Inaugural-Dissertation

zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich-Mathematischen Gesamtfakultät
der
Ruprecht-Karls-Universität
Heidelberg

vorgelegt von
Dipl.-Ing., M.Sc. Chen Song
aus Shandong, China

Tag der mündlichen Prüfung:

Uncertainty Quantification for a Blood Pump Device with Generalized Polynomial Chaos Expansion

Advisor:	Prof. Dr. Vincent Heuveline
Co-Advisor:	Prof. Dr. Raffaele De Simone

To my family.

Abstract

Nowadays, an increasing number of numerical modeling techniques, notably by means of the finite element method (FEM), are involved in the industrial design process and play a vital role in the area of the biomedical engineering. Particularly, the computational fluid dynamics (CFD) has become a promising tool for investigating the fluid behavior and has also been used to study the cardiovascular hemodynamics to predict the blood flow in the cardiovascular system over the recent decades.

However, simulating a fluid in rotational frames is not trivial, as the classical fluid calculation considers that the geometry of the fluid domain does not alter along the time. In the meanwhile, due to the high rotating speed and the complex geometry of the ventricular assist device (VAD), a turbulent flow must be developed inside the pump housing. The Navier-Stokes equations are not applicable in respect of our available computing resource, additional assumptions and approaches are often applied as a means to model the eddy formation and cope with numerical instabilities.

For many applications, there is still a big gap between the experimental data and the numerical results. Some of the discrepancies come especially from uncertain data which are used in the physical model, therefore, Uncertainty Quantification (UQ) comes into play. The Galerkin-based polynomial chaos expansion method delivers directly the mean and higher stochastic moments in a closed form. Due to the Galerkin projection's properties, the spectral convergence is achieved.

This thesis is dedicated to developing an efficient model to simulate the blood pump assuming uncertain parametric input sources. In a first step, we develop the shear layer update approach built on the Shear-Slip Mesh Update Method (SSMUM), our proposition facilitates the update procedure in parallel computing by forcing the local vector to retain the same structure. In a second step, we focus on the Variational Multiscale method (VMS) in order to handle the numerical instability and approximate the turbulent behavior in the blood. As a consequence of utilizing the intrusive Polynomial Chaos formulation, a highly coupled system needs to be solved in an efficient manner. Accordingly, we take advantage of the Multilevel preconditioner to precondition our stochastic Galerkin system, in which the Mean-based preconditioner is prescribed to be the smoother. Besides, the mean block is preconditioned with the Schur-Complement method, which leads to an acceleration of the solution process. Hence, by developing and combining the proposed solvers and preconditioners, dealing with a large coupled stochastic fluid problem on a modern computer architecture is then feasible. Furthermore, based on the stochastic solutions obtained from the previous described system, we obtain valuable information about the blood flow accompanied with certain level of confidence, which is beneficial for designing a new blood-handle device or improving the current model.

Zusammenfassung

Heutzutage spielen in zunehmender Zahl numerische Modellierungstechniken - allen voran Techniken, welche die Finite-Elemente-Methode (FEM) verwenden - eine Rolle bei Prozessen im Industriedesign, Dar uber hinaus sie sind ebenso nicht mehr wegzudenken aus dem Bereich der biomedizinischen Ingenieurwissenschaften. Insbesondere die Numerische Strömungsmechanik (CFD) hat sich zu einem wichtigen Werkzeug entwickelt, um das Verhalten von Fluiden zu untersuchen, und wurde in den letzten Jahrzehnten unter anderem dazu verwendet, kardiovaskuläre Hämodynamik zu studieren und den Blutfluss im kardiovaskulären System vorherzusagen.

Allerdings birgt die Simulation einer Flüssigkeit in einem rotierenden Gebiet einige Herausforderungen. Denn bei klassischen Flüssigkeitsberechnungen wird die Geometrie eines Strömungsgebiets als unabhängig von zeitlichen Veränderungen betrachtet. Wegen der hohen Rotationsgeschwindigkeit und der komplexen Geometrie des ventrikulären Unterstützungssystems muss sich im Pumpgehäuse eine turbulente Strömung entwickeln. Die Navier-Stokes-Gleichungen sind nicht direkt anwendbar im Hinblick auf die uns zur Verfügung stehenden Berechnungsressourcen, zusätzliche Annahmen und Herangehensweisen werden oft eingesetzt, um beispielweise die Formierung von Kehrwasser zu modellieren und numerische Instabilitäten zu bewältigen.

Für viele Anwendungen besteht noch eine große Lücke zwischen den experimentellen Daten und den numerischen Ergebnissen. Einige Diskrepanzen haben ihre Ursache insbesondere in unsicheren Daten, die für die physikalischen Modelle verwendet werden; daher kommt die Unsicherheitsquantifizierung (UQ) ins Spiel. Die Galerkin-basierte polynomielle Chaosentwicklung liefert direkt den Erwartungswert und höhere stochastische Momente in einer geschlossenen Form. Aufgrund der Eigenschaften der Galerkin-Projektion erhält man direkt die Konvergenz im Spektralraum.

Die vorliegende Arbeit widmet sich der Entwicklung eines effizienten Modells, um eine Blutpumpe unter der Annahme unsicherer parametrischer Eingabe-Quellen zu simulieren. In einem ersten Schritt entwickeln wir dabei den sogenannten Shear Layer Update Approach, gestützt auf die Shear-Slip Mesh Update Method (SSMUM); dieses Vorgehen erleichtert die Aktualisierungsprozedur im Parallelrechner, indem sie den Erhalt der Struktur für den lokalen Vektor erzwingt. In einem zweiten Schritt konzentrieren wir uns auf die Variational Multiscale Method (VMS), um die numerische Instabilität zu bewältigen und das turbulente Verhalten des Blutes zu approximieren. Die Verwendung der intrusiven polynomiellen Chaosentwicklung führt dazu, dass wir ein stark gekoppeltes System auf effiziente Weise lösen müssen. Dazu bedienen wir uns des Multilevel-Vorkonditionierers, um unser stochastisches Galerkin-System vorzukonditionieren, bei welchem der Vorkonditionierer des Erwartungswertes als Glätter für alle stochastischen Momente verwendet wird. Weiterhin wird der Erwartungswert-Block mit dem Schur-Komplement-Verfahren vorkonditioniert, was zu einer Beschleunigung des gesamten Lösungsprozesses führt. Auf diese Weise wird es, durch die Entwicklung und Kombination der vorgeschlagenen Löser und Vorkonditionierer, praktikabel, ein großes, gekoppeltes stochastisches Strömungsproblem auf einer modernen parallelen Computer-Architektur zu behandeln. Ferner erhalten wir, basierend auf den aus dem vorherigen System erzielten stochastischen Lösungen, wertvolle Informationen über den Blutfluss mit einer gewissen stochastischen Verlässlichkeit. Dies ist von Vorteil sowohl für das Design eines neuen blutführenden Medizinproduktes als auch für die Verbesserung des gegenwärtigen Modells.

Acknowledgements

I would like to express my special thanks to my advisor Prof. Dr. Vincent Heuveline for proving me the great opportunity of being his Ph.D. student at the faculty of Mathematics and Computer Science at Heidelberg University. I deeply appreciate his kind support and fruitful scientific discussion, which were notably helpful for my doctoral study. This fascinating journey of exploration in Mathematics and medical applications can not be accomplished without his gentle assistance.

I also want to thank Dr. Michael Schick, who gave me enormous support especially at the very beginning of my Ph.D. time. He helped me to shape the idea about my thesis and shared his expertise in the field of Uncertainty Quantification (UQ).

Many thanks to Heidelberg Institute for Theoretical Studies (HITS) for providing me the grant to accomplish my thesis. Along my Ph.D. time, the diverse research activities at HITS colored the exchange with other researchers and also opened my mind in different research areas. In addition, the HITS administration took care of many practical issues for me, such that I could concentrate on the research. Their professionalism is thoroughly remarkable. Besides, I would like to thank the colleagues at Laboratoire de Biomécanique et Mécanique des Chocs (LBMC) and Laboratoire de Mécanique des Fluides et d'Acoustique (LMFA) during my study in Lyon, they brought me at the first place into research, the experience with them leads me forward after my engineering study.

Hereby, I like to crave the attention that I profoundly appreciate the colleagues at Engineering Mathematics and Computing Lab (EMCL), we spent many great moments together. Their kindness, intelligence and enthusiasm in research helped me from the beginning throughout these wonderful years, and the countless discussions during our coffee breaks were not only about the social interaction, but the insightful scientific comments were priceless. It was my privilege to work with them and the friendship is a great treasure in my life. A particular thanks for Ms. Lydia Mehra, who put everyone in the group together, and the whole process can not look like so smooth and surgeless without her.

Last but not least, I want to thank my whole family for their many years support. My sincere appreciation devotes to my father Zhishun Song, my mother Shenping Fan, grandmother Guifang Zhang and grandfather Rongxiang Fan, who I could not every see again but taught me to be a better person.

Contents

Chapter 1

Introduction

This work is concerned with the quantification of uncertainties in a rotating device considering of the generalized Polynomial Chaos (gPC) method. More precisely, we are interested in modeling the propagation of input uncertainties, and how they can influence the blood flow through a blood pump device [55]. Some main challenges need to be addressed: the modelization of the rotating mechanism, the simulation of a high Reynolds number flow, the development of efficient solvers and the preconditioning techniques for an intrusive stochastic Galerkin system.

There is considerable high mortality caused by the heart failure worldwide. The patients suffer from the insufficient hearts' pumping action, which triggers several symptoms, such as shortness of breath, fatigue, confusion, high heart rate. The ventricular assist devices (VADs) became one of the most effective solutions over the last two decades for the patients who have cardiovascular diseases. The main functionality of the blood pump is to provide an additional blood flow into the cardiovascular system according to the body needs. In that context, there is still on important need to increase the safety under different operational conditions for the blood pump devices, especially for the long-term users.

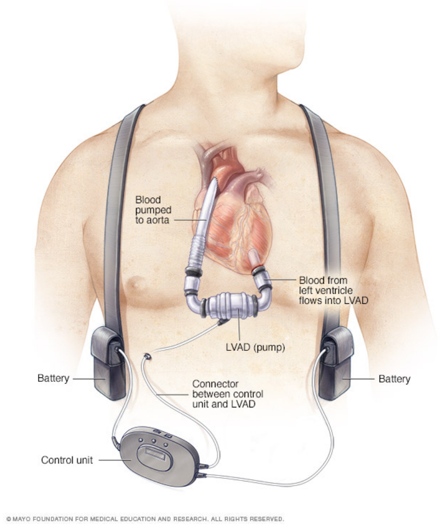
Numerical methods play already an important key role in the industrial design process, in particular the Computational Fluid Dynamics (CFD), which provides a qualitative and quantitative prediction about the fluid flow and improves the development process significantly. However, the ability to investigate the impact of uncertain parameters is a key issue in order to achieve a reliable design. Uncertainty Quantification (UQ) is an important ingredient with respect to this aspect, it aims at quantifying the impact of uncertainty on the computed solutions. In our case particularly, it implies that the UQ techniques can help engineers to better understand how the blood pump device works under different conditions and ensure the reliability and the fidelity of the instruments. Furthermore, UQ can also enhance the patient-specific assessment, especially, it provides a possibility to take the variance of anatomical parameters of an individual patient into account.

1.1 Heart Failure and Blood Pump

Heart failure (HF) generally refers to the congestive heart failure, this disease mainly happens when the heart can not support enough blood as the human body requires. Due to the structural or functional changing of the heart, it leads to two kinds of heart failures: The **systolic HF**, it happens when the heart ventricles can not pump the blood strong enough during the systole. Or the **diastolic HF**, it happens when there is not enough blood to fill the heart chamber during the diastole. In both situations, the blood moves back into the lung, which causes a further congestion. For that reason, the heart failure is often referred to as the congestive heart failure (CHF). The heart failure can affect the left ventricle (left-sided HF), the right ventricle (right-sided HF) or even both (bi-ventricular HF). A large quantity of diseases can trigger the heart failures, which result in either thickened and stiffened ventricular walls or thinned and stretched ventricular walls. The heart cells are then ruffled, this effect causes ultimately the arrhythmia.

There are more than 26 million people suffering from the heart failure globally and 15 million new patients a year in Europe [3]. However, the prospect of HF patients is normally not very optimistic, the survival rate is even lower than the cancer. Furthermore, this illness places tremendous strain not only on the patients, but also on the social health system. Especially, considering the growth of the percentage of ageing populations in the next decades, the demands on the efficient and affective health care services are urgently needed. The ventricular assist device is for the moment one of the most effective solutions for replacing partially or completely the functionalities of a defected heart.

Ventricular assist device (VAD), also known as **blood pump**, is a mechanical instrument that supports the heart to pump the blood through the body. The ventricular assist devices have mainly three different models: the right ventricular assist device (RVAD), the left ventricular assist device (LVAD) and the bi-ventricular assist device (BiVAD). They support either the right ventricle, the left ventricle or both. The LVAD is the most implanted model among these three, as the left ventricle pumps the blood through the aorta under a high pressure (the left ventricle is the left low chamber of the heart).



© MAYO FOUNDATION FOR MEDICAL EDUCATION AND RESEARCH. ALL RIGHTS RESERVED.

Figure 1.1: The aorta and the heart are connected additional by a ventricular assist device. The battery provides the power to the pump. The control unit can conduct the flow rate, only the pipe is implanted inside the body (source: http://www.mayoclinic.org/).

In general, the ventricular assist devices have two types of purposes. One type is designed to be used by the patients who are waiting for a surgery, it is made for the short-term use, which is also called "bridge to transplant". This type of VADs can support patients after the surgery until the heart recovers. Another type is made for the long-term use, the device serves as a long time treatment solution, which is also called "destination therapy". This type of VADs is designed for the patients who are not eligible candidates for the heart transplantation.

1.2 Uncertainty Quantification

Uncertainty Quantification (UQ) aims at investigating the impact of errors, which are associated with input data and/or model parameters, in order to provide more reliable results in real world applications. UQ is a broad and interdisciplinary topic, it involves different research fields, such as applied mathematics, computer science, physics, statistics, engineering. UQ is the end-to-end study of the reliability of scientific inferences and is also a technique to discover the dependencies and correlations between data [28, 160, 82].

Overall, there are two categories of uncertainties:

- **Aleatoric Uncertainty**: irreducible uncertainties. It means that certain parameters or information are inherently "unstable" (in the sense that the values of characteristic features are fluctuating and unpredictable precisely). For example, the angle of attack of an airfoil during the fly, it can have some small variations. Aleatoric uncertainty is conventionally unbiased and typically treated under the probabilistic framework.
- **Epistemic Uncertainty**: reducible uncertainties. This kind of uncertainty comes from the simplification of the physical models or simply the lack of knowledge. It can be diminished by improving the techniques or using the advanced methods. For instance, employing the boundary layer model assumption for certain flows or using the nonlinear models instead of linear models for the soft tissue simulation. It is generally biased, and less clearly modeled under the probabilistic framework.

Uncertainty Quantification is usually divided into two disciplines, the forward problems and the inverse problems. The forward problems concern with the uncertainties in the system outputs propagated from the uncertain input data through a computational model. The inverse problems are focused on using the outcomes of the simulations to examine the values of the parameters which characterize the model.

In the first place, we write an abstract forward problem as follows:

$$u(\xi) = \mathcal{M}(X(\xi)) \ , \tag{1.1}$$

where, $X(\xi)$ represents the non-deterministic input data, which is characterized by the random variable ξ (a random variable is a variable whose domain is a set of random events θ, and its range is a numerical representation of these outcomes, denoted by $\xi = \xi(\theta)$). $\mathcal{M}$ is our system of interest, and we are interested in the random output $u(\xi)$. For the forward problem, we want to calculate $u(\xi)$, which contains the information of the error propagation from $X(\xi)$ through $\mathcal{M}$. In contrast to the latter case, if we want to study the sensitivity of $\mathcal{M}$ to the input, the inverse problem needs to be involved. For the inverse problems, $\mathcal{M}$ is still the system of interest, whereas $u(\xi)$ becomes the observation data or the output information. The goal is to infer $X(\xi)$, such that $\mathcal{M}(X(\xi)) = u(\xi)$. In the most common situation, the inverse problem is ill-defined.

As mentioned above, UQ is about measuring the impact of errors on the numerical solutions. The accuracy of the numerical calculations can be influenced mostly by the following three error contribution [109]:

- Model errors: The numerical calculation relies on the mathematical formulation, which should describe the phenomena of the considered system precisely, although this is not always the case. When one tries to describe a physical occurrence, some assumptions and simplifications are commonly taken into account. As the result, even an analytical solution obtained from the mathematical formulation could lead to wrong results due to erroneous assumptions associated to the mathematical model. For example, the turbulence models can not fully resolve the vortex in a small scale.
- Numerical errors: In practice, the numerical solutions are generally obtained by the approximation techniques, all these techniques introduce the numerical errors. For instance, the mesh discretization can not exactly describe a physical domain, or using the finite polynomials as the shape functions has always truncation errors.
- Data errors: This class of error is connected with the data used by the numerical model, and these data only reflect partially the reality because of inaccurate measurements. The errors may involve the geometry of computing domain, the initial conditions and the parameter

variabilities. For instance, the gravity is often considered as constant for many applications, the geographical influence is simply ignored. The deterministic simulation tries to view this kind of variation as a noise and minimize their influence on the final results as much as possible. Nevertheless, the sensitivity of the system needs to be further analyzed in regard to the noise of the input, such that we can obtain a reasonable relation between the numerical results and the reality. Under the Uncertainty Quantification framework, we seek to model these errors with the aid of the stochastic models.

The main consideration of this work is focusing on the forward uncertainty propagation problem, where the system parameters are regarded as uncertain. The aim is to quantify the influence on the numerical solutions by taking into account the input data errors with the help of an intrusive Stochastic Galerkin Method [109]. Particularly, a blood flow is subject to a non-deterministic system with the uncertain parametric information through a blood pump instrument [55]. Furthermore, the focal point of this work is about solving an intrusive Polynomial Chaos (PC) system and investigating the uncertainties.

1.2.1 Non-intrusive Method

The non-intrusive method treats the deterministic simulation as a "black box" solver, it does not require any modification from the existing code. The stochastic moments are only computed based on the realizations of the random variable ξ (Equation (1.1)). The standard technique is to create a limited number of samples $\xi^{(1)}, \xi^{(2)}, ..., \xi^{(N)}, N \in \mathbb{N}$. After that, realizing the computation individually with each $\xi^{(i)}$, $i \in [1, N]$ as follows:

$$u(\xi^{(i)}) = \mathcal{M}(X(\xi^{(i)})) \; , \tag{1.2}$$

for $i = 1, ..., N$. For instance, the mean value can be obtained via a straightforward way:

$$\mathbb{E}(u) = \lim_{N \to \infty} \sum_{i=1}^{N} u(\xi^{(i)}) w_i \; , \tag{1.3}$$

where, $u(\xi^{(i)})$ is the response of the system $\mathcal{M}$ with respect to $\xi^{(i)}$, w_i is pre-defined weight. The "non-intrusive" concept implies that once $\xi^{(i)}$ and w_i are available, the deterministic system needs to run N times according to the different value of $\xi^{(i)}$.

There are several non-intrusive methods, such as, the Monte-Carlo method [125], the Stochastic Collocation method [121, 175], the Sparse Grid technique [156] and the non-intrusive Polynomial Chaos [138, 120].

The Monte-Carlo method is a well-known non-intrusive approach. Considering Equation (1.3), the general Monte-Carlo method suggests having an equal weight, i.e. $w_i = \frac{1}{N}$, and the values of ξ_i are generated purely random. Therefore, the sampling points are unbiased and consistent. However, the convergence rate of the standard Monte-Carlo method is $\mathcal{O}(1/\sqrt{N})$, it means that we need to sample four times more in order to reduce the error by half. Hence, various techniques are proposed for improving the sampling quality, e.g.: The quasi Monte-Carlo [129, 71], it gives an improved convergence rate as $\mathcal{O}((\log N)^k N^{-1})$ (k is the number of random variables). Or the Latin hypercube sampling [95], this stratified sampling method ensures an even number of sampling points for each stochastic variable. In recent years, the Multilevel Monte-Carlo method [74, 64] has gained a lot of attention because of the fast convergence in the variance.

The Stochastic Collocation (SC) method [121, 122, 13] models the uncertain parameters via the pre-selected basis, usually the Lagrange interpolating polynomials. The stochastic solution relies simply on the interpolation instead of approximating the solution in the pre-defined stochastic subspaces, a decomposition of the stochastic parameters and physical parameters is achievable. However, the number of Collocation points increases exponentially due to the extension of the tensor product. In order to circumvent the *curse of dimensionality* [24], the Sparse

Grid (SG) method is introduced based on the Smolyak sparse grid [156]. In contrast to the Stochastic Collocation, the Sparse Grid can reduce the amount of sampling points from an order $\mathcal{O}(N^k)$ to $\mathcal{O}(N(\log N)^{(k-1)})$ with the same accuracy of the solution (k is the number of random variables) [130].

Another type of the non-intrusive method is the non-intrusive Polynomial Chaos (NIPC) [138]. This method combines the spectral method with the sampling techniques. The NIPC relies on the numerical integration to evaluate the coefficient of each spectral mode instead of substituting the model parameters and variables into the governing equations. Moreover, by using advanced sampling techniques, especially if they correspond to the probability distribution of the random parameters, NIPC can achieve a fast convergence comparing to the Monte-Carlo method [85].

1.2.2 Intrusive Method

In this work we concentrate on the intrusive approach based on the Polynomial Chaos expansion (PCE). Ghanem and Spanos [62] were the very first scholars, who applied this technique to the real world applications. Their work was mainly focusing on the uncertainty propagation caused by the uncertain input within a mechanical structure. Thenceforward, the Polynomial Chaos expansion is employed for the incompressible Navier-Stokes equations [119, 120, 177] and also for more complex problems [45, 178, 148, 81].

The main concept of the intrusive PCE is to build a direct functional connection between the random variable ξ and the stochastic solution $u(\xi)$, such as:

$$u(\xi) = \lim_{P\to\infty} \sum_{i=0}^{P} u_i \psi_i(\xi) \ . \tag{1.4}$$

Here, the set of ψ_i is defined for the pre-selected basis functionals, u_i denotes the stochastic Polynomial Chaos mode solution. As a consequence, there are some regularity assumptions for the solution $u(\xi)$ in order to be able of constructing the series representation. The stochastic solution $u(\xi)$ can be obtained by means of the stochastic Galerkin projection method. This approach projects the model system $\mathcal{M}$ (Equation (1.1)) onto a stochastic space, which is spanned by the basis functionals ψ_i. As the solution and the input random data are both expressed by using the Chaos Polynomials (e.g. Equation (1.4)), the intrusive method leads to a coupled system of equations including stochastic modes. Therefore, we have to further investigate the solving strategies for this highly coupled system in order obtain the stochastic solutions efficiently (Chapter 4).

1.3 The Important Ingredients of the Thesis

This work focuses on applying the intrusive Polynomial Chaos expansion technique to a blood pump simulation. Modeling a rotating device has a high potential for industrial applications, particularly how one can deal with a moving mesh. We propose a shear layer update approach, which is similar to the Shear-Slip Mesh Update Method [23]. Our suggestion is more suitable in the context of the High Performance Computing (HPC). Furthermore, since simulating the high Reynolds number flow is also very challenging, we propose to use the Variational Multiscale method (VMS) [91] to cope with the turbulence modeling. The VMS considers a fluid flow to be decomposed into different scales and introduces the variational projections for building the links between the scales. In order to simulate the uncertainties propagation within a blood pump device, we employ the generalized Polynomial Chaos expansion (gPCE) technique [178]. The Galerkin projection forms the governing system based on a weighted residual formalism, and it inherits the corresponding convergence property from the Galerkin methods. However, owing to the special construction of Galerkin method, the resulting coupled stochastic system becomes very large in regard to the order of Chaos Polynomials. The direct solvers and the standard iterative solvers are then not able to deal with this system efficiently. Therefore, in this work, we consider a Polynomial Chaos expansion Multilevel preconditioner in order to handle the stochastic system. Furthermore, the stochastic Mean-based preconditioner is used as the smoother in the Multilevel approach, and the Schur Complement method is applied to precondition the linear system associated with the mean block matrix. Finally, based on our efficient solving strategies, the stochastic solutions can be obtained within a reasonable time frame.

1.4 Outline

This work is organized as follows: Chapter 2 describes the modelization of the blood pump device, it depicts the shear layer update approach for the moving mesh strategy and the Variational Multiscale method for the high Reynolds number flow modeling. Chapter 3 provides an overview of the generalized Polynomial Chaos expansion (PCE) technique. The special hierarchical structure of the PCE is studied. The governing equation, which combines the Variational Multiscale approach and the intrusive Polynomial Chaos expansion based on the spectral-stochastic finite element method (SSFEM), is introduced. In addition, the numerical discretization schemes are also presented. Chapter 4 concentrates on the numerical algorithms and the solving strategies related to the stochastic Galerkin system and the deterministic Variational Multiscale problem. Chapter 5 presents the scalability studies of the fluid solver and the PCE Multilevel preconditioner. Chapter 6 provides the numerical experiments for this blood pump simulation in consideration of the parameterized input uncertainties. Chapter 7 concludes this work and describes the perspectives for possible further developments.

Chapter 2

Modelization of the FDA Blood Pump

2.1 FDA Blood Pump

The prototype (Figure 2.1), which we consider in this work, is the FDA (U.S. Food & Drug Administration) blood pump. This project is established under the framework of the FDA's Critical Path Initiative (CPI) Computational Fluid Dynamic/Blood Damage Project [55]. In this project, the FDA will provide the laboratory experiments of the blood flow in this blood pump device, and the project participants can simulate the blood flow in this device with their own software and models. The comparison between the numerical results and the experiments will be taken place in the end of the project. The purpose of this project is to develop a benchmark model for a centrifugal blood pump in order to enhance the performance of this instrument and promote further developments.

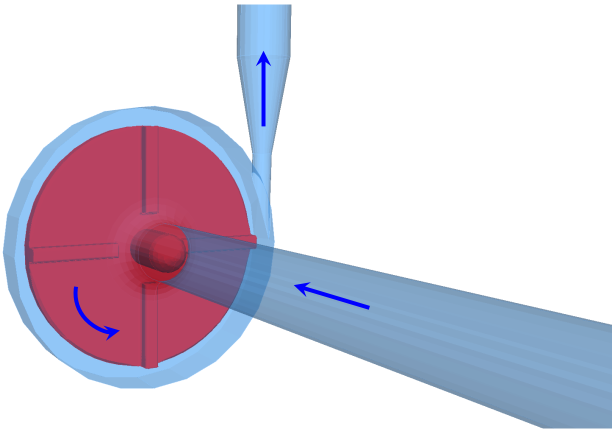


Figure 2.1: A simple schematic diagram of the functionality of the FDA blood pump. The arrows indicate the flow direction.

As mentioned in Chapter 1, the pump device is placed between the lower heart chamber and the aorta in order to provide an additional blood flow into the circulatory system for the heart failure patients. In the pump instrument, the blood flow is driven by the rotor (red in Figure 2.1), whose power is supplied by an external motor. The blood moves toward a nozzle structure for a further acceleration after passing through the pump housing. Concerning this FDA ventricular assist device, the diameter of the pump housing is only 60 mm, and 52 mm for the rotor, the angular speed is 2500 RPM (revolution per minute).

The main challenge of modeling a blood pump device can be considered as two major parts: modeling a high Reynolds number flow and simulating a rotating effect. The following of this chapter provides the detailed information in regard to these two topics.

2.2 Modelization of High Reynolds Number Flow

According to [55], the blood pump must operate in a high rotational speed, the blood is driven by the impeller from the housing to the aorta. In order to study a fluid, the Reynolds number can be used to forecast the flow patterns, namely the laminar flow or the turbulent flow. For our rotating device, the Reynolds number is defined as [31]:

$$Re = \frac{\rho \omega D^2}{\mu} \, . \tag{2.1}$$

Here, ρ denotes the fluid density, ω is the angular speed, D is the diameter of the rotor and μ is the dynamic viscosity. The simulation conditions provided by FDA [55] are stated in Table 2.1:

ρ	$1035\,\mathrm{kg/m^3}$	ω	$261.667\,\mathrm{rad/s}$
D	$52\,\mathrm{mm}$	μ	$0.0035\,\mathrm{kg/m/s}$

Table 2.1: Simulation conditions.

As a result, the Reynolds number in our FDA blood pump is approximately $210,000$, it means that the convection mechanism is dominant compared to the diffusion phenomena. Choosing an appropriate flow model is very crucial to this blood pump simulation. Hence, we give a short overview of the existing methods of the high Reynolds number modelization.

2.2.1 Overview of Methods for Turbulent Flow Modeling

As the turbulent models are derived from the Navier-Stokes equations, we state at first the incompressible Navier-Stokes equations:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} - \frac{\mu}{\rho} \Delta \boldsymbol{u} + \frac{1}{\rho} \nabla p = \boldsymbol{f} \, , \qquad \text{in } [0,T] \times \mathcal{D} \, , \tag{2.2a}$$

$$\nabla \cdot \boldsymbol{u} = 0 \, , \qquad \text{in } [0,T] \times \mathcal{D} \, . \tag{2.2b}$$

Here, $\mathcal{D} \subset \mathbb{R}^d, d = 2,3$, $\mathcal{D}$ is a bounded domain. $\boldsymbol{u}(\boldsymbol{x},t)$ is the velocity, $p(\boldsymbol{x},t)$ is the pressure, and $[0,T]$ is a finite time interval. $\boldsymbol{f}(\boldsymbol{x},t)$ describes the external body force, ρ represents the density, and μ is the dynamic viscosity.

RANS: The Reynolds-Averaged Navier-Stokes equations. The RANS is a statistical approach and is derived by using the Reynolds decomposition (Definition 2.2.1) of the Navier-Stokes equations [136].

Definition 2.2.1 (Reynolds decomposition)**.** *Let $\boldsymbol{\Phi}(\boldsymbol{x},t)$ be the quantity of interest, which is defined on a domain $\Omega \subset \mathbb{R}^d$, $d = 2,3$. t is the time variable, $t \in [0,T]$. We assume that $\overline{\boldsymbol{\Phi}}(\boldsymbol{x})$ exists in terms of the time averaged sense, thus:*

$$\boldsymbol{\Phi}(\boldsymbol{x},t) = \overline{\boldsymbol{\Phi}}(\boldsymbol{x}) + \boldsymbol{\Phi}'(\boldsymbol{x},t) \, . \tag{2.3}$$

$\boldsymbol{\Phi}'(\boldsymbol{x},t)$ is the fluctuating part. $\overline{\boldsymbol{\Phi}}(\boldsymbol{x})$ is the time averaged part, which does not depend on time. $\boldsymbol{\Phi}$ represents the velocity and the pressure in the Navier-Stokes equations.

One additional interpretation by employing the time average is:

$$\overline{\overline{\boldsymbol{\Phi}}} = \overline{\boldsymbol{\Phi}} \, , \quad \text{and} \quad \overline{\boldsymbol{\Phi}'} = 0 \, . \tag{2.4}$$

We insert Equation (2.3) into the incompressible Navier-Stokes equations. Then we take the time average. As $\overline{\boldsymbol{u}}$ is independent of time, the RANS yields [136]:

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\overline{u}_i\overline{u}_j - \frac{\mu}{\rho}\frac{\partial^2 \overline{u}_i}{\partial x_i \partial x_j} + \frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{\overline{\partial u_i' u_j'}}{\partial x_j} = f_i \; , \tag{2.5a}$$

$$\frac{\partial u_i}{\partial x_i} = 0 \; , \tag{2.5b}$$

where, $\overline{u_i' u_j'}$ is the Reynolds stress tensor, which establishes the link between the fluctuating field and the average field.

If we consider the 3D case, there exist six unknowns introduced by the Reynolds stress tensor $\overline{u_i' u_j'}$, which implies that Equation (2.5) is not closed. Hence, several approaches try to provide a closure of the RANS. The Boussinesq hypothesis [30] is built on the turbulent viscosity hypothesis. Boussinesq presumed that the Reynolds stress is proportional to the local mean strain rate. It means that the turbulent stress is linearly dependent with respect to the mean strain (i.e. the turbulent viscosity is constant). The fluid transport effect is simply ignored, because he assumed that the Reynolds stress is generated locally. Therefore, this approach is not valid in general situations. Afterward, Prandtl attempted to correctly derive the turbulent eddy viscosity under Boussinesq's framework with his mixing length theory. This approach can be considered as a zero-equation model or an algebraic model, as the scale of the turbulent velocity is completely determined by the mean flow. More specifically, the zero-, one- or two-equation models are referring to how many partial differential equations needed to make RANS equation to be closed. Moreover, the turbulent kinetic energy model [105] belongs to the one-equation model. This approach replaces the mean flow with the kinetic energy obtained from the transport equation in order to achieve a closure of the equations. The $k-\omega$ and $k-\varepsilon$ models are also widely used, both approaches belong to the two-equation models. The advantage of these two approaches is that the turbulent viscosity can be determined consistently, a specification of the length scale is no more necessary.

LES: The Large Eddy Simulation. The LES is another kind of turbulence model, which is lying between the RANS and the direct numerical simulation [131]. The general LES consists of different steps [128]:

- Decompose the fluid into large-scale and small-scale by using filtering techniques.
- Filter the governing equation into the solvable part and the unsolvable part.
- Model the small eddies with the subgrid-scale model.
- Solve the governing system and obtain the solution only on the large-scale.

$$\boldsymbol{\Phi}(\boldsymbol{x}, t) = \overline{\boldsymbol{\Phi}}(\boldsymbol{x}, t) + \boldsymbol{\Phi}'(\boldsymbol{x}, t) \; . \tag{2.6}$$

Considering the LES decomposition in Equation (2.6), it has a same form as the Reynolds decomposition in Equation (2.3), whereas, the interpretation is different. For LES, $\overline{\boldsymbol{\Phi}}$ is the solvable-scale part (or filtered part), and $\boldsymbol{\Phi}'$ is the subgrid-scale part (or residual part). In contrast, $\overline{\boldsymbol{\Phi}}$ is a random field, $\overline{\boldsymbol{u}'} \neq 0$ and $\overline{\overline{\boldsymbol{\Phi}}} \neq \overline{\boldsymbol{\Phi}}$.

The filtering process can be regarded as a convolution of the velocity with a filtering kernel G:

$$\overline{\boldsymbol{u}}(\boldsymbol{x}, \boldsymbol{t}) = \int_{-\infty}^{+\infty} G(\boldsymbol{x} - \boldsymbol{x}')\boldsymbol{u}(\boldsymbol{x}')dx' \; . \tag{2.7}$$

Following the same technique in Equation (2.5), we have:

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\overline{u}_i\overline{u}_j - \frac{\mu}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{1}{\rho}\frac{\overline{p}}{\partial x_i} + \frac{1}{\rho}\frac{\partial \tau_{ij}}{\partial x_j} = 0 \ , \tag{2.8a}$$

$$\frac{\partial u_i}{\partial x_i} = 0 \ . \tag{2.8b}$$

Here, τ_{ij} is the subgrid stress:

$$\tau_{ij} = \overline{u_i u_j} - \overline{u}_i\overline{u}_j \ . \tag{2.9}$$

According to the Leonard decomposition [110], $\tau_{ij} = L_{ij} + C_{ij} + R_{ij}$. L_{ij} is the Leonard stress, which reflects the interactions within the large-scale, the cross tensor C_{ij} represents the interactions between the large-scale and the small-scale, and the subgrid-scale Reynolds stress R_{ij} stands for the interactions among the subgrid-scales.

Similarly, the subgrid stress introduces an unclosed system, so τ_{ij} must be modeled with supplementary models. The most common subgrid-scale models employ the Boussinesq hypothesis in order to compute the subgrid-scale stress, e.g.:

$$\tau_{ij} = 2\mu_t\overline{S}_{ij} - \frac{2}{3}K\delta_{ij} \ , \tag{2.10}$$

where $\overline{S}_{ij}$ is the traceless mean rate of strain tensor, K is the turbulent kinetic energy, and μ_t is the eddy viscosity. Various subgrid-scale Reynolds stress models are proposed besides the previous approach, e.g., the Smagorinsky model [155], the dynamic subgrid-scale model [61] [112], the similarity model [16], the mixed model [180, 6, 146].

DNS: The direct numerical simulation. From the modeling point of view, the DNS is truly straightforward. It solves the unsteady Navier-Stokes without making any auxiliary assumptions. It hence computes the flow problems directly on the scale of the viscous dissipation. In other words, the mesh grid size and the time step must be fine enough to capture the dynamics numerically [131].

The DNS is used currently only for theoretical studies, the cost of the DNS computations is proportional to the Reynolds number in the order of $Re^{37/14}$ [40]. For this reason, computing an engineering application with the DNS requires a huge amount of computing resources.

SUPG/PSPG/GLS: The Streamline Upwind Petrov-Galerkin / Pressure-Stabilizing Petrov-Galerkin stabilized / Galerkin least-squares scheme. Using the Galerkin finite element method for solving the incompressible Navier-Stokes equations introduces two major sources of the instability, which are the advection term in the momentum equation and the inappropriate combination of interpolation functions for the velocity and the pressure [165]. So as to reduce these instabilities, a combination of the following three stabilization schemes is commonly employed.

The SUPG formulation [34] considers the momentum conservation equation as a parameter. This approach introduces an artificial diffusion only along the streamline direction, and the amount of the diffusion is typically controlled by the stabilization parameter τ_{SUPG}. If we consider a general partial differential equation (PDE) with a form:

$$\mathcal{L}u = f \ , \tag{2.11}$$

where, $\mathcal{L}$ is a general differential operator. The weak form of Equation (2.11) can be formulated like:

$$\int_{\mathcal{D}} v^* \cdot (\mathcal{L}u - f) = 0 \ , \tag{2.12}$$

where $\mathcal{D}$ is the spatial domain. The test function v^* is written as:

$$v^* = v + \tau_{SUPG}\mathcal{L}_{adv}v \ . \tag{2.13}$$

Here, $\mathcal{L}_{adv}$ is the advection part of the differential operator $\mathcal{L}$. v is in the same function space as the solution u. However, the test function v^* does not have to be in the same function space. Therefore, the SUPG introduces an extra weighted advection part of the governing equation into the test function. Furthermore, the consistency of this formulation is fulfilled as the exact solution satisfies the weak formulation.

The PSPG formulation [167] is very similar to the SUPG, and its aim is to stabilize the coupling problems. Analogously to the SUPG, τ_{PSPG} is also used to regulate the magnitude of the stabilization. If we only look at the NS equations, the PSPG prefers to introduce the product of the residual of momentum equation and the perturbation $\tau_{SUPG}\nabla q$, here q is the test function of the continuity equation.

The GLS formulation [92] is encouraged by the numerical analysis rather than introducing artificial diffusions. This approach is normally used to evade the LBB condition [65]. In contrast to the SUPG and the PSPG, the test function is constructed as:

$$v^* = v + \tau_{GLS}\mathcal{L}v \ , \tag{2.14}$$

$\mathcal{L}$ is the total differential operator. For that reason, the GLS stabilization systematically allows an arbitrary combination of basis functions [92].

Thus, if we consider some suitable finite-dimensional function spaces, V^h and Q^h, with the Galerkin finite element method, then the weak formulation of the general stabilized incompressible Navier-Stokes equations combined with three previous techniques is given as:

Find $\boldsymbol{u}_h \in V^h$ and $p_h \in Q^h$, such that,

$$\begin{aligned}\int_{\mathcal{D}} (\frac{\partial \boldsymbol{u}_h}{\partial t} + \boldsymbol{u}_h \cdot \nabla \boldsymbol{u}_h) \cdot \boldsymbol{v}_h \, d\boldsymbol{x} + \int_{\mathcal{D}} \frac{\mu}{\rho} \nabla \boldsymbol{u}_h : \nabla \boldsymbol{v}_h \, d\boldsymbol{x} \\ - \int_{\mathcal{D}} \frac{1}{\rho} p_h \nabla \cdot \boldsymbol{v}_h \, d\boldsymbol{x} - \int_{\mathcal{D}} \boldsymbol{f}_h \cdot \boldsymbol{v}_h \, d\boldsymbol{x} \\ + \sum_{e=1}^{n_e} \int_{\mathcal{D}^e} \boldsymbol{r}_M \cdot \tau_{SUPG} \boldsymbol{u}_h \cdot \nabla \boldsymbol{v}_h \, d\boldsymbol{x} \\ + \sum_{e=1}^{n_e} \int_{\mathcal{D}^e} r_C \cdot \tau_{GLS} \nabla \cdot \boldsymbol{v}_h \, d\boldsymbol{x} = \boldsymbol{0} \ , \qquad \forall \boldsymbol{v}_h \in V^h \ ,\end{aligned} \tag{2.15a}$$

$$\int_{\mathcal{D}} q_h \nabla \cdot \boldsymbol{u}_h \, d\boldsymbol{x} + \sum_{e=1}^{n_e} \int_{\mathcal{D}^e} \boldsymbol{r}_M \cdot \tau_{PSPG} \nabla q_h = 0 \ , \qquad \forall q_h \in Q^h \ , \tag{2.15b}$$

where,

$$\boldsymbol{r}_M = \frac{\partial \boldsymbol{u}_h}{\partial t} + \boldsymbol{u}_h \cdot \nabla \boldsymbol{u}_h - \frac{\mu}{\rho} \Delta \boldsymbol{u}_h + \frac{1}{\rho} \nabla p_h - \boldsymbol{f}_h \ , \tag{2.16a}$$

$$r_C = \nabla \cdot \boldsymbol{u}_h \ . \tag{2.16b}$$

$\boldsymbol{r}_M$, r_C are the residuals of the momentum equation and the continuity equation, respectively. $\mathcal{D}$ is the spatial domain, $\mathcal{D}^e$ is the spatial domain of a mesh cell e, n_e is the number of mesh cells. $\boldsymbol{u}_h$ and p_h are the discrete solutions for the velocity and the pressure, and $\boldsymbol{f}_h$ is the external body force.

VMS: The Variational Multiscale method. The VMS is a theoretical framework for the scale separation that one can cope with the turbulent flow. If we are only interested in the solution on the macroscopic level, the homogeneous theory is very robust and efficient. But once the physical behavior on the subscale becomes non-negligible, a sub-linear scaling algorithm is often required. The VMS provides a general framework for covering the influence from the fine-scale into the governing system [90, 170]. The VMS method suggests a splitting of the physical models in order to handle the multiscale phenomena in the engineering applications [90, 170]. First, we state a two-scale decomposition (Figure 2.2) for the Navier-Stokes equations:

$$\boldsymbol{u} = \overline{\boldsymbol{u}} + \tilde{\boldsymbol{u}} \ , \tag{2.17a}$$

$$p = \overline{p} + \tilde{p} \ . \tag{2.17b}$$

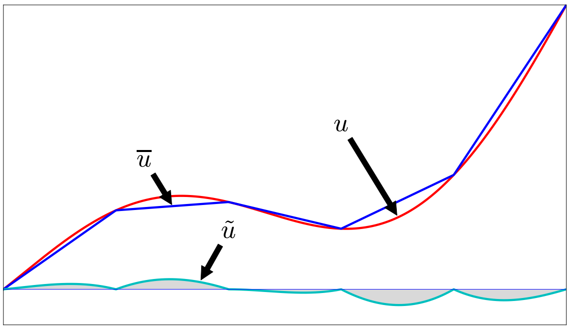


Figure 2.2: Two-scale sum decomposition.

This decomposition consists of a division of the exact solution $(\boldsymbol{u}, p)$ into two separated scales. $(\overline{\boldsymbol{u}}, \overline{p})$, $(\tilde{\boldsymbol{u}}, \tilde{p})$ denote the solution on the resolvable scale (or coarse-scale) and the solution on the unresolvable scale (or fine-scale), respectively. By this means, $(\overline{\boldsymbol{u}}, \overline{p})$ can be generally approximated by any kind of numerical methods. $(\tilde{\boldsymbol{u}}, \tilde{p})$ is placed on the small-scale, which is not captured conversely. Under the finite element method framework, $(\overline{\boldsymbol{u}}, \overline{p})$ may reflect the solution obtained on a rough mesh, while $(\tilde{\boldsymbol{u}}, \tilde{p})$ represents the solution on a fine mesh.

There exists also the three-scale decomposition by means of the large resolvable scale, the small resolvable scale and the unresolvable scale [41]. The scale decomposition is defined as:

$$\boldsymbol{u} = \overline{\boldsymbol{u}} + \hat{\boldsymbol{u}} + \tilde{\boldsymbol{u}} \ , \tag{2.18a}$$

$$p = \overline{p} + \hat{p} + \tilde{p} \ . \tag{2.18b}$$

Here, $(\overline{\boldsymbol{u}}, \overline{p})$ is the solution on the large resolvable scale, $(\hat{\boldsymbol{u}}, \hat{p})$ is the solution on the small resolvable scale, and $(\tilde{\boldsymbol{u}}, \tilde{p})$ is the solution on the unresolvable scale. The main purpose of the three-scale separation is to take explicitly into account the impact of the eddy dissipation arising from the unresolvable scale $(\tilde{\boldsymbol{u}}, \tilde{p})$ on the resolvable scale. Likewise, the solution relying on the unresolvable scale is not meant to be solved directly, but rather projected on resolvable scales. Moreover, in order to simplify the model, it is also commonly assumed that the unresolvable scale $(\tilde{\boldsymbol{u}}, \tilde{p})$ has no influence on the large resolvable scale $(\overline{\boldsymbol{u}}, \overline{p})$.

With respect to the implementation purpose, we consider that the spaces where the resolvable scales rely on are finite-dimensional spaces, and an infinite-dimensional space is considered for the unresolvable case. Especially for the two-scale decomposition case, we can make use of the finite element spaces to be the resolvable space, and the effects on the unresolvable space are nevertheless represented via additional models. Therefore, governing system can be expressed under the FEM framework.

Remark 2.2.1 (The relation to LES)**.** *One can consider the VMS as an extension of the LES method. On the other hand, it also establishes a transition from the LES to the DNS. The LES only tries to define the prediction on the large-scale which is averaged in the space, whereas, the VMS exploits the scale separation instead of the filter function. Owing to the filtering operators, the LES leads to commutation errors on a bounded domain [48] [27]. Additional treatments of the boundary condition and the boundary layer are often required for the LES as a consequence of the smooth convolution kernel. Moreover, the LES is based on the strong form of the NS equations. On the contrary, the VMS exploits the derived variational formulation of the NS equations.*

In this work, we choose the Variational Multiscale method for modeling our high Reynolds number flow within the pump geometry, because the VMS offers a fundamental system that links the standard stabilized methods and the large eddy simulation modeling. More specifically, we utilize the two-scale residual-based VMS, which has a similar feature as the stabilized finite element scheme, but it provides a more sophisticated theoretical framework. Therefore, we are more independent in selecting the upscaling algorithms, which is not only beneficial for our flow application, but there also exist fundamental studies about fluid problems. The basic concept of the VMS and the selected models will be addressed in Section 2.2.2.

2.2.2 Variational Multiscale Method for the Incompressible Navier-Stokes Equations

In this subsection, we present the basic theory about the two-scale Variational Multiscale method for the incompressible Navier-Stokes equations with the residual-based technique. Before discussing the concepts behind the Variational Multiscale method, we provide here the weak formulation of the incompressible Navier-Stokes equations (Equation (2.2)), as it is often mentioned in the following sections.

Taking $V := H_0^1(\mathcal{D})^d$ and $Q := L^2(\mathcal{D})$. Find $\boldsymbol{u} \in V$, $p \in Q$, such that:

$$\int_{\mathcal{D}} (\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}) \cdot \boldsymbol{v} \, d\boldsymbol{x} + \int_{\mathcal{D}} \frac{\mu}{\rho} \nabla \boldsymbol{u} : \nabla \boldsymbol{v} \, d\boldsymbol{x} \tag{2.19a}$$

$$- \int_{\mathcal{D}} \frac{1}{\rho} p \nabla \cdot \boldsymbol{v} \, d\boldsymbol{x} = \int_{\mathcal{D}} \boldsymbol{f} \cdot \boldsymbol{v} \, d\boldsymbol{x} \, , \qquad [0,T] \times \mathcal{D} \, ,$$

$$\int_{\mathcal{D}} q \nabla \cdot \boldsymbol{v} \, d\boldsymbol{x} = 0 \, , \qquad [0,T] \times \mathcal{D} \, . \tag{2.19b}$$

$\forall \boldsymbol{v} \in V$, $\forall q \in Q$.

Equation (2.19) can also be written by using the operator notation, i.e.:

Find $\boldsymbol{u} \in V$, $p \in Q$, such that,

$$A(\boldsymbol{u}; (\boldsymbol{u}, p), (\boldsymbol{v}, q)) = \boldsymbol{f}(\boldsymbol{v}) \, , \quad \forall (\boldsymbol{v}, p) \in V \times Q \, . \tag{2.20}$$

Two-Scale VMS

The flow is decomposed into the large- and small-scales [1]. A set of coupled equations is derived from this scale separation. The relationship between two scales is generally defined via the residual on the resolvable scale, and the flow problem on the small-scale is additionally modeled in order to achieve the closure of equations.

Let us consider the two-scale direct sum decomposition:

$$V = \overline{V} \oplus \tilde{V} \, , \tag{2.21a}$$

$$Q = \overline{Q} \oplus \tilde{Q} \, . \tag{2.21b}$$

Inserting Equation (2.17) into Equation (2.20), we take the same decomposition for the test functions in Equation (2.20). It yields:

$$A(\boldsymbol{u};(\overline{\boldsymbol{u}},\overline{p}),(\overline{\boldsymbol{v}},\overline{q}))+A(\boldsymbol{u};(\tilde{\boldsymbol{u}},\tilde{p}),(\overline{\boldsymbol{v}},\overline{q}))=\boldsymbol{f}(\overline{\boldsymbol{v}})\ , \qquad \forall(\overline{\boldsymbol{v}},\overline{q})\in\overline{V}\times\overline{Q}\ , \tag{2.22a}$$

$$A(\boldsymbol{u};(\overline{\boldsymbol{u}},\overline{p}),(\tilde{\boldsymbol{v}},\tilde{q}))+A(\boldsymbol{u};(\tilde{\boldsymbol{u}},\tilde{p}),(\tilde{\boldsymbol{v}},\tilde{q}))=\boldsymbol{f}(\tilde{\boldsymbol{v}})\ , \qquad \forall(\tilde{\boldsymbol{v}},\tilde{q})\in\tilde{V}\times\tilde{Q}\ . \tag{2.22b}$$

Equation (2.22a) is then defined for the resolvable scale and Equation (2.22b) for the unresolvable scale. If we decompose $A(\cdot;\cdot,\cdot)$ into a linear part $A_{lin}(\cdot,\cdot)$ and a trilinear part $b(\cdot;\cdot,\cdot)$ (convective term) [97, 93], Equation (2.20) becomes:

$$A(\boldsymbol{u};(\boldsymbol{u},p),(\boldsymbol{v},q)):=A_{lin}((\boldsymbol{u},p),(\boldsymbol{v},q))+b(\boldsymbol{u};\boldsymbol{u},\boldsymbol{v})\ . \tag{2.23}$$

Here, $b(\boldsymbol{u};\boldsymbol{u},\boldsymbol{v})$ consists of several terms because of the sum decomposition. Among these terms, $b(\tilde{\boldsymbol{u}},\tilde{\boldsymbol{u}},\overline{\boldsymbol{v}})$ is actually the Reynolds stress term, $b(\overline{\boldsymbol{u}},\tilde{\boldsymbol{u}},\overline{\boldsymbol{v}})$ and $b(\tilde{\boldsymbol{u}},\overline{\boldsymbol{u}},\overline{\boldsymbol{v}})$ are the cross-stress terms. Equation (2.22b) can be rearranged as follows [1]:

$$A_{lin}((\tilde{\boldsymbol{u}},\tilde{p}),(\tilde{\boldsymbol{v}},\tilde{q}))+b(\tilde{\boldsymbol{u}};\overline{\boldsymbol{u}},\tilde{\boldsymbol{v}})+b(\overline{\boldsymbol{u}};\tilde{\boldsymbol{u}},\tilde{\boldsymbol{v}})+b(\tilde{\boldsymbol{u}};\tilde{\boldsymbol{u}},\tilde{\boldsymbol{v}})=\langle\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p}),(\tilde{\boldsymbol{v}},\tilde{q})\rangle\ , \tag{2.24}$$

where $\langle\cdot,\cdot\rangle=\langle\cdot,\cdot\rangle_{\tilde{V}^*\times\tilde{Q}^*,\tilde{V}^*\times\tilde{Q}^*}$, and

$$\langle\boldsymbol{r}(\overline{\boldsymbol{u}}),\overline{p}),(\tilde{\boldsymbol{u}},\tilde{q})\rangle=\boldsymbol{f}(\tilde{\boldsymbol{u}})-A_{lin}((\overline{\boldsymbol{u}},\overline{p}),(\tilde{\boldsymbol{u}},\tilde{p}))-b(\overline{\boldsymbol{u}},\overline{\boldsymbol{u}},\tilde{\boldsymbol{v}})\ . \tag{2.25}$$

Here, $\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p})=\boldsymbol{r}_M(\overline{\boldsymbol{u}},\overline{p})+r_C(\overline{\boldsymbol{u}})$, $\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p})\in\tilde{V}^*\times\tilde{Q}^*$. $\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p})$ is the system residual on the solvable scale which contains the residual of the momentum equation $\boldsymbol{r}_M(\overline{\boldsymbol{u}},\overline{p})$ and the residual of the continuity equation $r_C(\overline{\boldsymbol{u}})$. Furthermore, Equation (2.24) provides another interpretation that the unresolvable scale can be obtained via a function which takes the residual on the resolvable scale as the parameter, i.e.:

$$(\tilde{\boldsymbol{u}},\tilde{p})\simeq F(\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p}))\ . \tag{2.26}$$

Equation (2.22) can be then simplified to a single function:

$$A(\overline{\boldsymbol{u}}+\tilde{\boldsymbol{u}};(\overline{\boldsymbol{u}},\overline{p})+F(\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p}),(\overline{\boldsymbol{v}},\overline{q}))=\boldsymbol{f}(\boldsymbol{v})\ , \quad \forall(\overline{\boldsymbol{v}},\overline{q})\in\overline{V}\times\overline{Q}\ . \tag{2.27}$$

Here, Equation (2.27) depends only on the solution on the resolvable scale $(\overline{\boldsymbol{u}},\overline{p})$, whereas it also represents the governing equation for the full scales. The turbulence modeling is therefore "closed", because $(\overline{\boldsymbol{u}},\overline{p})$ is obtained by solving Equation (2.27), and $(\tilde{\boldsymbol{u}},\tilde{p})$ depends on the large-scale solution. The goal of the two-scale VMS is to model the function $F(\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p}))$ and solve the complete system on the resolvable scale at once. Besides, $F(\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p}))$ does not have to rely on any physical hypothesis.

Two-Scale Residual-Based VMS Formulation

The two-scale residual-based VMS method [20], which is employed in this work, can be viewed as a generalization of the stabilized finite element method [166, 164].

Modeling $(\tilde{\boldsymbol{u}},\tilde{p})$ is in fact the key point for the two-scale VMS (Equation (2.27)). One can make use of the perturbation series of a quantity which is based on the residual, i.e. $\varepsilon:=\|\boldsymbol{r}(\overline{\boldsymbol{u}},\overline{p})\|_{\tilde{V}^*\times\tilde{Q}^*}$. The perturbation series yield as:

$$(\tilde{\boldsymbol{u}},\tilde{p})=\varepsilon(\tilde{\boldsymbol{u}}_1,\tilde{p}_1)+\varepsilon^2(\tilde{\boldsymbol{u}}_2,\tilde{p}_2)+\cdots=\sum_{i=1}^{\infty}\varepsilon^i(\tilde{\boldsymbol{u}}_i,\tilde{p}_i)\ . \tag{2.28}$$

It is obvious that if the resolvable space becomes larger, ε gets smaller as $(\overline{\boldsymbol{u}},\overline{p})$ approximates better to the real solution $(\boldsymbol{u},p)$.

In [20], the authors propose to truncate Equation (2.28) after the first order, which means:

$$(\tilde{\boldsymbol{u}}, \tilde{p}) \approx \varepsilon(\tilde{\boldsymbol{u}}_1, \tilde{p}_1) \ , \tag{2.29}$$

combining it with a so-called fine-scale Green's operator [94], $(\tilde{\boldsymbol{u}}_1, \tilde{p}_1)$ can be approximated via an explicit formulation:

$$(\tilde{\boldsymbol{u}}_1, \tilde{p}_1) \approx \boldsymbol{\tau\prime} \frac{\boldsymbol{r}(\overline{\boldsymbol{u}}, \overline{p})}{\varepsilon} \ , \tag{2.30}$$

Furthermore, Equation (2.30) can be rewritten as:

$$(\tilde{\boldsymbol{u}}, \tilde{p}) \approx -\boldsymbol{\tau}\boldsymbol{r}(\overline{\boldsymbol{u}}, \overline{p}) \ . \tag{2.31}$$

Here, $\boldsymbol{\tau}$ is a tensor-valued function, which is defined as:

$$\boldsymbol{\tau} := \begin{pmatrix} \tau_M \mathbb{1}_d & \mathbf{0}_d \\ \mathbf{0}_d^T & \tau_C \end{pmatrix} \ , \tag{2.32}$$

d is the dimension of the spatial space. We take advantage of the finite element space to represent the resolvable space as both of them are finite. If V^h and Q^h are the finite element spaces obtained from the triangulation $\mathcal{T}_h$ with a cell size h, then we set then $\overline{V} = V^h$, $\overline{Q} = Q^h$. The approximation of $(\tilde{\boldsymbol{u}}, \tilde{p})$ is explicitly expressed as:

$$(\tilde{\boldsymbol{u}}, \tilde{p}) \simeq \begin{pmatrix} -\boldsymbol{\tau}_M(\frac{\boldsymbol{\partial u_h}}{\boldsymbol{\partial t}} + \boldsymbol{u}_h \cdot \nabla \boldsymbol{u}_h - \frac{\mu}{\rho}\Delta \boldsymbol{u}_h + \frac{1}{\rho}\nabla p_h - \boldsymbol{f}_h) \\ -\tau_C(\nabla \cdot \boldsymbol{u}_h) \end{pmatrix} = \begin{pmatrix} -\boldsymbol{\tau}_M \boldsymbol{r}_{M,h} \\ -\tau_C r_{C,h} \end{pmatrix} \ , \tag{2.33}$$

where, $\boldsymbol{u}_h$ and p_h are the discrete solutions of the velocity and the pressure, respectively.

We can now state the variational formulation of the residual-based VMS for the incompressible Navier-Stokes equations. Let us take $V := H_0^1(\mathcal{D})^d$, $Q := L^2(\mathcal{D})$ [153]. Further let $V^h \subset V$ and $Q^h \subset Q$ be the finite element spaces.

Find $\boldsymbol{u}_h \in V^h$, $p_h \in Q^h$, such that:

$$\begin{aligned} &(\frac{\partial \boldsymbol{u}_h}{\partial t}, \boldsymbol{v}_h) + (\boldsymbol{u}_h \cdot \nabla \boldsymbol{u}_h, \boldsymbol{v}_h) && \quad (2.34a)\\ &+\frac{\mu}{\rho}(\nabla \boldsymbol{u}_h, \nabla \boldsymbol{v}_h) - \frac{1}{\rho}(p_h, \nabla \cdot \boldsymbol{v}_h) \\ &+(\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \boldsymbol{u}_h \cdot \nabla \boldsymbol{v}_h) + (\tau_C r_{C,h}, \nabla \cdot \boldsymbol{v}_h) \\ &-(\boldsymbol{\tau}_M \boldsymbol{r}_{M,h} \cdot \nabla \boldsymbol{u}_h, \boldsymbol{v}_h) - (\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \boldsymbol{\tau}_M \boldsymbol{r}_{M,h} \cdot \nabla \boldsymbol{v}_h) = (\boldsymbol{f}_h, \boldsymbol{v}_h) \ , && \text{in } [0,T] \times \mathcal{D} \ , \\ &(\nabla \cdot \boldsymbol{u}_h, q_h) + (\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \nabla q_h) = 0 \ , && \text{in } [0,T] \times \mathcal{D} \ . \quad (2.34b) \end{aligned}$$

$\forall(\boldsymbol{v}_h, q_h) \in V^h \times Q^h$. Note that, $(\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \boldsymbol{u}_h \cdot \nabla \boldsymbol{v}_h)$, $(\boldsymbol{\tau}_M \boldsymbol{r}_{M,h} \cdot \nabla \boldsymbol{u}_h, \boldsymbol{v}_h)$ are again the cross-stress terms, $(\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \boldsymbol{\tau}_M \boldsymbol{r}_{M,h} \cdot \nabla \boldsymbol{v}_h)$ is the Reynolds-stress term. If we omit the last two terms from Equation (2.34a) on the left-hand side, the formulation is identical to the classical stabilization method (Equation (2.15)). One can consider that the VMS is a generalization of the stabilization scheme with the cross-stress terms and the Reynolds stress term in addition.

Following the suggestion in [113], we rearrange Equation (2.34) with an encoded convective velocity $\hat{\boldsymbol{u}}_h$, $\hat{\boldsymbol{u}}_h := \boldsymbol{u}_h - \boldsymbol{\tau}_M \boldsymbol{r}_{M,h}$. Equation (2.34) becomes:

$$(\frac{\partial \boldsymbol{u}_h}{\partial t}, \boldsymbol{v}_h) + (\hat{\boldsymbol{u}}_h \cdot \nabla \boldsymbol{u}_h, \boldsymbol{v}_h) \tag{2.35a}$$

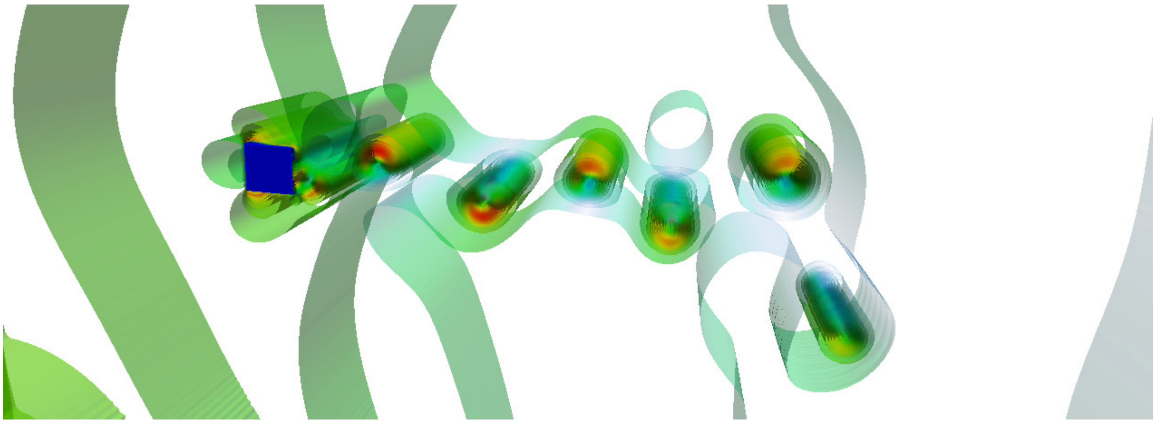


Figure 2.3: Show case: the pressure isosurface colored by the velocity magnitude for a flow around a rectangular cylinder. The computation is accomplished by using the Variational Multiscale method with 52 Millions degrees of freedom, and the Reynolds number $Re = 390,000$.

$$
\begin{aligned}
&+\frac{\mu}{\rho}(\nabla \boldsymbol{u}_h, \nabla \boldsymbol{v}_h) - \frac{1}{\rho}(p_h, \nabla \cdot \boldsymbol{v}_h) & \\
&+(\boldsymbol{\tau_M} \boldsymbol{r}_{M,h}, \hat{\boldsymbol{u}}_h \cdot \nabla \boldsymbol{v}_h) + (\tau_C r_{C,h}, \nabla \cdot \boldsymbol{v}_h) = (\boldsymbol{f}_h, \boldsymbol{v}_h) \;, \qquad & \text{in } [0,T] \times \mathcal{D} \;, \\
&(\nabla \cdot \boldsymbol{u}_h, q_h) + (\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \nabla q_h) = 0 \;, & \text{in } [0,T] \times \mathcal{D} \;. \qquad (2.35b)
\end{aligned}
$$

According to [113], the encoded convective velocity can produce a better approximation for the eddy vorticity. Therefore, we use this version in the following work. The explicit formulation of $\boldsymbol{\tau}$ can be computed based on the scaling arguments [17]. However, we focus on the stabilized Variational Multiscale method in this work. The explicit expression of $\boldsymbol{\tau}$ can be found in [94, 20].

2.3 Rotating Device Modeling

Modeling the transient flow in a rotating device is a challenging task. There are several ways to achieve that. However, they all belong to the moving mesh techniques, which means that we need to obtain a deformed computing domain, and at the same time the continuity of the flow must be preserved. The main approaches can be divided into three classes: the mesh patching, the mesh embedding and the mesh adapting [25].

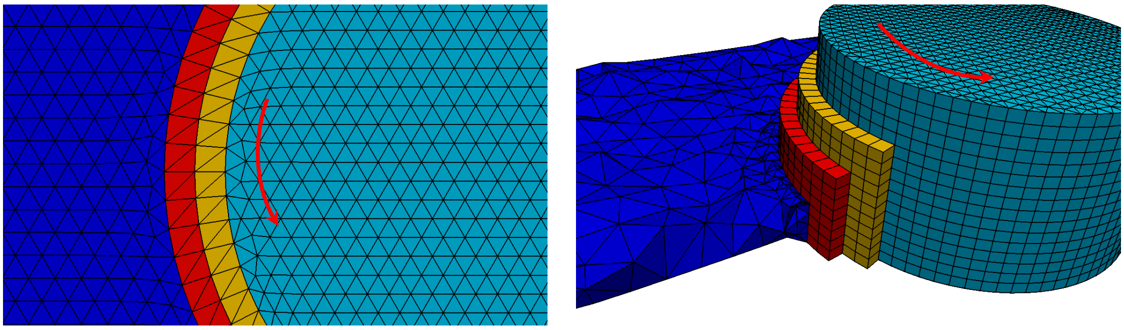
The mesh patching technique brings into play the common interface between the subdivisions, and it requires the shared boundaries to be continuous when the flow passes through. The Sliding Mesh algorithm [159] is one of the most applied methods, especially in industry. It relies on the interpolation and projection [54] procedure in order to ensure the continuity of the flow between disjointed subdomains. Besides, the subdomains can be even adjacent in case that an arbitrary movement is needed. The Shear-Slip Mesh Update Method (SSMUM) [23] focuses more on the revolving movement. It recommends constructing a regular layer between the stationary domain and the transient domain. Only the layer is deformed in order to acquire the independence of rotation and sustain the connectivity between meshes. The deformation in the layer is accomplished while a maximum level is achieved, the shared interface is then disconnected and reconnected again to the subsequent grids. The mesh patching technique has a good efficiency in terms of the computation if the movement is regular, all subdomains can be computed simultaneously once the interface condition is defined.

The mesh embedding technique requires an overlapped layer instead of a shared boundary, this overlapping area is supposed to assist the construction of the solution mapping. The Overset/Chimera method [107, 158, 86] belongs to this category, and it requires a background mesh and a sub-mesh, the sub-mesh needs to accommodate the moving object. Except the overlapping area, the embedded mesh is cut after the sub-mesh changes, a supplementary interpolation between the neighboring domains maintains the continuity of the flow.

The mesh adapting technique tries to evolve the mesh by utilizing the adaptive algorithm [89]. The advantage of this method is to employ the initial mesh rather than reconstruct a new mesh, and the quality of the solution can be improved by adjusting the mesh locally.

We give here two different approaches, which we studied in this work, for modeling the rotating effect: The shear layer update approach, in which we suggest having two layers instead of one layer as in SSMUM in favor of accelerating the update process. The combination of Continuous and Discontinuous Galerkin approach proposes to utilize the continuous Galerkin method (CG) on the whole grid, except the interface between the adjacent meshes, where the discontinuous Galerkin method (DG) ensures the necessary conditions for the continuity of the flow.

2.3.1 Shear Layer Update Approach



(a) 2D.

(b) 3D.

Figure 2.4: Illustration of the mesh for the shear layer update approach. Two corresponding layers (yellow and red) are identical in terms of the grid shape, the dark blue and red parts belong to the static domain, the light blue and yellow parts belong to the rotating domain.

The shear layer update approach can be characterized in the group of the mesh patching technique, and it is mainly inherited from the Shear-Slip Mesh Update Method (SSMUM). The main difference is that we build two identical layers by means of the grid shape instead of one layer (Figure 2.4). Under this situation, the mirror layers can expedite the update process in case that a mesh regeneration is desired.

For the mesh patching technique and the mesh embedding technique, there are always some parts of the mesh which are in the immovable position, and the other parts of the mesh are changing their location over time (the global computing domain is separated naturally into the different subdivisions). In our method, we also consider two main subdivisions (dark blue and light blue in Figure 2.4), one of them is supposed to be fixed, and another one, which accommodates the rotating object, rotates under certain specifications. Besides, it can be more than just one rotating object, it means that we have to build more subdivisions in order to hold the objects, but they all belong to the rotating mesh. Between those two domains, we require two additional mesh layers (yellow and red in Figure 2.4), which have the identical shape with respect to the mesh cell. Eventually, the dark blue and red parts are categorized as the static domain, and the rest belongs to the rotating domain. All those four subdivisions build up the global mesh, this mesh is conforming everywhere thanks to the coincided interfaces.

The parallelization of a finite element computation is built on the mesh decomposition. At the beginning of the decomposition process, one processor holds all the information of the global geometry, the global mesh is decomposed and distributed to each processor by this "master"

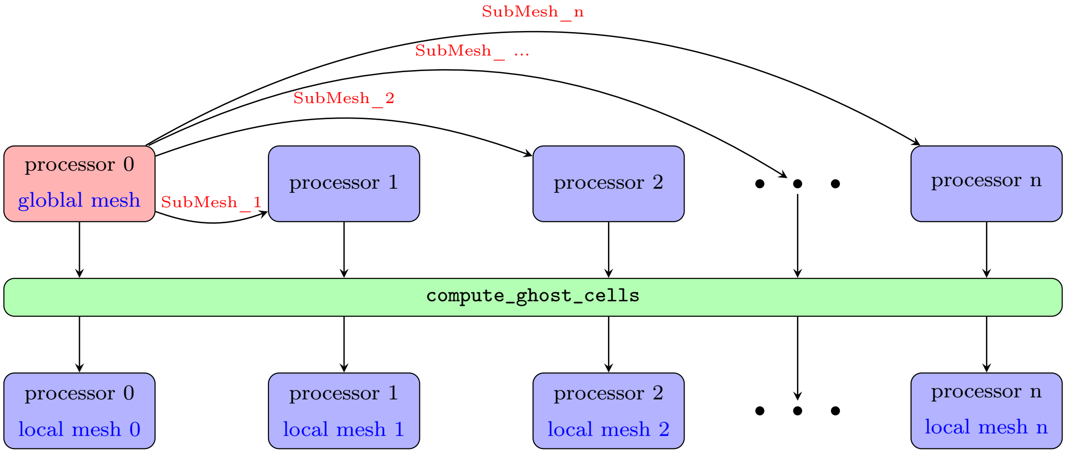


Figure 2.5: The common procedure of mesh decomposition. At the beginning, processor 0 (red), i.e. the "master" processor, handles the global mesh, then it distributes the sub-meshes to each processor. After computing ghost cells, each processor possesses a local mesh, which contains the interior cells (sub-mesh) and the ghost cells.

processor. After each processor receives a decomposed mesh (SubMesh in Figure 2.5), one additional layer will be added on this decomposed mesh. This additional layer is built based on the surrounding decomposed meshes in order to share information between processors. Hence, the final local mesh consists of two parts: one part is completely local, another part is this additional layer. Sometimes, this extra layer is also referred to as "ghost cells" (Figure 2.5).

Many mesh decomposition algorithms rely on the graph theory [100, 99], the main purpose is to achieve a good load balancing between subdomains and a convenient numbering of entities. Still, the premeditated location of the decomposition is usually not taken into account. In other words, the local vector and the local matrix may contain a part of the static domain and another part of the rotating domain simultaneously. However, for the rotating device modeling, the mesh regeneration is frequently required. Because of the modification of the physical location in certain parts of the local meshes, the local vector and the local matrix can be different as before. Here, the regeneration means the relationship between sub-meshes is modified. If just cell vertices move, we rather consider it as a mesh deformation but not a regeneration. A mapping procedure from the old vector space to the current vector space is inevitable in order to advance the computation. For this reason, the local solution vector needs to be projected to the new local vector. However, this operation often involves a physical point search routine, which is computationally expensive.

In order to overcome this barrier of the expensive vector update process, we suggest using two communication groups [18] instead of one for the mesh decomposition (Figure 2.6). Hence, we have two "master" processors, which manage the static mesh and the rotating mesh separately. For that reason, each decomposed mesh consists only of a part of the static mesh or the rotating mesh. After decomposing the mesh inside the groups, the `compute_ghost_cells` process handles all sub-meshes together. In consequence, all added ghost cells on the sub-mesh are in the same category, it means that they come from either the static mesh or the rotating mesh. Except those which have the shared interface between the static and rotating mesh, their ghost cells come from both domains (Figure 2.7). In general, the local vector also contains two parts: the interior cells part and the ghost cells part. They both are contributed by interior cells and ghost cells (Figure 2.6), respectively. The advantage of our method is that the local vector keeps the same structure after the mesh regeneration. The value of the interior cells part stays unchanged, only those of the ghost cells part need to be modified. However, the ghost cells part can be updated by using the `update_ghost` function, which is commonly available for the most of finite element implementations.

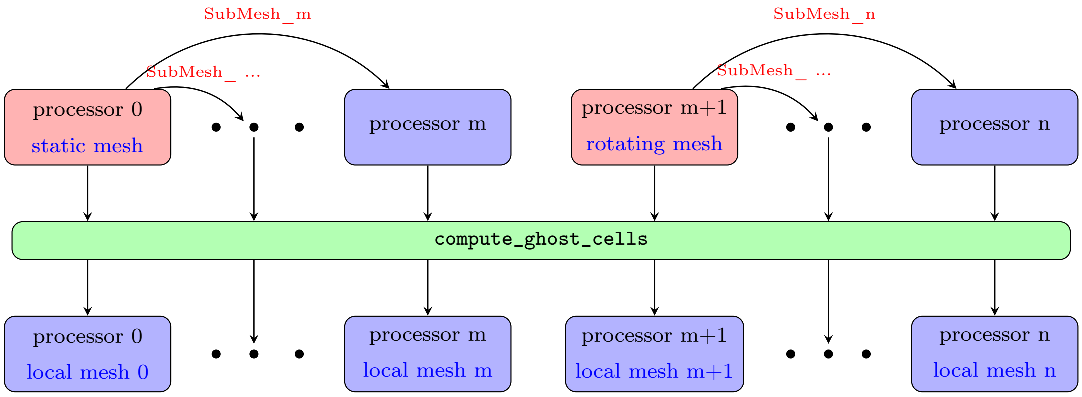


Figure 2.6: The mesh decomposition procedure for the shear layer update approach. At the beginning, processor 0 (red) and m+1 (red), which are our "master" processors, handle the static mesh and the rotating mesh separately. They distribute sub-meshes to each processor within their own communication group. But the `compute_ghost_cells` function performs within all processors, namely a global communication. In the end, each processor possesses a local mesh, which contains the interior cells (sub-mesh) and ghost cells.

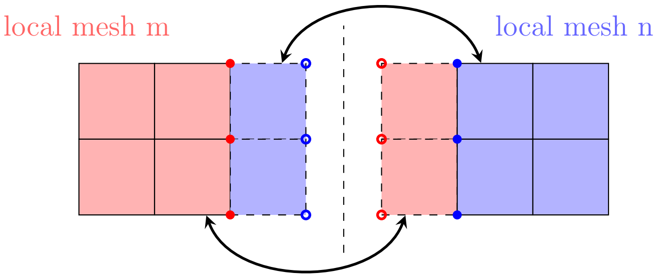


Figure 2.7: Ghost cells for two local meshes, which are located on the interface between the static and rotating mesh. Local mesh m has its own interior cells (red left) and ghost cells (blue left), which comes from the local mesh n, vice versa for the local mesh n.

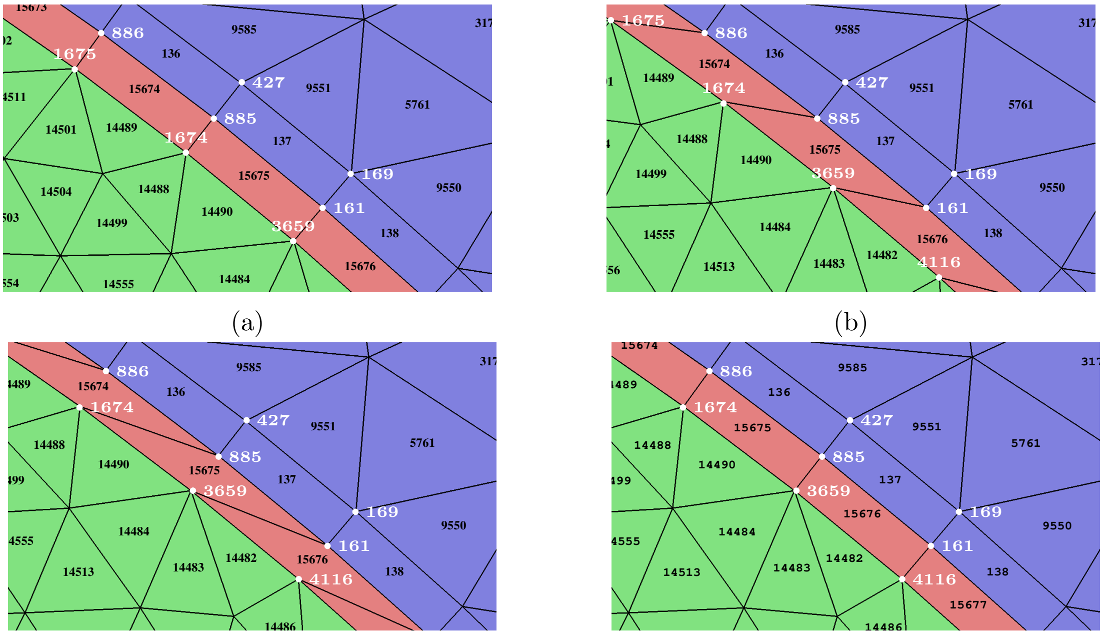
On the other hand, since they are two geometrically identical layers (Figure 2.4), the revolution of the rotating mesh is limited by the number of cells on the perimeter of the interface. Although, changing the size of the time step can give us some flexibilities, it is not optimal for complex simulations. Thence, we propose the similar technique as in SSMUM [23], we shear one of these two layers during the computation.

Shear Layer Procedure

We demonstrate the shear layer procedure with the help of Figure 2.8:

(a): The initial position. The red layer allows a shearing effect, and the green part can rotate, both of them belong to the rotating mesh. The blue part is the static mesh. e.g., cell 15675 and 137 are both ghost cells for each other, they have a shared interface in common. Cell 15675 consists of the vertices 3659, 161, 885 and 1674.

(b): Shearing process. Deforming the layer (red) regarding a revolving speed of the rotating part. The vertices on the shared interface stay unaltered. In this case, the global mesh is just deformed, but no new mesh generation is required. Therefore, we do not have to change the local vector, only the mesh's geometry needs to be updated.



(a) (b) (c) (d)

Figure 2.8: Illustration of the procedure for shear layer update approach.

- **(c)**: Reaching the maximum deformation. The maximal displacement has the same length as the cell on the interface. However, the status of the mesh is similar as in step **(b)**.
- **(d)**: Disconnect and reconnect. The vertices on the shared interface disconnect from the original location, then they reconnect to the upstream vertices. Whence, certain cells' connectivity is modified: e.g. cell 15675 is constructed now by vertices 1675, 1674, 885 and 886. But there is no deformation within the static mesh, only the solution on the ghost cells have to be updated.

ALE Formulation of Navier-Stokes Equations for Moving Mesh Modeling

After presenting the main procedure of our shear layer update approach, we also have to show the needed modification in the governing equations. The formulation of the moving mesh problem can be defined with the Arbitrary Lagrange-Euler (ALE) method [46, 87]. We briefly show the modification by using the incompressible Navier-Stokes equations, but the principle also applies to other systems. The strong formulation of the incompressible NSE is given as:

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} - \boldsymbol{u}^r) \cdot \nabla \boldsymbol{u} - \frac{\mu}{\rho}\Delta \boldsymbol{u} + \frac{1}{\rho}\nabla p = \boldsymbol{f} \;, \qquad \text{in } \mathcal{D}^t \times [0,T] \;, \tag{2.36a}$$

$$\nabla \cdot \boldsymbol{u} = 0 \;, \qquad \text{in } \mathcal{D}^t \times [0,T] \;, \tag{2.36b}$$

$$\boldsymbol{u^r} = \boldsymbol{d} \times \boldsymbol{\omega} \;, \qquad \text{in } \mathcal{D}^t_{rot} \times [0,T] \;, \tag{2.36c}$$

$$\boldsymbol{u^r} = \boldsymbol{0} \;, \qquad \text{in } \mathcal{D}^t_{stat} \times [0,T] \;. \tag{2.36d}$$

Here, $\boldsymbol{u}$ is the velocity, and p is the pressure. μ and ρ are the dynamic viscosity and the density, respectively. $\mathcal{D}^t_{rot}$ is the rotating domain, and $\mathcal{D}^t_{stat}$ is the static domain, $\mathcal{D}^t = \mathcal{D}^t_{rot} \cup \mathcal{D}^t_{stat}$, $\mathcal{D}^t_{rot} \cap \mathcal{D}^t_{in} = \emptyset$. The superscript t indicates that the mesh is evolving with respect to time t, $t \in [0,T]$. $\boldsymbol{u}^r$ is the evolution velocity of the mesh, it means the rotating domain revolves with the same velocity as the accommodated object. $\boldsymbol{d}$ is the distance from a physical point in $\mathcal{D}^t_{rot}$ to the rotation axis and $\boldsymbol{\omega}$ is the angular speed.

$\boldsymbol{u^r}$ plays actually the extra convective effects on the momentum conservation equation (Equation (2.36a)), it represents the additional relative motion between the flow particles and the computational grid of the reference domain due to the mesh rotation.

One remark of the practical aspect for the finite element implementation is, when we need to assemble the matrix and the vector in the shear layer, the material velocity $\boldsymbol{u^r}$ does not depend on the distance $\boldsymbol{d}$ anymore (Equation (2.36c)). The angular speed $\boldsymbol{\omega}$ is not constant with respect to the quadrature points due to the shearing.

Thence, the variational formulation with the Variational Multiscale method can be written as:

Find $\boldsymbol{u}_h \in V^h$, $p_h \in Q^h$, such that:

$$\begin{aligned}\int_{\mathcal{D}^t} (\frac{\partial \boldsymbol{u}_h}{\partial t} + (\hat{\boldsymbol{u}}_h - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h) \cdot \boldsymbol{v}_h \, d\boldsymbol{x} + \int_{\mathcal{D}^t} \frac{\mu}{\rho} \nabla \boldsymbol{u}_h : \nabla \boldsymbol{v}_h \, d\boldsymbol{x} \\ + \int_{\mathcal{D}^t} \frac{1}{\rho} p_h \nabla \cdot \boldsymbol{u}_h \, d\boldsymbol{x} - \int_{\mathcal{D}^t} \boldsymbol{f}_h \cdot \boldsymbol{v}_h \, d\boldsymbol{x} \\ + \int_{\mathcal{D}^t} [\frac{\partial \boldsymbol{u}_h}{\partial t} + (\boldsymbol{u}_h - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h - \frac{\mu}{\rho} \Delta \boldsymbol{u}_h + \frac{1}{\rho} \nabla p_h - \boldsymbol{f}_h] \cdot \boldsymbol{\tau}_M (\hat{\boldsymbol{u}}_h - \boldsymbol{u}^r) \cdot \nabla \boldsymbol{v}_h \, d\boldsymbol{x} \\ + \int_{\mathcal{D}^t} \tau_C \nabla \cdot \boldsymbol{u}_h \cdot \nabla \cdot \boldsymbol{v}_h \, d\boldsymbol{x} = \boldsymbol{0} \,,\end{aligned} \tag{2.37a}$$

$$\begin{aligned}\int_{\mathcal{D}^t} q_h \nabla \cdot \boldsymbol{u}_h \, d\boldsymbol{x} \\ + \int_{\mathcal{D}^t} \boldsymbol{\tau}_M \nabla q_h \cdot [\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u}_h - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h - \frac{\mu}{\rho} \Delta \boldsymbol{u}_h + \frac{1}{\rho} \nabla p_h - \boldsymbol{f}_h] \, d\boldsymbol{x} = 0 \,.\end{aligned} \tag{2.37b}$$

$\forall \boldsymbol{v}_h \in V^h$, $\forall q_h \in Q^h$.

2.3.2 Combination of Continuous and Discontinuous Galerkin Approach

The shear layer update approach we proposed in the previous subsection focuses on the continuous Galerkin (CG) method [182]. The advantage is that there are already well established theories for flow problems, and a large amount of efficient parallel solving strategies are available. However, the static mesh and the rotating mesh should constantly keep conformed, therefore recalculating the ghost cells and updating the solutions are inevitable. Thus, we use the continuous Galerkin method and the discontinuous Galerkin (DG) method together in order to avoid this inconvenience.

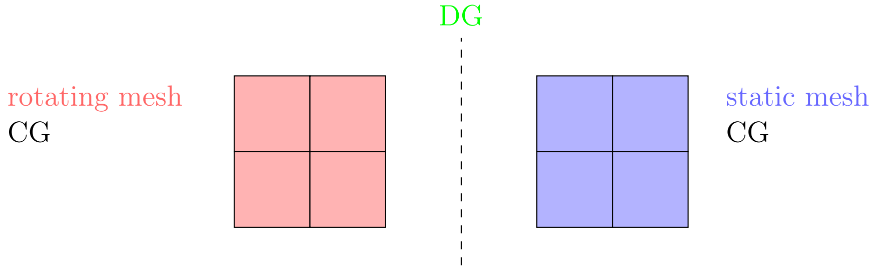


Figure 2.9: The static mesh and the rotating mesh are computed with the Continuous Galerkin method, only for the computation of the shared interface, we used the discontinuous Galerkin method.

The main concept is to consider the DG method on the interface between the static and rotating mesh. Within both domains, the flow problem is still modeled by using the CG method

(Figure 2.9). Yet, the degrees of freedoms (DOFs) on the shared interface are duplicated comparing to the shear layer update approach, the communication between two subdomains are linked by specific jump terms [88]. Therefore, we need to append the following terms to the weak formulation (Equation (2.37)):

$$-\int_{\Gamma^t_{interface}} t(\boldsymbol{u}_{stat}, \boldsymbol{u}_{rot}, p_{stat}, p_{rot}) \cdot (\boldsymbol{v}_{stat} - \boldsymbol{v}_{rot})\, d\boldsymbol{x} \tag{2.38a}$$

$$-\int_{\Gamma^t_{interface}} (\boldsymbol{u}_{stat} - \boldsymbol{u}_{rot}) \cdot \tilde{t}(\boldsymbol{v}_{stat}, \boldsymbol{v}_{rot}, q_{stat}, q_{rot})\, d\boldsymbol{x} \tag{2.38b}$$

$$+\int_{\Gamma^t_{interface}} (\boldsymbol{u}_{stat} - \boldsymbol{u}_{rot}) \cdot \tau_B (\boldsymbol{v}_{stat} - \boldsymbol{v}_{rot})\, d\boldsymbol{x}\ . \tag{2.38c}$$

$\Gamma^t_{interface} = \mathcal{D}^t_{stat} \cap \mathcal{D}^t_{rot}$ denotes the shared interface between the static domain and the rotating domain. $\boldsymbol{v}_{stat}$ and $\boldsymbol{v}_{rot}$ represent the test functions on $\Gamma^t_{interface}$ from the side of $\mathcal{D}^t_{stat}$ and $\mathcal{D}^t_{rot}$, they have the identical shape functions. However, all those basis functions are duplicated by the virtue of the concept of the discontinuous Galerkin method and belong to different subdomains. The rest of the parameters in Equation (2.38) are defined as follows:

$$\begin{aligned} t(\boldsymbol{u}_{stat}, \boldsymbol{u}_{rot}, p_{stat}, p_{rot}) &= \frac{1}{2}(-\frac{1}{\rho} p_{stat} \boldsymbol{n}_{stat} + \frac{\mu}{\rho}(\nabla \boldsymbol{u}_{stat} \boldsymbol{n}_{stat} + (\nabla \boldsymbol{u}_{stat})^T \boldsymbol{n}_{stat}) \\ &\quad + \frac{1}{\rho} p_{rot} \boldsymbol{n}_{rot} - \frac{\mu}{\rho}(\nabla \boldsymbol{u}_{rot} \boldsymbol{n}_{rot} + (\nabla \boldsymbol{u}_{rot})^T \boldsymbol{n}_{rot}))\ , \end{aligned} \tag{2.39a}$$

$$\begin{aligned} \tilde{t}(\boldsymbol{v}_{stat}, \boldsymbol{v}_{rot}, q_{stat}, q_{rot}) &= \frac{1}{2}(\frac{1}{\rho} q_{stat} \boldsymbol{n}_{stat} + \frac{\mu}{\rho} \nabla \boldsymbol{v}_{stat} \boldsymbol{n}_{stat} - \frac{1}{\rho} q_{rot} \boldsymbol{n}_{rot} - \frac{\mu}{\rho} \nabla v_{rot} \boldsymbol{n}_{rot}) \\ &\quad + (\boldsymbol{v}_{stat} - \boldsymbol{v}_{rot})(\{\boldsymbol{u}_{stat} \cdot \boldsymbol{n}_{stat}\}\_ + \{\boldsymbol{u}_{rot} \cdot \boldsymbol{n}_{rot}\}\_)\ , \end{aligned} \tag{2.39b}$$

$$\tau_B = \frac{1}{2}(\frac{C_{stat}\mu}{h_{stat}\rho} + \frac{C_{rot}\mu}{h_{rot}\rho})\ . \tag{2.39c}$$

$\boldsymbol{n}_{stat}$, $\boldsymbol{n}_{rot}$ are the normal vectors on the static side and the rotating side. $\{A\}\_$ indicates the negative part of A, i.e., $\{A\}\_ = A, if\ A < 0$, and $\{A\}\_ = 0, if\ A \geq 0$. h_{stat}, h_{rot} denote the length of the cells, C_{stat}, C_{rot} are positive constants.

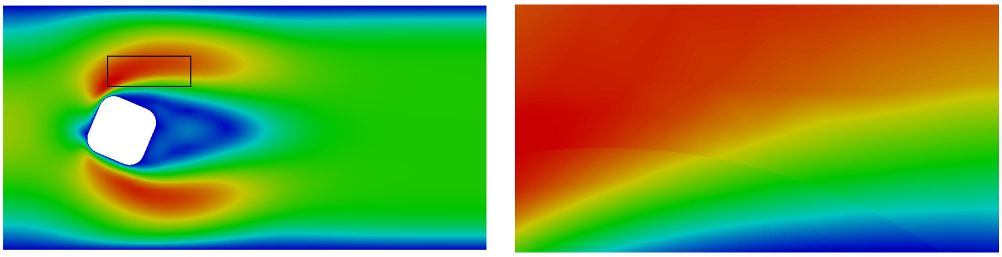


Figure 2.10: Show case: 2D channel benchmark using the combination of Continuous and Discontinuous Galerkin approach for a rotating rectangle. The second picture shows a discontinuity on the shared interface.

However, our final computation is accomplished based on the shear layer update approach instead of the combination of Continuous and Discontinuous Galerkin approach, because the additional interface conditions from the latter model need to be treated in a different way compared to the continuous Galerkin part. Otherwise, our developed flow solver is not efficient. Therefore, we decide to adhere to the shear layer update approach in the following work.

2.3.3 Pyramid Element

One arising issue for the shear layer update approach is the mesh generation, especially when we have a complex geometry in 3D. Generating two layers, whose cells are equally sized and shaped is a nontrivial task. Meanwhile, the orientation of the cells should also be suitable for the shearing. In general, we expect a situation as in Figure 2.4 that the cells of the layers are hexahedrons in order to handle the shearing. It implies that the other two domains should have one surface with quadrilateral meshes. However, as the rotating domain accommodates the moving object which holds a complex shape, an unstructured mesh is preferable.

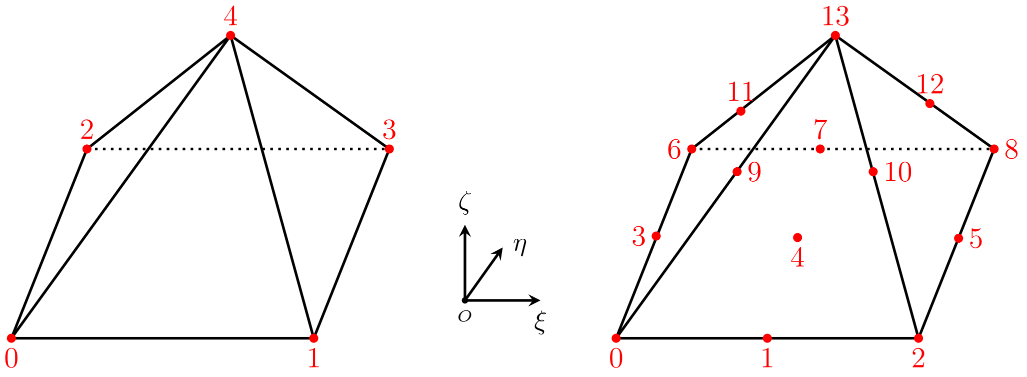


Figure 2.11: The location and the numbering of degrees of freedom. The left one is the first order element, the right one is the second order element. For the first order case, the DOF numbering also overlaps the numbering of vertices.

Therefore, we would like to have structured meshes for the layers, it implies that we need to utilize an adequate "glue element" for connecting structured and unstructured meshes. There exist two kinds of standard elements, which are suitable for this situation: the pyramid element and the prism element. In our hybrid mesh, we choose the pyramid element as the "glue element".

The construction of shape functions for a pyramid element is unfortunately ambiguous, in addition, maintaining the conforming feature of the cell is not trivial. It is proven that it is impossible to obtain conforming continuously differentiable basis functions on a pyramid element [172, 115]. We state the theorem as follows:

Theorem 2.3.1 (Pyramid shape function construction)**.** *There exists no continuously differentiable conforming shape functions for the pyramid element, which is linear on triangle facet and bilinear on rectangle facet.*

The detailed proof can be found in [172, 116, 115]

Therefore, we can not construct the quadratic shape functions on a whole pyramid element, we have to use composite elements instead. In our implementation, we divide a pyramid by four equal-sized tetrahedrons. Figure 2.11 shows the location of the degrees of freedom for the first and the second order pyramid element. We state here only the first order shape functions, the higher order functions are shown in [115, 172, 116, 38], and the Gaussian quadrature points for the numerical integration can be found in [108, 39].

$$\varphi_0 = \begin{cases} -\frac{1}{2}\zeta + (\frac{1}{4}\xi - \frac{1}{4})(\eta - \zeta - 1)\ , & \xi > \eta \text{ and } \xi < -\eta\ , \\ \frac{1}{4}\zeta(\xi + \eta - 2) + (\frac{1}{4}\xi - \frac{1}{4})(\eta - \zeta - 1)\ , & \xi \geq \eta \text{ and } \xi \geq -\eta\ , \\ (\frac{1}{4}\xi - \frac{1}{4})(\eta + \zeta - 1)\ , & \xi < \eta \text{ and } \xi > -\eta\ , \\ \frac{1}{4}\zeta(\xi - \eta - 2) + (\frac{1}{4}\xi - \frac{1}{4})(\eta - \zeta - 1)\ , & \xi \leq \eta \text{ and } \xi \leq -\eta\ , \end{cases} \tag{2.40a}$$

$$\varphi_1 = \begin{cases} -\frac{1}{2}\zeta + (\frac{1}{4}\xi + \frac{1}{4})(-\eta + \zeta - 1)\ , & \xi > \eta \,\text{and}\, \xi < -\eta\ , \\ -\frac{1}{4}\zeta(\xi + \eta + 2) + (\frac{1}{4}\xi + \frac{1}{4})(-\eta + \zeta + 1)\ , & \xi \geq \eta \,\text{and}\, \xi \geq -\eta\ , \\ (\frac{1}{4}\xi + \frac{1}{4})(-\eta - \zeta + 1)\ , & \xi < \eta \,\text{and}\, \xi > -\eta\ , \\ -\frac{1}{4}\zeta(\xi - \eta + 2) + (\frac{1}{4}\xi + \frac{1}{4})(-\eta + \zeta + 1)\ , & \xi \leq \eta \,\text{and}\, \xi \leq -\eta\ , \end{cases} \tag{2.40b}$$

$$\varphi_2 = \begin{cases} (-\frac{1}{4}\xi + \frac{1}{4})(\eta - \zeta + 1)\ , & \xi > \eta \,\text{and}\, \xi < -\eta\ , \\ -\frac{1}{4}\zeta(\xi + \eta) + (-\frac{1}{4}\xi + \frac{1}{4})(\eta - \zeta + 1)\ , & \xi \geq \eta \,\text{and}\, \xi \geq -\eta\ , \\ -\frac{1}{2}\xi\zeta(-\frac{1}{4}\xi + \frac{1}{4})(\eta - \zeta + 1)\ , & \xi < \eta \,\text{and}\, \xi > -\eta\ , \\ -\frac{1}{4}\zeta(\xi - \zeta) + (-\frac{1}{4}\xi + \frac{1}{4})(\eta - \zeta + 1)\ , & \xi \leq \eta \,\text{and}\, \xi \leq -\eta\ , \end{cases} \tag{2.40c}$$

$$\varphi_3 = \begin{cases} (\frac{1}{4}\xi + \frac{1}{4})(\eta - \zeta + 1)\ , & \xi > \eta \,\text{and}\, \xi < -\eta\ , \\ \frac{1}{4}\zeta(\xi + \eta) + (\frac{1}{4}\xi + \frac{1}{4})(\eta - \zeta + 1)\ , & \xi \geq \eta \,\text{and}\, \xi \geq -\eta\ , \\ \frac{1}{2}\xi\zeta(\frac{1}{4}\xi + \frac{1}{4})(\eta - \zeta + 1)\ , & \xi < \eta \,\text{and}\, \xi > -\eta\ , \\ \frac{1}{4}\zeta(\xi - \eta) + (\frac{1}{4}\xi + \frac{1}{4})(\eta - \zeta + 1)\ , & \xi \leq \eta \,\text{and}\, \xi \leq -\eta\ , \end{cases} \tag{2.40d}$$

$$\varphi_4 = \zeta\ . \tag{2.40e}$$

Chapter 3

Stochastic-Spectral Finite Element Method for the Incompressible Navier-Stokes Equations with the Variational Multiscale Method

The aforementioned chapter focused on the modelization of high Reynolds number and the mesh techniques for a rotating system. Within this chapter, we show how to apply the Variational Multiscale method for the blood pump modeling. After defining a proper model for the rotating machinery simulation, we investigate the influence of the uncertain input data on the numerical solution by considering the blood pump as a stochastic dynamic system. As mentioned already in Chapter 1, we quantify the uncertainties in the numerical model by means of the generalized Polynomial Chaos expansion (gPCE). We use the intrusive UQ technique in the blood pump modelization, which is further detailed in this chapter.

This chapter begins with the basic knowledge in probability theory and partial differential equations with the random input data. Afterward, the generalized Polynomial Chaos expansion and the standard procedure of constructing an intrusive Polynomial Chaos (PC) formulation is presented. Moreover, with the help of the abstract representation of the weak form of the stochastic problem, the structure of the specific stochastic system is also discussed. The Galerkin tensor plays a key role in building the PC formulation and is also essential for developing the preconditioners, which will be presented in Chapter 4. The chapter ends with the spectral-stochastic finite element method, more specifically, the discretized stochastic Variational Multiscale method formulation of the blood handling device is given.

3.1 Uncertainty Quantification

Over the last decade, Uncertainty Quantification has gained a great deal of attention in the field of engineering and applied mathematics. The goal of UQ is to explore the influence on the computational results from uncertainties in the input data or in the models. This uncertainty study may subsequently be used to provide more reliable information and or quantify the uncertainties within the systems. In this work, we investigate uncertainties in the fluid problem associated with the blood pump, which is formulated as a stochastic partial differential equation (SPDE). Therefore, we first introduce some basic needed theories for the derivation of the overall model.

3.1.1 Probability Theory

In the following section, we give an overview of the necessary definitions in probability theory. This overview is an excerpt from [118, 109] with the definitions of the terms and concepts.

Definition 3.1.1 (σ-algebra). *Let Θ be a set. A family $\mathcal{F}$ of subsets of Θ is a σ-algebra on Θ, if and only if it satisfies the following conditions:*

1. $\emptyset \in \mathcal{F}$.
2. $A \in \mathcal{F} \implies A^c \in \mathcal{F}$.
3. $A_i \in \mathcal{F}\ , i \in I\ , I$ *is countable* $\implies \bigcup_i A_i \in \mathcal{F}$.

Definition 3.1.2 (Topological space). *Let Θ be a set, τ is a family of subsets of Θ with the following properties:*

1. *the empty set $\emptyset$ and Θ are both in τ.*
2. *τ is closed under finite intersections.*
3. *τ is closed under arbitrary unions.*

The pair (Θ, τ) is a topological space.

Definition 3.1.3 (Borel σ-algebra). *For a topological space Θ, $\mathcal{B}(\Theta)$ represents the Borel σ-algebra, it equals to the smallest σ-algebra containing all open subsets of Θ.*

Definition 3.1.4 (Measurable space). *Any $A \in \mathcal{F}$, $\mathcal{F}$ is a σ-algebra, is known as a measure set, and the pair $(\Theta, \mathcal{F})$ is known as a measurable space.*

Definition 3.1.5 (Measure). *A measure μ on a measurable space $(\Theta, \mathcal{F})$ is a mapping, $\mu : \mathcal{F} \to [0, +\infty]$, such that:*

$$\mu(\bigcup_{i=0}^{\infty} A_i) = \sum_{i=0}^{\infty} \mu(A_i)\ , \quad \forall A_i \in \mathcal{F}\ , A_i \cap A_j = \emptyset\ , i \neq j\ ,$$

i.e. it is countably additive. A triple $(\Theta, \mathcal{F}, \mu)$ is called a measure space.

Definition 3.1.6 (Measurable set). *Let $(\Theta, \mathcal{F})$ be a measurable space. A subset $A \subseteq \Theta$ is measurable, if and only if $A \in \mathcal{F}$.*

Definition 3.1.7 (Measurable function). *Let $(\Theta, \mathcal{F})$, $(\Psi, \mathcal{T})$ be two measurable spaces. A function $u : \Theta \to \Psi$ is measurable, if:*

$$u^{-1}(E) := \{x \in \Theta | u(x) \in E\} \in \mathcal{F}\ , \quad \forall E \in \mathcal{T}\ .$$

If we focus on the stochastic process, one important measurable space is the probability space.

Definition 3.1.8 (Probability space). *A probability space is an ordered triple $(\Theta, \mathcal{F}, \mathbb{P})$, where $(\Theta, \mathcal{F})$ is a measurable space, $\mathcal{F}$ are the measurable subsets of Θ, $\mathbb{P}$ is a measure over $\mathcal{F}$.*

$\mathbb{P} : \mathcal{F} \to [0, 1]$ *with* $\mathbb{P}(\Theta) = 1$*:*

- *Θ is the sample space, which is a nonempty set.*
- *$\mathcal{F}$ is the event space.*
- *$\mathbb{P}$ is the probability function.*

Lemma 3.1.1 (Properties of the probability function)**.** *Let* $(\Theta, \mathcal{F}, \mathbb{P})$ *be a probability space.* $\forall A, B \in \mathcal{F}$*, then:*

1. $\mathbb{P}(B) \leq \mathbb{P}(A)\ , if B \subseteq A$.
2. $\mathbb{P}(A) = 1 - \mathbb{P}(A^c)$.
3. $\mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \cap B)$.

Definition 3.1.9 (Random variable)**.** *Let* $(\Theta, \mathcal{F}, \mathbb{P})$ *be a probability space and* $(\Psi, \mathcal{G})$ *be a measurable space. A measurable function* $X : (\Theta, \mathcal{F}) \to (\Psi, \mathcal{G})$ *is called* $\mathcal{F}$*-measurable random variable or random variable. The observed value of* $X(\theta)$ *for a given* $\theta \in \Theta$ *is also called a realization of* X*.*

For the most common case, $(\Psi, \mathcal{G}) = (\mathbb{R}, \mathcal{B})$*, with the real number and a Borel* σ*-algebra* $\mathcal{B}(\mathbb{R})$*,* X *is a real-valued random variable. Each random variable* $X : \Theta \to \Psi$ *has an associated probability distribution* $\mathbb{P}_X$*.*

Definition 3.1.10 (Random vector)**.** *The random vector* $\boldsymbol{X}$ *is a vector of random variables:*

$$\boldsymbol{X} = [X_1, ..., X_n]^T \ , \quad n \in \mathbb{N} \ .$$

Definition 3.1.11 (Probability distribution)**.** *Let* X *be a* Ψ*-random variable,* $(\Psi, \mathcal{G})$ *is a measurable space. and* $(\Theta, \mathcal{F}, \mathbb{P})$ *is the corresponding probability space. The probability distribution* $\mathbb{P}_X$ *of* X*,* $\mathbb{P}_X := g \to [0, 1]$*, is the probability measure on* $(\Psi, \mathcal{G})$*:*

$$\mathbb{P}_X(G) := \mathbb{P}(X^{-1}(G))\ , where,\ X^{-1}(G) := \{\theta \in \Theta : X(\theta) \in G\}\ , \quad G \in \mathcal{G}\ .$$

In particular, if $(\Psi, \mathcal{G}) = (\mathbb{R}, \mathcal{B})$*, the probability distribution can be written as:*

$$\mathbb{P}_X(x) := \mathbb{P}(X^{-1}((-\infty, x])) = \mathbb{P}(\{\theta : X(\theta) \leq x\}) = \mathbb{P}(X \leq x)\ .$$

Proposition 3.1.1. *If the distribution function* $\mathbb{P}_X$ *is absolutely continuous with respect to the Lebesgue measure on* $\mathbb{R}$*, there exists an integrable function,* $\rho : \mathbb{R} \to [0, +\infty)$ *such that:*

$$\mathbb{P}_X(b) - \mathbb{P}_X(a) = \int_a^b \rho(x) dx, \quad a < b\ .$$

Definition 3.1.12 (Probability density function (PDF))**.** ρ *is called the probability density function. Notably, if* $\rho(x)$ *is continuous at* x*, the probability density function can be also obtained by:*

$$\rho(x) = \frac{d}{dx}\mathbb{P}_X(x)\ . \tag{3.1}$$

Considering a random variable $X : \Theta \to \mathbb{R}$, defined on a probability space $(\Theta, \mathcal{F}, \mathbb{P})$. Two well-known density functions, Gaussian ($X \sim \mathcal{N}(\mu, \sigma^2)$) and uniform ($X \sim \mathcal{U}(a, b)$), are widely used in engineering and research:

- Gaussian or normal density function:

$$\rho_{\mu,\sigma^2} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}\ . \tag{3.2}$$

- Uniform density function:

$$\rho_{a,b} = \begin{cases} \frac{1}{b-a} & x \in [a, b] \\ 0 & \text{otherwise} \end{cases}\ . \tag{3.3}$$

Definition 3.1.13 (Joint probability distribution and density function)**.** *For a random vector $\boldsymbol{X}$, the joint probability distribution $\mathbb{P}_{\boldsymbol{X}} : \mathbb{R}^d \to [0,1]$ is defined as:*

$$\mathbb{P}_{\boldsymbol{X}}(\boldsymbol{x}) := \mathbb{P}(\bigcap_{i=1}^{d}(X_i \leq x_i)) \ ,$$

where, $\boldsymbol{x} = (x_1, ..., x_d) \in \mathbb{R}^d$. If $\mathbb{P}_X$ is absolutely continuous with respect to the Lebesgue measure on $\mathbb{R}^d$, the joint probability density function $\rho : \mathbb{R}^d \to [0,1]$ is defined as:

$$\rho := \frac{\partial^d \mathbb{P}_{\boldsymbol{X}}(\boldsymbol{x})}{\partial x_1 \ldots \partial x_d} \ .$$

Definition 3.1.14 (Integrability)**.** *Let $(\Theta, \mathcal{F}, \mathbb{P})$ be a probability space and X be a random variable. $X : (\Theta, \mathcal{F}) \to (\Psi, \mathcal{G})$, $(\Psi, \mathcal{G})$ is a measurable space. The integral of X with respect to $\mathbb{P}$ over the event $F \in \mathcal{F}$ is defined as:*

$$\int_\Theta I_F(\theta) X(\theta) d\mathbb{P}(\theta) = \int_F X(\theta) d\mathbb{P}(\theta) \ .$$

Here, I_F is the indicator function of F. If the integral exists and is finite, X is called $\mathbb{P}$-integrable over F.

Definition 3.1.15 (Expectation)**.** *Let $X : \Theta \to \mathbb{R}$ be an integrable real-valued random variable on a probability space $(\Theta, \mathcal{F}, \mathbb{P})$. The expectation of X is defined:*

$$\mathbb{E}(X) := \int_\Theta X(\theta) d\mathbb{P}(\theta) \ . \tag{3.4}$$

The expectation is also called mean.

Definition 3.1.16 (Variance)**.** *Let X be a real-valued random variable over a probability space $(\Theta, \mathcal{F}, \mathbb{P})$, $\mu = \mathbb{E}(X)$. The variance is defined as:*

$$Var(X) := \mathbb{E}[(X - \mu)^2] \ .$$

$\sigma := \sqrt{Var(X)}$ is called standard deviation.

Definition 3.1.17 (Covariance)**.** *Considering two real-valued random variables X and Y over a probability space, $\mu_X = \mathbb{E}(X)$ and $\mu_Y = \mathbb{E}(Y)$, the covariance is defined:*

$$Cov(X, Y) := \mathbb{E}[(X - \mu_X)(Y - \mu_Y)] \ .$$

If X and Y are integrable, then:

$$Cov(X, Y) = \mathbb{E}[XY] - \mu_X \mu_Y \ .$$

Particularly, $Cov(X, X) = Var(X)$.

Definition 3.1.18 (Correlation)**.** *The covariance can be rescaled into the range $[-1, 1]$:*

$$\rho(X, Y) := \frac{Cov(XY)}{\sigma_X \sigma_Y} \ ,$$

which is defined as the correlation coefficient. σ_X is the standard deviation of X, and σ_Y is the standard deviation of Y.

Definition 3.1.19 (Uncorrelated random variables)**.** *If $Cov(X, Y) = 0$, the random variables X, Y are called uncorrelated.*

Definition 3.1.20 (Independence of events). *Two events $F, G \in \mathcal{F}$, if $\mathbb{P}(F \cap G) = \mathbb{P}(F)\mathbb{P}(G)$, they are independent.*

Definition 3.1.21 (Independence of σ-algebra). *Two sub-σ-algebras $\mathcal{F}_1$, $\mathcal{F}_2$ of the σ-algebra $\mathcal{F}$ are independent, if the events F_1, F_2 are independent for all $F_1 \in \mathcal{F}_1$ and $F_2 \in \mathcal{F}_2$.*

Definition 3.1.22 (Independence of random variables). *Two random variables, X and Y, are independent on a probability space $(\Theta, \mathcal{F}, \mathbb{P})$, if the σ-algebra $\sigma(X)$ and $\sigma(Y)$ generated by X and Y are independent.*

Definition 3.1.23 (Stochastic process). *Given a probability space $(\Theta, \mathcal{F}, \mathbb{P})$ and a measurable space $(\Psi, \mathcal{G})$. A stochastic process is defined as a family of random variables $\{\boldsymbol{X}(t), t \in T\}$ for some index set T and $X(\Theta, \mathcal{F}) \to (\Psi, \mathcal{G}), \forall t \in T$.*

Definition 3.1.24 (Almost surely). *Let $(\Theta, \mathcal{F}, \mathbb{P})$ be a probability space, an event $F \in \mathcal{F}$ happens almost surely (a.s.), if $\mathbb{P}(F) = 1$.*

3.1.2 Partial Differential Equation with Stochastic Data

With the previously discussed elementary knowledge in probability theory, we can focus now on partial differential equations (PDEs) with stochastic data, as it governs the physical behavior which is studied in this work.

Let us first consider an abstract partial differential equation (PDE):

$$\mathcal{L}(\boldsymbol{a})\boldsymbol{u} = \boldsymbol{f}\ , \quad \text{in}\, \mathcal{D} \times [0, T]\ . \tag{3.5}$$

$\mathcal{L}$ is a partial differential operator in a domain $\mathcal{D} \subset \mathbb{R}^d$, $d \in \mathbb{N}$ denotes the spatial dimension. The partial differential operator includes the initial and boundary conditions, $\boldsymbol{f}$ is the external forcing term. $\mathcal{L}$ may also depend on some space-time dependent parameter function $\boldsymbol{a} : \mathcal{D} \times [0, T] \to \mathbb{R}^r$, where r is the dimension of the parametric space. In general, we consider Equation (3.5) as a parametrized partial differential equation (PPDE) [76]. Especially, we consider a quantity of interest $qi(\boldsymbol{a}) \in \mathbb{R}$, which is regarded as a functional of $\boldsymbol{u}(\boldsymbol{a})$, and $\boldsymbol{u}(\boldsymbol{a})$ is the solution of a PDE. Here, "parameterized" refers to the input parameters.

In order to concentrate on parameterized partial differential equations with random data, we introduce first a stochastic space $(\Theta, \mathcal{F}, \mathbb{P})$ and consider the input parameters $\boldsymbol{a}$ to be random. In other words, a stochastic process is added into the system, $\boldsymbol{a} : \Theta \times \mathcal{D} \times [0, T] \to \mathbb{R}^r$, $\boldsymbol{a}$ becomes thus a space-time stochastic parameter. In the meantime, we also specify the system forcing term $\boldsymbol{f}(\boldsymbol{x}, \theta, t)$ to be random, i.e. $\boldsymbol{f} : \Theta \times \mathcal{D} \times [0, T] \to \mathbb{R}^s$ and $\theta \in \Theta$, $\boldsymbol{x} \in \mathcal{D}$, $s \in \mathbb{N}$. The solution $\boldsymbol{u}$ is also dependent on the set of outcomes, $\boldsymbol{u}(\boldsymbol{x}, \theta, t) : \Theta \times \mathcal{D} \times [0, T] \to \mathbb{R}^s$.

The abstract PDE with stochastic data problem can be defined as follows:

Definition 3.1.25 (Partial differential equation with stochastic data). *For a given stochastic space $(\Theta, \mathcal{F}, \mathbb{P})$, find a solution $\boldsymbol{u} : \Theta \times \mathcal{D} \times [0, T] \to \mathbb{R}^s$, such that*

$$\mathcal{L}(\boldsymbol{a})\boldsymbol{u} = \boldsymbol{f}\ , \quad in\, \Theta \times \mathcal{D} \times [0, T]\ . \tag{3.6}$$

Here, $\mathcal{D} \subset \mathbb{R}^d$ is the spatial domain, $T > 0$, $T \in \mathbb{R}$ is the final time. $\boldsymbol{a} : \Theta \times \mathcal{D} \times [0, T] \to \mathbb{R}^r$ is the space-time stochastic parameter. $d, s, r \in \mathbb{N}$. $\mathcal{L}$ is a partial differential operator with appropriate boundary and initial conditions. $\boldsymbol{f}$ is the external forcing term to the PDE.

For a Banach space $W(\mathcal{D})$ [123] with a function $v : \mathcal{D} \to \mathbb{R}$, the stochastic Banach space [130] looks like:

$$\begin{aligned} L^q_P(\Theta, W(\mathcal{D})) := \Big\{ v : \Theta \to W(\mathcal{D}) | v \text{ is strongly measurable and} \\ \int_\Theta \|v(\cdot, \theta)\|^q_{W(\mathcal{D})} dP(\theta) < +\infty \Big\}\ , \end{aligned} \tag{3.7}$$

for $q \in [1, +\infty)$, and for $q = \infty$:

$$L_P^{\infty}(\Theta, W(\mathcal{D})) := \Big\{ v : \Theta \rightarrow W(\mathcal{D}) | v \text{ is strongly measurable and} \\ P - \text{ess sup}_{\theta \in \Theta} \|v(\cdot, \theta)\|_{W(\mathcal{D})}^2 < +\infty \Big\} . \tag{3.8}$$

Particularly, we are interested in $L_P^2(\Theta, W(\mathcal{D}))$, whose elements have finite second stochastic moments.

Therefore, one can view the solution of Equation (3.6) as a random field, and the random variables are indexed by a multidimensional parameter, which can be even infinite-dimensional. As mentioned previously, investigating the influence of the random parameters of a fluid is one of the focusing points of this thesis. According to the definitions above, $\boldsymbol{u}$ must exist for all $\theta \in \Theta$, whereas the existence and the uniqueness of strong solutions to the Navier-Stokes equations are still not proven. However, we have to assume that we can obtain the solution numerically in our study of the considered stochastic flow problem.

3.1.3 Assumption of Finite-Dimensional Probability Space

Throughout the rest of this thesis, we assume firstly that we are only dealing with a low dimensional probability space, i.e. a small number of random variables is considered. Even though, the infinite-dimensional stochastic PDEs cover more general situations and numerous physical phenomena, focusing only on certain important criteria with respect to the random resources in order to quantify the stochastic numerical solutions of a specific system is the most common situation for engineering applications.

Therefore, we assume in our considered problem that just a small number of independent random variables is subject to a pre-defined probability space. We state the similar assumption as in [12]:

Assumption 3.1.1 (Finite-dimensional noise)**.** *For a given probability space* $(\Theta, \mathcal{F}, \mathbb{P})$*,* $\theta \in \Theta$ *is a single outcome. The coefficients in Equation* (3.6) *have the following form:*

$$\boldsymbol{a}(\theta, \boldsymbol{x}, t) = \boldsymbol{a}(Y_1(\theta), ..., Y_N(\theta), \boldsymbol{x}, t) , \qquad (\Theta, \boldsymbol{x}, t) \in \Theta \times \mathcal{D} \times [0, T] , \tag{3.9a}$$

$$\boldsymbol{f}(\theta, \boldsymbol{x}, t) = \boldsymbol{f}(Y_1(\theta), ..., Y_N(\theta), \boldsymbol{x}, t) , \qquad (\Theta, \boldsymbol{x}, t) \in \Theta \times \mathcal{D} \times [0, T] , \tag{3.9b}$$

where, $N > 0$*,* $N \in \mathbb{N}$*.* $\{Y_i\}_{i=1}^N$ *are real-valued random variables on* $(\Theta, \mathcal{F})$ *with zero mean value and unit variance.*

Theorem 3.1.1 (Karhunen-Loève expansion [109, 174])**.** *Let* $(\Theta, \mathcal{F}, \mathbb{P})$ *be a probability space, and* $a : \Theta \times \mathcal{D} \rightarrow \mathbb{R}, a \in L^2(\Theta, \mathcal{F}, \mathbb{P})$ *is a random field with* $a(\cdot, \theta) \in L^2, \forall \theta \in \Theta$*, then* $a \in L^2(\Theta \times \mathcal{D})$*. Therefore, we can describe this random field by using the Karhunen-Lòve expansion:*

$$a(\boldsymbol{x}, \theta) = \mathbb{E}[a](\boldsymbol{x}) + \sum_{i=1}^{\infty} \sqrt{\lambda_i} \psi_i(\boldsymbol{\xi}) a_i(\theta) , \tag{3.10}$$

where, ψ_i *is a sequence of eigenfunctions, and* λ_i *is a sequence of eigenvalues of the following eigenvalue problem:*

$$\int_{\mathcal{D}} Cov[a](\boldsymbol{x}, \boldsymbol{x}') \psi_i(\boldsymbol{x}') d\boldsymbol{x}' = \lambda_i \psi_i(\boldsymbol{x}) , \quad \forall \boldsymbol{x} \in \mathcal{D} . \tag{3.11}$$

Moreover, $\{a_i(\theta)\}$ *are mutually independent random variables:*

$$\mathbb{E}[a_i] = 0, \quad \mathbb{E}[a_i a_j] = \delta_{ij} . \tag{3.12}$$

Here, δ_{ij} *is the Kronecker delta.* a_i *is defined as:*

$$a_i(\theta) = \frac{1}{\sqrt{\lambda_i}} \int_{\mathcal{D}} (a_i(\theta) - \mathbb{E}[a](\boldsymbol{x})\psi_i(\boldsymbol{x}) d\boldsymbol{x} \; . \tag{3.13}$$

Example 3.1.1 (Karhunen–Loève expansions)**.** *If we consider the random variables* $\boldsymbol{a}(\theta, \boldsymbol{x}, t)$ *and* $\boldsymbol{f}(\theta, \boldsymbol{x}, t)$ *in Assumption 3.1.1 and assume them can be decomposed by using Karhunen-Loève expansion as described in Theorem 3.1.1. These two random variables can be approximated with the following forms:*

$$\boldsymbol{a}(\theta, \boldsymbol{x}, t) \approx \boldsymbol{a}_0 + \sum_{n=1}^{N} \sqrt{\lambda_n^a} \psi_n^a(\theta) \boldsymbol{a}_n(\boldsymbol{x}, t) \; . \tag{3.14}$$

$$\boldsymbol{f}(\theta, \boldsymbol{x}, t) \approx \boldsymbol{f}_0 + \sum_{n=1}^{N} \sqrt{\lambda_n^f} \psi_n^f(\theta) \boldsymbol{f}_n(\boldsymbol{x}, t) \; . \tag{3.15}$$

Referring to Mercer's theorem [124], any random field, e.g. $\boldsymbol{a}(\theta, \boldsymbol{x}, t)$, with a continuous covariance function can be modeled as a sum of the infinite random variables based on a bi-orthogonal decomposition. For many simulations, the expression of the series is truncated to a finite number of terms, in Equation (3.14), $\lambda_n, \boldsymbol{a}_n, n = 1, ..., N$, are the dominant eigenvalues and eigenfunctions. It holds the same for Equation (3.15).

There are several ways to represent a random field with a finite-dimensional basis. Apart from the Karhunen-Loève expansion, the Polynomial Chaos expansion is extensively applied in many fields. The Polynomial Chaos representation for the random variables is the main technique that we employ in this work for quantifying uncertainties, it will be introduced in the next section (Section 3.2).

3.1.4 Uncertainty Quantification for PDEs with Random Inputs

Considering the partial differential equations with stochastic data (Equation (3.6)), we briefly illustrate the model-based PDE simulation for the Uncertainty Quantification. As outlined in Figure 3.1, we concentrate in the first place on our PDE system with random inputs. We model the input random field into a finite-dimensional manner as mentioned in the previous assumption, then we solve the stochastic system in order to obtain the stochastic solution $\boldsymbol{u}(\theta, \boldsymbol{x}, t)$. This process is generally considered as a forward propagation of the uncertainty. It means that we are interested in observing the impact of the input parameters with respect to their random behavior on the solution $\boldsymbol{u}(\theta, \boldsymbol{x}, t)$.

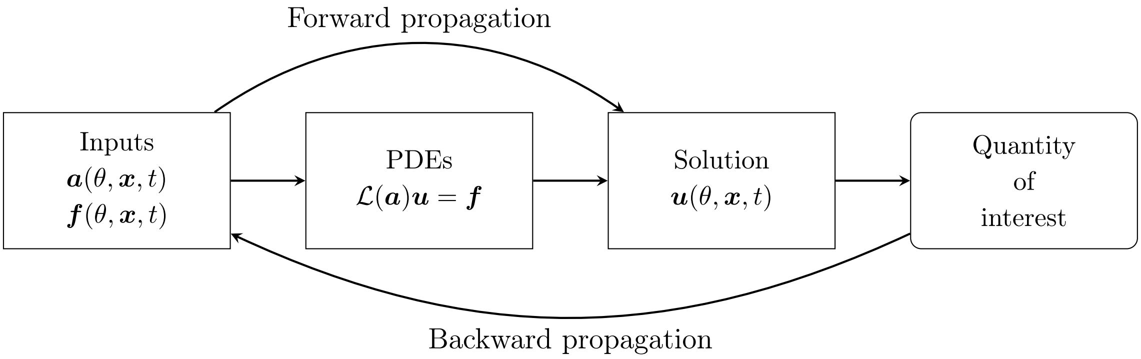

Forward propagation
Inputs
$\boldsymbol{a}(\theta, \boldsymbol{x}, t)$
$\boldsymbol{f}(\theta, \boldsymbol{x}, t)$
PDEs
$\mathcal{L}(\boldsymbol{a})\boldsymbol{u} = \boldsymbol{f}$
Solution
$\boldsymbol{u}(\theta, \boldsymbol{x}, t)$
Quantity
of
interest
Backward propagation

Figure 3.1: Illustration of the Uncertainty Quantification for the partial differential equations with the random inputs.

As the solution $\boldsymbol{u}(\theta, \boldsymbol{x}, t)$ depends on stochastic parameters, we also want to extract certain information, which is more valuable to our perspective, from the stochastic solution. Therefore, the forward propagation can be even extended to the process of computing a given quantity of interest (QoI). On the other hand, it is also important to identify the unknown objects from the input parameters so as to improve the modeling of the random inputs, this procedure is stated as backward propagation. In this work, we focus only on the forward problem of the Uncertainty Quantification, but, it has to be mentioned that the inverse problem is also a very important step in the real world applications. In the rest of this chapter, modeling the input random field and setting the random partial differential equation for our fluid problem are discussed, the solving techniques will be studied in the next chapter (Chapter 4).

3.2 Generalized Polynomial Chaos Expansion

The term *Polynomial Chaos* is initially introduced by Wiener [171], who modeled a countable sequence of uncorrelated Gaussian random variables with the aid of a series of Hermite polynomials. Soon afterward, Ghanem proposed using the truncated polynomial chaos expansions as the trial functions in the finite element method to solve structural mechanics problems with some parametric random parameters [62]. This approach is then named as the *stochastic finite element method* (SFEM), also known as the *stochastic Galerkin method*. Thereafter, Xiu carried this concept to a more general framework in order to adapt general probability distributions, it stands for the *generalized Polynomial Chaos* (gPC) [176]. This subsection lays emphasis on the concept of the generalized Polynomial Chaos expansion and the structural pattern of the stochastic Galerkin system.

3.2.1 Wiener Polynomial Chaos Expansion

Let $(\Theta, \mathcal{F}, \mathbb{P})$ be a probability space, and a real-valued random variable $X := X(\theta) : \Theta \to \mathbb{R}$. $\theta \in \Theta$ is a random event in the sample space Θ. Moreover, we assume that X is square-integrable, i.e.:

$$X \in L^2(\Theta) := \{X(\theta) : \theta \to \mathbb{R}, \mathbb{E}(X^2) < \infty\} . \tag{3.16}$$

Here, $\mathbb{E}(X^2)$ denotes the expectation value (Equation (3.4)) of X^2 with respect to $\mathbb{P}$. We obtain therefore a Hilbert space of real-valued random variables with inner product $(\cdot,\cdot)_{L^2(\Theta)}$ and norm $\|\cdot\|_{L^2(\Theta)}$. Considering a Gaussian Hilbert space $\mathcal{H} \subset L^2(\Theta, \mathcal{F}, \mathbb{P})$ consisting only of the normalized, centered and uncorrelated Gaussian variables $\{\xi_i\}_{i=1}^M$, where, $\xi_i : \Theta \to \mathbb{R}$, $\xi_i \sim \mathcal{N}(0,1)$. As the moments of the independent Gaussian random variables are purely the products of each individual moment, for any Gaussian Hilbert space $\mathcal{H}$, we have:

$$\begin{aligned} P_K(\mathcal{H}) := \Big\{ p_k(\xi_1, ..., \xi_M) |\, p \,\text{is a polynomial of degree}\, k \ , \ k \leq K \ , \ K \in \mathbb{N}_0 \ , \\ \xi_i \in \mathcal{H} \ , i = 1, ..., M \ , M \in \mathbb{N} \Big\} \ . \end{aligned} \tag{3.17}$$

is a linear subspace of $L^2(\Theta, \mathcal{F}, \mathbb{P})$. $P_K(\mathcal{H})$ consists of polynomials with arbitrary number of random variables, which are selected from $\mathcal{H}$.

Theorem 3.2.1 (Cameron-Martin theorem [52]). *Under the conditions mentioned above, a space $\tilde{P}_K$ is supported by the polynomial P_K in Equation (3.17) at degree K, which is defined as:*

$$\tilde{P}_K = \overline{P}_K(\mathcal{H}) \cap P_{k-1}(\mathcal{H})^{\perp} \ , \quad K \in \mathbb{N} \ . \tag{3.18}$$

$\overline{P}_K$ is the closure with respect to L^2. The spaces $\{\tilde{P}_k\}_{k \in \mathbb{N}_0}$ form a sequence of closed and pairwise orthogonal linear subspaces of $L^2(\Theta, \mathcal{F}, \mathbb{P})$, such that:

$$\bigoplus_{k=0}^{\infty} \tilde{P}_k = L^2(\Theta, \sigma(\mathcal{H}), \mathbb{P}) \; . \tag{3.19}$$

$\sigma(\mathcal{H})$ is the σ-algebra generated by a set $\mathcal{H}$ of random variables. Particularly, if $\sigma(\mathcal{H}) = \mathcal{F}$, the orthogonal decomposition yields:

$$L^2(\Theta, \mathcal{F}, \mathbb{P}) = \bigoplus_{k=0}^{\infty} \tilde{P}_k \; . \tag{3.20}$$

The first proof can be found in [36], a more general formulation is stated in [96].

Therefore, the original random variable X can be expressed as [178]:

$$\begin{aligned} X(\theta) &= a_0 H_0 \\ &+ \sum_{i_1=1}^{M} a_{i_1} H_1(\xi_{i_1}(\theta)) \\ &+ \sum_{i_1=1}^{M} \sum_{i_2=1}^{i_1} a_{i_1 i_2} H_2(\xi_{i_1}(\theta), \xi_{i_2}(\theta)) \\ &+ \sum_{i_1=1}^{M} \sum_{i_2=1}^{i_1} \sum_{i_3=1}^{i_2} a_{i_1 i_2 i_3} H_3(\xi_{i_1}(\theta), \xi_{i_2}(\theta), \xi_{i_3}(\theta)) \\ &+ \cdots \; . \end{aligned} \tag{3.21}$$

Here, $H_p(\xi_{i_1}, ..., \xi_{i_p})$ is the multivariate or multidimensional univariate Hermite polynomial of degree p with the standard Gaussian variables $(\xi_{i_1}, ..., \xi_{i_p})$, i.e. $\xi_j \sim \mathcal{N}(0,1), j = 1, ..., M$. This orthogonal decomposition converges almost surely to X in a mean-square sense:

$$\lim_{p \to \infty} \mathbb{E}[(a_0 H_0 + \cdots + \sum_{i_1=1}^{M} \cdots \sum_{i_p=1}^{i_{p-1}} a_{i_1 \cdots i_p} H_p(\xi_{i_1}, \cdots, \xi_{i_p}) - X)^2] = 0 \; . \tag{3.22}$$

The multivariate Hermite polynomials are built on the one-dimensional Hermite polynomials. The general expression is stated as [176]:

$$H_p(\xi_{i_1}, \cdots, \xi_{i_p}) = e^{\frac{1}{2}\boldsymbol{\xi}^T \boldsymbol{\xi}} (-1)^p \frac{\partial^p}{\partial \xi_{i_1} \cdots \partial \xi_{i_p}} e^{-\frac{1}{2}\boldsymbol{\xi}^T \boldsymbol{\xi}} \; . \tag{3.23}$$

Equation (3.21) can be hence written in a compact form:

$$X(\theta) = \sum_{i=0}^{\infty} x_i \psi_i(\boldsymbol{\xi}(\theta)) \; , \quad \boldsymbol{\xi} = \{\xi_1, ..., \xi_M\} \; . \tag{3.24}$$

Here, x_i are called Polynomial Chaos (PC) coefficients. Each $\psi_i(\boldsymbol{\xi})$ has one-to-one mapping to a corresponding Hermite polynomial $H_p(\xi_{i_1}, \cdots, \xi_{i_p})$. In the following, $\psi_i(\boldsymbol{\xi})$ is employed more often for representing the general polynomial basis for the conventional purpose. The polynomial ψ_i can be constructed generally by a tensor product of one-dimensional polynomials ϕ_i with a suitable multi-index $\boldsymbol{\alpha^i} = (\alpha_1^i, ..., \alpha_p^i) \in \mathbb{N}_0$, i.e.:

$$\psi_i(\xi_1, ..., \xi_p) := \prod_{k=1}^{p} \phi_{\alpha_k^i}(\xi_k) \, . \tag{3.25}$$

Remark 3.2.1. *If the number of the random variables is more than one, there are more than one polynomial for* $p > 0$*. However, the input setting* $(\xi_{i_1}, ..., \xi_{i_p})$ *can determine* H_p *differently for the same order of polynomial* p*. Moreover,* $\psi_i(\boldsymbol{\xi})$ *is only a single polynomial, not a set of polynomials.*

Example 3.2.1 (Three-dimensional Hermite polynomials)**.** *The multi-index* $\boldsymbol{\alpha}^i$ *has a compact support, there exits only a finite number of indices, which is non-zero.* i *indicates the global index and* p *denotes the degree of polynomials (Table 3.1).*

i	p	ψ_i	α^i
0	0	1	(0,0,0)
1	1	ξ_1	(1,0,0)
2	1	ξ_2	(0,1,0)
3	1	ξ_3	(0,0,1)
4	2	$\xi_1^2 - 1$	(2,0,0)
5	2	$\xi_1\xi_2$	(1,1,0)
6	2	$\xi_1\xi_3$	(0,1,1)
7	2	$\xi_2^2 - 1$	(0,2,0)
8	2	$\xi_2\xi_3$	(0,1,1)
9	2	$\xi_3^2 - 1$	(0,0,2)
...	...	...	...

Table 3.1: The polynomials of the three-dimensional Hermite polynomials (up to the second order).

As mentioned in the theorem above, the Hermite-Chaos Polynomials offer a complete orthogonal basis:

$$\langle \psi_i, \psi_j \rangle = \langle \psi_i, \psi_i \rangle \delta_{ij} \ , \tag{3.26}$$

δ_{ij} is the Kronecker delta function, $\langle \cdot, \cdot \rangle$ is the inner product, it implies:

$$\langle f(\boldsymbol{\xi}), g(\boldsymbol{\xi}) \rangle := \int f(\boldsymbol{\xi}) g(\boldsymbol{\xi}) w(\boldsymbol{\xi}) d\boldsymbol{\xi} \ , \tag{3.27}$$

where, $w(\boldsymbol{\xi})$ is the weight function or the probability density function of the multi-dimensional standard Gaussian random variable $\boldsymbol{\xi}$. Furthermore, according to Equation (3.1), the probability density function can be calculated from the derivative of the joint probability distribution $P(\boldsymbol{\xi})$ of $\boldsymbol{\xi}$, hence:

$$\langle f(\boldsymbol{\xi}), g(\boldsymbol{\xi}) \rangle := \int f(\boldsymbol{\xi}) g(\boldsymbol{\xi}) dP(\boldsymbol{\xi}) \ . \tag{3.28}$$

However, for the practical reasons, we can not employ an expansion (Equation (3.24)) with an infinite number of terms, we have to utilize a truncated decomposition of the form:

$$X(\theta) = \sum_{i=0}^{P} x_i \psi_i(\boldsymbol{\xi}(\theta)) + \varepsilon(M, p) \ . \tag{3.29}$$

$X(\Theta)$ depends on a finite number of random variables M. We truncate the degree of the polynomials up to L, the total number of polynomials $(P+1)$ can be computed via:

$$P + 1 = \frac{(M+L)!}{M!L!} \ . \tag{3.30}$$

$\varepsilon(L)$ is the error with respect to L. It vanishes if we decompose $\boldsymbol{X}(\theta)$ into the infinite number of basis random variables and use the infinite degree of polynomials, it implies:

$$\lim_{L \to \infty} \varepsilon(L) \to 0 \ . \tag{3.31}$$

3.2.2 Generalization

The Cameron-Martin theorem [36] provides the ability of modeling stochastic differential equations (SPDEs) with Gaussian random inputs. Another types of random input can also be modeled with this technique. Nonetheless, using the Hermite polynomials to approximate the non-Gaussian inputs is not optimal, and it leads to a slow convergence behavior.

Whence, the Wiener-Askey Polynomial Chaos expansion [176] is introduced for dealing with more general random inputs based on the Wiener-Chaos expansion. This technique generates different classes of polynomials by making use of the Askey-scheme [4, 11, 104]. These polynomials are optimal in terms of applying the correct measure of the probability distribution function to each random variable. This scheme offers a structure of the hierarchical relations between the hypergeometric orthogonal polynomials, and the lower hierarchy polynomial can be then obtained by taking the limit of certain parameters from the upper polynomial. Additionally, the Hermite polynomial is placed to the end of the structure.

Example 3.2.2. *The Hermite polynomial $H_n(x)$ can be calculated by taking the limit of α in the Jacobi polynomial $P_n^{\alpha,\alpha}(x)$ [161].*

$$\lim_{\alpha \to \infty} \alpha^{-\frac{1}{2}n} P_n^{(\alpha,\alpha)}\left(\frac{x}{\sqrt{\alpha}}\right) = \frac{H_n(x)}{2^n n!} \ .$$

We can rewrite the decomposition from the Hermite-Chaos expansion (Equation (3.21)) in a more general form, i.e., the general second order random process $X(\theta)$:

$$\begin{aligned} X(\theta) &= a_0 I_0 \\ &+ \sum_{i_1=1}^{M} a_{i_1} I_1(\xi_{i_1}(\theta)) \\ &+ \sum_{i_1=1}^{M} \sum_{i_2=1}^{i_1} a_{i_1 i_2} I_2(\xi_{i_1}(\theta), \xi_{i_2}(\theta)) \\ &+ \sum_{i_1=1}^{M} \sum_{i_2=1}^{i_1} \sum_{i_3=1}^{i_2} a_{i_1 i_2 i_3} I_3(\xi_{i_1}(\theta), \xi_{i_2}(\theta), \xi_{i_3}) \\ &+ \cdots \ . \end{aligned} \tag{3.32}$$

Here, $I_p(\xi_{i_1}, ..., \xi_{i_p})$ denotes the Askey polynomial at the order p, and $(\xi_{i_1}, ..., \xi_{i_p})$ are defined similarly (Equation (3.25)) by the multi-index. The construction is identical as in Equation (3.25) by taking the product of the one-dimensional polynomials regarding the multi-index $\boldsymbol{\alpha}^i$. The general compact form of the truncated stochastic spectral expansion can be stated as:

$$X(\theta) \simeq \sum_{i=0}^{P} x_i \psi_i(\boldsymbol{\xi}) \ , \quad \boldsymbol{\xi} = \{\xi_1, ..., \xi_M\} \ . \tag{3.33}$$

Here, we use the truncated version and assume that the truncation error $\varepsilon(L)$ is relatively small. Note that, $\psi_i(\boldsymbol{\xi})$ is the general Wiener-Askey polynomial, which takes a random vector $\boldsymbol{\xi}$ consisting of the random variables defined in the Askey-scheme as the parameter. The orthogonality of the Wiener-Askey chaos polynomials is defined analogously as in Equation (3.26). Now we consider the random process $X(\theta)$ being a function of the random vector $\boldsymbol{\xi}(\theta) = [\xi_1(\theta), ..., \xi_M(\theta)]$, and we exploit directly the relation between the realizations $\theta \in \Theta$ and the random vector $\boldsymbol{\xi}(\theta)$. Subsequently, we note $X(\boldsymbol{\xi})$ instead of $X(\theta)$ for the sake of convenience, because $\boldsymbol{\xi}$ is also defined directly on the sample space Θ. The Askey scheme forms a complete Hilbert space for each class of polynomial, the orthogonality of the multidimensional polynomials with multiple random variables applies in the same way as in Equation (3.26). Besides, the generalized Polynomial Chaos expansion (gPCE) is not restricted to standard probability distributions anymore, the weight function in Equation (3.27) is constructed with respect to each individual polynomial. A summary of the Wiener-Askey polynomials with the corresponding random variables can be found in Table 3.2.

	Random variables $\boldsymbol{\xi}$	Wiener-Askey chaos $\psi(\boldsymbol{\xi})$	Support
Continuous	Gaussian	Hermite-chaos	$[-\infty, +\infty]$
	Uniform	Legendre-chaos	$[a, b]$
	Gamma	Laguerre-chaos	$[0, +\infty]$
	Beta	Jacobi-chaos	$[a, b]$
Discontinuous	Poisson	Charlier-chaos	$\{0, 1, 2, ...\}$
	Binomial	Krawtchouk-chaos	$\{0, 1, ..., N\}$
	Negative Binomial	Meixner-chaos	$\{0, 1, 2, ...\}$
	Hypergeometric	Hahn-chaos	$\{0, 1, ..., N\}$

Table 3.2: The Wiener-Askey polynomials with the corresponding random variables.

The generalized polynomial basis is then built by taking the most optimal univariate basis from the Wiener-Askey scheme for each random input, and the multivariate polynomials are simply the product of Wiener-Askey polynomial chaos basis according to the multi-index. In a similar fashion, the weight function is also the product of one-dimensional weight functions for each univariate basis. Nonetheless, one important assumption is to presume that the random variables $\boldsymbol{\xi}$ are mutually independent. Because of that, the construction of the Polynomial Chaos basis in the multidimensional case is less complicated, and we use this assumption also within this thesis. However, the generalized PC can also deal with dependent random variables, more details can be found in [109, 80]. By exploiting the orthogonality of the Chaos Polynomials, the mean value and the variance of the random process $X(\boldsymbol{\theta})$ can be written as:

$$\begin{aligned} \mathbb{E}[X(\theta)] &= \mathbb{E}[\sum_{i=0}^{P} x_i \psi_i(\boldsymbol{\xi}(\theta))] \\ &= x_0 \mathbb{E}[\psi_0(\boldsymbol{\xi}(\theta))] + \sum_{i=1}^{P} x_i \mathbb{E}[\psi_i(\boldsymbol{\xi}(\theta))] \\ &= x_0 \;. \end{aligned} \tag{3.34}$$

$$\begin{aligned} Var(X(\theta)) &= \mathbb{E}[(X(\boldsymbol{\xi}(\theta)) - \mathbb{E}[X(\boldsymbol{\xi}(\theta))])^2] \\ &= \mathbb{E}[(\sum_{i=0}^{P} x_i \psi_i(\boldsymbol{\xi}(\theta)) - x_0)^2] \end{aligned} \tag{3.35}$$

$$
\begin{aligned}
&= \mathbb{E}[(\sum_{i=1}^{K} x_i \psi_i(\boldsymbol{\xi}(\theta)))^2] \\
&= \sum_{i=1}^{P} x_i^2 \langle \psi_i(\boldsymbol{\xi}(\theta)), \psi_i(\boldsymbol{\xi}(\theta)) \rangle \ .
\end{aligned}
$$

Concerning Equation (3.33), x_i is regarded as the stochastic mode henceforth. The final goal of studying a second order random process with the help of the generalized Polynomial Chaos expansion is to define the stochastic mode x_i and the appropriate Polynomial Chaos basis ψ_i. The k-th stochastic mode can be obtained by utilizing the orthogonality:

$$
\begin{aligned}
\langle X, \psi_k \rangle &= \langle \sum_{i=0}^{P} x_i \psi_i, \psi_k \rangle \\
&= \sum_{i=0}^{P} u_i \langle \psi_i, \psi_k \rangle = x_k \langle \psi_k, \psi_k \rangle \ .
\end{aligned} \tag{3.36}
$$

Likewise, the higher order moments can also be computed in a similar way by exploiting the properties of the orthogonal PC basis.

3.2.3 Stochastic Galerkin Method

We seek the stochastic solution from a parameterized partial differential equation with random inputs:

$$
\mathcal{L}(\boldsymbol{a}(\boldsymbol{x}, t, \boldsymbol{\xi}))\boldsymbol{u}(\boldsymbol{x}, t, \boldsymbol{\xi}) = \boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{\xi}) \ , \quad \text{in}\, \mathcal{D} \times [0, T] \times \Theta \ . \tag{3.37}
$$

$\mathcal{L}$ represents a partial differential operator, it contains the spatial derivatives with a spatial variable $\boldsymbol{x} \in \mathcal{D} \subset \mathbb{R}^d$, the time derivatives with a time variable $t \in [0, T]$ and the linear or non-liner parts. $\boldsymbol{\xi}$ is a random vector $\boldsymbol{\xi} := [\xi_1, ..., \xi_M] : \Theta \to \mathbb{R}$, which is defined on the sample space Θ. The random vector contains the initial conditions, the boundary conditions and the equation parameters, etc. $\boldsymbol{u}$ is denoted the solution of this partial differential system, which is dependent on time, space and the stochastic parameter as well. The stochastic Galerkin method employs the weighted residual function to construct a Polynomial Chaos system consisting of the PC coefficients and the corresponding projected solutions, which build the relation between the system solutions and the random data.

Given a pre-defined approximated stochastic Polynomial Chaos basis ψ_i, the representation of the solution $u(\boldsymbol{x}, t, \boldsymbol{\xi})$ is given as:

$$
\boldsymbol{u}(\boldsymbol{x}, t, \boldsymbol{\xi}) = \sum_{i=0}^{P} \boldsymbol{u}_i(\boldsymbol{x}, t) \psi_i(\boldsymbol{\xi}) \ . \tag{3.38}
$$

$\boldsymbol{\xi} = \{\xi_1, ..., \xi_M\}$ and $\{\psi_i\}$ are chosen with respect to the stochastic properties of the random inputs, the highest number of the Polynomial Chaos mode P can be obtained from Equation (3.30). ψ_i is a multidimensional orthogonal polynomial by taking $\boldsymbol{\xi}$ as the random variables. The orthogonality is given with respect to an inner product over the parameter space. $\boldsymbol{u}(\boldsymbol{x}, t, \boldsymbol{\xi}) \in \mathcal{V} \bigotimes \mathcal{S}^P$, $\mathcal{V}$ is a well-defined Hilbert space for the deterministic solution and independent from the random inputs. $\mathcal{S}^P$ is the approximated stochastic space spanned by the Chaos Polynomials:

$$
\mathcal{S}^P = \text{span}\{\psi_0, ..., \psi_P\} \ . \tag{3.39}
$$

Remark 3.2.2. *The approximated stochastic space is a subset of the full stochastic space, which is defined by expanding the stochastic process onto the infinite stochastic basis:*

$$\mathcal{S}^P \subset \mathcal{S} \ , \quad \mathcal{S} = \bigcup_{i=0}^{\infty} span\{\psi_0, \psi_1, ..., \psi_i\} \ . \tag{3.40}$$

Substituting $\boldsymbol{u}(\boldsymbol{x}, t, \boldsymbol{\xi})$ in Equation (3.37) by Equation (3.38) yields:

$$\mathcal{L}(\boldsymbol{a}(\boldsymbol{x}, t, \boldsymbol{\xi}))(\sum_{i=0}^{P} \boldsymbol{u}_i(\boldsymbol{x}, t)\psi_i(\boldsymbol{\xi})) = \boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{\xi}) \ , \quad \text{in}\, \mathcal{D} \times [0, T] \times \Theta \ . \tag{3.41}$$

The Galerkin projection employs the same approximated stochastic space for the solution and the test random variables [109]. It conveys that we multiply the basis polynomial ψ_i on each side of Equation (3.41), then we take the inner product:

$$\langle \mathcal{L}(\boldsymbol{a}(\boldsymbol{x}, t, \boldsymbol{\xi})(\sum_{i=0}^{P} \boldsymbol{u}_i \psi_i(\boldsymbol{\xi})), \psi_j(\boldsymbol{\xi})\rangle = \langle \boldsymbol{f}(\boldsymbol{x}, t, \boldsymbol{\xi}), \psi_j(\boldsymbol{\xi})\rangle \ , \quad j = 0, ..., P \ . \tag{3.42}$$

By taking the inner product defined in Equation (3.27), we project Equation (3.41) thus onto each basis polynomial ψ_i. Thanks to the orthogonality of the PC basis polynomials, the error comes only from the truncation (Equation (3.38)).

Moreover, the stochastic partial differential equation (Equation (3.37)) reduces to a system consisting of the coupled deterministic partial differential equations weighted by the coefficients, which are formed from the stochastic Galerkin projection.

Stochastic Linear System

We shall first look at the deterministic partial differential operator:

$$\mathcal{L}(\boldsymbol{a}(\boldsymbol{x}, t))\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{f}(\boldsymbol{x}, t) \ , \quad \text{in}\, \mathcal{D} \times [0, T] \ . \tag{3.43}$$

$\forall \boldsymbol{u} \in \mathcal{V}$. $\mathcal{V}$ is an appropriate solution space which can accommodate the solution vector $\boldsymbol{u}$. In this work, the deterministic PDE system is solved by applying the finite element method with time-stepping scheme. The numerical solution of Equation (3.43) needs to lay on a finite-dimensional functional space $\mathcal{V}^h \subset \mathcal{V}$. The finite element discretization leads, possibly after a linearization in case of nonlinear $\mathcal{L}$, to a linear system as follows:

$$A\boldsymbol{u} = \boldsymbol{b} \ . \tag{3.44}$$

A is a $(N \times N)$ matrix, $N = dim(\mathcal{V}^h)$. $\boldsymbol{u}$ and $\boldsymbol{b}$ are vectors of a dimension $\mathbb{R}^N$. If we introduce the random inputs as in Equation (3.37), Equation (3.44) can be extended to be a stochastic linear system, it is given as:

$$A(\boldsymbol{\xi})\boldsymbol{u}(\boldsymbol{\xi}) = \boldsymbol{b}(\boldsymbol{\xi}) \ . \tag{3.45}$$

The stochastic solution $\boldsymbol{u}(\boldsymbol{\xi})$ is in the space $\mathbb{R}^N \bigotimes \mathcal{S}^P$, which is a subspace of $\mathbb{R}^N \bigotimes L^2(\Theta, \mathbb{P})$. The coupled system of Equation (3.45) can be expressed:

$$\begin{bmatrix} A_{00} & \cdots & A_{0P} \\ \vdots & \ddots & \vdots \\ A_{P0} & \cdots & A_{PP} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_0 \\ \vdots \\ \boldsymbol{u}_P \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_0 \\ \vdots \\ \boldsymbol{b}_P \end{bmatrix} \ . \tag{3.46}$$

The latter equation is viewed as a spectral problem consisted of several PC modes $\boldsymbol{u}_i, i = 0, 1, ..., P$. The sub-matrices above have a similar structure as in the deterministic case (Equation (3.44)). $\boldsymbol{b}_i$ is computed by taking the inner product (Equation (3.27)) as follow:

$$\boldsymbol{b}_i := \langle \boldsymbol{b}, \psi_i \rangle \; . \tag{3.47}$$

The determination of the sub-matrix A_{ij} depends on the property of the stochastic differential operator $\mathcal{L}(\boldsymbol{a}(\boldsymbol{x}, t, \boldsymbol{\xi}))$, it has an important effect on the sparsity structure of the system matrix of Equation (3.46). To be more specific, after the Galerkin projection procedure, we obtain higher order tensor structures (≥ 3), because the orthogonality occurs only mutually between two polynomials. Therefore, the tensor structure introduces more non-zero elements by means of the sub-matrices into the system matrix in Equation (3.46).

Structure Pattern for the Linear Case

A special case is that the linear operator $A(\boldsymbol{\xi})$ is independent from the random inputs, i.e. the influence of the random inputs appears purely on the right-hand side ($\boldsymbol{b}(\boldsymbol{\xi})$) of Equation (3.46). It means that the SPDE (Equation (3.37)) is a deterministic system with uncertain boundary conditions or/and uncertain external forcing terms, and $\mathcal{L}$ is nothing but a deterministic operator. Equation (3.46) is simplified as follows:

$$\begin{bmatrix} A_0 & 0 & \cdots & \cdots & 0 \\ 0 & A_0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & A_0 & 0 \\ 0 & \cdots & \cdots & 0 & A_0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_{P-1} \\ \boldsymbol{u}_P \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_0 \\ \boldsymbol{b}_1 \\ \vdots \\ \boldsymbol{b}_{P-1} \\ \boldsymbol{b}_P \end{bmatrix} . \tag{3.48}$$

The global matrix is then only a diagonal matrix filled by sub-matrix A_0, and its inverse is just the inversion of sub-matrices accordingly. Therefore, we obtain essentially a decoupled system, each Polynomial Chaos mode can be obtained in a same manner by solving the deterministic problem (Equation (3.44)), i.e.:

$$\boldsymbol{u}_i = A_0^{-1} \boldsymbol{b}_i \; . \tag{3.49}$$

The spectral problem generated by this special case only needs to be solved for a set of $(P+1)$ linear systems with a matrix size $(N \times N)$. As they are independent, $(P+1)$ systems can be computed sequentially or in parallel.

Structure Pattern for the General Case

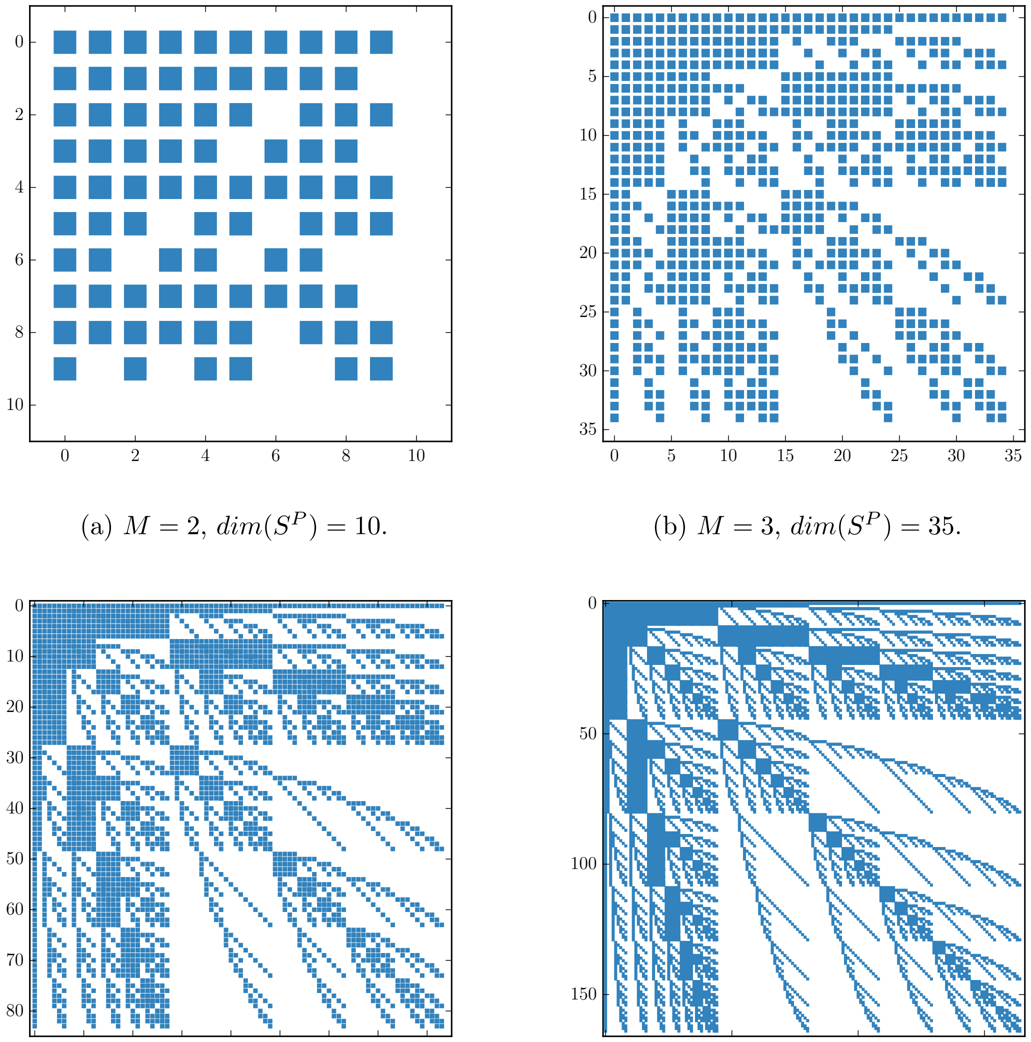
In contrary to the linear case, the matrix $A(\boldsymbol{\xi})$ is no longer a deterministic operator when the product of two or more stochastic quantities appear in the SPDE system. For the case that only the product of two random properties arises, the third-order tensor C_{ijk} comes into play due to the stochastic Galerkin projection:

$$A_{ij} := \sum_{k=0}^{P} A_k C_{kji} \; , \tag{3.50}$$

where,

$$C_{ijk} := \frac{\langle \psi_i \psi_j, \psi_k \rangle}{\langle \psi_k, \psi_k \rangle} \; . \tag{3.51}$$

The third-order tensor C_{ijk} is one of the essential ingredients for the generalized Polynomial Chaos expansion technique, particularly when the governing equations exhibit nonlinear terms with respect to the stochastic parameters. C_{ijk} is named as the Galerkin tensor or the multiplication tensor [120]. This tensor is symmetric with respect to the first two indices, i.e. $C_{ijk} = C_{jik}$,



(a) $M = 2$, $dim(S^P) = 10$.

(b) $M = 3$, $dim(S^P) = 35$.

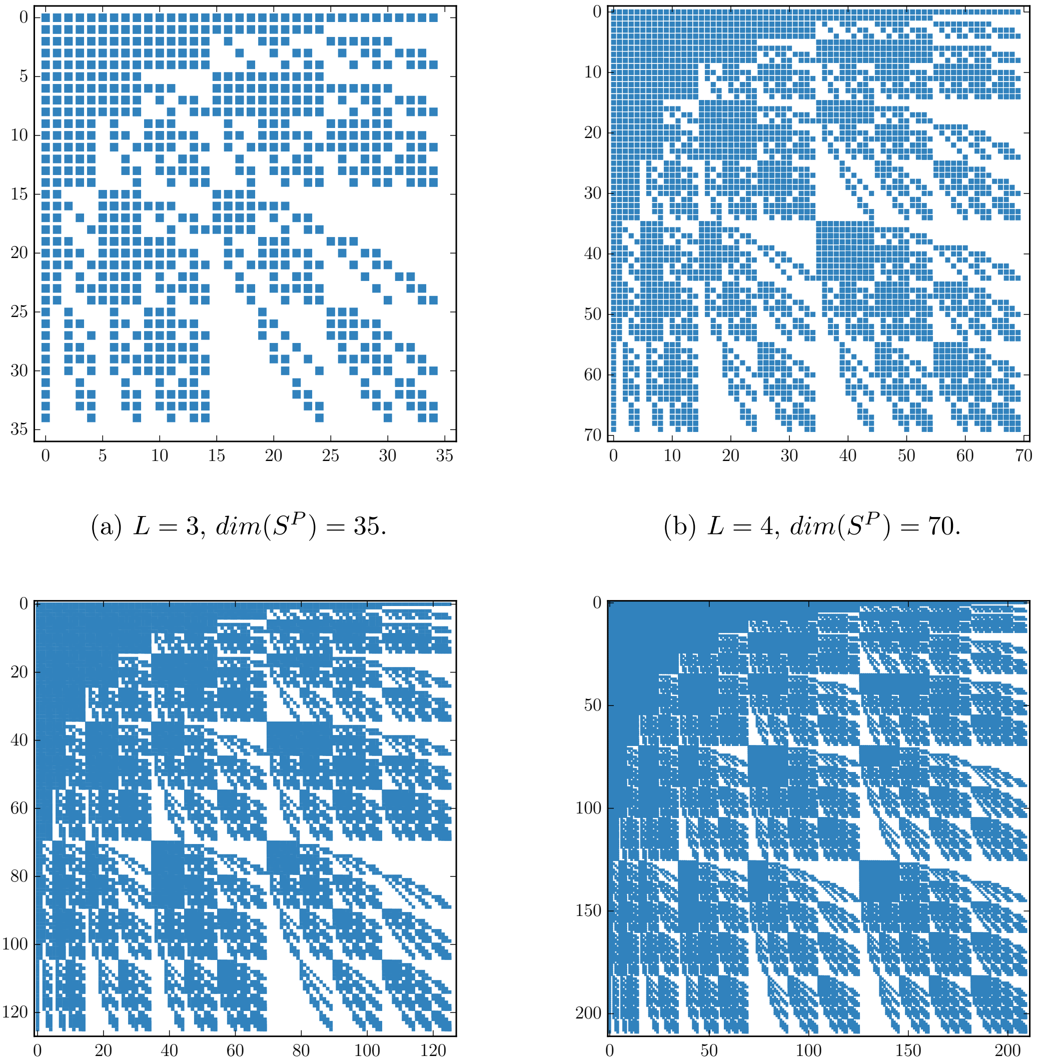
(c) $M = 4$, $dim(S^P) = 84$.

(d) $M = 5$, $dim(S^P) = 165$.

Figure 3.2: The sparsity structure of the global matrix when $L = 3$ with the different number of random variables M, each block represents A_{ij}.

and the symmetric property affects also the global system matrix, i.e. $A_{ij} = A_{ji}$. In general, there are no symmetries respecting the third index k.

Because of the orthogonality of the stochastic basis, many entries of the Galerkin tensor on the off-diagonal are zero. Figures 3.2 and 3.3 illustrate the sparsity structure of the block system of the spectral problem (Equation (3.48)). Hereby, we define a sparsity ratio R_s of the matrix, which is calculated by the number of non-zeros block entries over the size of total blocks $(P+1)^2$. Figure 3.4 shows the sparsity ratio R_s and the number of non-zero entries. One observes that R_s increases when the expansion order grows, but becomes small when we increase the number of random variables for a fixed expansion order. Anyhow, the size of the block system expands exponentially as claimed in Equation (3.30), the number of the non-zero entries are therefore very large. In spite of the spectral convergence behavior of gPCE, the curse of the dimensionality becomes important when M and L increase. In practical situations, $dim(\mathcal{V}^h) \gg dim(\mathcal{S}^P)$, as we envisage applying the finite element method for our PDEs, the spatial discretization induces a large system in order to well represent the physical domain and obtain the accurate solution. For



(a) $L = 3$, $dim(S^P) = 35$.

(b) $L = 4$, $dim(S^P) = 70$.

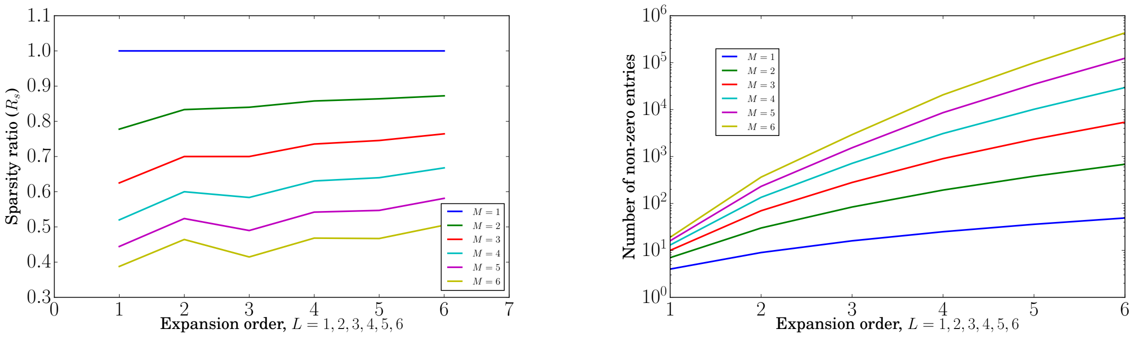
(c) $L = 5$, $dim(S^P) = 126$.

(d) $L = 6$, $dim(S^P) = 210$.

Figure 3.3: The sparsity structure of the global matrix when $M = 4$ with the different polynomial degree L, each block represents A_{ij}.

instance, if we consider a typical situation where $M = 10$, $L = 10$, it results $dim(\mathcal{S}^P) = 184756$. The dimension of this stochastic space is equivalent to the number of unknowns for a 2D Poisson problem on a unit square domain discretized by 400×400 grids with using only the linear elements. Therefore, we keep normally the gPCE to be dealing with a small dimension of the random variables and a low Chaos Polynomial degree.

Note that, Figures 3.2 to 3.4 demonstrate the situation when the random variables have the same polynomial order as the expansion order in the stochastic basis. However, in many cases, the first order or lower order expansions are used for describing the random inputs. The Galerkin tensor becomes sparser, because the integration of certain multivariate polynomials is zero. Figure 3.5 illustrates the sparsity structure of the spectral problem based on the selected combinations of M and L when the expansion order of the random inputs L_t has only the first order, viz. $L_t = 1$. Hence, the block structure is more diagonal dominated. For our blood pump modelization, the random input are also chosen to be modeled with the first order expansion. Therefore, we concern in this work rather with the structure in Figure 3.5, and how to solve such



(a) The sparsity ratio of the block system.

(b) The number of non-zeros entries.

Figure 3.4: The sparsity ratio and the number of non-zero entries evolve along the dimension of random inputs M and the expansion order L.

a coupled structure will be discussed in Chapters 4 and 5.

For the practical purpose, the Galerkin tensor C_{ijk} is computed as a pre-processing step instead of calculating each component during the assembly process. The construction of the multi-index is the first procedure once the dimension of random variables and the expansion order are defined. Throughout the evaluation of multidimensional polynomials, one can store the integration of the polynomial at each order in regard to the different probability distributions in advance. It avoids the repeated computations in the situation where the random variables have the same probability distribution. The implementation details can be found in [62, 120].

Remark 3.2.3. *Considering the double product $m(\boldsymbol{\xi}) := u(\boldsymbol{\xi})v(\boldsymbol{\xi})$, $u, v \in \mathcal{S}^P$ with the pre-defined expansion:*

$$u(\boldsymbol{\xi}) := \sum_{i=0}^{P} u_i \psi_i(\boldsymbol{\xi}) \ , \quad v(\boldsymbol{\xi}) := \sum_{i=0}^{P} v_i \psi_i(\boldsymbol{\xi}) \ . \tag{3.52}$$

Then,

$$m(\boldsymbol{\xi}) := \sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j \psi_i(\boldsymbol{\xi}) \psi_j(\boldsymbol{\xi}) \ . \tag{3.53}$$

It implies, $m(\boldsymbol{\xi}) \notin \mathcal{S}^P$, yet, it is always in $L^2(\Theta, \mathbb{P})$, if Θ is bounded. By the Galerkin projection:

$$m_k := \frac{\langle m, \psi_k \rangle}{\langle \psi_k, \psi_k \rangle} = \sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j C_{ijk} \ , \quad \forall k = 0, ..., P \ , \tag{3.54}$$

or,

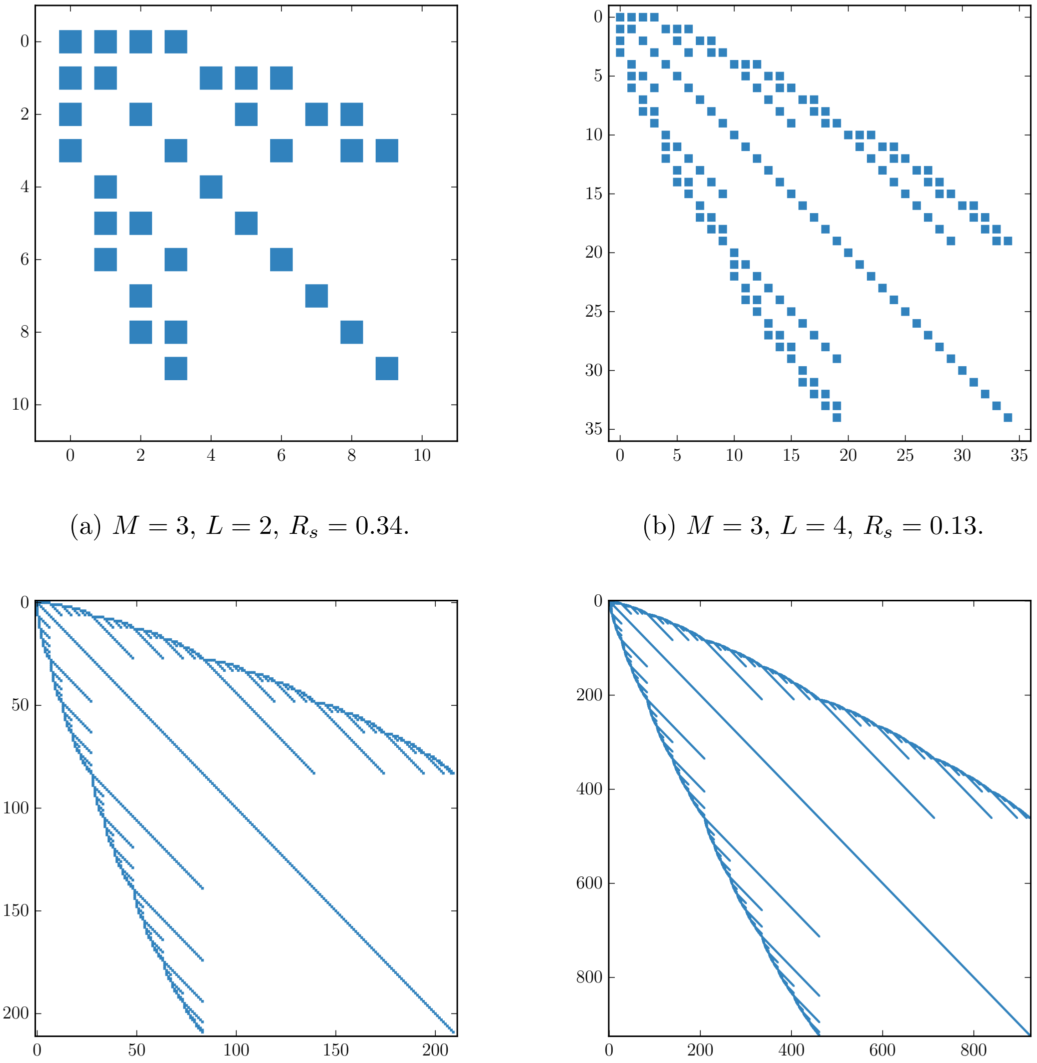
$$\tilde{m} := \sum_{k=0}^{P} (\sum_{i=0}^{P} \sum_{j=0}^{P} u_i v_j C_{ijk}) \psi_k \ . \tag{3.55}$$

The result of Equation (3.54) *is called the Galerkin product of u and v and is an element of $\mathcal{S}^P$.*

The Galerkin tensor plays a fundamental role in the generalized Polynomial Chaos expansion technique, it serves for the most common physical models. Still, some physical phenomena also possess a higher-order nonlinearity, therefore, a higher-order tensor needs to be constructed.

Let us consider a triple product $m(\boldsymbol{\xi}) := u(\boldsymbol{\xi})v(\boldsymbol{\xi})w(\boldsymbol{\xi})$, $u, v, w \in \mathcal{S}^P$:

$$u(\boldsymbol{\xi}) := \sum_{i=0}^{P} u_i \psi_i(\boldsymbol{\xi}) \ , \quad v(\boldsymbol{\xi}) := \sum_{i=0}^{P} v_i \psi_i(\boldsymbol{\xi}) \ , \quad w(\boldsymbol{\xi}) := \sum_{i=0}^{P} w_i \psi_i(\boldsymbol{\xi}) \ . \tag{3.56}$$



(a) $M = 3$, $L = 2$, $R_s = 0.34$.

(b) $M = 3$, $L = 4$, $R_s = 0.13$.

(c) $M = 6$, $L = 4$, $R_s = 0.28$.

(d) $M = 6$, $L = 6$, $R_s = 0.01$.

Figure 3.5: The sparsity structure of the global matrix when the random variables are represented by the first order polynomials ($L_t = 1$).

Then,

$$m(\boldsymbol{\xi}) := \sum_{i=0}^{P}\sum_{j=0}^{P}\sum_{k=0}^{P} u_i v_j w_k \psi_i(\boldsymbol{\xi})\psi_j(\boldsymbol{\xi})\psi_k(\boldsymbol{\xi}) \ . \tag{3.57}$$

Using the Galerkin projection:

$$m_l := \sum_{i=0}^{P}\sum_{j=0}^{P}\sum_{k=0}^{P} \frac{\langle \psi_i\psi_j\psi_k, \psi_l\rangle}{\langle \psi_l, \psi_l\rangle} u_i v_j w_k = \sum_{i=0}^{P}\sum_{j=0}^{P}\sum_{k=0}^{P} u_i v_j w_k C_{ijkl} \ , \quad \forall l = 0, ..., P \tag{3.58}$$

or,

$$\tilde{m} := \sum_{l=0}^{P}(\sum_{i=0}^{P}\sum_{j=0}^{P}\sum_{k=0}^{P} u_i v_j w_k C_{ijkl})\psi_l \ . \tag{3.59}$$

$C_{ijkl} := \frac{\langle \psi_i \psi_j \psi_k, \psi_l \rangle}{\langle \psi_l, \psi_l \rangle}$ is the fourth-order tensor.

The construction of this fourth-order tensor follows the same principle as in third-order tensor (Galerkin tensor). Since the computation becomes $(P+1)$ times more expensive as well as the storage, the complexity grows exponentially with the order of the tensor. In practice, we approximate the higher-order tensor with the help of the Galerkin tensor rather than evaluating it exactly.

The fourth-order Galerkin product can be approximated by:

$$\begin{aligned} \tilde{m} &\approx \sum_{l=0}^{P} \big(\sum_{m,k=0}^{P} \big(\sum_{i,j=0}^{P} u_i v_j C_{ijm} \big) w_k C_{mkl} \big) \psi_l \\ &\approx \sum_{l=0}^{P} \big(\sum_{i,j,m,k=0}^{P} u_i v_j w_k C_{ijm} C_{mkl} \big) \psi_l \ , \end{aligned} \tag{3.60}$$

where, $C_{ijm} := \frac{\langle \psi_i \psi_j, \psi_m \rangle}{\langle \psi_m, \psi_m \rangle}$, $C_{mkl} := \frac{\langle \psi_m \psi_k, \psi_l \rangle}{\langle \psi_l, \psi_l \rangle}$. Therefore, the higher order tensor can be constructed recursively as in Equation (3.60).

3.3 Spectral-Stochastic Finite Element Method for Flow Problem

The previous two sections focus on the stochastic problem setting and the generalized Polynomial Chaos expansion approach. This section is dedicated to the spectral-stochastic finite element method (SSFEM), because we intend to solve the flow problem within the blood pump device by using the finite element method (FEM). This section will concentrate on the discretization of the deterministic space and the probability space for the Variational Multiscale model with random data and the stochastic Galerkin projection system.

3.3.1 The Deterministic Incompressible Navier-Stokes Equations

It is reasonable to model the flow within a blood pump device by utilizing the unsteady incompressible Navier-Stokes equations, because the blood represents well the incompressibility and no important additional physical driving forces in the system. For our case, the main driving force is the rotation of the rotor, and it introduces whereas no additional physical effects, e.g. thermodynamic forces, electrodynamic forces. We consider hence an instationary flow defined in $\mathcal{D} \times [0,T] \subset \mathbb{R}^d \times \mathbb{R}$, $d = 2,3$ without considering the blood pump scenario. The unsteady incompressible Navier-Stokes equations with the homogeneous boundary condition read:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} - \frac{\mu}{\rho} \Delta \boldsymbol{u} + \frac{1}{\rho} \nabla p = \boldsymbol{f} \ , \qquad \text{in } \mathcal{D} \times [0,T] \ , \tag{3.61a}$$

$$\nabla \cdot \boldsymbol{u} = 0 \ , \qquad \text{in } \mathcal{D} \times [0,T] \ , \tag{3.61b}$$

$$\boldsymbol{u} = 0 \ , \qquad \text{in } \partial\mathcal{D} \times [0,T] \ , \tag{3.61c}$$

$$\boldsymbol{u}(t=0) = \boldsymbol{u}_0 \ , \qquad \text{in } \mathcal{D} \ . \tag{3.61d}$$

Here, $\mathcal{D}$ is a bounded domain with Lipschitz boundary. $\boldsymbol{u} \in \mathcal{D} \times [0,T]$ is the velocity of the fluid field, $p \in \mathcal{D} \times [0,T]$ states for the pressure, $T < \infty$ is the final time. Regarding the assumption of the incompressibility, $\rho \in \mathbb{R}^+$ is constant, it also means that we utilize only the Newtonian fluid, as μ is considered to be constant. $\mu \in \mathbb{R}^+$ is the dynamic viscosity, which represents the internal resistance or the friction to the flow. Sometimes, we also prefer to operate with the kinematic viscosity $\nu := \frac{\mu}{\rho} \in \mathbb{R}^+$, which can be considered as a measure for the diffusivity of the momentum. $\boldsymbol{f} \in \mathcal{D} \times [0,T]$ is the external body force. $\boldsymbol{u}_0$ indicates the initial condition for the velocity, it is defined on the whole domain $\mathcal{D}$ only when $t = 0$.

If $\boldsymbol{u}$, p are classical solutions for Equation (3.61), which means that $\boldsymbol{u}$ is twice differentiable in space and once differentiable in time, continuous on the boundary and initial condition. p is once differentiable in space. We obtain the energy equality by multiplying the momentum equation (Equation (3.61)) with $\boldsymbol{u}$ and integrating over $[0,T]$, it yields:

$$\frac{1}{2}\|\boldsymbol{u}(t)\|^2 + \int_0^t \frac{\mu}{\rho}\|\nabla \boldsymbol{u}(t')\|^2 dt' = \frac{1}{2}\|\boldsymbol{u}_0\|^2 + \int_0^t (\boldsymbol{f}(t'), \boldsymbol{u}(t'))dt' \ . \tag{3.62}$$

It implies the balance between two sums of energies, one side is contributed by the kinetic energy and the total dissipated energy, another side is provided from the initial kinetic energy and the total input energy. $(\cdot,\cdot)$ here stands for the inner product of $L(\mathcal{D})$.

Remark 3.3.1. *The incompressible NSE is derived from Newton's second law with a constant density. The momentum conservation equation (Equation* (3.61a)*) can be considered as a special form of the Cauchy momentum equation [153]:*

$$\frac{D\boldsymbol{u}}{Dt} = \frac{1}{\rho}\nabla \cdot \boldsymbol{\sigma} + \boldsymbol{f} \ ,$$

$\frac{D}{Dt} := \frac{\partial}{\partial t} + (\boldsymbol{u} \cdot \nabla)$ *denotes the material derivative operator,* $\boldsymbol{\sigma}$ *is the stress tensor. More information about the continuum mechanics theory can be found in e.g. [37].*

One of the fundamental concepts for understanding the incompressible Navier-Stokes equations in the mathematical framework is the weak solution. We define at first the function spaces which are used in the following sections:

$$\begin{aligned}
V(\mathcal{D}) &:= \left\{ \boldsymbol{v} \in H_0^1(\mathcal{D})^d \right\} , \\
Q(\mathcal{D}) &:= \left\{ q \in L^2(\mathcal{D}) \mid \int_{\mathcal{D}} q \, dx = 0 \right\} , \\
H_{div}(\mathcal{D}) &:= \left\{ \boldsymbol{v} \in L^2(\mathcal{D})^d \mid \nabla \cdot \boldsymbol{v} = 0, \boldsymbol{v} \cdot \boldsymbol{n} = 0 \text{ on } \partial\mathcal{D} \right\} , \\
V_{div}(\mathcal{D}) &:= \left\{ \boldsymbol{v} \in V(\mathcal{D}) \mid \nabla \cdot \boldsymbol{v} = 0, \boldsymbol{v} = 0 \text{ on } \partial\mathcal{D} \right\} .
\end{aligned}$$

Definition 3.3.1 (The variational formulation of the incompressible NSE)**.** *Find* $(\boldsymbol{u}, p) \in V(\mathcal{D}) \times Q(\mathcal{D})$*, such that:*

$$(\frac{\partial \boldsymbol{u}}{\partial t}, \boldsymbol{v}) + (\boldsymbol{u} \cdot \nabla \boldsymbol{u}, \boldsymbol{v}) + \frac{\mu}{\rho}(\nabla \boldsymbol{u}, \nabla \boldsymbol{v}) - \frac{1}{\rho}(p, \nabla \cdot \boldsymbol{v}) = (\boldsymbol{f}, \boldsymbol{v}) \ , \tag{3.64a}$$

$$(\nabla \cdot \boldsymbol{u}, q) = 0 \ . \tag{3.64b}$$

$\forall \boldsymbol{u} \in V(\mathcal{D})$*,* $\forall q \in Q(\mathcal{D})$*.*

Definition 3.3.2 (The weak solution of NSE)**.** *Given any* $\boldsymbol{f} \in \boldsymbol{L}^2([0,T], \mathcal{D})$ *and* $\boldsymbol{u}_0 \in H_{div}$*, find* $\boldsymbol{u} \in L^2([0,T], V_{div}) \cap L^\infty([0,T], H_{div})$*, such that:*

$$\frac{\partial}{\partial t}(\boldsymbol{u}, \boldsymbol{v}) + (\boldsymbol{u} \cdot \nabla \boldsymbol{u}, \boldsymbol{v}) + \frac{\mu}{\rho}(\nabla \boldsymbol{u}, \nabla \boldsymbol{v}) = (\boldsymbol{f}, \boldsymbol{v}) \ , \tag{3.65a}$$

$$\boldsymbol{u}(t = 0) = \boldsymbol{u}_0 \ . \tag{3.65b}$$

$\forall \boldsymbol{v} \in V_{div}$*.*

It is well known that the existence of the weak solution (Definition 3.3.2) is guaranteed by the Leray-Hopf theorem, it is given as:

Theorem 3.3.1 (The existence of the weak solutions)**.** *Let $\mathcal{D} \in \mathbb{R}^d, d = 2, 3$. Given any $\boldsymbol{u}_0 \in H_{div}(\mathcal{D})$ and $\boldsymbol{f} \in \boldsymbol{L}^2([0,T], \mathcal{D})$, there exists at least one weak solution accordingly to Definition 3.3.2. This solution satisfies the following two additional properties:*

- $\frac{1}{2}\|\boldsymbol{u}(t)\|_2^2 + \int_0^t \frac{\mu}{\rho}\|\nabla \boldsymbol{u}(t')\|_2^2 dt' \leq \frac{1}{2}\|\boldsymbol{u}_0\|_2^2 + \int_0^t (\boldsymbol{f}(t'), \boldsymbol{u}(t'))dt' \quad t \in [0, T]$.
- $\lim_{t\to\infty} \|\boldsymbol{u}(t) - \boldsymbol{u}_0\|_2 = 0$.

The theorem is proven by applying the Faedo-Galerkin method, and the detailed proof can be found in [59, 111, 84, 163].

Equation (3.62) denotes the energy equality which is satisfied the strong solution of Equation (3.61), as well as the Galerkin approximation to the weak solution. The inequality from theorem 3.3.1 associated with the weak solution is the current optimal result for the weak solutions.

Remark 3.3.2. *The Leray-Hopf theorem (Theorem 3.3.1) shows the existence of the weak solution of the Navier-Stokes equations with the homogeneous boundary condition, and the uniqueness for the two-dimension situation is also proven [114]. However, the uniqueness of the weak solution in three-dimension is not clear yet based on Theorem 3.3.1. There is also another way to provide the uniqueness of the weak solution [163] by assuming $\boldsymbol{u} \in L^2([0,T], H^1_{div}) \cap L^\infty([0,T], H_{div})$ and $\boldsymbol{u} \in L^8([0,T], \boldsymbol{L}^4(\mathcal{D}))$, but for this solution space, the existence is still unclear.*

Remark 3.3.3. *The results above are only valid for a homogeneous boundary condition, in contrast, the mixed boundary conditions need to be applied for the blood pump simulation, e.g. the inflow boundary is applied with a Poisseuille profile condition, the outflow is considered as the "do-nothing" boundary condition. Still, a complete proof to the existence for a such situation is not clear.*

3.3.2 Residual-Based Variational Multiscale Method for Blood Pump Modeling

As mentioned in Chapter 2, we restrict ourselves to the two-scale Variational Multiscale method, more specifically, the two-scale residual-based VMS model. This model creates a single set of equations, which consists of the resolved scale and the unresolved scale. Furthermore, the solution on the subgrid scale is approximated and inserted in the resolved scale with the help of the additional stabilization formulations. Moreover, the residual-based techniques do not have to rely on the statistical hypothesis of the turbulent flow, to put it differently, there are no eddy viscosity models involved and the inertial interactions are already covered by the subgrid models. It is also contrast to the three-scale VMS models that additional eddy viscosity terms are obliged to be coupled on the small resolved scale.

Before presenting the full two-scale residual-based Variational Multiscale method for the blood pump modeling, we provide the finite element spaces which we wish to work with. V^h and Q^h are the finite element space for the velocity and the pressure, respectively. V^h and Q^h can be constructed on general triangulations with both first finite elements under the conforming finite element method framework.

$$V^h \subset V(\mathcal{D}) = \left\{ \boldsymbol{v} \in H_0^1(\mathcal{D})^d \right\} ,$$
$$Q^h \subset Q(\mathcal{D}) = \left\{ q \in L^2(\mathcal{D}) \mid \int_{\mathcal{D}} q \, dx = 0 \right\} .$$

The weak formulation is given as:

Find $\boldsymbol{u}_h \in V^h$, $p_h \in Q^h$:

$$\begin{aligned}
&(\frac{\partial \boldsymbol{u}_h}{\partial t}, \boldsymbol{v}_h) + ((\hat{\boldsymbol{u}}_h - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h, \boldsymbol{v}_h) && && (3.67a)\\
&+\frac{\mu}{\rho}(\nabla \boldsymbol{u}_h, \nabla \boldsymbol{v}_h) - \frac{1}{\rho}(p_h, \nabla \cdot \boldsymbol{v}_h) && && \\
&+(\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, (\hat{\boldsymbol{u}}_h - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{v}_h) + (\tau_C r_{C,h}, \nabla \cdot \boldsymbol{v}_h) = (\boldsymbol{f}_h, \boldsymbol{v}_h)\ , && \text{in } \mathcal{D}^t \times [0,T]\ , && \\
&(\nabla \cdot \boldsymbol{u}_h, q_h) + (\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \nabla q_h) = 0\ , && \text{in } \mathcal{D}^t \times [0,T]\ . && (3.67b)
\end{aligned}$$

$\forall \boldsymbol{v}^h \in V^h$, $\forall q^h \in Q^h$. V^h and Q^h are both finite element spaces. Additionally,

$$\begin{aligned}
\hat{\boldsymbol{u}}_h &= \boldsymbol{u}_h - \boldsymbol{\tau}_M \boldsymbol{r}_{M,h}\ , && \text{in } \mathcal{D}^t \times [0,T]\ , && (3.68a)\\
\boldsymbol{r}_{M,h} &= \frac{\partial \boldsymbol{u}_h}{\partial t} + (\boldsymbol{u}_h - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h - \frac{\mu}{\rho} \Delta \boldsymbol{u}_h && && (3.68b)\\
&\quad + \frac{1}{\rho} \nabla p_h - \boldsymbol{f}_h\ , && \text{in } \mathcal{D}^t \times [0,T]\ , && \\
r_{C,h} &= \nabla \cdot \boldsymbol{u}_h\ , && \text{in } \mathcal{D}^t \times [0,T]\ , && (3.68c)\\
\boldsymbol{u}_h^r &= \boldsymbol{d} \times \boldsymbol{\omega}\ , && \text{in } \mathcal{D}^t_{rot} \times [0,T]\ , && (3.68d)\\
\boldsymbol{u}_h^r &= \boldsymbol{0}\ , && \text{in } \mathcal{D}^t_{stat} \times [0,T]\ , && (3.68e)\\
\boldsymbol{u}_h &= \boldsymbol{u}_I\ , && \text{on } \Gamma^t_{in} \times [0,T]\ , && (3.68f)\\
(-\mathbb{1} p_h + \frac{\mu}{\rho} \nabla \boldsymbol{u}_h) \cdot \boldsymbol{n} &= \boldsymbol{0}\ , && \text{on } \Gamma^t_{out} \times [0,T]\ , && (3.68g)\\
\boldsymbol{u}_h &= \boldsymbol{d} \times \boldsymbol{\omega}\ , && \text{on } \Gamma^t_{rotor} \times [0,T]\ , && (3.68h)\\
\boldsymbol{u}_h &= \boldsymbol{0}\ , && \text{on } \Gamma^t_{wall} \times [0,T]\ . && (3.68i)
\end{aligned}$$

$(\cdot, \cdot)$ represents the inner product, the residuals $\boldsymbol{r}_{M,h}$, $r_{C,h}$ are defined in a similar way as in the previous chapter (Chapter 2). The superscript t on the domains and the boundaries indicates that the mesh can alter along the time because of the shear layer update approach (Section 2.3). $\boldsymbol{u}^r$ is the revolving speed and acts only on the rotating domain. The computing domain is divided into the rotating part and the static part, i.e., $\mathcal{D}^t = \mathcal{D}^t_{rot} \cup \mathcal{D}^t_{stat}$, $\mathcal{D}^t_{rot} \cap \mathcal{D}^t_{stat} = \emptyset$. Regarding the boundaries, $\partial\mathcal{D}^t = \Gamma^t_{in} \cup \Gamma^t_{out} \cup \Gamma^t_{rotor} \cup \Gamma^t_{wall}$. $\boldsymbol{u}_I$ indicates for the inflow boundary condition, whose explicit form will be given in the following chapter as it is one of three uncertain sources under our consideration. We apply the "do-nothing" boundary condition on the outflow boundary Γ^t_{out}. The "do-nothing" condition implies that the pressure has mean zero value on the outflow boundary [68, 67]. The "no-slip" condition is used on the rigid walls Γ^t_{wall}, and the rotor's surface undergoes the rotating speed. We show the discretized weak formulation directly because the two-scale residual-based VMS can only be seen in the discretized weak form, whereas, the further details about the spatial discretization will be shown in the following subsection.

3.3.3 Discretization of the Deterministic VMS

This subsection is dedicated to the temporal and spatial discretization of the deterministic VMS formulation (Equation (3.67)) by using the finite element method, later, it will be extended under the SSFEM framework in Section 3.3.4.

Concerning the unsteady NSE, we utilize the Rothe time discretization method [139, 101]. This method proposes that a discretization of the time variable must be considered first, then the standard finite element techniques are used to solve the derived stationary PDEs. Therefore, varying the type of trial functions is possible, i.e. an independent spatial discretization between each time step is allowed. The Rothe method differs from another common time stepping approach "method of lines" (MOL), which employs the spatial discretization before the time discretization, after that, a set of Ordinary Differential Equations (ODEs) needs to be solved. However, since the MOL is clearly unsuitable for the mesh adaptation, we do not consider it for the blood pump modelization.

Time Discretization

Concerning the temporal discretization, it is common to employ the standard techniques developed for the Ordinary Differential Equations (ODEs) [47, 169]. We utilize the Crank-Nicolson scheme [43] to approximate the transient solution from one time step to another. This method can be regarded as a finite difference approach and it is a second-order A-stable scheme. The basic idea behind the Crank-Nicolson scheme is to average with a pre-defined weight the explicit scheme at the time step n and $n+1$, i.e., the θ-scheme with $\theta = 0.5$. It can be demonstrated with an abstract unsteady initial value problem. For instance:

$$\frac{du}{dt} + F(u) = f(t) \ , \quad t \in [0, T], \quad u(0) = u_0 \ , \tag{3.69}$$

where $u(t) \in \mathbb{R}^N$. The Crank-Nicolson method reads:

$$\frac{u^{n+1} - u^n}{\Delta t} + \theta F(u^{n+1}) + (1-\theta) F(u^n) = \theta f(t_{n+1}) + (1-\theta) f(t_n) \ , \quad \theta = 0.5 \ . \tag{3.70}$$

Noted that, $\theta > 0$ yields an implicit scheme, it often allows a larger time step size comparing with the explicit methods. In Equation (3.70), Δt is the time step size, and it is defined as $\Delta t := t_{n+1} - t_n$. We consider an equal-sized time length, i.e. $T := N_T \Delta t$, $t_n := n\Delta t$, $t = 0, ..., N_T$, $N_T \in \mathbb{R}$. In [56], they suggest using a semi-implicit Backward Differentiation Formulas (BDF) time discretization in order to cope with the higher order nonlinear terms which appear in Jacobian assembly for the VMS. The way we employ here is to embed two stabilization terms into the advection velocity (Equations (2.34) and (2.35)) and consider the modified velocity $\hat{\boldsymbol{u}}$ as a whole. Accordingly, we are able to discretize Equation (3.67) with the standard Crank-Nicolson scheme, it is given as:

Seeking $\boldsymbol{u}_h^{n+1} \in V^h$, $p_h^{n+1} \in Q^h$, such that,

$$\begin{aligned}
&(\boldsymbol{u}_h^{n+1} - \boldsymbol{u}_h^n, \boldsymbol{v}_h) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(3.71a)}\\
&+ \Delta t\, \theta\, ((\hat{\boldsymbol{u}}_h^{n+1} - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h^{n+1}, \boldsymbol{v}_h) + \Delta t\, (1-\theta)\, ((\hat{\boldsymbol{u}}_h^{n} - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h^{n}, \boldsymbol{v}_h)\\
&+ \Delta t\, \theta\, (\frac{\mu}{\rho} \nabla \boldsymbol{u}_h^{n+1}, \nabla \boldsymbol{v}_h) + \Delta t\, (1-\theta)\, (\frac{\mu}{\rho} \nabla \boldsymbol{u}_h^{n}, \nabla \boldsymbol{v}_h)\\
&- \Delta t\, (\frac{1}{\rho} p_h^{n+1}, \nabla \cdot \boldsymbol{v}_h)\\
&- \Delta t\, \theta\, (\boldsymbol{f}_h^{n+1}, \boldsymbol{v}_h) - \Delta t\, (1-\theta)\, (\boldsymbol{f}_h^{n}, \boldsymbol{v}_h)\\
&+ (\boldsymbol{\tau}_M [(\boldsymbol{u}_h^{n+1} - \boldsymbol{u}_h^n) + \Delta t\, \frac{1}{\rho} \nabla p_h^{n+1}\\
&+ \Delta t\, \theta\, ((\boldsymbol{u}_h^{n+1} - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h^{n+1} - \frac{\mu}{\rho} \Delta \boldsymbol{u}_h^{n+1} - \boldsymbol{f}_h^{n+1})\\
&+ \Delta t\, (1-\theta)\, ((\boldsymbol{u}_h^{n} - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{u}_h^{n} - \frac{\mu}{\rho} \Delta \boldsymbol{u}_h^{n} - \boldsymbol{f}_h^{n})], (\hat{\boldsymbol{u}}_h^{n+1} - \boldsymbol{u}_h^r) \cdot \nabla \boldsymbol{v}_h)\\
&+ \Delta t\, (\tau_C \nabla \cdot \boldsymbol{u}_h^{n+1}, \nabla \cdot \boldsymbol{v}_h)
\end{aligned}$$

$$
\begin{aligned}
&+ \Delta t\,\frac{1}{\rho}(\nabla \cdot \boldsymbol{u}_h^{n+1}, q_h) \qquad\qquad (3.71b)\\
&+ (\boldsymbol{\tau}_M[(\boldsymbol{u}_h^{n+1} - \boldsymbol{u}_h^{n}) + \Delta t\,\frac{1}{\rho}\nabla p_h^{n+1}\\
&+ \Delta t\,\theta\,((\boldsymbol{u}_h^{n+1} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{u}_h^{n+1} - \frac{\mu}{\rho}\Delta \boldsymbol{u}_h^{n+1} - \boldsymbol{f}_h^{n+1})\\
&+ \Delta t\,(1-\theta)\,((\boldsymbol{u}_h^{n} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{u}_h^{n} - \frac{\mu}{\rho}\Delta \boldsymbol{u}_h^{n} - \boldsymbol{f}_h^{n})], \nabla q_h) = \boldsymbol{0}\;.
\end{aligned}
$$

$\forall \boldsymbol{v}_h \in V^h$, $\forall q_h \in Q^h$.

In Equation (3.71), $(\boldsymbol{u}_h^{n+1}, p_h^{n+1})$ are the solutions for the new time step, and $(\boldsymbol{u}_h^{n}, p_h^{n})$ are for the old time step. The issue with the implicit scheme is that the system is still nonlinear after the temporal discretization. The convective terms in VMS provide the nonlinearity in the weak formulation, and an additional linearization procedure is required before applying the Newton-Raphson method [72].

The Newton scheme is an iterative method and relies on the first order terms of the Taylor series from the considered nonlinear system. The basic algorithm is demonstrated as:

$$
J_F(x_k)\,\delta x_k = -F(x_k)\;. \qquad (3.72)
$$

F is an arbitrary nonlinear system, x_k is an approximation of the solution at the k-th Newton step, δx is the increment (or the correction), $J_F(x_k)$ is the Fréchet derivative of F, i.e.:

$$
\lim_{\|z\| \to 0} \frac{\|F(x_k + z) - F(x_k) - J_F(x_k)z\|}{\|z\|} = 0\;. \qquad (3.73)
$$

The next Newton step proceeds with a new approximation $x_{k+1} := x_k + \delta x$ until it reaches certain convergence criteria.

We note that $[\boldsymbol{u}_h^{ns}, p_h^{ns}]$ are the solutions obtained from the last Newton step, and $[\boldsymbol{u}_h^{ts}, p_h^{ts}]$ are the last time step solutions, $[\boldsymbol{u}_h^{ns+1}, p_h^{ns+1}] = [\boldsymbol{u}_h^{ns} + \delta \boldsymbol{u}_h^{ns}, p_h^{ns} + \delta p_h^{ns}]$ are the solutions for the next Newton step, $[\delta \boldsymbol{u}_h^{ns}, \delta p_h^{ns}]$ are the increments between two Newton steps. Hence, we illustrate the necessary linearized terms, which occur in the Fréchet derivative or the Jacobian matrix, $J_F(\boldsymbol{u}_h^{ts}, p_h^{ts}, \boldsymbol{u}_h^{ns}, p_h^{ns}, \boldsymbol{v}_h, q_h) = [J_{F1}, J_{F2}]^T$. It is given as:

$$
\begin{aligned}
&J_{F1}(\boldsymbol{u}_h^{ts}, p_h^{ts}, \boldsymbol{u}_h^{ns}, p_h^{ns}, \boldsymbol{v}_h, q_h)[\delta \boldsymbol{u}_h^{ns}, \delta p_h^{ns}]^T := \qquad\qquad (3.74a)\\
&(\delta \boldsymbol{u}_h^{ns}, \boldsymbol{v}_h) + \Delta t\,\theta\,(\delta \boldsymbol{u}_h^{ns} \cdot \nabla \boldsymbol{u}_h^{ns}, \boldsymbol{v}_h) + \Delta t\,\theta\,((\hat{\boldsymbol{u}}_h^{ns} - \boldsymbol{u}^{r}) \cdot \nabla \delta \boldsymbol{u}_h^{ns}, \boldsymbol{v}_h)\\
&+ \Delta t\,\theta\,(\frac{\mu}{\rho}\nabla \delta \boldsymbol{u}_h^{ns}, \nabla \boldsymbol{v}_h) - \Delta t\,(\frac{1}{\rho}\delta p_h^{ns}, \nabla \cdot \boldsymbol{v}_h)\\
&+ (\boldsymbol{\tau}_M \delta \boldsymbol{u}_h^{ns}, (\hat{\boldsymbol{u}}_h^{ns} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{v}_h) + \Delta t\,\theta\,(\boldsymbol{\tau}_M \delta \boldsymbol{u}_h^{ns} \cdot \nabla \boldsymbol{u}_h^{ns}, (\hat{\boldsymbol{u}}_h^{ns} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{v}_h)\\
&+ \Delta t\,\theta\,(\boldsymbol{\tau}_M(\boldsymbol{u}_h^{ns} - \boldsymbol{u}_h^{r}) \cdot \nabla \delta \boldsymbol{u}_h^{ns}, (\hat{\boldsymbol{u}}_h^{ns} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{v}_h) - \Delta t\,\theta\,(\boldsymbol{\tau}_M \frac{\mu}{\rho}\Delta \delta \boldsymbol{u}_h^{ns}, (\hat{\boldsymbol{u}}_h^{ns} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{v}_h)\\
&+ \Delta t\,\theta\,(\boldsymbol{\tau}_M \frac{1}{\rho}\nabla \delta p_h^{ns}, (\hat{\boldsymbol{u}}_h^{ns} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{v}_h)\\
&+ (\boldsymbol{\tau}_M[(\boldsymbol{u}_h^{ns} - \boldsymbol{u}_h^{ts}) + \Delta t\,\frac{1}{\rho}\nabla p_h^{ns}\\
&+ \Delta t\,\theta\,((\boldsymbol{u}_h^{ns} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{u}_h^{ns} - \frac{\mu}{\rho}\Delta \boldsymbol{u}_h^{ns} - \boldsymbol{f}_h^{n+1})\\
&+ \Delta t\,(1-\theta)\,((\boldsymbol{u}_h^{ts} - \boldsymbol{u}_h^{r}) \cdot \nabla \boldsymbol{u}_h^{ts} - \frac{\mu}{\rho}\Delta \boldsymbol{u}_h^{ts} - \boldsymbol{f}_h^{n})], \delta \boldsymbol{u}_h^{ns} \cdot \nabla \boldsymbol{v}_h)\\
&+ \Delta t\,(\tau_C \nabla \cdot \delta \boldsymbol{u}_h^{ns}, \nabla \cdot \boldsymbol{v}_h)\;,
\end{aligned}
$$

$$
J_{F2}(\boldsymbol{u}_h^{ts}, p_h^{ts}, \boldsymbol{u}_h^{ns}, p_h^{ns}, \boldsymbol{v}_h, q_h)[\delta \boldsymbol{u}_h^{ns}, \delta p_h^{ns}]^T := \qquad (3.74b)
$$

$$\Delta t\,(\frac{1}{\rho}\nabla\cdot\delta\boldsymbol{u}_h^{ns},q_h)+(\boldsymbol{\tau}_M\delta\boldsymbol{u}_h^{ns},\nabla q_h)$$
$$+\,\Delta t\,\theta\,(\boldsymbol{\tau}_M\delta\boldsymbol{u}_h^{ns}\cdot\nabla\boldsymbol{u}_h^{ns},\nabla q_h)+\Delta t\,\theta\,(\boldsymbol{\tau}_M(\boldsymbol{u}_h^{ns}-\boldsymbol{u}_h^{r})\cdot\nabla\delta\boldsymbol{u}_h^{ns},\nabla q_h)$$
$$-\,\Delta t\,\theta\,(\boldsymbol{\tau}_M\frac{\mu}{\rho}\Delta\delta\boldsymbol{u}_h^{ns},\nabla q_h)+\Delta t\,(\boldsymbol{\tau}_M\frac{1}{\rho}\nabla\delta p^{ns},\nabla q_h)\;.$$

Here, $\hat{\boldsymbol{u}}_h$ is defined as $\hat{\boldsymbol{u}}_h := \boldsymbol{u}_h-\boldsymbol{\tau}_M\boldsymbol{r}_{M.h}$. The residual of the momentum equation is evaluated with the solutions from the previous Newton step $[\boldsymbol{u}_h^{ns},p_h^{ns}]$:

$$\hat{\boldsymbol{u}}_h^{n+1} := \boldsymbol{u}_h^{n+1}+\boldsymbol{\tau}_M(\frac{\boldsymbol{u}_h^{ns}-\boldsymbol{u}_h^{ts}}{\Delta t}+\boldsymbol{u}_h^{ns}\cdot\nabla\boldsymbol{u}_h^{ns}-\frac{\mu}{\rho}\Delta\boldsymbol{u}_h^{ns}+\frac{1}{\rho}\nabla p_h^{ns}-\boldsymbol{f}_h^{n+1})\;, \tag{3.75a}$$
$$\hat{\boldsymbol{u}}_h^{ns} := \boldsymbol{u}_h^{ns}+\boldsymbol{\tau}_M(\frac{\boldsymbol{u}_h^{ns}-\boldsymbol{u}_h^{ts}}{\Delta t}+\boldsymbol{u}_h^{ns}\cdot\nabla\boldsymbol{u}_h^{ns}-\frac{\mu}{\rho}\Delta\boldsymbol{u}_h^{ns}+\frac{1}{\rho}\nabla p_h^{ns}-\boldsymbol{f}_h^{n+1})\;. \tag{3.75b}$$

Additionally, the two stabilization parameters $\boldsymbol{\tau}_M$, τ_C are also evaluated with the solutions on the previous Newton step respectively. Note that, our formulation can also be considered as a modified Crank-Nicolson scheme where the residual part in $\hat{\boldsymbol{u}}_h$ is handled with the backward Euler method.

If we replace $[\boldsymbol{u}_h^{n+1},p_h^{n+1},\hat{\boldsymbol{u}}_h^{n+1}]$ by $[\boldsymbol{u}_h^{ns},p_h^{ns},\hat{\boldsymbol{u}}_h^{ns}]$ in Equation (3.71), the left-hand side is exactly F as in Equation (3.72). Hence, a Newton step is defined according to Equation (3.72) with the definitions above, it yields:

$$J_F(\boldsymbol{u}_h^{ns},p_h^{ns},\boldsymbol{u}_h^{ts},p_h^{ts})[\delta\boldsymbol{u}_h^{n+1}\;\delta p_h^{n+1}]^T=-F(\boldsymbol{u}_h^{ns},p_h^{ns},\boldsymbol{u}_h^{ts},p_h^{ts})\;. \tag{3.76}$$

In practice, we employ the Inexact Newton approach [50] in order to achieve a better control and the flexibility on the local convergence.

Space Discretization

Although we used already the notation of space discretization $(\boldsymbol{u}_h,p_n)$ in our time discretization formulation, we give more detailed information about the finite element discretization in this subsection. We redefine the functional spaces for the weak form of the incompressible of NSE by taking into account the Dirichlet boundary condition.

$$V(\Gamma_D;\mathcal{D}) := \Big\{\boldsymbol{v}\in H_0^1(\mathcal{D})^d \;\mid\; \boldsymbol{v}=0\,\text{on}\,\Gamma_D\Big\}\;, \tag{3.77a}$$
$$Q(\mathcal{D}) := \Big\{q\in L^2(\mathcal{D}) \;\mid\; \int_{\mathcal{D}} q\,dx=0\Big\}\;, \tag{3.77b}$$

where Γ_D is the Dirichlet boundary. Regarding the incompressible NSE, the pressure p plays essentially a role of the Lagrange multiplier for the incompressibility.

The finite element method is employed to solve the Newton step (Equation (3.76)), the computing domain $\mathcal{D}$ is triangulated to construct the finite-dimensional vector spaces for the trial function and the test function. As we utilize a conforming method, the finite-dimensional spaces are actually subspaces of the continuous spaces in Equation (3.77), i.e. $V^h\subset V$, $Q^h\subset Q$.

Definition 3.3.3 (Sobolev space $H^k(\Omega)$). *The Hilbert space of all elements $u\in L^2(\mathcal{D})$ with square-integrable weak derivatives $\partial^\alpha u\in L^2(\mathcal{D})$ for all α with $|\alpha|_1\leq k$, it yields:*

$$H^k(\mathcal{D}) := \{u\in L^2(\Omega):\partial^\alpha\in L^2(\Omega)\;,\quad\forall 0\leq|\alpha|_1\leq k\}\;. \tag{3.78}$$

The Sobolev space $H^k(\Omega)$ is equipped with the inner product:

$$(u, v)_{k,\mathcal{D}} := \sum_{0 \leq |\alpha|_1 \leq k} \int_{\Omega} (\partial^{\alpha} u)(\partial^{\alpha} v) \, d\boldsymbol{x} \, . \tag{3.79}$$

Theorem 3.3.2. *Let $\mathcal{D}$ be a bounded domain, which is partitioned into a finite number of subdomains $\{z_1, ..., z_N\}$ and X^h is a space of functions, where $x|_{z_i} \in \mathcal{C}^{\infty}$, $\forall x \in X^h$. Then $X^h \in H^k(\mathcal{D})$, $k \leq 1$, if and only if $X^h \subset \mathcal{C}^{k-1}(\overline{\mathcal{D}})$ [19].*

The theorem above provides a process to construct a set of finite-dimensional subspaces of Hilbert spaces, and any finite-dimensional vector space is spanned by a set of piecewise polynomial basis functions φ_i, it implies:

$$X^h := \text{span}\{\varphi_1^h, ..., \varphi_{N_h}^h\} \, . \tag{3.80}$$

$$x_h := \sum_{i=0}^{N_h} c_i^h \varphi_i^h \, , \tag{3.81}$$

where $c_i^h \in \mathbb{R}$. The result of the partitioning process in Theorem 3.3.2 is the triangulation $\mathcal{T}_h$, which consists of a finite number of regular elements:

$$\mathcal{T}_h := \{z_1, ..., z_n\} \, , \tag{3.82}$$

and,

$$\bigcup_{i=1}^{N} \overline{z}_i = \overline{\mathcal{D}} \, , \quad \overline{z}_i \cap \overline{z}_j = \emptyset \quad \text{if} \, i \neq j \, . \tag{3.83}$$

The essential argument for the velocity/pressure approximations in $V^h \times Q^h$ to be stable is the Ladyshenskaja-Babuška-Brezzi (LBB) condition [32]:

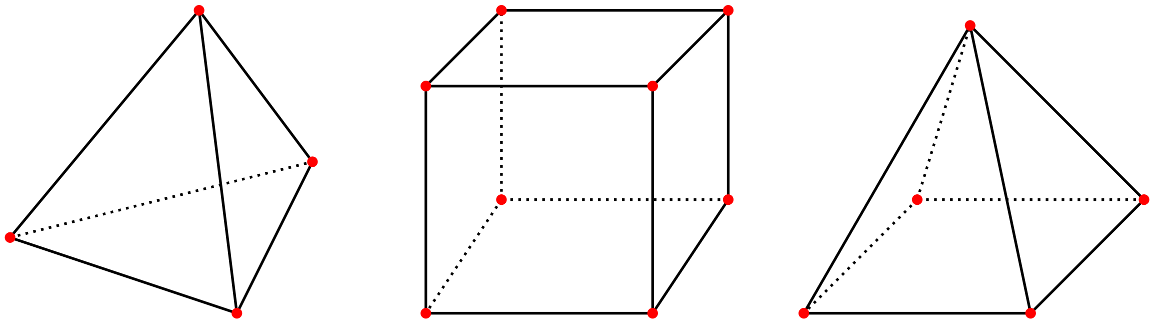
$$\inf_{q_h \in Q^h} \sup_{\boldsymbol{v}_h \in V^h} \frac{(q_h, \nabla \cdot \boldsymbol{v}_h)}{\|\nabla \boldsymbol{v}_h\| \|q_h\|} \geq \beta_0 > 0 \, . \tag{3.84}$$

Therefore, the mixed elements (e.g. the Taylor-Hood elements [65]) need to be considered for the incompressible NSE simulation in order to obtain a well-posed discrete saddle point problem. However, our residual-based VMS formulation employs the stabilization techniques, which overcome the numerical instabilities even on non-LBB-stable finite element spaces, e.g. the equal-order finite elements. Hence, we focus only on the equal-order linear finite elements in this work, since they are easier to be handled. Figure 3.6 shows three different cells which are involved in the three-dimensional case under the requirement of the shear layer update approach (Chapter 2).

After being discretized in time, Equation (3.67) can be solved with the Newton-Raphson method [72], the structure of a single Newton step is stated as:

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} u_h \\ p_h \end{bmatrix} = \begin{bmatrix} -F_1 \\ -F_2 \end{bmatrix} \, , \tag{3.85}$$

where, $[u_h, p_h]$ are the unknowns of the linear system, and $u_h \in \mathbb{R}^{dN}$, $p_h \in \mathbb{R}^N$. d is the dimension of the computing domain $\mathcal{D}$. Moreover, $A \in \mathbb{R}^{dN,dN}$, $B \in \mathbb{R}^{dN,N}$, $C^{N,dN}$, $D^{N,N}$, $F_1 \in \mathbb{R}^{dN}$, $F_2 \in \mathbb{R}^N$. Besides, the linearized system of the incompressible NSE has a saddle point structure, it infers $D = 0$, $C = B^T$. Whereas, for Equation (3.67), the PSPG stabilization term contributes to the sub-block D, the linearized system is no longer saddle point system because of the stabilization formulations.



(a) Tetrahedron (b) Hexahedron (c) Pyramid

Figure 3.6: Three basic elements are involved in the three-dimensional situation: tetrahedron, hexahedron and pyramid. The red dots indicate the location of DOFs for the linear elements.

3.3.4 Stochastic VMS for Blood Pump Modeling

The previous section provides the insight of the discretization of the VMS formulation and establishes a framework together with Chapter 2 for solving a high speed rotating machinery. Now we extend our deterministic setting in the direction of the stochastic system. Henceforth, we restrict within this work that there are only three uncertain inputs which are deliberated: the inflow boundary condition $\boldsymbol{u}_I$, the dynamic viscosity μ and the rotation speed $\boldsymbol{u}^r$. Those uncertain sources cover mainly the different types of the modification on the global structure once we employ the generalized Polynomial Chaos expansion method to the nonlinear equations. Yet this restriction is carried out only owing to the perspective of the application. More random parameters can be drawn into the system in a similar manner, whereas, the essential patterns of the system do not alter. Thereafter, the corresponding Polynomial Chaos system is first developed for the incompressible NSE, afterward, it is extended to the VMS formulation as desired.

gPCE Galerkin Method for the Incompressible NSE

Based on Section 3.2, our random vector is precisely defined as $\boldsymbol{\xi} := [\xi_1, \xi_2, \xi_3] \in \mathbb{R}^3$. We denote that ξ_1, ξ_2 and ξ_3 correspond to the boundary condition, the dynamic viscosity and the rotating speed, respectively. We restrict $\boldsymbol{\xi}$ to be real-valued random variables and mutually independent. Moreover, each random variable is defined on a continuous probability space. Therefore, the two unknown variables of the incompressible NSE, namely the velocity $\boldsymbol{u}$ and the pressure p, are also described with respect to the random vector $\boldsymbol{\xi}$.

$$\frac{\partial \boldsymbol{u}(\boldsymbol{\xi})}{\partial t} + (\boldsymbol{u}(\boldsymbol{\xi}) - \boldsymbol{u}^r(\boldsymbol{\xi})) \cdot \nabla \boldsymbol{u}(\boldsymbol{\xi}) \tag{3.86a}$$

$$-\frac{\mu(\boldsymbol{\xi})}{\rho}\Delta \boldsymbol{u}(\boldsymbol{\xi}) + \frac{1}{\rho}\nabla p(\boldsymbol{\xi}) = \boldsymbol{f}\ , \qquad \text{in } \mathcal{D}^t \times [0,T]\ ,$$

$$\nabla \cdot \boldsymbol{u}(\boldsymbol{\xi}) = 0 \qquad \text{in } \mathcal{D}^t \times [0,T]\ , \tag{3.86b}$$

$$\boldsymbol{u}^r(\boldsymbol{\xi}) = \boldsymbol{d} \times \boldsymbol{\omega}(\boldsymbol{\xi})\ , \qquad \text{in } \mathcal{D}^t_{rot} \times [0,T]\ , \tag{3.86c}$$

$$\boldsymbol{u}^r(\boldsymbol{\xi}) = \boldsymbol{0}\ , \qquad \text{in } \mathcal{D}^t_{stat} \times [0,T]\ , \tag{3.86d}$$

$$\boldsymbol{u}(\boldsymbol{\xi}) = \boldsymbol{u}_I(\boldsymbol{\xi})\ , \qquad \text{on } \Gamma^t_{in} \times [0,T]\ , \tag{3.86e}$$

$$(-\mathbb{1}p(\boldsymbol{\xi}) + \frac{\mu}{\rho}\nabla \boldsymbol{u}(\boldsymbol{\xi})) \cdot \boldsymbol{n} = \boldsymbol{0}\ , \qquad \text{on } \Gamma^t_{out} \times [0,T]\ , \tag{3.86f}$$

$$\boldsymbol{u}(\boldsymbol{\xi}) = \boldsymbol{d} \times \boldsymbol{\omega}(\boldsymbol{\xi})\ , \qquad \text{on } \Gamma^t_{rotor} \times [0,T]\ , \tag{3.86g}$$

$$\boldsymbol{u}(\boldsymbol{\xi}) = \boldsymbol{0} \;, \qquad\qquad \text{on } \Gamma^t_{wall} \times [0,T] \;. \tag{3.86h}$$

This set of equations denotes the incompressible NSE for a rotating machinery in strong formulation under a random $\boldsymbol{\xi}$ of input parameters. We recall that the main theorem about the incompressible NSE in Section 3.3.1 holds for Equation (3.86) for a fixed $\boldsymbol{\xi}$, only if $\mu(\boldsymbol{\xi}) > 0$. Analogously to Equation (3.38), the Polynomial Chaos expansions of the velocity and the pressure are approximated as a linear combination of Chaos Polynomials:

$$\boldsymbol{u}(\boldsymbol{x},t,\boldsymbol{\xi}) \simeq \sum_{i=0}^{P} \boldsymbol{u}_i(\boldsymbol{x},t)\psi_i(\boldsymbol{\xi}) \;, \tag{3.87a}$$

$$p(\boldsymbol{x},t,\boldsymbol{\xi}) \simeq \sum_{i=0}^{P} p_i(\boldsymbol{x},t)\psi_i(\boldsymbol{\xi}) \;. \tag{3.87b}$$

$(P+1)$ is the dimension of the PC space, which is defined by the number of random variables and the truncated order of the polynomials (Equation (3.30)). The discretized stochastic space can be written as:

$$\mathcal{S}^P = \operatorname{span}\{\psi_0(\boldsymbol{\xi}), ..., \psi_P(\boldsymbol{\xi})\} \subset \mathcal{S} := \bigcup_{i=0}^{\infty} \operatorname{span}\{\psi_0(\boldsymbol{\xi}), ..., \psi_i(\boldsymbol{\xi})\} \;. \tag{3.88}$$

The three uncertain parameters can also be expressed analogously to Equation (3.87):

$$\mu(\boldsymbol{\xi}) = \sum_{i=0}^{P} \mu_i \psi_i(\boldsymbol{\xi}) \;, \tag{3.89a}$$

$$u_I(\boldsymbol{x},\boldsymbol{\xi}) = \sum_{i=0}^{P} u_{I,i}(\boldsymbol{x})\psi_i(\boldsymbol{\xi}) \;, \tag{3.89b}$$

$$\boldsymbol{\omega}(\boldsymbol{\xi}) = \sum_{i=0}^{P} \boldsymbol{\omega}_i \psi_i(\boldsymbol{\xi}) \;. \tag{3.89c}$$

Therefore, we write Equation (3.86) by inserting $\boldsymbol{u}, p, \mu, u_I$ and $\boldsymbol{\omega}$ from Equation (3.87) and Equation (3.89):

$$\sum_{i=0}^{P} \frac{\partial \boldsymbol{u}_i}{\partial t}\psi_i + \sum_{i=0}^{P}\sum_{j=0}^{P} (\boldsymbol{u}_i - \boldsymbol{u}_i^r) \cdot \nabla \boldsymbol{u}_j \psi_i \psi_j \tag{3.90a}$$

$$- \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \Delta \boldsymbol{u}_j \psi_i \psi_j + \sum_{i=0}^{P} \nabla p_i \psi_i = \sum_{i=0}^{P} \boldsymbol{f}_i \psi_i \;, \qquad \text{in } \mathcal{D}^t \times [0,T] \;,$$

$$\sum_{i=0}^{P} \nabla \cdot \boldsymbol{u}_i \psi_i = 0 \;, \qquad \text{in } \mathcal{D}^t \times [0,T] \;, \tag{3.90b}$$

$$\sum_{i=0}^{P} \boldsymbol{u}_i^r \psi_i = \sum_{i=0}^{P} \boldsymbol{d} \times \boldsymbol{\omega}_i \psi_i \;, \qquad \text{in } \mathcal{D}^t_{rot} \times [0,T] \;, \tag{3.90c}$$

$$\sum_{i=0}^{P} \boldsymbol{u}_i^r \psi_i = \boldsymbol{0} \;, \qquad \text{in } \mathcal{D}^t_{stat} \times [0,T] \;, \tag{3.90d}$$

$$\sum_{i=0}^{P} \boldsymbol{u}_i \psi_i = \sum_{i=0}^{P} \boldsymbol{u}_{I,i} \psi_i \;, \qquad \text{on } \Gamma_{in} \times [0,T] \;, \tag{3.90e}$$

$$(-\sum_{i=0}^{P} \mathbb{1} p_i \psi_i + \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \nabla \boldsymbol{u}_j \psi_i \psi_j) \cdot \boldsymbol{n} = \boldsymbol{0} \;, \qquad \text{on } \Gamma_{out} \times [0,T] \;, \tag{3.90f}$$

$$\sum_{i=0}^{P} \boldsymbol{u}_i \psi_i = \sum_{i=0}^{P} \boldsymbol{d} \times \boldsymbol{\omega}_i \psi_i \;, \qquad \text{on } \Gamma_{rotor} \times [0,T] \;, \tag{3.90g}$$

$$\sum_{i=0}^{P} \boldsymbol{u}_i \psi_i = \boldsymbol{0} \;, \qquad \text{on } \Gamma_{wall} \times [0,T] \;. \tag{3.90h}$$

Multiplying Equation (3.90) by ψ_k and computing the expectation, the Galerkin projection procedure gives us:

$$\frac{\partial \boldsymbol{u}_k}{\partial t} + \sum_{i=0}^{P}\sum_{j=0}^{P} (\boldsymbol{u}_i - \boldsymbol{u}_i^r) \cdot \nabla \boldsymbol{u}_j \, C_{ijk} \tag{3.91a}$$

$$- \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \Delta \boldsymbol{u}_j \, C_{ijk} + \frac{1}{\rho} \nabla p_k = \boldsymbol{f}_k \;, \qquad \text{in } \mathcal{D}^t \times [0,T] \;,$$

$$\nabla \cdot \boldsymbol{u}_k = 0 \;, \qquad \text{in } \mathcal{D}^t \times [0,T] \;, \tag{3.91b}$$

$$\boldsymbol{u}_k^r = \boldsymbol{d} \times \boldsymbol{\omega}_k \;, \qquad \text{in } \mathcal{D}^t_{rot} \times [0,T] \;, \tag{3.91c}$$

$$\boldsymbol{u}_k^r = \boldsymbol{0} \;, \qquad \text{in } \mathcal{D}^t_{stat} \times [0,T] \;, \tag{3.91d}$$

$$\boldsymbol{u}_k = \boldsymbol{u}_{I,k} \;, \qquad \text{on } \Gamma_{in} \times [0,T] \;, \tag{3.91e}$$

$$(-\mathbb{1} p_k + \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \nabla \boldsymbol{u}_j \, C_{ijk}) \cdot \boldsymbol{n} = \boldsymbol{0} \;, \qquad \text{on } \Gamma_{out} \times [0,T] \;, \tag{3.91f}$$

$$\boldsymbol{u}_k = \boldsymbol{d} \times \boldsymbol{\omega}_k \;, \qquad \text{on } \Gamma_{rotor} \times [0,T] \;, \tag{3.91g}$$

$$\boldsymbol{u}_k = \boldsymbol{0} \;, \qquad \text{on } \Gamma_{wall} \times [0,T] \;, \tag{3.91h}$$

for $k = 0, ..., P$. C_{ijk} is defined by the third-order tensor product (Equation (3.51)), and it substantially establishes the couplings between the stochastic modes in the stochastic Galerkin formulation (Equation (3.91)). We can state the stochastic variational formulation by applying the functions $\boldsymbol{v} \in V$, $q \in Q$:

$$(\frac{\partial \boldsymbol{u}_k}{\partial t}, \boldsymbol{v}) + \sum_{i=0}^{P}\sum_{j=0}^{P} ((\boldsymbol{u}_i - \boldsymbol{u}_i^r) \cdot \nabla \boldsymbol{u}_j, \boldsymbol{v}) \, C_{ijk} \tag{3.92a}$$

$$+ \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} (\nabla \boldsymbol{u}_j, \nabla \boldsymbol{v}) \, C_{ijk} - \frac{1}{\rho}(p_k, \nabla \cdot \boldsymbol{v}) = (\boldsymbol{f}_k, \boldsymbol{v}) \;, \qquad \text{in } \mathcal{D}^t \times [0,T] \;,$$

$$(\nabla \cdot \boldsymbol{u}_k, q) = 0 \;, \qquad \text{in } \mathcal{D}^t \times [0,T] \;. \tag{3.92b}$$

for $k = 0, ..., P$, $\{\boldsymbol{u}_k\} \subset V$ and $\{p_k\} \subset Q$, i.e., $\boldsymbol{u} \in V \otimes \mathcal{S}^P$, $p \in Q \otimes \mathcal{S}^P$. We neglect the boundary conditions from Equation (3.91) for the sake of simplicity. In addition, we leave the modeling part of the random inputs $\boldsymbol{u}_I(\boldsymbol{x}, \boldsymbol{\xi})$, $\mu(\boldsymbol{\xi})$ and $\boldsymbol{u}^r(\boldsymbol{x}, \boldsymbol{\xi})$ open for the moment, even though, their modeling can affect the sparsity of the global system matrix, the precise formulations will be given in Chapter 6. Equation (3.92) is a semi-discrete function, knowing that only the stochastic space $\mathcal{S}$ is discretized. Furthermore, Equation (3.92) involves a set of deterministic problems, which have a similar structure as the incompressible NSE. Therefore, the stochastic spectral problem develops into finding a set of solutions for the velocity and the pressure ($[\boldsymbol{u}_0, ..., \boldsymbol{u}_N]$ and $[p_0, ..., p_N]$). We can pursue the spatial discretization on V and Q in order to obtain $\boldsymbol{u}_{h,k} \in V^h \subset V$, $p_{h,k} \in Q^h \subset Q$. Equation (3.92) becomes, therefore, a stochastic Galerkin finite element variational formulation. It is important to note that the stochastic weak form under the stochastic-spectral finite element method framework is derived at first by performing the Galerkin projection on the discretized stochastic space, then determining the corresponding variational formulation. The alternative derivation of the spectral problem is to start with the weak form of the governing equations, then performing the Galerkin projection [119]. The stochastic spectral problem for the VMS is obtained with the latter suggestion by inserting Equation (3.87) into Equation (3.67).

gPCE Galerkin Method for VMS

We rewrite Equation (3.87) in consideration of a spatial discretization:

$$\boldsymbol{u}_h(\boldsymbol{x}, t, \boldsymbol{\xi}) \simeq \sum_{i=0}^{P} \boldsymbol{u}_{h,i}(\boldsymbol{x}, t)\psi_i(\boldsymbol{\xi}) \ , \qquad \in V^h \otimes \mathcal{S}^P \ , \tag{3.93a}$$

$$p_h(\boldsymbol{\xi}, t, \boldsymbol{\xi}) \simeq \sum_{i=0}^{P} p_{h,i}(\boldsymbol{x}, t)\psi_i(\boldsymbol{\xi}) \ , \qquad \in Q^h \otimes \mathcal{S}^P \ , \tag{3.93b}$$

where $\boldsymbol{u}_{h,i} \in V^h$, $p_{h,i} \in Q^h$. The superscript h stands for the spatial discretization, and i stands for the stochastic mode. Therefore, we provide the stochastic variational formulation of Equation (3.67):

Find $\boldsymbol{u}_h \in V^h \otimes \mathcal{S}^P$, $p_h \in Q^h \otimes \mathcal{S}^P$, such that,

$$\begin{aligned}
&(\frac{\boldsymbol{u}_{h,k}}{\partial t}, \boldsymbol{v}_h) + \sum_{i=0}^{P}\sum_{j=0}^{P}((\hat{\boldsymbol{u}}_{h,i} - \boldsymbol{u}^r_{h,i}) \cdot \nabla \boldsymbol{u}_{h,j}, \boldsymbol{v}_h)\, C_{ijk} \\
&+ \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho}(\nabla \boldsymbol{u}_{h,j}, \nabla \boldsymbol{v}_h)\, C_{ijk} - \frac{1}{\rho}(p_{h,k}, \nabla \cdot \boldsymbol{v}_h) - (\boldsymbol{f}_{h,k}, \boldsymbol{v}_h) \\
&+ (\boldsymbol{\tau}_M[\frac{\partial \boldsymbol{u}_{h,k}}{\partial t} + \sum_{i=0}^{P}\sum_{j=0}^{P}(\boldsymbol{u}_{h,i} - \boldsymbol{u}^r_{h,i}) \cdot \nabla \boldsymbol{u}_{h,j}\, C_{ijk} \\
&- \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \Delta \boldsymbol{u}_{h,j}\, C_{ijk} + \frac{1}{\rho} p_{h,k} - \boldsymbol{f}_{h,k}], (\hat{\boldsymbol{u}}_{h,0} - \boldsymbol{u}^r_{h,0}) \cdot \nabla \boldsymbol{v}_h) \\
&+ (\tau_C \nabla \cdot \boldsymbol{u}_{h,k}, \nabla \cdot \boldsymbol{v}_h) = \boldsymbol{0} \ ,
\end{aligned} \tag{3.94a}$$

$$\begin{aligned}
&(\frac{1}{\rho} \nabla \boldsymbol{u}_{h,k}, q_h) \\
&+ (\boldsymbol{\tau}_M[\frac{\partial \boldsymbol{u}_{h,k}}{\partial t} + \sum_{i=0}^{P}\sum_{j=0}^{P}(\boldsymbol{u}_{h,i} - \boldsymbol{u}^r_{h,i}) \cdot \nabla \boldsymbol{u}_{h,j}\, C_{ijk} \\
&- \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \Delta \boldsymbol{u}_{h,j}\, C_{ijk} + \frac{1}{\rho} \nabla p_{h,k} - \boldsymbol{f}_{h,k}], \nabla q_h) = 0 \ ,
\end{aligned} \tag{3.94b}$$

$\forall k = 0, ..., P$, $\forall \boldsymbol{v}_h \in V^h$ and $\forall q_h \in Q^h$. The explicit form of $\hat{\boldsymbol{u}}_{h,i}$ is given as:

$$\hat{\boldsymbol{u}}_{h,i} := \boldsymbol{u}_{h,i} + \boldsymbol{\tau}_M[\frac{\boldsymbol{u}_{h,i}}{\partial t} + (\boldsymbol{u}_{h,i} - \boldsymbol{u}^r_{h,i}) \cdot \nabla \boldsymbol{u}_{h,i} - \frac{\mu_i}{\rho}\Delta \boldsymbol{u}_{h,i} + \frac{1}{\rho}\nabla p_{h,i} - \boldsymbol{f}_{h,i}]\,. \tag{3.95}$$

Remark 3.3.4 (Separable expansion)**.** *The stochastic finite element solution $\boldsymbol{u}_h$ and p_h can be written in the following fashion:*

$$\boldsymbol{u}_h(\boldsymbol{x}, \boldsymbol{\xi}) = \sum_{i=0}^{N}\sum_{j=0}^{P} \boldsymbol{u}_{i,j}\varphi_i^u(\boldsymbol{x})\psi_j(\boldsymbol{\xi})\ , \quad p_h(\boldsymbol{x}, \boldsymbol{\xi}) = \sum_{i=0}^{N}\sum_{j=0}^{P} p_k\varphi_{i,j}^p(\boldsymbol{x})\psi_j(\boldsymbol{\xi})\ , \tag{3.96}$$

$\psi_i(\boldsymbol{\xi})$ represents the i-th generalized Polynomial Chaos basis function, $\varphi^{\boldsymbol{u}}(\boldsymbol{x})$ and $\varphi^p(\boldsymbol{x})$ are the finite element basis function for the velocity and the pressure, respectively. They are separable in terms of the spatial part and the stochastic part, i.e. $\boldsymbol{u}_u(\cdot, \boldsymbol{\xi}) \in \mathcal{S}^p$, $\boldsymbol{u}_h(\boldsymbol{x}, \cdot) \in V^h$.

Structure of Stochastic Discretized VMS

Equation (3.85) gives a matrix structure of the linearized Equation (3.67) on a single Newton step, the linearization of the stochastic weak formulation (Equation (3.94)) is carried out in a systematic manner as in Equation (3.71) and Equation (3.74). As stated above, the discretization in space V^h and Q^h remains the same for each mode, undoubtedly, each individual sub-block in the global matrix has a similar structure as in the deterministic case. For a stochastic mode k, the matrix-vector structure is given by:

$$\sum_{i=0}^{P}\sum_{j=0}^{P}\left(C_{ijk}\begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix}\begin{bmatrix} \boldsymbol{u}_{h,j} \\ p_{h,j} \end{bmatrix}\right) = \begin{bmatrix} -F_{1,k} \\ -F_{2,k} \end{bmatrix}\ , \quad k = 0, ..., P\ . \tag{3.97}$$

Here, $A_i \in \mathbb{R}^{dN,dN}$, $B_i \in \mathbb{R}^{dN,N}$, $C_i \in \mathbb{R}^{N,dN}$, $D_i \in \mathbb{R}^{N,N}$, $\boldsymbol{u}_{h,j} \in \mathbb{R}^{dN}$, $p_{h,j} \in \mathbb{R}^{N}$, $F_{1,k} \in \mathbb{R}^{dN}$ and $F_{2,k} \in \mathbb{R}^{N}$. It is equivalent to:

$$\sum_{i=0}^{P}\begin{bmatrix} C_{i00}\begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix} & \cdots & C_{iP0}\begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix} \\ \vdots & \cdots & \vdots \\ \vdots & \cdots & \vdots \\ \vdots & \cdots & \vdots \\ C_{i0P}\begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix} & \cdots & C_{iPP}\begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix} \end{bmatrix}\begin{bmatrix} \boldsymbol{u}_{h,0} \\ p_{h,0} \\ \vdots \\ \vdots \\ \vdots \\ \boldsymbol{u}_{h,P} \\ p_{h,P} \end{bmatrix} = \begin{bmatrix} -F_{1,0} \\ -F_{2,0} \\ \vdots \\ \vdots \\ \vdots \\ -F_{1,P} \\ -F_{2,P} \end{bmatrix}\ . \tag{3.98}$$

Alternatively, Equation (3.98) can be rewritten as:

$$\sum_{i=0}^{P}\begin{bmatrix} \begin{bmatrix} A_i\,C_{i00} & \cdots & A_i\,C_{iP0} \\ \vdots & \cdots & \vdots \\ \vdots & \cdots & \vdots \\ A_i\,C_{i0P} & \cdots & A_i\,C_{iPP} \end{bmatrix} & \begin{bmatrix} B_i\,C_{i00} & \cdots & B_i\,C_{iP0} \\ \vdots & \cdots & \vdots \\ \vdots & \cdots & \vdots \\ B_i\,C_{i0P} & \cdots & B_i\,C_{iPP} \end{bmatrix} \\ \begin{bmatrix} C_i\,C_{i00} & \cdots & C_i\,C_{iP0} \\ \vdots & \cdots & \vdots \\ \vdots & \cdots & \vdots \\ C_i\,C_{i0P} & \cdots & C_i\,C_{iPP} \end{bmatrix} & \begin{bmatrix} D_i\,C_{i00} & \cdots & D_i\,C_{iP0} \\ \vdots & \cdots & \vdots \\ \vdots & \cdots & \vdots \\ D_i\,C_{i0P} & \cdots & D_i\,C_{iPP} \end{bmatrix} \end{bmatrix}\begin{bmatrix} \begin{bmatrix} \boldsymbol{u}_{h,0} \\ \vdots \\ \vdots \\ \boldsymbol{u}_{h,P} \end{bmatrix} \\ \begin{bmatrix} p_{h,0} \\ \vdots \\ \vdots \\ p_{h,P} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} -F_{1,0} \\ \vdots \\ \vdots \\ -F_{1,P} \end{bmatrix} \\ \begin{bmatrix} -F_{2,0} \\ \vdots \\ \vdots \\ -F_{2,P} \end{bmatrix} \end{bmatrix}\ . \tag{3.99}$$

The latter version collects all velocities and pressures from the stochastic modes together, which can be beneficial for certain preconditioners when we apply them directly on the global matrix. However, the solving technique suggested in this work prefers to operate with the former matrix-vector representation (Equation (3.98)) by virtue of the hierarchical structure, especially the sub-block is sparse according to the finite element discretization.

Moreover, the global matrix $A \in \mathbb{R}^{(P+1)(d+1)N,(P+1)(d+1)N}$ in Equation (3.98) can even be exhibited by using the Kronecker product:

$$A := \sum_{i=0}^{P} K_i \otimes \bar{A}_i \ , \tag{3.100}$$

$\bar{A}_i := \begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix} \in \mathbb{R}^{(d+1)N,(d+1)N}$, $(K_i)_{jk} := C_{ijk}$. The tensor product expression shows that the global matrix A depends on the total number of the stochastic mode $(P+1)$ and the sub-blocks $\bar{A}_i$. One can notice $\bar{A}_i$ are repeatedly employed for the construction of A, therefore, it is not needed to keep the large matrix A into the computing memory for the numerical implementation, the sub-matrices $\bar{A}_i$ are able to reconstruct the global matrix A dynamically during the computation.

Chapter 4

Solvers and Preconditioners

As the stochastic spectral problem increases the size of the governing system exponentially, solving such a set of coupled equations, notably regarding the spectral-stochastic finite element method, is very arduous. Besides, dealing with an unsteady high Reynolds number flow has additionally greater demands on the spatial and temporal discretization level, thus our problem becomes more computationally challenging. Within this chapter we concentrate particularly on the numerical algorithms and the solving strategies related to the stochastic Galerkin system as well as the deterministic problem, since the resolution of a deterministic problem can play a crucial role in constructing preconditioners for the coupled system arising from the stochastic Galerkin projection. A special concern of solving the linear system associated to the variational multiscale method formulation will also be presented by using a Schur Complement preconditioner as the accelerator. We avoid utilizing direct solvers by reasons of efficiency and feasibility. We restrict our focus to the stochastic linearized equations arising from the spectral-stochastic finite element method. In this chapter, we begin with the solving techniques for the global system, after that we draw our attention to the solvers for the VMS formulation.

4.1 Stochastic Galerkin Matrix-Vector Multiplication

Although the numerical implementation does not have an impact on the solver's efficiency on the theoretical level, it is worthwhile to provide a short remark about the stochastic Galerkin matrix-vector multiplication, before discussing the solving techniques for Equation (3.98). However, the solvers can profit from the pattern of the stochastic Galerkin matrix, once we are solving high dimensional algebraic systems.

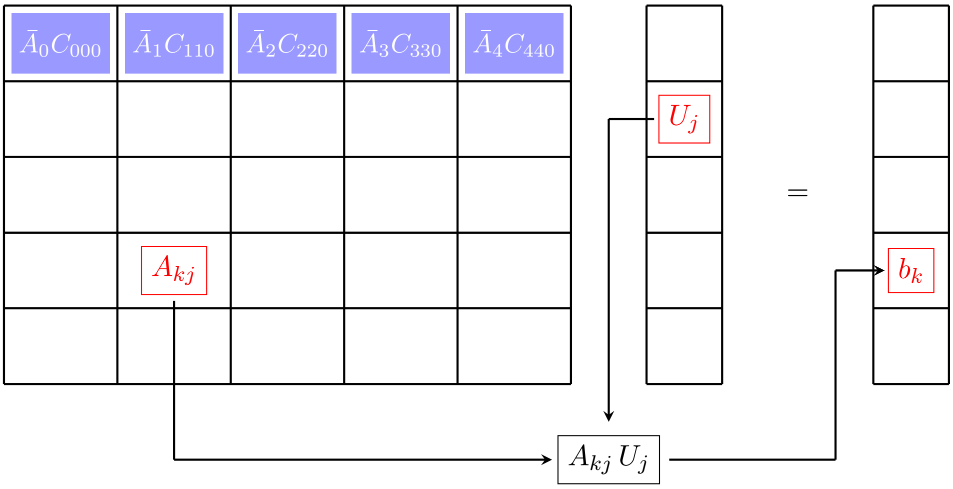


Figure 4.1: Illustration of the Stochastic Galerkin matrix-vector multiplication.

Figure 4.1 demonstrates the algorithm of the matrix-vector multiplication for the stochastic Galerkin structure. In our case, the linear equation is obtained by combining Equation (3.74) and Equation (3.94), A_{kj} is a sub-block of the global Galerkin matrix A, U_j is the solution of a stochastic mode j, b_k is the right-hand side for the stochastic mode k. The matrix-vector multiplication is given by:

$$A_{kj}U_j := \sum_{i=0}^{P} C_{ijk}\bar{A}_i U_j = b_k \ , \quad j,k = 0,...,P \ . \tag{4.1}$$

Equation (3.100) intends to build the global matrix A by collecting all matrices $\bar{A}_i, i = 0,...,P$. The matrices $\bar{A}_i$ can be easily constructed from the sub-blocks on the first row of A, i.e. A_{0j} (blue blocks in Figure 4.1), by exploiting the zero patter of C_{ijk}. It implies:

$$A_{0i} := C_{ii0}\bar{A}_i \ , \tag{4.2}$$

or,

$$\sum_{i=0}^{P} C_{ij0} = C_{ii0} \ . \tag{4.3}$$

Furthermore, if the random variables have different expansion orders than L, denoted by L_t. Equation (4.1) can be reduced to:

$$A_{kj}U_j := \sum_{i=0}^{P_t} C_{ijk}\bar{A}_i U_j = b_k \ , \quad j,k = 0,...,P \ , \tag{4.4}$$

where, $P_t + 1 := \frac{(M+L_t)!}{M!L_t!}$. More specifically, if $L_t = 1$, then $P_t = M$, which is obviously smaller than the total number of stochastic modes P (e.g. Figure 3.5). Therefore, the storage of the sub-matrices $\bar{A}_i$ has only the linear complexity with respect to M.

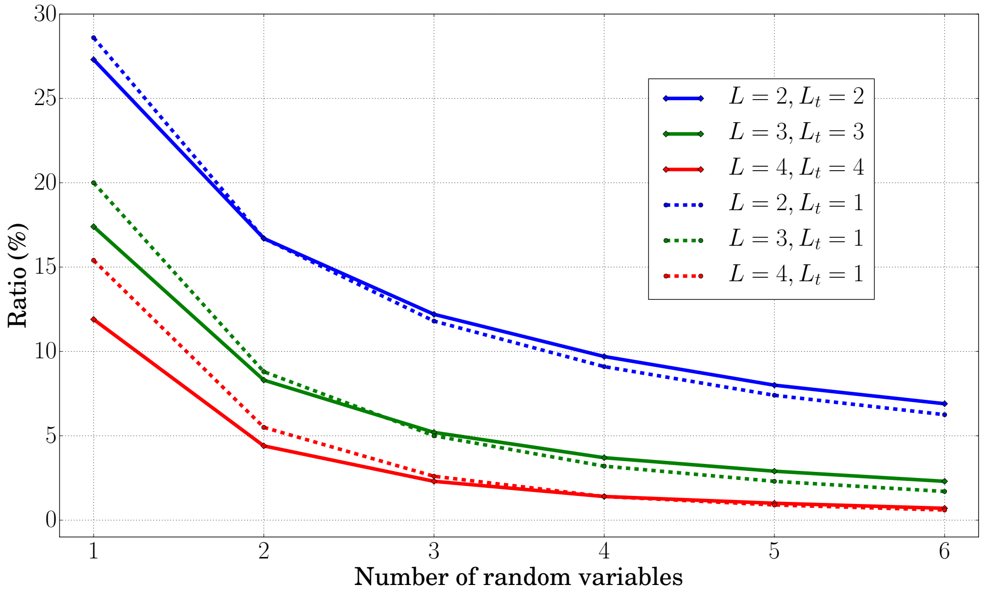


Figure 4.2: The ratio (in percentage) of the stored sub-matrices to the total number non-zero sub-matrices. The solid lines represent that the random inputs have a same order of polynomial expansion as L, i.e. $L_t = L$, and the dotted lines consider only the linear random inputs cases, i.e. $L_t = 1$. All random variables are modeled by the normalized Uniform distribution.

Figure 4.2 exhibits the ratio of the stored sub-matrices against the total non-zero blocks in the global matrix A accompanied by two different expansion orders of the random inputs, namely $L_t = L$ and $L_t = 1$. The ratio drops rapidly when the number of the random variables increases, as well as for the polynomial order L. This consideration is greatly helpful once A becomes very large, because the memory consumption is reduced by a significant factor.

In the following sections, the linearized equation (Equation (3.98)) in the Newton step has a form as:

$$AU = b \; , \tag{4.5}$$

where, $A \in \mathbb{R}^{(P+1)(d+1)N,(P+1)(d+1)N}$, $U \in \mathbb{R}^{(P+1)(d+1)N}$, $b \in \mathbb{R}^{(P+1)(d+1)N}$. In addition,

$$A_{kj} := \sum_{i=0}^{P} C_{ijk}\bar{A}_i \; , \quad \bar{A}_i \in \mathbb{R}^{(d+1)N,(d+1)N} \; . \tag{4.6}$$

$$U := [U_0, ..., U_P] \; , \quad U_i := [\boldsymbol{u}_{h,i}, p_{h,i}] \; , \quad \boldsymbol{u}_{h,i} \in \mathbb{R}^{dN}, p_{h,i} \in \mathbb{R}^N \; . \tag{4.7}$$

$$b := [b_0, ..., b_P] \; , \quad b_i = [-F_{1,i}, -F_{2,i}] \; , \quad F_{1,i} \in \mathbb{R}^{dN}, F_{2,i} \in \mathbb{R}^N \; . \tag{4.8}$$

The subscript i indicates the stochastic mode.

4.2 FGMRES for the Global Linear System

The ultimate intention for solving Equation (4.5) is to obtain the solution vector U, which has an analytical algebraic expression. It is given by:

$$U = A^{-1}b \; . \tag{4.9}$$

Notwithstanding, a direct inversion of a large dimensional matrix A can be impracticable. Instead, solving the large linear system is carried out with the iterative methods in our simulation. More specifically in this work, we employ the Krylov subspace methods, which are primarily based on a projection technique. These methods approximate $A^{-1}b$ by $q_m(A)b$, which yields:

$$U := A^{-1}b \approx U^{(0)} + q_{m-1}(A)r^{(0)} = U^{(m)} \; . \tag{4.10}$$

Here, $U^{(0)}$ is any initial guess of U, and $r^{(0)} = b - AU^{(0)}$ is the residual with respect to $U^{(0)}$. q_{m-1} is a polynomial of degree $(m-1)$, $U^{(m)}$ is the solution after the m-th iteration. The Krylov subspace is denoted as:

$$\mathcal{K}_m(A, v) := \text{span}\{v, Av, A^2v, ..., A^{m-1}v\} \; , \tag{4.11}$$

which is also referred to as the Krylov sequence. Besides, $U^{(m)} - U^{(0)} := q_{m-1}(A)r^{(0)}$ is in a Krylov subspace $\mathcal{K}_m(A, r^{(0)})$. Hence, the approximate solution $U^{(m)}$ can be defined by imposing the Petrov-Galerkin condition:

$$b - AU^{(m)} \perp \mathcal{L}_m \; , \tag{4.12}$$

where, $\mathcal{L}_m$ is another subspace with the dimension m. Consequently, $U^{(m)}$ can be expressed:

$$U^{(m)} = U^{(0)} + V(W^TAV)^{-1}W^Tr^{(0)} \; . \tag{4.13}$$

where V and W are matrices consisting of the basis of $\mathcal{K}_m$ and $\mathcal{L}_m$, respectively.

In our case, although the stochastic Galerkin tensor C_{ijk} has a symmetric structure [109], $\bar{A}_i$ is asymmetric owing to the VMS formulation (Equation (3.74)). The common choice for solving a non-symmetric linear system under the Krylov subspace iterative method is the generalized minimal residual method (GMRES) [144]. For the global stochastic system, we employ a modified version of GMRES solver, which is the flexible generalized minimal residual method (FGMRES) [143]. GMRES and FGMRES belong to the Arnoldi process [7], which is characterized as a projection method onto $\mathcal{K}_m$ for the general non-Hermitian matrices. FGMRES generalizes the preconditioned GMRES method by enabling the feasibility of making use of different right preconditioners in each iteration and satisfies the minimization property of the residual over the preconditioned Krylov subspace as in the GMRES scheme.

Algorithm 1 Flexible GMRES (FGMRES) with right preconditioning [145]

```
 1: Choose an initial guess $U^{(0)}$
 2: Start: $r^{(0)} = b - AU^{(0)}$, $\beta = \|r^{(0)}\|_2$ , $v_1 = r^{(0)}/\beta$
 3: for $j = 1, ..., n$ do
 4:     $z_j := M_i^{-1} v_j$
 5:     $w := Az_j$
 6:     for $i = 1, ..., j$ do
 7:         $h_{i,j} := (w, v_i)$
 8:         $w := w - h_{ij} v_i$
 9:     end for
10:     $h_{J+1,J} = \|w\|_2$
11:     $v_{j+1} = w/h_{j+1,j}$
12:     Define $Z_n := [z_1, ..., z_n], \bar{H}_n = \{h_{i,j}\}_{1 \leq i \leq j+1, 1 \leq j \leq n}$
13: end for
14: Compute $y_n = \mathrm{argmin}_y \|\beta e_1 - \bar{H}_n y\|_2$
15: Set $U^{(n)} = U^{(0)} + Z_n y_n$
16: Restart: if convergence stop, else $U^{(0)} := U^{(n)}$, GoTo 2
```

Algorithm 1 provides a pseudo algorithm of the FGMRES method, the steps $3 - 13$ are basically the Arnoldi process. Usually, the FGMRES requires more memory resources than the GMRES, because an additional storage of the vectors z_i is required. On the other hand, the additional flexibility may cover the extra cost comparing with the standard GMRES, but more important it is beneficial for non-fixed preconditioners [145]. It is obvious that our large coupled system can be extremely difficult to resolve only with the FGMRES solver, preconditioning processes are definitely necessitated. Accordingly, the FGMRES plays only the role as the global linear solver in our application, it guards the overall linear equation, whereas the convergence speedup also benefits from our customized preconditioners. The Polynomial Chaos expansion Multilevel preconditioner is designed to cope with the stochastic Galerkin system and will be discussed in the next section.

4.3 Polynomial Chaos expansion (PCE) Multilevel Preconditioner for the Global Linear System

Developing an efficient preconditioner is very crucial for solving the global linear system in our problem, and it is also a major obstacle for applying the intrusive stochastic Galerkin approach. This section is devoted specially to the Polynomial Chaos expansion (PCE) Multilevel preconditioning technique (or spectral Galerkin Multilevel preconditioner). The PCE Multilevel preconditioner [149] exploits the recursive hierarchical structure of the stochastic Galerkin matrix. From another point of view, it also inherits the central idea from the Multigrid methods [70, 2, 142] by eliminating the error on different scales accordingly, in our case, on different Chaos Polynomial degrees.

The stochastic space is discretized by the Chaos Polynomials, with a polynomial degree L, or to put it differently, the stochastic space is discretized with $(P+1)$ Chaos Polynomials. We rewrite the expression of the stochastic space spanned by the Chaos Polynomials in Equation (3.39) by taking into account the polynomial degree L:

$$\mathcal{S}_L = \mathcal{S}^{P_L} = \mathrm{span}\{\psi_0, ..., \psi_{P_L}\} \ . \tag{4.14}$$

We can then denote a subspace of $\mathcal{S}_L$:

$$\mathcal{S}_l = \mathcal{S}^{P_l} = \mathrm{span}\{\psi_0, ..., \psi_{P_l}\} \subseteq \mathcal{S}_L \ , \tag{4.15}$$

where, $0 < l \leq L$, $P_l + 1 := \frac{(M+l)!}{M!l!}$. The hierarchy is naturally defined by a nested sequence of spaces:

$$\mathcal{S}_0 \subseteq \mathcal{S}_1 \subseteq \cdots \subseteq \mathcal{S}_L \ . \tag{4.16}$$

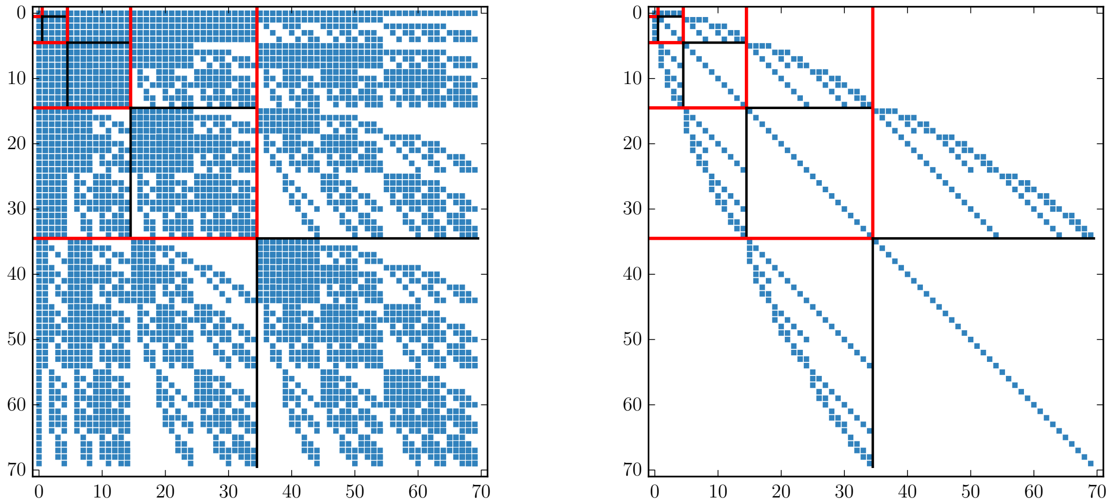
Therefore, the matrix K_i in Equation (3.100) has also a hierarchical structure, it reads (here K_l = K_i):

$$K_l := \begin{bmatrix} \tilde{K}_l & E_l \\ E_l^T & F_l \end{bmatrix} \ , \quad K_l \in \mathbb{R}^{P_l+1, P_l+1} \ , \tag{4.17}$$

with $\tilde{K}_l \in \mathbb{R}^{P_{l-1}+1, P_{l-1}+1}$, $E_l \in \mathbb{R}^{P_{l-1}+1, P_l - P_{l-1}}$, $F_l \in \mathbb{R}^{P_l - P_{l-1}, P_l - P_{l-1}}$. By recursion, the sub-matrix $\tilde{K}_l$ has the same construction as K_{l-1} and corresponds to the Chaos Polynomial degree $(l-1)$. The recursion terminates with $\tilde{K}_l \in \mathbb{R}^{1,1}$, which indicates the zero-th order Chaos Polynomial $(l = 0)$.

Corollary 4.3.1. *If the random inputs have only a first order expansion, then the off-diagonal elements of the sub-matrix F_l are all zero.*

The proof of Corollary 4.3.1 can be found in [141], a graphical illustration is shown in Figure 4.3



(a) $M = 4$, $L = 4$, $L_t = 4$.

(b) $M = 4$, $L = 4$, $L_t = 1$.

Figure 4.3: Sparsity patterns of the coefficients in matrix K_i.

We denote the solution vector with respect to the Polynomial Chaos expansion order, namely the PC level, which is given as:

$$U^l := [\ U_0, ..., U_{P_l}]\ , \quad l = 0, ..., L\ , \tag{4.18}$$

the corresponding matrix is defined by:

$$A^l := \sum_{i=0}^{P_l} (K_i \otimes \bar{A}_i)\ , \quad l = 0, ..., L\ . \tag{4.19}$$

Hence, we can adapt the concept of the sub-grids from the standard Multigrid algorithms to the subspaces $\mathcal{S}_l$ of the discretized stochastic space $\mathcal{S}_L$ by supposing the "grid level" to be l in Equation (4.15). In the following, we introduce the restriction operator $\mathcal{R}_l$ and the prolongation operator $\mathcal{P}_l$:

$$\mathcal{R}_l := \begin{bmatrix} I_{P_{l-1}} & \mathbf{0} \end{bmatrix} \otimes I_{(d+1)N}\ , \quad \mathcal{P}_l := \begin{bmatrix} I_{P_l} \\ \mathbf{0} \end{bmatrix} \otimes I_{(d+1)N}\ . \tag{4.20}$$

A smoothing process can be written as in Equation (4.21) at the corresponding level l. It is given by:

$$U^{l,(n+1)} := \mathcal{B}_l U^{l,(n)} = U^{l,(n)} + (I_l \otimes \bar{A}_0)^{-1}(b^l - (\sum_{i=0}^{P_l} K_i \otimes \bar{A}_i) U^{l,(n)})\ . \tag{4.21}$$

Here, n denotes the n-th iteration, $I_l \in \mathbb{R}^{P_l+1, P_l+1}$ is the identity matrix, $\mathcal{B}_l := (I_l \otimes \bar{A}_0)^{-1}$ denotes as the smoother, which is constructed by the Mean-based preconditioner [150] with utilizing an approximation of the inverse of the mean block A^0. The detailed information about this preconditioning technique will be presented in Section 4.4.

Algorithm 2 PCE Multilevel preconditioner/solver : M_{ML} [150]

```
 1: if $l = 0$ then
 2:     solve $A^0 U^0 = b^0$
 3: else
 4:     $U^l = \mathcal{B}_l^{\nu_1} U^l$
 5:     $r^l = b^l - \sum_{i=0}^{P_l} (K_i \otimes \bar{A}_i) U^l$
 6:     $r^{l-1} = \mathcal{R}_l r^l$
 7:     for $i = 1$ to $\mu$ do
 8:         $ML(b^{l-1}, U^{l-1}, l-1)$
 9:     end for
10:     $c^l = \mathcal{P}_l c^{l-1}$
11:     $U^l = U^l + c^l$
12:     $U^l = \mathcal{B}_l^{\nu_2} U^l$
13: end if
```

Algorithm 2 illustrates one cycle of the Multilevel preconditioner M_{ML}. The function $ML(U^l, b^l, l)$ takes the vector U^l and the right-hand side b^l on level l. The step 4 and 12 represent the pre- and post-smoothing process with the user-defined iteration parameters ν_1 and ν_2. μ represents the option setting for the V-cycle ($\mu = 1$) or the W-cycle ($\mu = 2$). The information exchange between $\mathcal{S}^{l-1}$ and $\mathcal{S}^l$ is done via the restriction and prolongation operators (Equation (4.20)), the steps 6 and 10 are the restriction and prolongation operations, respectively. When $l = 0$, Algorithm 2 tries to solve the deterministic problem, i.e. $A^0 U^0 = b^0$.

Remark 4.3.1. *The restriction operation and the prolongation operation can be implemented implicitly if the stochastic vector is built as a combination of sub-vectors U_i explicitly. Therefore, the restriction implies removing the sub-vectors, and the prolongation implies appending the vector with zero entries.*

The practical consideration of employing the spectral Galerkin Multilevel preconditioner is to calculate the inverse of A^0. Here, $A^0 = \bar{A}_0$. As the analytical expression of the inversion of VMS is not available in our case, $(A^0)^{-1}$ ought to be obtained by applying solvers directly on a linear system, e.g. $A^0 x = b$. Owing to the size of our problem, iterative solvers are preferable, one of the center complications of the Multilevel preconditioner is then about solving the deterministic VMS problem.

4.4 Stochastic Mean-Based Preconditioner for the Lobal Linear System

The Mean-based preconditioner [137, 51] is a simple but powerful preconditioning technique for the stochastic Galerkin system. This approach is in effect a block-diagonal preconditioner, which incorporates only the mean block. It has been shown numerically to be very efficient for the problems with a moderate variance with respect to the mean, that is to say, the stochastic Galerkin matrix K_i must be approximately diagonal dominant.

Note that, for our final blood pump simulation, the Multilevel preconditioner is applied instead of the Mean-based technique on the global linear system because of the better convergence behavior. Moreover, a numerical comparison between the Mean-based preconditioner and the different Multilevel preconditioners are provided in Chapter 6 in order to support this argument. On the other hand, the smoother proposed in Section 4.3 for the PCE Multilevel technique corresponds to the Mean-based approach. The Mean-based preconditioning can be conducted by considering a splitting method [141], i.e.:

$$\sum_{i=0}^{P} K_i^{+} \otimes \bar{A}_i U^{(n+1)} = b - \sum_{i=0}^{P} K_i^{-} \otimes \bar{A}_i U^{(n)} \ , \tag{4.22}$$

the Mean-based approach is regarded as a Jacobi-type splitting of K_i, which yields:

$$I_P \otimes \bar{A}_0 U^{(n+1)} = b - \sum_{i=1}^{P} K_i \otimes \bar{A}_i U^{(n)} \ . \tag{4.23}$$

Here, the preconditioner matrix is defined as:

$$M_{Mean} := I_P \otimes \bar{A}_0 \ , \quad \bar{A}_0 \in \mathbb{R}^{(d+1)N,(d+1)N} \ . \tag{4.24}$$

The most attractive feature about the Mean-based preconditioner is the effortless parallelization. The parallel scheme can be referred to as the block-Jacobi preconditioner acting on each stochastic mode, which infers that the parallelization is achieved on the spatial and stochastic level simultaneously. Therefore, this method enables the possibility of handling a larger size stochastic problem in comparison to the Multilevel method.

4.5 Schur Complement Preconditioner for Mean Block

As mentioned in Section 4.4, the linear system of the mean block needs to be solved repeatedly in Algorithm 2 because of the smoothing process (i.e. the Mean-based preconditioner). Concerning the mean block, we deploy again the Krylov subspaces solver FGMRES owing to the same arguments in Section 4.2. Furthermore, we suggest using the Schur Complement method to precondition the linear system resulting from the deterministic VMS formulation. The linear equation involved with the mean block can be written as:

$$\bar{A}_0 u = \begin{bmatrix} A_0 & B_0 \\ C_0 & D_0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} = \bar{b} \ . \tag{4.25}$$

We extend the notation of the linear system arising from VMS by introducing the sub-matrices of $\bar{A}_0$ (Equation (4.25)), in which A_0 is assumed to be nonsingular. $\bar{b}$ is any right-hand side vector from the preconditioning process of the global linear system. The block preconditioner technique is deployed for our mean block solving process as it has been shown to be flexible and robust [145]. For constructing a block preconditioner, we start first with the LDU decomposition of $\bar{A}_0$:

$$\bar{A}_0 := \mathcal{L}_b \mathcal{D}_b \mathcal{U}_b = \begin{bmatrix} A_0 & B_0 \\ C_0 & D_0 \end{bmatrix} = \begin{bmatrix} I & 0 \\ C_0 A_0^{-1} & I \end{bmatrix} \begin{bmatrix} A_0 & 0 \\ 0 & S \end{bmatrix} \begin{bmatrix} I & A_0^{-1} B_0 \\ 0 & I \end{bmatrix} . \tag{4.26}$$

We focus here directly on the matrix $\bar{A}_0$ in Equation (3.100), where $\bar{A}_0$ is the mean block associated with VMS formulation. Therefore, in Equation (4.25), x is the velocity vector, y is the pressure vector. In Equation (4.26), A_0 is the velocity-velocity block, B_0 is the velocity-pressure block, C_0 is the pressure-velocity block, D_0 is the pressure-pressure block. The most block preconditioners are built on a mixture of the sub-matrices above. Especially, S is known as the Schur Complement matrix, and it is given by:

$$S := D_0 - C_0 A_0^{-1} B_0 . \tag{4.27}$$

The block Schur Complement states as:

$$Sy = g - C_0 A_0^{-1} f , \tag{4.28a}$$

$$A_0 x = f - B_0 y . \tag{4.28b}$$

Here, Equation (4.28a) is referred as the pressure Schur Complement equation, and the solution process of the coupled system develops in the first place into a scalar problem of the pressure only. Once the solution of the pressure is available, the velocity vector is computed subsequently with Equation (4.28b). However, one practical concern is that the matrix S^{-1} must be easily accessible, as it can not be calculated explicitly. Especially for the finite element method, matrix A_0 is sparse, but its inverse is generally a dense matrix.

Finding an efficient way to solve Equation (4.28a) is vital. However, S^{-1} is normally calculated via some iterative methods, in other words, only the matrix-vector multiplication operation is required. Therefore, the splitting process (Equation (4.28)) can be very efficient as inverting the sub-matrices is less complicated than inverting the matrix $\bar{A}_0$.

The general approach of dealing with the inversion of S is to solve Equation (4.28a) by an iterative solver with an appropriate preconditioner $\tilde{S}_{app}$ [168]. The right preconditioning yields:

$$S\tilde{S}_{app}^{-1} y' = (g - C_0 A_0^{-1} f) , \quad \text{with } y = \tilde{S}_{app}^{-1} y' . \tag{4.29}$$

Here, $\tilde{S}_{app}$ is an approximation of matrix S, it must be constructed easily. There are several ways to build $\tilde{S}_{app}$, e.g., SIMPLE iteration [132, 73] suggests replacing A_0^{-1} by $\text{diag}(A_0)^{-1}$ or I_{A_0}. Or the pressure convection diffusion (PCD) preconditioner [152, 102, 154] suggests $\tilde{S}_{app}^{-1} = M_p^{-1} F_p A_p^{-1}$, where M_p and A_p are the projections of mass matrix and Neumann Laplacian matrix on the pressure, F_p is the projection of the velocity operator [117] in the context of the incompressible Navier-Stokes equations.

We state the Schur Complement algorithm (Algorithm 3), which is implemented in our flow solver. The FGMRES solver is explicitly imposed in the step 6, as $\tilde{S}_{app}^{-1}$ needs to be obtained numerically with an iterative solver. In our computation, $\tilde{S}_{app}$ is defined as follows:

$$\tilde{S}_{app} := D_0 + \varepsilon_1 M_p + \varepsilon_2 L_p , \tag{4.30}$$

Algorithm 3 Schur Complement

```
Given $\bar{A}_0$, $b$
Extract $(A_0, B_0, C_0, D_0)$ from $\bar{A}_0$, $(f, g)$ from $b$
Solve $A_0 h_x = f$
$h_y = C_0 h_x$
$g := g - h_y$
Solve $Sy = g$ by FGMRES with preconditioner $\tilde{S}_{app}^{-1}$
$f := f - B_0 y$
Solve $A_0 x = f$
Return $u := \begin{bmatrix} x & y \end{bmatrix}^T$
```

where $\varepsilon_1 = \nu$, $\varepsilon_2 = -\theta \Delta t^2 \nu$. θ is the control parameter of the θ-scheme for the time discretization. M_p is the pressure mass matrix, and L_p is the pressure Laplace matrix. This configuration is based on the augmented Lagrangian (AL) approach [66, 26, 168, 26, 57], they proposed an expression built on a linear combination of the pressure mass matrix and D_0. It is given as:

$$\tilde{S}_{app} = D_0 + \varepsilon M_p \; . \tag{4.31}$$

We have to mention that the block matrices of the VMS formulation are different from the general incompressible NSE, precisely:

$$\bar{A}_0 = \begin{bmatrix} A_{ns} & B_{ns} \\ C_{ns} & 0 \end{bmatrix} + \begin{bmatrix} A_{vms} & B_{vms} \\ C_{vms} & D_{vms} \end{bmatrix} \; . \tag{4.32}$$

Here, the first matrix represents the incompressible NSE block structure, the second matrix provides the sub-blocks derived from the VMS formulation with $D_{vms} = D_0$ (in Equation (4.25)). Hence, one possible simplification is to omit the contribution from $\{A_{vms}, B_{vms}, C_{vms}\}$ and only take D_{vms} into account for designing $\tilde{S}_{app}$, because the second matrix is negligible compared to the first matrix. However, the stabilized Navier-Stokes scheme offers the possibility to access the pressure-pressure block directly, then only the $C_0 A_0^{-1} B_0$ part needs to be approximated. Furthermore, one correction matrix $\varepsilon_2 L_p$ is also included into $\tilde{S}_{app}$ in order to improve the convergence rate. Still, ε_2 has the same order as Δt^2, when the time step is small, this correction matrix does not have a significant influence.

Hereon, we state the block preconditioner of our Schur Complement system, which can be further used by Algorithm 1, it is given as:

$$M_{SC} := \begin{bmatrix} A_0 & B_0 \\ 0 & \tilde{S}_{app} \end{bmatrix} \; . \tag{4.33}$$

As M_{SC} plays only the role of the preconditioner in the FGMRES algorithm, the accuracy of the approximation to S does not necessitate to be very strict, a trade-off between the computational cost of $\tilde{S}_{app}$ and the amount of FGMRES iterations needs to be taken into account.

In regard to Algorithm 3, several sub-solvers are involved, some further information is listed below (Table 4.1):

For the sake of completeness of Chapter 4, we also state the algorithm of the Conjugate Gradient (CG) and the Algebraic Multigrid [75, 77] in Algorithm 4 and Algorithm 6, respectively. Besides, the algorithm of GMRES is not given here because it is very similar to the FGMRES.

The Multigrid preconditioners have been shown to be very efficient in highly parallel systems [33, 49, 14], though the fine properties are originally based on the system matrix A being a M-matrix [135, 98]. However, concerning Table 4.1, A_0 and $\tilde{S}_{app}$ are mass matrix dominant, both are suitable to be associated with Multigrid preconditioners.

Before exercising the AMG algorithm (Algorithm 6), some components need to be defined in advance. Let us consider an abstract linear equation [75, 179]:

System matrix	Solver	Preconditioner
A_0	GMRES	AMG
S	FGMRES	$\tilde{S}_{app}^{-1}$
$\tilde{S}_{app}$	CG	AMG

Table 4.1: Sub-solvers with corresponding preconditioners in Algorithm 3.

Algorithm 4 CG with preconditioning [145] ($Ax = b$)

```
Choose an initial guess $u^{(0)}$
Start: $r^{(0)} = b - Ax^{(0)}$, $z_0 = M^{-1}\tilde{r}_0$, $p_0 = z_0$
for $j = 1, ...,$ until convergence do
    $\alpha_j = (r_j, z_j)/(Ap_j, p_j)$
    $x_{j+1} = x_j + \alpha_j p_j$
    $r_{j+1} = r_j - \alpha_j Ap_j$
    $z_{j+1} = M^{-1} r_{j+1}$
    $\beta_j = (r_{j+1}, z_{j+1})/(r_j, z_j)$
    $p_{j+1} = z_{j+1} + \beta_j p_j$
end for
```

$$Ax = b \; . \tag{4.34}$$

We use the superscripts to indicate the level number and set 1 to represent the finest level, i.e. $A^1 = A$ and $\Omega^1 = \Omega$. Ω is the pre-defined grid. The necessary components are listed as follows:

1. Grids $\Omega^1 \supset \Omega^2 \supset \cdots \supset \Omega^M$ with subsets:
   set of coarse points: $C^k, k = 1, 2, ..., M-1$, set of fine points: $F^k, k = 1, 2, ..., M-1$.
2. Grid operators: $A^1, A^2, ..., A^{M-1}$.
3. Interpolation operators: $P^k, k = 1, 2, ..., M-1$.
4. Restriction operators: $R^k, k = 1, 2, ..., M-1$.
5. Smoothers: $S^k, k = 1, 2, ..., M-1$.

Algorithm 5 AMG preconditioner/solver setup [179]

```
Set $k = 1$
Partition $\Omega^k$ into disjoint sets $C^k$ and $F^k$
Set $\Omega^{k+1} = C^k$
Define interpolation operator $P^k$
Define Restriction operator $R^k$
Setting $S^k$
if $\Omega^{k+1}$ is mall enough then
    $M := k + 1$
else
    GoTo 2
end if
```

These components above are constructed in a setup phase of AMG preconditioner, which is defined as in Algorithm 5.

Algorithm 6 AMG preconditioner/solver (V-cycle): M_{AMG} [75] $(Ax = b)$

```
if $k = M$ then
    Solve $A^M x^M = b^M$
else
    Apply smoother $S^k$ $\mu_1$ times on $A^k x^k = b^k$
    $r^k = b^k - A^k x^k$
    $r^{k+1} = R^k r^k$
    $M_{AMG}(A^{k+1}, R^{k+1}, P^{k+1}, S^{k+1}, e^{k+1}, r^{k+1})$
    $e^k = P^k e^{k+1}$
    $x^k := x^k + e^k$
    Apply smoother $S^k$ $\mu_2$ times on $A^k x^k = b^k$
end if
```

Moreover, we use directly an external library, Hypre [53], in our solver configuration, as a setup phase for the Multigrid preconditioner is tedious and thorny. The supplementary knowledge about the considered linear system is always preferable, the available options of the BoomerAMG are referred to [53, 75].

Remark 4.5.1. *Algorithm 2 inherits the main features from Algorithm 6, such as the coarse-grid correction, restriction, prolongation, etc. Yet, the PCE Multilevel preconditioner is even more straightforward on building different "sub-grids", because the hierarchy nature is directly deployed.*

4.6 Summary of Solvers and Preconditioners

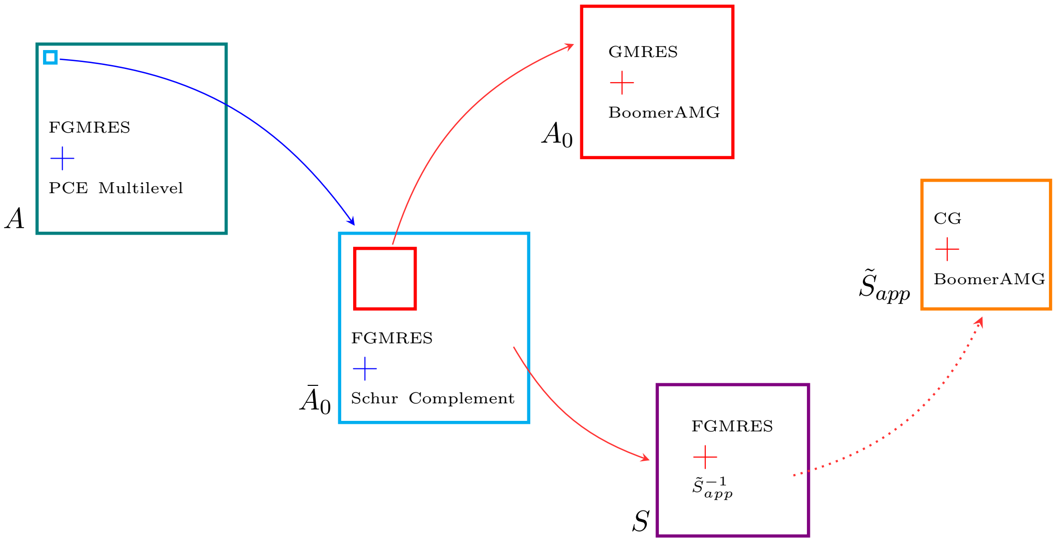


Figure 4.4: The combination of solvers and preconditioners on the different levels.

We give a brief summary about the solvers and the preconditioners which are employed in the final blood pump simulation. Figure 4.4 illustrates the combination of the solvers and the preconditioners on different linear systems. The ultimate goal of these solvers is to obtain a correct solution on the global linear system in terms of fulfilling the convergence criteria in the Newton scheme. For that reason, certain linear solvers and preconditioners in Figure 4.4 do not have to provide high quality results as long as the upper solving process can still converge. In other words, under the context of the Krylov subspace solvers, the search direction just needs to be appropriate. Hence, the FGMRES solver on the matrix A requires only a low quality solution from the PCE Multilevel preconditioner, as well as for the FGMRES solver and the Schur Complement preconditioner on $\bar{A}_0$. The blue line between block A and $\bar{A}_0$ indicates that the preconditioner (PCE Multilevel) demands only a rough quality of the solution from the inner linear solver (FGMRES). For the reason that the FGMRES on $\bar{A}_0$ plays the role as the smoother. The red lines signify the requirement of a high quality resolution from the upper preconditioner to the lower linear solvers, and the dotted line intimates that the matrix $\tilde{S}_{app}$ is obtained partially from S. Furthermore, we control the quality of solutions and the solving processes by setting a large relative error or a limited number of iterations, even using a combination of both.

Chapter 5

Scalability of Solvers and Preconditioners

In the previous chapters, the basic theories and the solving procedures for the blood pump simulation were presented. The turbulent character of the blood flow is modeled by the Variational Multiscale method as described in Chapter 2. The moving geometry is conducted by the shear layer update approach under the framework of the finite element method (Chapter 2). The propagation of the uncertainties from the random inputs is simulated with the stochastic Galerkin method as described in Chapter 3. Moreover, the solving strategies are explicitly defined in Chapter 4. We consider the Krylov subspace solvers as sub-solvers, and we combine them into our customized preconditioners, e.g., the PCE Multilevel and the Schur Complement preconditioner. Yet, for a time dependent high-dimensional coupled system combined with UQ, the performance of the solving process must also be prioritized. In this chapter, we show the scalability results of the flow solver and the PCE Multilevel preconditioner. The flow solver test is studied with a deterministic channel flow, as the mean block is used as the smoother in the PCE Multilevel preconditioner, then the efficiency of solving the deterministic flow equation is important. The scalability of the PCE Multilevel preconditioner is studied with a stochastic Poisson equation, because the deterministic Poisson problem can be solved efficiently, such that we can focus on the efficiency of the Multilevel preconditioner. Two different solving strategies of the PCE Multilevel preconditioner are also compared. Moreover, a modification of the Multilevel method is studied with a 2D channel flow problem in order to further reduce the computing time of our stochastic flow problem. We could not show the scalability study based on the blood pump simulation due to the large problem size and the available computing resources. However, the important parts which are involved in the blood pump computation, such as the flow solver and the PCE Multilevel preconditioner, are studied in this chapter.

5.1 Flow Solver

Let us focus on the VMS formulation for a deterministic system, especially with the inflow and outflow conditions (i.e. not a closed flow problem), which imitate the conditions as they also occur in the blood pump instrument. The channel flow with an obstacle problem [147, 21] is taken to be the test example for our scalability test. The corresponding VMS formulation (Chapter 2) with the boundary conditions is given below:

Find $\boldsymbol{u}_h \in V^h$, $p_h \in Q^h$:

$$\begin{aligned}&(\frac{\boldsymbol{u}_h}{\partial t}, \boldsymbol{v}_h) + (\hat{\boldsymbol{u}}_h \cdot \nabla \boldsymbol{u}_h, \boldsymbol{v}_h) \\ +&\frac{\mu}{\rho}(\nabla \boldsymbol{u}_h, \nabla \boldsymbol{v}_h) - \frac{1}{\rho}(p_h, \nabla \cdot \boldsymbol{v}_h)\end{aligned} \tag{5.1a}$$

$$
\begin{aligned}
+(\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \hat{\boldsymbol{u}}_h \cdot \nabla \boldsymbol{v}_h) + (\tau_C r_{C,h}, \nabla \cdot \boldsymbol{v}_h) &= (\boldsymbol{f}_h, \boldsymbol{v}_h) \ , && \text{in } [0,T] \times \mathcal{D} \ , \\
(\nabla \cdot \boldsymbol{u}_h, q_h) + (\boldsymbol{\tau}_M \boldsymbol{r}_{M,h}, \nabla q_h) &= 0 \ , && \text{in } [0,T] \times \mathcal{D} \ . && \text{(5.1b)}
\end{aligned}
$$

$\forall \boldsymbol{v}_h \in V^h$, $\forall q_h \in Q^h$. Here,

$$
\begin{aligned}
\hat{\boldsymbol{u}}_h &= \boldsymbol{u}_h - \boldsymbol{\tau}_M \boldsymbol{r}_{M,h} \ , && \text{in } \mathcal{D} \times [0,T] \ , && \text{(5.2a)} \\
\boldsymbol{r}_{M,h} &= \frac{\partial \boldsymbol{u}_h}{\partial t} + \boldsymbol{u}_h \cdot \nabla \boldsymbol{u}_h - \frac{\mu}{\rho} \Delta \boldsymbol{u}_h && && \text{(5.2b)} \\
&\quad + \frac{1}{\rho} \nabla p_h - \boldsymbol{f}_h \ , && \text{in } \mathcal{D} \times [0,T] \ , \\
r_{C,h} &= \nabla \cdot \boldsymbol{u}_h \ , && \text{in } \mathcal{D} \times [0,T] \ , && \text{(5.2c)} \\
\boldsymbol{u}_h &= \begin{bmatrix} 16 U_{max} y z (H-y)(H-z)/H^4 \\ 0 \\ 0 \end{bmatrix} \ , && \text{on } \Gamma_{in} \times [0,T] \ , && \text{(5.2d)} \\
(-\mathbb{1} p_h + \frac{\mu}{\rho} \nabla \boldsymbol{u}_h) \cdot \boldsymbol{n} &= \boldsymbol{0} \ , && \text{on } \Gamma_{out} \times [0,T] \ , && \text{(5.2e)} \\
\boldsymbol{u}_h &= \boldsymbol{0} \ , && \text{on } \Gamma_{wall} \times [0,T] \ . && \text{(5.2f)}
\end{aligned}
$$

$\boldsymbol{u}_h$ and p_h are the velocity and the pressure. $\boldsymbol{v}_h$ and q_h are the test functions for the velocity and the pressure. $\boldsymbol{r}_{M,h}$, $r_{C,h}$ are the residuals for the momentum and continuity equation, respectively. H is the height and the width of the channel. μ is the dynamic viscosity. ρ is the density. U_{max} is the maximal inflow velocity and plays only an effect on the direction of $\boldsymbol{x}$-axis. Γ_{in}, Γ_{out} and Γ_{wall} are the boundaries of inflow, outflow and solid walls, respectively. The channel geometry is detailed in [147], P1-P1 elements are utilized for the velocity and the pressure accordingly.

Physical parameter					
ρ	1035 kg/m^3	μ	0.0035 Pa · s	U_{max}	0.5 m/s
H	0.41 m				
Time discretization					
θ-Scheme	0.5	Δt	0.001 s		
Nonlinear solver (Newton)					
Tol_{abs}	1.0×10^{-10}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000
Forcing	EisenstatWalker2				
Linear solver (FGMRES)					
Tol_{abs}	1.0×10^{-10}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000
Preconditioner (Schur Complement)					
FGMRES for block S					
Tol_{abs}	1.0×10^{-16}	Tol_{rel}	1.0×10^{-16}	$Iter_{max}$	10
GMRES for block A_0 (preconditioned with BoomerAMG)					
Tol_{abs}	1.0×10^{-10}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000
CG for matrix $\tilde{S}_{app}$ (preconditioned with BoomerAMG)					
Tol_{abs}	1.0×10^{-10}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000

Table 5.1: General settings for the flow solver test.

Table 5.1 shows the basic settings for the flow solver test. Especially the physical parameters are chosen the same as for the blood simulation. It means that the flow solver test is simulating the same fluid as in the pump device. As mentioned in Chapter 4, the Schur Complement preconditioner is restrained by the total number of the iterations ($Iter_{max} = 10$), which implies that the solution quality obtained from the Schur Complement method is rather low but still good enough to precondition the linear system.

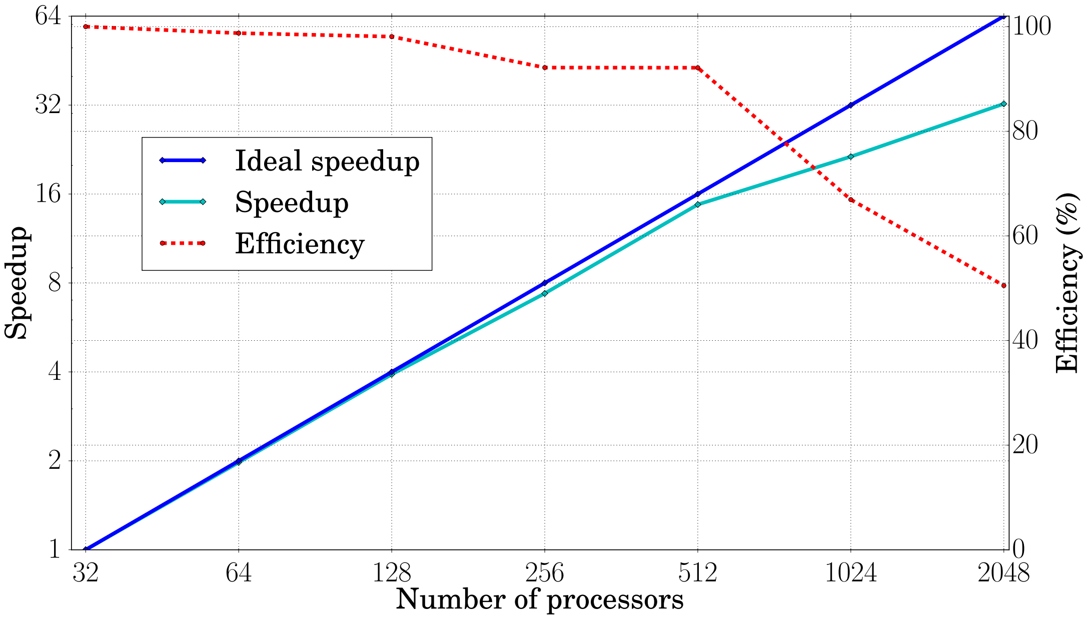


Figure 5.1: Scalability test for the flow solver.

Besides, the mesh contains 8, 257, 280 tetrahedrons, it results 5.7 Millions degrees of freedom (DOFs) with using P1-P1 linear elements. Figure 5.1 shows the scalability test of the flow solver with the settings mentioned above based on the first 100 time steps. The scalability test starts with 32 processors instead of 1 processor owing to the size of the problem (5.7 Millions DOFs). The speedup S_p and the efficiency E_p are defined as follows:

$$S_p := \frac{T_o}{T_p}\ , \qquad E_p := \frac{S_p}{p}\ . \tag{5.3}$$

The time T_p is the total solving time for the first 100 time steps, p stands for the number of the processors used, T_o is the time consumed by the minimum number of the processors, i.e., $T_o = T_{32}$. One can observe from Figure 5.1, the speedup performs favorably up to 512 cores. The efficiency drops down to 55% approximately at 2048 cores, because the communication time increases. However, the flow solver can still handle more than 2000 processors with a reasonable performance, which is crucial in practice for the PCE Multilevel preconditioner to be efficient. Figure 5.2 provides the time consumption for the different number of processors used. The reduction rate becomes very small on the last two columns.

In practice, the number of the iterations of the iterative solvers can be influenced by the problem size and the number of the processors. Figure 5.3 shows the amount of the iterations for solving a linear system associated with the matrix A_0 and the matrix S (Equation (4.28)), as both procedures are important for the Schur Complement's performance (Algorithm 3). One can observe that the amount of the iterations remains stable for the different quantities of the computing resources or the different problem sizes. It has to be mentioned that the effective performance benefits from the BoomerAMG solver [53], which guarantees that the number of the iterations has the minimum dependence of the mesh size and the number of the processors.

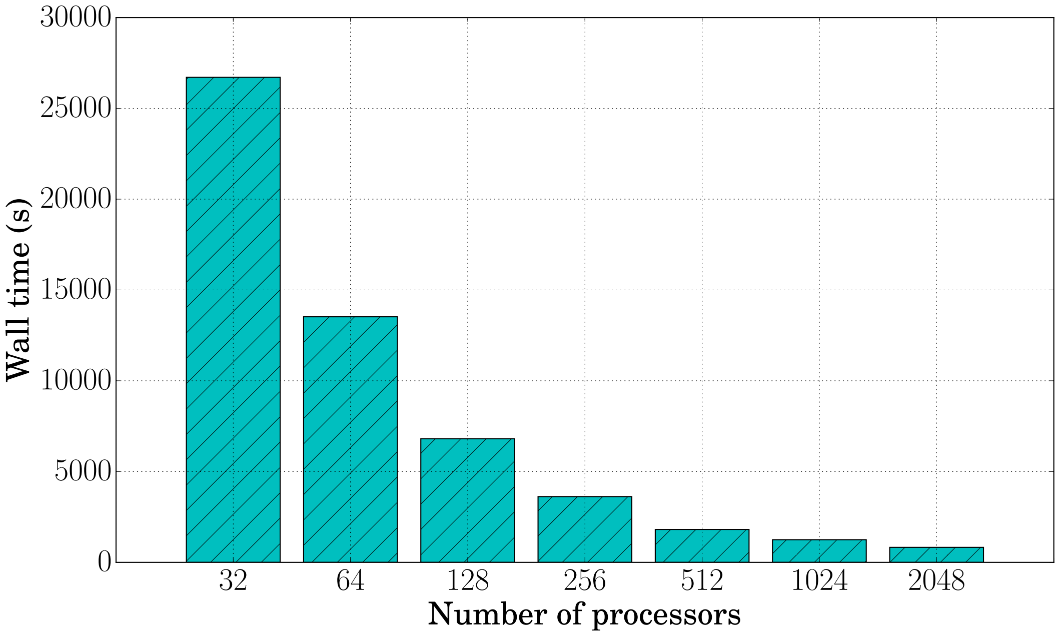
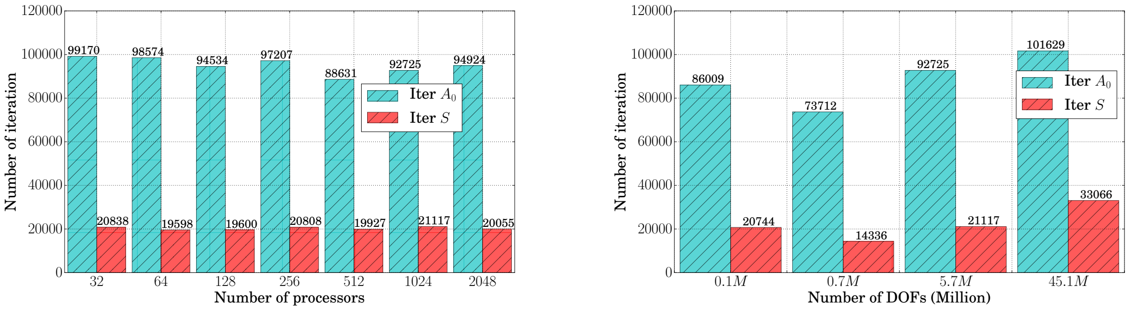


Figure 5.2: Computation time in seconds of the flow solver.



(a) Comparison of the number of iterations for different numbers of processors for a problem size of 5.7 Millions.

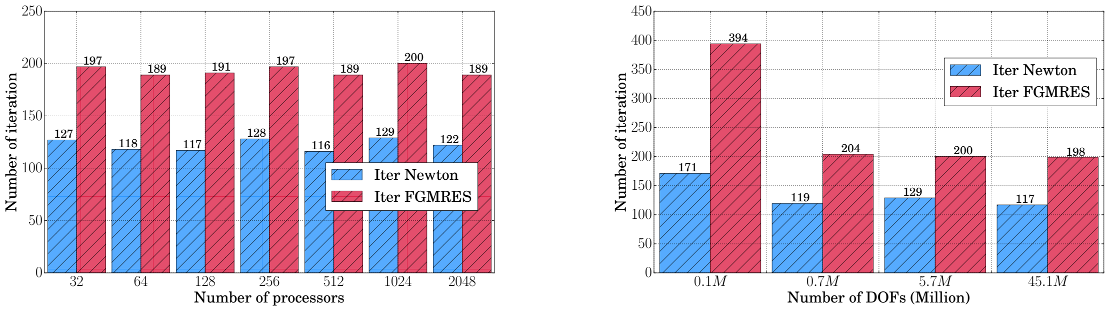
(b) Comparison of the number of iterations for different problem sizes by using 1024 processors.

Figure 5.3: The total number of the iterations of solving the linear system associated with the matrix A_0 (cyan) and the matrix S (red) for the first 100 time steps.

Furthermore, we can also state that the Schur Complement method gives a remarkable improvement in preconditioning our system with the FGMRES solver. Figure 5.4 illustrates the amount of Newton iterations and FGMRES iterations for the first 100 time steps, both quantities remain stable when the problem sizes increases. Additionally, each Newton step demands in average only two FGMRES iterations (Figure 5.4), it means that the Schur Complement algorithm can precondition our linear system in a very efficient way.

5.2 PCE Multilevel Preconditioner

Another important component of our solving strategy is the PCE Multilevel Galerkin spectral preconditioner (Section 4.3), which we use to cope with the global stochastic system. The main idea behind this preconditioner is to utilize the degree of the Chaos Polynomials as the level of the hierarchical structure in the stochastic spectral system. Based on the Multigrid



(a) Comparison of the number of iterations for different numbers of processors for a problem size of 5.7 Millions.

(b) Comparison of the number of iterations for different problem sizes by using 1024 processors.

Figure 5.4: The total number of Newton iterations (blue) and the FGMRES iterations (coral) for the first 100 time steps.

method, a satisfactory convergence property is achievable once we want to deal with a large dimensional problem. In contrast to Section 5.1, in this section we exploit a stochastic system based on the Poisson equation, where the viscosity parameter ν is designed as a multidimensional random variable. Because of the reduced complexity in the testing problem, we can focus on the performance of the Multilevel preconditioner. The governing equation is stated as below:

$$-\nabla \cdot (\nu(\boldsymbol{x},\theta)\nabla u(\boldsymbol{x},\theta)) = 1 \ , \qquad \text{in } \mathcal{D}, \theta \in \Theta \ , \tag{5.4a}$$

$$u(\boldsymbol{x},\theta) = 0 \ , \qquad \text{on } \partial\mathcal{D}, \theta \in \Theta \ . \tag{5.4b}$$

Here, Θ is the sample space, and θ represents the random event. We introduce an uncertain model of ν:

$$\nu(\boldsymbol{x},\theta) := \nu_0 + \nu_0\sigma \sum_{i=1}^{M} q^{i-1} \sin(2\pi i x_1)\sin(2\pi i x_2)\sin(2\pi i x_3)\xi_i(\theta) \ . \tag{5.5}$$

$\boldsymbol{\xi}(\theta) := [\xi_1(\theta), ..., \xi_M(\theta)]$ represents the random vector. We assume each of the components to be uniformly distributed on the interval $[-1,1]$, i.e. $\xi_i \sim \mathcal{U}(-1,1)$. Additionally they are assumed to be mutually independent. Concerning Equation (5.5), the mean value of the ν is exactly ν_0, so the second part of the model can be regarded as the artificial noise around ν_0, i.e. $\mathbb{E}(\nu) = \nu_0$. Moreover, σ is a variation factor, which parametrizes the magnitude of the fluctuations with respect to ν_0. q denotes the decay rate of the fluctuations. In order to ensure this problem to be well-posed, we impose one extra condition:

$$0 < q < 1 \ , \quad 0 < \frac{\sigma}{1-q} < 1 \ . \tag{5.6}$$

Therefore, we can employ the Polynomial Chaos expansion method and assume a square-integrable stochastic function $u(\boldsymbol{x},\cdot) \in L^2(\Xi)$, for all $\boldsymbol{x} \in [0,1]^3$ with Ξ being the range of the variables $\boldsymbol{\xi}$. The truncated stochastic spectral expansion of the solution $u(\boldsymbol{x},\boldsymbol{\xi})$ is given as:

$$u(\boldsymbol{x},\boldsymbol{\xi}) = \sum_{i=0}^{P} u_i(\boldsymbol{x})\psi_i(\boldsymbol{\xi}) \ . \tag{5.7}$$

Here, $P+1 = \frac{(M+L)!}{M!L!}$. M is the number of random variables, and L is the maximum polynomial order of ψ_i. u_i and ψ_i are the deterministic part and stochastic part respectively, $\psi_i(\boldsymbol{\xi})$ are the pre-defined basis functions. Because all ξ_i are modeled with the standard Uniform distribution in our example, $\psi_i(\boldsymbol{\xi})$ consequently denote the multivariate Legendre polynomials (Table 3.2). Further, Equation (5.4) can be expressed with the random variables $\boldsymbol{\xi}$ instead of θ. Thus, we insert then Equation (5.7) into Equation (5.4).

In order to apply the generalized Polynomial Chaos expansion approach, we multiply the preselected random basis polynomials on both sides of Equation (5.4a). As $u(\boldsymbol{x}, \theta)$ in Equation (5.4a) is extended with Equation (5.7), it gives:

$$- \langle \nabla \cdot (\nu(\boldsymbol{x}, \boldsymbol{\xi}) u(\boldsymbol{x}, \boldsymbol{\xi}), \psi(\boldsymbol{\xi}) \rangle = \langle 1, \psi(\boldsymbol{\xi}) \rangle \ , \quad \psi \in \mathcal{S}^P \ , \tag{5.8}$$

where $u(\boldsymbol{x}, \cdot) \in \mathcal{S}^P$, $\mathcal{S}^P$ is the approximated stochastic space, $\mathcal{S}^P = \operatorname{span}\{\psi_0, \psi_1, ..., \psi_P\}$. The viscosity ν is also projected into $\mathcal{S}^P$, which implies:

$$\nu(\boldsymbol{x}, \boldsymbol{\xi}) := \sum_{i=0}^{M} \nu_i \psi_i(\boldsymbol{\xi}) \ , \tag{5.9}$$

$$\begin{aligned} \nu_i(\boldsymbol{x}) &:= \nu_0 \ , && i = 0 \ , && (5.10a) \\ \nu_i(\boldsymbol{x}) &:= \nu_0 \sigma q^{i-1} \sin(2\pi x_1) \sin(2\pi x_2) \sin(2\pi x_3) && i = 1, ..., M \ . && (5.10b) \end{aligned}$$

We consider only the situation that the viscosity ν has a first order expansion. In addition, the first order polynomials of ψ_i are defined as:

$$\psi_i(\boldsymbol{\xi}) := \xi_i \ , \quad i = 1, ..., M \ . \tag{5.11}$$

Then, we exploit the orthogonality of the random basis polynomials, and Equation (5.8) becomes:

$$- \sum_{i=0}^{M} \sum_{j=0}^{P} \nabla \cdot (\nu_i(\boldsymbol{x}) \nabla u_j(\boldsymbol{x})) C_{ijk} = \delta_{0k} \ , \quad \forall k = 0, ..., P \ . \tag{5.12}$$

C_{ijk} is the Galerkin third-order tensor (Equation (3.51)). The stochastic weak discrete problem is given by:

Find $u_{h,i} \in V^h \subset H_0^1(\mathcal{D}), i = 0, ..., P$, such that,

$$\sum_{i=0}^{M} \sum_{j=0}^{P} \nu_i (\nabla u_{h,j}, \nabla v_h) C_{ijk} = (\delta_{0k}, v_h) \ , \quad \text{in } \mathcal{D} \ , \tag{5.13}$$

$\forall k = 0, ..., P$, $\forall v_h \in V^h$.

Hence, we are allowed to employ the stochastic finite element method to compute the stochastic solution of the considered system (Equation (5.4)) in order to study the scalability feature of the PCE Multilevel preconditioner. The calculation settings are stated in Table 5.2.

The computational domain is a unit cube $[0, 1]^3$ with two different refinement levels: 6 and 8. Therefore, the Poisson test is shown for two different ranges of the computing nodes according to the refinement levels (Table 5.3). It means that the refinement level 6 is subject to the computing node range 2 to 64 and the refinement level 8 is subject to the computing node range 128 to 2048. We restrict ourselves by taking only three random variables, because three random inputs are also considered in our blood pump model. Four different truncated polynomial orders, 2 to 5, are considered in this example, which implies that the dimension of the stochastic space also increases due to this alteration. Additionally, the random inputs are modeled only with the first order Chaos Polynomials (Equation (5.9)). Besides, we propose two different solving strategies regarding the PCE Multilevel preconditioner, namely the exact Multilevel preconditioner and the inexact Multilevel preconditioner. The main variation of these two strategies is that the inexact fashion merely requires a rough approximation. This is achieved by setting a large relative error

Stochastic discretization					
M	3	L	2, 3, 4, 5	$P+1$	10, 20, 35, 56
q	0.5	σ	0.2, 0.5, 0.8	ν	0.01
Global system					
Linear solver (FGMRES)					
Tol_{abs}	1.0×10^{-10}	Tol_{rel}	1.0×10^{-10}	$Iter_{max}$	1000
Preconditioner (PCE Multilevel)					
Tol_{abs}	1.0×10^{-10}	Tol_{rel}	1.0×10^{-10}, 1.0×10^{-1}	$Iter_{max}$	1000, 3
Mean block					
Linear solver (CG) (preconditioned with BoomerAMG)					
Tol_{abs}	1.0×10^{-10}	Tol_{rel}	1.0×10^{-10}, 1.0×10^{-1}	$Iter_{max}$	1000

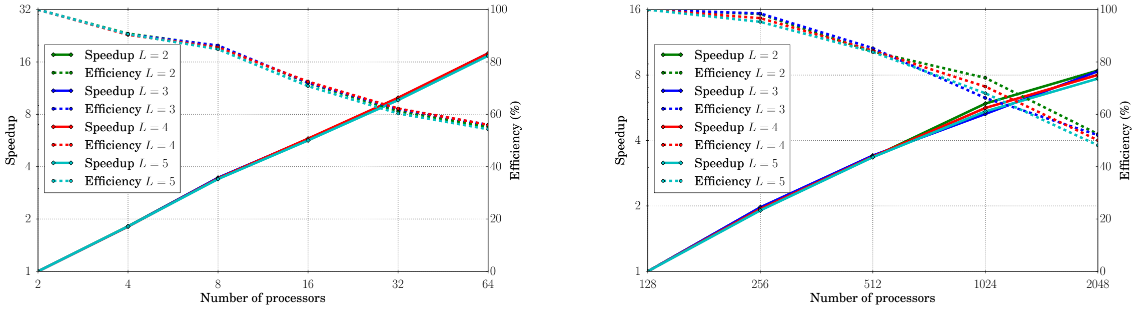
Table 5.2: Settings for stochastic Poisson test cases. The relative tolerance (Tol_{rel}) and the maximum number of iterations ($Iter_{max}$) have two values, they correspond to the exact strategy and the inexact strategy, respectively.

and a low iteration limit. The main idea is comparable to the Schur Complement preconditioner for the flow problem in Section 4.4. We also take the value of σ ($\sigma = 0.2, 0.5, 0.8$) into account, which in general influences the level of the diagonal dominance of the global system.

L	2	3	4	5
modes	10	20	35	56
The processor range: 2 **to** 64 **(refinement level** 6**)**				
DOFs	2.75 Millions	5.49 Millions	9.61 Millions	15.38 Millions
The processor range: 128 **to** 2048 **(refinement level** 8**)**				
DOFs	169.75 Millions	339.50 Millions	594.11 Millions	950.58 Millions

Table 5.3: Problem size (i.e. the number of DOFs).

5.2.1 Exact PCE Multilevel

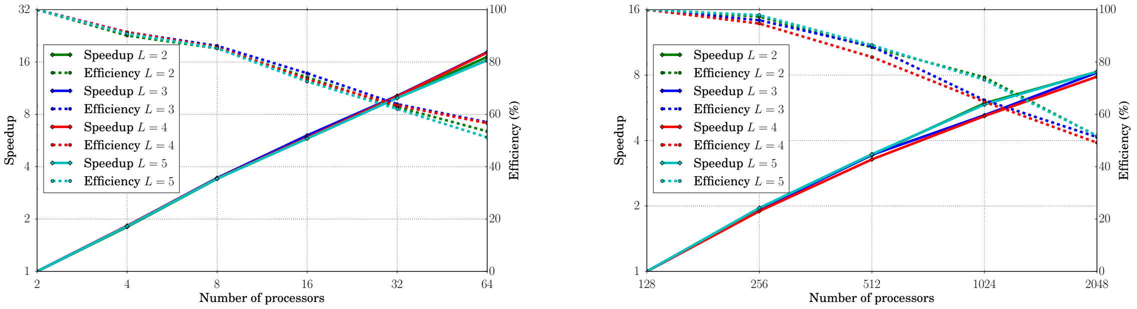


(a) Speedup and efficiency. The number of processors: 2 to 64.

(b) Speedup and efficiency. The number of processors: 128 to 2048.

Figure 5.5: The scalability test of the exact PCE Multilevel preconditioner, $\sigma = 0.2$.

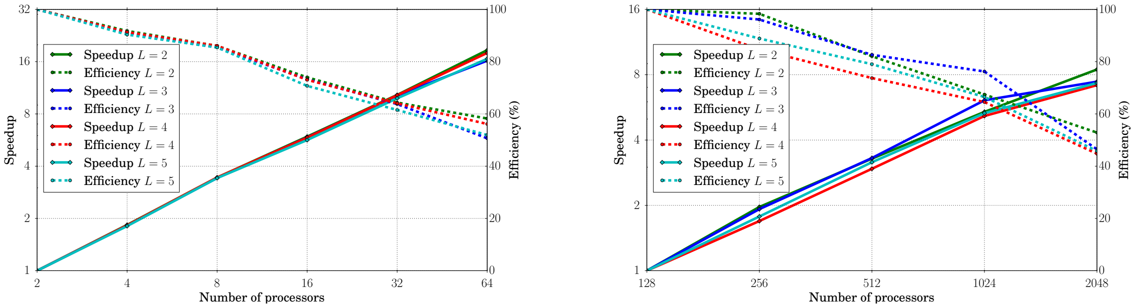
For the exact PCE Multilevel preconditioner, we set that the relative tolerance and the maximum number of iterations of the PCE Multilevel to be 1.0×10^{-10} and 1000, respectively.



(a) Speedup and efficiency. The number of processors: 2 to 64.

(b) Speedup and efficiency. The number of processors: 128 to 2048.

Figure 5.6: The scalability test of the exact PCE Multilevel preconditioner, $\sigma = 0.5$.



(a) Speedup and efficiency. The number of processors: 2 to 64.

(b) Speedup and efficiency. The number of processors: 128 to 2048.

Figure 5.7: The scalability test of the exact PCE Multilevel preconditioner, $\sigma = 0.8$.

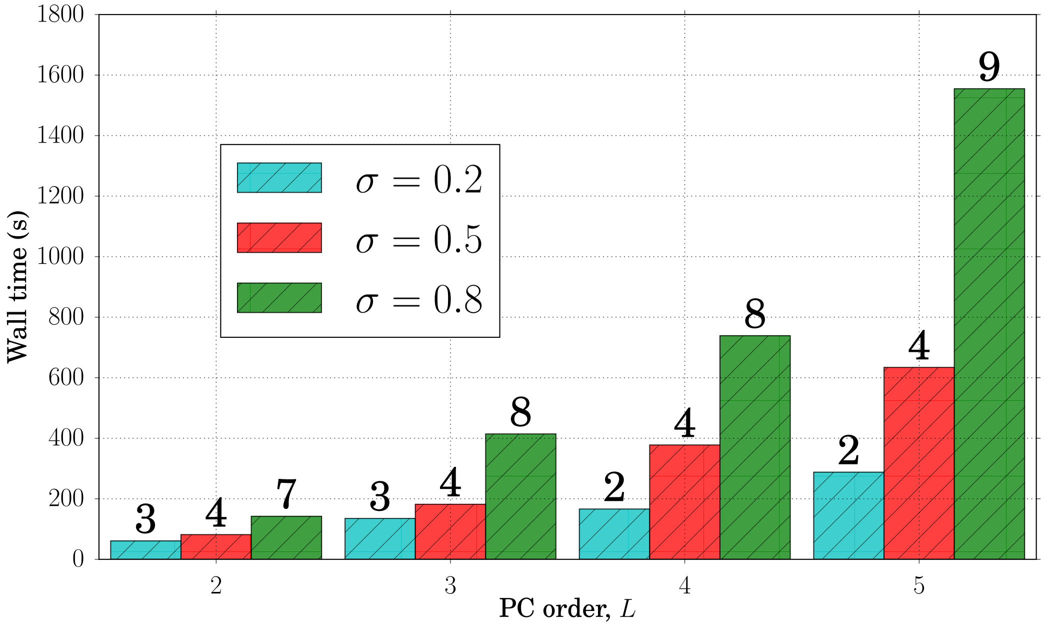


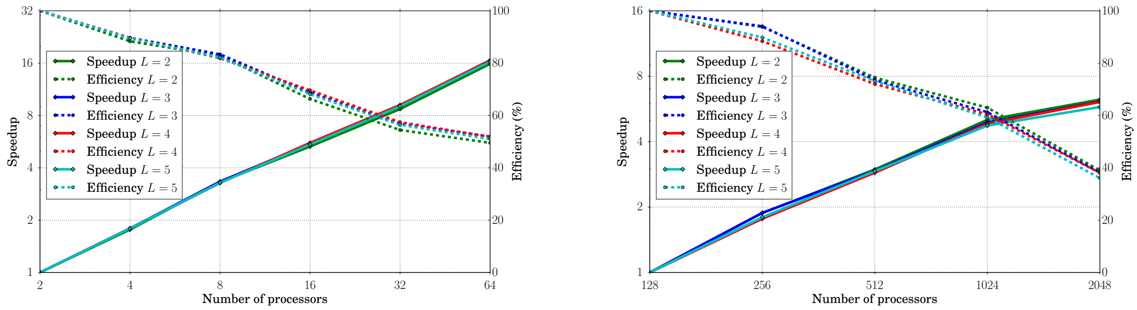
Figure 5.8: The computation time in second for the usage of 2048 processors. The numbers on top of the bars represent the number of FGMRES iterations (exact PCE Multilevel).

In other words, the PCE Multilevel solves the linear system relatively accurate. Three values of σ (0.2, 0.5 and 0.8) are also used for this comparison. When the value of σ increases, the global system matrix becomes less diagonal dominant. This effect can be seen in Equation (5.9) and Equation (5.13). It implies that the deviation of the input parameters in relation to the mean value becomes larger. The general consequence for the performance of the PCE Multilevel method is that it requires more iterations because the smoother for the Multilevel approach is the Mean-based preconditioner, which is actively influenced by the diagonal dominance of the global Galerkin matrix [150, 79].

In this scalability study (Figures 5.5 to 5.7), two ranges of the number of processors are considered, namely from 2 to 64 and 128 to 2048, also with different problem sizes (Table 5.3). We observe that the efficiency remains around 50% for both ranges of processors. The results of the range 2 to 64 are slightly below 50% and these of the range 128 to 2048 are marginally above 50%.

Figure 5.8 provides the information about the time consumption of the solving procedures and the number of FGMRES iterations by using the results obtained with 2048 processors. We observe that the solving time increases once the value of σ becomes larger, as well as the number of FGMRES iterations. This outcome meets the explanation which is related to the change in the structure of our global matrix. Nonetheless, the number of FGMRES iterations does not alter dramatically when we expand the stochastic solution into higher polynomials, it corresponds analogously to the behavior of the Multigrid method (i.e. it is independent from the mesh size). Besides, the number of FGMRES iterations stays in a relatively low level when the global matrix becomes less diagonally dominant, and each of FGMRES iterations requires only 1 or 2 Multilevel V-cycles. It means that the PCE Multilevel preconditioner is still effective with respect to this alteration.

5.2.2 Inexact PCE Multilevel



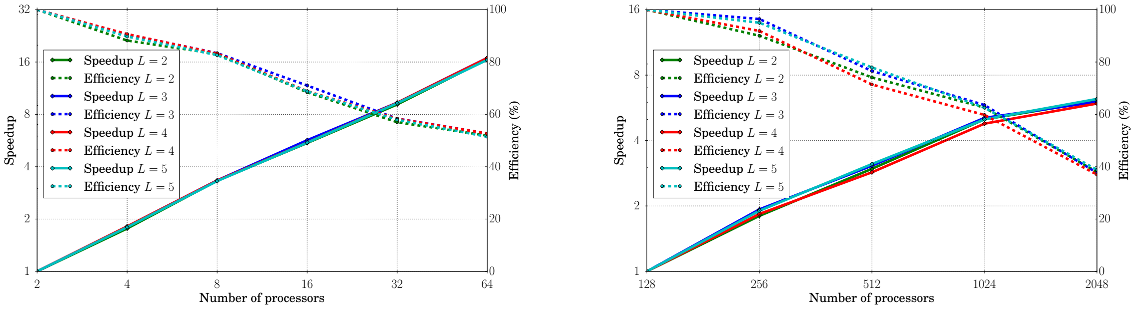
(a) Speedup and efficiency. The number of processors: 2 to 64.

(b) Speedup and efficiency. The number of processors: 128 to 2048.

Figure 5.9: Scalability test for the inexact PCE Multilevel preconditioner, $\sigma = 0.2$.

In Contrast to the exact PCE Multilevel solving strategy, the inexact Multilevel only necessitates an approximated solution from the computation. In this subsection, the inexact strategy is realized by adjusting the relative error and the maximum iteration of the PCE Multilevel preconditioner to 1.0×10^{-1} and 3, respectively.

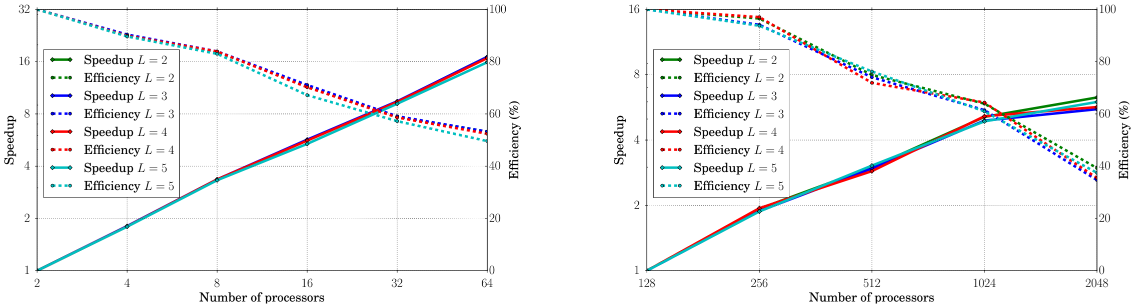
We consider the same problem sizes as in the previous case (Table 5.3) and obtain similar scalability results (Figures 5.9 to 5.11) as in Section 5.2.1. We observe that both preconditioning strategies show that they have over 50% efficiency up to 64 cores for the first range of processors (2 to 64). But, for the second range, the efficiency drops to 40% instead of 50% for the inexact Multilevel preconditioner. A significant decrease in the performance happens after 1024 processors. Similar to Figure 5.8, Figure 5.12 also shows the computational time and the number of



(a) Speedup and efficiency. The number of processors: 2 to 64.

(b) Speedup and efficiency. The number of processors: 128 to 2048.

Figure 5.10: Scalability test for the inexact PCE Multilevel preconditioner, $\sigma = 0.5$.



(a) Speedup and efficiency. The number of processors: 2 to 64.

(b) Speedup and efficiency. The number of processors: 128 to 2048.

Figure 5.11: Scalability test for the inexact PCE Multilevel preconditioner, $\sigma = 0.8$.

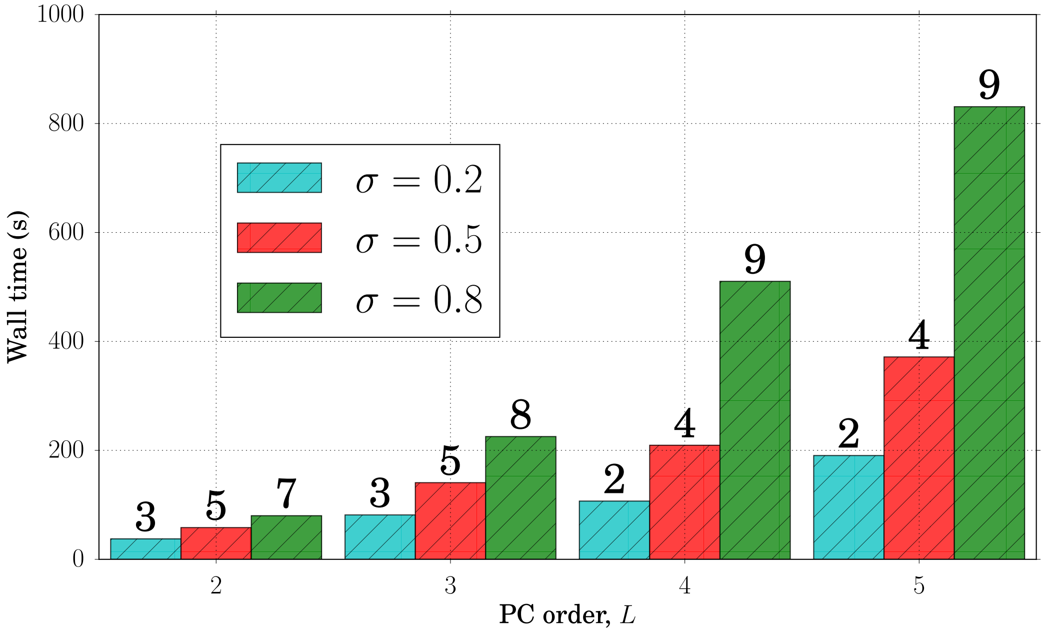


Figure 5.12: The computation time in second for the usage of 2048 processors. The numbers on the top of the bars represent the number of FGMRES iterations (inexact PCE Multilevel).

FGMRES iterations with respect to different PC orders. However, we restrict ourselves to 2048 computing cores for the final simulation based on the spatial discretization and the available computing resources.

5.2.3 Exact vs. Inexact

Although the PCE Multilevel preconditioner scales worse for the inexact solving strategy than for the exact case, it is still beneficial to consider the inexact version with respect to a reduction of computing time. For that reason, in this subsection we provide a brief comparison of these two solving strategies.

Figure 5.13 shows a comparison of the computing time and the amount of outer solver iterations of both solving strategies based on the same testing problem which is presented in Figures 5.8 and 5.12. The inexact Multilevel preconditioner takes less computing time comparing with the exact Multilevel preconditioner for each examined problem (Figure 5.13a). This superiority is even greater once the problem size becomes larger. In the meantime, the amount of FGMRES iterations does not alter. Only in few cases the number of iterations increases by 1. This outcome supports the same conceptional arguments in Section 5.1 that the preconditioners only need to provide a solution with a reasonable accuracy, such that the Krylov subspace solvers can generate a "good search direction" for the next iteration. We investigate also the amount of smoother iterations (i.e. the mean block solver) in the PCE Multilevel preconditioner. The inexact strategy requires only 50% of the iterations of the exact strategy (Figure 5.14). In addition, the selected simulation results are shown in Figure 5.15.

Remark 5.2.1. *Our implemented linear algebra structure for the Polynomial Chaos expansion is designed such that each computing processor holds all stochastic mode solution vectors (i.e. there is no decomposition with respect to the stochastic space). It means that the dimension of the stochastic modes can not be very large. This concept is based with regard to the efficiency of the PCE Multilevel method, because the restriction procedure and the prolongation procedure can be done efficiently within a single processor, otherwise, the communication between the processors can be expensive. However, as explained in Chapter 3, we restrict ourselves to a low dimensional stochastic space, therefore, we are not dealing with a large number of stochastic modes. Moreover, the Mean-based preconditioner can be more flexible if the stochastic mode solutions are also parallelized. However, we do not provide further demonstrations about the parallelization of the Mean-based preconditioner in this work.*

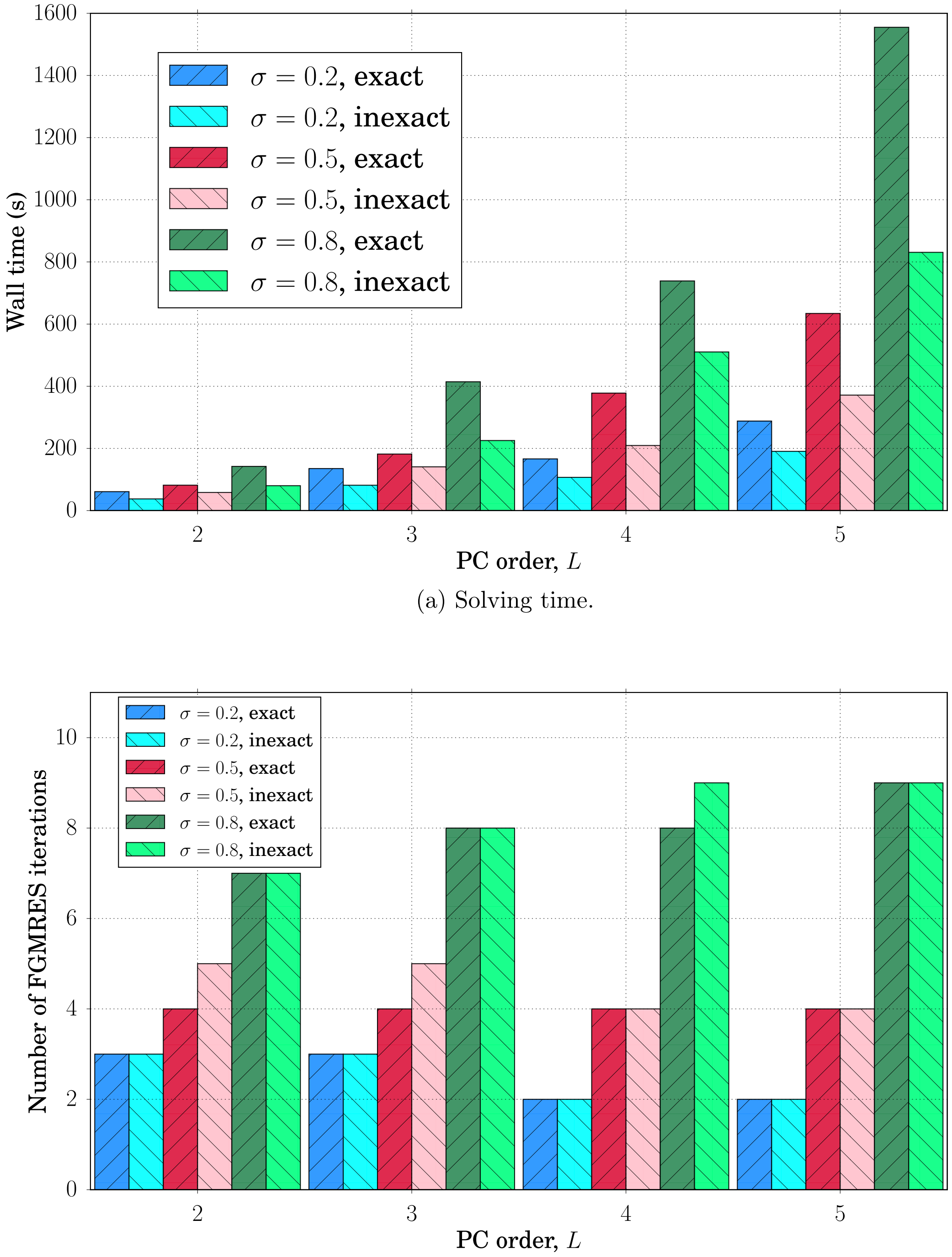
5.3 A Modification of the PCE Multilevel Preconditioner

As described and motivated in Chapter 4 and Section 5.2, we utilize the PCE Multilevel method to precondition our global linear system. Although the number of iterations of the PCE Multilevel solver is generally independent of the stochastic levels, the computing effort of one preconditioning cycle (V- or W-cycle) is effected by the number of levels and the random inputs. For example, for a V-cycle, the number of mean block computations is given by:

$$N := 2\sum_{i=0}^{M} \frac{(i+L)!}{i!L!} - 1 \; . \tag{5.14}$$

M is the number of random parameters, L is the truncated Chaos Polynomial degree.

Figure 5.16 shows that the amount of the mean block computations increases faster than the total number of the stochastic modes. This quantity can be considered analogously to the computational effort of the preconditioning process, because we only employ the Mean-based preconditioner as the smoother. Consequently, the number of mean block computations in the



(a) Solving time.

(b) FGMRES iterations.

Figure 5.13: Comparison of the computing time and the number of FGMRES iterations for the Poisson problem at 2048 processors.

preconditioning process grows exponentially expensive with respect to the dimension of the approximated stochastic space.

Thus, we try to reduce the amount of the mean block computations within the preconditioning process. One of our proposals is that we firstly keep the smoothing process only on level 0, 1 and L, then we select another level l, which is located between 1 and L, as a transition level. On the level l, the Mean-based smoother is also applied.

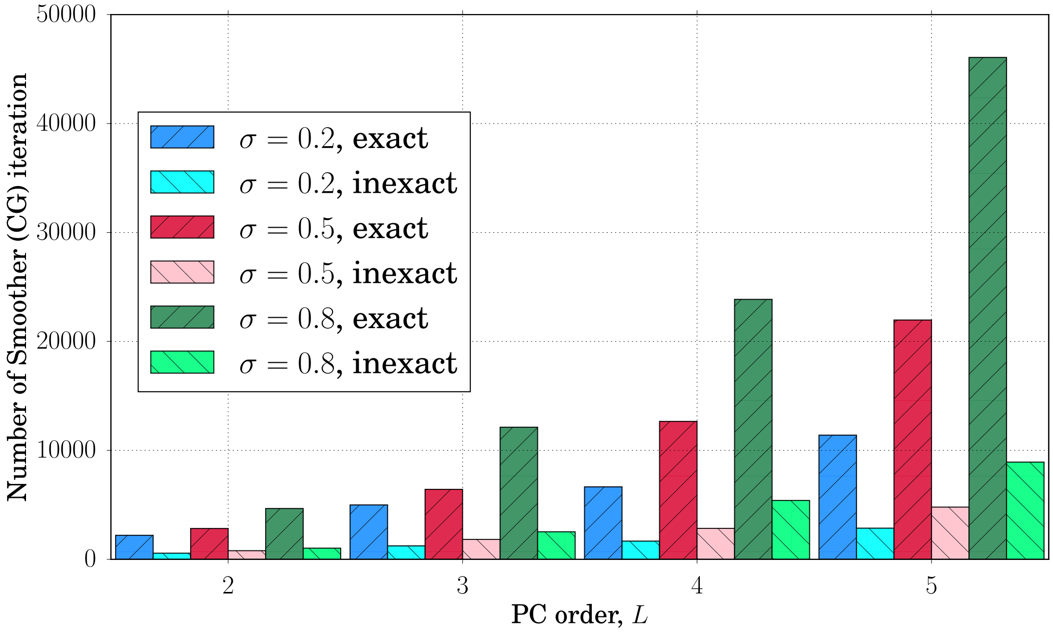
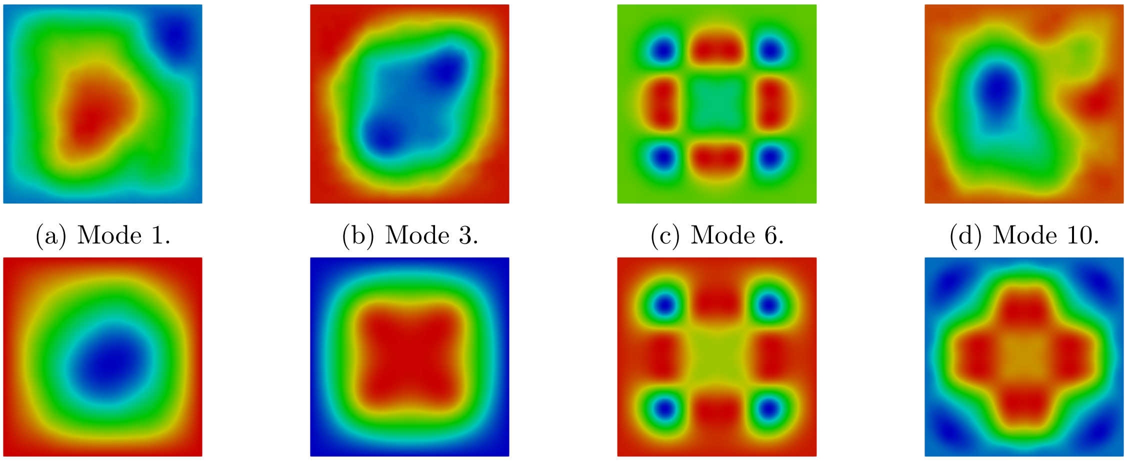


Figure 5.14: Comparison of the number of smoother (CG) iterations.



(a) Mode 1.

(b) Mode 3.

(c) Mode 6.

(d) Mode 10.

(e) Mode 15.

(f) Mode 20.

(g) Mode 22.

(h) Mode 33.

Figure 5.15: The numerical results on a cross-section with regard to selected stochastic modes. The cross-section is located in the middle of the unit cube along the direction of the z-axis.

Figure 5.17 demonstrates the modified version of the PCE Multilevel preconditioner. Each black dot represents a stochastic level. The red circles indicate the selected levels, which are utilized by the smoothing process. The restriction operator $\mathcal{R}$ and the prolongation operator $\mathcal{P}$ are only performed between these selected levels. The rest of the levels, which are represented by the black dots without the red circle, are simply omitted in the restriction and prolongation steps.

Hence, we have tested two configurations based on the 2D channel benchmark problem [147]. The first configuration is the channel flow with a circular shaped obstacle. The inflow boundary condition and the viscosity are modeled as the uncertain sources. The second configuration is a flow driven by a rotating object. The computing domain is very similar to the previous configuration, and only the circular obstacle is replaced by a rotating rotor (Figure 5.19). In addition, the rotor's velocity is also modeled as an uncertain parameter. Instead of applying the

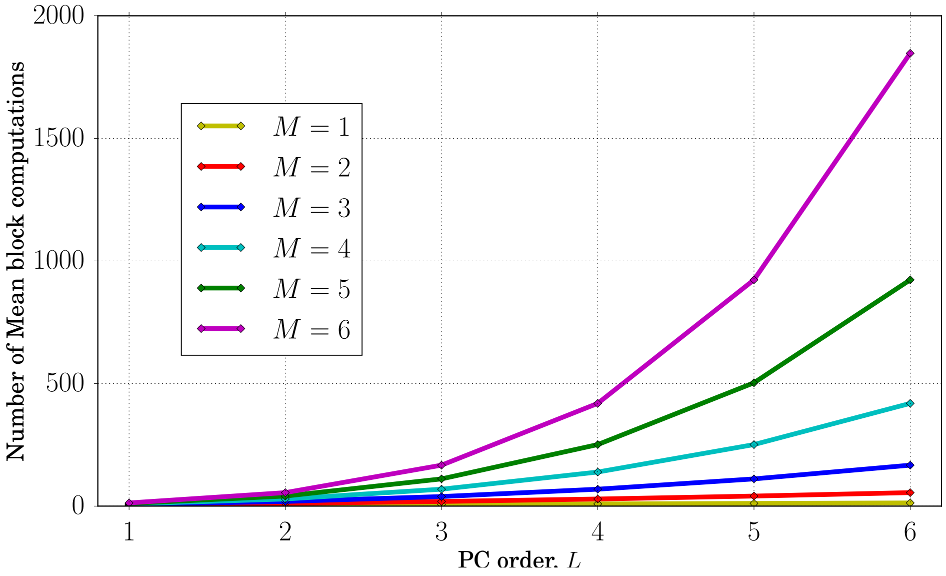


Figure 5.16: Number of mean block computations.

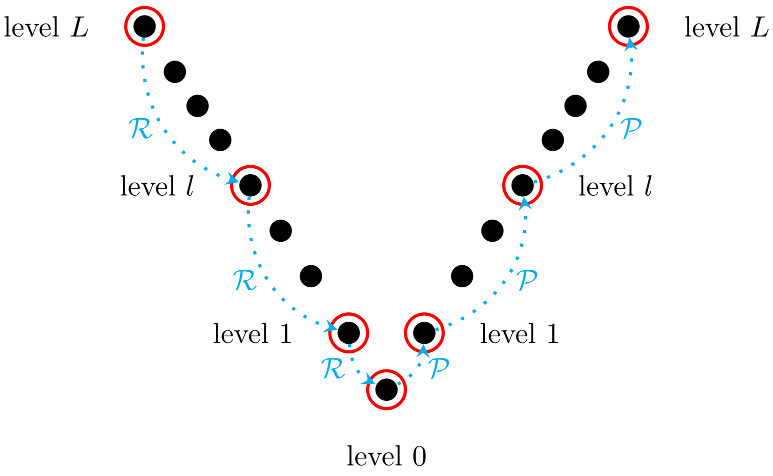


Figure 5.17: An illustration of the modified version of the PCE Multilevel preconditioner by means of a V-cycle.

incompressible NSE as in [147], we use the VMS formulation as defined in Chapter 3 in order to better adapt to our blood pump simulation. These two testing cases provide very promising results in regard to the performance. Here, we only give the numerical results of the second case.

Figure 5.18 presents a comparison of the computational time between the full levels PCE Multilevel preconditioner (algorithm 2) and the selected levels PCE Multilevel preconditioner (Figure 5.17). The red line represents the number of stochastic modes regarding the polynomial order L, the blue line is the total number of mean block computations needed for a V-cycle (Equation (5.14)). One can observe that the computational time for the full levels preconditioner follows proportionally the number of the stochastic modes per a V-cycle (blue line), because the amount of the V-cycles needed per time step remains basically the same. Furthermore, the computational effort is reduced by employing the modified version of the Multilevel preconditioner, the time increases slowly and follows the red line. Note that, when $L = 1, 2$ and 3, the time consumed by the modified version follows also the blue line, because we intend to always keep the level 1 and level L. Besides, we provide a visualization of the numerical solution of the second configuration (Figure 5.19). The middle figure shows the mean value of the velocity

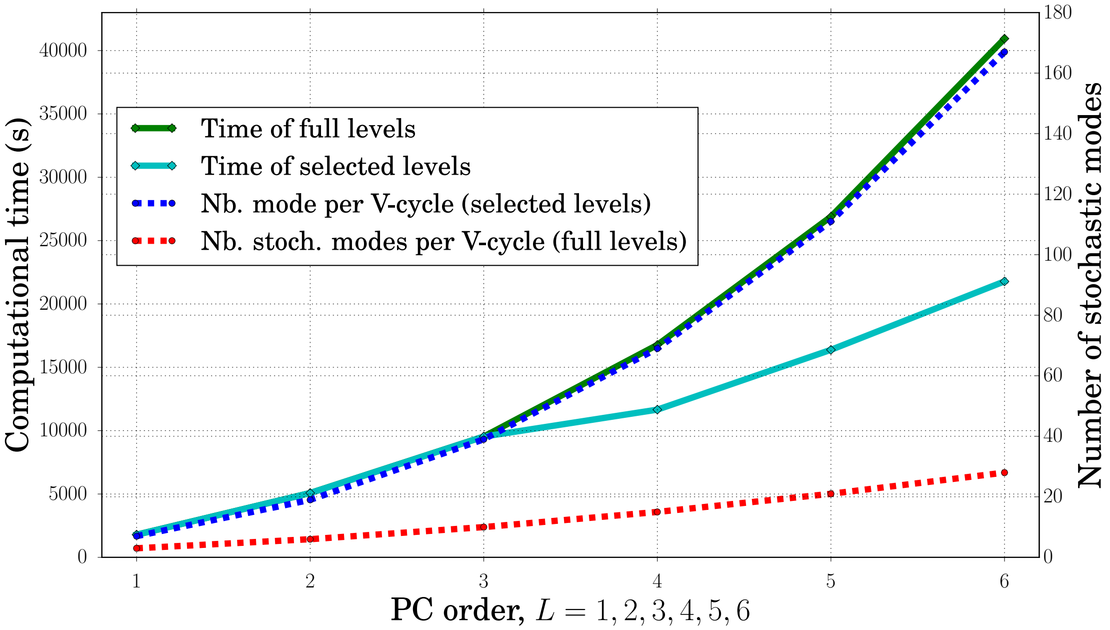


Figure 5.18: Solving time (the sum of first 200 time steps) vs. the number of stochastic modes. The computation is based on the 2D channel flow with a rotor.

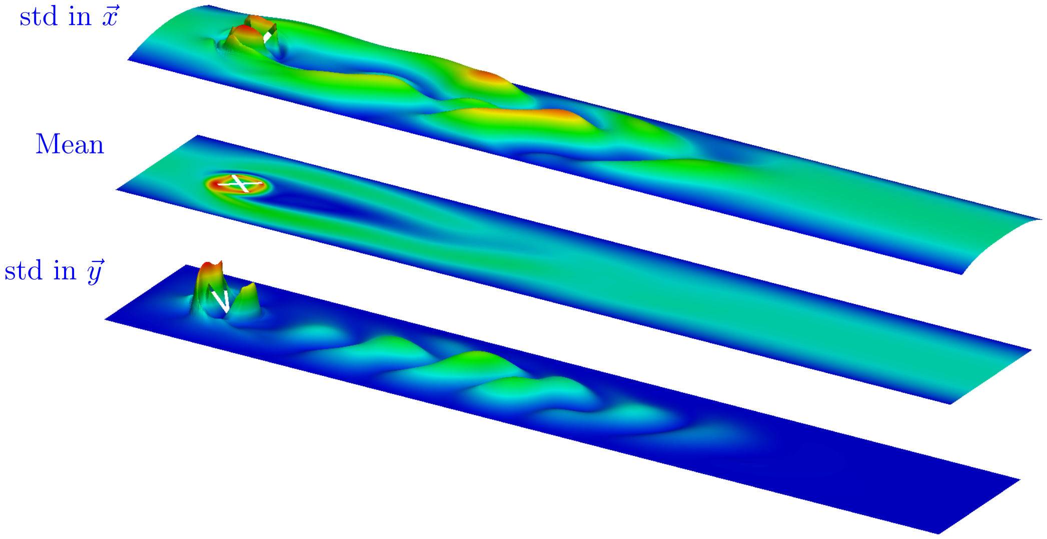


Figure 5.19: The numerical solution at the time step 100.

field at the time step 100, where the velocity around the rotor has higher magnitude due to the rotation. The upper one is colored by the standard deviation of the velocity along the direction of the $\boldsymbol{x}$-axis, the resulting shape is warped proportionally by the value of the standard deviation. Correspondingly, the lower case indicates the standard deviation along the direction of the $\boldsymbol{y}$-axis.

Chapter 6

Numerical Experiment

After presenting the scalability of the solvers and the preconditioners, we can summarize the important components, which are necessary for the blood pump simulation: The generalized Polynomial Chaos expansion method used for quantifying parameterized uncertainties (namely the inflow boundary condition, the blood viscosity and the revolving speed of rotor). The shear layer update approach used for realizing the moving mesh concept. The Variational Multiscale method used for modeling the turbulent flow. The Polynomial Chaos expansion Multilevel technique used for preconditioning the global stochastic linear system. The Schur Complement used for preconditioning the mean block.

This chapter is dedicated to the numerical experiments of the blood pump simulation in consideration of assessing the uncertainties. The chapter consists of two sections, the first part is devoted to a stationary case in a simplified pump geometry, this example is regarded as an intermediate step before proceeding the unsteady state calculation. We enhance again the importance of applying the inexact Multilevel preconditioner by using this steady flow simulation. The second part contributes to the model calibration, in which the numerical results of the stochastic model of the FDA's blood pump are illustrated. All the components, which are studied in the previous chapters, are brought into play.

6.1 Stationary Case

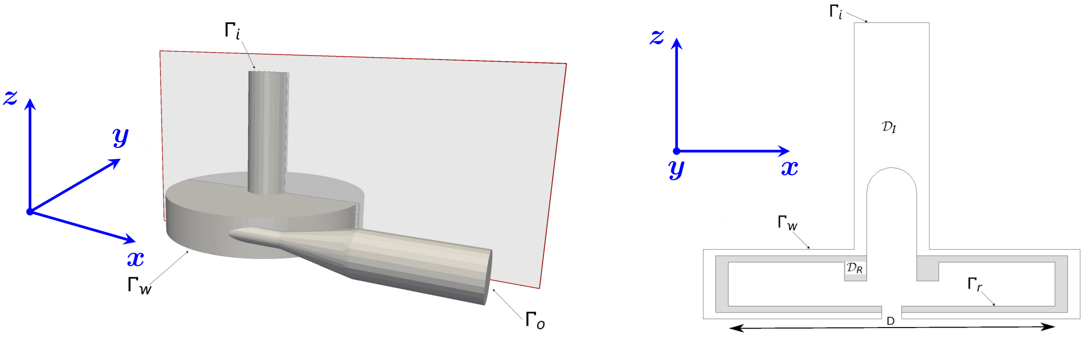
Performing a moving mesh computation for a high rotating speed system is always a tedious and expensive process. However, for engineering perspectives, a stationary result can be very serviceable to provide an overview about the flow behavior in the average sense. Moreover, the results obtained from the steady flow simulation can also be served as the initial guess in the unsteady simulation in order to accelerate the convergence of the first time step iteration.

The Multiple Reference Frame (MRF) method is designed for simulating an axisymmetric rotating structure [35, 103, 44, 106] in a steady state. This approach proposes to divide the computational domain into two adjacent areas, i.e. the stationary zone and the moving reference frame zone (Section 2.3). In our steady simulation, the stationary zone is governed by the steady incompressible Navier-Stokes equations. In the moving reference frame zone, the Coriolis force and the centrifugal force have to be included into the momentum equation.

Figure 6.1 shows the geometry of the computing domain $\mathcal{D}$, where $\mathcal{D}_R$ is the moving reference frame zone, $\mathcal{D}_I$ is the stationary frame zone, $\mathcal{D} = \mathcal{D}_R \cup \mathcal{D}_I$, $\mathcal{D}_R \cap \mathcal{D}_I = \emptyset$. Γ_i indicates the inflow boundary, Γ_o is the outflow boundary, Γ_r is the surface of the rotor, Γ_w is the boundary of the rigid wall, $\partial\mathcal{D} = \Gamma_i \cup \Gamma_o \cup \Gamma_r \cup \Gamma_w$. D is the diameter of the rotor.

Thence, the governing equations are given by:

$$\boldsymbol{u} \cdot \nabla \boldsymbol{u} - \frac{\mu}{\rho}\Delta \boldsymbol{u} + \frac{1}{\rho}\nabla p = \boldsymbol{0} \, , \qquad \text{in } \mathcal{D}_I \, , \tag{6.1a}$$



(a) 3D geometry with cutting plane for (a).

(b) 2D cross-section of the pump geometry.

Figure 6.1: The demonstration of the blood pump, the stationary zone $\mathcal{D}_I$ and the moving reference zone $\mathcal{D}_R$. (a) is the 3D geometry the blood pump. The 2D plan in (a) indicates the location of the cross-section. The moving reference frame zone $\mathcal{D}_R$ of the MRF method is highlighted in (b).

$$\boldsymbol{u}_R \cdot \nabla \boldsymbol{u}_R + 2\boldsymbol{\omega} \times \boldsymbol{u}_R + \boldsymbol{\omega} \times \boldsymbol{\omega} \times \boldsymbol{d} \tag{6.1b}$$

$$-\frac{\mu}{\rho}\Delta \boldsymbol{u}_R + \frac{1}{\rho}\nabla p = \boldsymbol{0} \,, \qquad \text{in } \mathcal{D}_R \,, \tag{6.1c}$$

$$\nabla \cdot \boldsymbol{u} = 0 \,, \qquad \text{in } \mathcal{D}_I \,, \tag{6.1d}$$

$$\nabla \cdot \boldsymbol{u}_R = 0 \,, \qquad \text{in } \mathcal{D}_R \,, \tag{6.1e}$$

$$\boldsymbol{u} = \boldsymbol{g}(\boldsymbol{x}) \,, \qquad \text{on } \Gamma_i \,, \tag{6.1f}$$

$$\boldsymbol{u} = \boldsymbol{0} \,, \qquad \text{on } \Gamma_w \,, \tag{6.1g}$$

$$\boldsymbol{u}_R = \boldsymbol{h}(\boldsymbol{x}) \,, \qquad \text{on } \Gamma_r \,, \tag{6.1h}$$

$$(-\mathbb{1}p + \frac{\mu}{\rho}\nabla \boldsymbol{u}) = \boldsymbol{0} \,, \qquad \text{on } \Gamma_o \,. \tag{6.1i}$$

Here, $\boldsymbol{u}$ is the velocity, and p is the pressure. μ is the dynamic viscosity, and ρ is the density. $\boldsymbol{\omega}$ is the angular speed and $\boldsymbol{g}(\boldsymbol{x})$ is the inflow boundary condition. $\boldsymbol{u}_R$ is the velocity on the rotating frame ($\mathcal{D}_R$), and $\boldsymbol{u}_R$ is defined by:

$$\boldsymbol{u}_R := \boldsymbol{u} - \boldsymbol{\omega} \times \boldsymbol{d} \,, \tag{6.2}$$

$\boldsymbol{u}_R$ is acting on $\mathcal{D}_R$ instead of $\boldsymbol{u}$ in order to better represent the rotating effect in the steady state. The angular speed is only nonzero along the direction of the $\boldsymbol{z}$-axis (i.e. the rotating axis) in our configuration (Figure 6.1), it means:

$$\boldsymbol{\omega} := \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} \,. \tag{6.3}$$

Note that, there are no additional boundary conditions needed between $\mathcal{D}_I$ and $\mathcal{D}_R$. The inflow boundary condition $\boldsymbol{g}(\boldsymbol{x})$ is modeled with a Poiseuille profile, whose center-line is aligned with the $\boldsymbol{z}$-axis. It is given as:

$$\boldsymbol{g}(\boldsymbol{x}) := \boldsymbol{g}(x, y, z) = \begin{bmatrix} 0 \\ 0 \\ -U_{max}(1 - (x^2 + y^2)/R^2) \end{bmatrix} \,. \tag{6.4}$$

Here, $\sqrt{x^2 + y^2} < R$, R indicates the radius of the inlet cylinder, which conducts the blood flow into the chamber. $U_{max} > 0$ is the maximum inflow velocity. Consequently, two additional terms are brought in:

$$2\boldsymbol{\omega} \times \boldsymbol{u}_R : \text{Coriolis force.}$$
$$\boldsymbol{\omega} \times \boldsymbol{\omega} \times \boldsymbol{d} : \text{centrifugal force.}$$

The Coriolis force represents the deflecting effect (inertial force) on the object's motion in a rotating reference frame. The centrifugal force is also an inertial force, which is pointed outward the rotating axis. Note that, there are normally three additional fictitious forces introduced by the rotating reference frame. Except these two forces, there is also the Euler force. However, the Euler force only appears when the system is in an unsteady rotating reference frame. Therefore, the Euler force does not appear in our case (Equation (6.1)). Moreover, $\boldsymbol{d}$ is the distance from an arbitrary coordinate on the rotating frame to the revolving axis (Figure 6.1).

Hence, with the aid of Equation (6.2), $\boldsymbol{u}$ or $\boldsymbol{u}_R$ can be eliminated from Equation (6.1). Therefore, no additional quantities are involved in regard to the incompressible NSE. The Dirichlet boundary condition on the blade (Figure 6.1b) is given as:

$$\boldsymbol{h}(\boldsymbol{x}) := \boldsymbol{h}(x, y, z) = \boldsymbol{\omega} \times \boldsymbol{d} = \begin{bmatrix} -x\omega \\ y\omega \\ 0 \end{bmatrix} . \tag{6.6}$$

We consider then three different random inputs: the inflow boundary condition $\boldsymbol{g}(\boldsymbol{x})$, the rotor's speed $\boldsymbol{\omega}$ and the dynamic viscosity μ. These three parametric uncertainties are modeled with the independent Uniform distributed random variables $\xi_i \sim U(-1, 1), i = 1, 2, 3$. The explicit forms are given by:

$$\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{g}_0(\boldsymbol{x}) + \sigma_1 \boldsymbol{g}_0(\boldsymbol{x})\xi_1 \ , \tag{6.7a}$$
$$\mu = \mu_0 + \sigma_2 \mu_0 \xi_2 \ . \tag{6.7b}$$
$$\boldsymbol{\omega} = \boldsymbol{\omega}_0 + \sigma_3 \boldsymbol{\omega}_0 \xi_3 \ , \tag{6.7c}$$

where, $\boldsymbol{g}_0(\boldsymbol{x})$, $\boldsymbol{\omega}_0$ and μ_0 are the mean values regarding the three random inputs. $\sigma_i, i = 1, 2, 3$ are the variation factors with respect to the mean values, σ_i satisfy the condition $0 < \sigma_i < 1$ in order to ensure the positivity. Note that, all three random parameters are essentially scalar values, therefore, the random variables ξ_i can be seemed to be assigned straightly to U_{max}, ω_0 and μ_0, respectively. Subsequently, we collect the three random variables into one random vector $\boldsymbol{\xi} := [\xi_1, \xi_2, \xi_3]$, it enables us to map a probability space $(\Theta, \mathcal{F}, \mathbb{P})$ with the sample space Θ, sigma-algebra $\mathcal{F} \subseteq 2^\Theta$ and the probability measure $\mathbb{P}$ to the subset $\Xi \subset \mathbb{R}^3$ (Chapter 3). Thence, all stochastic quantities can be expressed in terms of $\boldsymbol{\xi}$. We use the Polynomial Chaos expansion technique to represent the solutions of the velocity and the pressure, they are given as:

$$\boldsymbol{u}(\boldsymbol{x}) = \sum_{i=0}^{\infty} \boldsymbol{u}_i \psi_i(\boldsymbol{\xi}) \ , \quad p(\boldsymbol{x}) = \sum_{i=0}^{\infty} p_i \psi_i(\boldsymbol{\xi}) \ . \tag{6.8}$$

As mentioned above, the three random variables are modeled with the Uniform distribution, i.e. $\psi_i(\boldsymbol{\xi})$ correspond to the normalized multivariate Legendre polynomials [176]. One of the important features of the Chaos Polynomials is the orthogonality with respect to the probability density function, it means:

$$\int_{[-1,1]^3} \psi_i(\boldsymbol{\xi}) \psi_j(\boldsymbol{\xi}) \frac{1}{2^3} d\boldsymbol{\xi} = \delta_{ij} \ , \tag{6.9}$$

δ_{ij} is the Kronecker operator. However, Equation (6.8) has to be truncated up to a certain number P (Equation (3.30)) in order to cope with stochastic-spectral finite element method. The truncated expressions read:

$$\boldsymbol{u}(\boldsymbol{x}) = \sum_{i=0}^{P} \boldsymbol{u}_i \psi_i(\boldsymbol{\xi})\,, \quad p(\boldsymbol{x}) = \sum_{i=0}^{P} p_i \psi_i(\boldsymbol{\xi})\,. \tag{6.10}$$

We insert Equation (6.10) and Equation (6.7) into Equation (6.1). By considering the Galerkin projection technique, we obtain:

$$\sum_{i=0}^{P}\sum_{j=0}^{P} \boldsymbol{u}_i \cdot \nabla \boldsymbol{u}_j \, C_{ijk} - \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \Delta \boldsymbol{u}_j \, C_{ijk} + \frac{1}{\rho} \nabla p_k = \boldsymbol{0}\,, \qquad \text{in } \mathcal{D}_I\,, \tag{6.11a}$$

$$\begin{aligned} &\sum_{i=0}^{P}\sum_{j=0}^{P} \boldsymbol{u}_{R,i} \cdot \nabla \boldsymbol{u}_{R,j} \, C_{ijk} + \sum_{i=0}^{P}\sum_{j=0}^{P} 2\boldsymbol{\omega}_i \times \boldsymbol{u}_{R,j} \, C_{jik} \\ &+ \sum_{i=0}^{P}\sum_{j=0}^{P} \boldsymbol{\omega}_i \times \boldsymbol{\omega}_j \times \boldsymbol{r} \, C_{ijk} - \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \Delta \boldsymbol{u}_{R,j} \, C_{ijk} + \nabla p_k = \boldsymbol{0}\,, \qquad \text{in } \mathcal{D}_R\,, \end{aligned} \tag{6.11b}$$

$$\nabla \cdot \boldsymbol{u}_k = 0\,, \qquad \text{in } \mathcal{D}_I\,, \tag{6.11c}$$

$$\nabla \cdot \boldsymbol{u}_{R,k} = 0\,, \qquad \text{in } \mathcal{D}_R\,, \tag{6.11d}$$

$$\boldsymbol{u}_k = \boldsymbol{g}_k(\boldsymbol{x})\,, \qquad \text{on } \Gamma_i\,, \tag{6.11e}$$

$$\boldsymbol{u}_k = \boldsymbol{0}\,, \qquad \text{on } \Gamma_w\,, \tag{6.11f}$$

$$\boldsymbol{u}_{R,k} = \boldsymbol{h}_k(\boldsymbol{x})\,, \qquad \text{on } \Gamma_r\,, \tag{6.11g}$$

$$(-\mathbb{1} p_k + \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho} \nabla \boldsymbol{u}_j \, C_{ijk}) = \boldsymbol{0}\,, \qquad \text{on } \Gamma_o\,. \tag{6.11h}$$

Inflow maximal speed (m/s)	0.55	Inflow speed variation (σ_1)	10%
Dynamic viscosity ($\mathrm{N \cdot s/m^2}$)	0.0035	Angular speed variation (σ_2)	10%
Angular speed (rad/s)	261.8	Viscosity variation (σ_3)	10%
RPM (r/min)	2500	Density ($\mathrm{kg/m^3}$)	1.035
Radius R (m)	0.006	Diameter D (m)	0.06

Table 6.1: Model parameters.

The focus of this stationary flow computation is to investigate the flow behavior in the pump geometry and compare the different solution strategies of our problem setting [150]. The outcome verifies once again the decision in Chapter 5, it means that the inexact Multilevel approach outperforms than the other methods. Table 6.1 indicates the physical parameters of this numerical simulation, we choose synthetically σ_i to be 10%, namely the uncertainties within the input data have a 10% deviation from their mean value (Equation (6.7)). Note that, we reduce the density artificially by a factor of 1000 for the sake of simplicity.

Accordingly, the spatial part of Equation (6.11) is discretized by using the finite element method with the Lagrangian Taylor-Hood elements $P2 - P1$ [162]. In regard to the nonlinear equation, the inexact Newton method is applied on the global nonlinear system with a time stepping strategy "choice 1" of Eisenstat and Walker [150, 50]. For solving the linear system arising from the Newton iteration, we analyze three solution strategies as follows:

- GMRES + Mean-based preconditioner.

- Exact PCE Multilevel solver.
- Inexact PCE Multilevel solver.

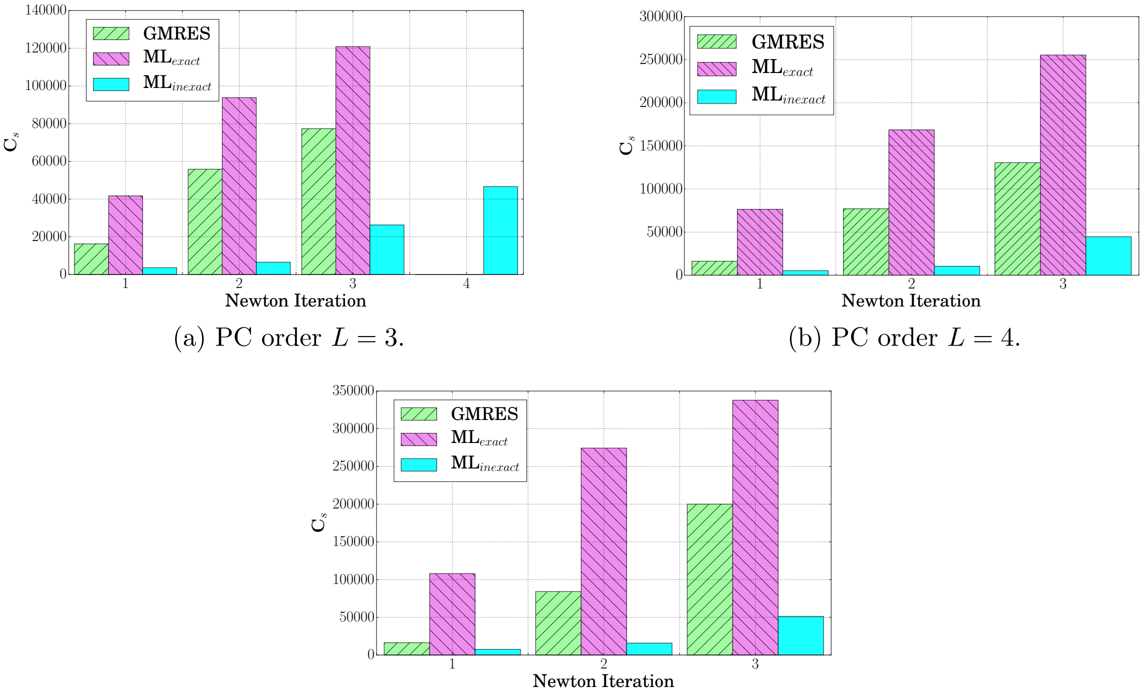
Note that, we employ here the PCE Multilevel scheme as a global solver instead of a preconditioner comparing with the test examples in Chapter 5. Both solvers, the GMRES and the PCE Multilevel, require multiple iterations with respect to the mean operator $\bar{A}_0$, because two solvers exploit the Mean-based solver/preconditioner on different levels. We define therefore an "inner" solver for the mean block system by utilizing the GMRES solver together with the incomplete LU factorization preconditioner (ILU). The inexact Multilevel solver is configured in a simultaneous manner as in Section 5.2, i.e., the relative tolerance of the mean block solver is assigned to be 10^{-1}. For the exact Multilevel solving strategy, the relative tolerance is adjusted to 10^{-12}. Furthermore, the Newton scheme restricts the absolute and relative tolerance to be under 10^{-9}.

As defined in [150] and Equation (2.1), the configuration above defines a Reynolds number around 200, the fluid should behave as a laminar flow. We compare three different PC orders ($L = 3, 4, 5$) of the Chaos Polynomials, which result in 20, 35 and 56 PC stochastic modes for the stochastic solution variables accordingly. The blood pump geometry (Figure 6.1) is triangulated by $192,451$ tetrahedral elements, it arises $919,334$ DOFs per PC mode.

		Newton Iter. ($L = 3$)				Newton Iter. ($L = 4$)			Newton Iter. ($L = 5$)		
		1	2	3	4	1	2	3	1	2	3
GMRES	N_{iter}	1	2	3	-	1	2	3	1	2	3
	N_{res}	2	3	4	-	2	3	4	2	3	4
	$N_{\bar{A}_0}$	23	60	80	-	23	102	140	23	136	224
	$\bar{N}_{iter,\bar{A}_0}$	706	931	967	-	706	755	933	706	619	893
ML_{exact}	N_{iter}	1	1	1	-	1	1	2	1	1	1
	N_{res}	11	11	11	-	14	14	27	17	17	17
	$N_{\bar{A}_0}$	55	68	67	-	107	139	278	180	251	251
	$\bar{N}_{iter,\bar{A}_0}$	758	1380	1804	-	715	1307	919	600	1093	1346
$\text{ML}_{\text{inexact}}$	N_{iter}	1	1	3	2	1	1	3	1	1	1
	N_{res}	11	11	31	21	14	14	40	17	17	33
	$N_{\bar{A}_0}$	57	69	207	138	108	139	417	183	251	502
	$\bar{N}_{iter,\bar{A}_0}$	63	95	137	338	49	75	107	41	63	102

Table 6.2: Comparison of the number of iterations for different solving procedures.

Table 6.2 provides a comparison of the number of iterations between the different PC degrees, also between the solving strategies. N_{iter} represents the number of the GMRES iterations or the V-cycles in the Multilevel solver. N_{res} denotes the number of residual computations. $N_{\bar{A}_0}$ is the number of mean block computations, it involves the matrix $\bar{A}_0$. $\bar{N}_{iter,\bar{A}_0}$ denotes the averaged number of the GMRES iterations in order to compute the solution from a linear system which is associated to $\bar{A}_0$, the average number means that it is chosen at every Newton iteration over all mean block solving parts. We notice from Table 6.2 that the amount of the Newton iterations needed is almost the same for these three solving strategies, the exact Multilevel solver (ML_{exact}) overcomes the other two methods in terms of the convergence behavior within the Newton iteration. However, the exact Multilevel solver requires more iterations in average for obtaining the solution of the mean block systems. The amount of the Newton iterations increases slightly for the inexact Multilevel approach ($\text{ML}_{\text{inexact}}$) because the relative error is less strict comparing to the exact Multilevel solver. Thus, additional Newton iterations are demanded. Yet, the inexact Multilevel approach necessitates much less mean block computations, the amount of

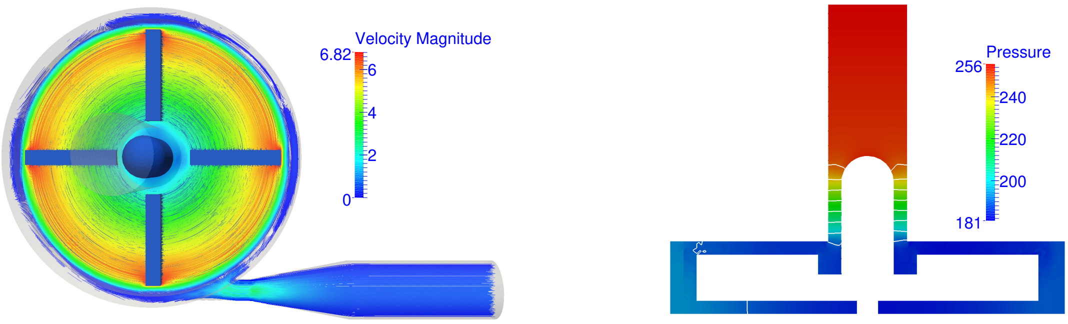


(a) PC order $L = 3$.

(b) PC order $L = 4$.

(c) PC order $L = 5$.

Figure 6.2: Comparison of the computational cost of GMRES, ML$_{exact}$ and ML$_{inexact}$.



(a) Streamlines and velocity magnitude.

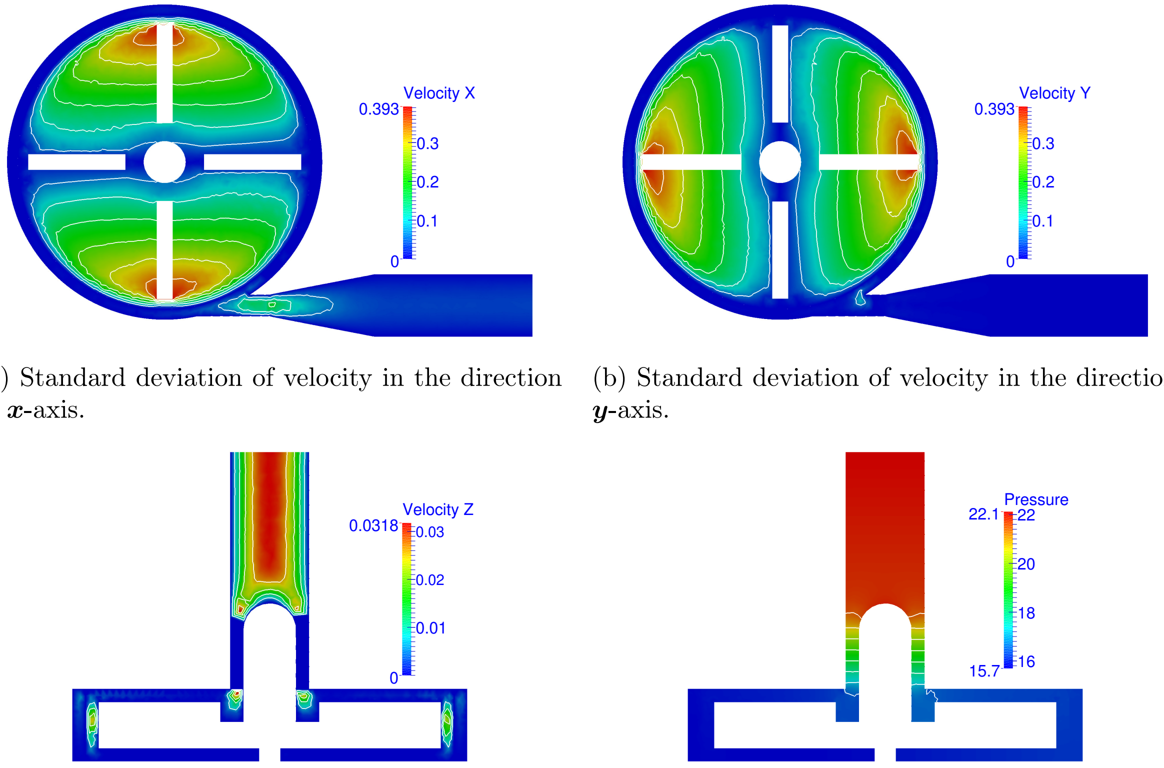
(b) Mean value of pressure.

Figure 6.3: Mean value of the flow velocity (m/s) and the pressure (Pa).

computations is reduced by a factor 10 or even higher. A comparison of the computational costs is highlighted in Figure 6.2.

Herewith, we define the computational cost for this test by taking $C_s := N_{\bar{A}_0} \bar{N}_{iter,\bar{A}_0}$. One can read from Figure 6.2 that the strategy with the GMRES as the global linear solver is more powerful by saving the computational effort in contrast to the exact Multilevel method. The GMRES solver preconditioned with the Mean-based method requires only $20 - 40\%$ of the computing power as the exact Multilevel solver needs, especially when PC order increases. On the other hand, the inexact Multilevel strategy has clearly a better performance than the other two methods, the saving of the computational cost is significant. Even when PC order $L = 3$, the inexact Multilevel strategy requires one more Newton iteration, the total computational cost is lower than any other strategy.

The mean value of the velocity and the pressure on a cross-section of the pump chamber is presented in Figure 6.3. The highest values of the pressure are placed in the region around



(a) Standard deviation of velocity in the direction of $\boldsymbol{x}$-axis.

(b) Standard deviation of velocity in the direction $\boldsymbol{y}$-axis.

(c) Standard deviation of velocity in the direction of $\boldsymbol{z}$-axis.

(d) Standard deviation of pressure.

Figure 6.4: Standard deviations for the velocity (m/s) and the pressure (Pa).

the inlet due to the specified inflow boundary condition. In the pump chamber, the pressure is differing from both sides by reason of the effect of the Coriolis force and the centrifugal force. Obviously, the velocity in the area around the rim (close to the outer edge) is higher than the region near to the hub because of the rotor's speed.

In Figure 6.4, the standard deviation of the three velocity components and the pressure are illustrated, the highest uncertainty occurs in the region where the mean value is extreme. Concerning the pressure, the uncertainty concentrates in the region around the inlet. The standard deviation of the velocity is located more around the rotor's rim and the diffuser in the direction of $\boldsymbol{x}$- and $\boldsymbol{y}$-axis. Along the direction of $\boldsymbol{z}$-axis, the majority of the uncertainties stays at the inlet, also around the outer edges. However, the mean value and the standard deviation indicate evidently a laminar regime due to the modification of the density (Table 6.1), the flow layers are rather ordered. Therefore, we expect more turbulent behavior of the fluid in the next section (Section 6.2). Notwithstanding, the three solving strategies are examined with the different settings, the results sustain the arguments in Chapter 5 that the inexact Multilevel method is a reasonable choice for our problem.

6.2 Model Calibration

This section is devoted to the final results of quantifying three uncertainty sources within a FDA blood pump by using the generalized Polynomial Chaos method. The key elements, which are responsible for dealing with different complexities in this work, are listed as follows:

- Moving mesh: shear layer update approach.
- Turbulent flow: Variational Multiscale method.

- Uncertainty Quantification: generalized Polynomial Chaos expansion.
- Stochastic Galerkin system: PCE Multilevel preconditioner.
- Flow problem: Schur Complement preconditioner.

The shear layer update approach enables the rotor's movement to be handled effortlessly, it preserves the possibility of staying in the framework of the continuous Galerkin method. By dividing the computational domain into separated functions, the revolving realization affects only the relevant regions, a merely limited number of cells has to be deformed in terms of the grid shape. Moreover, the spatial coordinate correction on the rotating domain has to be considered due to the moving effect, whereas, only the shape functions, which are located on the shearing elements, need to be updated because of the deformation of the grid. Therefore, a simplification is achieved for the moving mesh technique. Furthermore, with the help of a special treatment of the domain decomposition, the localization of the data is optimized. The solution correction procedure is embedded into the standard solution update routines, the scalability of the parallel calculation is then extended.

Inside the pump chamber, the blood flow undergoes a high rotational speed arising from the blades, this action brings an external thrust and compels the fluid to be in the turbulence regime. For this reason, the residual-based Variational Multiscale method comes into play and builds a link between the stabilized finite element methods and the turbulence models.

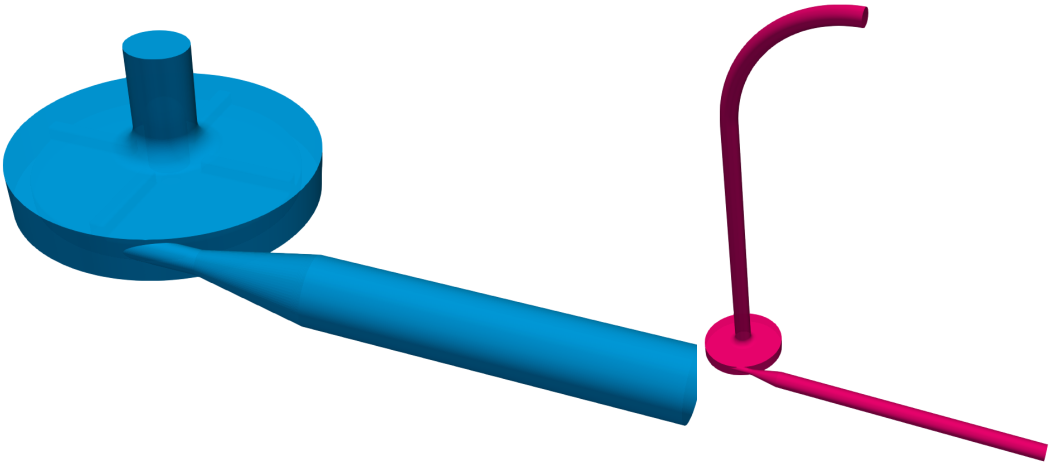
Despite the fact that the intrusive Polynomial Chaos expansion method requires more implementation effort in order to establish the Galerkin system, whereas this approach is the most effective way for modeling a stochastic system in a low-dimensional stochastic space. One has to admit that there are even more uncertain sources during the development of a complex mechanical instrument. But still, these three uncertain parameters, which we studied, cover the most aspects of quantifying uncertainties in a rotating device. The inflow boundary condition leads to a significant deviation on the numerical results of a blood flow, because it is taken as an idealized inflow profile description [58]. The dynamic viscosity is in consideration of the patient-specific criterion. The angular speed takes the operational variation into account. By expressing the stochastic solutions with the Chaos Polynomials, the stochastic space and spacial space are strongly coupled at each time step comparing to the non-intrusive methods. Furthermore, as the intrusive approach inherits the well-known properties of the Galerkin method, a spectral convergence behavior can be expected based on this fact.

One of the main challenges for exploiting the intrusive Galerkin method is the solution strategy, the Multilevel method is then proposed in this work to cope with this matter. Our Polynomial Chaos expansion Multilevel approach can be considered as a scale decomposition method, which is proven to be greatly efficient for many numerical methods. The Multilevel method, which is employed in this work, is inherited conceptually from the Multigrid method. In general, the Multigrid preconditioner tries to reduce errors on different wave-length components. Comparatively, the Polynomial Chaos expansion Multilevel preconditioner tries to reduce the errors components on different Chaos Polynomial orders, as the stochastic mode solutions on the higher order Chaos Polynomials contribute much less than those on the lower orders to the stochastic moments.

Undoubtedly, choosing an appropriate smoother is also a crucial point to ensure the efficiency of the PCE Multilevel strategy. Comparing to the Multigrid approach, the Jacobi and Gauss-Seidel schemes are commonly applied, because they are sufficient in the sense for the smoothing process and can easily be constructed. The smoother, we choose for the Multilevel method, is the Mean-based preconditioner. This approach can be understood as an adapted Block-Jacobi scheme, the efficiency is then also comparable. Furthermore, the feasibility of solving efficiently the linear system associated with the mean block is another practical challenge, thus the Schur Complement based preconditioning scheme comes into play. By applying the Variational Multiscale formulation, the flow solver does not have to suffer anymore from the saddle-point structure as the incompressible Navier-Stokes equations. Meanwhile, as the pressure-pressure block is virtually accessible, it facilitates therefore the construction of the preconditioner for the Schur Complement matrix in our VMS system. Additionally, the reusability of the Schur Complement preconditioner without the reconstruction of the full structure is workable, hence, the PCE Multilevel method can be further accelerated.

6.2.1 Model Geometry

We state here the geometry of the blood pump, which is referred to the information in [55]. We post-processed the original geometry provided from the CAD (Computer Aided Design) configuration by shortening the inflow and outflow pipes for the sake of saving the computing power. Another reason is that we concentrate more on the fluid within the pump's chamber rather than the flow in the pipes. Note that, defining the location and the analytical formulation of the inflow and outflow boundary conditions can be extremely difficult [83, 69, 15], whereas we ignore this complication in our work. The simplification of the geometry is taken as in Figure 6.5.



(a) The modified geometry used by this work. (b) The CAD geometry provided by FDA.

Figure 6.5: Illustration of the original CAD geometry (red) provided by FDA and the modified geometry considered in this work (blue).

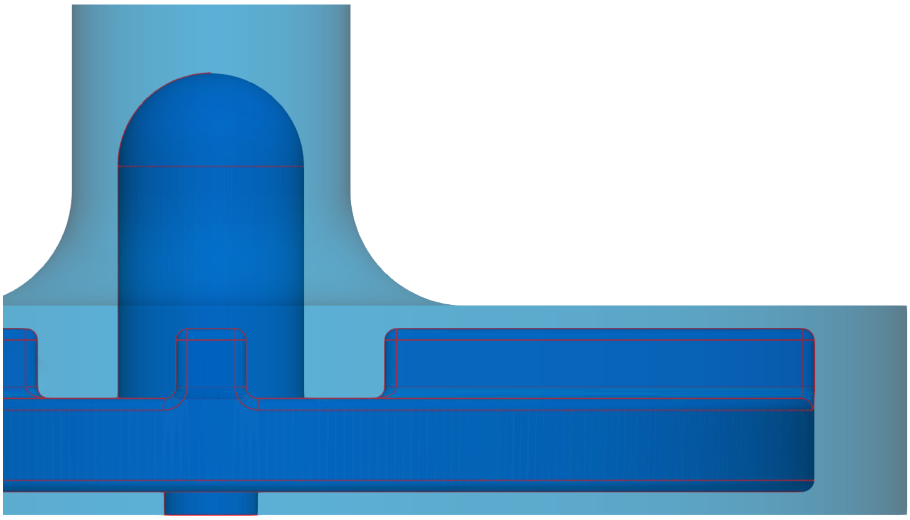
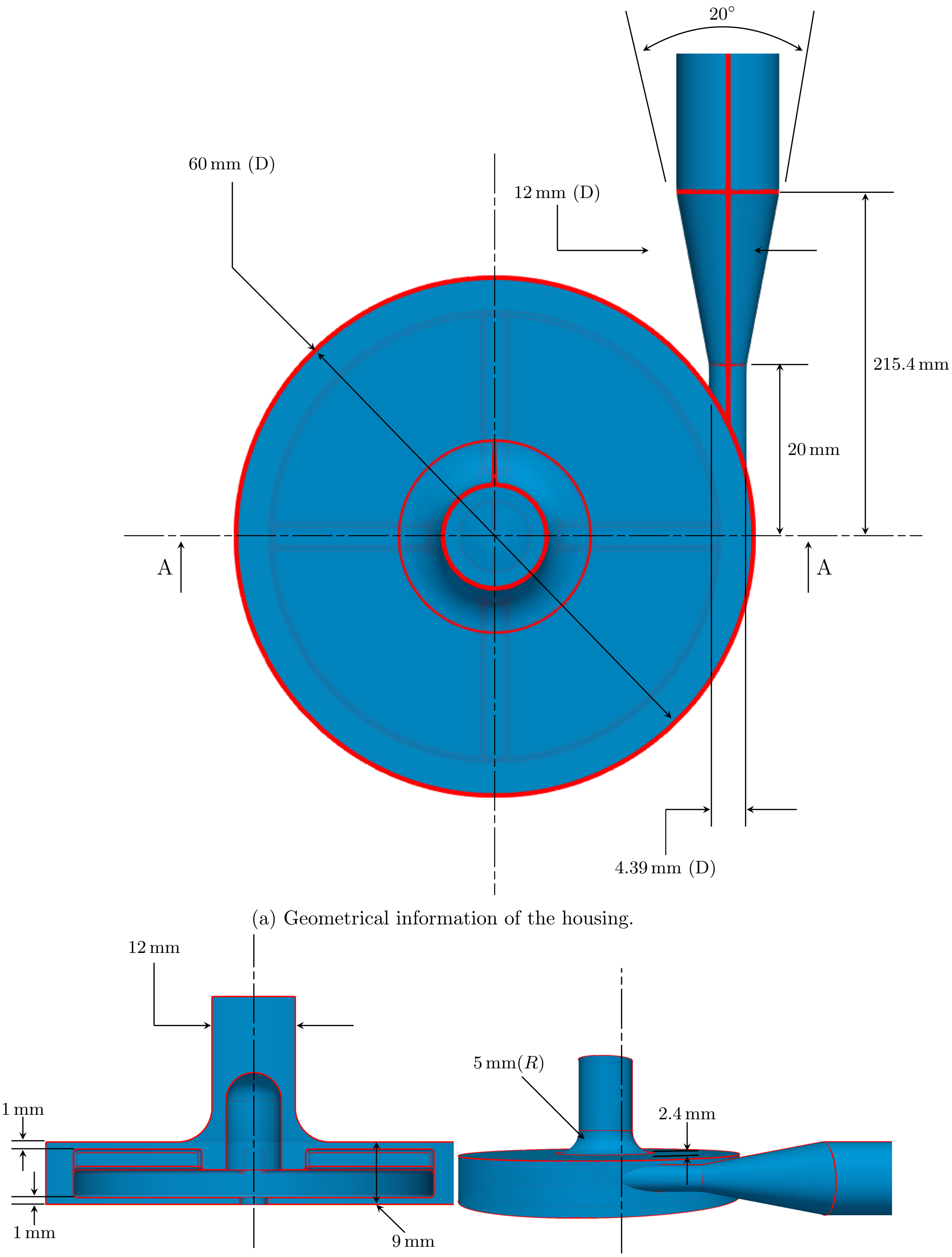


Figure 6.6: The visual clearance between the housing (light blue) and the rotor (dark blue).



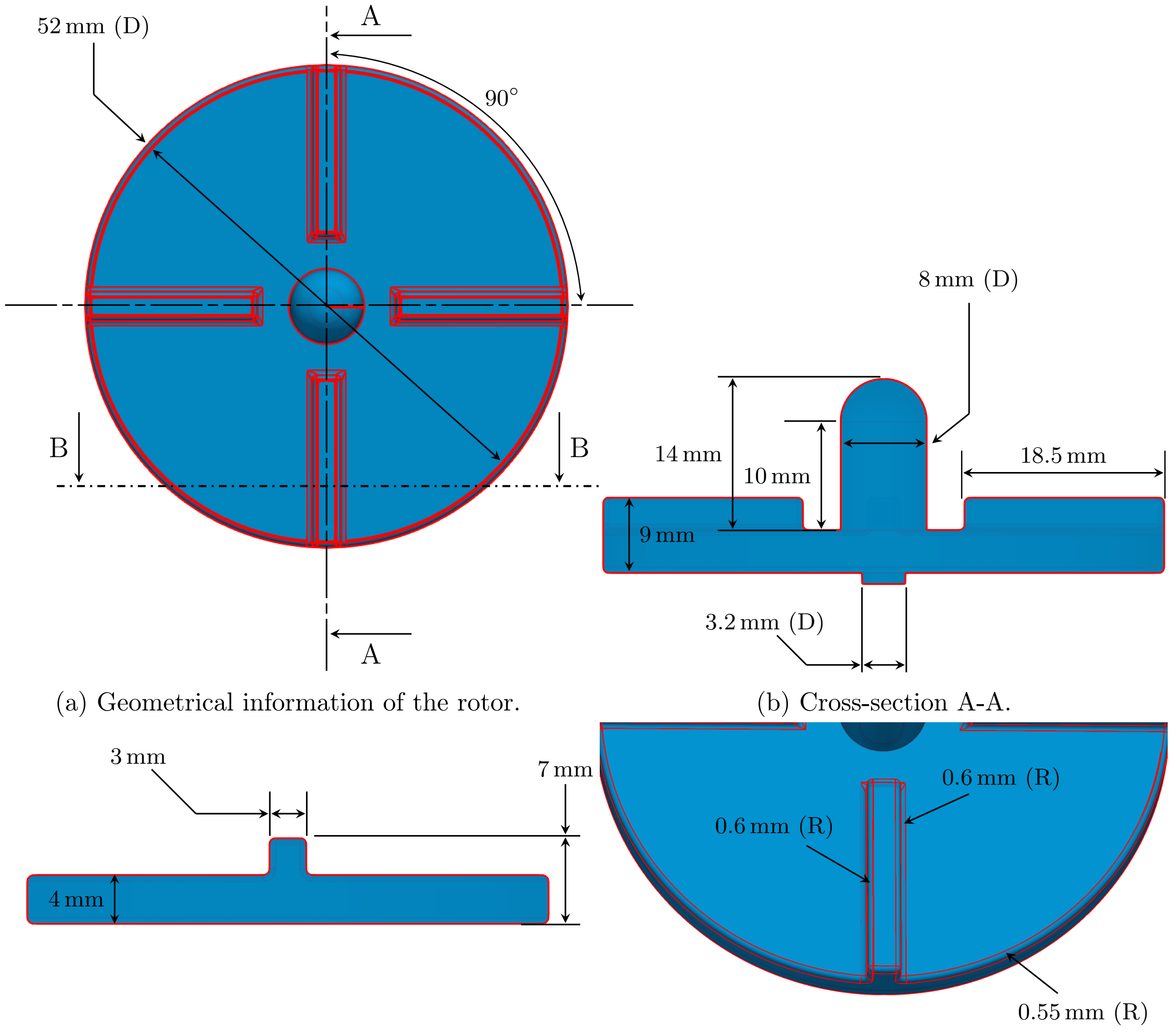
(a) Geometrical information of the housing.

(b) Cross-section A-A.

(c) Detailed information of the nozzle part.

Figure 6.7: Geometrical information of the blood pump's housing.

The clearance between the rotor and the housing is illustrated in Figure 6.6, the rotor is embedded clearly into the center of the chamber. However, the rotor holds the most volume of the chamber. In other words, the flow in the chamber can be extremely sheared due to the tiny space and the high rotating speed, the decrease of the section in the inlet caused by the rotor hub accelerates this occurrence as well. In reality, the shaft under the rotor is much longer than it is shown in Figure 6.6, because it is actually connected and moderated by an external motor. However, the motor is not directly modeled in this configuration, its outcome as the rotor's revolution is imitated instead.



(a) Geometrical information of the rotor.

(b) Cross-section A-A.

(c) Cross-section B-B.

(d) Details of the blade.

Figure 6.8: Geometrical information of the rotor.

Figure 6.7 provides the geometrical information about the blood pump, which consists mainly of a chamber and two pipes. The chamber is responsible for accommodating the rotor, and two pipes conduct the flow inward and outward the chamber. As mentioned already in Chapter 2, the complete device is rather small, the diameter of the chamber is only 60 mm, and two pipes have an equal diameter of 12 mm. In Figure 6.7c, a nozzle structure (or diffuser) is placed between the chamber and the outlet pipe. The diffuser's shape is designed to minimize the energy loss and maintain the fluid's axial orientation in the outlet, such that the conversion of the velocity head into the pressure head is more efficient. Besides, the nozzle has only 2.4 mm clearance to the upper side of the pump chamber. Therefore the fluid pushed by the rotor can enter the outlet without any deviation. The clearance between the housing and the rotor (Figure 6.6) is even more meticulous in Figure 6.8b, there is only 1 mm between the blades and the upper chamber, as well as for the rotor disk and the lower chamber.

In Figure 6.8, the further information of the rotor's geometry is detailed. The rotor has a diameter of 52 mm and its disk has a thickness of 4 mm. Four blades are located on the disk in an equiangular way, a hub is placed in the middle with a diameter of 8 mm. Referring also to Figure 6.7b, one can recognize that the fluid should be accelerated after it encounters the rotor hub because of a diminution in space of the inlet tube. The fluid is led quickly into the chamber and spread on the disk, afterward the blades shove the blood toward the outlet.

In principle, we are only interested in the fluid domain, namely Figure 6.7. As we have to deal with the moving mesh, the rotor's shape is thus also very important as it defines the fluid domain at each time step (Figure 6.8).

One can perceive that the confined space in the chamber and the high revolving speed of the rotor induce together a chaotic change in the velocity and the pressure, it is the central difficulty of the mathematical and physical modelings of our blood pump simulation.

6.2.2 Governing Equations

After introducing the blood pump's geometry, we continue with the mathematical modeling of the stochastic flow in this configuration. Our considered problem is essentially covered by the incompressible Navier-Stokes equations combined with the shear layer update approach. By reason of the turbulent flow, we extend the weak formulation of the incompressible NSE by using the Variational Multiscale Method. Moreover, as mentioned previously, three uncertain input sources are modeled with the generalized Polynomial Chaos expansion, a fully coupled system is therewith developed.

Strong Formulation

Recalling the stochastic unsteady incompressible Navier-Stokes equations under the context of the moving mesh technique (Section 2.3), it is given as:

$$
\begin{aligned}
&\frac{\partial \boldsymbol{u}(\boldsymbol{\xi})}{\partial t} + (\boldsymbol{u}(\boldsymbol{\xi}) - \boldsymbol{u}^r(\boldsymbol{\xi})) \cdot \nabla \boldsymbol{u}(\boldsymbol{\xi}) && && (6.12a)\\
&\qquad -\frac{\mu(\boldsymbol{\xi})}{\rho}\Delta \boldsymbol{u}(\boldsymbol{\xi}) + \frac{1}{\rho}\nabla p(\boldsymbol{\xi}) = \mathbf{0}\ , && \text{in } \mathcal{D}^t \times [0,T]\ , &&\\
&\nabla \cdot \boldsymbol{u}(\boldsymbol{\xi}) = 0\ , && \text{in } \mathcal{D}^t \times [0,T]\ , && (6.12b)\\
&\boldsymbol{u}^r(\boldsymbol{\xi}) = \boldsymbol{d} \times \boldsymbol{\omega}(\boldsymbol{\xi})\ , && \text{in } \mathcal{D}^t_{rot} \times [0,T]\ , && (6.12c)\\
&\boldsymbol{u}^r(\boldsymbol{\xi}) = \mathbf{0}\ , && \text{in } \mathcal{D}_{stat} \times [0,T]\ , && (6.12d)\\
&\boldsymbol{u}(\boldsymbol{\xi}) = \boldsymbol{u}_I(\boldsymbol{\xi})\ , && \text{on } \Gamma^t_{in} \times [0,T]\ , && (6.12e)\\
&(-\mathbb{1}p(\boldsymbol{\xi}) + \frac{\mu(\boldsymbol{\xi})}{\rho}\nabla \boldsymbol{u}(\boldsymbol{\xi})) \cdot \boldsymbol{n} = \mathbf{0}\ , && \text{on } \Gamma^t_{out} \times [0,T]\ , && (6.12f)\\
&\boldsymbol{u}(\boldsymbol{\xi}) = \boldsymbol{d} \times \boldsymbol{\omega}(\boldsymbol{\xi})\ , && \text{on } \Gamma_{rotor} \times [0,T]\ , && (6.12g)\\
&\boldsymbol{u}(\boldsymbol{\xi}) = \mathbf{0}\ , && \text{on } \Gamma_{wall} \times [0,T]\ . && (6.12h)
\end{aligned}
$$

$\boldsymbol{u}$ and p are the velocity and the pressure, respectively. ρ is the density, and μ is the dynamic viscosity. $\boldsymbol{u}^r$ is the moving velocity of the grid. $\boldsymbol{u}_I$ is the inflow boundary condition, and $\boldsymbol{\omega}$ is the angular speed. $\boldsymbol{d}$ is the distance from a physical point in the rotating domain to the rotating axis, which is the $\boldsymbol{x}$-axis in our configuration. The equations are defined in a bounded but open domain $\mathcal{D}^t \in \mathbb{R}^d, d = 3$. The domain $\mathcal{D}^t$ is subject to be different between two consecutive time steps. $\mathcal{D}^t_{rot}$ is the rotating domain, and $\mathcal{D}^t_{stat}$ is the static domain. $[0,T]$ is the time interval of interest.

Three Dirichlet boundary conditions are given on the inflow boundary (Γ_{in}) , the rigid wall (Γ_{wall}) and the rotor's surface (Γ_{rotor}), respectively (Figure 6.9). The outflow boundary (Γ_{out}) condition (Equation (6.12f)) is chosen as the "do-nothing" condition, also called the natural boundary condition. Additionally, $\boldsymbol{\xi}$ is a finite-dimensional random vector, which is defined on a probability space $(\Theta, \mathcal{F}, \mathbb{P})$ (Chapter 3). Note that, there is no extra body force applying to the system, thence, on the right-hand side of the momentum equation (Equation (6.12a)), $\boldsymbol{f}$ is directly set to $\mathbf{0}$.

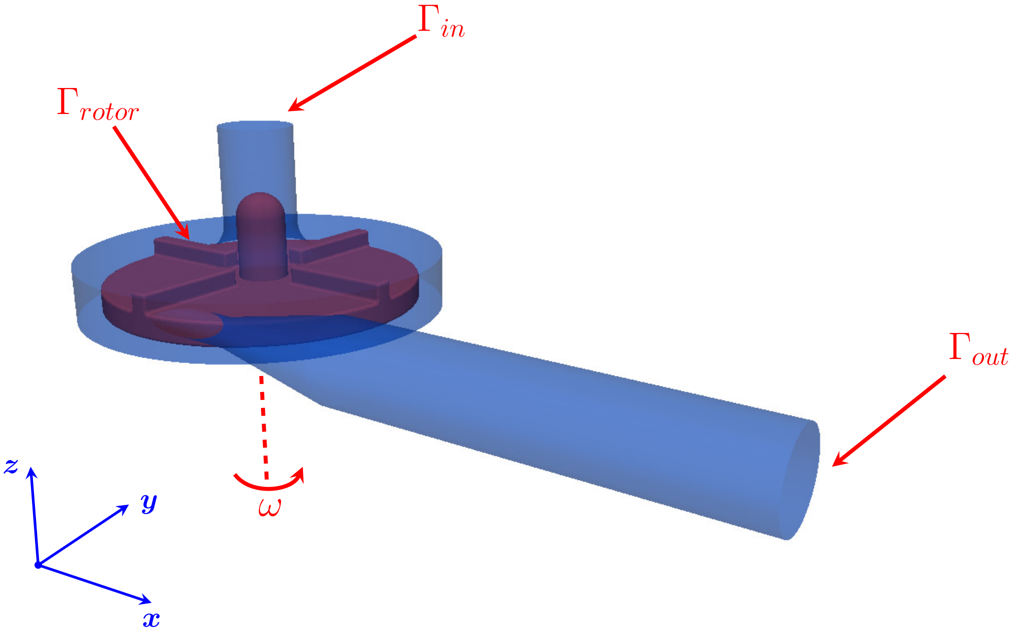


Figure 6.9: Illustration of the boundaries and the axis of rotation on the blood pump geometry.

We consider in this study only three uncertain sources, viz., the inflow boundary condition $\boldsymbol{u}_I(\boldsymbol{\xi})$, the dynamic viscosity $\mu(\boldsymbol{\xi})$ and the angular speed $\boldsymbol{\omega}(\boldsymbol{\xi})$. These three random inputs are modeled in the following way [157]:

$$\boldsymbol{u}_I(\boldsymbol{\xi}) = \boldsymbol{u}_{I,0}\psi_0(\boldsymbol{\xi}) + \sigma_1 \boldsymbol{u}_{I,0}\psi_1(\boldsymbol{\xi}) \ , \tag{6.13a}$$

$$\mu(\boldsymbol{\xi}) = \mu_0\psi_0(\boldsymbol{\xi}) + \sigma_2\mu_0\psi_2(\boldsymbol{\xi}) \ , \tag{6.13b}$$

$$\boldsymbol{\omega}(\boldsymbol{\xi}) = \boldsymbol{\omega}_0\psi_0(\boldsymbol{\xi}) + \sigma_3\boldsymbol{\omega}_0\psi_3(\boldsymbol{\xi}) \ . \tag{6.13c}$$

Three uncertain parameters above are mutually independent and assumed to be in the Uniform distribution, i.e. $\xi_i \sim U(-1,1), i = 1,2,3$. Here we employ the first order expansion for the random inputs (Equation (6.13)), which are similar to the definitions in Section 6.1. $\boldsymbol{u}_{I,0}$, $\boldsymbol{\omega}_0$ and μ_0 are the mean values for these three random inputs. $\sigma_i, i = 1,2,3$ are the decay factors, which assess the amount of the uncertainty with respect to their mean value.

The mean value of the inflow boundary condition is chosen as a Poisseuille profile, which is defined as follows:

$$\boldsymbol{u}_{I,0}(\boldsymbol{x}) := \boldsymbol{u}_{I,0}(x,y,z) = \begin{bmatrix} 0 \\ 0 \\ -U_{max}(1-(x^2+y^2)/R^2) \end{bmatrix} \ , \tag{6.14}$$

U_{max} is the maximum value of the inflow and R is the radius of the inflow cylinder (Figure 6.7).

Variational Formulation

Although we are intended to solve the incompressible Navier-Stokes equations on the pump geometry. However, because of the strong rotor's revolution, it is merely impossible to represent the turbulent flow based on our computing resource without further additional assumptions and models on the mathematical formulations. Hence, as introduced in Section 2.2, the Variational Multiscale Method is chosen for modeling the blood flow, and it introduces additional terms only in the discrete variational formulation based on the scale decomposition technique. We consider then the spectral-stochastic finite element method to treat the input uncertainties combining with the Chaos Polynomials. After having obtained the stochastic solutions from the coupled stochastic Galerkin system, the statistical moments can be computed and examined.

More specifically, the resulting system consists of multiple deterministic VMS systems, which can be studied by using the standard finite element method. We state here the approximation of the discrete solutions of velocity $\boldsymbol{u}_h$ and pressure p_h with the corresponding gPC expansions, they are defined as:

$$\boldsymbol{u}_h(\boldsymbol{x},t,\boldsymbol{\xi}) = \sum_{i=0}^{P} \boldsymbol{u}_{h,i}(\boldsymbol{x},t)\psi_i(\boldsymbol{\xi}) \in V^h \otimes \mathcal{S}^P \ , \tag{6.15a}$$

$$p_h(\boldsymbol{x},t,\boldsymbol{\xi}) = \sum_{i=0}^{P} p_{h,i}(\boldsymbol{x},t)\psi_i(\boldsymbol{\xi}) \in Q^h \otimes \mathcal{S}^P \ . \tag{6.15b}$$

Here, $\boldsymbol{u}_{h,i}$ and $p_{h,i}$ are the discrete stochastic mode solutions on the space V^h and Q^h in regard to the velocity and the pressure. $\psi_i(\boldsymbol{\xi})$ are the basis polynomials corresponding to the assumed probability density functions. The truncated number P is obtained by:

$$P+1 = \frac{(M+L)!}{M!L!} \ , \tag{6.16}$$

where, M is the number of random parameters, L is the desired polynomial expansion order. Hence, we can insert Equation (6.15) into the discrete variational formulation of Equation (6.12), the Galerkin-type projection constructs an expansion of the system variables by projecting the system onto the space spanned by the pre-defined orthonormal Chaos Polynomials. In this case especially, these pre-defined orthogonal polynomials are constructed based on our three random variables $\boldsymbol{\xi}$. We state immediately the discrete variational formulation combined with the two-scale residual-based Variational Multiscale model for our rotating system:

Find $\boldsymbol{u}_h \in V^h \otimes \mathcal{S}^P$, $p_h \in Q^h \otimes \mathcal{S}^P$, such that,

$$\begin{aligned}
&(\frac{\partial \boldsymbol{u}_{h,k}}{\partial t}, \boldsymbol{v}_h) + \sum_{i=0}^{P}\sum_{j=0}^{P}((\hat{\boldsymbol{u}}_{h,i} - \boldsymbol{u}^r_{h,i}) \cdot \nabla \boldsymbol{u}_{h,j}, \boldsymbol{v}_h)\, C_{ijk} \\
&\quad + \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho}(\nabla \boldsymbol{u}_{h,j}, \nabla \boldsymbol{v}_h)\, C_{ijk} - \frac{1}{\rho}(p_{h,k}, \nabla \cdot \boldsymbol{v}_h) \\
&\quad + (\tau_M[\frac{\boldsymbol{u}_{h,k}}{\partial t} + \sum_{i=0}^{P}\sum_{j=0}^{P}(\boldsymbol{u}_{h,i} - \boldsymbol{u}^r_{h,i}) \cdot \nabla \boldsymbol{u}_{h,j}\, C_{ijk} \\
&- \sum_{i=0}^{P}\sum_{j=0}^{P} \frac{\mu_i}{\rho}\Delta \boldsymbol{u}_{h,j}\, C_{ijk} + \frac{1}{\rho}p_{h,k}], (\hat{\boldsymbol{u}}_{h,0} - \boldsymbol{u}^r_{h,0}) \cdot \nabla \boldsymbol{v}_h) \\
&\quad + (\tau_C \nabla \cdot \boldsymbol{u}_{h,k}, \nabla \cdot \boldsymbol{v}_h) = \boldsymbol{0} \ , \qquad \text{in } \mathcal{D}^t \times [0,T] \ ,
\end{aligned} \tag{6.17a}$$

$$
\begin{aligned}
&(\frac{1}{\rho}\nabla \boldsymbol{u}_{h,k}, q_h) && (6.17b)\\
&+(\tau_M[\frac{\partial \boldsymbol{u}_{h,k}}{\partial t} + \sum_{i=0}^{P}\sum_{j=0}^{P}(\boldsymbol{u}_{h,i} - \boldsymbol{u}^r_{h,i}) \cdot \nabla \boldsymbol{u}_{h,j}\, C_{ijk} \\
&-\sum_{i=0}^{P}\sum_{j=0}^{P}\frac{\mu_i}{\rho}\Delta \boldsymbol{u}_{h,j}\, C_{ijk} + \frac{1}{\rho}\nabla p_{h,k}], \nabla q_h) = 0 \;, && \text{in } \mathcal{D}^t \times [0,T] \;.
\end{aligned}
$$

$\forall k = 0, ..., P$, $\forall \boldsymbol{v}_h \in V^h$ and $\forall q_h \in Q^h$. Here, $\hat{\boldsymbol{u}}_{h,i}$ is defined as [9, 133]:

$$
\hat{\boldsymbol{u}}_{h,i} := \boldsymbol{u}_{h,i} + \tau_M[\frac{\boldsymbol{u}_{h,i}}{\partial t} + (\boldsymbol{u}_{h,i} - \boldsymbol{u}^r_{h,i}) \cdot \nabla \boldsymbol{u}_{h,i} - \frac{\mu_i}{\rho}\Delta \boldsymbol{u}_{h,i} + \frac{1}{\rho}\nabla p_{h,i}] \;. \tag{6.18}
$$

We provide also the explicit forms of the stabilization coefficients which are employed in this study:

$$
\tau_M := (\frac{4}{\Delta t^2} + \frac{\|\bar{\boldsymbol{u}}_{h,0}\|}{h^4} + C\frac{\mu_0}{\rho h^2})^{1/2} \;, \tag{6.19a}
$$

$$
\tau_C := (\frac{\tau_M}{h^2})^{-1} \;. \tag{6.19b}
$$

Here, Δt is the time step size, h is the mesh size parameter and the coefficient C is chosen as 0.5 in our computation.

The variational formulation (Equation (6.17)) is obtained by multiplying the basis polynomials $\psi_i(\boldsymbol{\xi})$ on the both sides of the weak form generated by the VMS (with an additional consideration of our shear layer update approach). V^h and Q^h are the conforming finite element spaces for the velocity and the pressure, which are already defined in Section 3.3. The mesh consists explicitly of three different mesh cells, viz., tetrahedrons, hexahedrons and pyramids. And yet, only the first order elements are utilized for the velocity and the pressure. $\mathcal{S}^P$ is the approximated stochastic space, which is spanned by a set of basis polynomials $\{\psi_0, ..., \psi_P\}$ (Section 3.2). Moreover, C_{ijk} is again the third-order tensor product, and it is given by:

$$
C_{ijk} := \frac{\langle \psi_i \psi_j, \psi_k \rangle}{\langle \psi_k, \psi_k \rangle} \;. \tag{6.20}
$$

6.2.3 Numerical Simulation

Our numerical simulation is about solving the coupled stochastic Galerkin system (Equation (6.17)) in order to obtain the stochastic flow in the pump geometry. Several iterative solvers, as previously introduced in Chapter 4, are involved in the solving algorithm. The settings of these solvers and the physical parameters are shown in the Table 6.3.

In Table 6.3, #element represents the amount of mesh cells which is used for triangulating our pump geometry, DOFs (d) is the degrees of freedom for the deterministic problem, DOFs (g) is for the global system. h_{max} and vol_{max} are two important restriction criteria for our mesh generation, h_{max} is the maximum length of an individual mesh cell, and vol_{max} is the maximum value of the cell volume. σ_1, σ_2 and σ_3 are the variation factors with respect to the mean value for the uncertain parameters, we choose synthetically $\sigma_{1,2,3} = 10\%$ in order to keep consistent with the stationary case (Section 6.1). Note that, the pump geometry is partitioned into $2,984,259$ grids, it results $2,274,904$ degrees of freedom in the deterministic case and $45,498,080$ degrees of freedom in the global linear system. Because we truncate the Chaos Polynomials to degree 3, i.e. there are 20 stochastic modes (Equation (6.20)).

As multiple linear solvers and preconditioners are involved, we list the relationships between the solvers/preconditioners and the linear systems in our solving algorithm:

Physical parameter					
ρ	$1035\,\mathrm{kg/m^3}$	μ	$0.0035\,\mathrm{Pa\cdot s}$	U_{max}	$0.55\,\mathrm{m/s}$
RPM	$2500\,\mathrm{r/min}$	ω	$261.8\,\mathrm{rad/s}$	R	$0.006\,\mathrm{m}$
Stochastic discretization					
M	3	L	3	$P+1$	20
σ_1	0.1	σ_2	0.1	σ_3	0.1
Time discretization					
θ-Scheme	0.5	Δt	$0.0001\,\mathrm{s}$	T	0.12
Space discretization					
d	3	#element	2984259	DOFs (d)	2274904
DOFs (g)	45498080	h_{max}	$3.5\times10^{-4}\,\mathrm{m}$	vol_{max}	$5\times10^{-12}\,\mathrm{m}^3$
Global system					
Nonlinear solver (Newton)					
Tol_{abs}	1.0×10^{-9}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000
Forcing	EisenstatWalker2				
Linear solver (FGMRES)					
Tol_{abs}	1.0×10^{-9}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000
Preconditioner (PCE Multilevel)					
Tol_{abs}	1.0×10^{-9}	Tol_{rel}	1.0×10^{-1}	$Iter_{max}$	3
Mean block					
Linear solver (FGMRES)					
Tol_{abs}	1.0×10^{-9}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000
Preconditioner (Schur Complement)					
FGMRES for block S					
Tol_{abs}	1.0×10^{-16}	Tol_{rel}	1.0×10^{-16}	$Iter_{max}$	10
GMRES for block A_0 (preconditioned with BoomerAMG)					
Tol_{abs}	1.0×10^{-9}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000
CG for matrix $\tilde{S}_{app}$ (preconditioned with BoomerAMG)					
Tol_{abs}	1.0×10^{-9}	Tol_{rel}	1.0×10^{-6}	$Iter_{max}$	1000

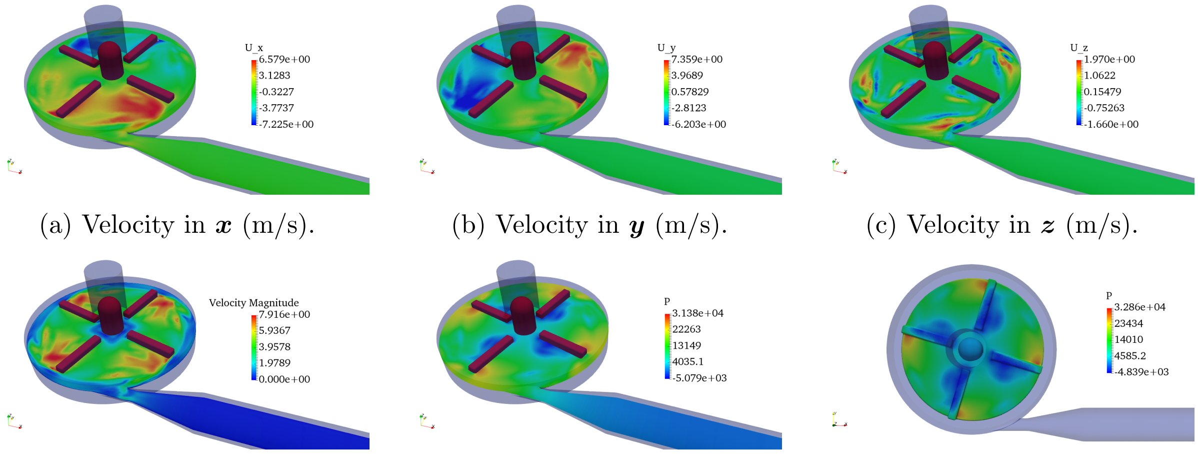
Table 6.3: General setting for the blood pump simulation.

- Nonlinear equation: Inexact Newton scheme + "choice 1" Eisenstat and Walker time steeping.
- Global linearized equation: FGMRES + PCE Multilevel preconditioner.
- Mean block (Mean-based preconditioner from PCE Multilevel): FGMRES + Schur Complement preconditioner.
- Schur Complement preconditioner:
  - Block S: FGMRES.
  - Block A_0: GMRES + BoomerAMG.
  - Matrix $\tilde{S}_{app}$: CG + BoomerAMG.

Furthermore, the computation is performed on bwForCluster ML&WISO cluster with 128 computing nodes, each node has 16 Intel Xeon Haswell processors and 64 GB memory. Accordingly, among these 128 computing nodes, 40 nodes are used for the static domain and 88 nodes are used for the rotating domain. The simulation is implemented in our open source finite element library HiFlow^3 [5, 78, 140, 60]. The total computational time lasts for approximately 96 hours.

Mean and Standard Deviation

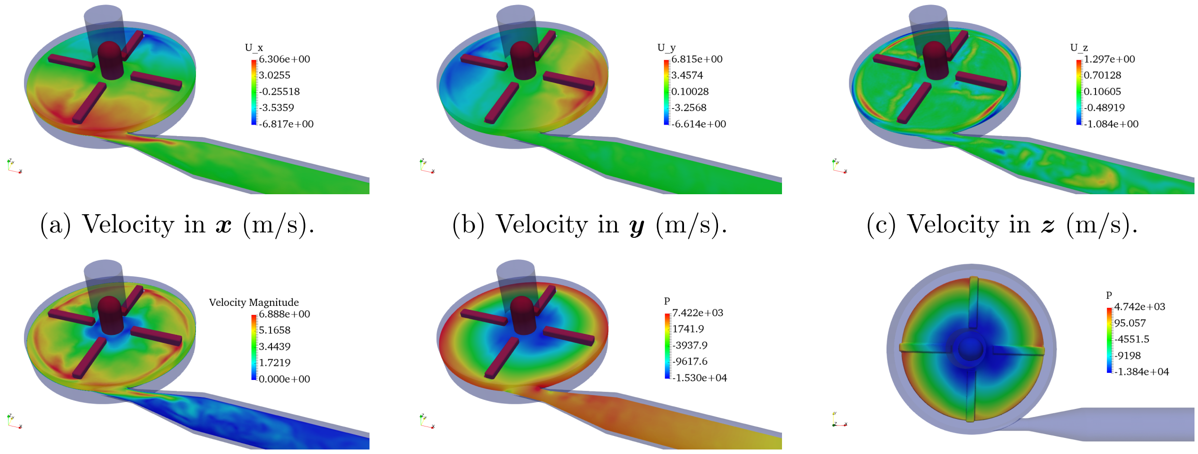
We illustrate the selected results of the mean values and the standard deviations for the blood flow in the pump geometry at two different instants, namely at the time step 50 and 1200. Figures 6.10, 6.11, 6.13 and 6.14 are split into 6 subfigures, the first five subfigures are the velocity in the direction of $\boldsymbol{x}$-, $\boldsymbol{y}$-, $\boldsymbol{z}$-axis, the velocity magnitude and the pressure on a cross-section (the cross-section is located at 6.5 mm from the bottom of the pump housing). The last subfigure shows the pressure value on the rotor's surface.



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.10: Mean value at the time step 50.

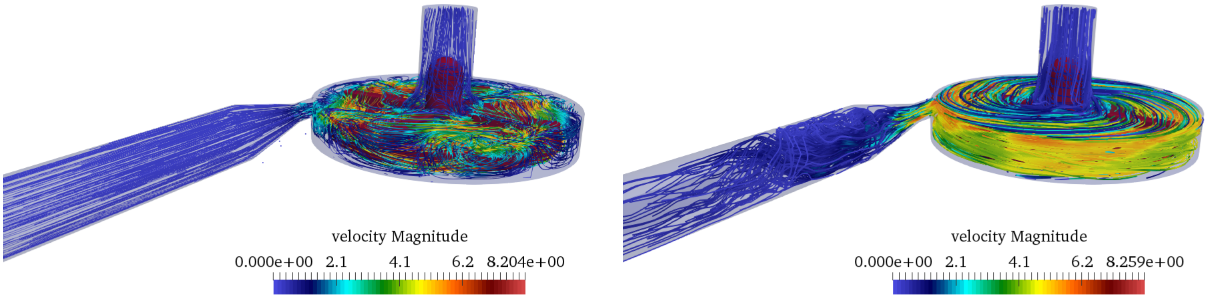


(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.11: Mean value of at the time step 1200.

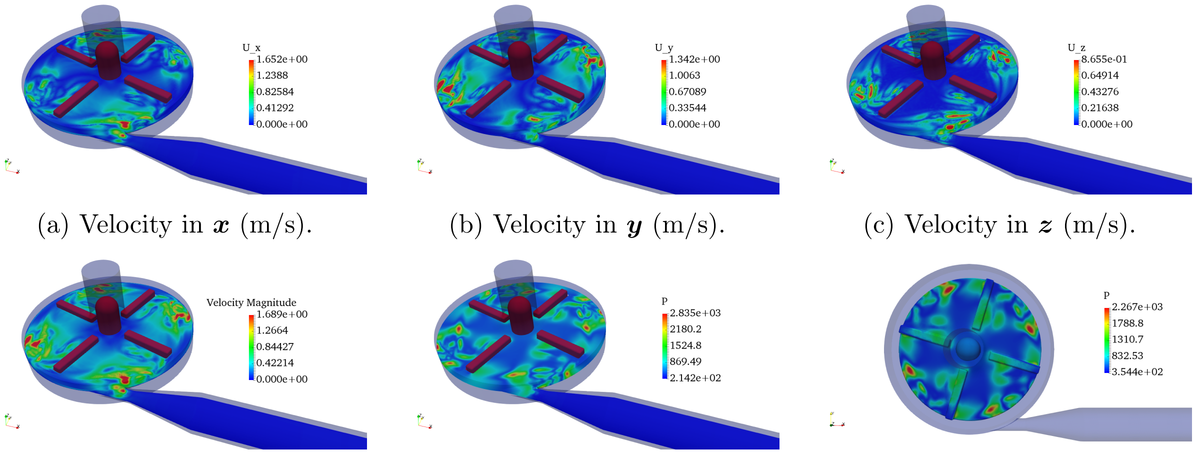
The results at the time step 50 and 1200 show two stages of the fluid. The first time instant is when the rotor operates during the first revolution, i.e. at the very beginning when the impellers start pressing the blood. And the latter case is when the flow is already fully developed after several completions of the revolution. For the first case (Figure 6.10), the high velocity regime is mainly generated in the pump chamber. No flow velocity is produced after the diffuser (Figure 6.10d), and the pressure is even negative in this region (Figure 6.10e). On the other hand, after the fluid is fully developed (Figure 6.11), the flow coming from the inlet is diverted in the blade-to-blade passage, the shockless velocity is therefore no longer guaranteed. The separation of the flow can be observed at the outer edge of the blades (Figure 6.11d), this separation is the main source of the energy loss and can further affect the flow direction toward the outlet. As a result of the inertia effect, an increase in flow velocity can occur a decrease of the pressure (Figures 6.11e and 6.11f). Therefore, the pressure on the suction side is slightly higher than on the pressure side, the difference can be even larger if the blades are curved. Furthermore, the 3D visualization of the streamline of these two time steps can be found in Figure 6.12.


velocity Magnitude
0.000e+00 2.1 4.1 6.2 8.204e+00
velocity Magnitude
0.000e+00 2.1 4.1 6.2 8.259e+00

(a) Time step 50.

(b) Time step 1200.

Figure 6.12: The streamline of the mean value at the time step 50 and 1200.



(a) Velocity in $\boldsymbol{x}$ (m/s).

(b) Velocity in $\boldsymbol{y}$ (m/s).

(c) Velocity in $\boldsymbol{z}$ (m/s).

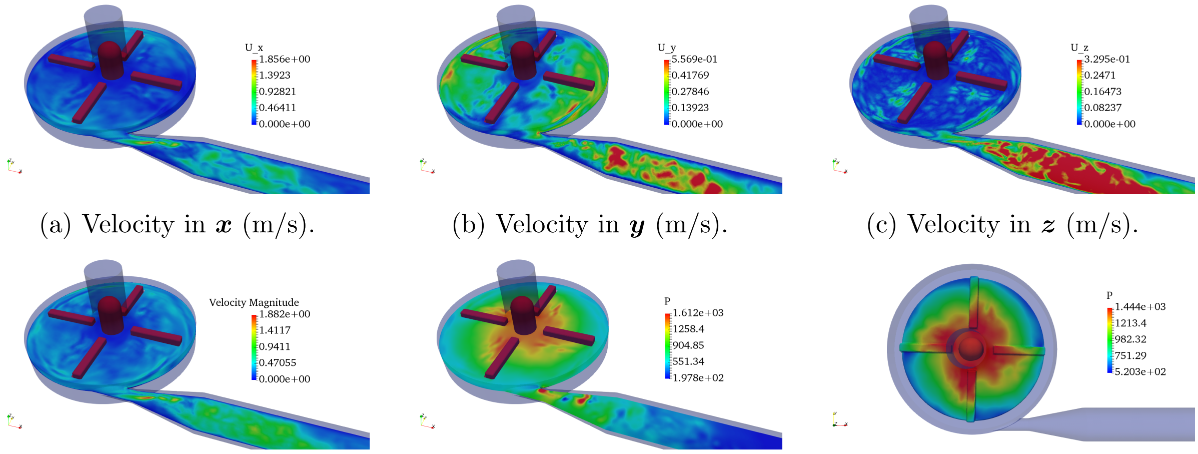
(d) Velocity magnitude (m/s).

(e) Pressure (Pa).

(f) Pressure on rotor (Pa).

Figure 6.13: Standard deviation at the time step 50.

Regarding the standard deviation, at the time step 50, the uncertainties are mainly located in the housing (Figure 6.13), it is comparable to the situation for the mean value, where the high velocity and the high pressure occur mainly in the housing. However, the distribution of the standard deviation does not follow the mean value's allocation, particularly for the pressure (Figure 6.13e). At the time step 1200, the uncertainties are more located in the area around the throat and the diffuser (Figure 6.14), the standard deviation is smaller in the chamber. In contrast to the previous example, the pressure distribution (Figure 6.14e) corresponds mostly to the pressure's mean value (Figure 6.11e), it implies that the standard deviation becomes larger

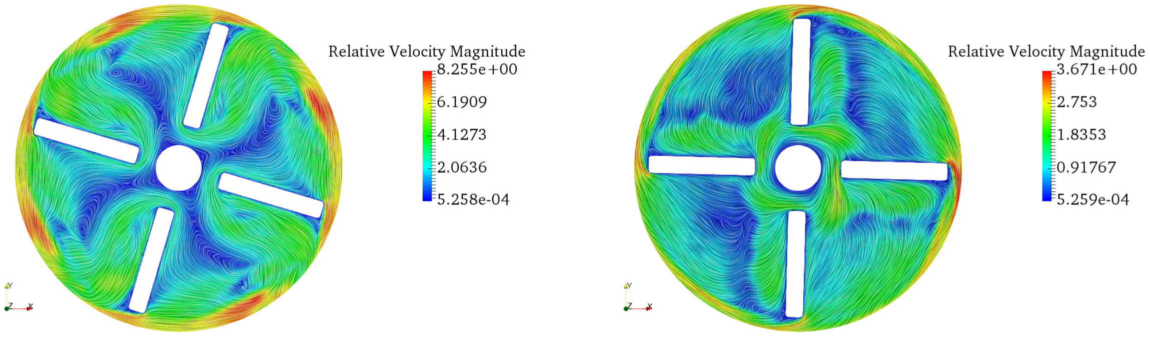


(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

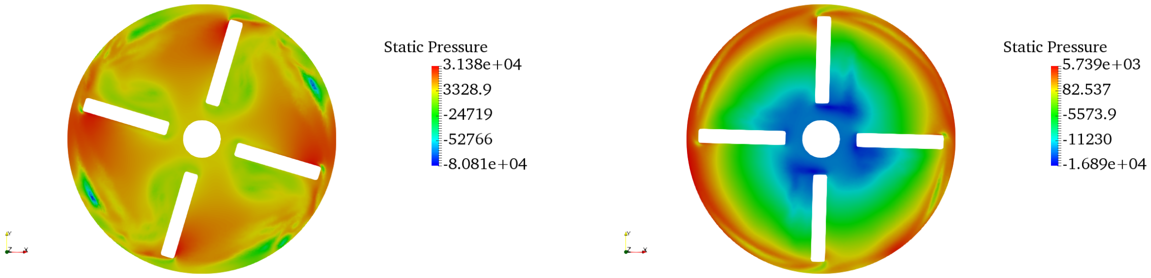
Figure 6.14: Standard deviation at the time step 1200.

where the pressure magnitude is high.



(a) Time step 50. (b) Time step 1200.

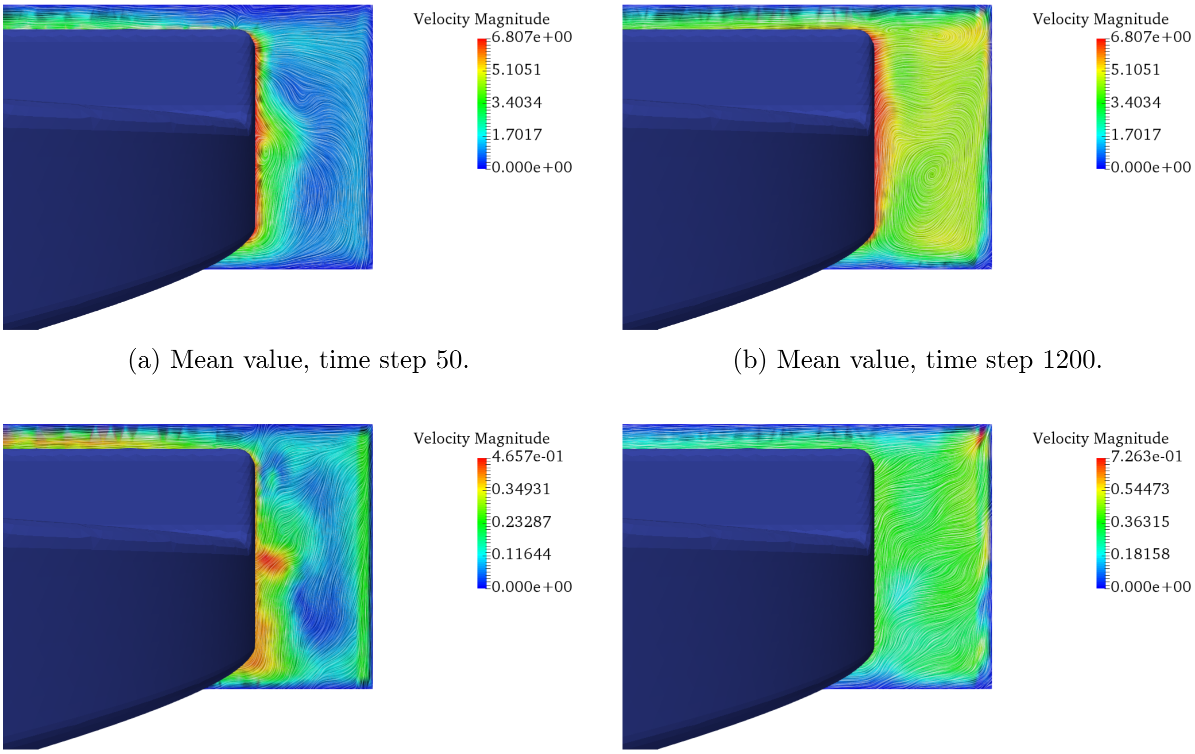
Figure 6.15: The mean value of the relative velocity (m/s) at the time step 50 and 1200.



(a) Time step 50. (b) Time step 1200.

Figure 6.16: The mean value of the static pressure (Pa) at the time step 50 and 1200.

Figure 6.15 shows the relative velocity in the region around the rotor. The eddies occur in the area between two blades for both cases. In Figure 6.15a, one big eddy can be found in each flow passage, they are roughly axisymmetric. At the time step 1200, only small eddies appear in three passages. The passage, which is connected to the outlet, has basically no eddies occurred due to the outflow. An analogous separation of the flow is observed in both examples around the trailing edge as described previously in Figure 6.11d. Furthermore, the jet-wake structure



(a) Mean value, time step 50.

(b) Mean value, time step 1200.

(c) Standard deviation, time step 50.

(d) Standard deviation, time step 1200.

Figure 6.17: The streamline (mean value and standard deviation) of the velocity (m/s) at the time step 50 and 1200 in the casing.

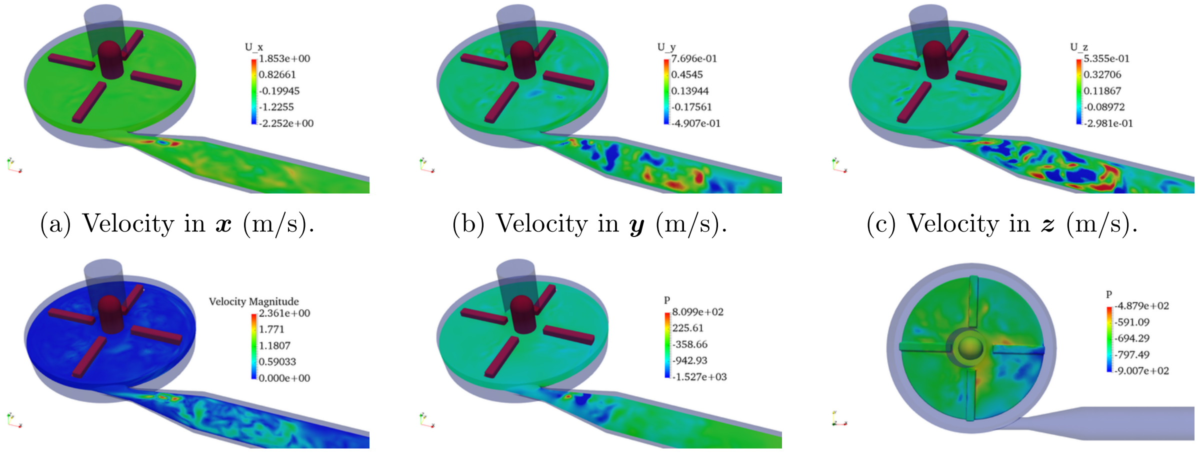
[31] is clearly noticeable in each passage as well. This occurrence can be strongly disturbed by the backflow in the passage that is linked to the outlet.

In Figure 6.16, the static pressure (or hydrostatic pressure) around the rotor is illustrated. The static pressure represents in general the resistance to the fluid and can be obtained with Equation (6.21):

$$p_t := p_s + \frac{1}{2}\rho \boldsymbol{u}^2 \; . \tag{6.21}$$

Here, p_t is the total pressure ($p_t = p$), p_s is the static pressure, and $\frac{1}{2}\rho\boldsymbol{u}^2$ is the dynamic pressure which is related to the kinetic energy. The static pressure increases progressively from the inner side to the outer side along the radial direction, the pressure drop also happens in the areas right after the outer side of the blades following the rotational direction. Furthermore, as the static pressure is essentially calculated with the velocity and the total pressure (Equation (6.21)), the standard deviation of the velocity and the pressure contribute together to the uncertainty of the static pressure.

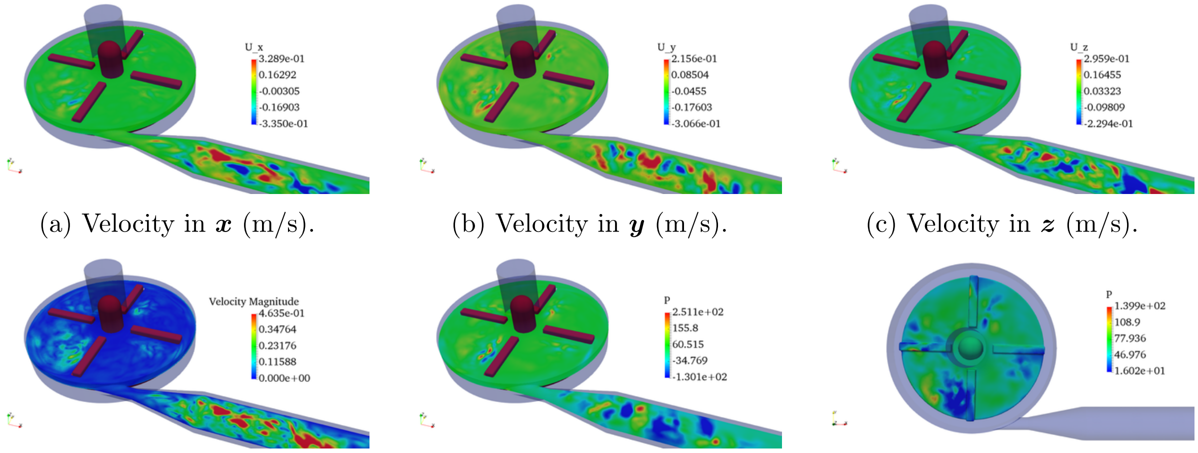
The blood flow enters into the casing after being discharged from the impellers, vortices are formed as a consequence of a strong flow recirculation in this area (Figures 6.17a and 6.17b). At the time step 1200, the lower vortex is more dominant than the upper one, it can become even more prominent once this structure approaches to the throat. Besides, these vortices and the backflow provoke a pressure fluctuation, which indicates an energy loss in the outlet. Contrariwise, the standard deviation (Figures 6.17c and 6.17d) does not possess any clear vortex structure, yet the uncertainty is higher in the region where the shearing flow is developed.



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

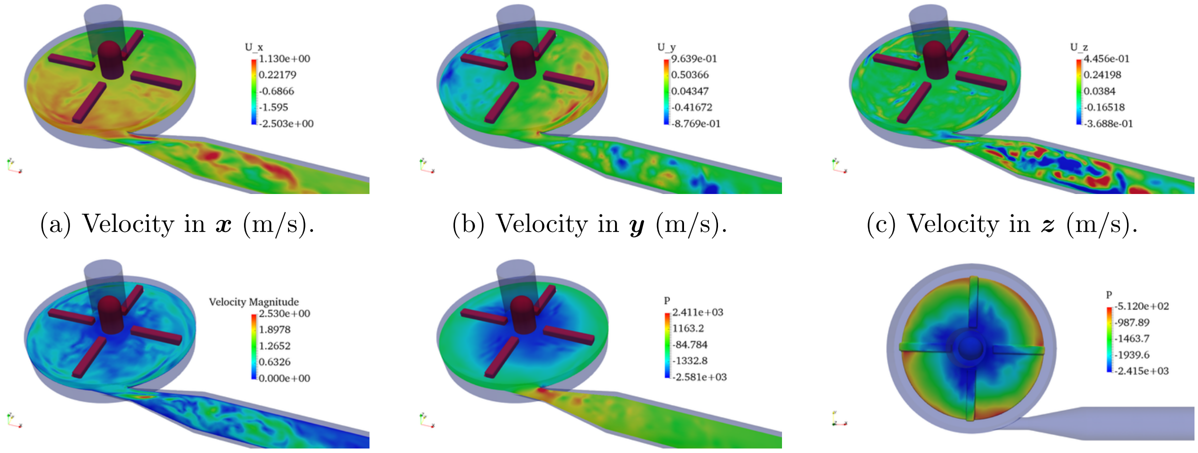
Figure 6.18: Stochastic mode 1 at the time step 1200.



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.19: Stochastic mode 2 at the time step 1200.



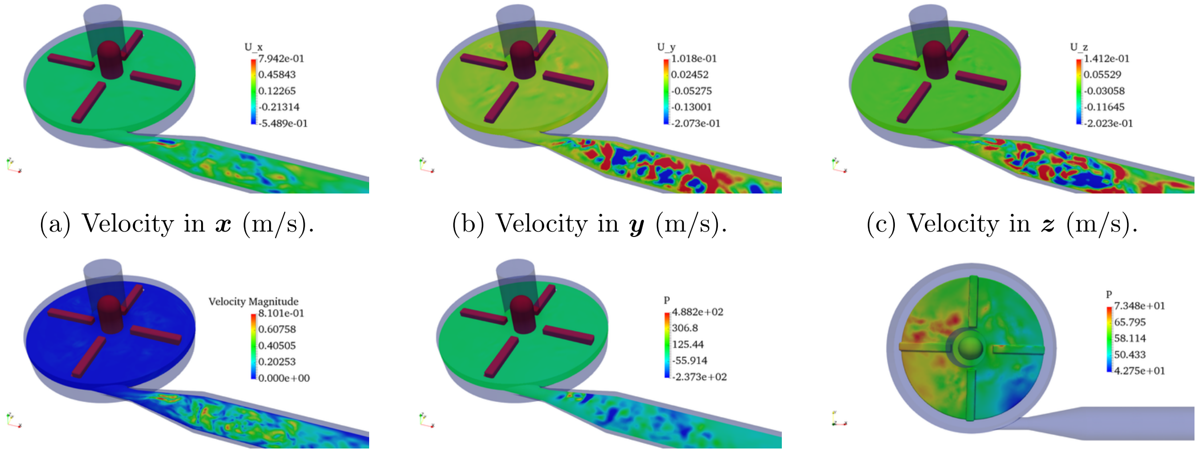
(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.20: Stochastic mode 3 at the time step 1200.

Stochastic Modes

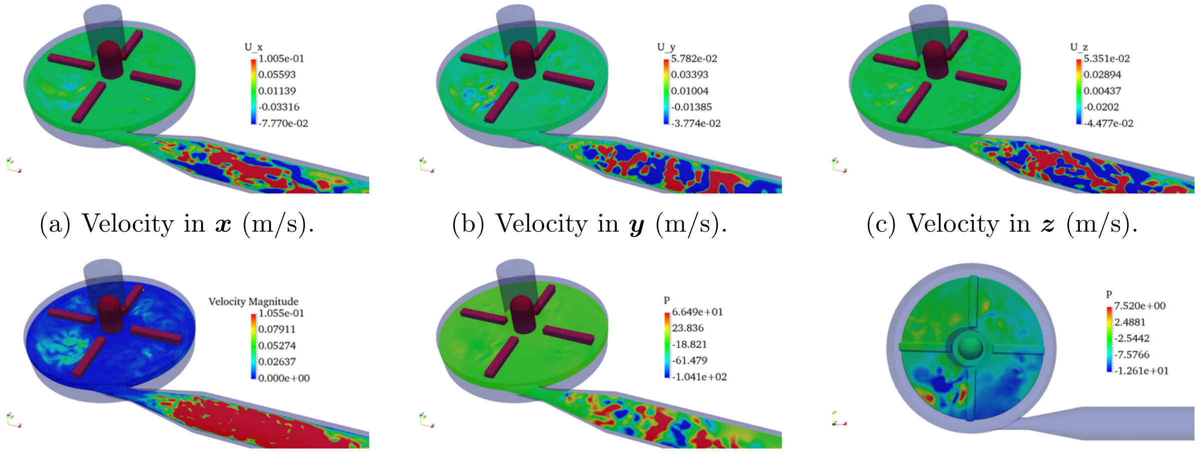
As introduced in Chapter 3, our discrete stochastic solutions are expressed by using the Polynomial Chaos expansion. A system of coupled equations is formed by using the Galerkin



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (Pa). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

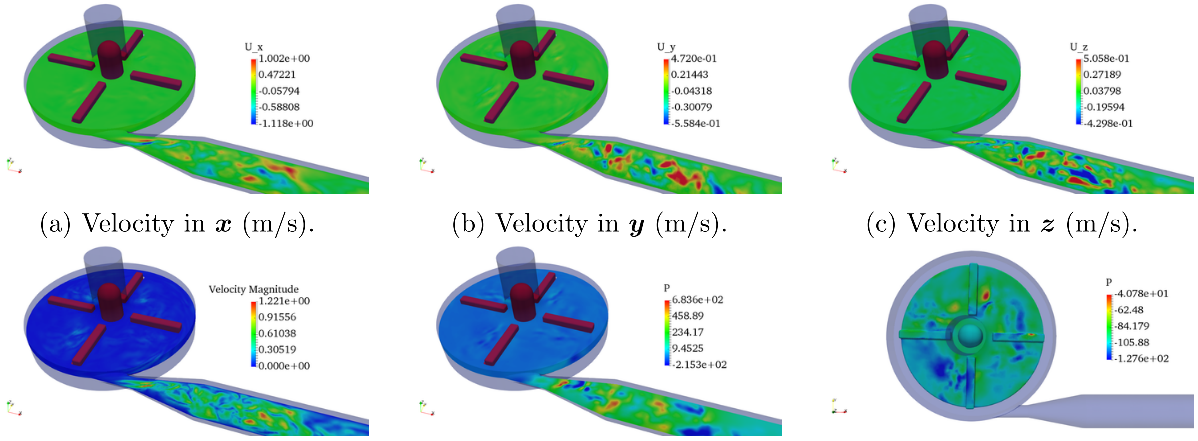
Figure 6.21: Stochastic mode 4 at the time step 1200.



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.22: Stochastic mode 5 at the time step 1200.

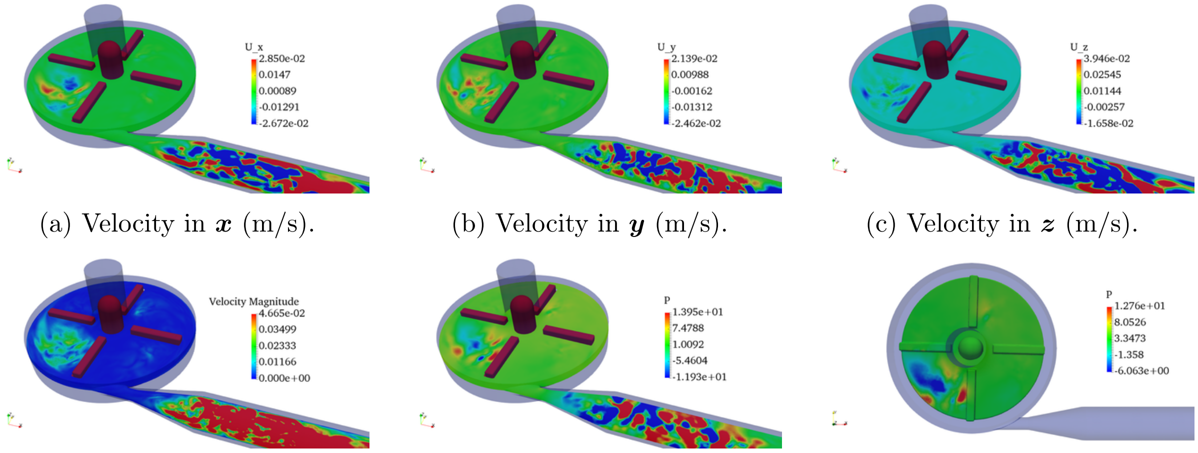


(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.23: Stochastic mode 6 at the time step 1200.

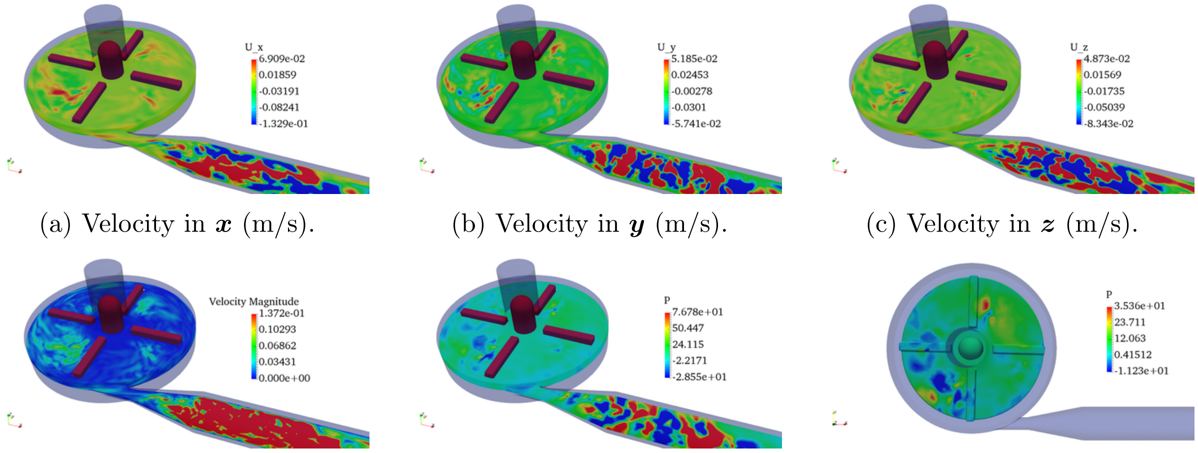
projection as described in Chapter 3. Accordingly, the stochastic solution of the velocity $\boldsymbol{u}_h$ and the pressure p_h can be written by using $(P+1)$ Chaos Polynomials as follows:



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

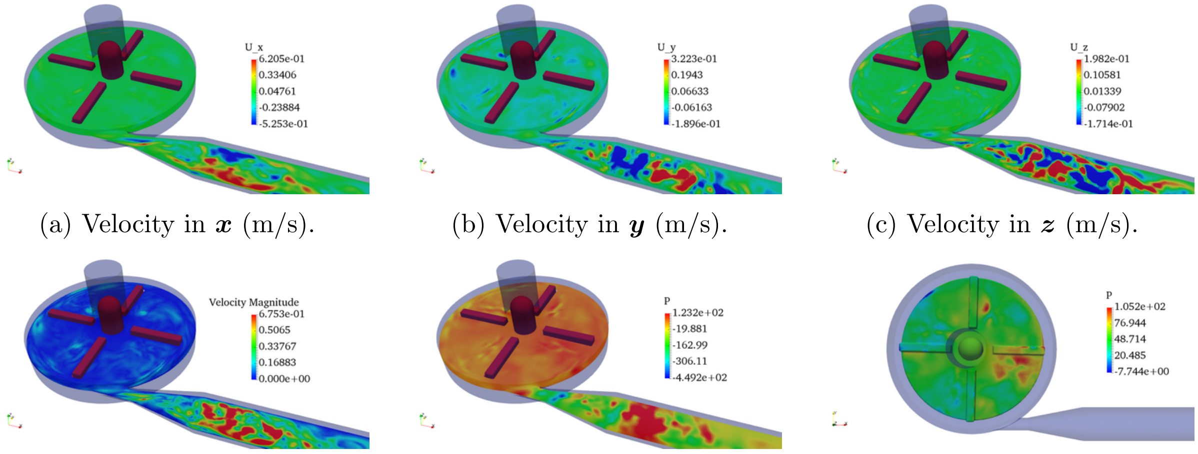
Figure 6.24: Stochastic mode 7 at the time step 1200.



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.25: Stochastic mode 8 at the time step 1200.

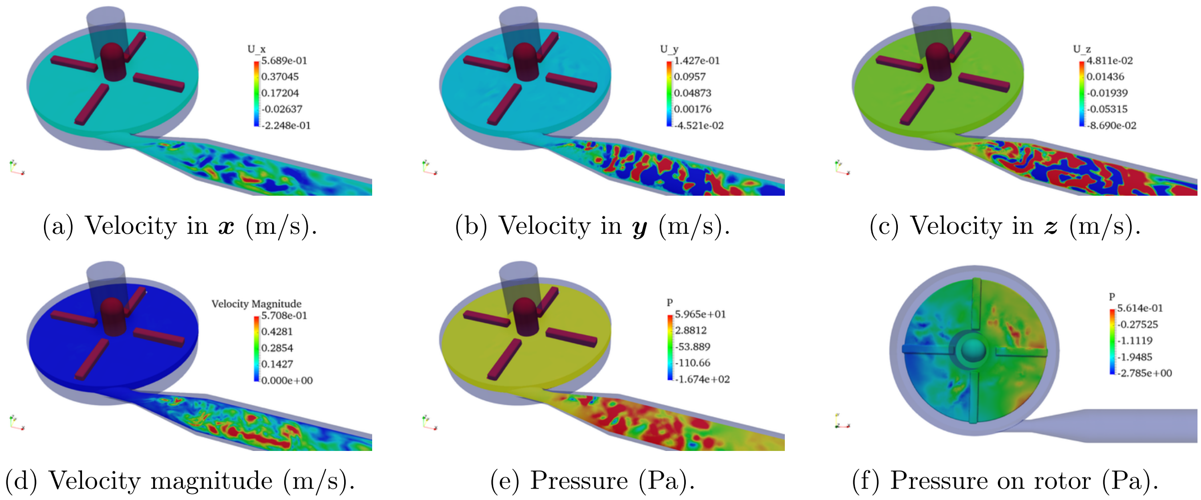


(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.26: Stochastic mode 9 at the time step 1200.

$$\boldsymbol{u}_h(\boldsymbol{\xi}) = \sum_{k=0}^{P} \boldsymbol{u}_{h,k}\psi_k(\boldsymbol{\xi}) \ , \quad p_h(\boldsymbol{\xi}) = \sum_{k=0}^{P} p_{h,k}\psi_k(\boldsymbol{\xi}) \ . \tag{6.22}$$



(a) Velocity in $\boldsymbol{x}$ (m/s).

(b) Velocity in $\boldsymbol{y}$ (m/s).

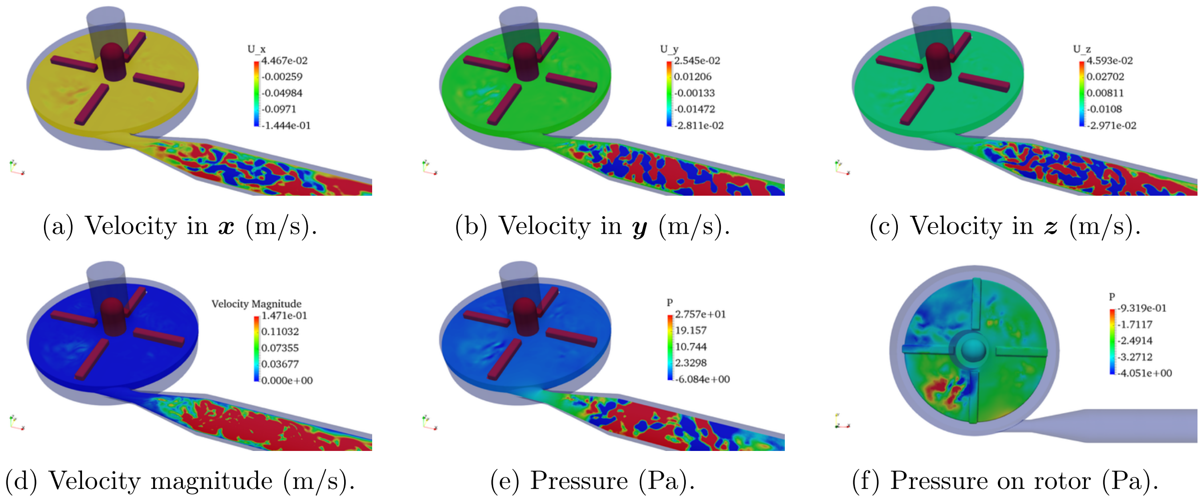
(c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s).

(e) Pressure (Pa).

(f) Pressure on rotor (Pa).

Figure 6.27: Stochastic mode 10 at the time step 1200.



(a) Velocity in $\boldsymbol{x}$ (m/s).

(b) Velocity in $\boldsymbol{y}$ (m/s).

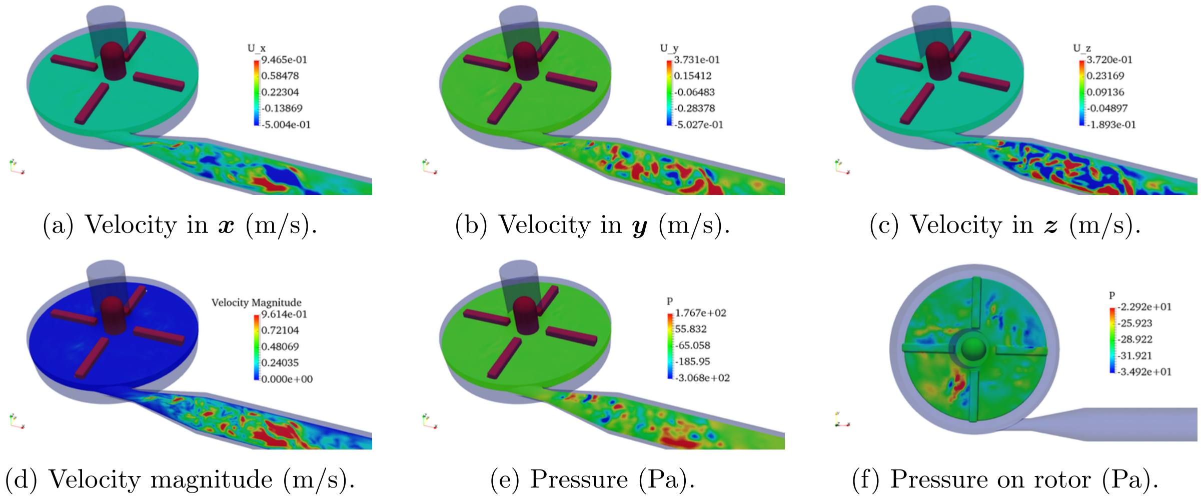
(c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s).

(e) Pressure (Pa).

(f) Pressure on rotor (Pa).

Figure 6.28: Stochastic mode 11 at the time step 1200.



(a) Velocity in $\boldsymbol{x}$ (m/s).

(b) Velocity in $\boldsymbol{y}$ (m/s).

(c) Velocity in $\boldsymbol{z}$ (m/s).

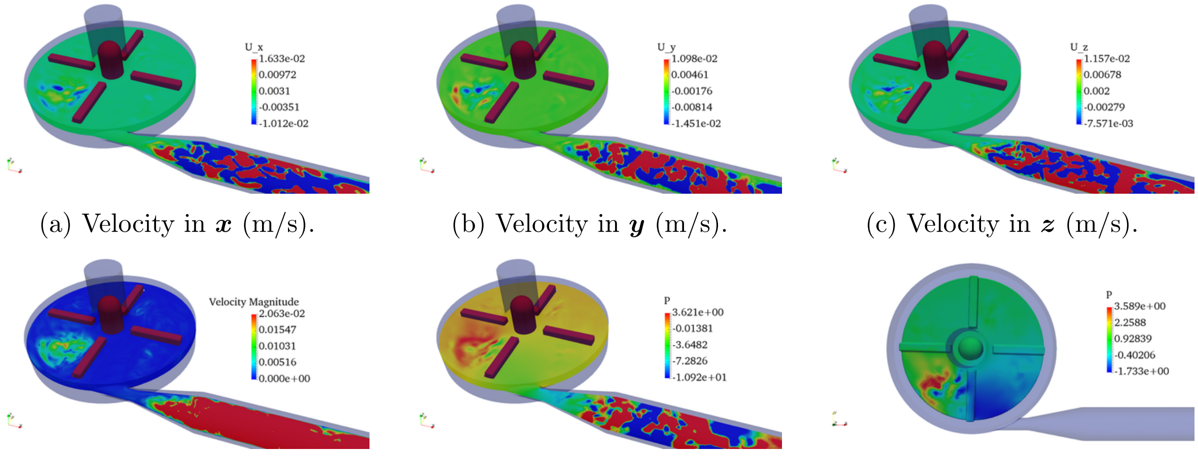
(d) Velocity magnitude (m/s).

(e) Pressure (Pa).

(f) Pressure on rotor (Pa).

Figure 6.29: Stochastic mode 12 at the time step 1200.

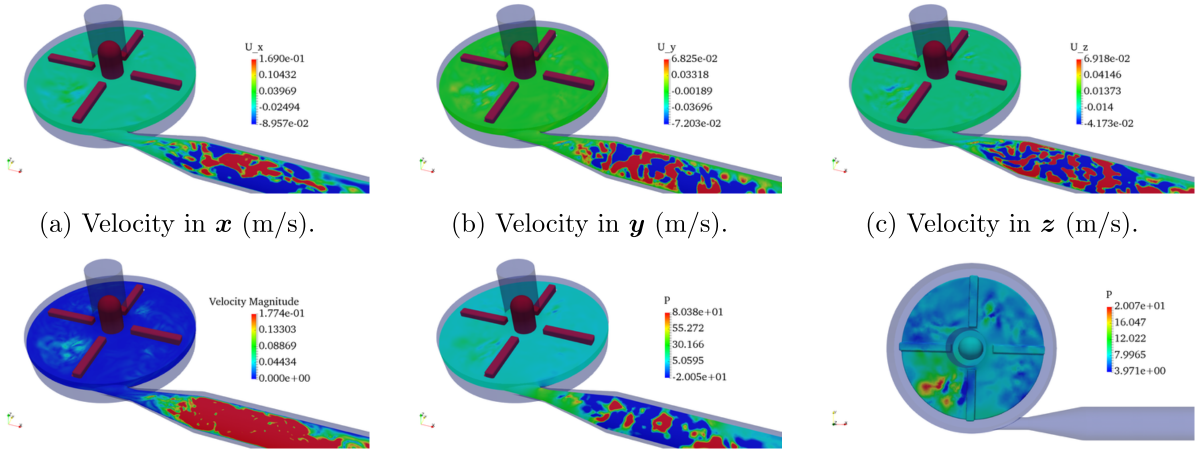
Here, $\boldsymbol{u}_{h,k}, p_{h,k}, k = 0, ..., P$ are the stochastic mode solutions, the individual mode solutions (the zero-th mode solution is the mean value) at the time step 1200 are shown in Figures 6.18 to 6.36.



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

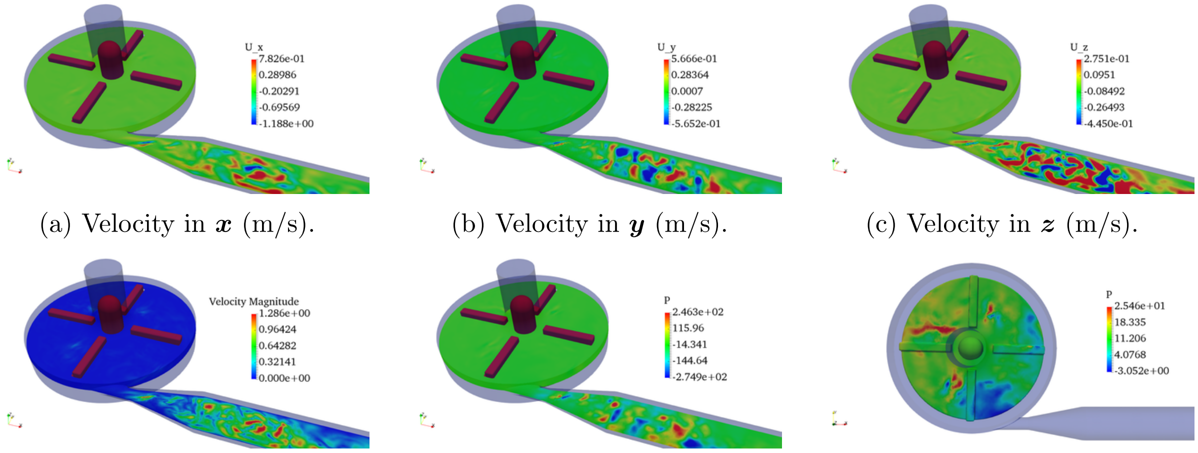
Figure 6.30: Stochastic mode 13 at the time step 1200.



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.31: Stochastic mode 14 at the time step 1200.

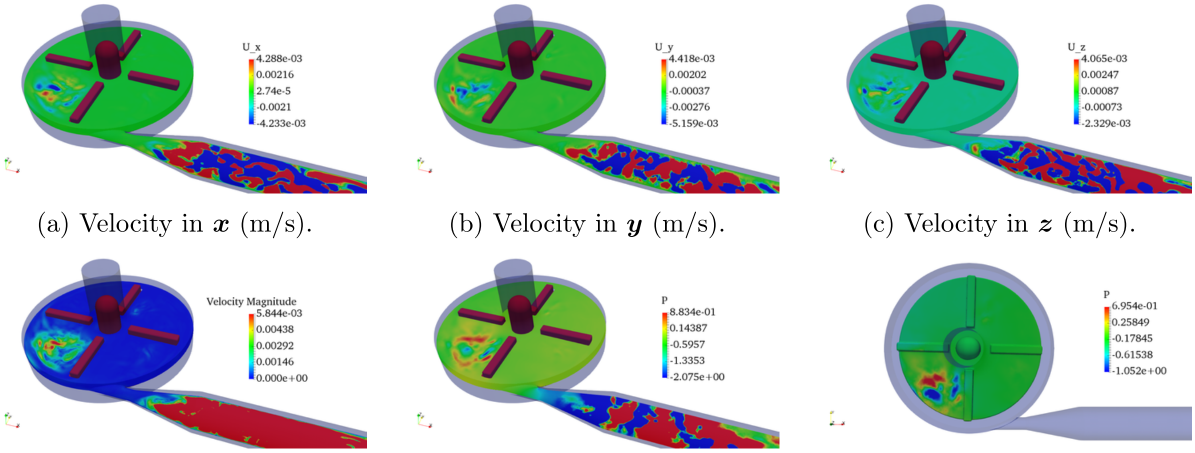


(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.32: Stochastic mode 15 at the time step 1200.

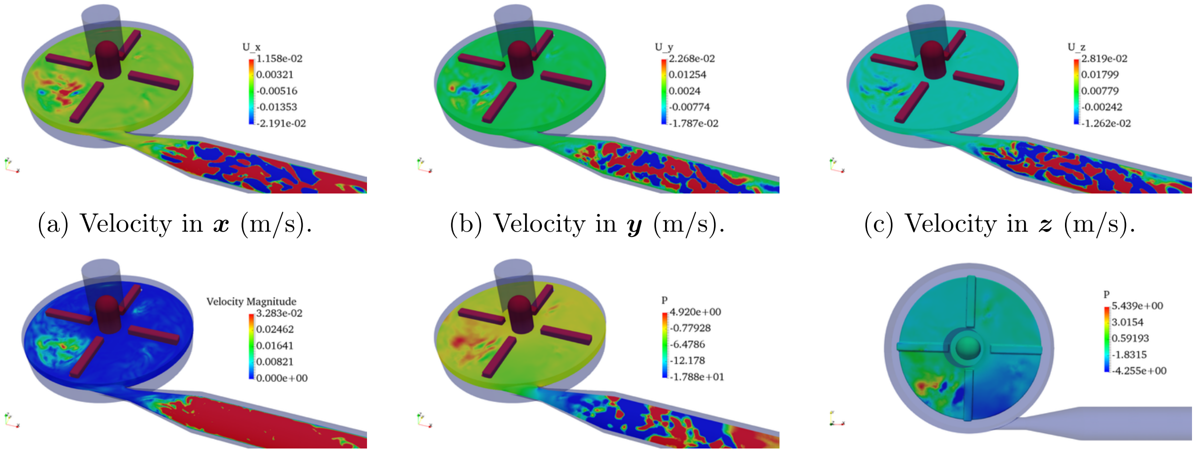
The first order mode solutions are shown in Figures 6.18 to 6.20, they correspond to the polynomials $\psi_k, k = 1, ..., 3$. These three stochastic modal solutions have mostly the same im-



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

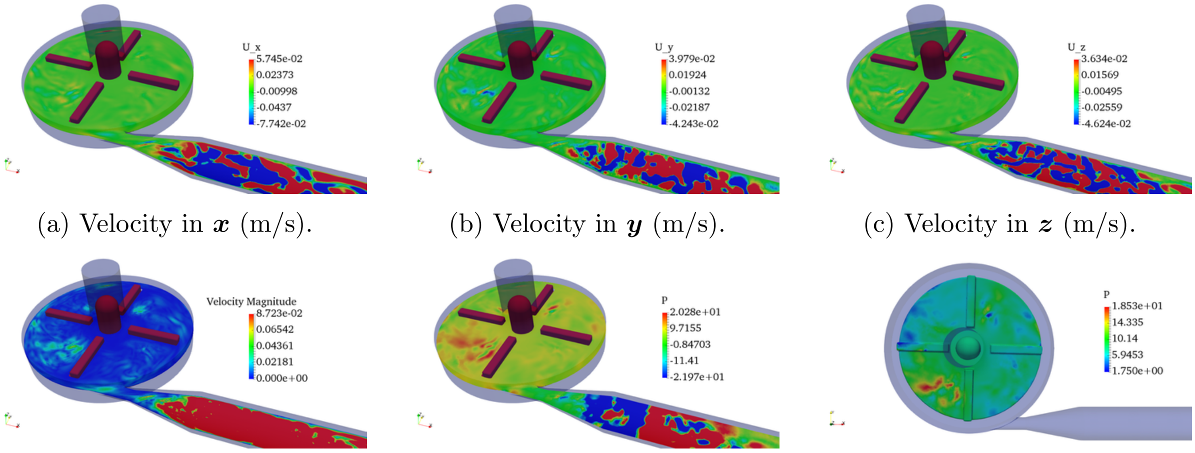
Figure 6.33: Stochastic mode 16 at the time step 1200.



(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.34: Stochastic mode 17 at the time step 1200.

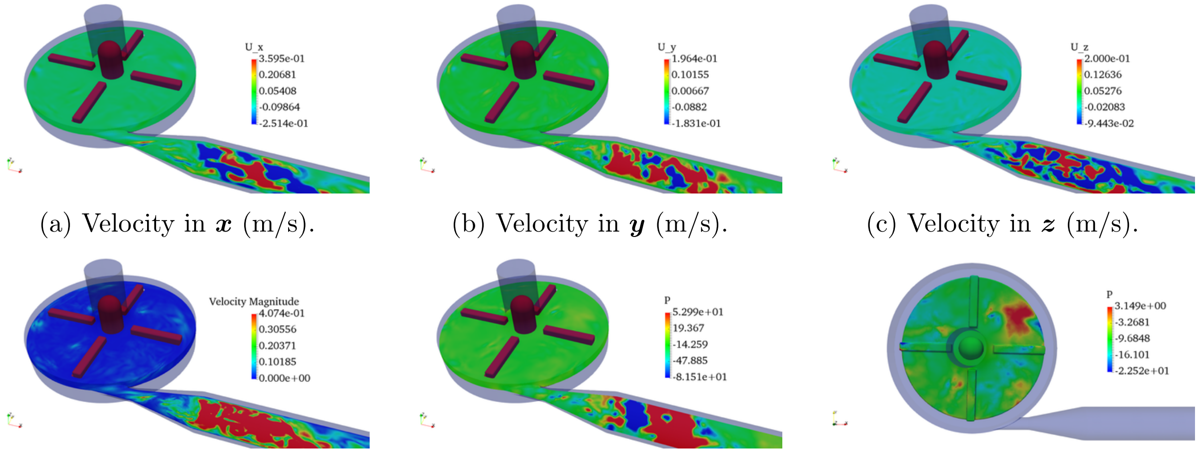


(a) Velocity in $\boldsymbol{x}$ (m/s). (b) Velocity in $\boldsymbol{y}$ (m/s). (c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s). (e) Pressure (Pa). (f) Pressure on rotor (Pa).

Figure 6.35: Stochastic mode 18 at the time step 1200.

portance in terms of the contribution to the standard deviation. Referring to Equation (6.13), the uncertain sources are assigned to three Uniform variables in the following order:



(a) Velocity in $\boldsymbol{x}$ (m/s).

(b) Velocity in $\boldsymbol{y}$ (m/s).

(c) Velocity in $\boldsymbol{z}$ (m/s).

(d) Velocity magnitude (m/s).

(e) Pressure (Pa).

(f) Pressure on rotor (Pa).

Figure 6.36: Stochastic mode 19 at the time step 1200.

- Inflow boundary condition $(\boldsymbol{u}_I) \sim \xi_1$.
- Dynamic viscosity $(\mu) \sim \xi_2$.
- Angular speed $(\boldsymbol{\omega}) \sim \xi_3$.

Note that, in regard to our multidimensional Legendre polynomials ($M = 3$), there is only one polynomial for the zero-th polynomial, viz. $\psi_0(\boldsymbol{\xi}) = 1$. Accordingly, the first order polynomials are only dependent on each particular random variable. They are defined as:

$$\psi_1(\boldsymbol{\xi}) = \xi_1 \ , \quad \psi_2(\boldsymbol{\xi}) = \xi_2 \ , \quad \psi_3(\boldsymbol{\xi}) = \xi_3 \ . \tag{6.23}$$

Hence, the first order stochastic mode solutions (Figures 6.18 to 6.20) are independent from each other with respect to the uncertain parameters ξ_1, ξ_2 and ξ_3. One can observe that the stochastic mode 3 (Figure 6.20), which is strongly related to the rotating speed, has a similar pattern as in the standard deviation (Figure 6.14). Notably for the pressure, the absolute value of the pressure for the PC mode 3 and the pressure standard deviation appear correspondingly in the same location. Besides, the velocity distribution has also a comparable appearance in both results. Also, the mode solutions 1 and 2 do not possess a clear common pattern from either the mean value or the standard deviation. The high velocity region appears in the same locations as the standard deviation. In other words, the strong fluctuation of the velocity happens mainly in the outlet area.

The second order stochastic modal solutions (Figures 6.21 to 6.26) correspond to the polynomials $\psi_k, k = 4, ..., 9$, they capture principally the dependence between the random variables pairwise. Yet, these results are difficult to interpret the physical meanings, because there are the nonlinearities of the dynamics in the flow and the nonlinearities between the stochastic modes. Still, the velocity distribution shows some chaotic behaviors among all figures, the velocity and the pressure are strongly correlated especially in the outlet region. Moreover, the absolute value of these results is one order of magnitude lower than the first order modal solutions, which demonstrates the spectral convergence of the gPCE for the stochastic moments.

The third order stochastic mode solutions are illustrated in Figures 6.27 to 6.36. They contribute less to the standard deviation comparing to the first order modal solutions, but they can enhance the accuracy of the results. Furthermore, it is unclear how to explain the physical meanings and the relations to the standard deviation. This is due to the complexity of our considered system.

Pressure Loading and Force on the Impeller

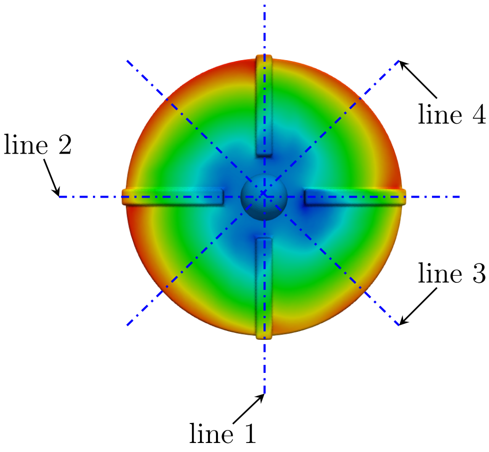
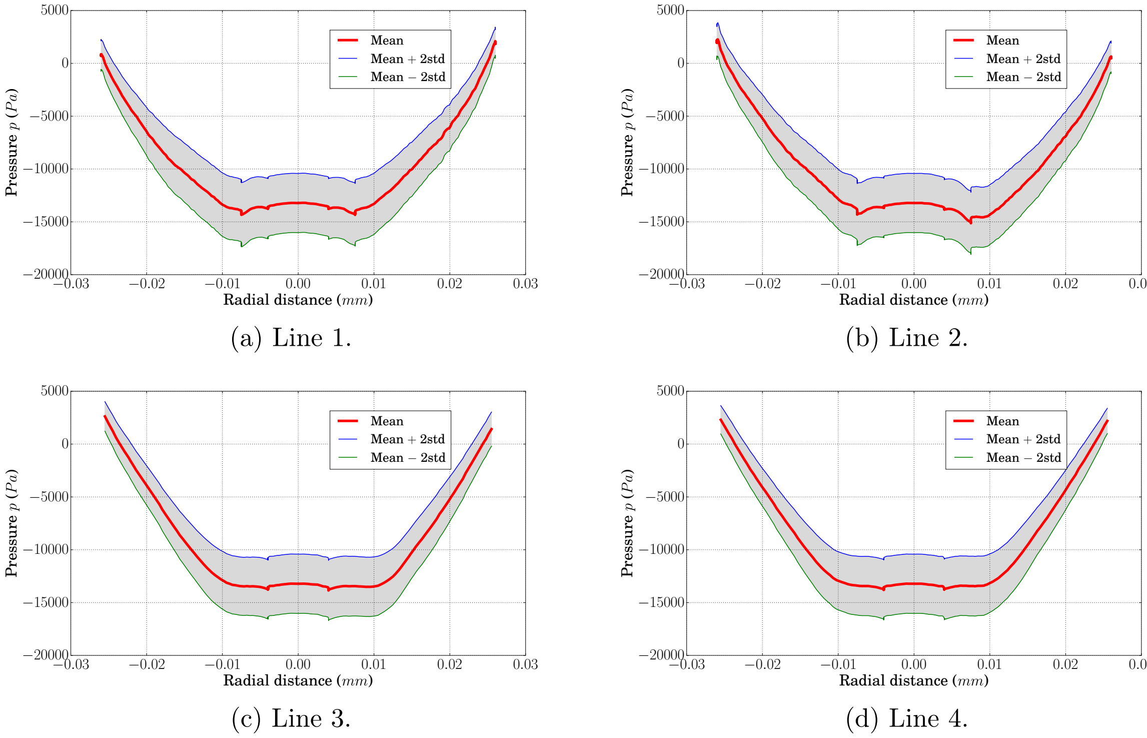


Figure 6.37: Illustration of the location for calculating the pressure loading (the rotor is colored by the pressure value at the time step 1200).

As the local pressure fluctuation on the blades are the main reason for material fatigue, which can lead to a mechanical fracture. Therefore, it is necessary to survey the pressure loading on the impeller and its uncertainties.



(a) Line 1.

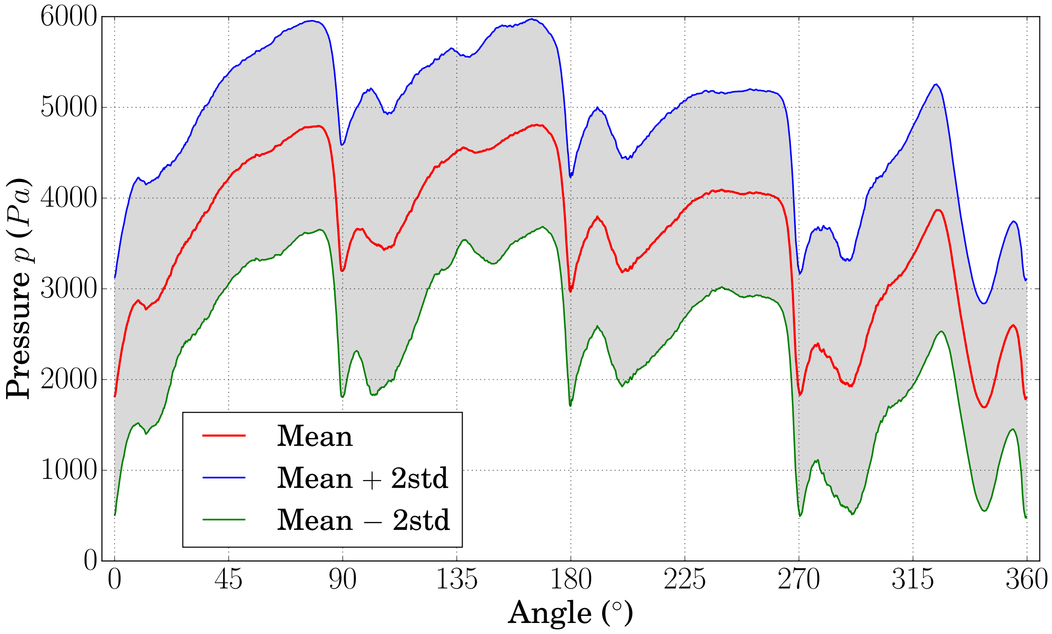
(b) Line 2.

(c) Line 3.

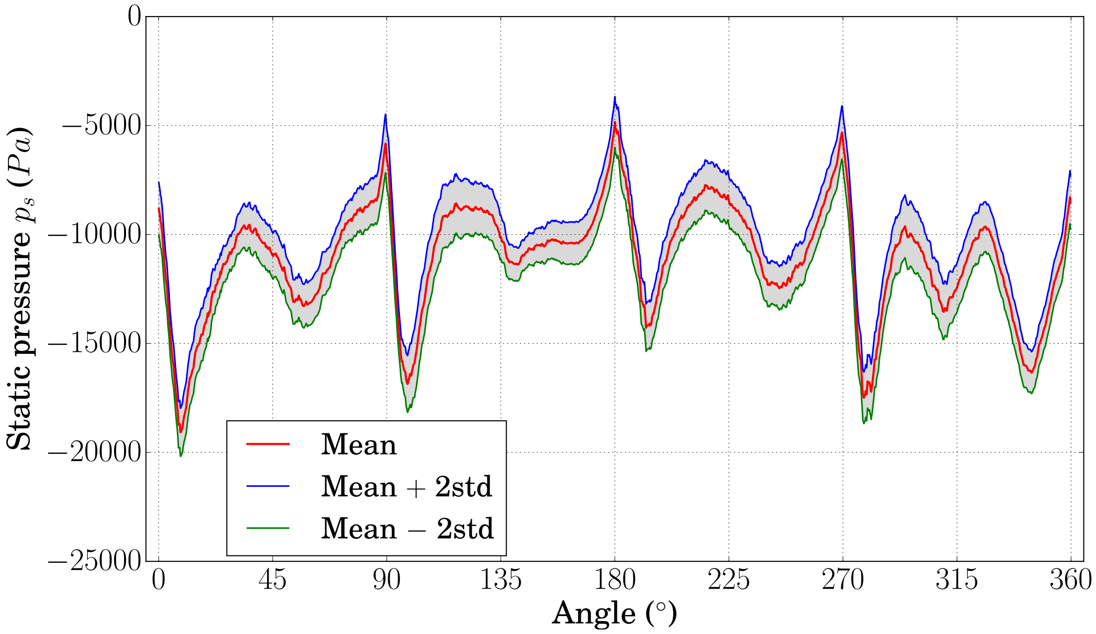
(d) Line 4.

Figure 6.38: Pressure loading with twice the standard deviation confidence interval at the time step 1200.

Figure 6.37 demonstrates four locations, which are chosen on the upper surface of the impeller, for investigating the pressure loading. Line 1 and Line 2 are focused on the blades, Line 3 and Line 4 are concentrated in the rotor's surface. Additionally, the four lines all pass through the rotor hub. Similar to the previous subsection, we also choose the solution at the time step 1200 to illustrate the numerical results. One can observe from Figure 6.38 that the linear pressure loading increases steadily from the hub center (the middle) toward the rotor rims. The mean value (red) of the pressure loading is embedded into a confidence interval (gray) which is defined in a range of $(\bar{x}-2\sigma, \bar{x}+2\sigma)$, where $\bar{x}$ is the mean, σ is the standard deviation. The shape of these four plots are nearly indistinguishable. They correspond well to the pressure allocation on the rotor (Figure 6.37), the middle curves represent the disposition of the rub. Yet, the confidence interval of Line $1, 2$ is slightly larger than Line $3, 4$, the reason behind is that the velocity and the pressure are much more unstable on the trailing edges.

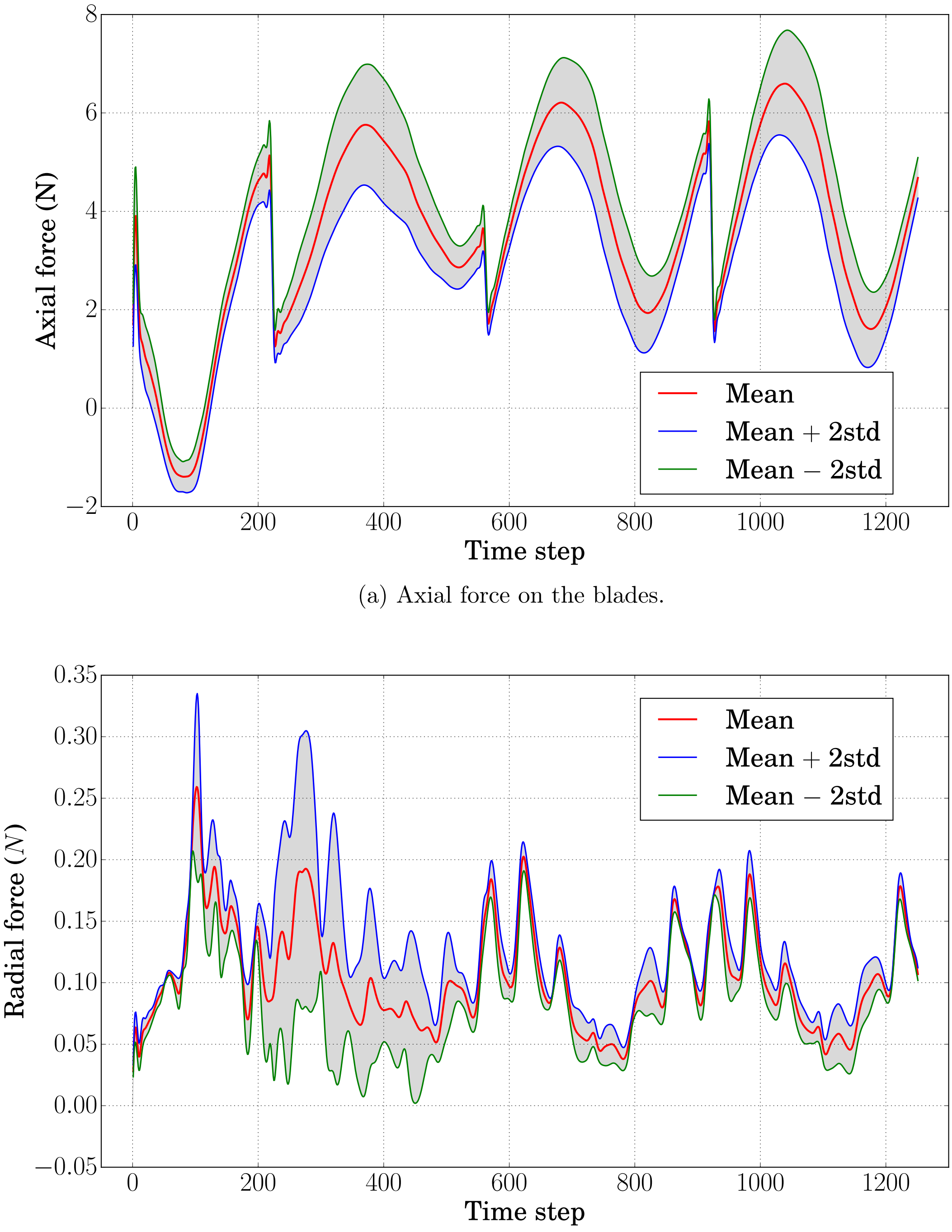


(a) Total pressure.



(b) Static pressure.

Figure 6.39: The total pressure and the static pressure on the periphery of the impeller at the time step 1200.



(a) Axial force on the blades.

(b) Radial force on the blades.

Figure 6.40: The axial force and the radial force on the blades.

The impeller's revolution results in a non-uniform pressure distribution at its periphery, Figure 6.39 shows the corresponding result at the time step 1200. The total pressure and the static pressure behave dissimilarly in terms of the shape, the range of the confidence interval, etc. Moreover, four pulsations can be found in both figures (the pulsations in Figure 6.39a are in the negative direction). One can observe that the size of the confidence interval of the total pressure is much larger than the confidence interval of the static pressure. We can also state that the velocity is much higher at every peak of the pulsation in Figure 6.39a than their neighboring region. On account of the fact that the static pressure is obtained by subtracting the dynamic pressure from the total pressure (Equation (6.21)), therefore the uncertainties are smaller.

The axial force is typically toward the suction, because a low pressure occurs on the upper surface of the rotor while the pump operates. The thrust bearings are normally selected based on the estimated axial forces. An incorrect bearing size can damage the pump, even induce a breakdown during the performance. Hence, it is very important to well predict the axial thrust in order to retain the impeller's balance as much as possible. Figure 6.40a displays the fluctuation of the axial force on the impeller by varying the time steps, and the mean value is bounded again with twice the standard deviation. In our computation, each revolution contains 240 time steps. We can then state that the axial force becomes negative only within the first revolution, whereas it is alway positive during the regular operation. This effect of the negative values can be occurred because the flow is not yet fully developed. One also observes that there is a rapid decrease in the magnitude of the axial force right before the rotor returns to the initial position, this effect is due to a discharge of fluid from the chamber to the outlet.

Analogously to the axial force, the radial force is another quantity for characterizing the dynamic working condition. It is mostly influenced by the rotating frequency and the volute tongue clearance. Figure 6.40b shows again the mean value and the related confidence interval. The fluctuation of the radial force is noticed to be more unstable than the axial force, and the radial force acts only in a quasi-periodic way. However, the periodic manner does not appear persuasively within each rotation, this can be explained with the strong impact of turbulence, thus the flow is highly unstable. Moreover, the standard deviation becomes relatively large within the second revolution, because the flow is not yet fully developed analogously to the situation of the negative value in Figure 6.40a. However, it is not very clear why this phenomenon can not be seemed during the first revolution. We state that the radial force is much smaller than the axial force, as well as the standard deviation. This implies that the axial force is more important in choosing the mechanical components of the rotor.

Hemolysis

The human blood consists mainly of the red blood cells (RBCs), the white blood cells (WBCs), the platelets and the plasma. The RBCs account for 45% of the blood volume and are responsible for transporting the oxygen from the lung to the living tissues with the help of the circulatory system. The WBCs constitute comparatively a small portion (1% of the total blood volume), they are the first defenders for our immune system. The platelets are the cell fragments of cytoplasm without the nuclei in the bone marrow, their main function is to stop bleeding by clotting blood vessels. The platelets can on the other hand cause clots which plug blood vessels, this effect can potentially induce a stroke. The plasma occupies 55% of the blood, 90% of the plasma is water, and the rest consists only of dissolved proteins.

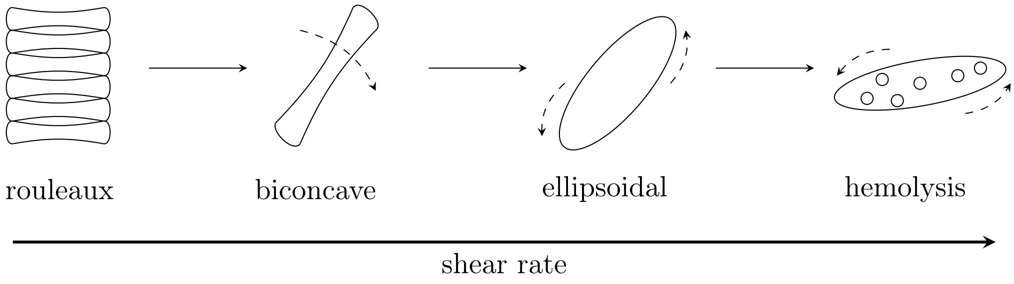


Figure 6.41: The development of RBCs under shearing from rouleaux to hemolysis.

When we study a blood pump, one important factor which always should be considered is the hemolysis. The hemolysis is commonly provoked by a sublethal shattering of RBCs. Under this condition, RBCs start being developed into the plasma and further losing the capability of transporting oxygen. Still, the sublethal damage does not occur directly under a stress but rather is a time-demanding process. If the shear rate of the blood increases, the RBCs can be deformed by the undergoing shear into different stages [22] (Figure 6.41).

When the plasma protein concentration is high, the RBCs transform into stacks, which are called the rouleaux (Figure 6.41). The rouleaux structure is disassembled due to the increase of the shear rate, but the RBCs can still preserve the biconcave shape and start tumbling in the flow if the shear stress is below 0.1 Pa [151]. Additionally, if the shear stress is above 1 Pa, the shape of RBCs is developed into an ellipsoidal form, whose major axis attempts to be aligned with the streamline of the flow. Therefore, the RBCs can be hemolyzing once the shear stress and the exposure time both surpass their critical values. As a consequence, the microscopic pores on the membrane open, the hemoglobin can penetrate into the blood. Furthermore, the release of the hemoglobin (ΔHb) into the plasma can be considered as a measurement for assessing the hemolysis.

There exist several clinical indices for estimating the hemolysis [42, 127, 181, 134], we employ the so-called index of hemolysis (IH) in this work. IH is defined as:

$$IH := (1 - \frac{Hct}{100})\frac{\Delta Hb}{Hb} \times 100 \ . \tag{6.24}$$

Here, Hct is the hematocrit in %, it represents the amount of RBCs in the blood. Hb is the total amount of hemoglobin, ΔHb is the released hemoglobin. However, this index is impractical to be calculated with the results obtained from the simulations, an empirical model is suggested by using the power law. It is given by:

$$\frac{\Delta Hb}{Hb} = A_{Hb}\sigma_s^{\alpha_{Hb}}\tilde{t}^{\beta_{Hb}} \ . \tag{6.25}$$

Equation (6.25) creates a relationship between the plasma-free hemoglobin (A_{Hb}), a scalar shear stress (σ_s) and the exposure time ($\tilde{t}$). Moreover, there are mainly two kinds of models, the stress-based hemolysis model [29, 63] and the strain-based hemolysis model [8], for estimating the blood damage. We consider the stress-based version in this work, but the strain-based approach can also be computed in a systematic way by using the same data obtained from our computation. The major difference between the these two approaches is how to define the scalar shear stress. For the stress-based approach, the stress parameter σ_s is defined as:

$$\sigma_s := \mu G_f = \mu\sqrt{2\boldsymbol{\varepsilon} : \boldsymbol{\varepsilon}} \ , \tag{6.26}$$

where, G_f is the shear rate of the flow field, $\boldsymbol{\varepsilon}$ is the rate of deformation tensor. $\boldsymbol{\varepsilon}$ is defined as follows:

$$\boldsymbol{\varepsilon} := \frac{1}{2}(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T) \ . \tag{6.27}$$

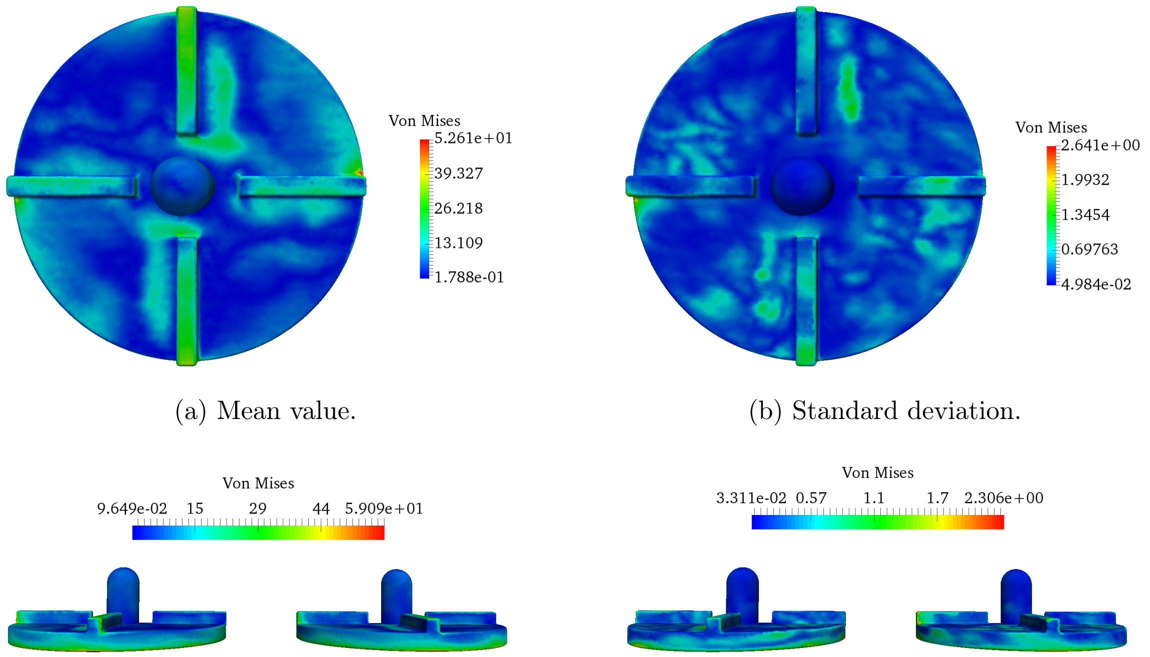
Sometimes, the von Mises criterion [126] is employed in the place of σ_s in Equation (6.25) [173, 10]. Although, it is not the case for this work, it is still interesting to show the von Mises stress distribution on the rotor, as this criterion is often referred in engineering applications. Thence, we first state the formulation of the von Mises criterion:

$$\sigma_{vm} := \sqrt{\frac{1}{6}((\sigma_{xx} - \sigma_{yy})^2 + (\sigma_{yy} - \sigma_{zz})^2 + (\sigma_{zz} - \sigma_{xx})^2) + \sigma_{xy}^2 + \sigma_{yz}^2 + \sigma_{zx}^2} \ . \tag{6.28}$$

Here, $\boldsymbol{\sigma}$ is the stress tensor of the fluid:

$$\boldsymbol{\sigma} := -p\delta_{ij} + 2\mu\boldsymbol{\varepsilon} \ , \tag{6.29}$$

where $\boldsymbol{\varepsilon}$ is again the rate of deformation tensor as in Equation (6.27).



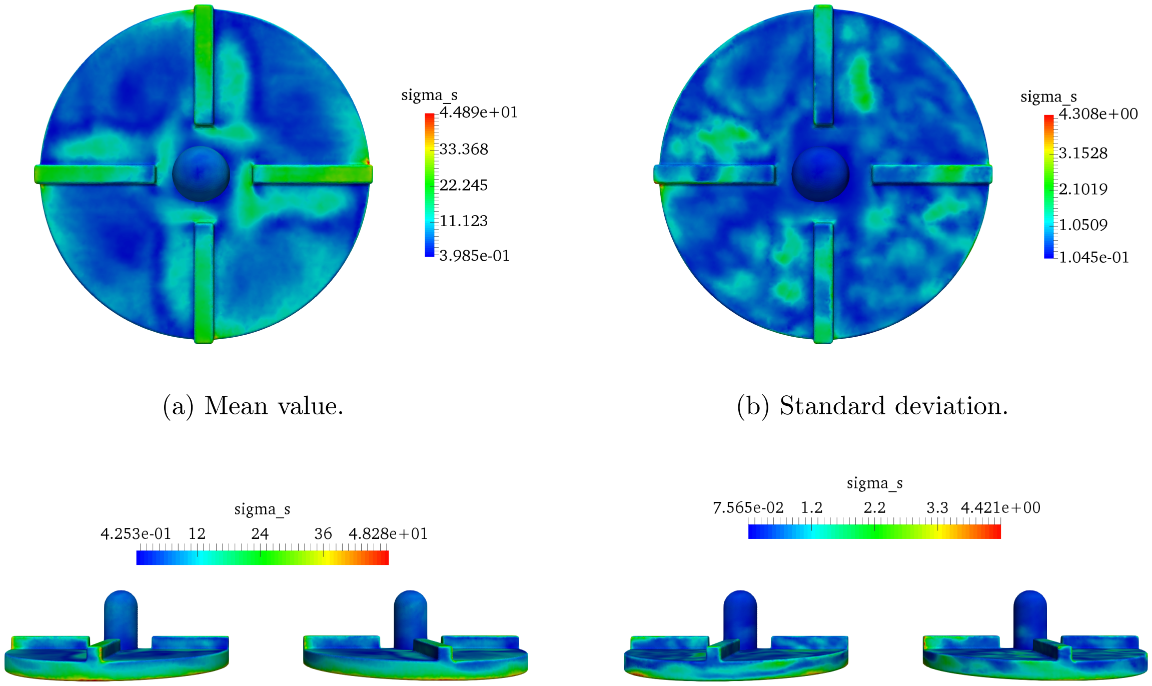
(a) Mean value.

(b) Standard deviation.

(c) Mean value.

(d) Standard deviation.

Figure 6.42: The mean value and the standard deviation of the von Mises stress distribution at the time step 1200 on the rotor.



(a) Mean value.

(b) Standard deviation.

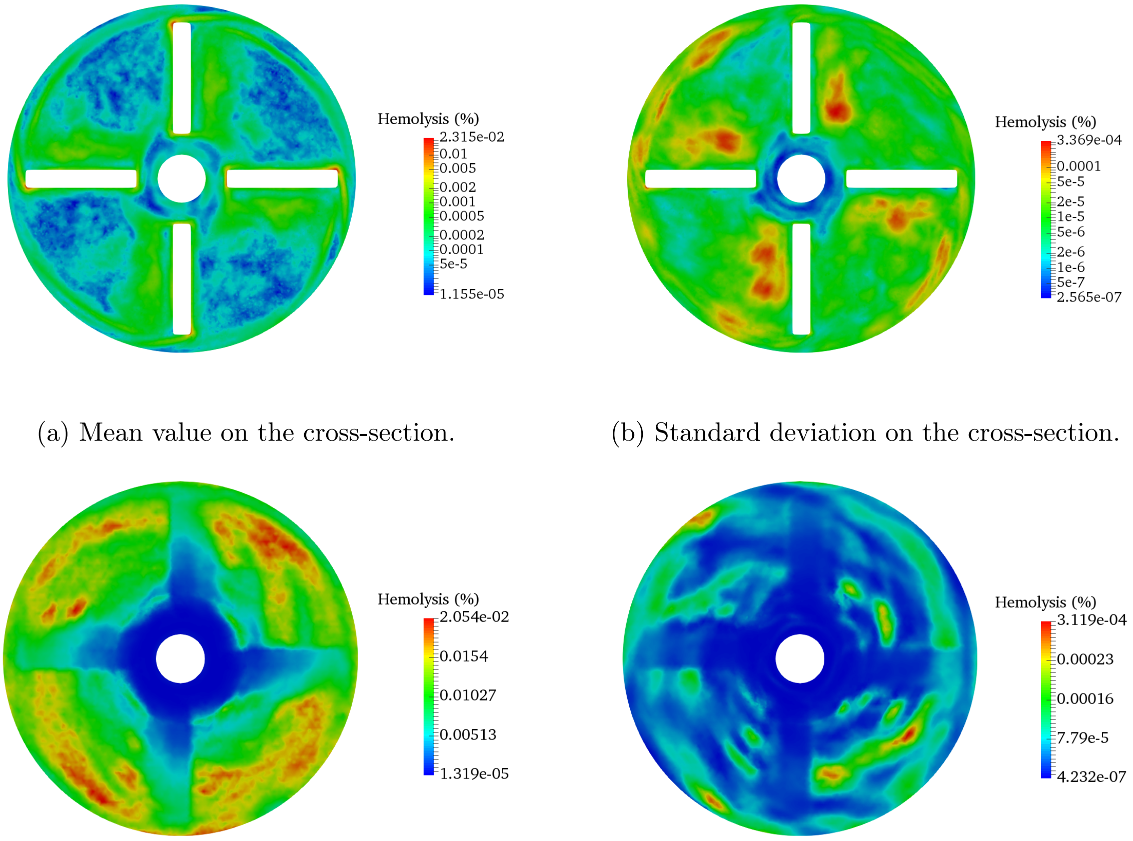
(c) Mean value.

(d) Standard deviation.

Figure 6.43: The mean value and the standard deviation of σ_s distribution at the time step 1200 on the rotor.

On the upper surface of the rotor, we observe that the mean value of the von Mises criterion is rather located around the blades, especially on the trailing edges and the area after the blades following the rotation direction (Figure 6.42a). However, the standard deviation varies over the rotor irregularly and is one order of magnitude lower than the mean solution, which is similar to the situation in Figures 6.11 and 6.14. The mean value and the standard deviation on the outer edges (Figures 6.42c and 6.42d) have a similar behavior, by this means, they both (the mean value and the standard deviation) retain the high value of σ_{vm}. More particularly, their highest values are all located on the lower edge.

Before presenting the hemolysis in the pump chamber, we provide the distribution of the scalar shear stress σ_s (Equation (6.26)) on the rotor's surface. As the von Mises criterion can play a similar role in the estimation of the hemolysis, e.g. in [10], σ_s (Figures 6.43a and 6.43c) should act in a similar way on the rotor comparing to Figures 6.42a and 6.42c. In other words, the order of the magnitude and the distribution on the upper surface should behave similarly. Figure 6.43b shows that the standard deviation of σ_s has also an irregular distribution as in the previous plot for the von Mises stress, even so, its value pursues more distinct the distribution of the mean value. In spite of the fact that the mean value of σ_s is smaller in comparison with the von Mises criterion, the standard deviation seems to be estimated twice larger. Moreover, the mean value and the standard deviation on the outer edge (Figures 6.43c and 6.43d) have a comparable distribution as in the case of the von Mises stress. Besides, we also notice from our numerical results that σ_s is very high in the region which is next to the outlet, whereas this effect is not marked by the von Mises stress.



(a) Mean value on the cross-section.

(b) Standard deviation on the cross-section.

(c) Mean value on the upper wall.

(d) Standard deviation on the upper wall.

Figure 6.44: The mean value and the standard deviation of IH distribution at the time step 1200.

Finally, the free plasma hemoglobin ratio is calculated with Equation (6.25). We consider the model in [181]. The coefficients are defined as follows:

$$A_{Hb} = 1.228 \times 10^{-5} \ , \quad \alpha_{Hb} = 1.9918 \ , \quad \beta_{Hb} = 0.6606 \ . \tag{6.30}$$

In addition, based on the information provided by the FDA [55], the hematocrit is equal to 36%, the hemolysis index can be thus computed with Equation (6.24).

We present the hemolysis index at two different locations (Figure 6.44), one is on the cross-section ($z = 6.5\,\mathrm{mm}$ from the bottom of the pump chamber), another one is on the upper wall of the pump chamber. We observe that the hemolysis is very high around the outer fringe of the trailing edges on the cross-section, as well as in the surroundings of the blades (Figure 6.44a). On the other hand, the index of hemolysis on the upper wall is much higher in the area between the blades (Figure 6.44c), and the highest concentrations are close to 0.02%. Regarding the standard deviations (Figures 6.44b and 6.44d), both cases follow a similar pattern as their mean value, which means that the IH increases approximately where the mean values become important. Besides, unlike in Figure 6.43, the standard deviation of the index of hemolysis is almost two magnitudes lower with respect to the mean value, it implies that the confidence interval is rather narrow (only a few percentages with respect to the mean value).

Chapter 7

Conclusion

The aim of this thesis is devoted to the numerical simulation of the flow in a blood pump device assuming uncertainty quantification. In order to achieve this goal, we have to overcome several difficulties: modeling the high Reynolds number flow; modeling the moving mesh effect; deploying the generalized Polynomial Chaos technique for quantifying the uncertainties; constructing efficient solvers and preconditioners in order to accelerate the solving process.

We derive the modelings for the high Reynolds number flow in Chapter 2. By comparing the existing models, we choose the two-scale Variational Multiscale Method (VMS) to govern the flow behavior in the blood pump configuration. Because the VMS does not only provide a general consistent framework for the multiscale problems, but it enables also the stabilized finite element methods, which are widely employed in many fields. We then introduce two techniques for coping with the moving mesh requirement. Although the combination of the Continuous and Discontinuous Galerkin techniques fit better the domain separation for the rotating device modelization, the shear layer update approach is definitely much more efficient. By taking advantage of the special treatments on the mesh decomposition, the shear layer update approach is able to reuse many functions, which are available for the most of finite element libraries. The solution update procedure, which is required after the mesh regeneration, becomes almost effortless. Moreover, the pyramid element which acts as a "glue" element is also implemented.

For the Uncertainty Quantification, we consider the intrusive stochastic Galerkin method, since the generalized Polynomial Chaos expansion enables the possibility of representing the square-integrable stochastic process in a very efficient manner. However, the increase of the size of the global system and the possible convergence breakdown due to the higher degree random inputs in a nonlinear system can be a barrier for utilizing this method. However, as described in Chapter 3, when the random inputs have only the first order expansion, the system matrix of the stochastic spectral problem becomes sparse. This matrix structure is easier to be deployed during the solving processing.

Concerning solving the coupled stochastic system, we propose a new approach for the different components in the global linear system. We start with the stochastic Galerkin matrix-vector multiplication in Chapter 4, such that, we are able to reduce the memory storage by a considerable factor. We combine the FGMRES solver and the Polynomial Chaos expansion Multilevel preconditioner to cope with the global system. The PCE preconditioner deploys the hierarchy structure of the Chaos Polynomials and tries to eliminate the errors on the different levels. This preconditioning technique is shown to be very efficient by using several test examples in this contribution (Chapter 5 and Chapter 6). Especially, the inexact version can be scaled with more than 2000 processors and reduce the computing time significantly. Analogously to the Multigrid method, we have to choose a suitable smoother for the PCE Multilevel preconditioner. The Mean-based preconditioner comes into play, because this preconditioner is very easy to be constructed and comparably powerful. Moreover, solving the linear system associated with the mean block matrix efficiently is another breaking point for designing the proposed solving strategy. We use here the Schur Complement preconditioner to precondition the linear system in the FGMRES iteration. Thanks to the VMS, the pressure-pressure block is directly accessible in the system matrix. Based on the augmented Lagrangian approach, the approximation of the Schur Complement is hence built.

In Chapter 6, we provide the numerical results of the blood pump simulation. We start with a steady flow computation in order to further prove the arguments of choosing the inexact Multilevel preconditioner for our computation. We select three uncertain inputs in our numerical model: the inflow boundary condition, the dynamic viscosity and the rotational speed. These three parameters cover different uncertain sources: the modeling error, the variance of the anatomical parameters and the operative variation of the machines. Afterward, the numerical solution of our stochastic partial differential equation is presented. We show the flow behavior of the mean value and the standard deviation at two different stages. Some quantities of interest are also studied, such as the axial and radial forces on the rotor, the pressure loading on the impeller, the static pressure on the periphery of the impeller. Furthermore, we provide the hemolysis quantification based on the stress-based hemolysis model. The regions with high hemolysis index in the blood pump are identified by using our stochastic solution.

Our final simulation computes 20 Polynomial Chaos modes with the Chaos Polynomial degree 3. The total degrees of freedom is about 45.5 Millions, the computation is performed with 2048 processors. Thence, it would be very interesting in a further work to increase the mesh resolution of the deterministic problem to investigate whether the flow has a dramatic change on a finer mesh. Further, we could also increase the Chaos Polynomial degree in order to examine the contribution from the higher order PC mode solutions to the stochastic moments. Concerning the deterministic problem, the stabilization parameters are chosen from the existing literatures, a further numerical investigation of those coefficients may also be needed.

Currently, experimental data are in the process of preparation, it is important after their release to compare our simulation outcomes with the experiments. We can then pursue the inverse Uncertainty Quantification techniques and the Data Assimilation techniques in order to identify the unobservable objects in the data and the simulations.

List of Figures

List of Tables

Bibliography

[1] Naveed Ahmed, Tomás Chacón Rebollo, Volker John, and Samuele Rubino. A review of variational multiscale methods for the simulation of turbulent incompressible flows. *Archives of Computational Methods in Engineering*, 24(1):115–164, 2017.

[2] RE Alcouffe, Achi Brandt, JE Dendy, Jr, and JW Painter. The multi-grid method for the diffusion equation with strongly discontinuous coefficients. *SIAM Journal on Scientific and Statistical Computing*, 2(4):430–454, 1981.

[3] Andrew P. Ambrosy, Gregg C. Fonarow, Javed Butler, Ovidiu Chioncel, Stephen J. Greene, Muthiah Vaduganathan, Savina Nodari, Carolyn S.P. Lam, Naoki Sato, Ami N. Shah, and Mihai Gheorghiade. The global health and economic burden of hospitalizations for heart failure: Lessons learned from hospitalized heart failure registries. *Journal of the American College of Cardiology*, 63(12):1123 – 1133, 2014.

[4] George E. Andrews and Richard Askey. *Classical orthogonal polynomials*, pages 36–62. Springer Berlin Heidelberg, Berlin, Heidelberg, 1985.

[5] H. Anzt, W. Augustin, M. Baumann, T. Gengenbach, T. Hahn, A. Helfrich-Schkarbanenko, V. Heuveline, E. Ketelaer, D. Lukarski, A. Nestler, S. Ritterbusch, S. Ronnas, M. Schick, M. Schmidtobreick, C. Subramanian, J.-P. Weiss, F. Wilhelm, and M. Wlotzka. Hiflow3: A hardware-aware parallel finite element package. In *Tools for High Performance Computing 2011*, pages 139–151. Springer Berlin Heidelberg, 2012.

[6] Vincenzo Armenio and Ugo Piomelli. A lagrangian mixed subgrid-scale model in generalized coordinates. *Flow, Turbulence and Combustion*, 65(1):51–81, 2000.

[7] W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quarterly of Applied Mathematics*, 9(1):17–29, 1951.

[8] D Arora, M Behr, O Coronado-Matutti, and M Pasquali. Estimation of hemolysis in centrifugal blood pumps using morphology tensor approach. In *Proceedings of 3rd MIT Conference on Computational Fluid and Solid Dynamics*, pages 578–582, 2005.

[9] Dhruv Arora, Marek Behr, and Matteo Pasquali. Hemolysis estimation in a centrifugal blood pump using a tensor-based measure. *Artificial Organs*, 30(7):539–547, 2006.

[10] Arash Arvand, Nicole Hahn, Marcus Hormes, Mustafa Akdis, Michael Martin, and Helmut Reul. Comparison of hydraulic and hemolytic properties of different impeller designs of an implantable rotary blood pump by computational fluid dynamics. *Artificial Organs*, 28(10):892–898, 2004.

[11] R. Askey and J.A. Wilson. *Some Basic Hypergeometric Orthogonal Polynomials that Generalize Jacobi Polynomials*. Number 319 in American Mathematical Society: Memoirs of the American Mathematical Society. American Mathematical Soc., 1985.

[12] Ivo Babuška, Fabio Nobile, and Raul Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 45(3):1005–1034, 2007.

[13] Ivo Babuška, Fabio Nobile, and Raúl Tempone. A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 45(3):1005–1034, 2007.

[14] Allison H. Baker, Robert D. Falgout, Tzanio V. Kolev, and Ulrike Meier Yang. *Scaling Hypre's Multigrid Solvers to 100,000 Cores*, pages 261–279. Springer London, London, 2012.

[15] W. Bangerth and R. Rannacher. *Adaptive Finite Element Methods for Differential Equations*. Lectures in Mathematics. Springer, 2003.

[16] Jorge Bardina, J Ferziger, and WC Reynolds. Improved subgrid-scale models for large-eddy simulation. In *13th Fluid and PlasmaDynamics Conference*, page 1357, 1980.

[17] G.I. Barenblatt. *Scaling, Self-similarity, and Intermediate Asymptotics: Dimensional Analysis and Intermediate Asymptotics*. Cambridge Texts in Applied Mathematics. Cambridge University Press, 1996.

[18] Brandon Barker. Message passing interface (mpi). In *Workshop: High Performance Computing on Stampede*, 2015.

[19] Peter Bastian. Lecture notes on parallel solution of large sparse linear systems, July 2015.

[20] Y Bazilevs, VM Calo, JA Cottrell, TJR Hughes, A Reali, and G Scovazzi. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 197(1):173–201, 2007.

[21] R. Becker, V. Heuveline, and R. Rannacher. An optimal control approach to adaptivity in computational fluid mechanics. *International Journal for Numerical Methods in Fluids*, 40(1-2):105–120, 2002.

[22] M Behbahani, M Behr, M Hormes, U Steinseifer, D Arora, O Coronado, and M Pasquali. A review of computational fluid dynamics analysis of blood pumps. *European Journal of Applied Mathematics*, 20(4):363–397, 2009.

[23] M. Behr and T. Tezduyar. The shear-slip mesh update method. *Computer Methods in Applied Mechanics and Engineering*, 174(3–4):261 – 274, 1999.

[24] R.E. Bellman. *Dynamic Programming*. Dover Books on Computer Science Series. Dover Publications, 2003.

[25] JA Benek, P Buning, and J Steger. A 3-d chimera grid embedding technique. In *7th Computational Physics Conference*, page 1523, 1985.

[26] Michele Benzi, Maxim A Olshanskii, and Zhen Wang. Modified augmented lagrangian preconditioners for the incompressible navier-stokes equations. *International Journal for Numerical Methods in Fluids*, 66(4):486–508, 2011.

[27] Luigi C Berselli and Volker John. On the comparison of a commutation error and the reynolds stress tensor for flows obeying a wall law. *Preprint*, 18, 2004.

[28] A Bishop, P Messina, et al. Scientific grand challenges in national security: The role of computing at the extreme scale. In *The High Energy Physics extreme scale workshop was held December, 9-11, 2008 at the SLAC National Accelerator Laboratory, Menlo Park, CA*. Report from the US Department of Energy (DOE) Workshop, 2009.

[29] Perry L Blackshear Jr, Frank D Dorman, and Joseph H Steinbach. Some mechanical effects that influence hemolysis., 1965.

[30] J. Boussinesq. *Essai sur la théorie des eaux courantes*. Mémoires présentées par divers savants à l'Académie des Sciences. Imprimerie Nationale, 1877.

[31] Christopher E Brennen. *Hydrodynamics of pumps*. Cambridge University Press, 2011.

[32] Susanne Brenner and Ridgway Scott. *The mathematical theory of finite element methods*, volume 15. Springer Science & Business Media, 2007.

[33] William L Briggs, Van Emden Henson, and Steve F McCormick. *A multigrid tutorial*. SIAM, 2000.

[34] Alexander N. Brooks and Thomas J.R. Hughes. Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 32(1–3):199 – 259, 1982.

[35] W. Bujalski, Z. Jaworski, and A.W. Nienow. CFD study of homogenization with dual rushton turbines—comparison with experimental results. *Chemical Engineering Research and Design*, 80(1):97 – 104, 2002. Process and Product Development.

[36] R. H. Cameron and W. T. Martin. The orthogonal development of non-linear functionals in series of fourier-hermite functionals. *Annals of Mathematics*, 48(2):385–392, 1947.

[37] P. Chadwick. *Continuum Mechanics: Concise Theory and Problems*. Dover books on physics. Dover Publications, 1999.

[38] Vasiliki Chatzi and Franco P. Preparata. Using pyramids in mixed meshes - point placement and basis functions. Technical report, Brown University, Providence, RI, USA, 2000.

[39] Chuanmiao Chen, Michal Křížek, and Liping Liu. Numerical integration over pyramids. *Advances in Applied Mathematics and Mechanics*, 5(03):309–320, 2013.

[40] Haecheon Choi and Parviz Moin. Grid-point requirements for large eddy simulation: Chapman's estimates revisited. *Physics of Fluids*, 24(1):011702, 2012.

[41] S. Scott Collis. Monitoring unresolved scales in multiscale turbulence modeling. *Physics of Fluids*, 13(6):1800–1806, 2001.

[42] ASTM Committee et al. Standard practice for assessment of hemolysis in continuous flow blood pumps. *Annual Book of ASTM Standards, F1844-97*, 13:1–5, 1998.

[43] John Crank and Phyllis Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 43, pages 50–67. Cambridge Univ Press, 1947.

[44] Samuel MA Cruz, AJ Marques Cardoso, and Hamid A Toliyat. Diagnosis of stator, rotor and airgap eccentricity faults in three-phase induction motors based on the multiple reference frames theory. In *Industry Applications Conference, 2003. 38th IAS Annual Meeting. Conference Record of the*, volume 2, pages 1340–1346. IEEE, 2003.

[45] Bert J Debusschere, Habib N Najm, Alain Matta, Omar M Knio, Roger G Ghanem, and Olivier P Le Maître. Protein labeling reactions in electrochemical microchannel flow: Numerical simulation and uncertainty propagation. *Physics of fluids*, 15(8):2238–2250, 2003.

[46] Jean Donea, S Giuliani, and Jean-Pierre Halleux. An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions. *Computer methods in applied mechanics and engineering*, 33(1-3):689–723, 1982.

[47] Jean Donea and Antonio Huerta. *Finite element methods for flow problems.* John Wiley & Sons, 2003.

[48] A. Dunca, V. John, and W. J. Layton. *The Commutation Error of the Space Averaged Navier-Stokes Equations on a Bounded Domain*, pages 53–78. Birkhäuser Basel, Basel, 2004.

[49] F. Durst and M. Schäfer. A parallel block-structured multigrid method for the prediction of incompressible flows. *International Journal for Numerical Methods in Fluids*, 22(6):549–565, 1996.

[50] Stanley C Eisenstat and Homer F Walker. Choosing the forcing terms in an inexact newton method. *SIAM Journal on Scientific Computing*, 17(1):16–32, 1996.

[51] O.G. Ernst, C.E. Powell, D.J. Silvester, and E. Ullmann. Efficient solvers for a linear stochastic galerkin mixed formulation of diffusion problems with random data. *SIAM Journal on Scientific Computing*, 31(2):1424–1447, 2009.

[52] Oliver G Ernst, Antje Mugler, Hans-Jörg Starkloff, and Elisabeth Ullmann. On the convergence of generalized polynomial chaos expansions. *ESAIM: Mathematical Modelling and Numerical Analysis*, 46(2):317–339, 2012.

[53] Robert D Falgout and Ulrike Meier Yang. hypre: A library of high performance preconditioners. In *International Conference on Computational Science*, pages 632–641. Springer Berlin Heidelberg, 2002.

[54] P.E. Farrell and J.R. Maddison. Conservative interpolation between volume meshes by local galerkin projection. *Computer Methods in Applied Mechanics and Engineering*, 200(1–4):89 – 100, 2011.

[55] U.S. Food and Drug Administration (FDA). Fda's "critical path" computational fluid dynamics (cfd)/blood damage project, 2013.

[56] Davide Forti and Luca Dedè. Semi-implicit bdf time discretization of the navier–stokes equations with vms-les modeling in a high performance computing framework. *Computers & Fluids*, 117:168–182, 2015.

[57] M. Fortin and R. Glowinski. *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems.* Studies in Mathematics and its Applications. Elsevier Science, 2000.

[58] Yuan-cheng Fung. *Biomechanics: motion, flow, stress, and growth.* Springer Science & Business Media, 2013.

[59] Giovanni P. Galdi. *An Introduction to the Navier-Stokes Initial-Boundary Value Problem*, pages 1–70. Birkhäuser Basel, Basel, 2000.

[60] Simon Gawlok, Philipp Gerstner, Saskia Haupt, Vincent Heuveline, Jonas Kratzke, Philipp Lösel, Katrin Mang, Mareike Schmidtobreick, Nicolai Schoch, Nils Schween, Jonathan Schwegler, Chen Song, and Martin Wlotzka. Hiflow3 – technical report on release 2.0. *Preprint Series of the Engineering Mathematics and Computing Lab*, 0(06), 2017.

[61] Massimo Germano, Ugo Piomelli, Parviz Moin, and William H Cabot. A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, 3(7):1760–1765, 1991.

[62] Roger G. Ghanem and Pol D. Spanos. *Stochastic Finite Elements: A Spectral Approach*. Springer-Verlag New York, Inc., New York, NY, USA, 1991.

[63] M Giersiepen, LJ Wurzinger, R Opitz, and H Reul. Estimation of shear stress-related blood damage in heart valve prostheses–in vitro comparison of 25 aortic valves. *The International journal of artificial organs*, 13(5):300–306, 1990.

[64] Michael B Giles. Multilevel monte carlo methods. *Acta Numerica*, 24:259, 2015.

[65] V. Girault and P.A. Raviart. *Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2012.

[66] Roland Glowinski. Finite element methods for the numerical simulation of incompressible viscous flow. introduction to the control of the navier-stokes equations. *Lectures in Applied Mathematics*, 28:219–301, 1991.

[67] Roland Glowinski and Olivier Pironneau. Finite element methods for navier-stokes equations. *Annual review of fluid mechanics*, 24(1):167–204, 1992.

[68] P.M. Gresho. Some current cfd issues relevant to the incompressible navier-stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 87(2):201 – 252, 1991.

[69] J.L. Guermond, P. Minev, and Jie Shen. An overview of projection methods for incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 195(44):6011 – 6045, 2006.

[70] Wolfgang Hackbusch. *Multi-grid methods and applications*, volume 4. Springer Science & Business Media, 2013.

[71] J.H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2:84–90, 1960.

[72] R.W. Hamming and R.W. Hamming. *Numerical Methods for Scientists and Engineers*. Dover books on engineering. Dover, 1973.

[73] Vahé haroutunian, Michael S. Engelman, and Isaac Hasbani. Segregated finite element algorithms for the numerical solution of large-scale incompressible flow problems. *International Journal for Numerical Methods in Fluids*, 17(4):323–348, 1993.

[74] Stefan Heinrich. *Multilevel Monte Carlo Methods*, pages 58–67. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[75] Van Emden Henson and Ulrike Meier Yang. Boomeramg: A parallel algebraic multigrid solver and preconditioner. *Applied Numerical Mathematics*, 41(1):155 – 177, 2002.

[76] J.S. Hesthaven, G. Rozza, and B. Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. SpringerBriefs in Mathematics. Springer International Publishing, 2015.

[77] V. Heuveline and C. Bertsch. On multigrid methods for the eigenvalue computation of nonselfadjoint elliptic operators. *East-West Journal of Numerical Mathematics*, 8(4):275–297, 2000.

[78] Vincent Heuveline, Eva Ketelaer, Staffan Ronnas, Mareike Schmidtobreick, and Martin Wlotzka. Scalability study of hiflow3 based on a fluid flow channel benchmark. *Preprint Series of the Engineering Mathematics and Computing Lab*, 0(05), 2012.

[79] Vincent Heuveline, Dimitar Lukarski, Nico Trost, and Jan-Philipp Weiss. *Parallel Smoothers for Matrix-Based Geometric Multigrid Methods on Locally Refined Meshes Using Multicore CPUs and GPUs*, pages 158–171. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[80] Vincent Heuveline and Michael Schick. Towards a hybrid numerical method using generalized polynomial chaos for stochastic differential equations. *Preprint Series of the Engineering Mathematics and Computing Lab*, 0(03), 2011.

[81] Vincent Heuveline and Michael Schick. A hybrid generalized polynomial chaos method for stochastic dynamical systems. *International Journal for Uncertainty Quantification*, 4(1):37–61, 2014.

[82] Vincent Heuveline, Michael Schick, Clayton Webster, and Peter Zaspel. Uncertainty quantification and high performance computing (dagstuhl seminar 16372). In *Dagstuhl Reports*, volume 6, 2017.

[83] J. G. Heywood, R. Rannacher, and S. Turek. Artificial boundaries and flux and pressure conditions for the incompressible navier-stokes equations. *International Journal for Numerical Methods in Fluids*, 22(5):325–352, 1996.

[84] Eberhard Hopf. über die anfangswertaufgabe für die hydrodynamischen grundgleichungen. erhard schmidt zu seinem 75. geburtstag gewidmet. *Mathematische Nachrichten*, 4(1-6):213–231, 1950.

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

[86] Guillaume Houzeaux and Ramon Codina. A chimera method based on a dirichlet/neumann(robin) coupling for the navier–stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 192(31–32):3343 – 3377, 2003.

[87] Ming-Chen Hsu, Ido Akkerman, and Yuri Bazilevs. Wind turbine aerodynamics using ale–vms: validation and the role of weakly enforced boundary conditions. *Computational Mechanics*, 50(4):499–511, 2012.

[88] Ming-Chen Hsu and Yuri Bazilevs. Fluid–structure interaction modeling of wind turbines: simulating the full machine. *Computational Mechanics*, pages 1–13, 2012.

[89] W. Huang and R.D. Russell. *Adaptive Moving Mesh Methods*. Applied Mathematical Sciences. Springer, 2010.

[90] Thomas J.R. Hughes. Multiscale phenomena: Green's functions, the dirichlet-to-neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Computer Methods in Applied Mechanics and Engineering*, 127(1):387 – 401, 1995.

[91] Thomas J.R. Hughes, Gonzalo R. Feijóo, Luca Mazzei, and Jean-Baptiste Quincy. The variational multiscale method—a paradigm for computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, 166(1):3 – 24, 1998.

[92] Thomas J.R. Hughes, Leopoldo P. Franca, and Marc Balestra. A new finite element formulation for computational fluid dynamics: V. circumventing the babuška-brezzi condition: a stable petrov-galerkin formulation of the stokes problem accommodating equal-order interpolations. *Computer Methods in Applied Mechanics and Engineering*, 59(1):85 – 99, 1986.

[93] Thomas JR Hughes, Luca Mazzei, and Kenneth E Jansen. Large eddy simulation and the variational multiscale method. *Computing and visualization in science*, 3(1):47–59, 2000.

[94] Thomas JR Hughes and Giancarlo Sangalli. Variational multiscale analysis: the fine-scale green's function, projection, optimization, localization, and stabilized methods. *SIAM Journal on Numerical Analysis*, 45(2):539–557, 2007.

[95] Ronald L Iman, James M Davenport, and Diane K Zeigler. Latin hypercube sampling (program user's guide).[lhc, in fortran]. Technical report, Sandia Labs., Albuquerque, NM (USA), 1980.

[96] Svante Janson. *Gaussian hilbert spaces*, volume 129. Cambridge university press, 1997.

[97] Volker John, Songul Kaya, and William Layton. A two-level variational multiscale method for convection-dominated convection–diffusion equations. *Computer Methods in Applied Mechanics and Engineering*, 195(33):4594–4603, 2006.

[98] Charles R. Johnson. Inverse m-matrices. *Linear Algebra and its Applications*, 47:195 – 216, 1982.

[99] George Karypis. Metis and parmetis. In *Encyclopedia of Parallel Computing*, pages 1117–1124. Springer, 2011.

[100] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.

[101] Jozef Kačur. Method of rothe in evolution equations. In *Equadiff 6*, pages 23–34. J. E. Purkyně University, Department of Mathematics, 1986.

[102] David Kay, Daniel Loghin, and Andrew Wathen. A preconditioner for the steady-state navier–stokes equations. *SIAM Journal on Scientific Computing*, 24(1):237–256, 2002.

[103] F. Kerdouss, A. Bannari, and P. Proulx. Cfd modeling of gas dispersion and bubble size in a double turbine stirred tank. *Chemical Engineering Science*, 61(10):3313 – 3322, 2006.

[104] Roelof Koekoek and Rene F Swarttouw. The askey-scheme of hypergeometric orthogonal polynomials and its q-analogue. *arXiv preprint math/9602214*, 1996.

[105] A. N. Kolmogorov. Equations of turbulent motion of an incompressible fluid. *Izv. Akad. Nauk SSSR, Ser. Fiz*, 6(56-58), 1942.

[106] P. C. Krause. Method of multiple reference frames applied to the analysis of symmetrical induction machinery. *IEEE Transactions on Power Apparatus and Systems*, PAS-87(1):218–227, Jan 1968.

[107] Barbro Kreiss. Construction of a curvilinear grid. *SIAM Journal on Scientific and Statistical Computing*, 4(2):270–279, 1983.

[108] Ethan J Kubatko, Benjamin A Yeager, and Ashley L Maggi. New computationally efficient quadrature formulas for pyramidal elements. *Finite Elements in Analysis and Design*, 65:63–75, 2013.

[109] O. Le Maître and O.M. Knio. *Spectral Methods for Uncertainty Quantification: With Applications to Computational Fluid Dynamics*. Scientific Computation. Springer Netherlands, 2010.

[110] A. Leonard. Energy cascade in large-eddy simulations of turbulent fluid flows. In F.N. Frenkiel and R.E. Munn, editors, *Turbulent Diffusion in Environmental PollutionProceedings of a Symposium held at Charlottesville*, volume 18, Part A of *Advances in Geophysics*, pages 237 – 248. Elsevier, 1975.

[111] Jean Leray. Sur le mouvement d'un liquide visqueux emplissant l'espace. *Acta mathematica*, 63(1):193–248, 1934.

[112] Douglas K Lilly. A proposed modification of the germano subgrid-scale closure method. *Physics of Fluids A: Fluid Dynamics*, 4(3):633–635, 1992.

[113] Erb F Lins, Renato N Elias, Gabriel M Guerra, Fernando A Rochinha, and Alvaro LGA Coutinho. Edge-based finite element implementation of the residual-based variational multiscale method. *International Journal for Numerical Methods in Fluids*, 61(1):1–22, 2009.

[114] Jacques-Louis Lions and Giovanni Prodi. Un théoreme d'existence et unité dans les équations de navier-stokes en dimension 2. *Comptes rendus hebdomadaires des seances de l'academie des sciences*, 248(25):3519–3521, 1959.

[115] Liping Liu, Kevin B Davies, Michal Křížek, and Li Guan. On higher order pyramidal finite elements. *Advances in Applied Mathematics and Mechanics*, 3(02):131–140, 2011.

[116] Liping Liu, Kevin B Davies, Kewei Yuan, and M Krizek. On symmetric pyramidal finite elements. *Dynamics of Continuous Discrete and Impulsive Systems Series B*, 11:213–228, 2004.

[117] Daniel Loghin and AJ Wathen. Schur complement preconditioners for the navier-stokes equations. *International journal for numerical methods in fluids*, 40(3-4):403–412, 2002.

[118] Gabriel J. Lord, Catherine E. Powell, and Tony Shardlow. *Introduction to Computational Stochastic PDEs*. Number 50 in Cambridge Texts in Applied Mathematics. Cambridge University Press, 2014.

[119] Olivier P Le Maître, Omar M Knio, Habib N Najm, and Roger G Ghanem. A stochastic projection method for fluid flow. *Journal of Computational Physics*, 173(2):481 – 511, 2001.

[120] Olivier P. Le Maître, Matthew T. Reagan, Habib N. Najm, Roger G. Ghanem, and Omar M. Knio. A stochastic projection method for fluid flow: Ii. random process. *Journal of Computational Physics*, 181(1):9 – 44, 2002.

[121] Lionel Mathelin and M. Yousuff Hussaini. A stochastic collocation algorithm for uncertainty analysis. *NASA*, pages 2003–212153, 2003.

[122] Lionel Mathelin, M. Yousuff Hussaini, and Thomas A. Zang. Stochastic approaches to uncertainty quantification in cfd simulations. *Numerical Algorithms*, 38(1):209–236, 2005.

[123] R.E. Megginson. *An Introduction to Banach Space Theory*. Graduate Texts in Mathematics. Springer New York, 2012.

[124] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 209:415–446, 1909.

[125] Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949. PMID: 18139350.

[126] R. v. Mises. Mechanik der festen körper im plastisch- deformablen zustand. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, 1913:582–592, 1913.

[127] Kozo Naito, Kazumi Mizuguchi, and Yukihiko Nosé. The need for standardizing the index of hemolysis. *Artificial Organs*, 18(1):7–10, 1994.

[128] F. Nicoud and F. Ducros. Subgrid-scale stress modelling based on the square of the velocity gradient tensor. *Flow, Turbulence and Combustion*, 62(3):183–200, 1999.

[129] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods.* Society for Industrial and Applied Mathematics, 1992.

[130] F. Nobile, R. Tempone, and C. G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2309–2345, 2008.

[131] Steven A. Orszag. Analytical theories of turbulence. *Journal of Fluid Mechanics*, 41(2):363–386, 1970.

[132] SV Patankar. *Numerical Heat Transfer and Fluid Flow.* McGraw Hill, 1980.

[133] Lutz Pauli, Jaewook Nam, Matteo Pasquali, and Marek Behr. Transient stress-based and strain-based hemolysis estimation in a simplified blood pump. *International Journal for Numerical Methods in Biomedical Engineering*, 29(10):1148–1160, 2013.

[134] Lutz Helmut Pauli. *Stabilized finite element methods for computational design of blood-handling devices; 1. Auflage.* Dissertation, RWTH Aachen University, München, 2016. Deutsche und englische Zusammenfassung; Dissertation, RWTH Aachen University, 2016.

[135] R.J. Plemmons. M-matrix characterizations.i—nonsingular m-matrices. *Linear Algebra and its Applications*, 18(2):175 – 188, 1977.

[136] S.B. Pope. *Turbulent Flows.* Cambridge University Press, 2000.

[137] Catherine E Powell and Howard C Elman. Block-diagonal preconditioning for spectral stochastic finite-element systems. *IMA Journal of Numerical Analysis*, 29(2):350–375, 2009.

[138] Matthew T. Reagan, Habib N. Najm, Roger G. Ghanem, and Omar M. Knio. Uncertainty quantification in reacting-flow simulations through non-intrusive spectral projection. *Combustion and Flame*, 132(3):545 – 555, 2003.

[139] Karel Rektorys. The method of discretization in time and partial differential equations. *Equadiff 5*, pages 293–296, 1982.

[140] Staffan Ronnas, Thomas Gengenbach, Eva Ketelaer, and Vincent Heuveline. Design and implementation of distributed meshes in hiflow3. In *Competence in High Performance Computing 2010*, pages 61–71. Springer Berlin Heidelberg, 2012.

[141] Eveline Rosseel and Stefan Vandewalle. Iterative solvers for the stochastic finite element method. *SIAM Journal on Scientific Computing*, 32(1):372–397, 2010.

[142] John W Ruge and Klaus Stüben. Algebraic multigrid. *Multigrid methods*, 3(13):73–130, 1987.

[143] Youcef Saad. A flexible inner-outer preconditioned gmres algorithm. *SIAM Journal on Scientific Computing*, 14(2):461–469, 1993.

[144] Youcef Saad and Martin H Schultz. Conjugate gradient-like algorithms for solving nonsymmetric linear systems. *Mathematics of Computation*, 44(170):417–424, 1985.

[145] Yousef Saad. *Iterative methods for sparse linear systems.* SIAM, 2003.

[146] Pierre Sagaut. *Large eddy simulation for incompressible flows: an introduction.* Springer Science & Business Media, 2006.

[147] Michael Schäfer, Stefan Turek, Franz Durst, Egon Krause, and Rolf Rannacher. *Benchmark computations of laminar flow around a cylinder.* Springer, 1996.

[148] M. Schick, V. Heuveline, and O. P. Le Ma. A newton–galerkin method for fluid flow exhibiting uncertain periodic dynamics. *SIAM/ASA Journal on Uncertainty Quantification*, 2(1):153–173, 2014.

[149] Michael Schick. A parallel multilevel spectral galerkin solver for linear systems with uncertain parameters. In *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on*, pages 352–359. IEEE, 2014.

[150] Michael Schick, Chen Song, and Vincent Heuveline. A polynomial chaos method for uncertainty quantification in blood pump simulation. In *UNCECOMP 2015 - 1st ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering*, pages 93–106. Scopus, ElsevierNational, 2015.

[151] Holger Schmid-Schönbein and Roe Wells. Fluid drop-like transition of erythrocytes under shear. *Science*, 165(3890):288–291, 1969.

[152] A Segal, M ur Rehman, and C Vuik. Preconditioners for incompressible navier-stokes solvers. *Numerical Mathematics: Theory, Methods and Applications*, 3(3):245–275, 2010.

[153] Lee A Segel and GH Handelman. *Mathematics applied to continuum mechanics.* SIAM, 2007.

[154] David Silvester, Howard Elman, David Kay, and Andrew Wathen. Efficient preconditioning of the linearized navier–stokes equations for incompressible flow. *Journal of Computational and Applied Mathematics*, 128(1):261 – 279, 2001. Numerical Analysis 2000. Vol. VII: Partial Differential Equations.

[155] Joseph Smagorinsky. General circulation experiments with the primitive equations: I. the basic experiment. *Monthly weather review*, 91(3):99–164, 1963.

[156] S. A. Smolyak. Quadrature and interpolation formulas for tensor products of certain class of functions. *Dokl. Akad. Nauk SSSR*, 148(5):1042–1053, 1963. Transl.: Soviet Math. Dokl. 4:240-243, 1963.

[157] Chen Song and Vincent Heuveline. Multilevel preconditioner of polynomial chaos method for quantifying uncertainties in a blood pump. In *International Conference on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP), Greece, 2017.* Scopus, Elsevier, 2017.

[158] Joseph L Steger, F Carroll Dougherty, and John A Benek. A chimera grid scheme.[multiple overset body-conforming mesh system for finite difference adaptation to complex aircraft configurations]. In *Advances in grid generation; Proceedings of the Applied Mechanics, Bioengineering, and Fluids Engineering Conference, Houston, TX*, pages 59–69, 1983.

[159] Rene Steijl and George Barakos. Sliding mesh algorithm for cfd analysis of helicopter rotor–fuselage aerodynamics. *International journal for numerical methods in fluids*, 58(5):527–549, 2008.

[160] Timothy John Sullivan. *Introduction to uncertainty quantification*, volume 63. Springer, 2015.

[161] G. Szegö. *Orthogonal Polynomials.* Number Bd. 23 in American Mathematical Society colloquium publications. American Mathematical Society, 1939.

[162] C. Taylor and P. Hood. A numerical solution of the navier-stokes equations using the finite element technique. *Computers & Fluids*, 1(1):73 – 100, 1973.

[163] R. Temam. *Navier-Stokes Equations: Theory & Numerical Analysis.* Studies in Mathematics and Its Applications. North-Holland, 1984.

[164] Tayfun Tezduyar and Sunil Sathe. Stabilization parameters in supg and pspg formulations. *Journal of computational and applied mechanics*, 4(1):71–88, 2003.

[165] Tayfun E Tezduyar. Stabilized finite element formulations for incompressible flow computations. *Advances in applied mechanics*, 28:1–44, 1991.

[166] Tayfun E Tezduyar and Yasuo Osawa. Finite element stabilization parameters computed from element matrices and vectors. *Computer Methods in Applied Mechanics and Engineering*, 190(3–4):411 – 430, 2000.

[167] T.E. Tezduyar, S. Mittal, S.E. Ray, and R. Shih. Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements. *Computer Methods in Applied Mechanics and Engineering*, 95(2):221 – 242, 1992.

[168] Stefan Turek. *Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approache*, volume 6. Springer Science & Business Media, 1999.

[169] Gerhard Wanner and Ernst Hairer. Solving ordinary differential equations ii. *Stiff and Differential-Algebraic Problems*, 1991.

[170] E Weinan. *Principles of multiscale modeling.* Cambridge University Press, 2011.

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

[172] Christian Wieners. Conforming discretizations on tetrahedrons, pyramids, prisms and hexahedrons. *Preprint, University of Stuttgart*, 1997.

[173] Jingchun Wu, Bradley E. Paden, Harvey S. Borovetz, and James F. Antaki. Computational fluid dynamics analysis of blade tip clearances on hemodynamic performance and blood damage in a centrifugal ventricular assist device. *Artificial Organs*, 34(5):402–411, 2010.

[174] Dongbin Xiu. *Numerical Methods for Stochastic Computations: A Spectral Method Approach.* Princeton University Press, Princeton, NJ, USA, 2010.

[175] Dongbin Xiu and Jan S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 27(3):1118–1139, 2005.

[176] Dongbin Xiu and George E Karniadakis. The wiener-askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, 24(2):614–644, 2002.

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

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

[179] Ulrike Meier Yang. *Parallel Algebraic Multigrid Methods — High Performance Preconditioners*, pages 209–236. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[180] Yan Zang, Robert L Street, and Jeffrey R Koseff. A dynamic mixed subgrid-scale model and its application to turbulent recirculating flows. *Physics of Fluids A: Fluid Dynamics*, 5(12):3186–3196, 1993.

[181] Tao Zhang, M Ertan Taskin, Hong-Bin Fang, Adam Pampori, Robert Jarvik, Bartley P Griffith, and Zhongjun J Wu. Study of flow-induced hemolysis using novel couette-type blood-shearing devices. *Artificial organs*, 35(12):1180–1186, 2011.

[182] O.C. Zienkiewicz, R.L. Taylor, and J.Z. Zhu. *The Finite Element Method: Its Basis and Fundamentals*. Butterworth-Heinemann, Oxford, seventh edition edition, 2013.

人生是艺术，与命运无关