Time-Dependent Methods for Stochastic Differential Equations

Master Thesis

J.C. Oostelbos

February 12, 2010

Supervisor: Dr. Ir. M.I. Gerritsma

Aerodynamics Group, Faculty of Aerospace Engineering
Delft University of Technology

Preface

This report is the conclusion of my Master's thesis project. It is the result of nine months of research on the improvement of non-statistical probabilistic models. Besides that, it is of course also the finalization of my career as an Aerospace Engineering student.

Some prior knowledge of statistics, differential equations, etc. is required to fully follow the line of reasoning of this report.

But I would foremost like to take this opportunity to thank my supervisor Dr. M.I. Gerritsma for his guidance and advice. Furthermore, I am in debt to Prof. G.E. Karniadakis at Brown University for his help, encouragement and the chance to visit his Crunch group in Providence. But above all I am thankful for the support of my fiancee, Sanne.

List of Abbreviations

Abbreviation	Description
CFD	Computational Fluid Dynamics
DE	Differential Equation
FEM	Finite Element Method
GLL	Gauss-Lobatto-Legendre
gPC	generalized Polynomial Chaos
ME-gPC	Mulit-Element generalized Polynomial Chaos
ME-PCM	Multi-Element Probabilistic Collocation Method
ODE	Ordinary Differential Equation
PCM	Probabilistic Collocation Method
PDE	Partial Differential Equation
PDF	Probability Density Function
RK4	Runge-Kutta 4^{th} order
TDgPC	Time-Dependent generalized Polynomial Chaos
TDPCM	Time-Dependent Probabilistic Collocation Method

Nomenclature

Symbol	Description
a	Advection velocity
E	Expected value
f	Probability density function
g	Source term
k	Growth factor
$\mathcal{L}$	Differential operator
M	Total number of expansion terms
N	Number of collocation points in TDPCM
P	Polynomial order
Q	Number of quadrature points in TDgPC
t	Time
t^*	Time at transformation
u	Solution variable
w	Weights for quadrature integration
x	Variable of physical space
δ	Dirac delta function
Δt	Time step
ϵ	Threshold value
ϵ_{mean}	Mean error
ϵ_{var}	Variance error
ζ	Transformed stochastic variable
κ	Heat coefficient
μ	Average
ξ	Stochastic variable
σ	Variance
ϕ_i	Polynomial of order i
ω	Point in random space

Abstract

In the last few decades, enormous progress has been made in the field of Computational Fluid Dynamics (CFD). Computers became powerful enough for full-scale simulations and smart algorithms enabled engineers to simulate intricate phenomena like for example turbulence. However, these developments do not stop, and a promising, new area of research is the incorporation of uncertainty quantification in these CFD simulations. Since traditional statistical methods like the Monte Carlo method are useless in these cases due to computational cost, deterministic solutions are sought.

The two major non-statistical methods now in use are the generalized Polynomial Chaos (gPC) and the Probabilistic Collocation Method (PCM). The former relies on a polynomial expansion of the solution variable together with a Galerkin projection. It delivers exponential convergence rates and is extendable to a high number of stochastic variables. However, it is an intrusive method, i.e. the equations themselves have to be altered in the implementation of the algorithm. It also has difficulty coping with non-linear equations.

The PCM on the other hand, is a non-intrusive method. It post-processes the outcome of several deterministic simulations, using cleverly chosen values for the stochastic input, i.e. collocation points. Polynomial interpolation is then used to calculate the behaviour of the random variable in stochastic space. Integrating the random variable delivers the statistical moments, e.g. the mean and variance. The PCM procedure has the disadvantage that is becomes computationally expensive very rapidly for more than a few random variables.

The PCM and gPC methods are similar in many aspects, but most notably in their incapability to deliver accurate results in long-term integrations. Both methods suffer from a phenomena called stochastic drift. At the start of a simulation, certain assumptions are made on the distribution of the stochastic variable. For example, it is assumed to be uniformly distributed. According to the initial distribution, the gPC algorithm is started using the best possible polynomial basis. Similarly, the PCM chooses its collocation points with respect to the initial distribution. So different initial distributions will result in different set-ups for both methods.

During the simulation, the distribution of the solution variable will change. It will not stay uniformly distributed, but might e.g. cluster around one extreme of the stochastic domain. As a result, the initial polynomial expansion/collocation points will not be ideally suited any more for the current distribution of the stochastic variable. This is stochastic drift and will result in rapidly increasing errors during time integration.

As suggested by Prof. G.E. Karniadakis, Vos started the research on Time-Dependent generalized Polynomial Chaos (TDgPC). The main idea was to transform the stochastic variable and its polynomial basis over time, according to the current distribution, thus countering the stochastic drift. Van der Steen continued this research and considerably improved the algorithm.

In this graduation project, further improvements were sought for the TDgPC method. Using the differential equation known as the growth equation as a test bed, different new formulations were tested. Recognizing the characteristics of the growth equation led to a new expansion of the random variable, resulting in substantial increases in accuracy of the calculations.

However this formulation proved to be too case sensitive to be widely applicable, so a generalization was sought and found. When applied to the partial differential equation called the heat equations, results were excellent. The accuracy gained for the errors in variance and mean was around six orders of magnitude.

Afterwards, the new formulation of TDgPC was analyzed more closely. It was recognized that the TDgPC

method had evolved into a collocation method. The TDgPC procedure in its new form turned out to be equivalent to the PCM. No longer was the accuracy of the simulation dependent on the order of the polynomial expansion, but on the number of collocation points.

Thus, switching attention from gPC to PCM, time-dependent methods were sought to counter stochastic drift in PCM simulation. First of all, p-refinement was investigated, i.e. the addition of collocation points during the time integration. Since only a few collocation points are necessary to capture the behaviour of the solution in the first few time step iterations, it is computationally inefficient to initiate the simulation using many points. When the accuracy of the simulation is compromised, p-refinement can be applied to keep the accuracy at a certain predefined level. Especially for full-scale calculations where computational costs becomes an issue, p-refinement could be a powerful tool.

Furthermore, the principles of TDgPC were applied to the PCM, i.e., the initial set-up of the simulation will not be the optimal basis further along the time integration. In the case of the PCM procedure, this means that the positioning of the collocation points in stochastic space might be ideal early on in the simulation, but not necessarily so after several seconds. Using again the example that the distribution of the solution might cluster around an extreme of the stochastic domain, one would like the collocation points to cluster around these points as well. This will enable the collocation points to capture the behaviour of the solution more efficiently, leading to better accuracy.

Thus, the Time-Dependent Probabilistic Collocation Method was formulated. Even though some limitations were found that need to be addressed in further research, the algorithm proved to be beneficial for the accuracy of the calculations. The collocation points indeed 'move' towards the region in the stochastic domain where they were most efficiently used.

Table of Contents

Preface i

List of Abbreviations ii

Nomenclature iii

Abstract iv

1 Introduction 1

2 Stochastic methods: motivation and history 2
2.1 Motivation . . . 2
2.2 Overview of existing methods . . . 2

3 Time-Dependent generalized Polynomial Chaos 5
3.1 Generalized polynomial chaos . . . 5
3.2 Gauss-Lobatto-Legendre quadrature . . . 8
3.3 Adjustments to transform gPC into TDgPC . . . 9

4 TDgPC applied to the growth equation 11
4.1 Problem definition . . . 11
4.2 TDgPC applied to the growth equation . . . 12
4.3 TDgPC shortcomings . . . 12
4.3.1 Polynomial expansion of the natural logarithm . . . 12
4.4 Adjustment of the TDgPC method . . . 14

5 TDgPC applied to the heat equation 16
5.1 Definition of the heat equation . . . 16
5.2 From growth equation to heat equation . . . 16
5.3 Generalized formulation . . . 17
5.4 Discretization and the Galerkin projection . . . 17
5.5 Results . . . 18
5.6 Comparison with semi-discrete analytical system . . . 20
5.7 Comparison with fully discrete analytical system . . . 22
5.8 Influence of threshold value ϵ . . . 24

6 From TDgPC to PCM 25
6.1 The Probabilistic Collocation Method . . . 25
6.2 The growth equation revisited . . . 25
6.3 Shortcoming of the PCM . . . 26

7 P-refinement applied to the PCM **29**
7.1 Interpolation . . . 29
7.2 Refinement criterium . . . 29
7.3 Manner of refinement . . . 30
7.4 Results . . . 30

8 Time-Dependent Probabilistic Collocation Method **31**
8.1 Working principle of TDPCM . . . 31
8.2 The Probabilistic Collocation Method applied to the growth equation . . . 31
8.3 The TDPCM algorithm . . . 32
8.4 Considerations on the algorithm . . . 33
8.4.1 Interpolations and integrals . . . 33
8.4.2 Calculation of the collocation points . . . 34
8.4.3 Rootfinders . . . 34
8.4.4 Transformation criterium . . . 35
8.5 Results . . . 35
8.5.1 Influence of the number of collocation points . . . 37
8.5.2 Influence of the time of transformation . . . 37
8.5.3 Influence of the number of transformations . . . 37

9 Conclusions and recommendations **40**
9.1 Conclusions . . . 40
9.2 Recommendations . . . 40
9.2.1 The PCM paradox . . . 41

Bibliography **42**

Chapter 1

Introduction

Although the field of Computational Fluid Dynamics (CFD) has made enormous progress with the advent of powerful computers, researchers are constantly trying to incorporate more physical phenomena into their programs. Ideally, a true representation of reality is provided, not a simplified model. A promising and relatively new area is the incorporation of uncertainties into the governing equations. One could for example aim to implement an unknown fluctuation in material properties, inlet conditions or forcing terms. The procedures used for the modeling of these uncertainties are called probabilistic or stochastic methods.

One of the major problems in this field is the treatment of time-dependent equations. The traditional methods are incapable of yielding accurate results when integrating over even more than a few seconds.

One way of tackling this issue has been researched by P. Vos and J.B. van der Steen and has been dubbed Time-Dependent generalized Polynomial Chaos (TDgPC) [1, 2]. This thesis describes the continuation of their research.

The outline of this report is as follows: Chapter 2 will explain why probabilistic method are necessary and give an overview of existing stochastic methods. Chapter 3 will describe the TDgPC method in detail, which will be applied to an ordinary differential equation in Chapter 4. Here the TDgPC algorithm is modified, leading to significantly better results. This new approach is generalized and applied to a partial differential equation in Chapter 5. In Chapter 6 this new formulation is examined more closely, eventually leading to the Probabilistic Collocation Method (PCM). To increase the accuracy of the Probabilistic Collocation Method, p-refinement is explored in Chapter 7. In Chapter 8, the principles of the time-dependent approach of TDgPC are applied to the PCM procedure, resulting in the Time-Dependent Probabilistic Collocation Method (TDPCM). This thesis will be concluded with a chapter containing the conclusions of this research and recommendations for further research.

Chapter 2

Stochastic methods: motivation and history

This chapter explains the use of stochastic methods and gives an overview of work already performed in this field.

2.1 Motivation

As stated in the introduction, researchers are nowadays trying to include more and more of the physical world into their models. One way of incorporating more 'physics' into a computational model is to include a realistic measure of uncertainty. This leads to more accurate predictions of relevant quantities in e.g. flow computations. As a consequence, the data on which design decisions are made, are improved. Ultimately, this helps to optimize the design process. However, there is still a lot of research needed before probabilistic modeling can be used on typical flow equations and arbitrary domains.

In this light, time-dependent approaches of stochastic differential equations have the potential to become a widely applicable tool for designers. It promises an exponential convergence rate, with accuracy in the case of long term integration, even though the method is new and far from fully developed.

2.2 Overview of existing methods

The main division in probabilistic methods is between the procedures using a statistical approach and the procedures using a non-statistical approach. In general, the former method uses non-deterministic equations, whereas the later method entails a deterministic formulation, see Figure 2.1. However, research has been conducted on methods that use a combination of both approaches.

The most well-known example of a statistical method is the Monte Carlo concept. It relies on the calculation of a large number of samples. From this data, the statistical moments can be calculated in a bootstrapping manner. Although this method is straightforward to implement, it lacks elegance and its accuracy relies heavily on the sample size. As a result of the later, this method can become too computationally expensive to be useful for research purposes. This is especially true if the model and its equations are already complicated in the deterministic case.

A popular non-statistical method is the perturbation method. Its principle lies in a Taylor series expansion of the random field. However, this method has the disadvantage of becoming too complicated when using third order expansions or higher. Furthermore, this method deals inherently with small variations. Therefore, the uncertainties to be modeled cannot exceed approximately 10% of the mean value [3]. As a result, it is not a practical tool in CFD, where conditions may vary significantly and the equations are rarely simple.

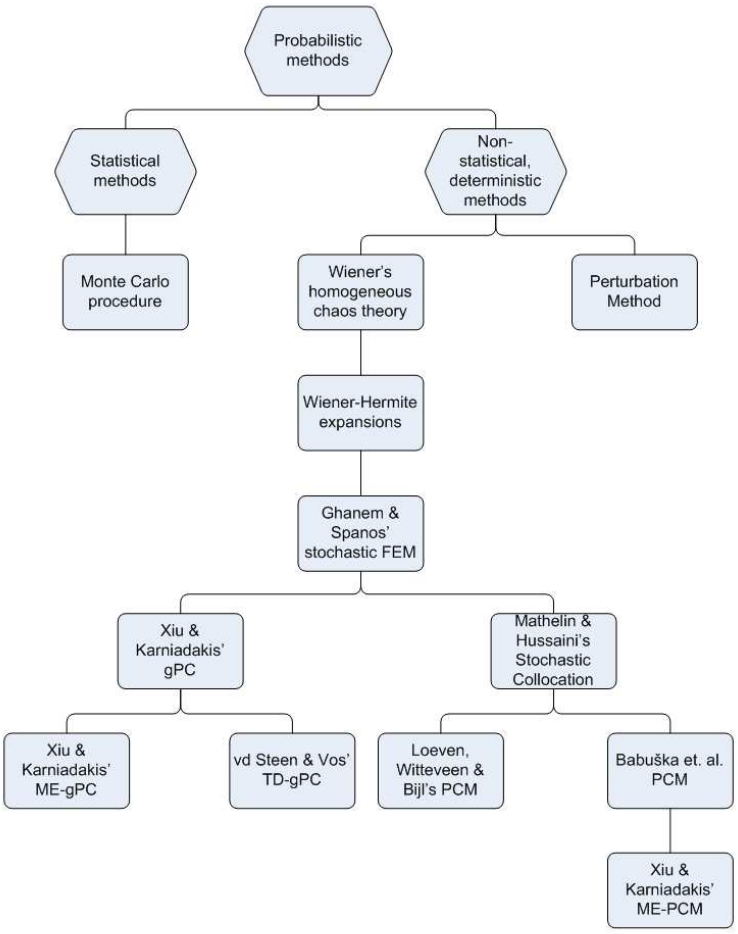


Figure 2.1: Overview of existing probabilistic methods

Another branch of non-statistical models is the one initiated in the 1930's by Wiener. Initially dubbed the "Homogeneous Chaos," Wiener outlines the basic theory for the representation of chaos through the use of polynomials [4]. However, this research was not pursued again until the 1960's, when Imamura, Sieger and Meecham proposed an expansion of a random function in Wiener-Hermite functionals. [5]. This procedure was then applied to a near-Gaussian random function. This method was neglected for several decades, mainly due to a lack of computing power.

The interest for chaos expansions was rekindled when Ghanem and Spanos combined the concept with a Finite Element Method (FEM) [6]. This combination was then applied to model uncertainties in several solid mechanics cases. They were able to attain an exponential convergence rate for Gaussian stochastic processes.

A major step was made by Xiu and Karniadakis with the introduction of generalized Polynomial Chaos (gPC) [7, 8]. The core idea of gPC was the linking of the weighting function in the orthogonality relations, to the probability density function (PDF) of the random distribution. More specifically, for certain standard random distributions (e.g. Gaussian, uniform, etc.), related expansion polynomials were found. Thus, also non-Gaussian processes could be represented by a family of orthogonal polynomials (the so-called Askey-scheme) with the classical Hermite polynomials chaos as a subset. The next step was to extend the procedure to arbitrary random distributions. Whereas the optimal expansion polynomials for standard distributions

are found in the Askey-scheme, the optimal trial basis for arbitrary random distributions is calculated using e.g. the Gram-Schmidt orthogonalization technique.

The gPC technique proved to be efficient, with an exponential convergence of the solution for several cases. However, difficulties were encountered in two specific areas. First of all, long-term integration proved to be troublesome. This stemmed from the fact that a certain random distribution is assumed at time $t = 0$, which does not necessarily correspond to the distribution at later times. For example, periodic stochastic solutions with random frequencies experience amplified phase shifts [8]. This results in an error that increases in time and eventually becomes unacceptable. Secondly, gPC has difficulties converging for discontinuous distributions in the random space. Both problems cannot be overcome by simply increasing the polynomial order or reducing the time step.

To overcome the aforementioned problems with gPC, Wan and Karniadakis developed the multi-element generalized polynomial chaos method (ME-gPC). Basically, probability space is discretized in a similar way to the spectral element method. To do this, a mesh adaption scheme is applied which decomposes the space of random inputs if a certain threshold is reached for the relative error in the variance [9]. The gPC method is then applied to each element in random space. This allows for a probabilistic kind of grid refinement in the case of discontinuities in the random space. The procedure was successfully applied to the Kraichnan-Orszag three-mode problem and a stochastic advection-diffusion equation.

Vos and Van der Steen have adopted a different approach to deal with long term integration and discontinuities in the random space. They recognized that a significant part of the unstable behaviour originated from the fact that the distribution assumed at the initial state, was not necessarily equal to the distribution further in time. In most cases, this random distribution can and will become increasingly different from the original state. By recalculating the orthogonal basis using the Gram-Schmidt procedure, they ensured an optimal trial basis throughout the integration. Vos applied this method to a one dimensional growth equation with uniformly distributed reproduction rate [1]. Van der Steen improved the algorithm and applied it to the above mentioned Kraichnan-Orszag three-mode problem, with excellent results [2]. This TDgPC method is described in more detail in Chapter 3.

Another way of coping with the aforementioned problems of gPC was initiated by Mathelin, Hussaini and Zang [10], with a method termed stochastic collocation (SC). There are many similarities with the gPC method, e.g. a one dimensional gPC simulation with polynomials of order N will yield the same results as a SC simulation with $N + 1$ collocation points. But the main difference with the gPC method is that collocation points in the random space are used in combination with polynomial interpolation, instead of the aforementioned orthogonal polynomial expansion. This procedure was refined by both Babuška et. al. and Loeven et. al. and was termed the Probability Collocation Method (PCM) by both [11, 12]. The main advantage of the PCM over the gPC is threefold. First of all, the method results in a non-coupled system of equations, which makes it a non-intrusive method. As a result, it facilitates the implementation of the method, since an existing solver can be used as a black box operator. The intrusive gPC requires far more effort to implement. Secondly, PCM can cope with non-linear equations more easily than the gPC method. This stems from the fact that in the gPC method, a Galerkin projection is performed that can be simplified for the linear case (due to the orthogonality of the polynomials), but which has to be numerically evaluated for non-linear equations. PCM circumvents this problem. Thirdly, for a small number of random variables, the PCM is computationally less expensive than the gPC method. However, the PCM suffers 'from the so-called curse of dimensionality,' as Babuška states [11]. Meaning that, when using more random variables, the computational cost increases rapidly and gPC will again be more efficient.

Foo, Wan and Karniadakis have incorporated the aforementioned multi-element method into the PCM procedure to form the multi-element probabilistic collocation method (ME-PCM). When applied to, among other cases, a noisy flow past a 2D stationary circular cylinder, they reported a higher accuracy and a lower computational cost compared to the ME-gPC method [13].

Chapter 3

Time-Dependent generalized Polynomial Chaos

This chapter will give an overview of the TDgPC method. First of all, the gPC procedure is described, then the extension to the time-dependent method is explained.

3.1 Generalized polynomial chaos

Let $(\Omega, \mathcal{F}, \mathcal{P})$ be a complete probability space, where Ω denotes the sample space, $\mathcal{F} \subset 2^{\Omega}$ is the σ-algebra of subsets of Ω, and $\mathcal{P}$ is the associated probability measure. Furthermore, let $D \subset \mathbb{R}^d \times T\,(d = 1, 2, 3)$ be a combination of spatial and temporal dimensions. Then a stochastic process can be seen as function $\mathbf{u}(\mathbf{x}, t, \omega) : D \times \Omega \to \mathbb{R}^b$. Here, $\mathbf{x}$ denotes an element of physical space, t is a point in time and ω is a point in stochastic space. Since a continuous random space is quite unwieldy, we describe this space by a finite number of random variables

$$\xi_1, \xi_2, \ldots, \xi_n : \Omega \to \mathbb{R}^b$$

With this, the stochastic process can be regarded as

$$\mathbf{u}(\mathbf{x}, t, \xi) : D \times \mathbb{R}^n \to \mathbb{R}^b$$

where ξ is an $n-$dimensional vector of random variables.

A general random process can then be expanded as

$$\mathbf{u}(\mathbf{x}, t, \boldsymbol{\xi}(\omega)) = \sum_{i=0}^{\infty} \mathbf{u}_i(\mathbf{x}, t)\Phi_i(\boldsymbol{\xi}(\omega)) \tag{3.1}$$

where ω is the random event, and $\Phi_i(\boldsymbol{\xi}(\omega))$ are polynomial functionals in terms of the n-dimensional vector of random variables $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_n)$. The family $\{\Phi_i(\boldsymbol{\xi}(\omega))\}$ forms a complete orthogonal basis in $L^2(\Omega, \mathcal{F}, \mathcal{P})$ with orthogonality relation

$$\langle \Phi_i, \Phi_j \rangle = \langle \Phi_i^2 \rangle \delta_{ij}$$

where δ_{ij} denotes the Kronecker delta and $\langle \cdot, \cdot \rangle$ denotes the ensemble average. This statistical average can be regarded as the inner product in the Hilbert space. The inner product then takes on its usual form

$$\langle F(\boldsymbol{\xi}), G(\boldsymbol{\xi}) \rangle = \int_{\mathrm{supp}(f(\boldsymbol{\xi}))} F(\boldsymbol{\xi}) G(\boldsymbol{\xi}) f_{\boldsymbol{\xi}}(\boldsymbol{\xi}) d\boldsymbol{\xi} \tag{3.2}$$

or

$$\langle F(\boldsymbol{\xi}), G(\boldsymbol{\xi})\rangle = \sum_{\boldsymbol{\xi}} F(\boldsymbol{\xi})G(\boldsymbol{\xi})f_{\boldsymbol{\xi}}(\boldsymbol{\xi})$$

in the discrete case. Important is the $f_{\boldsymbol{\xi}}(\boldsymbol{\xi})$ term. It is the PDF of the random variables of $\boldsymbol{\xi}$ but serves as a weighting function in the orthogonality relation for $\{\Phi_i(\boldsymbol{\xi}(\omega))\}$. This is the direct link between the PDF of the random variables and the orthogonal polynomials, since the latter are directly related to the weighting function. An overview of related PDF's and orthogonal polynomials can be found in Table 3.1. According to Cameron and Martin, these expansion converge to any stochastic process in the L^2 sense [14].

Table 3.1: Relation between the distribution of the random variables and the optimal trial basis [8]

	PDF distribution of $\boldsymbol{\xi}$	Expansion polynomials $\{\Phi_i(\boldsymbol{\xi})\}$
Continuous	Gaussian	Hermite
	Gamma	Laguerre
	Beta	Jacobi
	Uniform	Legendre
Discrete	Poisson	Charlier
	Binomial	Krawchouk
	Negative binomial	Meixner
	Hypergeometric	Hahn

However, as stated in Section 2.2, these relations can be generalized to truly arbitrary random distributions. If the PDF of the random variables is known, an optimal trial basis can be generated through any orthogonalization technique. A straightforward and commonly used procedure is the Gram-Schmidt method. Taking $\phi_0 = 1$, the higher order (for clarity, one dimensional) orthogonal polynomials up to order P are generated as

$$\phi_p(\xi) = \xi^p - \sum_{k=0}^{p-1} c_{pk}\phi_k(\xi), \qquad p = 1, 2, \ldots, P$$

with

$$c_{pk} = \frac{\langle \xi^p \phi_k(\xi)\rangle}{\langle \phi_k^2(\xi)\rangle}$$

Note that c_{pk} is the orthogonal projection of ξ^p onto ϕ_k and the ensemble average is the same as defined by Equation 3.2.

The drawback of the Gram-Schmidt method is that it is numerically unstable, i.e. when generated numerically, the polynomials are often not completely orthogonal due to rounding errors. This effect can be negated by doing the procedure a second time using the (almost orthogonal) polynomials from the first run. However, this reorthogonalization obviously doubles the computational cost. A second possibility as proposed by Van der Steen is to use a modified Gram-Schmidt procedure. In the classical Gram-Schmidt method, ϕ_1 is taken as the first entry, which is orthogonalized to all polynomials of lower degree. The procedure is then repeated for $\phi_2 \ldots \phi_P$, again orthogonalizing this entry to all polynomials of lower degree, see Figure 3.1. In the modified Gram-Schmidt method, the algorithm starts with ϕ_0 and updates all orthogonal polynomials of higher degree to be orthogonal to ϕ_0. The same is then done for $\phi_1 \ldots \phi_P$, where ϕ_P is only orthogonalized to ϕ_{P-1}. This modified algorithm is less sensitive to rounding errors. Van der Steen tested all three methods (classical Gram-Schmidt, classical Gram-Schmidt with reorthogonalization and modified Gram-Schmidt) for $P = 2$ and found no significant differences, but suggested that the modified Gram-Schmidt algorithm could be more accurate for basis polynomials of higher degree [2].

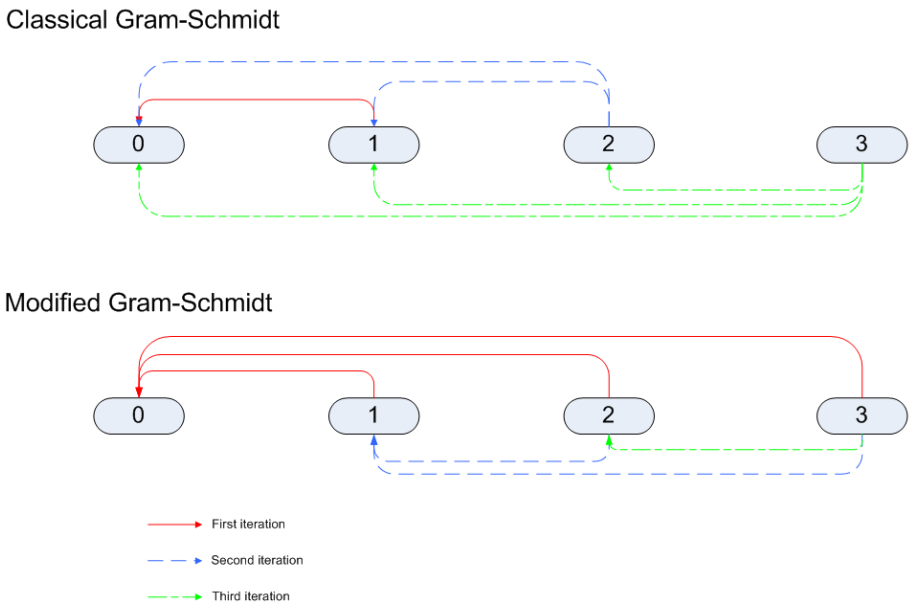


Figure 3.1: Difference between the classical and modified Gram-Schmidt procedure. The polynomial from which the arrow originates is the one being updated.

To keep the simulation computationally feasible, the infinite series of Equation 3.1 has to be truncated. The expansion polynomials are truncated at their maximum degree, P. Denoting the dimensionality of the random vector $\boldsymbol{\xi}$ by n, it can be proven that the P, n and the number of terms in the expansion $(M+1)$ are related as [2]

$$M+1 = \begin{pmatrix} P+n \\ P \end{pmatrix} = \frac{(P+n)!}{P!n!}$$

Equation 3.1 can then be rewritten as

$$\mathbf{u}(\mathbf{x}, t, \boldsymbol{\xi}(\omega)) = \sum_{i=0}^{M} \mathbf{u}_i(\mathbf{x}, t)\Phi_i(\boldsymbol{\xi}(\omega)) \tag{3.3}$$

which can be substituted into a general differential equation

$$\mathcal{L}(\mathbf{x}, t, \boldsymbol{\xi}(\omega); \mathbf{u}) = \mathbf{g}(\mathbf{x}, t, \boldsymbol{\xi}(\omega)) \tag{3.4}$$

Here, $\mathcal{L}$ denotes a differential operator and $\mathbf{g}$ is a source term. The substitution yields

$$\mathcal{L}\left(\mathbf{x}, t, \boldsymbol{\xi}; \sum_{i=0}^{M} \mathbf{u}_i(\mathbf{x}, t)\Phi_i(\boldsymbol{\xi})\right) = \mathbf{g}(\mathbf{x}, t, \boldsymbol{\xi}) \tag{3.5}$$

The ω is dropped for convenience. To turn this stochastic equation into a set of $M+1$ coupled, deterministic equation for $M+1$ expansion coefficients, a Galerkin projection is performed, i.e. Equation 3.5 is multiplied by every polynomial of the expansion basis $\{\Phi_i\}$. Afterwards, the statistical average is taken

$$\left\langle \mathcal{L}\left(\mathbf{x}, t, \boldsymbol{\xi}; \sum_{i=0}^{M} \mathbf{u}_i(\mathbf{x}, t)\Phi_i(\boldsymbol{\xi})\right), \Phi_j(\boldsymbol{\xi}) \right\rangle = \langle \mathbf{g}(\mathbf{x}, t, \boldsymbol{\xi}), \Phi_j(\boldsymbol{\xi}) \rangle, \qquad j = 0, 1, \ldots, M \tag{3.6}$$

This set of equations can be solved through standard methods, e.g. a Euler or Runga-Kutta integration in time and a spectral/hp element method in space.

3.2 Gauss-Lobatto-Legendre quadrature

Since the Galerkin projection results in a system of equations involving integrated polynomials, a numerical integration routine is used to evaluate these integrals. The preferred method to do this, is the Gauss-Lobatto-Legendre (GLL) algorithm. It is a very accurate integration scheme for integrals of the form

$$\int_{-1}^{1} u(x)\mathrm{d}x \tag{3.7}$$

especially if the integrand is smooth [15]. In this procedure, the integrand is represented as a Lagrange polynomial using Q points x_i, which are to be specified, i.e.

$$u(x) = \sum_{i=0}^{Q-1} u(x_i)h_i(x) + \epsilon(u)$$

where $\epsilon(u)$ is the integration error. Substituting this into Equation 3.7 yields

$$\int_{-1}^{1} u(x)\mathrm{d}x = \sum_{i=0}^{Q-1} w_i u(x_i) + R(u)$$

where

$$w_i = \int_{-1}^{1} h_i(x)\mathrm{d}x$$

$$R(u) = \int_{-1}^{1} \epsilon(u)\mathrm{d}x$$

It is natural to assume that when integrating a polynomial $u(x)$ of order $Q-1$ or less, the integration is exact. In other words, $R(u) = 0$ if $u(x) \in \mathcal{P}_{Q-1}([-1,1])$. However, with a specific choice of integration points, an exact integration can be achieved for polynomials of order higher than $Q-1$. This surprising result was discovered by Gauss and is the core of Gaussian quadrature.

The Gauss-Lobatto-Legendre method is one of a few derived Gauss quadrature procedures, and differs only in the choice of the end points. Now let $x_{i,P}^{\alpha,\beta}$ be the P zeros of the Pth-order Jacobi polynomial such that

$$P_P^{\alpha\beta}(x_{i,P}^{\alpha,\beta}) = 0\ , \qquad i = 0, 1, \ldots, P-1,$$

where

$$x_{0,P}^{\alpha,\beta} < x_{1,P}^{\alpha,\beta} < \ldots < x_{P-1,P}^{\alpha,\beta}$$

These zeros usually cannot be calculated analytically, so they are either tabulated or attained through a numerical algorithm themselves. The Gauss-Lobatto-Legendre quadrature can now be defined by

$$\begin{aligned}
x_i &= \begin{cases} -1 & i = 0, \\ x_{i-1,Q-2}^{1,1} & i = 1, \ldots, Q-2 \\ 1 & i = Q-1 \end{cases} \\
w_i &= \frac{2}{Q(Q-1)[L_{Q-1}(x_i)]^2} \qquad i = 0, \ldots, Q-1 \\
R(u) &= 0 \qquad \text{if } u(x) \in P_{2Q-3}([-1,1])
\end{aligned}$$

where $L_{Q-1}(x_i)$ is the Legendre polynomial of order $Q-1$. The last line of the above equation states that the integration is exact if the order of the polynomial is less than $2Q-1$. In other words, taking a fixed order P of the integrand yields

$$Q_{\min} \geq \frac{P+1}{2}$$

Naturally, this is a very desirable property when integrating polynomials.

3.3 Adjustments to transform gPC into TDgPC

One of the problems of the gPC method is that it fails for long-term integration. As mentioned before, this is caused by a change in the distribution of the random variables over time, also known as stochastic drift. One way to deal with this issue is to recalculate the trial basis polynomials after a certain amount of time steps. Vos investigated the possibility of recalculating the trial basis at a threshold criterion for the non-linear terms in the expansion, while Van der Steen experimented with a recalculation after a fixed time, e.g. after four seconds or even after every time step iteration. The result turned out to be too case sensitive to draw conclusions.

To use the TDgPC method, one starts with applying the gPC method to a certain differential equation, i.e. Equation 3.4. After a certain time, say $t = t^*$, the original trial basis loses its efficiency and accuracy and the trial basis is recalculated. To do this, a transformation of the random variable is performed

$$\boldsymbol{\zeta} = \mathbf{u}(\mathbf{x}, t, \boldsymbol{\xi}) = \sum_{i=0}^{M} \mathbf{u}_i(\mathbf{x}, t)\Phi_i(\boldsymbol{\xi})$$

The vector of new random variables $\boldsymbol{\zeta}$ has a new PDF $f_{\boldsymbol{\zeta}}(\boldsymbol{\zeta})$. For each f_{ζ_i} the Gram-Schmidt orthogonalization procedure can be used to produce a new trial basis $\phi_p^{\zeta_i}(\zeta_i)$, with $p = 0, \ldots, P$. This can then be used for the integrations in time and space, as for the gPC.

If the transformation of the random variable is denoted as $\zeta = Z(\xi)$ (assuming one dimension in random space for convenience), one can calculate the new PDF's as [16]

$$f_\zeta(\zeta) = \sum_{\xi_n} \frac{f_\xi(\xi_n)}{\left|\frac{\mathrm{d}Z(\xi)}{d\xi}|\xi = \xi_n\right|}$$

The computation of this PDF is a relatively expensive and can be quite sensitive to round-off and interpolation errors. So ideally, one would like to avoid the calculation of this PDF.

The way Van der Steen avoided this recalculation of $f_\zeta(\zeta)$ was by keeping the integrations needed for the Galerkin projection and the calculation of the statistical moments (i.e. the mean and variance) in terms of $\boldsymbol{\xi}$, instead of the new random variables $\boldsymbol{\zeta}$. To be able to do this, he recognized that in the case of a transformation to three random variables, such as in the Kraichnan-Orszag problem, the following must hold for every realizable point $(\zeta_1^*, \zeta_2^*, \zeta_3^*)$

$$f_{\zeta_1,\zeta_2,\zeta_3}(\zeta_1^*, \zeta_2^*, \zeta_3^*)\mathrm{d}\zeta_1\mathrm{d}\zeta_2\mathrm{d}\zeta_3 = \sum_{\xi^*} f_\xi(\xi^*)\mathrm{d}\xi \tag{3.8}$$

where the summation is over all points ξ^* for which $Z_1(\xi^*) = \zeta_1^*$, $Z_2(\xi^*) = \zeta_2^*$ and $Z_3(\xi^*) = \zeta_3^*$. What Equation 3.8 expresses, is that the probability of $(\zeta_1, \zeta_2, \zeta_3)$ being inside the 'volume' $\mathrm{d}\zeta_1^*\mathrm{d}\zeta_2^*\mathrm{d}\zeta_3^*$, should be equal to the probability of ξ being inside the summation of 'volumes' $\mathrm{d}\xi^*$. See Figure 3.2 for a graphical interpretation in the case of a transformation to two random variables. The reason that there can be several $\mathrm{d}\xi^*$-volumes, is that the transformation $\zeta = Z(\xi)$ can have several roots. If Equation 3.8 holds, then it follows that

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \cdots f_{\zeta_1,\zeta_2,\zeta_3}\mathrm{d}\zeta_1\mathrm{d}\zeta_2\mathrm{d}\zeta_3 = \int_{-1}^{1} \cdots f_\xi \mathrm{d}\xi \tag{3.9}$$

Note that the limits for the integration over the original variable depend on the basis for the initial distribution. Since Van der Steen assumed a uniform distribution over $[-1, 1]$, the integration is obviously also taken over this interval.

Apart from the very desirable consequence that the computationally expensive calculation of the PDF is not performed, Equation 3.9 also means that the calculation of the statistical moments becomes less complicated. This reduction in computing time made the recalculation of the orthogonal basis at every time step feasible.

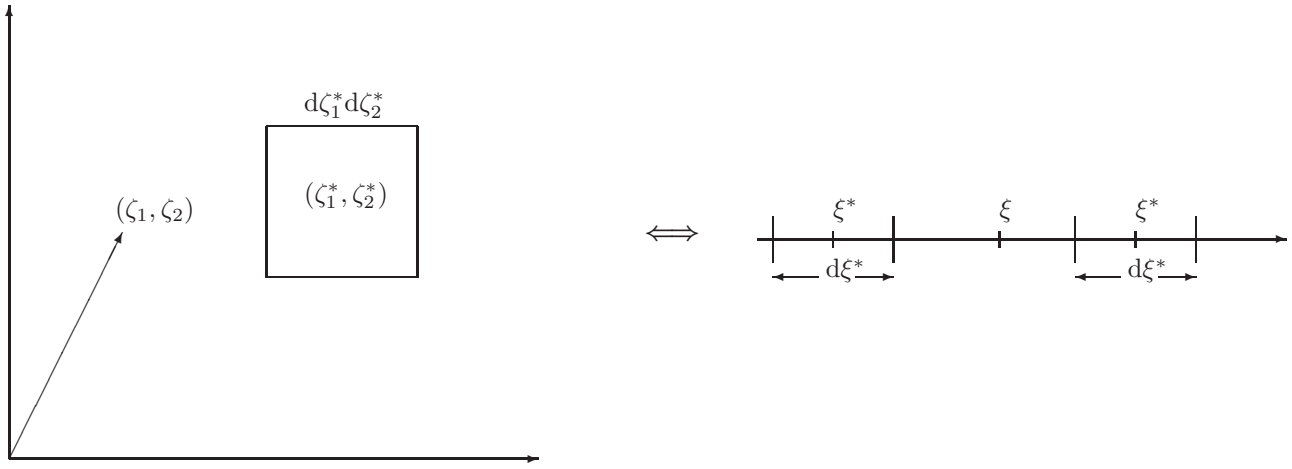


Figure 3.2: Graphical representation of Equation 3.8 for a transformation to two random variables.

Chapter 4

TDgPC applied to the growth equation

The growth equation was chosen as the object of this research. Although it is a simple equation from a programming point of view, it is quite challenging in its stochastic behaviour. The equation represent e.g. the growth of a population.

A new formulation of the TDgPC method is proposed, leading to significant improvements in accuracy.

4.1 Problem definition

The growth equation is defined as

$$\frac{\mathrm{d}u(t,\xi)}{\mathrm{d}t} + k(\xi)u(t,\xi) = 0, \qquad u(0,\xi) = 1, \qquad t \geq 0, \qquad -1 \leq \xi \leq 1 \tag{4.1}$$

with the growth factor

$$k(\xi) = \frac{1}{2}(1+\xi)$$

The stochastic variable ξ is uniformly distributed over $[-1,1]$ with $f(\xi) = \frac{1}{2}$, which means that k has an average of $\mu_k = \frac{1}{2}$. So strictly speaking, this is a decay problem. However, both Vos and Van der Steen dubbed it the growth equation. To avoid confusion, this convention is adopted.

One of the advantages of studying this equation, is that it has an analytical solution

$$u_{exact}(t,\xi) = e^{-k(\xi)t}$$

which means that the exact mean and variance are given as

$$\begin{aligned} \mu_{exact}(t) &= E[u_{exact}(t,\xi)] \\ &= \int_{-1}^{1} e^{-k(\xi)t} f(\xi)\mathrm{d}\xi \\ &= \frac{1-e^{-t}}{t} \end{aligned}$$

and

$$\begin{aligned} \sigma_{exact}(t) &= E[(u_{exact}(t,\xi) - \mu_{exact}(t))^2] \\ &= \int_{-1}^{1} \left(e^{-k(\xi)t} - \frac{1-e^{-t}}{t}\right)^2 f(\xi)\mathrm{d}\xi \\ &= \frac{1-e^{-2t}}{2t} - \left(\frac{1-e^{-t}}{t}\right)^2 \end{aligned}$$

4.2 TDgPC applied to the growth equation

First consider the method as used by Van der Steen and Vos. Substitute the polynomial expansion into Equation 4.1

$$\frac{\mathrm{d}}{\mathrm{d}t}\sum_{i=0}^{P} u_i(t)\phi_i(\xi) = -\sum_{i=0}^{P} k(\xi)u_i(t)\phi_i(\xi)$$

Applying the Galerkin projection yields

$$\frac{\mathrm{d}u_j}{\mathrm{d}t} = \frac{-\sum_{i=0}^{P} u_i\langle k\phi_i, \phi_j\rangle}{\langle\phi_j^2\rangle}, \qquad j = 0\ldots P$$

After a certain time t^*, a transformation of the stochastic variable from ξ to ζ is performed

$$\zeta = u(t^*, \xi) = \sum_{i=0}^{P} u_i(t^*)\phi_i(\xi) \tag{4.2}$$

Next, a new polynomial basis $\left\{\tilde{\phi}(\zeta)\right\}$ is calculated using the aforementioned Gram-Schmidt procedure. The 'initial' conditions (conditions at $t = t^*$) are updated accordingly

$$u_i(t^*) = \begin{cases} -\tilde{\phi}_{1,0} & \text{if} \quad i = 0 \\ 1 & \text{if} \quad i = 1 \\ 0 & \text{otherwise} \end{cases}$$

4.3 TDgPC shortcomings

This method was used by Van der Steen to investigate the effect of certain parameters on the accuracy of the solution, which revealed that a transformation at every time-step was not the optimal setting. It was found that when using a higher polynomial order, the accuracy improved when transforming less often. This was curious, since one would expect that the transformation benefits the accuracy, regardless of polynomial order and the number of transformations, see also Figure 4.1. The relative errors are defined as

$$\epsilon_{mean} = \left|\frac{\mu_{exact} - \mu_{TDgPC}}{\mu_{exact}}\right|, \qquad \epsilon_{var} = \left|\frac{\sigma_{exact} - \sigma_{TDgPC}}{\sigma_{exact}}\right|$$

These observations let to the belief that the method has some shortcomings, at least when applied to this specific problem. The most prominent one is described in the next section.

4.3.1 Polynomial expansion of the natural logarithm

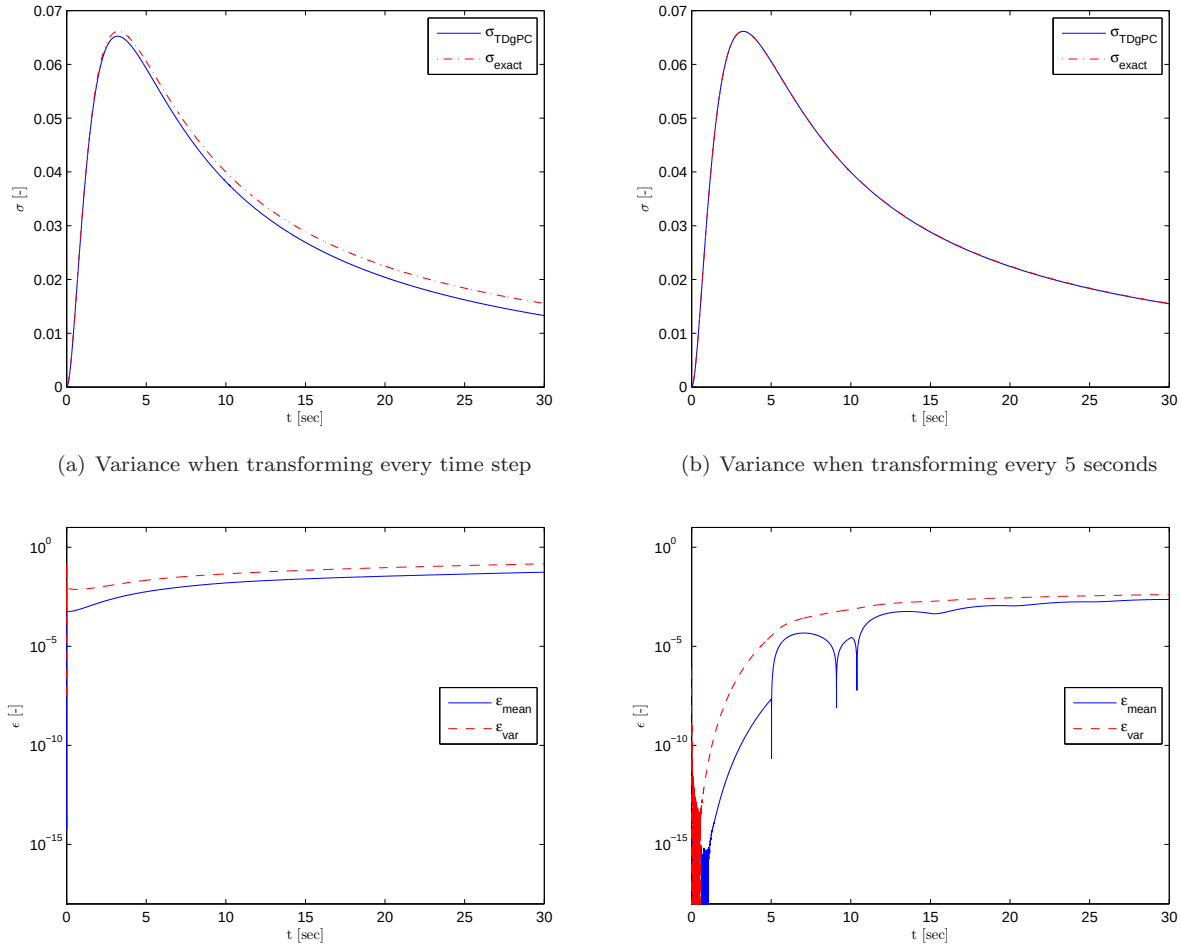
First of all, when transforming the stochastic variable as shown in Equation 4.2, ζ would ideally behave as

$$\zeta(\xi) = e^{-(1+\xi)\frac{t}{2}}, \qquad e^{-t} \leq \zeta \leq 1$$

since this is the exact solution. This means that

$$k(\xi) = \frac{1}{2}(1+\xi) = -\frac{1}{t}\ln\zeta$$

Because the TDgPC method tries to expand the stochastic process, i.e. $k(\xi)$, using polynomials, the above statement actually implies that one tries to approximate the natural logarithm through polynomials. Here lies a problem.



(a) Variance when transforming every time step

(b) Variance when transforming every 5 seconds

(c) Errors when transforming every time step

(d) Errors when transforming every 5 seconds

Figure 4.1: Results for the TDgPC scheme, using $P = 5$, $Q = 100$

For this distribution of $k(\xi)$, the corresponding transformed PDF is given by

$$f(\zeta, t) = \frac{1}{\zeta t}, \qquad e^{-t} \leq \zeta \leq 1$$

and $k(\xi)$ can be expressed as

$$k(\xi) = -\frac{1}{t} \ln \zeta = \sum_{i=0}^{\infty} a_i \phi_i(\zeta)$$

Since the first of the monic polynomials is equal to 1, the other polynomials can be calculated analytically. For example, the first order polynomial is

$$\phi_1(\zeta) = \zeta - \frac{1}{t} + \frac{1}{t} e^{-t}$$

The expressions for the higher order polynomials become increasingly elaborate.

Plotting the expansion, it becomes clear from Figure 4.2 that the expansion cannot follow the logarithm as

it gets closer to $\zeta = 0$, i.e. as time increases. This implies that as time progresses, the polynomial expansion has more and more trouble following the behaviour of the exact solution. This again explains the observed increase of the error in time of any gPC method. The current TDgPC procedure is no exception.

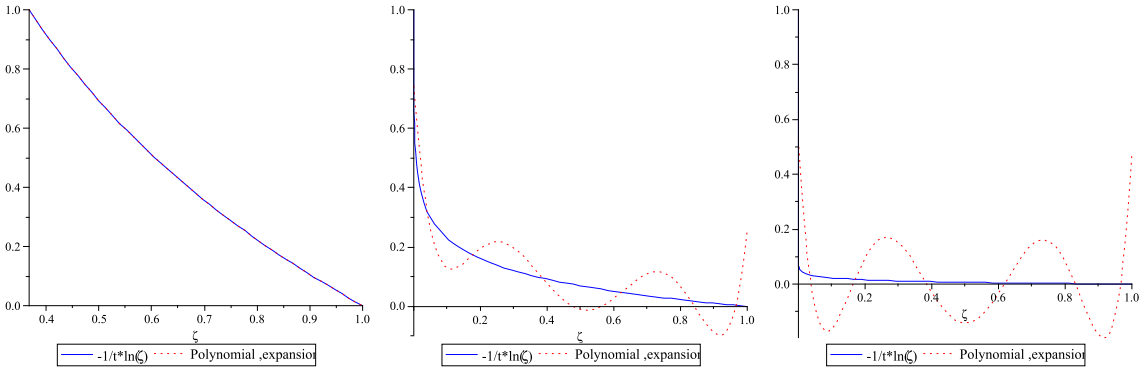


Figure 4.2: Natural logarithm and its 6^{th} order expansion for $t = 1$, $t = 10$, $t = 100$ (left to right)

4.4 Adjustment of the TDgPC method

The first step to tackle these problems, is to recognize that the PDF of the solution actually differs from the PDF of $k(\xi)$, i.e. ξ is uniformly distributed over $[-1, 1]$ regardless of time. The solution on the other hand, starts with a deterministic initial condition, implying a very 'slim' distribution, and proceeds to become increasingly 'uncertain,' which would mean a 'wide' distribution, see Figure 4.3.

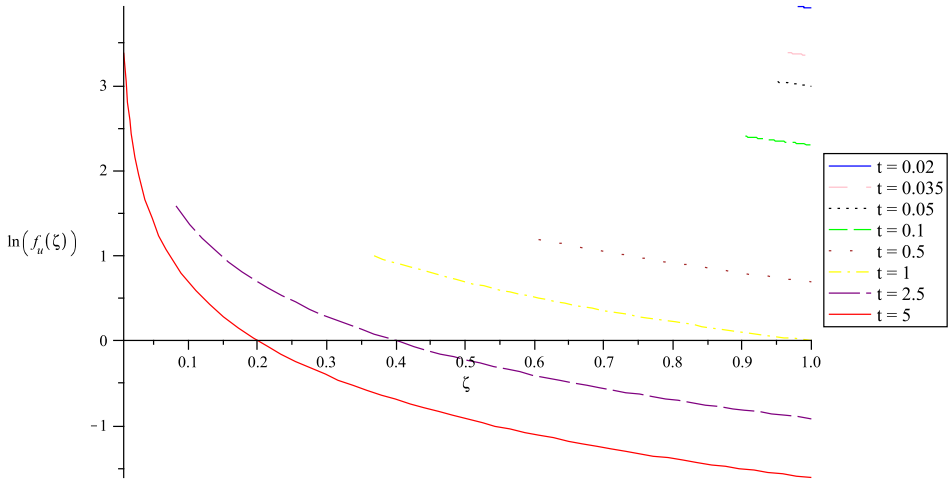


Figure 4.3: Logarithm of PDF's of the solution for various instants in time

Therefore, the transformed stochastic variable ζ associated with the solution should be introduced and updated in time. This differs from the original TDgPC approach, since in that method, ζ was directly linked to the distribution of the growth coefficient k.

The proposed expansion of u then is

$$u(t,\xi) = \zeta(t^*,\xi)\sum_{i=0}^{\infty} u_i(t)\phi_i(\xi), \qquad \zeta(t^*,\xi) = u(t^*,\xi) \tag{4.3}$$

Substituting into Equation 4.1 yields

$$\zeta(t^*,\xi)\sum_{i=0}^{\infty}\frac{\mathrm{d}u_i(t)}{\mathrm{d}t}\phi_i(\xi) + k(\xi)\zeta(t^*,\xi)\sum_{i=0}^{\infty}u_i(t)\phi_i(\xi) = 0$$

Since $\zeta \neq 0$, this amounts to

$$\sum_{i=0}^{\infty}\frac{\mathrm{d}u_i(t)}{\mathrm{d}t}\phi_i(\xi) + k(\xi)\sum_{i=0}^{\infty}u_i(t)\phi_i(\xi) = 0$$

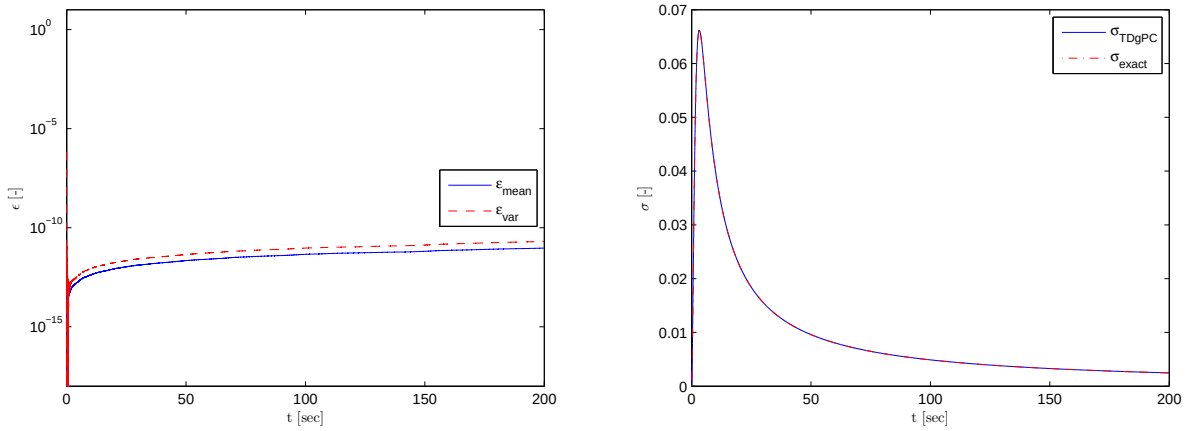
which is exactly the gPC formulation and can thus be solved with the now familiar Galerkin projection. The difference is then that once the coefficients at the new time step have been calculated, this gPC solution needs to be multiplied by $\zeta(\xi,t^*)$, i.e. u at the previous time step. The mean and variance can then be calculated in the normal fashion. After this, the ζ for the next time step can be calculated and the 'initial' conditions (conditions at $t = t^*$) are updated according to

$$u_i(t^*,\xi) = \begin{cases} 1 & \text{if} \quad i = 0 \\ 0 & \text{otherwise} \end{cases}$$

This has the added advantage of a very low order approximation, since only two polynomials are needed to capture $k(\xi)$.

Another advantage of this approach, is that one does not need to recalculate the orthogonal polynomial base. This is particularly beneficial for the treatment of PDE's, where one needs to recalculate this base not only for every time step, but also every grid point in the spatial domain.

The results for this approach are shown in Figure 4.4. Using a fourth order Runge-Kutta (RK4) time integration with $\Delta t = 0.001$ and transforming every time step, the errors are below 10^{-10} throughout the whole time integration, even up to $t = 200$.



(a) Errors

(b) Variance

Figure 4.4: Results for the adjusted TDgPC scheme, using $P = 5$, $Q = 100$

Chapter 5

TDgPC applied to the heat equation

This chapter will describe the polynomial chaos method as applied to a PDE known as the heat equation. The method as introduced in Chapter 4 was generalized to be applicable to a wider range of equations. Also, a convergence study was performed.

5.1 Definition of the heat equation

The heat equation is a typical example of a diffusion equation. It is related to the growth equation, but it is extended to the spatial domain. The following form was chosen

$$\frac{\partial u(x,t,\xi)}{\partial t} = \kappa(\xi)\frac{\partial^2 u(x,t,\xi)}{\partial x^2}, \qquad 0 \leq x \leq 1, \quad t \geq 0, \quad -1 \leq \xi \leq 1$$

with an uncertain heat coefficient $\kappa = \frac{1}{2}(1+\xi)$ uniformly distributed over $[-1,1]$ and deterministic initial condition

$$u(x,0) = \sin(2\pi x)$$

As for the growth equation, an analytical solution is known

$$u_{exact}(x,t,\xi) = \sin(2\pi x)e^{-4\pi^2\kappa(\xi)t}$$

This again facilitates the error analysis, since we can calculate the mean and variance of the solution analytically as

$$\mu_{exact}(x,t) = \sin(2\pi x)\frac{1-e^{-4\pi^2 t}}{4\pi^2 t} \qquad \sigma_{exact} = \sin^2(2\pi x)\left(\frac{1-e^{-8\pi^2 t}}{8\pi^2 t} - \left(\frac{1-e^{-4\pi^2 t}}{4\pi^2 t}\right)^2\right)$$

5.2 From growth equation to heat equation

When attempting to apply the method of Section 4.4 to the heat equation, a few of its drawbacks became apparent. The approach proved to be

- awkward to implement for a PDE, since it raised similar questions about the spatial derivative of a stochastic variable, as it did for the time derivative,
- disastrous for equations where the solution becomes (too close to) zero. Since ζ will be set to zero, it will keep the solution at zero. This way, ζ forces the solution to remain zero, even though the analytical solution might not. Think for example of a harmonic oscillator, where the amplitude of the swing will 'go through zero.'

- in general, too case-tailored.

Therefore, a generalization was sought.

5.3 Generalized formulation

This generalization was found in the following procedure:

1. Define the first $P+1$ polynomials of the expansion as if applying the gPC method. So for a uniform distribution, construct $P+1$ Legendre polynomials up to order P. These will be denoted as $\phi_i(\xi)$, $i = 0, \ldots, P$

2. Define the next $P+1$ polynomials as

   $$\hat{\phi}_{P+i+1}(\xi) = \zeta(\xi)\phi_i(\xi), \qquad i = 0, \ldots, P$$

   This means that, e.g. for $P = 1$ and a uniform distribution, the four polynomials will look like

   $$\phi_0 = 1, \quad \phi_1 = \xi, \quad \hat{\phi}_2 = \zeta, \quad \hat{\phi}_3 = \zeta\xi \tag{5.1}$$

3. In general, these polynomials are not orthogonal. But because orthogonality is a very desirable trait for the rest of the analysis, we apply a Gramm-Schmidt procedure to set up an orthogonal system $\phi_i(\xi)$ for $i = 0, \ldots, 2P+1$, using

   $$\phi_i(\xi) = \hat{\phi}_i - \sum_{k=0}^{i-1} \frac{\langle \hat{\phi}_i, \phi_k \rangle}{\langle \phi_k^2 \rangle} \phi_k, \qquad i = P+1, \ldots, 2P+1$$

   If however, the norm of the new function is sufficiently small, i.e. $\langle \phi_i^2 \rangle \leq \epsilon$ for small ϵ, we consider ϕ_i a linear combination of the previous polynomials. Therefore, this polynomial is discarded.

4. Calculate the according new initial conditions by projecting the solution onto the basis vector

   $$u_i(t^*) = \frac{\langle u(t^*)\phi_i \rangle}{\langle \phi_i^2 \rangle}$$

5. Use this new polynomial basis for the next time step.

6. Repeat at next time step, so transform at every time iteration.

Note that $\hat{\phi}_2$ and $\hat{\phi}_3$ in Equation 5.1 represent the method of Section 4.4. However, if $\zeta = 0$, the first two polynomials will prevent the solution from being 'stuck' at zero. This formulation can thus indeed be seen as a generalization of the aforementioned approach.

5.4 Discretization and the Galerkin projection

The above formulation expansion can now be inserted in the heat equation. We use a central discretization scheme in the space domain

$$\frac{\partial u^k}{\partial t} = \kappa \frac{u^{k+1} - 2u^k + u^{k-1}}{\Delta x^2}$$

and the expansion with our new polynomials

$$u^k(t, \xi) = \sum_{i=0}^{2P+1} u_i^k(t)\phi_i^k(\xi)$$

where k denotes the grid point. This yields

$$\sum_{i=0}^{2P+1} \frac{\partial u_i^k}{\partial t}\phi_i^k = \kappa \sum_{i=0}^{2P+1} \frac{u_i^{k+1}\phi_i^{k+1} - 2u_i^k\phi_i^k + u_i^{k-1}\phi_i^{k-1}}{\Delta x^2}$$

Applying the Galerkin projected results in

$$\frac{\partial u_j^k}{\partial t} = \frac{\sum_{i=0}^{2P+1} \left\langle \kappa\phi_i^{k+1}, \phi_j^k \right\rangle u_i^{k+1} - 2\sum_{i=0}^{2P+1} \left\langle \kappa\phi_i^k, \phi_j^k \right\rangle u_i^k + \sum_{i=0}^{2P+1} \left\langle \kappa\phi_i^{k-1}, \phi_j^k \right\rangle u_i^{k-1}}{\Delta x^2 \left\langle \left(\phi_j^k\right)^2 \right\rangle}, \; j = 0, \ldots, 2P+1 \quad (5.2)$$

5.5 Results

Equation 5.2 was discretized in time using an Euler scheme, and integrated over time. The results are shown in Figures 5.1 and 5.2. A comparison was made between gPC using $P = 3$, which means four polynomials, and our new formulation using $2P + 1 = 3$, so also (at most) four polynomials.

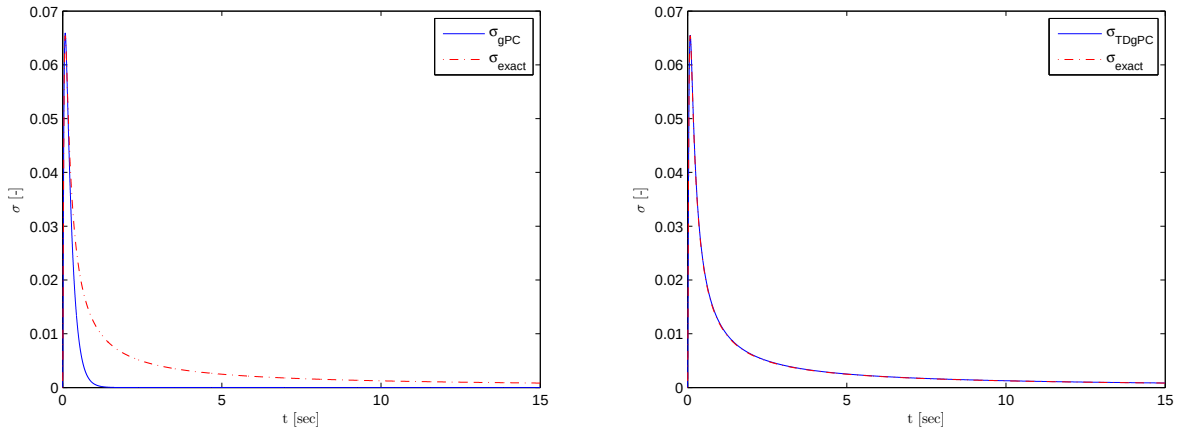
In reality, the third and fourth polynomial are not used every iteration, e.g. for five grid points, using a threshold value of $\epsilon = 10^{-10}$ in the third step of the algorithm and integrating over the first second, 8% of the new polynomials is discarded. This number increases as we let the program integrate over a longer time interval: after fifteen seconds, 25% of the polynomials is not used. From this we can conclude that the new polynomials are especially beneficial for the capturing of the behaviour in the first second, where we are dealing with the steepest slopes.

Figures 5.1 and 5.2 clearly show that the new formulation behaves much better than the gPC scheme. Using only four polynomials, the trend of the variance is captured very decently and the errors in variance and mean have decreased considerably. Figure 5.1a also shows us where much of the accuracy is gained, i.e. in the first second. The gPC method clearly has difficulty following the slope of the curve, whereas the new formulation does not.

To verify that the errors in Figure 5.2 can be reduced by increasing the number of grid points in the spatial domain, a convergence study was performed on the cells size. Figure 5.3 shows the convergence of the error with respect to grid refinement. The different norms are defined as

$$\begin{aligned} |\epsilon_{mean}|_1 &= \sum_k \left(|\mu_{exact} - \mu_{TDgPC}|\Delta x\right) \\ |\epsilon_{mean}|_2 &= \sqrt{\sum_k \left(|\mu_{exact} - \mu_{TDgPC}|\Delta x\right)^2} \\ |\epsilon_{mean}|_\infty &= \max_k \left(|\mu_{exact} - \mu_{TDgPC}|\right) \end{aligned}$$

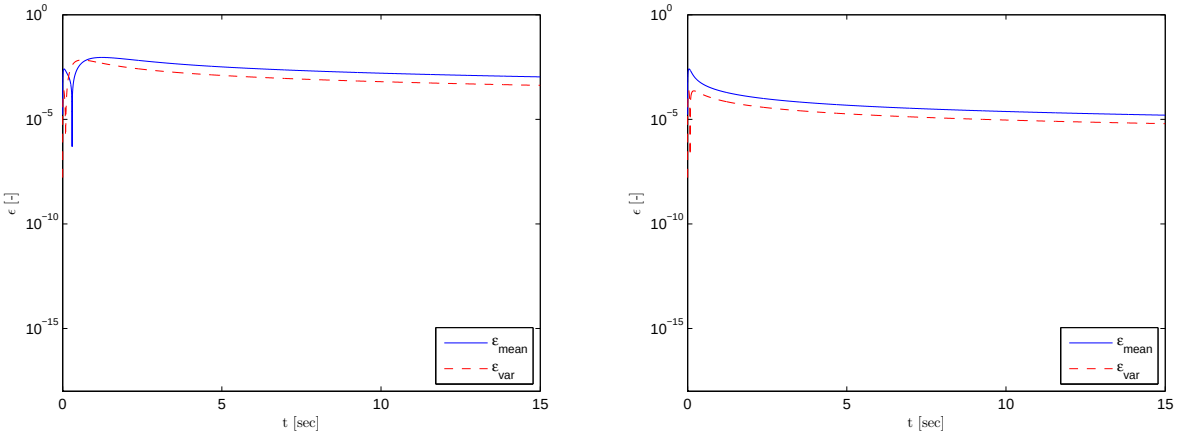
and similarly for the variance. The convergence graphs show a second order h-convergence for all norms, which is to be expected from this central discretization.



(a) classic gPC, $P = 3$

(b) the new formulation, $2P + 1 = 3$

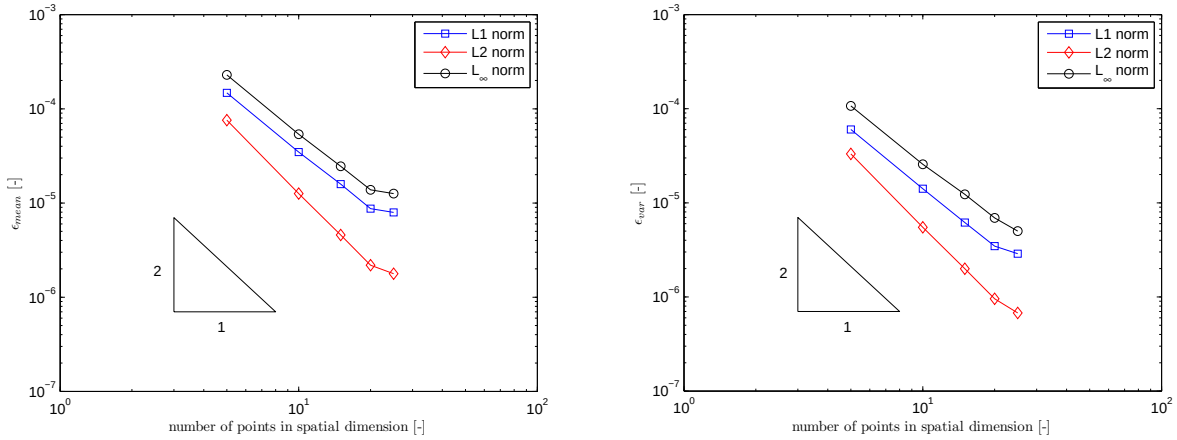
Figure 5.1: Variance at $x = \frac{4}{15}$, 15 elements in spatial domain, $\Delta t = 0.0001$, Q=60



(a) classic gPC, $P = 3$

(b) the new formulation, $2P + 1 = 3$

Figure 5.2: Errors in L^1 sense, 15 elements in spatial domain, $\Delta t = 0.0001$, Q=60



(a) Mean (b) Variance

Figure 5.3: Convergence of the errors in L^1, L^2 and L^∞ sense, $2P+1=3$, $\Delta t = 0.0001$, Q=60, $t_{end} = 15\,s$

5.6 Comparison with semi-discrete analytical system

To confirm that the errors in Figure 5.2 were dominated by the discretization error in the spatial domain and to check the convergence with respect to the time integration, a comparison was made with an analytical semi-discrete system.

Consider the heat equation, centrally discretized in space, but not in time:

$$\frac{\partial u^k}{\partial t} = \kappa(\xi)\frac{u^{k+1} - 2u^k + u^{k-1}}{\Delta x^2}$$

or in matrix notation

$$\frac{\partial}{\partial t}\underline{u} = \kappa(\xi)\underline{\underline{A}}\,\underline{u}$$

This system can be diagonalized and decoupled as

$$\frac{\partial}{\partial t}\underline{\underline{X}}^{-1}\underline{u} = \kappa(\xi)\underline{\underline{X}}^{-1}\underline{\underline{A}}\,\underline{\underline{X}}\,\underline{\underline{X}}^{-1}\underline{u} = \kappa(\xi)\underline{\underline{\Lambda}}\,\underline{\underline{X}}^{-1}\underline{u}$$

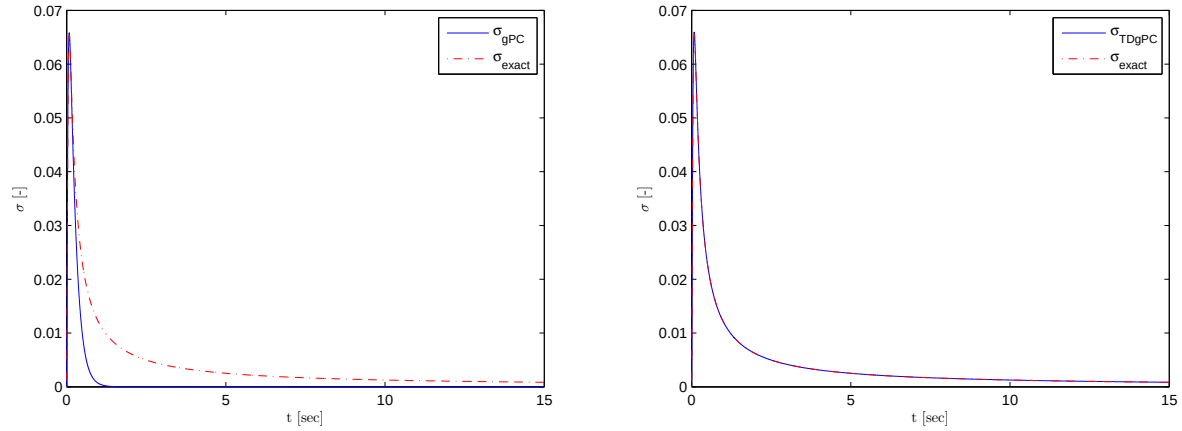
Where $\underline{\underline{\Lambda}}$ is the diagonal matrix containing the eigenvalues of $\underline{\underline{A}}$ and $\underline{\underline{X}}$ is the matrix containing the eigenvectors of $\underline{\underline{A}}$. Now set

$$\underline{\underline{X}}^{-1}\underline{u} = \underline{w}$$

which yields the following set of decoupled equations (note that this is actually a set of growth equations)

$$\frac{\partial w^k}{\partial t} = \kappa(\xi)\lambda^k w^k \qquad \Rightarrow \qquad w^k(t,\xi) = w_0 e^{\kappa(\xi)\lambda^k t} \tag{5.3}$$

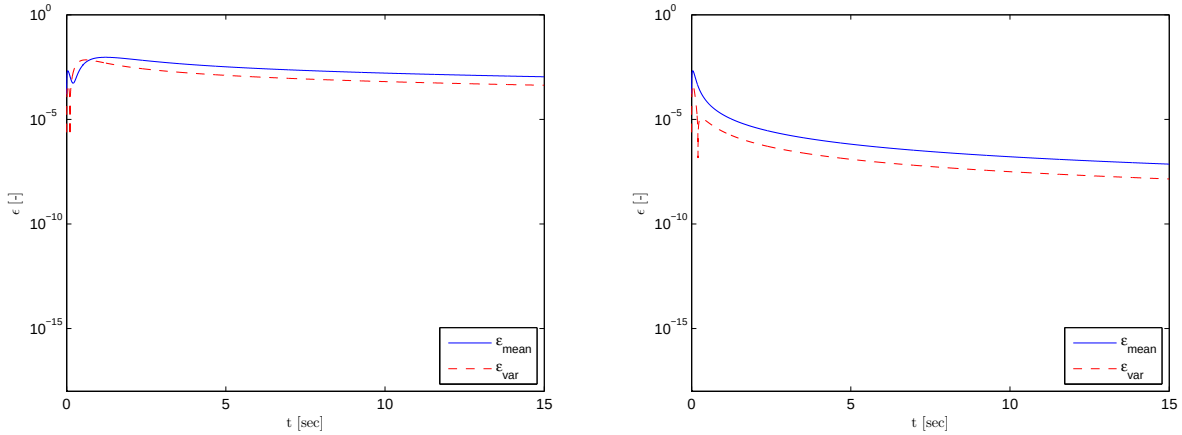
These analytical solutions to the semi-discrete system can be compared to the our TDgPC calculations. The results are shown in Figures 5.4 and 5.5. An Euler integration method was used, which is first order accurate. This meant that the time integration was now the main source of error. Increasing the polynomial order or number of quadrature points did not increase the accuracy of the scheme. This conclusion is confirmed by Figure 5.6, which shows the convergence study for the time step size. As expected, the new method is first order convergent with respect to the time step.



(a) classic gPC, $P = 3$

(b) the new formulation, $2P + 1 = 3$

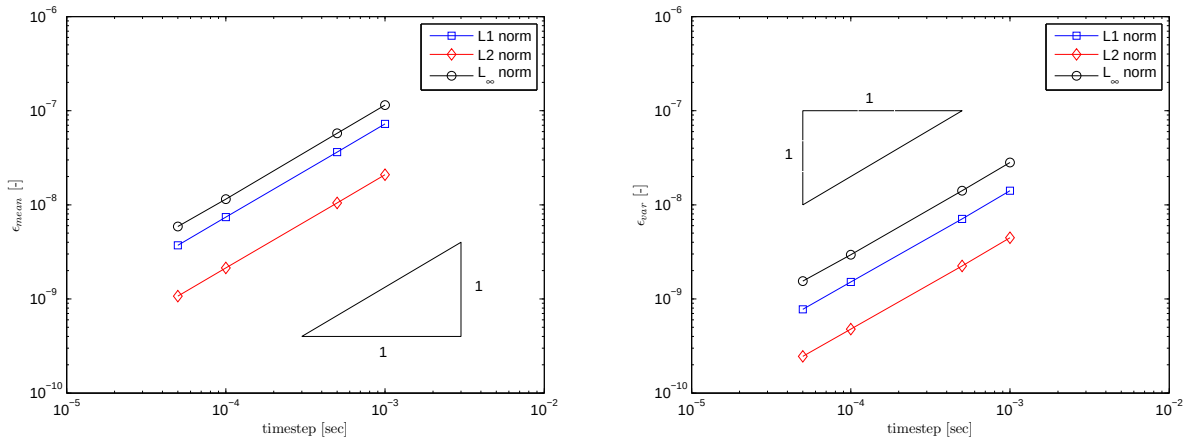
Figure 5.4: Variance at $x = \frac{4}{15}$ for the semi-discrete system, 15 elements in spatial domain, $\Delta t = 0.001$, Q=60



(a) classic gPC, $P = 3$

(b) the new formulation, $2P + 1 = 3$

Figure 5.5: Errors in L^1 sense for the semi-discrete system, 15 elements in spatial domain, $\Delta t = 0.001$, Q=60



(a) Mean (b) Variance

Figure 5.6: Convergence of the errors in L^1, L^2 and L^∞ sense for the semi-discrete system, $2P+1=3$, 15 elements in spatial domain, Q=60, $t_{end} = 15\,s$

5.7 Comparison with fully discrete analytical system

This process can be taken one step further, by comparing the method with the fully discretized system. This allows one to really see the error introduced by the TDgPC method, instead of the discretization error in time and space. Another way would be to use a higher order time discretization scheme, such as the Runge-Kutta method. But because the problem at hand is simple enough to construct a completely objective comparison between gPC and the new TDgPC method, the following procedure was chosen.

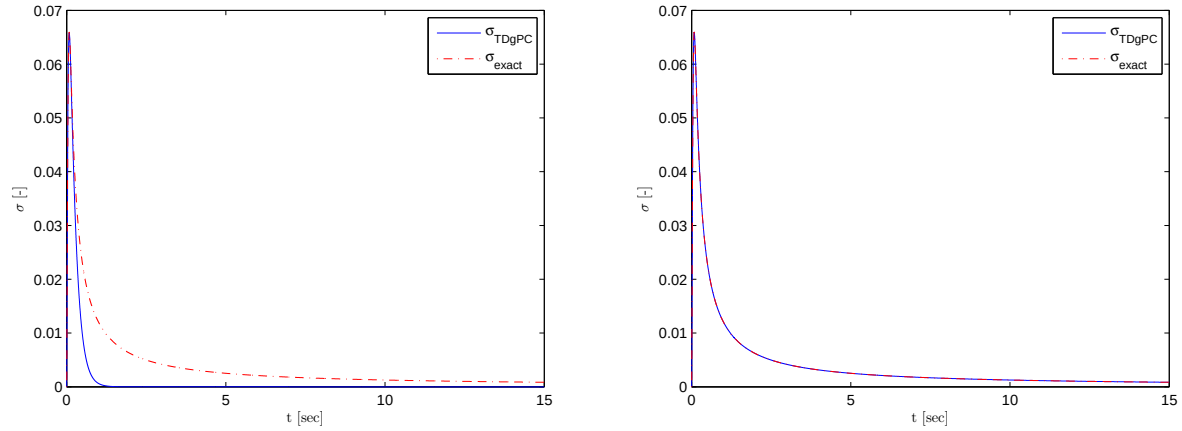
So consider now not the analytical solution of Equation 5.3, but the fully discretized system, using an Euler explicit procedure

$$\frac{w_{n+1}^k - w_n^k}{\Delta t} = \kappa(\xi)\lambda^k w_n^k$$

where superscript k again denotes the grid point and subscript n denotes the solution at time step n. The solution at the new time step is then

$$w_{n+1}^k = \left(1 + \Delta t \kappa(\xi)\lambda^k\right) w_n^k$$

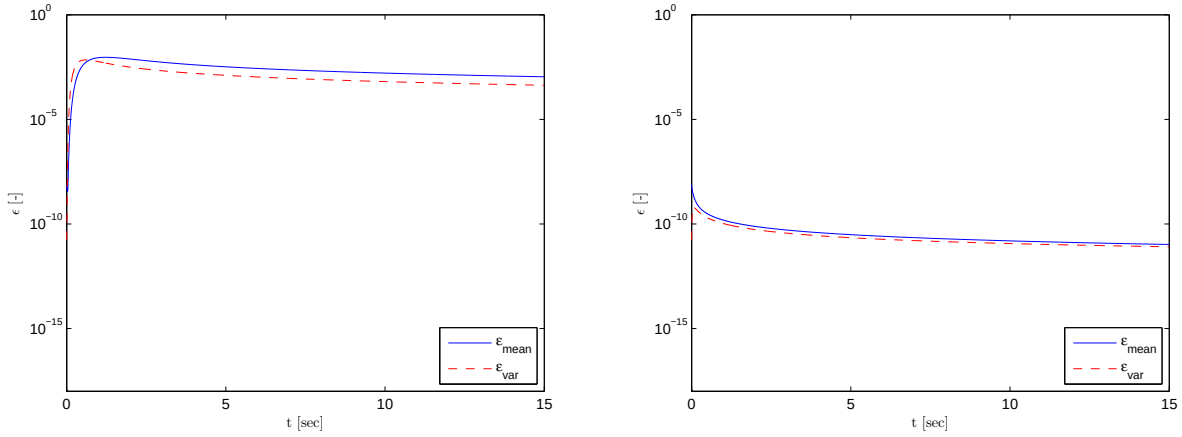
The results are shown in Figures 5.7 and 5.8. The error shown is the L^1 norm. This can be seen as the real accuracy of the scheme. It becomes clear that the new formulation really is able to deliver very accurate results. The accuracy can be increased even further by using more quadrature points.



(a) classic gPC, $P = 3$

(b) the new formulation, $2P + 1 = 3$

Figure 5.7: Variance at $x = \frac{4}{15}$ for the fully discrete system, 15 elements in spatial domain, $\Delta t = 0.001$, Q=60



(a) classic gPC, $P = 3$

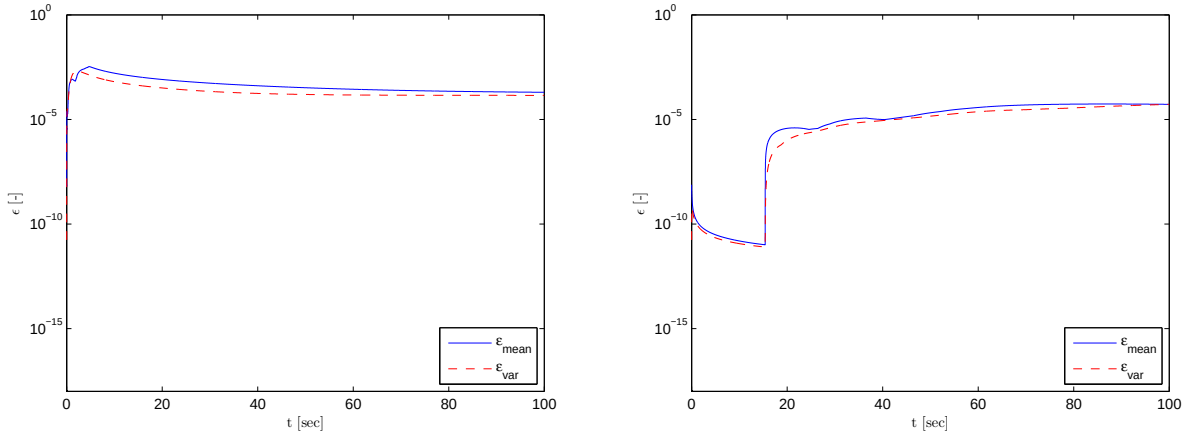
(b) the new formulation, $2P + 1 = 3$

Figure 5.8: Errors in L^1 sense for the fully discrete system, 15 elements in spatial domain, $\Delta t = 0.001$, Q=60

5.8 Influence of threshold value ϵ

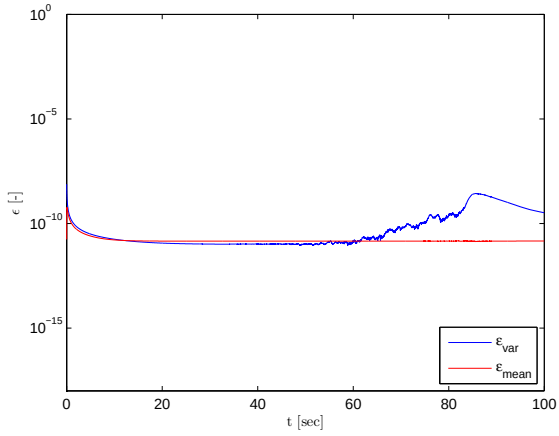
The value of ϵ, i.e. the threshold value for the norm of the orthogonalized polynomials as described in Section 5.3, was found to have a substantial influence on the accuracy of the results for long time integration.

In Figure 5.9 the L^1 errors are shown, when varying the value of ϵ. It is clear from this figure that the choice of ϵ has a significant effect on the outcome. When polynomials are discarded quickly, e.g. $\epsilon = 10^{-5}$, there is a visible jump in the error. When lowering the value of ϵ, this jump is postponed and lowered, until it is so low that round-of errors in the orthogonalization become the predominant source of error.

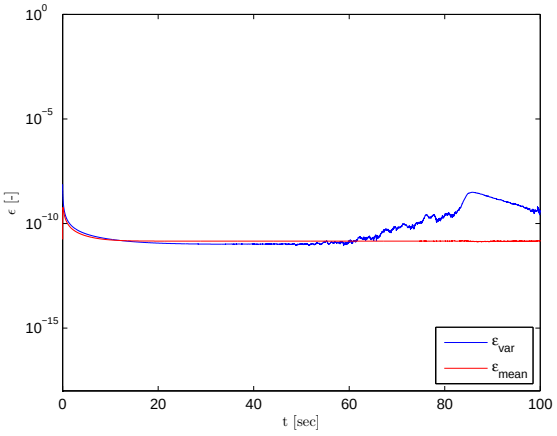


(a) $\epsilon = 10^{-5}$

(b) $\epsilon = 10^{-10}$



(c) $\epsilon = 10^{-20}$



(d) $\epsilon = 10^{-30}$

Figure 5.9: Error results for varying values of ϵ, $2P + 1 = 3$, 15 elements in spatial domain, $\Delta t = 0.001$, Q=60

Chapter 6

From TDgPC to PCM

Although the results of Chapter 5 were encouraging, a realization led to a different perspective on the new TDgPC algorithm. In this chapter, a brief overview of the PCM procedure will be given, after which the growth equation will be used to take a better look at the new-found TDgPC algorithm. Finally, a major disadvantage of PCM will be explained as a prelude to the next chapters.

6.1 The Probabilistic Collocation Method

The implementation of a basic PCM algorithm is straightforward: calculate the deterministic solution at several collocation points in stochastic space and use a polynomial fitting to determine the statistical moments from this data. Or to put in more intuitively: perform a Monte Carlo simulation in a very clever way by carefully choosing the 'random' input.

The statistical moments are again found using some form of numerical quadrature, using the PDF of the solution as a weighting function. The exact location of these collocation points and the value of the quadrature weights are dependent on the method one uses, and can be modified accordingly, as will be shown in Chapter 8.

As stated before, this method always results in a decoupled system of equations and is non-intrusive, i.e. one does not need to modify the flow equations. Rather, one can obtain the statistical moments by using this method as a post-processing tool.

6.2 The growth equation revisited

Consider the growth equation, i.e. Equation 4.1, and apply a forward Euler discretization

$$\begin{aligned} u^{n+1}(\xi) &= u^n(\xi) - \Delta t k(\xi) u^n(\xi) \\ &= \left(1 - \frac{\Delta t}{2} - \frac{\Delta t \xi}{2}\right) u^n(\xi) \end{aligned} \tag{6.1}$$

The term $u^n(\xi)$ on the right hand side of the above equation can be regarded in two ways.

First of all, one can again write this term as a function of the previous time step, so

$$u^n(\xi) = \left(1 - \frac{\Delta t}{2} - \frac{\Delta t \xi}{2}\right) u^{n-1}(\xi)$$

This step can be repeated, until

$$\begin{aligned} u^{n+1}(\xi) &= \left(1 - \frac{\Delta t}{2} - \frac{\Delta t \xi}{2}\right)^{n+1} u^0(\xi) \\ &= \left(1 - \frac{\Delta t}{2} - \frac{\Delta t \xi}{2}\right)^{n+1} \end{aligned} \tag{6.2}$$

Remember that this expression is exact in the stochastic sense, even though a time integration error is being made. So the exact solution of the growth equation at time step n can be represented as a polynomial in ξ of order n. This again shows why in normal gPC procedures, the original orthogonal basis will over time become incapable of capturing the exact solution in stochastic space.

Secondly, because of the definition of ζ,

$$u^n(\xi) = \zeta(\xi)$$

Therefore, expression 6.1 can also be written as

$$u^{n+1}(\xi) = \left(1 - \frac{\Delta t}{2} - \frac{\Delta t \xi}{2}\right) \zeta(\xi)$$

From this, it becomes clear that the exact solution can be written as a linear combination of ξ and ζ. But this is exactly what is done in the TDgPC procedure as described in Section 5.3! By updating $\zeta(\xi)$ every time step and expanding the solution as a linear combination of only four polynomials, i.e. 1, ξ, ζ, $\zeta\xi$, the exact discrete solution is fully represented! It is therefore not necessary to go to higher order polynomials.

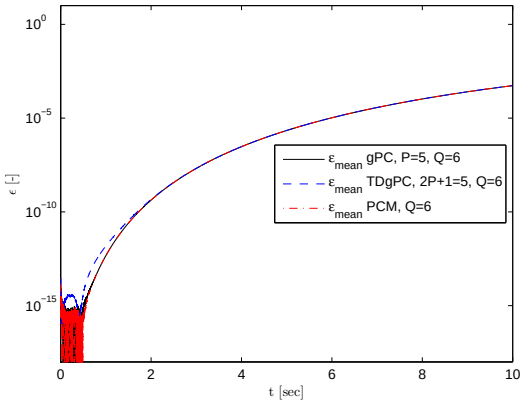
However, the above line of reasoning is written in a stochastically continuous form. In reality, these equations are discretized and only evaluated on certain quadrature points. With Gauss-Lobatto-Legendre quadrature integration, the values at these quadrature points are used to calculate the statistical moments, as explained in Section 3.2. Although this method of integrating is exact for integrands up to order $2Q - 1$ using only Q quadrature points, Equation 6.2 showed that after n time steps, a polynomial of order n needs to be integrated to calculate the first statistical moment. So e.g. after 99 time steps, already 50 quadrature points are needed to find the exact mean. This of course becomes even worse when considering the variance of the solution, since there the is integrand is the solution squared.

The result is then that not the polynomial order of the expansion is the limiting parameter, but the number of quadrature points. In fact, in every quadrature point, the exact solution is found (apart from the integration error) and a stochastic error is only made when performing the integrations needed for the calculation of the statistical moments. This is where the method seizes to be a gPC method and starts to be a PCM procedure. Because this is exactly what the Probabilistic Collocation Method represents: find the exact solution in several quadrature points (although in PCM they are dubbed collocation points) and calculate the statistical moments using only these points in stochastic space. If one then requires a higher accuracy, one should increase the number of collocation points.

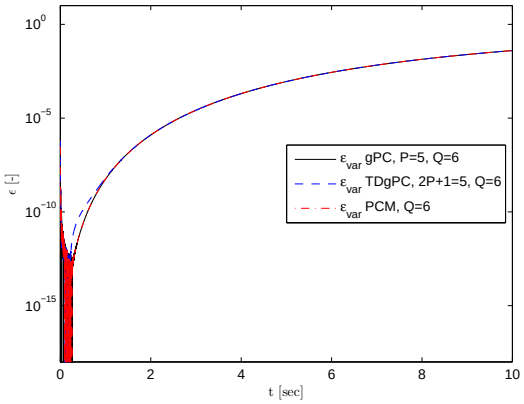
To verify this conclusion, see Figure 6.1. In this Figure, the results of three different methods are shown: gPC, TDgPC in the new formulation and PCM. The results for all three simulations are very similar. As soon as errors due to the stochastic drift become larger than the round-off and transformation errors, the results are even identical, as expected. Note that for the TDgPC simulation, a higher order polynomial than the initially proposed $2P + 1 = 3$ was used, since a RK4 integration method was performed.

6.3 Shortcoming of the PCM

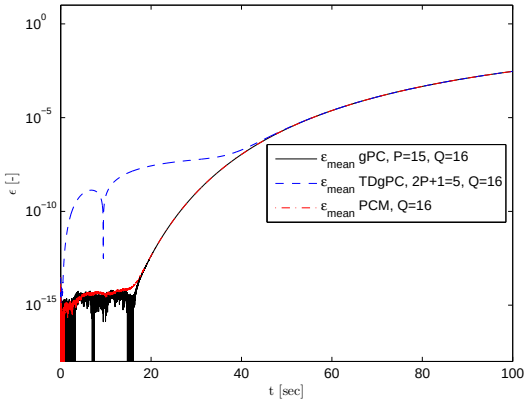
One of the major disadvantages of the PCM is very similar to that of the TDgPC procedure: it experiences stochastic drift. In the case of PCM, this means that more collocations points are needed for accurate results over longer time integrations.



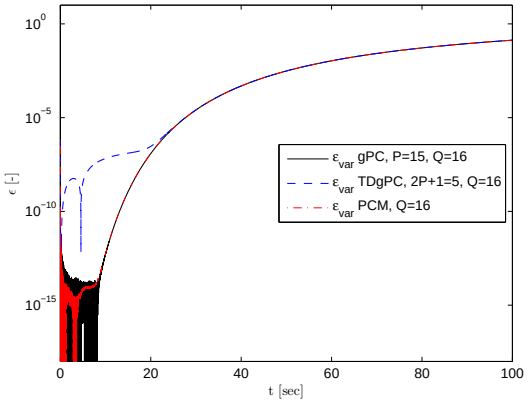
(a) Mean after 10 seconds, integration order 5



(b) Variance after 10 seconds, integration order 5



(c) Mean after 100 seconds, integration order 15



(d) Variance after 100 seconds, integration order 15

Figure 6.1: Errors in mean and variance for the gPC, TDgPC and PCM simulation using equivalent polynomial order and number of collocation points, $\Delta t = 0.001$ using a RK4 integration scheme.

This can be clearly demonstrated using the advection equation

$$\frac{\partial u(x,t,\xi)}{\partial t} = a(\xi)\frac{\partial u(x,t,\xi)}{\partial x}, \qquad u(x,0) = \cos(2\pi x), \qquad 0 \leq x \leq 1, \qquad t \geq 0 \tag{6.3}$$

with periodic boundary conditions ($u(x=0) = u(x=1)$) and

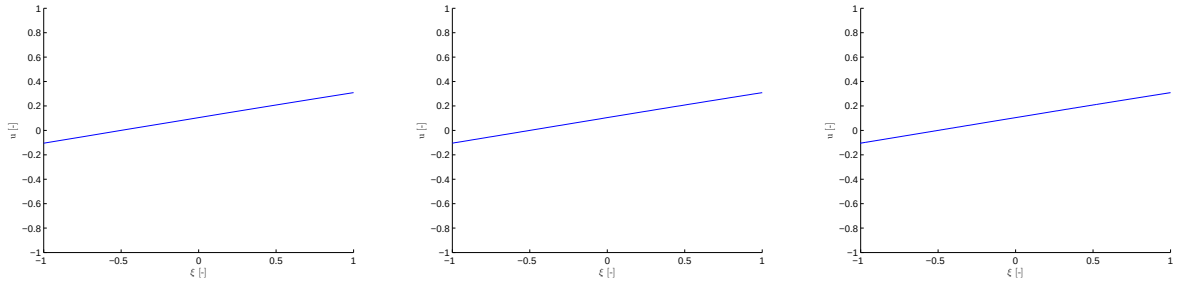
$$a(\xi) = \frac{1}{2}(1+\xi), \qquad -1 \leq \xi \leq 1$$

This equation describes a wave, moving at an uncertain, positive velocity. The analytical solution is known to be

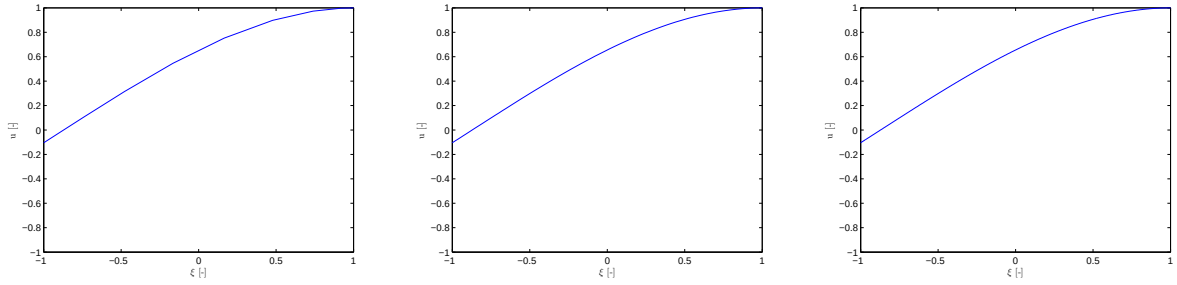
$$u(x,t,\xi) = \cos(2\pi(x - a(\xi)t))$$

This equation was solved using the PCM procedure and a leapfrogging integration scheme. The solutions in the stochastic domain are shown in Figure 6.2 using a varying number of collocations points. It becomes clear that as time progresses, more and more collocation points are needed to avoid aliasing and thus unacceptable errors in the calculation of the variance and mean.

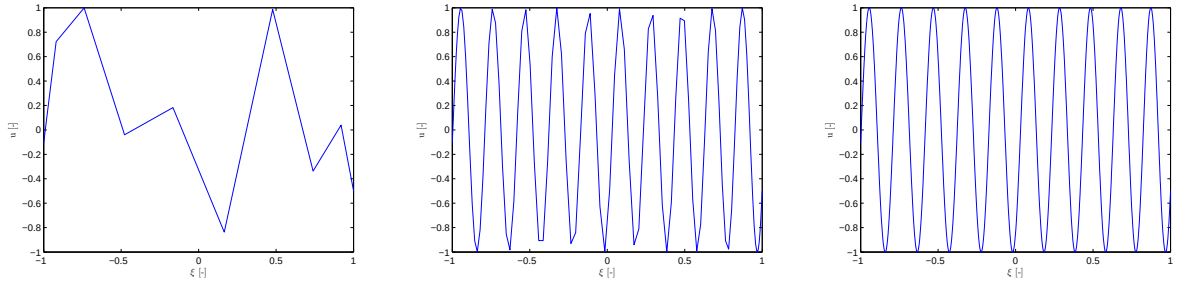
In the next chapters, time-dependent methods will be used to ameliorate the effects of this phenomenon.



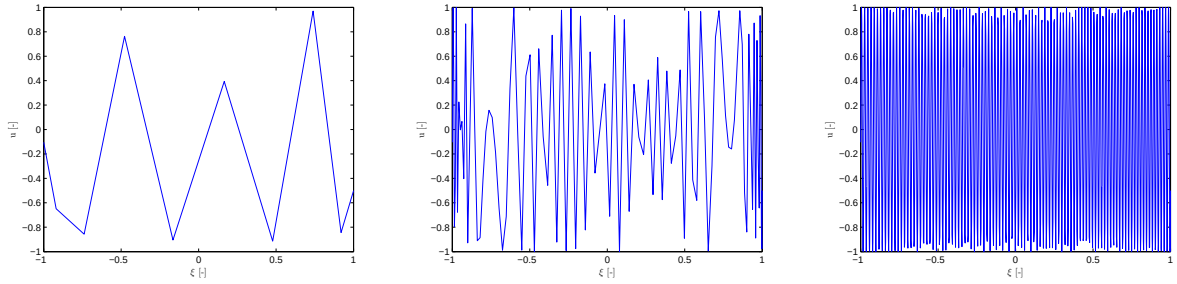
(a) $Q = 10$, $t = \Delta t = 0.0667$ s (b) $Q = 100$, $t = \Delta t = 0.0667$ s (c) $Q = 1000$, $t = \Delta t = 0.0667$ s



(d) $Q = 10$, $t = 5\Delta t = 0.33$ s (e) $Q = 100$, $t = 5\Delta t = 0.33$ s (f) $Q = 1000$, $t = 5\Delta t = 0.33$ s



(g) $Q = 10$, $t = 150\Delta t = 10$ s (h) $Q = 100$, $t = 150\Delta t = 10$ s (i) $Q = 1000$, $t = 150\Delta t = 10$ s



(j) $Q = 10$, $t = 1500\Delta t = 100$ s (k) $Q = 100$, $t = 1500\Delta t = 100$ s (l) $Q = 1000$, $t = 1500\Delta t = 100$ s

Figure 6.2: The solution as a function of the stochastic variable at $x = \frac{4}{15}$, using $Q = 10, 100, 1000$ at various points in time

Chapter 7

P-refinement applied to the PCM

One way of countering the stochastic drift during a PCM simulation is by applying a p-refinement, i.e. adding collocation points during the simulation.

7.1 Interpolation

Although the addition of collocation points during the simulation is straightforward enough, the interpolation routines require some caution. If the interpolation error becomes too great, it will negate the effect of adding more points. The Lagrangian interpolator was chosen, since it obviously is well suited for interpolation with polynomials.

7.2 Refinement criterium

An important parameter in this procedure is the refinement criterium. There are several possible criteria, each with specific advantages and disadvantages:

- A straightforward approach is to refine when the relative error of the mean or variance reaches a certain threshold value. This can also be used to stop the refinements, when the interpolation errors become greater than the gain in accuracy. The drawback of this approach is that one needs a priori knowledge of the exact solution. In the case of the growth equation this is trivial, but when considering a CFD simulation of e.g. a full airplane configuration, this criterium is not all that clear.
- The criterium as used by Vos for the TDgPC transformations was to set a threshold value for the non-linear coefficients of the polynomial expansion. Although this was an elegant solution for the TDgPC algorithm, it does not make sense in the PCM framework. It would plainly defy the whole point of the PCM to project the solution back onto a set of orthogonal polynomials to find these coefficients. The strength of the PCM is exactly to perform the calculations without this Galerkin projection.
- An alternative is to refine at fixed times. This bypasses the above mentioned disadvantages of threshold criteria, but it has quite a severe one itself. For this criterium to work, one needs some degree of knowledge of the progress of the simulation. For example, one should have an idea of the rate at which the solution will change over time. This could be achieved by first running a lower order simulation without refining the random variable and then performing the full-scale simulation.

Since the growth equation is straightforward and well understood, the refinement at fixed times was chosen. However, for more interesting simulations, i.e. those that fluctuate heavily in time, a more appropriate criterium should be constructed.

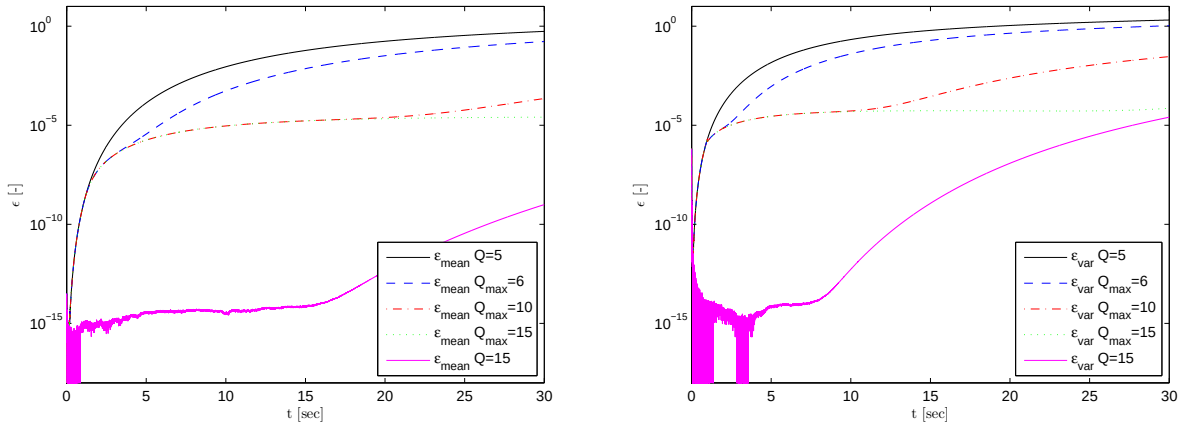
7.3 Manner of refinement

The goal of p-refinement is to add collocation points during the simulation. The amount of points added, can of course be varied. One could for example chose to multiply the current number of points by a factor, or to simply add a fixed number of points. This as opposed to refining less often, but adding more points.

Testing showed that the most effective results were achieved when adding just one point, but performing this refinement regularly.

7.4 Results

The effect of the p-refinement can be seen in Figure 7.1. These simulations were all started with five collocation points. Every second, one collocation points was added, up to a maximum of collocation points, Q_{max}. This maximum value was varied. So for the simulation with $Q_{max} = 15$, refinements were performed at $t = 1, 2, \ldots, 10$. As a reference, the results for a simulation with $Q = 15$ without refinement are shown as well.



(a) Mean

(b) Variance

Figure 7.1: Errors in mean and variance using p-refinement on the PCM procedure

Although it is obvious that the simulation run with $Q_{max} = 15$ cannot achieve the same accuracy as a simulation initiated with 15 collocation points, the advantages of p-refinement are clear. Adding only one collocation point every second has an significant stabilizing effect on the calculations. Interpolation errors are well within acceptable limits, i.e. they are negligible compared to the error produced by the PCM procedure itself.

For large simulations where computational cost becomes an issue, the benefits of p-refinement are most clear. If one can add points only when the accuracy of the simulation is compromised, there is no need for expensive calculations in points that are not (yet) necessary.

Chapter 8

Time-Dependent Probabilistic Collocation Method

The basic idea behind TDgPC is that the variable used to express the problem, should change and adapt in time. This is done to prevent the aforementioned stochastic drift. This principle can also be applied to the PCM procedure. This chapter will explain the implementation of this Time-Dependent Probabilistic Collocation Method (TDPCM) and show its application to the growth equation.

8.1 Working principle of TDPCM

The main cause for error in normal PCM calculations, is that the points in stochastic space cannot capture the behaviour of the solution. For example, for the growth equation, the PDF of the solution starts out uniformly, but gradually centers around the origin of the stochastic domain. This can also be seen from Figures 4.2 and 4.3. So at the start of the simulation, only two points would suffice to describe the exact behaviour. However, when integrating further in time, one needs more and more points to capture the steep slope of the solution around the origin. Moreover, these points would ideally be clustered at this steep slope. Having more points around the 'tail' of the solution is unnecessary, since this virtually horizontal line segment can be described using very few points.

So even though one can start out with many (computationally expensive) collocation points, at some point in the integration, these points will not be able to capture the solution, resulting in increasing errors.

Applying the principles of time-dependence of the stochastic variable on the PCM, one can ameliorate this problem. Not only will the transformation of the stochastic variable have a steadying effect on the order of the error, but the relocating of the collocation points will enable us to use said points more effectively.

8.2 The Probabilistic Collocation Method applied to the growth equation

Using again the growth equation as a test bed for our new method, we start the simulation using the standard Gauss-Lobatto-Legendre (GLL) collocation points and weights for the PCM procedure. The variable $u(\xi, t)$ is discretized on these N points. The Ordinary Differential Equation (ODE) is then expressed as a decoupled system of equations

$$\frac{\mathrm{d}u_i(t)}{\mathrm{d}t} + k_i u_i(t) = 0, \qquad \forall i = 1 \dots N$$

The variance and mean are calculated as

$$\begin{aligned}\mu(t) &= \int_{-1}^{1} u(\xi,t)f(\xi)\mathrm{d}\xi = \sum_{i=1}^{N} u_i(t)w_i \\ \sigma(t) &= \int_{-1}^{1} \left((u\xi,t)-\mu(t)\right)^2 f(\xi)\mathrm{d}\xi = \sum_{i=1}^{N} u_i^2(t)w_i - \mu^2(t)\end{aligned}$$

8.3 The TDPCM algorithm

At some point during the time integration, say at $t = t^*$, we wish to reevaluate the collocation points and the random variable. In order to do this, we first introduce the transformed random variable ζ, as was done in the TDgPC as well

$$\zeta = u(\xi, t^*)$$

The difference with the TDgPC method is that $u(\xi, t)$ does not have to be constructed out of the orthogonal polynomial basis. Instead, a Lagrange polynomial is constructed through the collocation points. Let us call this polynomial T

$$\zeta = T(\xi)$$

The PDF of this new variable is then approximately given by [16]

$$f_\zeta(\zeta) = \sum_n \frac{f_\xi(\xi_n)}{\left|\frac{\mathrm{d}T(\xi)}{\mathrm{d}\xi}\Big|_{\xi=\xi_n}\right|}$$

This new PDF is used as a weighting function to find the optimal orthogonal basis. To find these polynomials, the Stieltjes procedure is used [17]. This is a stable method to find the coefficients of polynomials for arbitrary distribution functions. These orthogonal coefficients satisfy the three-term recurrence relation

$$\phi_{i+1}(\zeta) = (\zeta - \alpha_i)\phi_i - \beta_i\phi_{i-1}, \qquad i = 2, \dots N \tag{8.1}$$

with $\phi_0 = 0$, $\phi_1 = 1$. The recurrence coefficients α and β are then found as

$$\begin{aligned}\alpha_i &= \frac{\langle\zeta\phi_i, \phi_i\rangle}{\langle\phi_i, \phi_i\rangle}, \qquad i = 1, \dots N \\ \beta_i &= \frac{\langle\phi_i, \phi_i\rangle}{\langle\phi_{i-1}, \phi_{i-1}\rangle}, \qquad i = 1, \dots N\end{aligned}$$

where the $\langle, \rangle$ still denotes the ensemble average, but now with respect to the new PDF $f_\zeta(\zeta)$. For a normalized PDF, $\beta_1 = 1$.

To find the desired Gauss quadrature points, the roots of the derivative of polynomial ϕ_{N-1} are calculated. This gives $N-2$ collocation points. The minimum and maximum value of the new random variable are taken as the remaining two points.

Next, the associated weights are calculated. To ensure continuity of the statistical moments over the

transformation, the following requirements are imposed

$$\begin{aligned}
\sum_{i=0}^{N} \tilde{w}_i &= 1 \\
\sum_{i=0}^{N} \zeta_i \tilde{w}_i &= \sum_{i=0}^{N} u_i w_i \\
\sum_{i=0}^{N} \zeta_i^2 \tilde{w}_i &= \sum_{i=0}^{N} u_i^2 w_i \\
&\vdots \\
\sum_{i=0}^{N} \zeta_i^N \tilde{w}_i &= \sum_{i=0}^{N} u_i^N w_i
\end{aligned} \tag{8.2}$$

Here, u_i is the solution in the 'old' collocation points, ζ_i the solution in the 'new' collocation points and $\tilde{w}$ denotes the weights after the transformation. This results in a system of $N+1$ equations with $N+1$ unknowns, yielding the new weights.

Finally, the new 'initial' conditions for $\tilde{u}_i(t^*)$ need to be imposed. Note that $\tilde{u}_i(t^*) \neq u_i(t^*)$, since $\tilde{u}_i(t^*)$ is of course evaluated at the new collocation points. Regarding the definition of the new variable this is trivially

$$\tilde{u}_i(t^*) = \zeta_i$$

The algorithm can then be summarized as:

- *apply the PCM on a stochastic ODE*
- *integrate in time*
- *at specified time t^*:*
- *construct new variable ζ*
- *calculate new PDF as a function of ζ*
- *construct orthogonal polynomial basis through Stieltjes procedure*
- *use derivative of polynomial of order $N-1$ to find $N-2$ new collocation points*
- *add boundaries as two remaining points*
- *use continuity of statistical moments to find weights at new collocation points*
- *set new initial conditions*
- *interpolate growth coefficient and PDF at new points*
- *calculate mean and variance*
- *postprocessing*

8.4 Considerations on the algorithm

Although this algorithm is in principle straightforward, some considerations are in place.

8.4.1 Interpolations and integrals

The TDPCM as described above clearly depends on numerous interpolations and numerical integrations. As a rule, all interpolations and curve fittings were done using a Lagrangian interpolation. Although in previous versions of the program cubic spline and least square fittings were used, Lagrangian interpolation outperformed both.

Also, even though Van der Steen and Vos used a Gram-Schmidt orthogonalization, for this method the Stieltjes procedure was chosen. It proved to be more accurate and numerically stable.

But even with these precautions, interpolation errors are an important source of errors. This will be further discussed in Section 8.5.

8.4.2 Calculation of the collocation points

Another method of finding the collocation points from the set of orthogonal polynomial is the Golub-Welsch algorithm [18]. This method relies on finding the eigenvalues of the matrix

$$J = \begin{bmatrix} \alpha_1 & \sqrt{\beta_1} & & & & \\ \sqrt{\beta_2} & \alpha_2 & \sqrt{\beta_3} & & \emptyset & \\ & \sqrt{\beta_3} & \alpha_3 & \sqrt{\beta_4} & & \\ & & \ddots & \ddots & \ddots & \\ & \emptyset & & \sqrt{\beta_{N-1}} & \alpha_{N-1} & \sqrt{\beta_N} \\ & & & & \sqrt{\beta_N} & \alpha_N \end{bmatrix}$$

The α and β in this system are the recurrence coefficients of Equation 8.1. This is equivalent to finding the roots of polynomial ϕ_N. However, this method has the disadvantage of 'shrinking' the domain, i.e. it always finds the new points *within* the domain, see Figure 8.1. When using few points, e.g. five collocation points, this resulted in severe loss of accuracy when transforming a second time. The end points are necessary to fully describe the PDF.

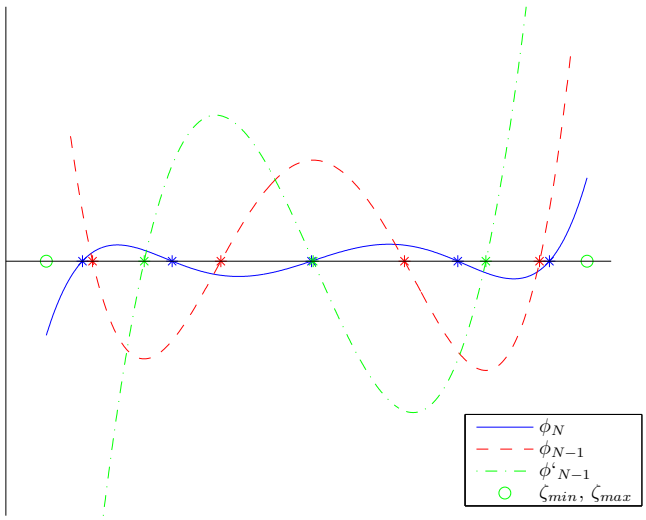


Figure 8.1: Position of different roots in domain of ζ

8.4.3 Rootfinders

Furthermore, rootfinders are extensively used throughout the program. The default Newton rootfinder was chosen for its convergence rate and accuracy. However, this routine proved to be inadequate for simulations using more than ten collocation points. The algorithm was not able to find all roots in the considered interval, which led to unacceptable errors. Therefore, no results will be presented using polynomials of such order.

8.4.4 Transformation criterium

One of the issues using this procedure, is to determine the time at which to transform the random variable and recalculate the collocation points. The possible criteria are the same as discussed in Section 7.2 for the p-refinement procedure.

A specific case of transforming at fixed times is to transform at every time step. Although this is a very straightforward implementation, it is also the computationally most expensive one. It is therefore not considered to be practically feasible, especially for more intricate problems.

Considering the characteristics of the problem at hand, the simulations for this problem were also run with transformations at fixed times. However, more research is desirable to find a more general and elegant transformation criterium.

8.5 Results

The benefits of this new method become apparent when considering Figure 8.2. Using the TDPCM scheme results in a gain of about three orders of magnitude for the relative errors at t_{end}. Apparently, it is easier to follow the exact solution in stochastic space by using the new collocation points. As can be seen in Figure 4.3, a clustering of collocation points near $\zeta = 0$ would be desirable as time proceeds. This is where the solution will be most difficult to capture. Table 8.1 shows this is indeed the case. The newfound collocation points are indeed 'moving' in the right direction.

Although an error is introduced when transforming, this is very quickly negated by the increased capability to capture the behaviour of the exact solution. This error is a combination of factors. First of all, round-off errors play a role. For example, when solving the system of equations 8.2, the program has difficulty maintaining an accuracy of 10^{-14}, most notably when performing matrix inversions. Also, the aforementioned rootfinder introduces an round-off error.

But most notably, the interpolation routines are responsible for these errors. For example, when interpolation k at the new collocation points, an error is made. Since $k(\zeta)$ is known analytically, one can investigate the size of this error. From the analytical solution for the growth equation we know that

$$u(\xi, t) = \mathrm{e}^{-k(\xi)t}$$

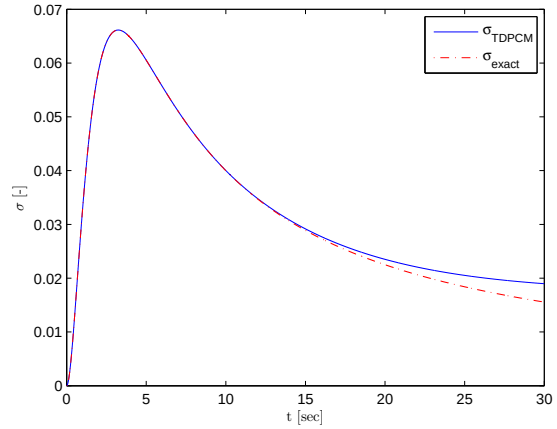
and at the time of transformation $u(\xi, t^*) = \zeta$, so

$$\zeta = \mathrm{e}^{-k(\xi)t} \quad \Rightarrow \quad k = -\frac{1}{t}\ln\zeta$$

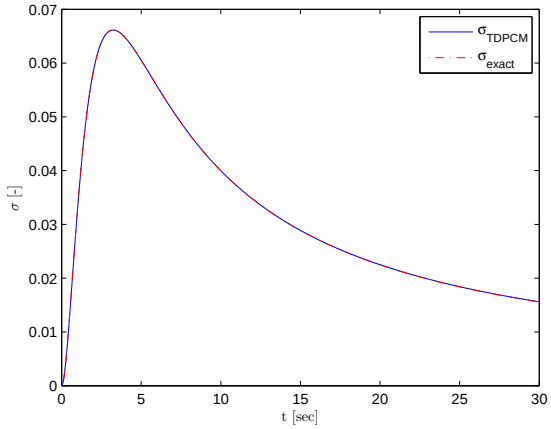
When the analytical values for k are inserted into the routine, rather than the interpolated values, one sees significantly smaller 'jumps' in the relative errors at the time of transformation, see Figure 8.3. The same holds for the interpolation of the PDF.

Table 8.1: Position of collocation points as percentage of domain, $N = 7$

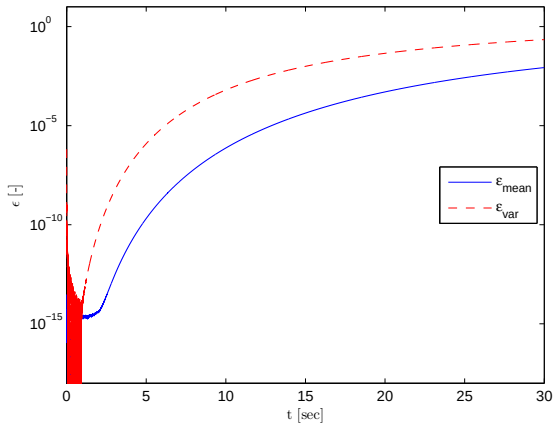
collocation point	original position	position after 1st transformation	position after 2nd transformation
1	0.0	0.0	0.0
2	8.5	7.9	7.2
3	26.6	25.2	23.9
4	50.0	48.4	47.0
5	73.4	72.3	71.4
6	91.5	91.1	90.8
7	100.0	100.0	100.0



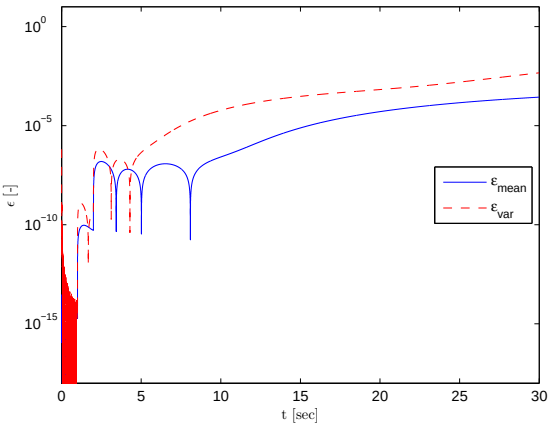
(a) Variance using PCM



(b) Variance using TDPCM, transforming at $t = 1, 2$

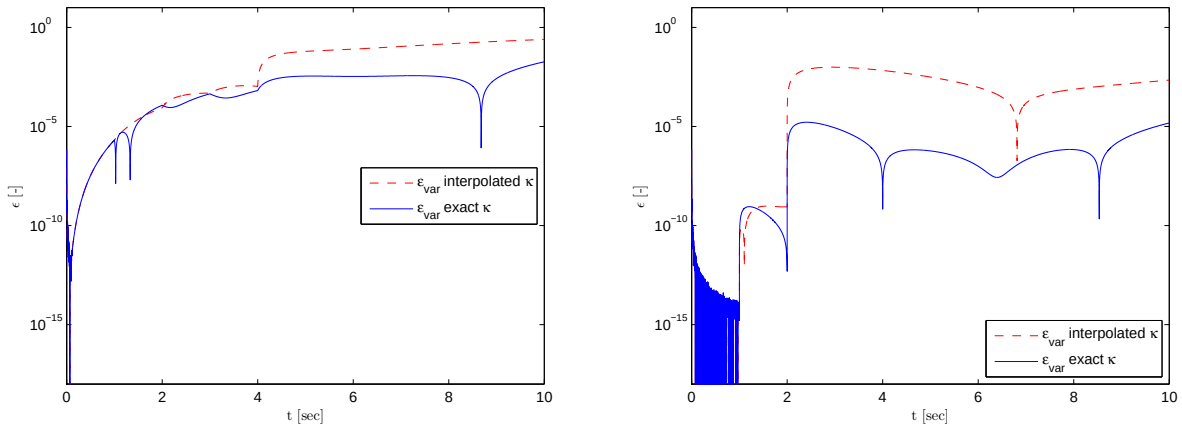


(c) Errors using PCM



(d) Errors using TDPCM, transforming at $t = 1, 2$

Figure 8.2: Errors and variance for PCM and TDPCM, $N = 8$



(a) $N = 5$, transforming at $t = 1, 2, 3, 4$

(b) $N = 9$, transforming at $t = 1, 2$

Figure 8.3: Difference in relative variance error when using exact or interpolated k

8.5.1 Influence of the number of collocation points

Keeping the same criterium for the transformation, we can see the influence of the number of the collocation points (and therefore the polynomial order), see Figure 8.4. Increasing the polynomial order has several effects. Initially, it increases accuracy. This is as expected, since the use of more collocation points counters the effect of stochastic drift. However, at $N = 9$ the interpolation error introduced at the transformation starts to undo the benefit of using better collocation points. Even though the interpolation error of e.g. k will decrease when using more points, the influence of the error is amplified even further, see Figure 8.3. Also, the rootfinder and matrix inversion deliver less accurate results when using more points.

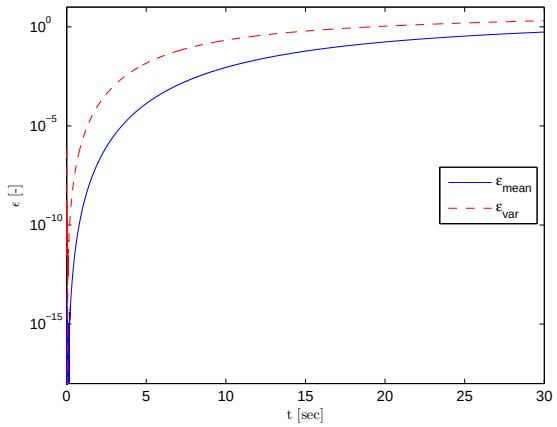
8.5.2 Influence of the time of transformation

The influence of the time of transformation was investigated as well, see Figure 8.5. If one transforms too early, the PDF is not clustered around $\zeta = 0$ yet, and the new collocation points will not differ much from the original ones. The interpolation error introduced when transforming will even be higher than the benefit of the transformation. Transform too late in the simulation and the original points will have great difficulty capturing the PDF and the benefits from calculating new points will already be negated by the errors from the PCM part of the simulation.

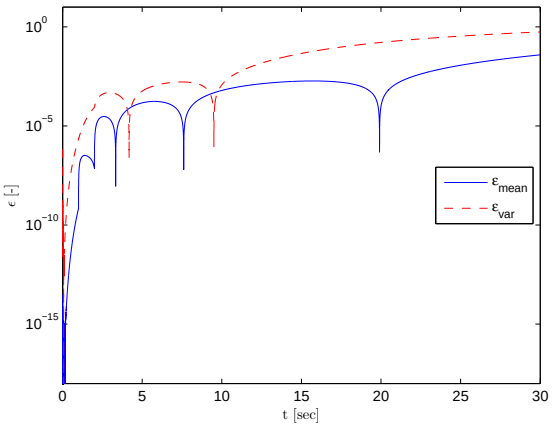
Furthermore, if the first transformation was performed after $t = 20$, the polynomial fitting of the PDF proved to be inadequate and the calculation of the new collocation points went awry. This is the same effect as can be seen in Figure 4.2.

8.5.3 Influence of the number of transformations

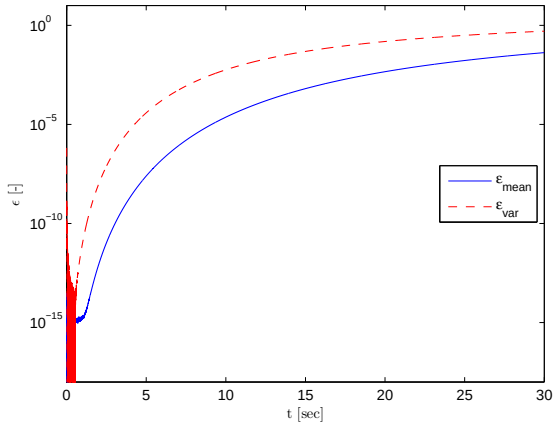
From Figure 8.6 it becomes clear that it is most beneficial to transform several times early on in the simulation. If the interval between two transformations is too large, a similar effect as described in the Section 8.5.2 occurs, i.e. the polynomial fitting of the PDF generates less than optimal values for the new collocation points.



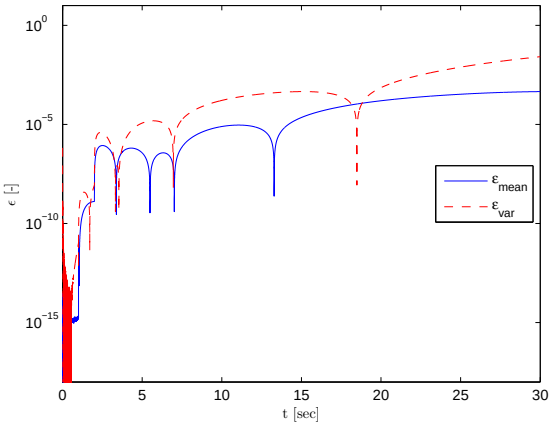
(a) Variance using PCM, $N = 5$



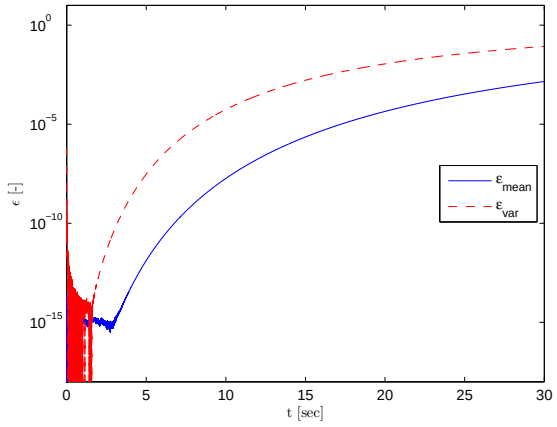
(b) Variance using TDPCM, transforming at $t = 1, 2$, $N = 5$



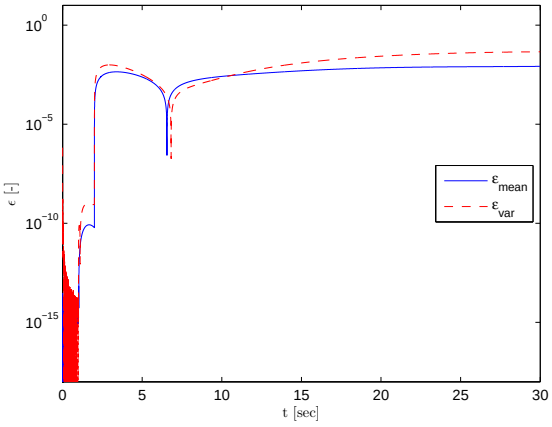
(c) Variance using PCM, $N = 7$



(d) Variance using TDPCM, transforming at $t = 1, 2$, $N = 7$



(e) Variance using PCM, $N = 9$



(f) Variance using TDPCM, transforming at $t = 1, 2$, $N = 9$

Figure 8.4: Errors and variance for PCM and TDPCM, using varying values of N

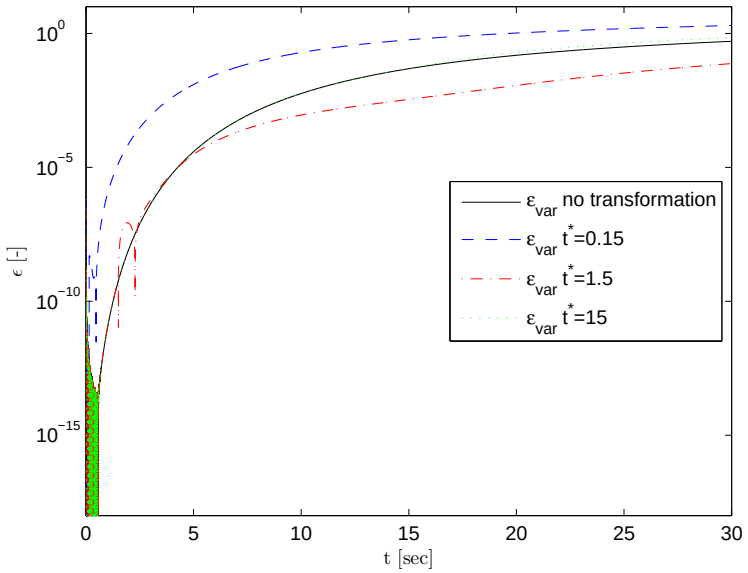


Figure 8.5: Relative variance errors for TDPCM, using varying values of t^*, $N = 7$

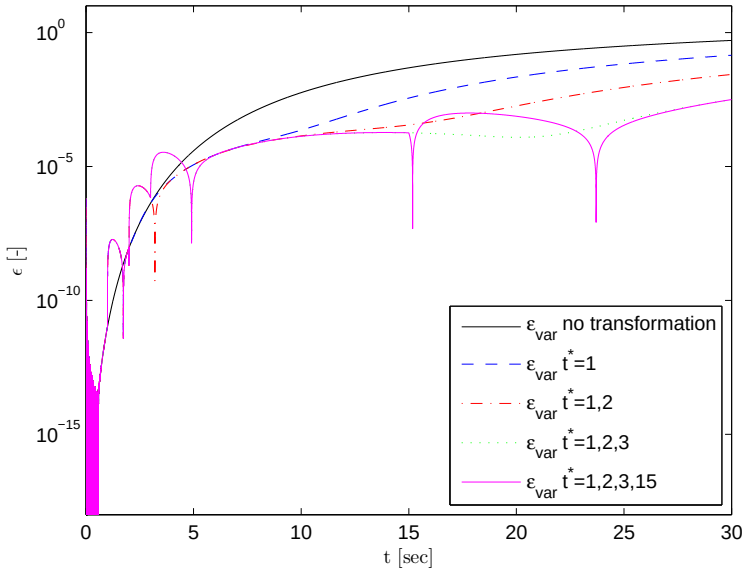


Figure 8.6: Relative variance errors for TDPCM, using varying number of transformations, $N = 7$

Chapter 9

Conclusions and recommendations

This chapter will recapitulate the results of this thesis and give some recommendations for further research.

9.1 Conclusions

During this graduation project, several methods were investigated to enhance to performance of stochastic methods in long-term integration. The starting point of this research was the TDgPC method as formulated by Vos and van der Steen.

An alternative formulation for the expansion of the solution variable was investigated and applied to the growth equation, see Section 4.4. In first instance, the results were very promising, but the method proved to be too case sensitive to be generally applicable.

A generalization of this procedure was found and tested on the PDF known as the heat equation, as described in Section 5.3. Again, the gain in accuracy of the simulation was considerable. A convergence analysis was performed to show that the error introduced by the stochastic computations was negligible when compared to the space and time discretization errors.

When analyzed more thoroughly, the new TDgPC formulation proved to be equivalent to the standard Probabilistic Collocation Method, see Section 6.2. Since the PCM has its own shortcomings in long-term integrations, time-dependent methods were sought to soften the effects of stochastic drift.

The effects of p-refinement during a simulation, i.e. adding collocation points, were investigated in Chapter **??**. The interpolation error introduced at the time of refinement proved to be insignificant when compared to the overall relative error. Thus, a simulation could be initiated using a minimum of collocation points. When the accuracy of the simulation was compromised, more points were adding, resulting in a stabilization of the relative errors of the mean and variance. Therefore, p-refinement has the potential to be a valuable tool for PCM procedures, especially when computational cost becomes an issue.

Finally, the principles of TDgPC, i.e. to reevaluate the stochastic variable over time, were applied to the PCM in Chapter 8. The accuracy of the mean and variance relies heavily on the placement of the collocation points in stochastic space. Therefore, these positions were recalculated several times during the simulation. It was shown that the algorithm indeed moves the collocation points towards the region in stochastic space where they are most effective. Although an error was introduced when transforming, this was negated by the beneficial effects of a better placement of the collocation points.

9.2 Recommendations

A substantial limitation in the application of time-dependent methods on the PCM is the refinement/transformation criterium. For the TDgPC procedure, the values of the non-linear coefficients in the polynomial expansion were a good measure and could serve as a threshold criterium. In PCM simulation, this does not make sense. Other criteria rely on prior knowledge of either the exact solution or the progress of the simulation.

For full-scale simulations of proper flow equations this poses a serious issue. Although this problem could be circumvented by first performing a low-order simulation, this is not an elegant solution. Therefore, more research on this criterium would be appropriate.

The rootfinder routine in the TDPCM also turned out to be a limiting factor in the applicability of the method. Either a different algorithm should be used instead, or a linear transformation should be performed to 'blow up' the domain, making the algorithm more robust.

A logical next step in this research would be to apply the TDPCM to more intricate problems. The extension to simple partial differential equations will not be very difficult. However, the influence of more than one stochastic variable will be quite interesting to investigate. As stated before, the Probabilistic Collocation Method suffers from the 'curse of dimensionality.' So unfavorable side-effects are possible in these cases. On the other hand, this is also an opportunity for p-refinement to really prove its worth.

A very interesting development would be to combine the TDPCM with a multi-element scheme as used by Foo, Wan and Karniadakis [13]. The results achieved with ME-PCM are formidable, especially when considering higher order simulations. Also, to the knowledge of the author, p-refinement has not been tried in combination with ME-PCM. This too, could be a powerful combination.

9.2.1 The PCM paradox

Besides all this, the PCM in itself is quite paradoxical. The goal of these algorithms is to find a solution in several points in stochastic space as accurately as possible. When that is done, the statistical moments are calculated by performing an integration over this domain. In most cases, this is exactly what is required by an engineer: a mean or variance at a certain point in time. In general, the behaviour of the solution in stochastic space is not considered very important, as long as the mean and variance are known. They are the 'design parameters.'

First of all, calculating the probability in a point is awkward, mathematically speaking. Theoretically, the probability of an event in a point in stochastic space is zero. But more importantly, why would one want to calculate values at discrete points, if the main interest lies in integrated values?

It could for example be very interesting to abandon the concept of calculating the solution in certain collocation points, and use the mean over an interval as a new variable. So instead of solving the system of equations

$$\frac{\mathrm{d}u_i(\xi,t)}{\mathrm{d}t} + k_i(\xi)u_i(\xi,t) = 0, \qquad \forall i = 1 \ldots N$$

One could perform a variable transformation

$$\overline{u}_i = \int_{\xi_{i-1}}^{\xi_i} u(\xi,t)f(\xi)\mathrm{d}\xi, \quad \overline{k_i u_i} = \int_{\xi_{i-1}}^{\xi_i} k(\xi,t)u(\xi,t)f(\xi)\mathrm{d}\xi$$

and solve the system

$$\frac{\mathrm{d}\overline{u}_i(\xi,t)}{\mathrm{d}t} + \overline{k_i(\xi)u_i(\xi,t)} = 0, \qquad \forall i = 1 \ldots N$$

In this transformation, some information on the behaviour of the solution in the stochastic domain is lost. But as stated above, if the mean is the major object of interest, this might not matter.

This line of reasoning is at this moment being pursued by K. Myerscough and Dr. M.I. Gerritsma. Although it is a work in progress, preliminary results look promising.

Bibliography

[1] P. Vos. *Time-dependent polynomial chaos.* Master's thesis, Delft University of Technology, November 2006.

[2] J. van der Steen. *Time-Dependent General Polynomial Chaos.* Master's thesis, Delft University of Technology, December 2008.

[3] W.K. Liu, T. Belytschko, and A. Mani. Random field finite elements. *International Journal for Numerical Methods in Engineering*, 23:1831–1845, 1986.

[4] N. Wiener. The homogeneous chaos. *American Journal of Mathematics*, 60(4):897–936, October 1938.

[5] A. Siegel, T. Imamura, and W.C. Meecham. Symbolic calculus of the wiener process and wiener-hermite functionals. *Journal of Mathematical Physics*, 6(5):695–706, October 1965.

[6] R.G. Ghanem and P.D. Spanos. Spectral stochastic finite-element formulation for reliability analysis. *Journal of Engineering Mechanics*, 117(10):2351–2372, October 1991.

[7] D. Xiu and G.E. Karniadakis. The Wiener-Askey polynomial chaos for stochastic differential equations. *SIAM Journal for Scientific Computing*, 24(2):619–644, 2002.

[8] D. Xiu and G.E. Karniadakis. Modeling uncertainty in flow simulations via generalized polynomial chaos. *Journal of Computational Physics*, 187:137–167, 2003.

[9] X. Wan and G.E. Karniadakis. An adaptive multi-element generalized polynomial chaos method for stochastic differential equations. *Journal of Computational Physics*, 209(2):617–642, 2005.

[10] L. Mathelin, H.M. Yousuff, and T.A. Zang. Stochastic approaches to uncertainty quantification in CFD simulations. *Numerical Algorithms*, 38:209–236, 2005.

[11] I. Babuška, F. Nobile, and R. Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal of Numerical Analysis*, 45(3):1005–1034, 2007.

[12] G.J.A. Loeven, J.A.S. Witteveen, and H. Bijl. Probabilistic collocation: An efficient non-intrusive approach for arbitrarily distributed parametric uncertainties. *45th AIAA Aerospace Sciences Meeting and Exhibit*, January 2007. AIAA 2007-317.

[13] J. Foo, X. Wan, and G.E. Karniadakis. The multi-element probabilistic collocation method (ME-PCM): Error analysis and applications. *Journal of Computational Physics*, 227:9572–9595, 2008.

[14] R.H. Cameron and W.T. Martin. The orthogonal development of nonlinear functionals in series of fourier-hermite functionals. *Annals of Mathematics*, 48:385, 1947.

[15] G.E. Karniadakis and S. Sherwin. *Spectral/hp element methods for computational fluid dynamics.* Oxford science publications, second edition, 2005.

[16] P.Z. Peebles. *Probability, random variables, and random signal principles.* McGraw-Hill, 1993.

[17] M.J. Gander and A.H Karp. Stable computation of high order Gauss quadrature rules using discretization for measures in radiation transfer. *Journal of Quantitative Spectroscopy Radiative Transfer*, 68:213–223, 2001.

[18] G.H. Golub and J.H. Welsch. Calculation of Gauss quadrature rules. *Mathematics of Computation,*, 23 (106):221–230, 1969.