign(F^1)\delta F^1 = \delta F^0 Y_0(\boldsymbol{\xi}_s) + \delta F^1 Y_1(\boldsymbol{\xi}_s) + \underbrace{\delta F^2 Y_2(\boldsymbol{\xi}_s)}_{source \ \ of \ \ error}$$

After this issue is dealt with and a high enough C is chosen, the following steps have to be followed in every robust design optimization cycle

- Solve the primal iPCE equations
- Solve the least squares problem, to find $\boldsymbol{\xi}_S$
- Solve a primal and an adjoint problem without uncertainties, for $\boldsymbol{\xi} = \boldsymbol{\xi}_s$ and compute δJ

Instead of solving the primal problem without uncertainties in order to obtain $\boldsymbol{U}(\boldsymbol{\xi}_s)$, the iPCE primal equations can be used as a surrogate model. The spectral represantation of $\boldsymbol{U}$, which becomes available after the iPCE equations are solved, is an explicit expression of $\boldsymbol{U}$, as a function of $\boldsymbol{\xi}$. Setting $\boldsymbol{\xi} = \boldsymbol{\xi}_s$ yields $\boldsymbol{U}(\boldsymbol{\xi}_s)$.

The approach used in this diploma thesis is slightly different though. The only information required from the relatively costly step of solving the iPCE equations is to find $\text{sign}(F^i),\ i = 0, \ldots, q$. For this reason, the first step described above can be shortened to a great extent; one only needs to solve the iPCE equations up to the point where the signs of the spectral coefficients F^i are safely determined. To this end, a loose 'convergence' criterion is applied, making this step much more faster than completely solving the iPCE equations.

8.4 Applications/Comparison with adjoint iPCE

In this section, the DDSP method is applied to the isolated airfoil case presented in chapter 9, for laminar flow conditions. Recall that the airfoil was created from two Bezier curves and the coordinates of their control points are the design variables. In all cases, the QoI of the problem without uncertainties is

$$F = \beta(C_L - C_{L_{tar}})^2 + C_D\ ,\ C_{L_{tar}} = 0.18\ ,\ \beta = 0.2 \tag{8.12}$$

8.4.1 Accuracy Tests

First, we discuss the accuracy of the DDSP method in correctly evaluating derivatives. All tests are carried out for a single uncertain variable, the free–stream Mach number, and the derivatives are found by applying the finite difference method to the iPCE solver (yielding the 'exact' derivative). After that, eq. 8.5 is solved to find a fixed $\boldsymbol{\xi}_s$, for which finite differences are applied to the solver of the deterministic equations, yielding the derivating computed by the DDSP method.

First Test

The first test is carried out for $C = 1$ and a probability distribution of the free–stream Mach number given by

$$M_\infty \sim \mathcal{N}(0.5, 0.01) \Leftrightarrow M_\infty = 0.5 + 0.01\xi\ ,\ \xi \sim \mathcal{N}(0, 1) \tag{8.13}$$

while the objective function is defined as

$$J = |F^0| + 10|F^1| \tag{8.14}$$

which implies that $\zeta_0 = 1$ and $\zeta_1 = 10$. Note that the requirement that $\zeta_0 = Y_0 sign(F^0) = 1$ is satisfied in this case.

The solution of eq. 8.5 is straightforward

$$Y_1(\xi_s) = \zeta_1 sign(F^1) \Rightarrow \xi_s = 10 \tag{8.15}$$

since F^1 was positive. Therefore,

$$M_\infty = 0.5 + 0.01\xi_s = 0.6 \tag{8.16}$$

is the boundary condition for which the deterministic problem has to be solved, in order to compute the derivatives with the DDSP method. Hence, the DDSP derivatives were found after the application of finite differences to the primal problem, after setting $M_\infty = 0.6$. The comparison was made with the derivatives found by applying finite differences to the primal iPCE solver.

The results are shown in fig. 8.1 and in table 8.1. It is evident that there is very good agreement between the two.

	Pressure Side		Suction Side	
Control Point #	FD (iPCE)	DDSP	FD (iPCE)	DDSP
1	-0.55071	-0.55756	0.78802	0.79789
2	-0.58644	-0.59409	0.58944	0.60527
3	-0.37392	-0.38137	0.27344	0.28936
4	-0.19606	-0.20255	0.08028	0.09312
5	-0.08533	-0.09038	-0.00864	0.00050
6	-0.02707	-0.03060	-0.03625	-0.03049
7	-0.00433	-0.00652	-0.03277	-0.02953
8	-0.00585	-0.00712	-0.01478	-0.01332
9	-0.03228	-0.03357	0.01065	0.01144

Table 8.1: *First accuracy test. Comparison of finite difference derivatives computed by the iPCE solver and the deterministic solver for input boundary conditions determined by eq.8.5 (DDSP method), for one uncertain variable and* $C = 1$.

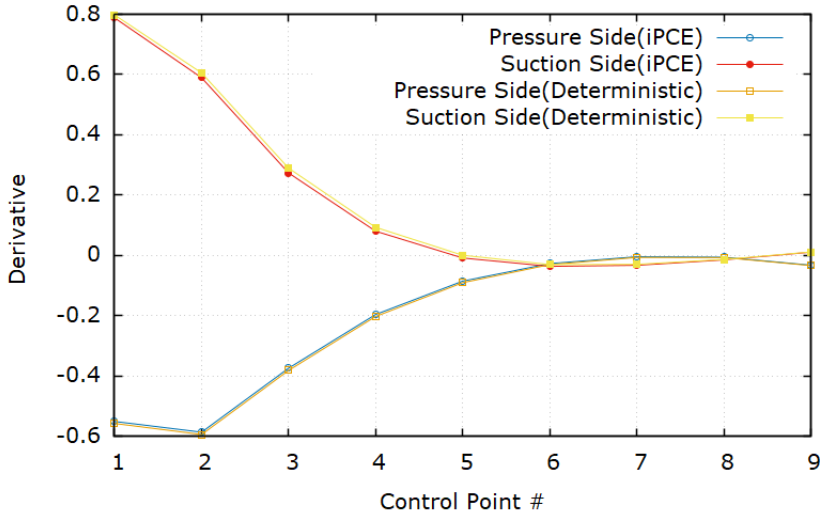


Figure 8.1: *First accuracy test. Comparison of finite difference derivatives computed by the iPCE solver and the deterministic solver for input boundary conditions determined by eq.8.5 (DDSP method), for one uncertain variable and* $C = 1$.

Second Test

The second test is similar to the first, with a higher standard deviation of the free–stream Mach number though, namely

$$M_\infty \sim \mathcal{N}(0.5, 0.07) \Leftrightarrow M = 0.5 + 0.07\xi \ , \ \xi \sim \mathcal{N}(0, 1) \tag{8.17}$$

and the objective function is defined as

$$J = |F^0| + 5|F^1| \tag{8.18}$$

which implies that $\zeta_0 = 1$ and $\zeta_1 = 5$. Again, the requirement that $\zeta_0 = Y_0 sign(F^0) = 1$ is satisfied. The solution of eq.8.5 yields $\xi_s = 5$ and now the imposed boundary condition is

$$M_\infty = 0.5 + 0.07\xi_s = 0.85 \tag{8.19}$$

which is aerodynamically much higher than 0.5. The results are summarized in fig. 8.2. It is evident that the DDSP method is somewhat inaccurate this time. This should be attributed to the relatively high variance of the boundary conditions, combined with the low chaos order chosen. This test highlights the main drawback of this chapter's idea, which is its dependence on the chosen chaos order, if we are to have accurate results.

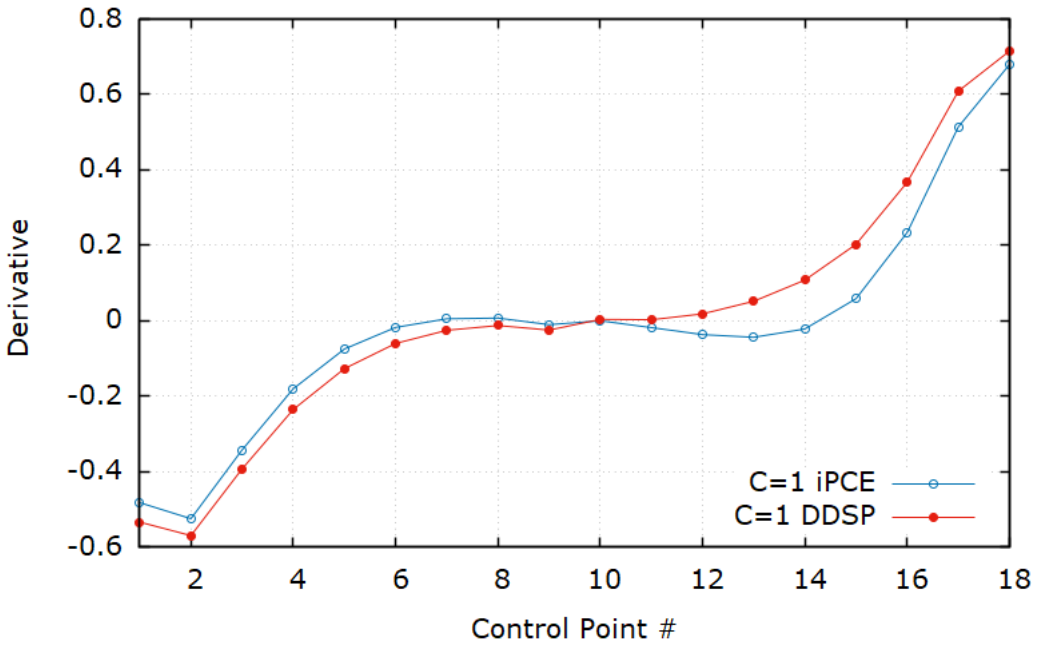


Figure 8.2: *Second accuracy test. Comparison of finite difference derivatives computed by the iPCE solver and the deterministic solver for input boundary conditions determined by eq.8.5 (DDSP method), for one uncertain variable and* $C = 1$.

Third Test

In this final test, the free–stream Mach number is again given by eq. 8.17. This time, however, the method is tested for $C = 2$ and $C = 3$. Recall that, for $C > 1$, a least squares problem has to be solved, eq. 8.8, to determine the value of ξ_s for some desired ζ_j. However, the resulting ξ_s will not of course satisfy eq. 8.5. For this reason, the derivatives computed by the DDSP method will correspond to different coefficients c_j than those given by the user, that are found from

$$c_j = \text{sign}(F^j) Y_j(\boldsymbol{\xi}_s)$$

In table 8.2 the solution $\boldsymbol{\xi}_s$ of the least squares problem is given, for $C = 2$ and $C = 3$, along with the user–defined ζ_j and the resulting c_j that are found from eq.8.11. Therefore, finite differences were applied to the deterministic equations for the values of the infinite Mach number found in table 8.2, which resulted in the computation of the derivatives that correspond to the coefficients c_j. For the comparison to be fair, finite differences were now applied to the iPCE solver, not for $\zeta_j = 5$, but for $\zeta_j = c_j$. Results are shown in fig.8.3. It is now clear that the proposed method is accurate and the accuracy increases with the chaos order.

	$C=2$	$C=3$
ζ_j	c_j	
5	2.9598	2.7918
5	5.4874	4.7646
5	-	5.3810
ξ_s	2.9598	2.7818
M_∞	0.7072	0.6947

Table 8.2: *Summary of least squares solution for $C = 2, 3$, along with the resulting value of the boundary condition.*

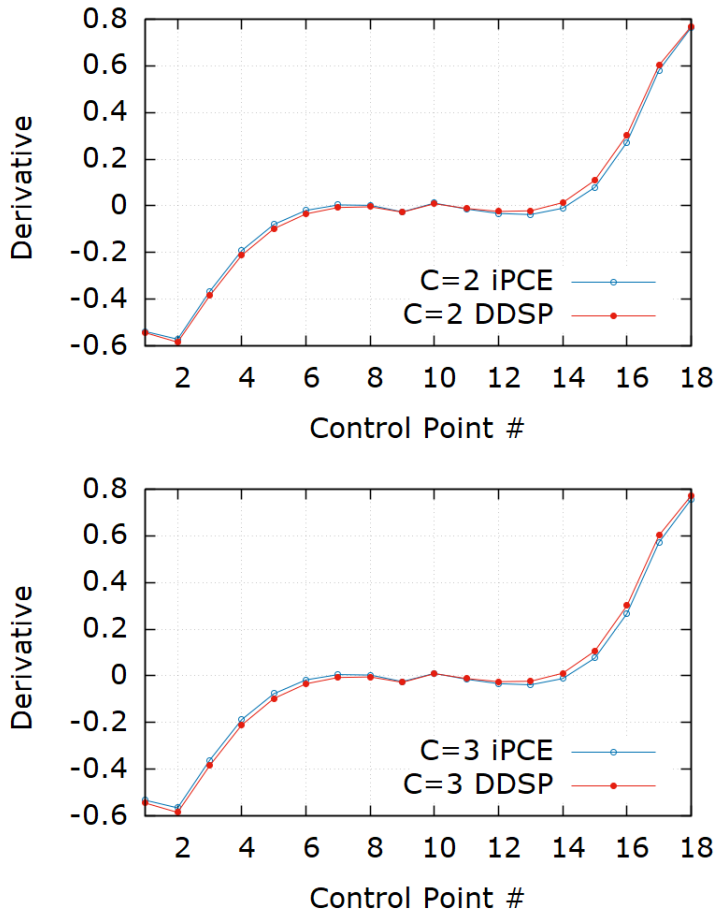


Figure 8.3: *Third accuracy test. Comparison of finite difference derivatives for $C = 2$ (top) and $C = 3$ (bottom), calculated by the iPCE solver and the DDSP method.*

The main conclusion of these accuracy tests is that the DDSP method correctly finds derivatives if the chaos order is relatively high ($C > 1$) or if the uncertainties are rather small, so that they can be correctly quantified for $C = 1$.

8.4.2 Optimization – Comparison of Computational Cost

The DDSP method is now applied to the optimization of an isolated airfoil at laminar flow conditions; the objective function is the drag coefficient in the case without uncertainties. Uncertainty is introduced through the free–stream Mach number, free–stream flow angle and chord–based Reynolds number, that have the following probability distributions

$$M_\infty \sim \mathcal{N}(0.5, 0.05) \quad , \quad a_\infty \sim \mathcal{U}(-1.5^o, 2.5^o) \quad , \quad Re_\infty \sim \mathcal{N}(5000, 300) \tag{8.20}$$

The objective function to be minimized is defined in this case as

$$J = \sum_{i=0}^{q} \zeta_j |C_D^j| \ , \ \zeta_0 = 1 \ , \ \zeta_j = 5 \ , \ j > 0 \tag{8.21}$$

Comparisons in terms of computational cost and drag minimization are made between the DDSP and the adjoint iPCE approach.

Optimization for $C = 1$

The results of the optimization for $C = 1$ are summarized in fig.8.4, where the mean drag coefficient, its standard deviation and the objective function are shown, as a function of the optimization cycles. Note that these values were available every two optimization cycles, when the DDSP method was applied; this is because the application of this method does not require the full convergence of the iPCE solver in every optimization cycle. So, to save time, the iPCE was allowed to fully converge once every second cycle.

It is evident that both the adjoint iPCE and the DDSP methods failed to properly lower the standard deviation of the QoI. Although small values were achieved (especially in the 12th optimization cycle), the convergence was not smooth but had many oscillations. In both methods, this should be attributed to the choice of chaos order, which seems not high enough to correctly estimate the true value of the standard deviation. It is also clear that the results of the two methods are close. The reason any differences are seen is unclear though; since the derivatives were calculated using the fast, but rather inaccurate, SI (Surface Integral) adjoint method, it is not certain whether they should be attributed to the DDSP method or to the accuracy of the adjoint SI calculation.

Finally, in terms of computational cost, the DDSP method is noticeably faster; it required 869 seconds, whereas the adjoint iPCE method took 1673 seconds to be completed. This difference could grow even more, if the iPCE solver used in the DDSP method was allowed to converge more rarely than once every two optimization cycles.

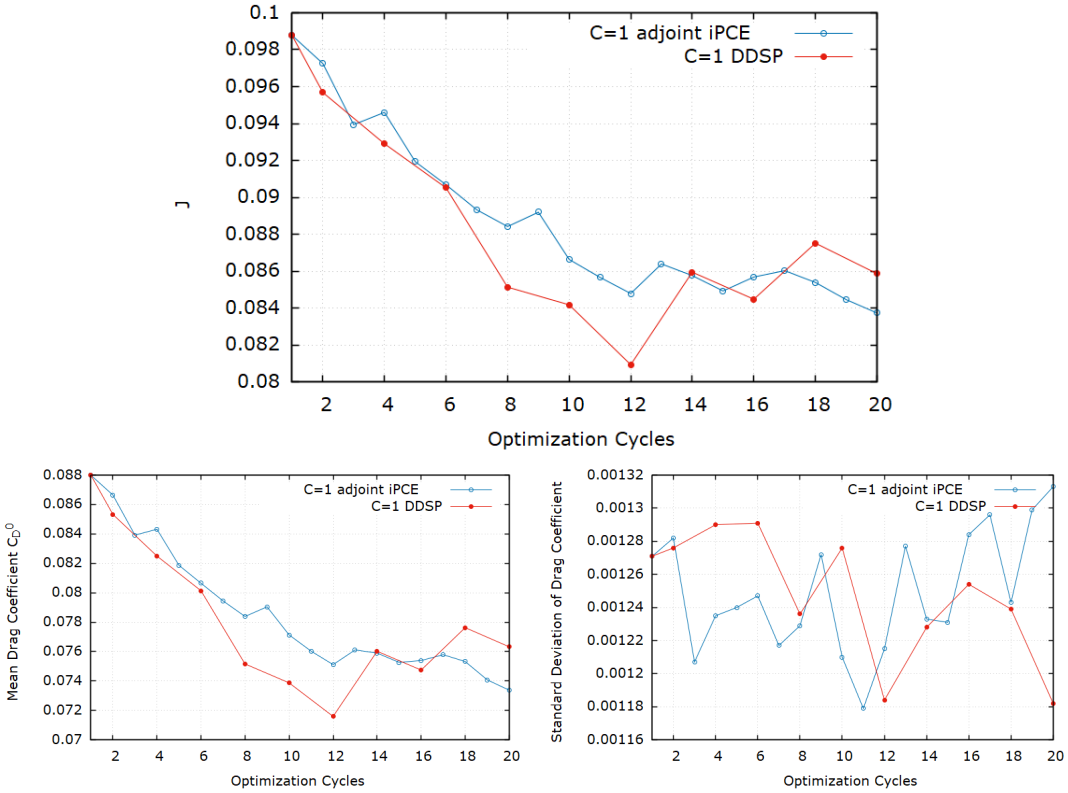


Figure 8.4: *Optimization results for $C = 1$. Value of the QoI (top), mean drag coefficient (bottom left) and standard deviation of drag coefficient (bottom right).*

Optimization for $C = 2$

Similar results to those previously presented for $C = 1$ are now shown in fig.8.5, for $C = 2$. This time, the convergence of the standard deviation value is somewhat smoother, although in the adjoint iPCE case it still has some oscillations. In terms of computational cost, the adjoint iPCE needed 5710 seconds, while the DDSP only took 1908 seconds to be completed, offering thus a major gain in computational burden.

In addition to that, it seems that the DDSP method yields lower values for the QoI. Although this might be coincidental, it can be argued that the DDSP method can sometimes compute derivatives more accurately than the adjoint iPCE method. More specifically, both the adjoint iPCE and the deterministic adjoint solver used in the DDSP take the values of the field variables $\boldsymbol{U}$ as input. In the adjoint iPCE, the PCE of $\boldsymbol{U}$ is truncated, based on the chosen chaos order; in the DDSP method, however, this is not the case. The value of $\boldsymbol{U}(\boldsymbol{\xi} = \boldsymbol{\xi}_s)$ that is used as an input has no truncation, which may be a reason why the DDSP derivatives can sometimes be more accurate.

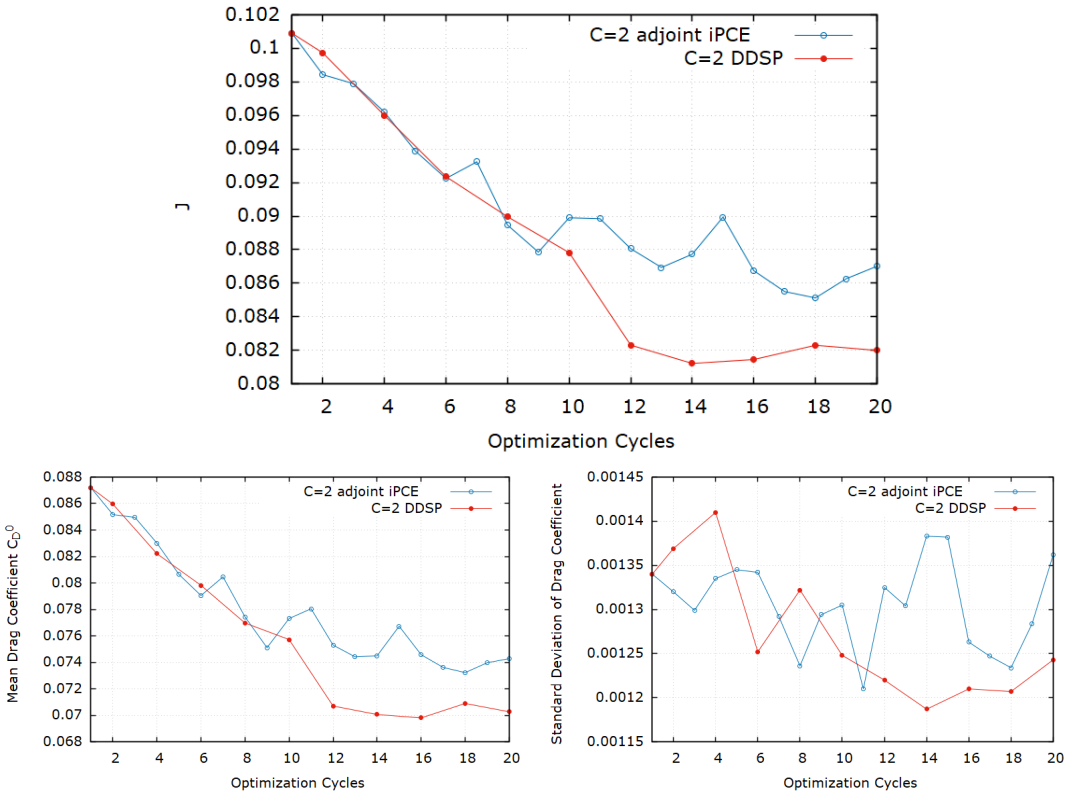


Figure 8.5: *Optimization results for $C = 2$. Value of the QoI (top), mean drag coefficient (bottom left) and standard deviation of drag coefficient (bottom right).*

8.4.3 Overall Conclusions Regarding the DDSP Method

Overall, it is clear that the DDSP method is preferable probably to the adjoint iPCE method, mainly due to the huge gain in terms of computational cost it involves and its simplicity (no adjoint iPCE programming needed). The limitations of the method have to be taken into account of course, as it generally fails for low chaos orders; that being said, it may sometimes be preferable to apply the DDSP for a higher chaos order, than the adjoint iPCE method for a lower one. Finally, the DDSP method limits the choices regarding the selection of the coefficients ζ_j of the objective function, as a least squares problem has to be solved and the resulting coefficients are those given from eq. 8.11; probably a small price to pay.

Chapter 9

Overview and Conclusions – Future Research Ideas

9.1 Overview

This work proposed an intrusive PCE method that also benefits from the simplicity of non–intrusive approaches. The changes in the software it requires are minimal and its applicability is wide, as it does not depend on the governing equations of a problem.

In **chapter 3**, several new definitions were given and some propositions were proven. This way, the theory behind the proposed iPCE method was established. It was shown that there is no need to derive the iPCE equations by hand, as is usually the case in 'conventional' intrusive approaches. Instead, non–intrusive operations were introduced, so that the residuals of the iPCE equations can be computed. Moreover, a numerical solution scheme was derived, which still does not ask for the explicit derivation of the iPCE equations, but is based on the solution scheme of their deterministic counterpart. Ways to drastically reduce the memory requirements and computational cost of the method were also presented. The proposed approach was not specific to any set of equations and was rather general; this is a huge advantage over other intrusive methods, that greatly depend on the set of governing equations.

In **chapter 4**, the proposed method was applied to aerodynamic problems. It was programmed for 2D and 3D flow problems, in laminar or turbulent flow conditions, using the Spalart–Allmaras turbulence model. Comparisons in terms of accuracy and/or computational cost were made with the Monte–Carlo and the niPCE method. It was found that the iPCE approach outperformed the other methods, as it was significantly faster and just as accurate.

In **chapter 5**, the continuous adjoint equations of the proposed iPCE method was derived. Again, the approach was general and not specific to the governing equations of a problem. Non–intrusive operations were again added to the method, so as to make it painless and easy to program, with as little interventions in the original deterministic code as possible.

In **chapter 7**, the previously mentioned continuous adjoint method was programmed and applied to an isolated airfoil in laminar flow conditons. Shape optimizations under uncertainty that aimed to minimize the drag coefficient were carried out. The optimized airfoil geometries were shown to have a robust performance in the presence of uncertainties, when compared to geometries that resulted from optimizations that did non acount for uncertainties.

In **chapter 8**, an alternative approach to compute the sensitivity derivatives of an objective function was proposed. The DDSP method replaced the adjoint iPCE equations with a deterministic adjoint problem. After several accuracy tests, the accuracy of the method was well–established. In addition to that, the DDSP method also allowed for looser convergence criteria of the iPCE equations. Thus, overall, the DDSP method offers a huge gain in computational cost, when compared to the adjoint iPCE equations. This was shown, as the method was applied to a 2D airfoil and compared to the adjoint iPCE solutions.

Overall, the conclusions of this diploma thesis are summarized as follows

- The proposed iPCE method combines the merits od both intrusive and non–intrusive approaches.
- The implementation of the method is painless, as the programming required is the least possible.
- No mathematical groundwork is required to derive and discretize the iPCE equations.
- The proposed approach is general, robust and applicable to any set of PDEs.
- The same is true for the proposed adjoint appoach and the DDSP method.
- The adjoint and the DDSP methods offer a computationaly affordable way to take uncertainties into account in optimization and robust design.

9.2 Proposals for Future Work

Regarding future research ideas, the following are proposed

- Application of the proposed iPCE approach using a method of estimating the Galerkin projections involved other than the Gauss Quadrature, such as least–squares. This could help speed up the 'non–intrusive' part of the method.
- Formulation of the method's continuous adjoint when there is uncertainty in the domain of the problem, i.e. shape uncertainties.
- Application of the adjoint iPCE approach (or the DDSP method) to real–world 3D problems.
- Use of the proposed iPCE approach as a surrogate model. For instance, the dependance of a QoI to the changes in the flows' boundary conditions can be found through its computed PCE coefficients.

- Use of arbitrary probability distributions, which result from curve–fitting of data points. The corresponding orthogonal polynomials can be created through the Gram–Schmidt process (see chapter 2).
- In all applications presented in this diploma thesis, the probability distributions of inputs/boundary conditions were arbitrarily chosen. Methods such as Bayesian UQ approaches can be used, so as to better define such distributions.

Appendices

Appendix A

Proof of Proposition 3.3.1

Let $\mathcal{D} := \{d_1, \ldots, d_d\}$ and $\mathcal{Q} := \{\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_d\}$ denote the sets of quadrature weights and nodes, corresponding to d value–sets of the uncertain variables $\boldsymbol{\xi} \in \mathbb{R}^m$, where $d = (C+1)^m$; C is the chosen chaos order.

Then, we assume that the chosen chaos order is such that $\boldsymbol{U}(\boldsymbol{\xi})$ is exactly equal to the truncated expansion of the flow variables $\boldsymbol{U}$ with q terms, namely,

$$\boldsymbol{U} = \sum_{i=0}^{q} \boldsymbol{U}^i Y_i(\boldsymbol{\xi}) \tag{A.1}$$

In what follows, let $\mathrm{G^q}\left[\boldsymbol{U}^{(p)}\right] \equiv [(\boldsymbol{U}^0)^{(p)}, \ldots, (\boldsymbol{U}^q)^{(p)}]^T$ denote the PCE coefficient fields at the p–th iteration, as found by the iPCE and $\mathrm{G^q}\left[\boldsymbol{U}'^{(p)}\right] \equiv [(\boldsymbol{U}'^0)^{(p)}, \ldots, (\boldsymbol{U}'^q)^{(p)}]^T$ denote the PCE coefficient fields at the p–th iteration, as found by the niPCE.

Upon completion of the p –th iteration of each non–intrusive run, $\boldsymbol{U}$ and $\boldsymbol{R}$ fields at each Gaussian node of $\mathcal{Q}$ are available; these are denoted by $\boldsymbol{U}'^{(p)}(\boldsymbol{\xi}_i)$ and $\boldsymbol{R}'^{(p)}(\boldsymbol{\xi}_i)$ respectively. Then

$$\begin{aligned}(\boldsymbol{U}'^g)^{(p)} &= \sum_{i=1}^{d} d_i Y_g(\boldsymbol{\xi}_i) \boldsymbol{U}'^{(p)}(\boldsymbol{\xi}_i) \\ (\boldsymbol{R}'^g)^{(p)} &= \sum_{i=1}^{d} d_i Y_g(\boldsymbol{\xi}_i) \boldsymbol{R}'^{(p)}(\boldsymbol{\xi}_i)\end{aligned} \tag{A.2}$$

To prove proposition 3.3.1, it suffices to show that

$$(\boldsymbol{U}^g)^{(p)} = (\boldsymbol{U}'^g)^{(p)} \;\text{ implies }\; (\boldsymbol{U}^g)^{(p+1)} = (\boldsymbol{U}'^g)^{(p+1)},\; g = 0, \ldots, q \tag{A.3}$$

The solution of eq. 3.8 for each $\boldsymbol{\xi} \in \mathcal{Q}$ leads to

$$\boldsymbol{U}'^{(p+1)}(\boldsymbol{\xi}_i) = \boldsymbol{U}'^{(p)}(\boldsymbol{\xi}_i) - \mathcal{J}^{-1} \boldsymbol{R}'^{(n)}(\boldsymbol{\xi}_i) \tag{A.4}$$

where $\mathcal{J} := \frac{\partial \boldsymbol{R}^{(p)}}{\partial \boldsymbol{U}^{(p)}}$. Assuming $\boldsymbol{U}_g^{(p)} = \boldsymbol{U}_g'^{(p)}$, which also results to

$$(\boldsymbol{R}^g)^{(p)} = (\boldsymbol{R}'^g)^{(p)},\ g = 0, \ldots, q \tag{A.5}$$

the application of eq. A.4 for all $\boldsymbol{\xi} \in \mathcal{Q}$, by considering eq. A.2, leads to

$$(\boldsymbol{U}'^g)^{(p+1)} = (\boldsymbol{U}^g)^{(p)} - \sum_{i=1}^{d} d_i Y_g(\boldsymbol{\xi}_i) \mathcal{J}^{-1}(\boldsymbol{\xi}_i) \boldsymbol{R}^{(p)}(\boldsymbol{\xi}_i) \tag{A.6}$$

Moreover

$$\mathrm{G^q}\left[\boldsymbol{U}^{(p+1)}\right] = \mathrm{G^q}\left[\boldsymbol{U}^{(p)}\right] - \mathrm{G^q}\left[\mathcal{J}^{-1}\right] \mathrm{G^q}\left[\boldsymbol{R}^{(p)}\right] \tag{A.7}$$

From eqs. A.6 and A.7, it can be seen that in order to prove that $(\boldsymbol{U}^g)^{(p+1)} = (\boldsymbol{U}'^g)^{(p+1)},\ g = 0, \ldots, q$, it suffices to show that

$$\sum_{i=1}^{d} d_i Y_g(\boldsymbol{\xi}_i) \mathcal{J}^{-1}(\boldsymbol{\xi}_i) \boldsymbol{R}^{(p)}(\boldsymbol{\xi}_i) = \mathrm{G^q}\left[\mathcal{J}^{-1}\right] \mathrm{G^q}\left[\boldsymbol{R}^{(p)}\right]$$

Thus

$$G^q[\mathcal{J}^{-1}] G^q[\boldsymbol{R}^{(p)}]|_g = \sum_{k=0}^{q} (J^{-1})^{gk} (\boldsymbol{R}^k)^{(p)} =$$

$$\sum_{i=1}^{d} d_i \mathcal{J}^{-1}(\boldsymbol{\xi}_i) Y_g(\boldsymbol{\xi}_i) \sum_{k=0}^{q} Y_k(\boldsymbol{\xi}_i) \sum_{j=1}^{d} d_j \boldsymbol{R}^{(p)}(\boldsymbol{\xi}_j) Y_k(\boldsymbol{\xi}_j) =$$

$$\sum_{i=1}^{d} d_i \mathcal{J}^{-1}(\boldsymbol{\xi}_i) Y_g(\boldsymbol{\xi}_i) \sum_{k=0}^{q} Y_k(\boldsymbol{\xi}_i) \boldsymbol{R}_k^{(p)} = \sum_{i=1}^{d} d_i \mathcal{J}^{-1}(\boldsymbol{\xi}_i) Y_g(\boldsymbol{\xi}_i) \boldsymbol{R}^{(p)}(\boldsymbol{\xi}_i)$$

Note that eq. A.1 holds only approximately due to truncation and so does eq. A.5. Therefore, it is expected that the results of the proposed iPCE formulation will tend to those of the corresponding niPCE, as chaos order increases. Also, since the correct solution $\mathrm{G^q}[\boldsymbol{U}]$ is obtained after all non–intrusive runs are completed, the iPCE will also converge after the last run is completed, that is, after $\max(n_1, \ldots, n_d)$ iterations.

Proposition 3.3.1 is an important property as it implies that the only prerequisite for the convergence of the iPCE equations is that the corresponding problem without uncertainties converges for all $\boldsymbol{\xi} \in \mathcal{Q}$. In other words, ensuring that the solver in eq. 3.8 converges for input uncertainties, taking the fixed values given by $\mathcal{Q}$ is sufficient for the iPCE equations to converge as well.

Appendix B

Discrete Adjoint of the iPCE Problem

The Discrete Adjoint of the iPCE equations is presented in this Appendix. Again, emphasis is laid on establishing a general, flexible method that is easy to program, given some existing software that solves the deterministic equations' adjoint problems.

B.1 Deterministic Discrete Adjoint Problem

The discrete primal residual equations are written as

$$\boldsymbol{R}(\boldsymbol{U}, \boldsymbol{b}) = \boldsymbol{0} \tag{B.1}$$

where $\boldsymbol{U}$ is their discrete solution, i.e. the field variables at every mesh node, and $\boldsymbol{b}$ is the array of design variables. Let $I(\boldsymbol{U}, \boldsymbol{b})$ be a chosen *discrete approximation* of the selected objective function. For instance, if the objective function is an integral, then $I(\boldsymbol{U}, \boldsymbol{b})$ is essentially a sum of terms involving $\boldsymbol{U}$ and $\boldsymbol{b}$. Then, the total derivative of I with respect to $\boldsymbol{b}$, $\delta \equiv \frac{\delta}{\delta \boldsymbol{b}}$ is

$$\delta I = \left(\frac{\partial I}{\partial \boldsymbol{U}} \right)^T \delta \boldsymbol{U} + \frac{\partial I}{\partial \boldsymbol{b}} \tag{B.2}$$

Also

$$\delta \boldsymbol{R} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} \delta \boldsymbol{U} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{b}} = \boldsymbol{0} \tag{B.3}$$

since $\boldsymbol{R} = \boldsymbol{0}$ regardless of the choice of $\boldsymbol{b}$. Solving eq.B.3 for $\delta \boldsymbol{U}$ and substituting into eq. B.2 leads to

$$\delta I = - \left(\frac{\partial I}{\partial \boldsymbol{U}} \right)^T \left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} \right)^{-1} \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{b}} + \frac{\partial I}{\partial \boldsymbol{b}}$$

which, through the discrete adjoint equation

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)^T \boldsymbol{\Psi} + \frac{\partial I}{\partial \boldsymbol{U}} = \mathbf{0} \tag{B.4}$$

is finally written as

$$\delta I = \boldsymbol{\Psi}^T \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{b}} + \frac{\partial I}{\partial \boldsymbol{b}} \tag{B.5}$$

B.2 Discrete Adjoint iPCE Problem

The discrete iPCE primal residual equations are written as

$$\mathrm{G}^{\mathrm{q}}[\boldsymbol{R}] = 0 \tag{B.6}$$

and $\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}]$ is their *discrete* solution. The objective function chosen in this case involves the spectral coefficients of the PCE of the deterministic objective function and its *discrete approximation* is

$$J = \sum_{j=0}^{q} \zeta_j |I^j| = \mathrm{G}^{\mathrm{q}}[\zeta]^T \mathrm{G}^{\mathrm{q}}[I] \tag{B.7}$$

where ζ_j are some user–defined weights, $\zeta := \sum_{j=0}^{q} \zeta_j sign(I^j) Y(\boldsymbol{\xi})$ and $I^j \equiv \langle I, Y_j \rangle$ are the spectral coefficients of the discrete deterministic objective function. To formulate the discrete adjoint equation, we first differentiate eq.B.6 with respect to the design variables (note that operators $\delta()$ and $\mathrm{G}^{\mathrm{q}}[]$ permute and recall proposition 3.5)

$$\delta(\mathrm{G}^{\mathrm{q}}[\boldsymbol{R}]) = \mathrm{G}^{\mathrm{q}}[\delta \boldsymbol{R}] = \mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right] \mathrm{G}^{\mathrm{q}}[\delta \boldsymbol{U}] + \mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{b}}\right] = \mathbf{0} \tag{B.8}$$

We then differentiate J and use propositions 3.5 and 3.6:

$$\begin{aligned}
\delta J &= \mathrm{G}^{\mathrm{q}}[\zeta]^T \mathrm{G}^{\mathrm{q}}\left[\left(\frac{\partial I}{\partial \boldsymbol{U}}\right)^T \delta \boldsymbol{U} + \frac{\partial I}{\partial \boldsymbol{b}}\right] \\
&= \mathrm{G}^{\mathrm{q}}[\zeta]^T \mathrm{G}^{\mathrm{q}}\left[-\left(\frac{\partial I}{\partial \boldsymbol{U}}\right)^T \left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)^{-1} \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{b}} + \frac{\partial I}{\partial \boldsymbol{b}}\right] \\
&= \mathrm{G}^{\mathrm{q}}\left[\zeta \frac{\partial I}{\partial \boldsymbol{U}}\right]^T \mathrm{G}^{\mathrm{q}}\left[-\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)^{-1}\right] \mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{b}}\right] + \mathrm{G}^{\mathrm{q}}[\zeta]^T \mathrm{G}^{\mathrm{q}}\left[\frac{\partial I}{\partial \boldsymbol{b}}\right] \\
&= \boldsymbol{\Psi}^T \mathrm{G}^{\mathrm{q}}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{b}}\right] + \mathrm{G}^{\mathrm{q}}[\zeta]^T \mathrm{G}^{\mathrm{q}}\left[\frac{\partial I}{\partial \boldsymbol{b}}\right]
\end{aligned} \tag{B.9}$$

where

$$\mathrm{G}^{\mathrm{q}}\left[\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)^{T}\right]\boldsymbol{\Psi}+\mathrm{G}^{\mathrm{q}}\left[\zeta\ \frac{\partial I}{\partial \boldsymbol{U}}\right]^{T}=\mathbf{0} \tag{B.10}$$

is the discrete adjoint equation, the solution of which allows for the calculation of the sensitivity derivatives, given from eq.B.9.

Appendix C

Gauss Quadrature Nodes and Weights

Below are summarized the values of the nodes and weights for GQ–based integration involving probability distributions, in the corresponding intervals of orthogonality. Recall that for a single variable and a chosen chaos order C, the order of the quadrature is given by

$$d = C + 1$$

which guarantees the exact evaluation of integrals involving two polynomials, each with degree up to C.

	Hermite		Legendre	
	Normal Distribution		Uniform Distribution	
	$w(\xi) = \frac{1}{\sqrt{2\pi}} e^{-\xi^2/2}$		$w(\xi) = \frac{1}{2}$	
	$\mathcal{E} = (-\infty, +\infty)$		$\mathcal{E} = (-1, 1)$	
d	ξ	ω	ξ	ω
2	1.00000000000000	0.50000000000000	0.57735026918963	0.50000000000000
2	−1.00000000000000	0.50000000000000	−0.57735026918963	0.50000000000000
3	0.00000000000000	0.66666666666667	0.00000000000000	0.44444444444444
3	1.73205080756888	0.16666666666667	0.77459666924148	0.27777777777778
3	−1.73205080756888	0.16666666666667	−0.77459666924148	0.27777777777778
4	0.74196378430273	0.45412414523192	0.33998104358486	0.32607257743127
4	−0.74196378430273	0.45412414523192	−0.33998104358486	0.32607257743127
4	2.33441421833898	0.04587585476807	0.86113631159405	0.17392742256873
4	−2.33441421833898	0.04587585476807	−0.86113631159405	0.17392742256873
5	0.00000000000000	0.53333333333331	0.00000000000000	0.28444444444444
5	1.35562617997427	0.22207592200559	0.53846931010568	0.23931433524968
5	−1.35562617997427	0.22207592200559	−0.53846931010568	0.23931433524968
5	2.85697001387281	0.01125741132772	0.90617984593866	0.11846344252809
5	−2.85697001387281	0.01125741132772	−0.90617984593866	0.11846344252809
6	0.61670659019260	0.40882846955603	0.66120938646626	0.18038078652407
6	−0.61670659019260	0.40882846955603	−0.66120938646626	0.18038078652407
6	1.88917587775371	0.08861574604194	0.93246951420315	0.08566224618959
6	−1.88917587775371	0.08861574604194	−0.93246951420315	0.08566224618959
6	3.32425743355212	0.00255578440206	0.23861918608320	0.23395696728635
6	−3.32425743355212	0.00255578440206	−0.23861918608320	0.23395696728635

Table C.1: *Hermite and Legendre GQ weights and nodes*

Εθνικο Μετσοβειο Πολυτεχνειο
Σχολη Μηχανολογων Μηχανικων
Τομεας Ρευστων
Εργαστηριο Θερμικων Στροβιλομηχανων
Παραλληλης ΥΡΔ & Βελτιστοποιησης

Επεμβατικό Ανάπτυγμα Πολυωνυμικού Χάους στην Υπολογιστική Ρευστοδυναμική και στη Βελτιστοποίηση Μορφής με τη Συζυγή Μέθοδο

Εκτενής Ελληνική Περίληψη Διπλωματικής Εργασίας

Μιχαήλ Χατζημανωλάκης

Επιβλέπων: Κ.Γ. Γιαννάκογλου, Καθηγητής

Αθήνα, 2018

Περιεχόμενα

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Ποσοτικοποίηση της Αβεβαιότητας

Στις περιπτώσεις που η στοχαστικότητα στα προβλήματα ρευστομηχανικής αμελείται, οι κώδικες Υπολογιστικής Ρευστομηχανικής (ΥΡΔ) προλέγουν ροές πολύ αποτελεσματικά. Όμως, σε πλήθος περιπτώσεων οι αβέβαιες παράμετροι έχουν σημαντική επίδραση στην απόδοση ενός συστήματος. Για παράδειγμα, η απόδοση ενός συμπιεστή αλλάζει δραστικά αν μεταβληθεί η γωνία της ροής στην είσοδό του. Για τον λόγο αυτό, αναπτύσσονται μέθοδοι Ποσοτικοποίησης Αβεβαιότητας (Uncertainty Quantification, UQ), ώστε να μπορεί να γίνει ποσοτική εκτίμηση της επίδρασης αβέβαιων παραμέτρων στην Ποσότητα Ενδιαφέροντος (Quantity of Interest, QoI)

Ορισμένες μέθοδοι UQ είναι οι εξής:

- Μέθοδοι Στοχαστικής Δειγματοληψίας (Stochastic Sampling)
  Είναι οι πιο ακριβείς αλλά και υπολογιστικά ακριβές μέθοδοι, αφού περιλαμβάνουν τη λήψη πολλών δειγμάτων, δηλαδή λύσεων του προβλήματος χωρίς αβεβαιότητες, ώστε να μπορέσει να εκτιμηθεί η κατανομή πιθανότητας της εξόδου που ενδιαφέρει το μηχανικό. Τέτοιες μέθοδοι είναι οι Monte Carlo, quasi– Monte Carlo και η Latin Hypercube , [1, 2, 3].

- Μέθοδοι Στατιστικών Ροπών (Method of Moments)
  Σε αυτήν τη μέθοδο χρησιμοποιείται το Taylor ανάπτυγμα της XoI, μέσω του οποίου εκφράζονται οι στατιστικές ροπές της QoI συναρτήσει των παραγώγων της ως προς τις αβέβαιες μεταβλητές, [4, 5].

- Στοχαστική Παράθεση (Stochastic Collocation)
  Σε αυτήν τη μέθοδο η QoI εκφράζεται ως ανάπτυγμα με βάση τα πολυώνυμα Legendre . Με λήψη τιμών της για διάφορες τιμές των αβέβαιων παραμέτρων γίνεται κατάλληλη παρεμβολή και προσεγγίζεται η στατιστική συμπεριφορά της [6, 7, 8].

- Φασματικές Μέθοδοι (Spectral Methods)
  Στις φασματικές μεθόδους, η QoI εκφράζεται μέσω συναρτήσεων βάσης που σχηματίζουν τον χώρο/φάσμα των στοχαστικών εισόδων. Ένα παράδειγμα φασματικής μεθόδου είναι το ανάπτυγμα Karhunen–Loève , [9, 10]. Το Ανάπτυγ-

μα Πολυωνυμικού Χάους (Polynomial Chaos Expansion, PCE) είναι μία άλλη μέθοδος, [11, 12]. Διακρίνεται σε επεμβατικό και μη–επεμβατικό, ανάλογα με το αν απαιτείται ή όχι τροποποίηση του λογισμικού επίλυσης των εξισώσεων [13, 14, 15, 16, 17, 18, 19, 20]. Εκτενής σύγκριση των δύο μεθόδων παρουσιάζεται στο [21].

1.2 Βελτιστοποίηση Στιβαρού Σχεδιασμού

Οι αβεβαιότητες στο περιβάλλον ή/και στις συνθήκες λειτουργίας ενός συστήματος δημιουργούν την ανάγκη για Στιβαρό Σχεδιασμό (Robust Desgin) , δηλαδή τον σχεδιασμό συστημάτων των οποίων η απόδοση δεν αλλάζει σημαντικά όταν μεταβάλλεται το περιβάλλον τους. Μαθηματικά, αντί της ελαχιστοποίησης μίας συνάρτησης F, ο Στιβαρός Σχεδιασμός ελαχιστοποιεί μια ποσότητα της μορφής

$$\mu_F + k\sigma_F \ , \ k \in \mathbb{R}^+ \tag{1.1}$$

όπου μ_F είναι η μέση τιμή και σ_F η τυπική απόκλιση της F, ενώ το k είναι ένας συντελεστής βαρύτητας που δίνει ο χρήστης. Μία μέθοδος UQ επιτρέπει τον υπολογισμό των τιμών των συναρτήσεων ως αυτή της εξ. 1.1. Με τη χρήση στοχαστικών μεθόδων βελτιστοποίησης, όπως οι εξελικτικοί αλγόριθμοι, αυτό είναι αρκετό για να γίνει η βελτιστοποίηση [22, 23, 24, 25, 26]. Η βελτιστοποίηση μπορεί φυσικά να γίνει και με χρήση της κλίσης της συνάρτησης–στόχου (όπως αυτή της εξ. 1.1) ως προς τις μεταβλητές σχεδιασμού, η οποία υπολογίζεται μέσω συζυγών τεχνικών [27, 28], όπως γίνεται και στη διπλωματική αυτή εργασία, όπου προτείνεται μία συνεχής συζυγής προσέγγιση για τον υπολογισμό παραγώγων ευαισθησίας συναρτήσεων όπως αυτή της εξ. 1.1.

Κεφάλαιο 2

Ορθογώνια Πολυώνυμα και PCE

Σε αυτό το κεφάλαιο εισάγονται τα ορθογώνια πολυώνυμα, που αποτελούν βασικό στοιχείο της θεωρίας του PCE, το οποίο αναλύεται στη συνέχεια.

2.1 Ορθογώνια Πολυώνυμα

Μίας μεταβλητής

***Ορισμός* 2.1.1.** *Μια σειρά πολυωνύμων* $\{p_n(\xi)\}_{n=0}^{\infty}$ *με degree* $[p_n] = n$ *ονομάζεται ορθογώνια ως προς τη συνάρτηση* $w(\xi)$ *στο διάστημα* (a, b) *αν*

$$\langle p_n, p_m \rangle \equiv \int_a^b p_n(\xi) p_m(\xi) w(\xi) d\xi = \delta_{mn} \ \langle p_n, p_n \rangle \tag{2.1}$$

όπου δ_{mn} *είναι το δέλτα του Κρόνεκερ. Αν* $< p_n, p_n >= 1 \ \forall n \in \mathbb{N}$*, τότε τα πολυώνυμα λέγονται ορθοκανονικά.*

Πολλών μεταβλητών

Υποθέτουμε m σειρές ορθογωνίων πολυωνύμων μίας μεταβλητής $p^k \equiv \{p_n^k(\xi_k)\}_{n=0}^{\infty}, k = 1, \dots, m$. Κάθε σειρά είναι ορθογώνια ως προς μία συνάρτηση βάρους $w_k(\xi_k)$ με πεδίο ορισμού $\mathcal{E}_k$. Μεταξύ οποιονδήποτε τέτοιων σειρών ορίζεται ένα τανυστικό γινόμενο, ως εξής:

***Ορισμός* 2.1.2.** *Το τανυστικό γινόμενο δύο σειρών συναρτήσεων* $A = \{a_n(\xi_1)\}_{n=0}^{\infty}$ *και* $B = \{b_n(\xi_2)\}_{n=0}^{\infty}$ *ορίζεται ως*

$$A \otimes B := \{a_{n_1}(\xi_1) b_{n_2}(\xi_2)\}_{n_1,n_2=0}^{\infty} = \{a_0 b_0, a_1 b_0, a_0 b_1, a_1 b_1, a_2 b_0, a_0 b_2, \dots\} \tag{2.2}$$

Συνεπώς, μπορεί να οριστεί η ακόλουθη σειρά πολυωνύμων m μεταβλητών

$$Y \equiv \{Y_n\}_{n=0}^{\infty} := \otimes_{k=1}^{m} p^k = \{p_{n_1}^1(\xi_1) p_{n_2}^2(\xi_2) \dots p_{n_m}^m(\xi_m)\}_{n_1,n_2,\dots,n_m=0}^{\infty} \tag{2.3}$$

Αυτά τα πολυώνυμα είναι ορθογώνια ως προς το εσωτερικό γινόμενο

$$< f, g >_W = \int_{\mathcal{E}} fgW d\xi_1 \dots d\xi_m \quad , \quad W := \prod_{j=1}^{m} w_j(\xi_j) \tag{2.4}$$

το οποίο αποδεικνύεται ως εξής

$$\int_{\mathcal{E}} Y_k Y_l W dx_1 \dots dx_m =$$

$$\int_{\mathcal{E}_1} p^1_{n_1}(x_1) p^1_{l_1}(x_1) w_1 dx_1 \cdots \int_{\mathcal{E}_m} p^m_{n_1}(x_m) p^m_{l_1}(x_m) w_m dx_m =$$

$$\delta_{n_1 l_1} < p_{n_1}, p_{n_1} >_{w_1} \dots \delta_{n_m l_m} < p_{n_m}, p_{n_m} >_{w_m} = \delta_{kl} < Y_k, Y_k >_W$$

2.2 PCE μιας συνάρτησης

Έστω $\phi = \phi(\boldsymbol{\xi})$ μία συνάρτηση του $\boldsymbol{\xi}$.
***Ορισμός* 2.2.1.** *Το ανάτυγμα πολυωνυμικού χάους PCE του $\phi(\boldsymbol{\xi})$ ορίζεται ως η άπειρη σειρά*

$$\phi(\xi) = \sum_{j=0}^{\infty} \phi^j Y_j(\boldsymbol{\xi}) \tag{2.5}$$

όπου τα πολυώνυμα Y_j είναι ορθογώνια ως προς τη συνάρτηση $W(\boldsymbol{\xi}) := \prod_{j=1}^{m} w_j(\xi_j)$ και οι συντελεστές της σειράς δίνονται από τη σχέση

$$\phi^j := \langle \phi(\xi), Y_j \rangle \tag{2.6}$$

Μια βασική ιδιότητα είναι η εξής
Πρόταση 2.2.1. *Οι φασματικοί συντελεστές του PCE του $\phi(\boldsymbol{\xi})$ ικανοποιούν τις σχέσεις*

$$E[\phi] \equiv \mu_\phi = \phi^0$$
$$Var[\phi] \equiv \sigma^2_\phi = \sum_{j=1}^{\infty} (< Y_j, Y_j > \phi^j)^2 \tag{2.7}$$

το οποίο αποδεικνύεται εύκολα, αφού

$$\mu_\phi \equiv \int_{\mathcal{E}} \phi W d\boldsymbol{\xi} = \int_{\mathcal{E}} \phi Y_0 W d\boldsymbol{\xi} = \phi^0$$

$$\sigma_\phi^2 \equiv \int_{\mathcal{E}} (\phi - \mu_\phi)^2 W d\boldsymbol{\xi} = \int_{\mathcal{E}} \left(\sum_{j=0}^{\infty} \phi^j Y_j(\xi) - \phi^0 \right)^2 W d\boldsymbol{\xi} =$$

$$\sum_{j=1}^{\infty}\sum_{k=1}^{\infty} \phi^j \phi^k \int_{\mathcal{E}} Y_j(\xi) Y_k(\xi) W d\boldsymbol{\xi} = \sum_{j=1}^{\infty}\sum_{k=1}^{\infty} \phi^j \phi^k \delta_{jk} < Y_j, Y_k >= \sum_{j=1}^{\infty} (< Y_j, Y_j > \phi^j)^2$$

2.3 Μη–Επεμβατικό PCE (niPCE)

Στο niPCE, η QoI γράφεται ως

$$F = \sum_{j=0}^{\infty} F^j Y_j(\boldsymbol{\xi}) \tag{2.8}$$

Επομένως

$$E[F] = F^0$$

$$Var[F] = \sum_{j=1}^{\infty} (< Y_j, Y_j > F^j)^2 \tag{2.9}$$

Το παραπάνω ανάπτυγμα αποκόπτεται, κρατώντας πεπερασμένο αριθμό $(q+1)$ όρων, ο οποίος καθορίζεται από την επιλογή τάξης χάους C και από το πλήθος m των αβέβαιων μεταβλητών και δίνεται από

$$q + 1 = \frac{(C+m)!}{C!m!} \tag{2.10}$$

Τότε

$$F = \sum_{j=0}^{q} F^j Y_j(\boldsymbol{\xi}) \tag{2.11}$$

που σημαίνει ότι οι φασματικοί συντελεστές που πρέπει να υπολογιστούν είναι οι

$$F^j \equiv \langle F, Y_j \rangle \equiv \int_{\mathcal{E}} F Y_j W d\boldsymbol{\xi} \quad , j = 0, \ldots, q \tag{2.12}$$

Αυτό γίνεται με αριθμητική ολοκλήρωση, που απαιτεί τη λήψη ορισμένων τιμών της F με επιμέρους τρεξίματα του λογισμικού αξιολόγησης, δηλαδή του κώδικα Υπολογιστικής Ρευστοδυναμικής στην περίπτωση της εργασίας αυτής (ή οτιδήποτε άλλο, στη γενική περίπτωση).

2.4 Επεμβατικό PCE (iPCE)

Στο iPCE γράφουμε

$$\boldsymbol{U} = \sum_{j=0}^{q} \boldsymbol{U}^j Y(\boldsymbol{\xi}) \tag{2.13}$$

όπου τα πεδία $\boldsymbol{U}^j$, $j = 0, \ldots, q$ είναι οι άγνωστοι του προβλήματος. Η έκφραση του $\boldsymbol{U}$ εισάγεται τότε στην εξίσωση $\boldsymbol{R}(\boldsymbol{U}) = \boldsymbol{0}$ του προβλήματος χωρίς αβεβαιότητες

$$\boldsymbol{R}\left(\sum_{j=0}^{q} \boldsymbol{U}^j Y(\boldsymbol{\xi})\right) = \boldsymbol{0} \tag{2.14}$$

και κατόπιν εφαρμόζονται προβολές Galerkin, οπότε προκύπτουν οι εξισώσεις

$$\int_{\mathcal{E}} \boldsymbol{R}\left(\sum_{j=0}^{q} \boldsymbol{U}^j Y(\boldsymbol{\xi})\right) Y_k W d\boldsymbol{\xi} = \boldsymbol{0} \ , k = 0, \ldots, q \tag{2.15}$$

που επιλύονται αριθμητικά αφού γίνουν αλλαγές στο λογισμικό επίλυσης των εξισώσεων χωρίς αβεβαιότητες.

Κεφάλαιο 3

Η Προτεινόμενη iPCE Προσέγγιση

Σε αυτό το κεφάλαιο παρατίθενται διάφοροι ορισμοί και προτάσεις που αναπτύχθηκαν για τις ανάγκες της προτεινόμενης iPCE μεθόδου. Όλες οι σειρές πολυωνύμων θα θεωρούνται ορθοκανονικές από εδώ και στο εξής, $\langle Y_n, Y_n \rangle = 1$, χωρίς βλάβη γενικότητας.

3.1 Μερικοί Ορισμοί

Σε όλους τους παρακάτω ορισμούς υποθέτουμε ένα σύνολο m ανεξάρτητων αβέβαιων μεταβλητών $\boldsymbol{\xi} \in \mathbb{R}^m$, με συναρτήσεις κατανομής πιθανότητας $w_k(\xi_k)$ και πεδία ορισμού $\mathcal{E}_k$. Επίσης, υποθέτουμε ένα σύνολο πολυωνύμων $Y = \{Y_n\}_{n=0}^{\infty}$ που είναι ορθογώνια ως προς τη συνάρτηση $W = \prod_{j=1}^{m} w_j$ στο $\mathcal{E} = \prod_{j=1}^{m} \mathcal{E}_j$.

***Ορισμός* 3.1.1** (Προβολή Galerkin βαθμωτού μεγέθους)**.** *Η προβολή Galerkin ενός βαθμωτού μεγέθους $\phi(\boldsymbol{\xi})$ στο πολυώνυμο Y_j ορίζεται ως*

$$\phi^j := \int_{\mathcal{E}} \phi Y_j W d\boldsymbol{\xi} \tag{3.1}$$

***Ορισμός* 3.1.2** (Προβολή Galerkin διανύσματος)**.** *Για κάθε διάνυσμα $\boldsymbol{U}(\boldsymbol{\xi}) = [U_1(\boldsymbol{\xi}), ..., U_n(\boldsymbol{\xi})]^T \in \mathbb{R}^n$, η προβολή Galerkin του, τάξης q, ορίζεται ως*

$$\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}] := [\boldsymbol{U}^0, \boldsymbol{U}^1, ..., \boldsymbol{U}^q]^T \tag{3.2}$$

με $\boldsymbol{U}^k = [U_1^k, U_2^k, ..., U_n^k]^T \in \mathbb{R}^n \ , \ k = 0, \ldots, q.$

Σημειώνεται ότι η εφαρμογή του τελεστή $\mathrm{G}^{\mathrm{q}}[]$ σε ένα βαθμωτό μέγεθος είναι ειδική περίπτωση του παραπάνω ορισμού, για $n = 1$. Αν $\phi = \phi(\boldsymbol{\xi}) \in \mathbb{R}$, τότε $\mathrm{G}^{\mathrm{q}}[\phi] = [\phi^0, \ldots, \phi^q]^T$.

***Ορισμός* 3.1.3** (Προβολή Galerkin πίνακα)**.** *Για κάθε πίνακα $A \in \mathbb{R}^{n\times n}$ με στοιχεία A_{ij}=$A_{ij}(\boldsymbol{\xi})$,η προβολή Galerkin του, τάξης q, ορίζεται ως το block μητρώο*

$$\mathrm{G}^{\mathrm{q}}[A] = \begin{bmatrix} A^{00} & A^{01} & \dots & A^{0q} \\ A^{10} & A^{11} & \dots & A^{1q} \\ \vdots & \vdots & \vdots & \vdots \\ A^{q0} & A^{q1} & \dots & A^{qq} \end{bmatrix} \tag{3.3}$$

όπου το (i,j) στοιχείο του $A^{\lambda\mu} \in \mathbb{R}^{n\times n}$ δίνεται από τη σχέση

$$A_{ij}^{\lambda\mu} := \int_{\mathcal{E}} A_{ij} Y_\lambda Y_\mu W d\boldsymbol{\xi} = \sum_{\rho=0}^{\infty} A_{ij}^{\rho} < Y_\rho, Y_\lambda, Y_\mu > \tag{3.4}$$

όπου $< Y_\rho, Y_\lambda, Y_\mu >:= \int_{\mathcal{E}} Y_\rho Y_\lambda Y_\mu W d\boldsymbol{\xi}$.

Στις εφαρμογές της προτεινόμενης μεθόδου, σε όλες τις ποσότητες εφαρμόζεται το ίδιο σχήμα αποκοπής (δηλαδή σε κάθε ανάπτυγμα διατηρείται ο ίδιος αριθμός $q+1$ όρων). Σε αυτή την περίπτωση, αποδεικνύονται οι ακόλουθες προτάσεις.

Π*ρόταση* 3.1.1. *Αν τα αναπτύγματα των συνιστωσών των A και $\boldsymbol{U}$ αποκοπούν στους $q+1$ όρους, δηλαδή*

$$A_{ij}=\sum_{k=0}^{q} A_{ij}^k Y_k(\boldsymbol{\xi}) \quad \textit{ανδ} \quad U_j=\sum_{k=0}^{q} U_j^k Y_k(\boldsymbol{\xi}) \quad \textit{ωιτη} \quad i,j=1,\dots,n$$

αποδεικνύεται ότι

$$\mathrm{G}^{\mathrm{q}}[A\boldsymbol{U}] = \mathrm{G}^{\mathrm{q}}[A]\,\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}] \tag{3.5}$$

Απόδειξη. Έστω $\boldsymbol{f}=A\boldsymbol{U}$ ή $f_i=A_{ij}U_j$. Τότε, για κάθε $0 \le p \le q$

$$f_i^p = (A_{ij}U_j)^p \equiv \int_{\mathcal{E}} A_{ij}U_j Y_p W d\boldsymbol{\xi} = U_j^\rho \int_{\mathcal{E}} A_{ij} Y_\rho Y_p W d\boldsymbol{\xi} = U_j^\rho A_{ij}^{\rho p}$$

που είναι το p-οστό στοιχείο του $\mathrm{G}^{\mathrm{q}}[A]\,\mathrm{G}^{\mathrm{q}}[\boldsymbol{U}]$. □

Π*ρόταση* 3.1.2. *Για δύο διανύσματα $\boldsymbol{g}=(g_1(\boldsymbol{\xi}),\dots)$ ανδ $\boldsymbol{h}=(h_1(\boldsymbol{\xi}),\dots)$ και μία σταθερά $\lambda(\boldsymbol{\xi})$, ισχύει ότι*

$$\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{g}^T\right]\mathrm{G}^{\mathrm{q}}[\lambda\boldsymbol{h}] = (\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{g}^T\boldsymbol{h}\right])^T\,\mathrm{G}^{\mathrm{q}}[\lambda] \tag{3.6}$$

αν τα PCE τους αποκοπούν μετά από $q+1$ όρους, δηλαδή

$$g_i = \sum_{j=0}^{q} g_i^j Y_j(\boldsymbol{\xi}) \quad , \quad h_i = \sum_{j=0}^{q} h_i^j Y_j(\boldsymbol{\xi}) \quad , \quad \lambda = \sum_{j=0}^{q} \lambda^j Y_j(\boldsymbol{\xi})$$

Απόδειξη.

$$\mathrm{G}^{\mathrm{q}}\left[\boldsymbol{g}^T\right]\mathrm{G}^{\mathrm{q}}[\lambda\boldsymbol{h}] = (\boldsymbol{g^j})^T(\lambda\boldsymbol{h})^j = (\boldsymbol{g^j})^T\lambda^k\boldsymbol{h}^i < Y_k, Y_i, Y_j >=$$

$$(\boldsymbol{g^j})^T \boldsymbol{h}^i < Y_k, Y_i, Y_j > \lambda^k = (\mathrm{G^q}\left[\boldsymbol{g}^T \boldsymbol{h}\right])^T \mathrm{G^q}\left[\lambda\right]$$

□

Οι προηγούμενες προτάσεις είναι βασικές για την παραγωγή του αριθμητικού σχήματος επίλυσης των iPCE εξισώσεων. Διευκολύνουν επίσης την παραγωγή των συζυγών iPCE εξισώσεων, στο κεφάλαιο 5.

3.2 Αριθμητική Επίλυση των iPCE Εξισώσεων

Έστω πρόβλημα από n ΜΔΕ, που γράφεται σε διακριτή μορφή ως

$$\boldsymbol{R}(\boldsymbol{U}) = \boldsymbol{0} \tag{3.7}$$

Για μη–γραμμικά προβλήματα, το σύστημα 3.7 επιλύεται μέσω του επαναληπτικού σχήματος (γνωστού και ως Δέλτα Διατύπωση)

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)_{old} \Delta \boldsymbol{U} = -\left(\boldsymbol{R}\right)_{old} \quad , \quad \Delta \boldsymbol{U} = \boldsymbol{U}_{new} - \boldsymbol{U}_{old} \tag{3.8}$$

που λύνεται ως προς $\Delta \boldsymbol{U}$ και ακολουθείται από την ακόλουθη ανανέωση

$$\boldsymbol{U}_{new} = \boldsymbol{U}_{old} + \Delta \boldsymbol{U} \tag{3.9}$$

των τιμών του $\boldsymbol{U}$ σε κάθε κόμβο του πλέγματος. Μετά από αυτό το βήμα, το σύστημα κατασκευάζεται ξανά και επιλύεται, μέχρι τη σύγκλιση (αρκετά μικρές τιμές του $\boldsymbol{R}$).

Η παραπάνω διαδικασία εφαρμόζεται στις iPCE εξισώσεις, εξ. 2.15, που με απλή αλλαγή συμβολισμού γράφονται ως

$$\mathrm{G^q}\left[\boldsymbol{R}\right] = \boldsymbol{0} \tag{3.10}$$

Λόγω της εξ. 3.8 και της πρότασης 3.5

$$\mathrm{G^q}\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right] \mathrm{G^q}\left[\Delta \boldsymbol{U}\right] = -\,\mathrm{G^q}\left[\boldsymbol{R}\right] \tag{3.11}$$

Το δεξί μέλος της εξ. 3.11 υπολογίζεται με αριθμητική ολοκλήρωση των υπολοίπων $\boldsymbol{R}$ του μη-στοχαστικού προβλήματος. Το ίδιο συμβαίνει και με το αριστερό μέλος, που βρίσκεται με ολοκλήρωση του $\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}$.

3.3 Εξοικονόμηση Μνήμης και Υπολογιστικού Κόστους στο iPCE

Σε αυτήν την ενότητα αναλύεται ένας τρόπος επιτάχυνσης της επίλυσης των iPCE εξισώσεων, ο οποίος οδηγεί σε εξοικονόνηση μνήμης και υπολογιστικού χρόνου. Η

εξ. 3.11 ξαναγράφεται σε μητρωική γραφή ως

$$\begin{bmatrix} \mathcal{J}^{00} & \mathcal{J}^{01} & \dots & \mathcal{J}^{0q} \\ \mathcal{J}^{10} & \mathcal{J}^{11} & \dots & \mathcal{J}^{1q} \\ \vdots & \vdots & \vdots & \vdots \\ \mathcal{J}^{q0} & \mathcal{J}^{q1} & \dots & \mathcal{J}^{qq} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{U}^0 \\ \Delta \boldsymbol{U}^1 \\ \vdots \\ \Delta \boldsymbol{U}^q \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}^0 \\ \boldsymbol{R}^1 \\ \vdots \\ \boldsymbol{R}^q \end{bmatrix} \tag{3.12}$$

όπου $\mathcal{J} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}$. Το σύστημα 3.12 μπορεί να διασπαστεί σε μικρότερα συστήματα. Με αυτό τον σκοπό, η μέση τιμή $\boldsymbol{U}^0$ των ροϊκών μεταβλητών προσεγγίζεται από το $\boldsymbol{U}$ πεδίο που προκύπτει από μία επίλυση του προβλήματος χωρίς αβεβαιότητες εξ. 3.7. Η επίλυση αυτή γίνεται θέτοντας $\boldsymbol{\xi} = \boldsymbol{\xi}_z$, όπου οι συνιστώσες του $\boldsymbol{\xi}_z$ είναι οι ρίζες όλων των ορθοκανονικών πολυωνύμων πρώτου βαθμού. Τότε, το σφάλμα της προσέγγισης είναι

$$\boldsymbol{U}(\boldsymbol{\xi}_z) - \boldsymbol{U}^0 = \sum_{i=1}^{q_1} \boldsymbol{U}^i Y_i(\boldsymbol{\xi}_z) + \sum_{i=q_1+1}^{\infty} \boldsymbol{U}^i Y_i(\boldsymbol{\xi}_z) = \sum_{i=q_1+1}^{\infty} \boldsymbol{U}^i Y_i(\boldsymbol{\xi} = \boldsymbol{\xi}_z) \tag{3.13}$$

όπου το q_1 βρίσκεται θέτοντας $C = 1$ στην εξ. 2.10.

Επιπλέον, για $C=1$,

$$\mathcal{J}_{ij}^{\lambda\mu} = \sum_{\rho=0}^{q_1} \mathcal{J}_{ij}^{\rho} < Y_\rho, Y_\lambda, Y_\mu >= \delta_{\lambda\mu} \mathcal{J}_{ij}^{00} \tag{3.14}$$

αφού

$$< Y_\rho, Y_\lambda, Y_\mu >= \delta_{0\rho}\delta_{\lambda\mu} \ , \quad 1 \leq \lambda, \ \mu \leq q_1 \ , \quad 0 \leq \rho \leq q_1 \tag{3.15}$$

Άρα, η εξ. 3.12 λαμβάνει τη μορφή

$$\begin{bmatrix} \mathcal{J}_{ij}^{00} & \mathcal{J}^{01} & \mathcal{J}^{02} & \dots & \mathcal{J}^{0q_1} \\ \mathcal{J}^{10} & \mathcal{J}_{ij}^{00} & \mathbf{0} & \dots & \mathbf{0} \\ \mathcal{J}^{20} & \mathbf{0} & \mathcal{J}_{ij}^{00} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathcal{J}^{q_1 0} & \mathbf{0} & \mathbf{0} & \dots & \mathcal{J}_{ij}^{00} \end{bmatrix} \begin{bmatrix} \Delta \boldsymbol{U}^0 \\ \Delta \boldsymbol{U}^1 \\ \Delta \boldsymbol{U}^2 \\ \vdots \\ \Delta \boldsymbol{U}^{q_1} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}^0 \\ \boldsymbol{R}^1 \\ \boldsymbol{R}^2 \\ \vdots \\ \boldsymbol{R}^{q_1} \end{bmatrix} \tag{3.16}$$

Υποθέτοντας ότι το $\boldsymbol{U}^0$ έχει προσεγγιστεί καλά, προκύπτει το συμπέρασμα ότι $\Delta \boldsymbol{U}^0 \approx \mathbf{0}$, το οποίο δικαιολογεί την απόφαση να κρατηθούν μόνο τα διαγώνια blocks του πίνακα της εξ. 3.16. Το απλοποιήμενο σύστημα αποτελείται από q_1+1 συστήματα, διάστασης ίδιας με αυτή του προβλήματος χωρίς αβεβαιότητες, με ίδιο αριστερό και διαφορετικό δεξί μέλος.

Επίσης, δεν χρειάζεται να υπολογιστεί το $\mathcal{J}_{ij}^{00}$, αφού μπορεί να προσεγιστεί από το $\mathcal{J}$ που υπολογίστηκε στην τελευταία επανάληψη της επίλυσης του προβλήματος χωρίς αβεβαιότητες που έδωσε την προσέγγιση του $\boldsymbol{U}^0$. Τα προηγούμενα βήματα υπολογίζουν τους συντελεστές PCE των ροϊκών μεγεθών για $C = 1$. Αν $C > 1$, η διαδικασία είναι παρόμοια. Θέτοντας $q = q(C)$ και υποθέτοντας ότι οι πρώτοι $q(C-1)$ όροι είναι γνωστοί, οι υπόλοιποι $q(C) - q(C-1)$ όροι βρίσκονται χρησιμοποιώντας την ίδια προσέγγιση, δηλαδή κρατώντας μόνο τα διαγώνια blocks του.

Κεφάλαιο 4

Εφαρμογή της Μεθόδου iPCE σε Προβλήματα Αεροδυναμικής

Σε αυτό το κεφάλαιο παρουσιάζεται η εφαρμογή της προτεινόμενης μεθόδου και ελέγχεται η ταχύτητα και η ακρίβειά της. Η μέθοδος προγραμματίστηκε για τις εξισώσεις RANS συμπιεστού ρευστού, μαζί με το μοντέλο τύρβης μίας εξίσωσης, των Spalart–Allmaras, [29]. Ο προγραμματισμός έγινε σε οικείο λογισμικό βασισμένο σε πεπερασμένους όγκους και την κεντροκομβική διατύπωσή τους.

Μεμονωμένη Αεροτομή

Επιλύεται η τυρβώδης ροή γύρω από μια αεροτομή, με χρήση του μοντέλου μίας εξίσωσης των Spalart–Allmaras ;;. Οι QoI είναι οι συντελεστές άνωσης και οπισθέλκουσας της αεροτομής, ενώ η αβεβαιότητα εισάγεται στις εξισώσεις μέσω των οριακών συνθηκών της γωνίας και του αριθμού Mach της ελεύθερης ροής και του αριθμού Reynolds που βασίζεται στη χορδή της αεροτομής a_∞, M_∞ και Re, αντίστοιχα. Οι κατανομές πιθανότητας που επιλέχθηκαν είναι οι

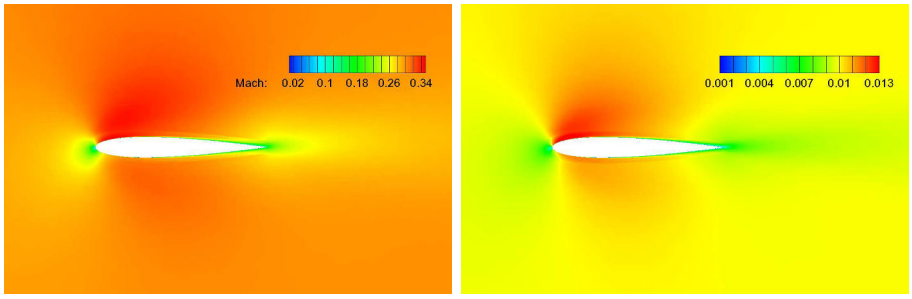
$$a_\infty \sim \mathcal{U}(1.5^o, 2.5^o) \qquad M_\infty \sim N(0.3, 0.01) \qquad Re \sim N(10^6, 2.5 \cdot 10^4)$$

όπου $\mathcal{N}(\mu, \sigma)$ συμβολίζει την κανονική κατανομή με μέση τιμή μ και τυπική απόκλιση σ ενώ $\mathcal{U}(a, b)$ συμβολίζει την ομοιόμορφη κατανομή στο διάστημα $[a, b]$.

Στον πίνακα 4.1 συνοψίζονται τα αποτελέσματα της προτεινόμενης μεθόδου (οι δύο στατιστικές ροπές των QoI που επιλέχθηκαν) και γίνεται σύγκρισή τους με αυτά του niPCE. Οι χρόνοι έχουν αδιαστατοποιηθεί με το υπολογιστικό κόστος του iPCE για $C = 1$. Είναι προφανές ότι οι δύο μέθοδοι δίνουν ίδια πρακτικά αποτελέσματα, όμως το iPCE είναι αρκετά γρηγορότερο. Η μέση τιμή του πεδίου του αριθμού Mach και η αντίστοιχη τυπική απόκλιση φαίνονται στο σχ. 4.1, για $C = 1$.

	iPCE	niPCE	iPCE	niPCE	iPCE	niPCE
	$C=1$		$C=2$		$C=3$	
μ_{C_L}	0.095598	0.095567	0.095591	0.095600	0.095611	0.095598
σ_{C_L}	0.013534	0.013546	0.013535	0.013438	0.013535	0.013512
μ_{C_D}	0.029460	0.029540	0.029426	0.029538	0.029319	0.029539
σ_{C_D}	0.000787	0.000764	0.000789	0.000768	0.000790	0.000768
CPU time units	*1*	*3.678*	*2.933*	*9.598*	*20.196*	*36.714*

Πίνακας 4.1: *Τυρβώδης ροή γύρω από μεμονωμένη αεροτομή, με τρεις αβέβαιες οριακές συνθήκες. Στατιστικές ροπές του C_L και του C_D που υπολογίστηκαν με τις μεθόδους iPCE και niPCE, για $C = 1, 2, 3$, και υπολογιστικό κόστους.*



Σχήμα 4.1: *Τυρβώδης ροή γύρω από μεμονωμένη αεροτομή, με τρεις αβέβαιες συνοριακές συνθήκες. Υπολογισμένο μέσο πεδίο (αριστερά) και τυπική απόκλιση (δεξιά) του αριθμού Mach (iPCE, $C = 1$).*

Κεφάλαιο 5

Συνεχής Συζυγής Μέθοδος του iPCE

Στο κεφάλαιο αυτό παρουσιάζεται η συνεχής συζυγής μέθοδος για τις iPCE εξισώσεις, σε αντιπαραβολή με τις συζυγείς εξισώσεις του προβλήματος χωρίς αβεβαιότητες. Με τη συνεχή συζυγή μέθοδο υπολογίζονται οι παράγωγοι της συνάρτησης–στόχου (εκφρασμένη με τους φασματικούς συντελεστές του PCE της QoI) ως προς τις μεταβλητές σχεδιασμού.

- Συνεχείς Συζυγείς Εξισώσεις

$$A\boldsymbol{\Psi} - \boldsymbol{g} = \mathbf{0} \;\; \text{(χωρίς αβεβαιότητες)} \;\; , \;\; \mathrm{G^q}\left[A\boldsymbol{\Psi} - \zeta\ \boldsymbol{g}\right] = \mathbf{0} \;\; \text{(iPCE)}$$

- Συζυγείς Οριακές Συνθήκες

$$B^*\boldsymbol{\Psi} - \boldsymbol{h} = \mathbf{0} \;\; \text{(χωρίς αβεβαιότητες)} \;\; , \;\; \mathrm{G^q}\left[B^*\boldsymbol{\Psi} - \zeta\ \boldsymbol{h}\right] = \mathbf{0} \;\; \text{(iPCE)}$$

  όπου $\zeta = \sum_{j=0}^{q} \zeta_j sign(F^j) Y_j(\boldsymbol{\xi})$

- Συνάρτηση–Στόχος

$$J = \sum_{j=0}^{q} \zeta_j |F^j| \;\; \text{(iPCE) ή η QoI (F) αν δεν υπάρχουν αβεβαιότητες}$$

- Παράγωγοι Ευαισθησίας

$$\delta F = \delta F_{SD} + \delta F_{SD}^{\Psi} \;\; \text{(χωρίς αβεβαιότητες)}$$

$$\delta J = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\delta F_{SD}\right] + \mathrm{G^q}\left[\delta F_{SD}^{\Psi}\right]^T \mathrm{G^q}\left[1\right] \;\; \text{(iPCE)}$$

  όπου $\mathrm{G^q}\left[1\right] = [1, 0, \ldots, 0]$

Κεφάλαιο 6

Εφαρμογή της Συζυγούς Μεθόδου του iPCE

Η προτεινόμενη συνεχής συζυγής διατύπωση παρουσιάζεται για τις εξισώσεις Euler σε δύο διαστάσεις. Οι εξισώσεις Euler γράφονται σε συντηρητική γραφή ως

$$\begin{aligned} \frac{\partial \boldsymbol{f}_i}{\partial x_i} = A_i \, \frac{\partial \boldsymbol{U}}{\partial x_i} = \boldsymbol{0} \ , \text{στο } \Omega \\ u_i n_i = 0 \ , \text{στο } S \\ \boldsymbol{U} = \boldsymbol{U}_\infty \ , \text{στο } S_\infty \end{aligned} \tag{6.1}$$

όπου S είναι το περίγραμμα της αεροτομής και $\boldsymbol{n} = [n_1, n_2]^T$ το κάθετο μοναδιαίο διάνυσμά του. Επίσης, το S_∞ είναι το απ΄ άπειρο όριο του χωρίου και $\boldsymbol{U}_\infty$ είναι η συνθήκες ελεύθερης ροής. Η QoI είναι η δύναμη της άνωσης

$$F = L \equiv \int_S p(n_2 \cos a_\infty - n_1 \sin a_\infty) dS \tag{6.2}$$

Με εφαρμογή του τελεστή $\mathrm{G^q}\,[]$ στις εξ. 6.1 προκύπτει

$$\begin{aligned} \mathrm{G^q}\left[\frac{\partial \boldsymbol{f}_i}{\partial x_i}\right] = \mathrm{G^q}\left[A_i \, \frac{\partial \boldsymbol{U}}{\partial x_i}\right] = \boldsymbol{0} \ , \text{ιν } \Omega \\ \mathrm{G^q}\left[u_i n_i\right] = \boldsymbol{0} \ , \text{ιν } S \, \mathrm{G^q}\left[\boldsymbol{U}\right] = \mathrm{G^q}\left[\boldsymbol{U}_\infty\right] \ , \quad \text{ιν } S_\infty \end{aligned} \tag{6.3}$$

και η συνάρτηση–στόχος ορίζεται ως

$$J = \sum_{j=0}^{q} \zeta_j |F^j| \tag{6.4}$$

ενώ η επαυξημένη συνάρτηση–στόχος είναι η

$$J_{aug} = J - \int_\Omega \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\frac{\partial \boldsymbol{f}_i}{\partial x_i}\right] d\Omega \tag{6.5}$$

Άρα

$$\begin{aligned}\delta J_{aug} = \delta J &- \int_{\Omega} \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\frac{\partial(\delta \boldsymbol{f}_i)}{\partial x_i}\right] d\Omega \\ &- \int_{\Omega} \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\frac{\partial(\delta \boldsymbol{f}_i)}{\partial x_k}\,\frac{\partial(\delta x_k)}{\partial x_i}\right] d\Omega\end{aligned} \tag{6.6}$$

με

$$\begin{aligned}\delta J &= \sum_{j=0}^{q} \zeta_j sign(F^j)\delta F^j = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\delta F\right] \\ &= \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\int_S \delta p(n_2 \cos a_\infty - n_1 \sin a_\infty) dS\right] + \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\delta F_{SD}\right]\end{aligned} \tag{6.7}$$

όπου $\zeta = \sum_{j=0}^{q} \zeta_j sign(F^j) Y_j(\boldsymbol{\xi})$. Στη συνέχεια, με παραγοντική ολοκλήρωση

$$\begin{aligned}&\int_{\Omega} \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\frac{\partial(\delta \boldsymbol{f}_i)}{\partial x_i}\right] d\Omega = \\ &- \int_{\Omega} \mathrm{G^q}\left[\frac{\partial \boldsymbol{\Psi}}{\partial x_i} A_i\right]^T \mathrm{G^q}\left[\delta \boldsymbol{U}\right] d\Omega + \int_S \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\delta \boldsymbol{f}_i\right] n_i dS\end{aligned} \tag{6.8}$$

Επίσης

$$\begin{aligned}&\int_S \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\delta \boldsymbol{f}_i\right] n_i dS = \\ &\int_S \mathrm{G^q}\left[\Psi_{i+1}\right] n_i \,\mathrm{G^q}\left[\delta p\right] dS + \underbrace{\int_S (\mathrm{G^q}\left[\Psi_{i+1}\right] \mathrm{G^q}\left[p\right] - \mathrm{G^q}\left[\boldsymbol{\Psi}\right]^T \mathrm{G^q}\left[\boldsymbol{f}_i\right]) \delta(n_i dS)}_{= \mathrm{G^q}\left[\delta F_{SD}^{\Psi}\right]^T \mathrm{G^q}[1]}\end{aligned} \tag{6.9}$$

Επομένως, προκύπτει ότι

$$\begin{aligned}\delta J_{aug} = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[F_{SD}\right] + \mathrm{G^q}\left[F_{SD}^{\Psi}\right]^T \mathrm{G^q}\left[1\right] + \int_{\Omega} \mathrm{G^q}\left[\frac{\partial \boldsymbol{\Psi}^T}{\partial x_i} A_i\right] \mathrm{G^q}\left[\delta \boldsymbol{U}\right] d\Omega \\ \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\int_S \delta p(n_2 \cos a_\infty - n_1 \sin a_\infty) dS\right] - \int_S \mathrm{G^q}\left[\Psi_{i+1}\right]^T \mathrm{G^q}\left[\delta p\right] n_i dS\end{aligned} \tag{6.10}$$

και οι συζυγείς iPCE Euler εξισώσεις είναι

$$\mathrm{G^q}\left[A_i^T \frac{\partial \boldsymbol{\Psi}}{\partial x_i}\right] = \mathbf{0} \tag{6.11}$$

με οριακές συνθήκες που προσδιορίζονται ως εξής

$$\mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[\int_S -\delta p(n_2 \cos a_\infty - n_1 \sin a_\infty) dS\right] - \int_S \mathrm{G^q}\left[\Psi_{i+1}\right]^T \mathrm{G^q}\left[\delta p\right] n_i dS = \mathbf{0} \Rightarrow$$

$$\mathrm{G^q}\left[\zeta\right]^T \int_S - \mathrm{G^q}\left[\delta p\right]\left(n_2 \cos a_\infty - n_1 \sin a_\infty\right) dS - \int_S \mathrm{G^q}\left[\Psi_{i+1}\right]^T \mathrm{G^q}\left[\delta p\right] n_i dS = \mathbf{0} \Rightarrow$$

$$\mathrm{G^q}\left[\zeta(n_2 \cos a_\infty - n_1 \sin a_\infty) + \Psi_{i+1} n_i\right] = \mathbf{0} \tag{6.12}$$

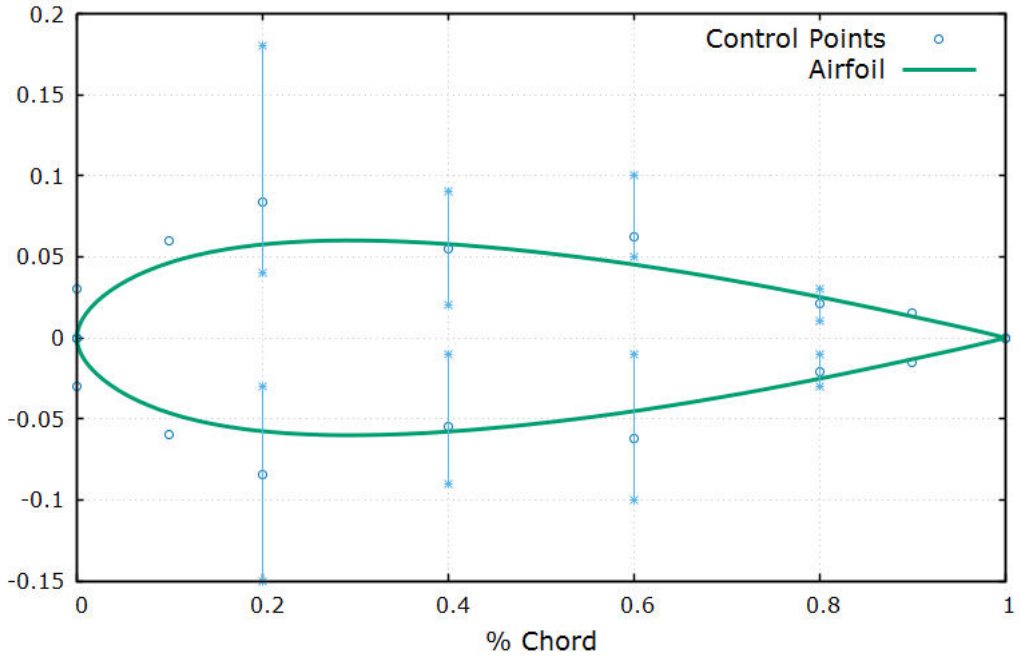
Τέλος, οι παράγωγοι ευαισθησίας είναι

$$\delta J = \mathrm{G^q}\left[\zeta\right]^T \mathrm{G^q}\left[F_{SD}\right] + \mathrm{G^q}\left[F_{SD}^{\Psi}\right]^T \mathrm{G^q}\left[1\right] \tag{6.13}$$

Κεφάλαιο 7

Αριθμητική Εφαρμογή της Συζυγούς iPCE Μεθόδου

Η μέθοδος που παρουσιάστηκε εφαρμόζεται στην στρωτή ροή γύρω από μια αεροτομή, η οποία παραμετροποιείται από δύο καμπύλες Bezier, τα σημεία ελέγχου των οποίων είναι οι μεταβλητές σχεδιασμού, σχ. 7.1.



Σχήμα 7.1: *Βελτιστοποίηση μορφής αεροτομής. Αρχική γεωμετρία και σημεία ελέγχου καμπυλών Bezier (μία ανά πλευρά).*

Ο συντελεστής οπισθέλκουσας είναι η συνάρτηση–στόχος, στην περίπτωση χωρίς αβεβαιότητες, δηλαδή $F = C_D$, ενώ οι οριακές συνθήκες είναι

$$M_\infty = 0.5 \quad , \quad a_\infty = 2^o \quad , \quad Re = 6000 \tag{7.1}$$

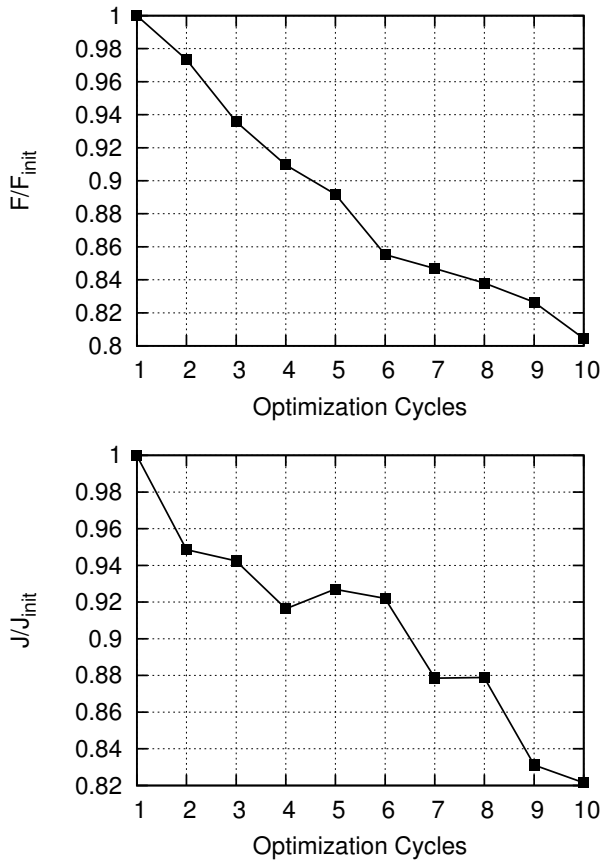
Στην περίπτωση με αβεβαιότητες, η συνάρτηση–στόχος ορίζεται ως

$$J = \sum_{j=0}^{q} \zeta_j |F^j|$$

με $q=19$ (για $m=3$ μεταβλητές και τάξη χάους $C=3$), $\zeta_0=1$ και $\zeta_j=3\ \forall\ j>0$. ενώ οι αβέβαιες συνοριακές συνθήκες είναι

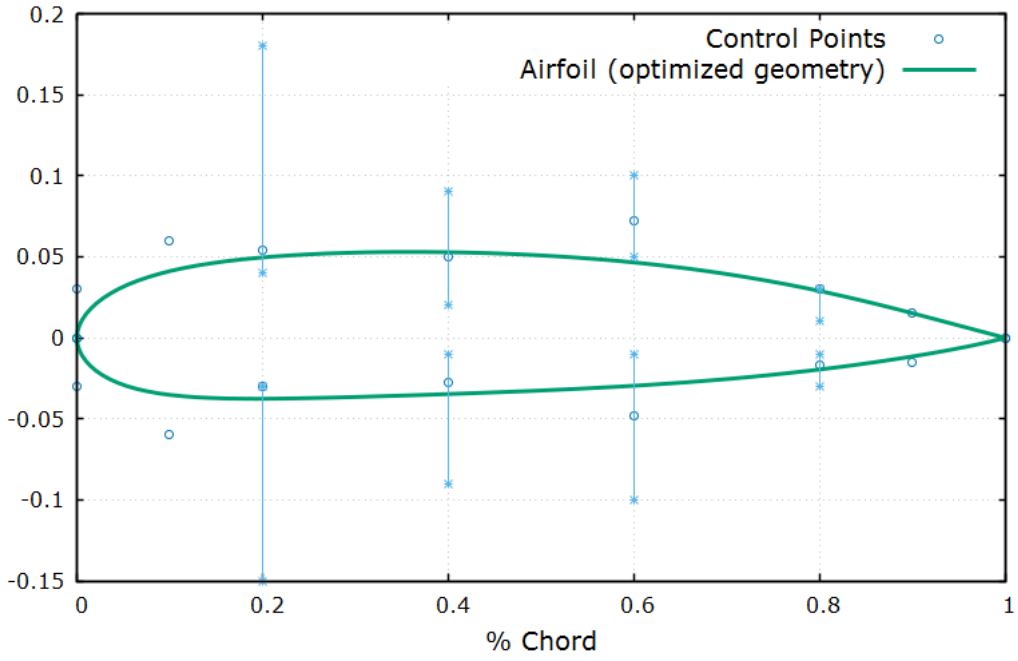
$$M_\infty \sim \mathcal{N}(0.5, 0.05) \quad , \quad a_\infty \sim \mathcal{U}(1.5^o, 2.5^o) \quad , \quad Re \sim \mathcal{N}(6000, 250)$$

Τα αποτελέσματα των δύο βελτιστοποιήσεων, με και χωρίς αβεβαιότητες, φαίνονται στο σχ. 7.2



Σχήμα 7.2: *Βελτιστοποίηση σχήματος αεροτομής, στρωτή ροή. Μείωση της συνάρτησης–στόχου χωρίς αβεβαιότητες (πάνω) και με αβεβαιότητες (κάτω).*

Στο σχ. 7.3 φαίνεται η βέλτιστη γεωμετρία για τη βελτιστοποίηση υπό αβεβαιότητες. Τέλος, στον πίνακα 7.1 συνοψίζονται τα αποτελέσματα της βελτιστοποίησης με αβεβαιότητες και συγκρίνονται με τις τιμές για την τυπική απόκλιση και τη μέση τιμή του συντελεστή οπισθέλκουσας που προκύπτουν με εφαρμογή του iPCE στη βέλτιστη γεωμετρία που προκύπτει από τη βελτιστοποίηση χωρίς αβεβαιότητες.



Σχήμα 7.3: *Βελτιστοποίηση σχήματος αεροτομής με αβεβαιότητες, στρωτή ροή. Βέλτιστη γεωμετρία.*

	Χωρίς Αβεβαιότητες	Με Αβεβαιότητες
μ_{C_D}	$6.81 \cdot 10^{-2}$	$6.97 \cdot 10^{-2}$
σ_{C_D}	$1.11 \cdot 10^{-3}$	$1.05 \cdot 10^{-3}$

Πίνακας 7.1: *Βελτιστοποίηση σχήματος αεροτομής, στρωτή ροή. Σύγκριση στατιστικών ροπών του συντελεστή οπισθέλκουσας για τις βέλτιστες γεωμετρίες των δύο βελτιστοποιήσεων.*

Κεφάλαιο 8

Μία Εναλλακτική της Συζυγούς iPCE Διατύπωσης

Σε αυτό το κεφάλαιο προτείνεται μία εναλλακτική μέθοδος της συζυγούς διατύπωσης του iPCE, η οποία είναι υπολογιστικά οικονομικότερη. Η μέθοδος αυτή θα αναφέρεται ως DDSP ('Deterministic Derivatives – Stochastic Primal').

8.1 Η μέθοδος DDSP

Η βασική ιδέα της μεθόδου είναι ο υπολογισμός της παραγώγου της συνάρτησης–στόχου του προβλήματος υπό αβεβαιότητες δJ να γίνει μέσω του υπολογισμού της παραγώγου δF, όπως αυτή προκύπτει από το συζυγές πρόβλημα χωρίς αβεβαιότητες. Για το σκοπό αυτό, γίνεται η υπόθεση ότι υπάρχει ένα σύνολο τιμών $\boldsymbol{\xi} = \boldsymbol{\xi}_s$ τέτοιο ώστε $\delta F(\boldsymbol{\xi}_s) = \delta J$, δηλαδή

$$\sum_{i=0}^{\infty} \delta F^i Y_i(\boldsymbol{\xi}_s) = \sum_{i=0}^{q} \zeta_i sign(F^i) \delta F^i \tag{8.1}$$

Η εξ. 8.1, μετά από την αποκοπή όρων από το άπειρο άθροισμα στο αριστερό της μέλος, οδηγεί στη σχέση

$$Y_i(\boldsymbol{\xi}_s) = \zeta_i sign(F^i), i = 0, 1, \ldots, q \tag{8.2}$$

Ικανοποίηση της εξ. 8.2, για κατάλληλο $\boldsymbol{\xi}_s$ αναμένεται να οδηγήσει στον υπολογισμό ενός $\delta F(\boldsymbol{\xi}_s)$ το οποίο θα ισούται με την παράγωγο δJ της συνάρτησης–στόχου του προβλήματος υπό αβεβαιότητες. Ασφαλώς, πριν την ικανοποίησή της απαιτείται η λύση των iPCE εξισώσεων, ώστε να προσδιοριστούν τα $sign(F^j)$.

8.2 Επίλυση της εξ. 8.2

Πριν επιλυθεί η εξ. 8.2 ως προς $\boldsymbol{\xi}_s$ πρέπει να ικανοποιηθεί η σχέση που προκύπτει θέτοντας $j = 0$ σε αυτή, δηλαδή

$$\zeta_0 = sign(F^0)Y_0 = sign(F^0) \tag{8.3}$$

Άρα, ο όρος ζ_0 δεν μπορεί να επιλεχθεί ελεύθερα από τον χρήστη, σε αυτήν την περίπτωση. Ωστόσο, θα μπορούσε ούτως ή άλλως να είχε τεθεί $|\zeta_0| = 1$ από την αρχή, χωρίς βλάβη γενικότητας και με το ίδιο αποτέλεσμα στη βελτιστοποίηση. Κατόπιν, επιλύονται οι υπόλοιπες q εξισώσεις της εξ. 8.2

Επίλυση της εξ. 8.2 για $C = 1$

Για $C = 1$ η εξ. 8.2 αποτελεί ένα γραμμικό σύστημα με ίσο αριθμό αγνώστων m και εξισώσεων, διότι $q = m$ όταν $C = 1$. Συνεπώς, σε αυτή την περίπτωση, το $\boldsymbol{\xi}_s$ προσδιορίζεται εύκολα.

Επίλυση της εξ. 8.2 για $C > 1$

Για $C > 1$, ο αριθμός των εξισώσεων γίνεται $q = (C + m)!/C!m! - 1$, ενώ το πλήθος των αγνώστων είναι m. Συνεπώς, επιλέγεται να ελαχιστοποιηθεί η έκφραση

$$M := \frac{1}{2}\sum_{i=1}^{q}[Y_i(\boldsymbol{\xi}) - \zeta_i sign(F^i)]^2 \tag{8.4}$$

και οι εξισώσεις προς επίλυση είναι οι

$$R_j := \frac{\partial M}{\partial \xi_j} = \sum_{i=1}^{q}[Y_i(\boldsymbol{\xi}) - \zeta_i sign(F^i)]\ \frac{\partial Y_i}{\partial \xi_j} = 0 \quad , \ j = 1, \ldots, m \tag{8.5}$$

Η εξ. 8.5 λύνεται επαναληπτικά μέσω του σχήματος

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\xi}}\right)_{old} \Delta\boldsymbol{\xi} = -\boldsymbol{R}_{old} \tag{8.6}$$

όπου $\boldsymbol{R} = (R_1, \ldots, R_m)$, $\boldsymbol{\xi}_{new} = \boldsymbol{\xi}_{old} + \Delta\boldsymbol{\xi}$ και

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\xi}}\right)_{jk} = \sum_{i=1}^{q}\left([Y_i - \zeta_i sign(F^i)]\frac{\partial^2 Y_i}{\partial \xi_k \partial \xi_j} + \frac{\partial Y_i}{\partial \xi_j}\ \frac{\partial Y_i}{\partial \xi_k}\right) \tag{8.7}$$

8.2.1 Βελτιστοποίηση και Συγκρίσεις Υπολογιστικού Κόστους

Σε αυτή την ενότητα εφαρμόζεται η μέθοδος DDSP σε μία μεμονωμένη αεροτομή, σε στρωτή ροή. Η αβεβαιότητα εισάγεται στις οριακές συνθήκες ως εξής

$$M_\infty \sim \mathcal{N}(0.5, 0.05) \ \ , \ \ a_\infty \sim \mathcal{U}(-1.5^o, 2.5^o) \ \ , \ \ Re_\infty \sim \mathcal{N}(5000, 300) \tag{8.8}$$

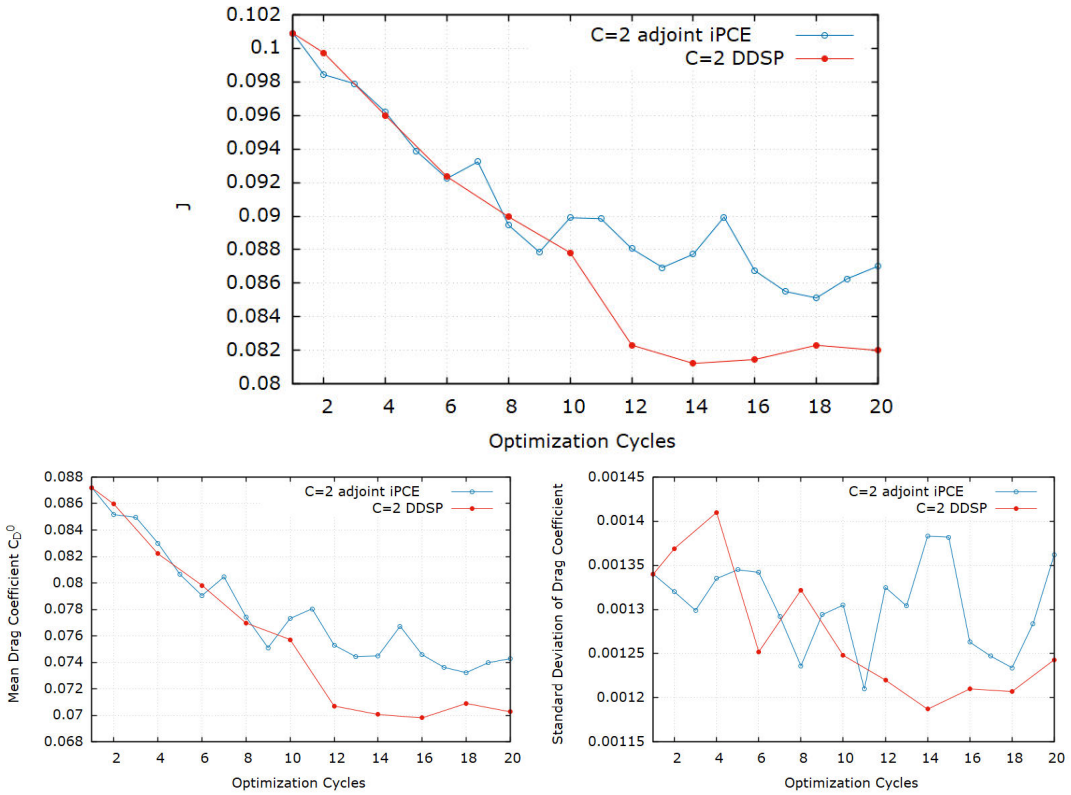
ενώ η συνάρτηση–στόχος, όταν υπάρχουν αβεβαιότητες ορίζεται ως

$$J = \sum_{i=0}^{q} \zeta_j |C_D^j| \ , \ \zeta_0 = 1 \ , \ \zeta_j = 5 \ , \ j > 0 \tag{8.9}$$

όπου C_D είναι ο συντελεστής οπισθέλκουσας.

Βελτιστοποίηση για $C = 2$

Στο σχ. 8.1 απεικονίζεται η πορεία της βελτιστοποίησης με τη μέθοδο DDSP και συγκρίνεται με τη συζυγή iPCE μέθοδο. Από άποψη υπολογιστικού κόστους, η συζυγής iPCE μέθοδος χρειάστηκε 5710 δευτερόλεπτα ενώ η μέθοδος DDSP μόλις 1908. Επιπλέον, η λύση της μεθόδου DDSP είναι καλύτερη από την λύση της συζυγούς iPCE μεθόδου. Το συμπέρασμα είναι ότι η μέθοδος DDSP μπορεί να είναι αποδοτικότερη και πιο οικονομική.



Σχήμα 8.1: *Αποτελέσματα βελτιστοποίησης για $C = 2$. Τιμή της συνάρτησης–στόχου (πάνω), μέση τιμή (κάτω αριστερά) και τυπική απόκλιση (κάτω δεξιά) του συντελεστή οπισθέλκουσας.*

Κεφάλαιο 9

Συμπεράσματα

Η προτεινόμενη μέθοδος iPCE προγραμματίστηκε σε οικείο λογισμικό για τις 3Δ εξισώσεις RANS συμπιεστού ρευστού, με το μοντέλο τύρβης μιας εξίσωσης των Spalart–Allmaras. Η συζυγής διατύπωσή της προγραμματίστηκε επίσης, για τις εξισώσεις Navier–Stokes για στρωτή ροή συμπιεστού ρευστού, σε δύο διαστάσεις. Και οι δύο μέθοδοι θεμελιώθηκαν μαθηματικά, έτσι ώστε η εφαρμογή τους να είναι εύκολη και άκοπη, χωρίς όμως να στερούνται ακρίβειας στα αποτελέσματα.

Το κύριο συμπέρασμα αυτής της διπλωματικής εργασίας είναι ότι η προτεινόμενη μέθοδος iPCE φαίνεται να συνδυάζει τα πλεονεκτήματα τόσο των επεμβατικών όσο και των μη–επεμβατικών εκδοχών. Πρώτα απ ΄όλα, είναι εύκολο να προγραμματιστεί / εφαρμοστεί. Απαιτεί ελάχιστες αλλαγές λογισμικού και απολύτως καμία μαθηματική επεξεργασία για την εξαγωγή και διακριτοποίηση των iPCE εξισώσεων. Επίσης, είναι γενική και ισχύει για οποιοδήποτε σύστημα εξισώσεων, αλλά και υπολογιστικά συμφέρουσα.

Το ίδιο ισχύει και για την προτεινόμενη συζυγή μέθοδο για προβλήματα βελτιστοποίησης με αβεβαιότητες. Η διατύπωση που προτείνεται είναι γενική ενώ ο προγραμματισμός της γίνεται άκοπα, με την προϋπόθεση της ύπαρξης του αντίστοιχου λογισμικού για το συζυγές πρόβλημα χωρίς αβεβαιότητες. Τέλος, η προσέγγιση DDSP, η οποία προτάθηκε, ως εναλλακτική της συζυγούς μεθόδου, την καθιστά ακόμα πιο οικονομική σε κόστος υπολογισμού, μέσω της επίλυσης του συζυγούς προβλήματος σε ᾽κατάλληλα μετατοπισμένο σημείο του χώρου των αβέβαιων μεταβλητών.

References

[1] B. Lapeyre. Introduction to monte–carlo methods. Halmstad University, Lecture Notes, 2007.

[2] W. Morokoff and R. Caflisch. Quasi-Monte Carlo integration. *Journal of Computational Physics*, 122(2):218–230, 1995.

[3] M.D. McKay. Latin hypercube sampling as a tool in uncertainty analysis of computer models. In *24th Conference on Winter Simulation, WSC '92*, New York, NY, USA, June 7-10 1992.

[4] D. Xiu. *Numerical Methods for Stochastic Computations: A Spectral Approach*. Princeton University Press, 2010.

[5] E.M. Papoutsis-Kiachagias, Papadimitriou D.I, and K.C. Giannakoglou. Robust design in aerodynamics using 3rd-order sensitivity analysis based on discrete adjoint. application to quasi-1d flows. *International Journal for Numerical Methods in Fluids*, 69(3):691–709, 2012.

[6] B. Ganapathysubramanian and N. Zabaras. Sparse grid collocation schemes for stochastic natural convection problems. *Journal of Computational Physics*, 225(1):652–685, 2007.

[7] MS Eldred. Recent advances in non-intrusive polynomial chaos and stochastic collocation methods for uncertainty analysis and design. In *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2009.

[8] D. Xiu. Fast numerical methods for stochastic computations: a review. *Communications in Computational Physics*, 5(2-4):242–272, 2009.

[9] A. Alexanderian. A brief note on the karhunen-loève expansion. North Carolina State University, Lecture Notes, 2015.

[10] F. Poirion. Karhunen Loève expansion and distribution of non-gaussian process maximum. *Probabilistic Engineering Mechanics*, 43, 2016.

[11] M.P. Pettersson, G. Iaccarino, and J. Nordström. *Polynomial Chaos Methods for Hyperbolic Partial Differential Equations*. Springer International Publishing, Switzerland, 2015.

[12] O.P. Le Maître, O.M. Knio, H.N. Najm, and R.G. Ghanem. A stochastic projection method for fluid flow I. Basic formulation. *Journal of Computational Physics*, 173(2):481–511, 2001.

[13] B.J. Debusschere, H.N. Najm, P.P. Pébray, O.M. Knio, R.G. Ghanem, and O.P. Le Maître. Numerical challenges in the use of polynomial chaos representations for stochastic processes. *SIAM Journal of Scientific Computing*, 26(2):698–719, 2004.

[14] O.M. Knio and O.P. Le Maître. Uncertainty propagation in CFD using polynomial chaos decomposition. *Fluid Dynamics Research*, 38:616–640, 2006.

[15] N. Wiener. The homogeneous chaos. *American Journal of Mathematics*, 60:897–936, 1938.

[16] D. Xiu and G.E Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *Journal of Computational Physics*, 187:37–67, 2003.

[17] T. Ghisu and S. Shahpar. Towards affordable uncertainty quantification for industrial problems-Part I: Theory and validation. In *Proceedings of ASME Turbo Expo 2017, GT2017-64842*, Charlotte, NC, USA, June 26-30 2017.

[18] T. Ghisu and S. Shahpar. Towards affordable uncertainty quantification for

industrial problems-Part II: Turbomachinery application. In *Proceedings of ASME Turbo Expo 2017, GT2017-64845*, Charlotte, NC, USA, June 26-30 2017.

[19] A. Cuneo, A. Traverso, and S. Shahpar. Comparative analysis of methodologies for uncertainty propagation and quantification. In *Proceedings of ASME Turbo Expo 2017, GT2017-63238*, Charlotte, NC, USA, June 26-30 2017.

[20] M. Emory, G. Iaccarino, and G.M. Laskowski. Uncertainty quantification in turbomachinery simulations. In *Proceedings of ASME Turbo Expo 2016, GT2016-56798*, Seoul, South Korea, June 13-17 2016.

[21] M. Eldred and J. Burkardt. Comparison of non-intrusive polynomial chaos and stochastic collocation methods for uncertainty quantification. *American Institute of Aeronautics and Astronautics*, 2009.

[22] S. Smolyak. Quadrature and interpolation formulas on tensor products of certain classes of functions. *Dokl. Akad. Nauk SSSR*, 148(5):1042–1045, 1963.

[23] M. Hadigol and A. Doostan. Least squares polynomial chaos expansion: A review of sampling strategies. *Computer Methods in Applied Mechanics and Engineering*, 332:382 – 407, 2018.

[24] K.-D. Kantarakias, M.E. Chatzimanolakis, V.G. Asouti, and K.C. Giannakoglou. On the development of the 3D Euler equations using intrusive pce for uncertainty quantification. In *2nd ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP 2017)*, Rhodes Island, Greece, June 15-17 2017.

[25] B. Debusschere. *Intrusive Polynomial Chaos Methods for Forward Uncertainty Propagation*. Springer International Publishing, 2016.

[26] G. Onorato, G. Loeven, G. Ghorbaniasl, B. Hester, and C. Lacor. Comparison of intrusive and non-intrusive polynomial chaos methods for cfd applications in aeronautics. pages 14–17, 07 2010.

[27] A.G. Liatsikouras, V.G. Asouti, K.C. Giannakoglou, G. Pierrot, and M. Megahed. Aerodynamic shape optimization under flow uncertainties using non-intrusive polynomial chaos and evolutionary algorithms. In *2nd ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP 2017)*, Rhodes Island, Greece, June 15-17 2017.

[28] R. Duvigneau. Aerodynamic shape optimization with uncertain operating conditions using metamodels. RR-6143, INRIA, 2007.

[29] D. Lee, L. Gonzalez, J. Periaux, and K. Srinivas. Robust design in aerodynamics using 3rd-order sensitivity analysis based on discrete adjoint. application to quasi-1d flows. *Computers and Fluids*, 37:565–583, 2008.

[30] J. Vigouroux, L. Deshayes, S. Foufou, J. Filliben, L. Welsch, and M. Donmez. Robust design of an evolutionary algorithm for machining optimization problems. *Journal of Computing and Information Science in Engineering*, 2016.

[31] S. Ho and Y. Shiyou. A fast robust optimization methodology based on polynomial chaos and evolutionary algorithm for inverse problems. *IEEE Transactions on Magnetics*, 48:259–262, ???

[32] T. Drzewieck. Adjoint based uncertainty quantification and sensitivity analysis for nuclear thermal-fluids codes. PhD thesis. University of Michigan, 2013.

[33] T. Seifried. Adjoint-based uncertainty quantification with mcnp. PhD thesis. University of California, Berkeley, 2011.

[34] K.C. Giannakoglou, D.I. Papadimitriou, E.M. Papoutsis-Kiachagias, and I. Kavvadias. Adjoint methods for shape optimization and robust design in fluid mechanics. In *OPT-i, International Conference on Engineering and Ap-*

plied Sciences Optimization, Kos Island, Greece, June 4-6 2014.

[35] K.C. Giannakoglou, D. Papadimitriou, E Papoutsis-Kiachagias, and I.S. Kavvadias. Adjoint methods for shape optimization and robust design in fluid mechanics. pages 2252–2265, 01 2014.

[36] T.S. Chihara. *An Introduction to Orthogonal Polynomials*. Gordon and Breach, New York, 1978.

[37] G.E. Andrews, R. Askey, and R. Roy. *Special Functions , Encyclopedia of Mathematics and Its Applications*. The University Press, Cambridge, 1999.

[38] T. Gerstner. Sparse grid quadrature methods for computational finance. University of Bonn, Lecture Notes, 2007.

[39] V. Kaarnioja. Smolyak quadrature. Master's thesis. University of Helsinki, Department of Mathematics and Statistics, 2013.

[40] G. Blatman and B. Sudret. Adaptive sparse polynomial chaos expansion based on least angle regression. *Journal of Computational Physics*, 230:2345–2367, 2011.

[41] Y. Sun, R. Mao, Z. Li, and W. Tian. Constant Jacobian matrix-based stochastic Galerkin method for probabilistic load flow. *Energies*, 9(3), 2016.

[42] P. Spalart and S. Allmaras. A one–equation turbulence model for aerodynamic flows. *La Recherche Aerospatiale*, 1:5–21, 1994.

[43] B. van Leer. Flux vector splitting for the euler equations. In *8th International Conference on Numerical Methods in Fluid Dynamics*, Aachen, Germany, June 28 - July 2 1982.

[44] O. Brodersen and A. Stürmer. Drag prediction of engine-airframe interference effects using unstructured Navier-Stokes calculations. In *16th AIAA Applied Aerodynamics Conference*, California, USA, June 11-14 2001.

[45] M. B. Giles and N A. Pierce. Adjoint equations in cfd - duality, boundary conditions and solution behaviour. *AIAA Journal*, 97, 2000.