Thèse n° 10 031

Multi-Level Monte Carlo Methods for Uncertainty Quantification and Risk-Averse Optimisation

Présentée le 15 décembre 2022

Faculté des sciences de base
Calcul scientifique et quantification de l'incertitude - Chaire CADMOS
Programme doctoral en mathématiques

pour l'obtention du grade de Docteur ès Sciences

par

Sundar Subramaniam GANESH

Acceptée sur proposition du jury

Prof. D. Kressner, président du jury
Prof. F. Nobile, directeur de thèse
Dr G. Geraci, rapporteur
Prof. B. Keith, rapporteur
Prof. M. Picasso, rapporteur

École
polytechnique
fédérale
de Lausanne

2022

Om Asato Maa Sad-Gamaya |
Tamaso Maa Jyotir-Gamaya |
Mrtyor-Maa Amrtam Gamaya |
Om Shaantih Shaantih Shaantih ||

Om, From falsehood lead me to truth,
From darkness lead me to light,
From death lead me to immortality.
Om peace peace peace.

To my parents...

Acknowledgements

This work was made possible through an endless amount of support from a wide community of colleagues, friends and family. These words are a humble attempt at expressing my eternal gratitude for this support.

I begin with my supervisor, Fabio Nobile, to whom I am extremely grateful for giving me the opportunity to pursue this work, and for his tireless patience and guidance in supervising me over the years, through all of the hills and valleys that this work has taken us through.

I would like to thank my jury; Daniel Kressner, Marco Piccaso, Brendan Keith and Gianluca Geraci, for their input on my research work, and for their agreement to examine my thesis. I am also extremely grateful for having met and worked with Sebastian Krumscheid, who has effectively been my second supervisor. Words cannot express how thankful I am for his support, guidance and motivation through the years. I am grateful to my colleages from the ExaQUte project; Quentin, Riccardo, Marc, Anoop, and the rest of the ExaQUte team. Working with them has been a great joy, and I have learned so much over the years from our weekly calls. I am proud of what we have managed to accomplish together in these four years. Special thanks goes to Shibu Clement, my bachelor thesis supervisor, and Hossein Gorji, my master thesis supervisor, who have laid the foundation for this work, and have opened many of the doors that led me here.

I cannot adequately express how amazing and supportive my colleagues, now my friends, from the CSQI group have been. I'm very thankful to have shared my office for the majority of this journey with Eva; we've supported each other through all of the PhD life's ups and downs, especially since therapy is so expensive these days. I'm also grateful to have shared this journey with Davide, who has taught be a thing or two about grit and determination, and my future finance-bro Juan, with whom I've had the absolute pleasure of collaborating and sharing much beer/coffee. I'm thankful for all of my coffee-talks with Celia, through which I've learnt much about keeping calm during a storm. I'm also grateful to have worked with Tommaso, whose insights have helped much during my research. I'm glad I was able to share this last bit of my PhD journey with Fabio, Thomas and Matteo, whose sheer energy for group activities brings all of us joy. I also with to thank Yoshihito and Panos, as well the many visitors who have stayed with us at CSQI, including Chiara and Jonas.

Acknowledgements

It's been one of my greatest joys to share these years with my friends from MCSS; Deep, Qian, Zhenying, Nicolino and especially Niccolo. We have caused an amount of global warming comparable to Shell and Exxon, through He Nan, Chez Bilia and hotpot. It's also an honour to have been a part of the MATHICSE family and the mathematics department. I've shared a great deal of hikes, climbs, football and math with Riccardo, Giacomo, Giacomino, Luca P. and Luca C., Ondine, Alice, Andrea, Fabian, Caterina and all the rest of the department.

Climbing has been a big part of my life these last few years. Through it, I've met some amazing people, who have supported me through all of the ups and downs of a PhD program. I am grateful for my friends Nacho, Liyan, Katie and Denisa. You've been my family here, and I'll forever cherish our many days of climbing trips, dinners and board games. I'm also grateful to have met the Alpaca gang; Nico, Davide, Matteo, Mateusz, Fiona, Erik, Aylwin and Pier. Apart from acquiring a partially functional knowledge of Italian, the sheer amount of warmth I've felt from you all during our crazy mountain days together has been overwhelming.

I want to thank Bharat and Sandra, who made Switzerland feel a little less foreign, and a lot more like home, for someone who arrived here from far away. Special thanks to Greg, who supported my baby steps into Switzerland, and who has been a staunch support through my time here. I'm also especially grateful to my roommate and good friend, Niccolo. We've shared all of the ups and downs, not just of a PhD program, but of the growing pains of life at a formative time in our journey. I'll forever be thankful for all our Luigia dinners on the balcony, shared video-game time, and late-night life-talks through the pandemic.

It has been one my life's greatest joys to have met Anusha, and to have shared this journey with her. I am eternally thankful for her unwaivering support, patience and warmth. Her companionship has made each challenge much less daunting, and she has lent me the strength to face them. Lastly, I want to thank my family; Amma, Appa, Malini, Shashvat, Shivansh and Rishabh. You've believed in me from day one, even at times when I didn't always believe in myself, and none of this would have been possible without your love and support.

Lausanne, December 1, 2022

Abstract

This thesis is devoted to studying the estimation and minimisation of risk in engineering problems, and is divided into three main parts.
The first part addresses the challenge of risk-estimation using the Multi-Level Monte Carlo (MLMC) method. Specifically, we tackle the problem of accurately estimating the probability density function, the cumulative distribution function, the quantile, and the expectation in the tail above the quantile, the so called Conditional-Value-at-Risk (CVaR), of a random output Quantity of Interest (QoI) of a complex differential model with input uncertainties. We propose to use the framework of MLMC estimators for parametric expectations developed in [85] and we present novel error estimates for these MLMC estimators, as well as a novel adaptive MLMC parameter selection procedure based on the novel error estimates to achieve a prescribed tolerance on the above-mentioned summary statistics in a cost-optimal manner.
The second part addresses the challenge of risk-averse engineering design. We seek to minimise the CVaR of a random output quantity of interest of a complex differential model using gradient-based approaches combined with the MLMC method. Specifically, we propose novel MLMC estimators for the sensitivities of the CVaR with respect to design parameters based on the framework of parametric expectations developed in the first part of the thesis. We propose combining this MLMC framework with an alternating minimisation-gradient descent algorithm, for which we prove exponential convergence in the optimisation iterations under the assumptions of strong convexity and Lipschitz continuity of the gradients.
In the third part, we present our work on the development of a new software library for hierarchical Monte Carlo methods, developed as a part of the Horizon 2020 European Union Project titled "ExaQUte". The software library is designed to mirror the hierarchical structure common to the MLMC, the multi-index and some multi-fidelity Monte Carlo estimators. In addition, the library also mirrors the common structure of several adaptive MLMC and Monte Carlo algorithms that are used to calibrate the parameters of the aforementioned estimators. This combination allows users to implement their own hierarchical Monte Carlo estimators and algorithms. In addition, the library is parallelised using an external task scheduler, where the tasks of computing independent samples across the various hierarchy levels, the error estimation and the adaptive parameter selection procedures are all scheduled in parallel in a manner that respects resource-locality and task-dependencies.
Lastly, we demonstrate the above developments on an array of simple demonstrative problems run in serial, as well as on more applied examples pertinent to the ExaQUte project for

which high-performance computational hardware was used to conduct parallelised simulations.

Résumé

Cette thèse est consacrée à l'étude de l'estimation et de la minimisation du risque dans les problèmes d'ingénierie, et est divisée en trois parties principales.
La première partie aborde le défi de l'estimation du risque en utilisant la méthode de Monte Carlo multi-niveaux (MLMC). Plus précisément, nous abordons le problème de l'estimation précise de la fonction de densité de probabilité, de la fonction de distribution cumulative, du quantile et de l'espérance dans la queue au-dessus du quantile, appelée valeur conditionnelle à risque (CVaR), d'une quantité d'intérêt (QoI) de sortie aléatoire d'un modèle différentiel complexe avec des incertitudes d'entrée. Nous proposons d'utiliser le cadre des estimateurs MLMC pour les attentes paramétriques développé dans [85] et nous présentons de nouvelles estimations d'erreur pour ces estimateurs MLMC, ainsi qu'une nouvelle procédure adaptative de sélection des paramètres MLMC basée sur les nouvelles estimations d'erreur pour atteindre une tolérance prescrite sur les statistiques sommaires susmentionnées d'une manière optimale en termes de coût.
La deuxième partie aborde le défi de la conception technique averse au risque. Nous cherchons à minimiser le CVaR d'une quantité de sortie aléatoire d'intérêt d'un modèle différentiel complexe en utilisant des approches basées sur le gradient combinées à la méthode MLMC. Plus précisément, nous proposons de nouveaux estimateurs MLMC pour les sensibilités du CVaR par rapport aux paramètres de conception basés sur le cadre des attentes paramétriques développé dans la première partie de la thèse. Nous proposons de combiner ce cadre MLMC avec un algorithme alternatif de minimisation et de descente de gradient, pour lequel nous prouvons une convergence exponentielle dans les itérations d'optimisation sous les hypothèses de forte convexité et de continuité Lipschitz des gradients.
Dans la troisième partie, nous présentons notre travail sur le développement d'une nouvelle software pour les méthodes de Monte Carlo hiérarchiques, développée dans le cadre du projet de l'Union européenne Horizon 2020 intitulé "ExaQUte". La software est conçue pour refléter la structure hiérarchique commune au MLMC, au multi-index et à certains estimateurs Monte Carlo multifidélité. En outre, la software reflète également la structure commune de plusieurs algorithmes MLMC et Monte Carlo adaptatifs qui sont utilisés pour calibrer les paramètres des estimateurs susmentionnés. Cette combinaison permet aux utilisateurs de mettre en œuvre leurs propres estimateurs et algorithmes de Monte Carlo hiérarchiques. En outre, la software est parallélisée à l'aide d'un planificateur de tâches externe, où les tâches de calcul d'échantillons indépendants à travers les différents niveaux de la hiérarchie, l'estimation des erreurs et les procédures de sélection adaptative des paramètres sont

toutes planifiées en parallèle d'une manière qui respecte la localisation des ressources et les dépendances des tâches.
Enfin, nous démontrons les développements ci-dessus sur un ensemble de problèmes démonstratifs simples exécutés en série, ainsi que sur des exemples plus appliqués pertinents pour le projet ExaQUte pour lequel du matériel de calcul haute performance a été utilisé pour effectuer des simulations parallélisées.

Contents

1 Introduction

Scientific computing has grown vastly in recent history, giving researchers the ability to simulate the behaviour of complex multiscale phenomena. In typical applications, the mathematical models used to describe complex phenomena typically do not have closed form solutions. Dubbed the "third pillar of the scientific method", computational methods allow us to simulate approximate solutions to these models using computers, and enable us to control the accuracy of these approximate solutions at a practically feasible computing cost. Advances in computing architecture, numerical algorithms and solution techniques have helped proliferate scientific computing as an important tool in nearly all technological disciplines. The focus of this thesis research is on two particular branches of scientific computing; namely Uncertainty Quantification (UQ) and Optimization Under Uncertainty (OUU).

Mathematical models typically contain several input parameters whose values need to be calibrated to accurately model natural phenomena. However, the values of some of these inputs can contain uncertainty due to noise or measurement error, or contain inherent fluctuations due to the phenomenon being modelled. Other input parameters may be controllable and can be used to steer the model towards desirable performances and/or reduce the influence of uncertainties on output QoIs. A classical example is of wind flow over a bluff body. The flow model may be subject to uncertainties in the wind conditions, and the effects of these uncertainties may be controlled by changing the shape parameters of the body. UQ methods aim at characterizing the input uncertainties and quantifying their effects on output Quantity of Interest (QoI). Many UQ approaches take a probabilistic point of view, describing such uncertainties as randomness in the input parameters, and quantify the uncertainty in an output QoI through the use of summary statistics such as the mean, central moments, or quantiles. Moreover, in the field of OUU, one is interested in selecting the control input parameters of a model such that these statistics are optimised. In particular, risk-averse OUU seeks to select the control parameters of a system under the influence of input noise, such that the system is robust to non-typical or unfavourable operating conditions.

In this research, we tackle the challenge of developing risk-averse optimisation algorithms for

complex engineering problems. Specifically, we are interested in the risk-averse shape optimisation of engineering structures subject to uncertain wind conditions such that they are robust to uncertainties in the wind. A large part of the work presented in this thesis has been carried out within the European Union Horizon 2020 project titled "Exascale Quantification of Uncertainties for Technology and Science Simulation", or ExaQUte for short.

In Section1.1, we provide an overview of the ExaQUte project and its structure, tailored towards achieving this goal. In Section 1.2, we highlight and describe the main contributions made by this author towards the ExaQUte project and towards this thesis research. The overall structure of the thesis will be detailed in Section 1.3.

1.1 The ExaQUte project

The European Union Horizon 2020 ExaQUte project (2018-2021) [40], short for "Exascale Quantification of Uncertainties for Technology and Science Simulation", aimed at developing a framework to enable the solution to UQ and OUU problems for complex engineering systems on exascale computing architecture. The project was a consortium of multiple universities and partners, the details of whom can be found in Table 1.1.

Participant	**Country**
Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE)	Spain
Barcelona Supercomputing Center (BSC)	Spain
Technische Universität München (TUM)	Germany
Institut national de recherche en sciences et technologies du numérique (Inria)	France
IT4Innovations National Supercomputing Center (IT4I)	Czech Republic
École Polytechnique Fédérale de Lausanne (EPFL)	Switzerland
Universitat Politècnica de Catalunya (UPC)	Spain
str.ucture GmbH	Germany

Table 1.1: ExaQUte project list of partners

The overall aim of the ExaQUte project was divided into several "work packages" (WP), aimed at distributing the component tasks of the framework based on the target expertise of each of the respective partners in Table 1.1. The work packages are detailed in Fig. 1.1, which also shows the interdependency of the ExaQUte work packages, with the arrows indicating that the developments achieved within one work package will be used in the following work package. The interested reader is referred to the project website for a detailed description of each of the work packages [132]. The majority of the research work pursued by EPFL, a significant portion of which is presented in this thesis, was conducted as a part of WP5 and WP6. The aim of WP5 was to propose theoretical and algorithmic extensions to the MLMC method that

would enable the use of MLMC estimators for complex differential models in a highly parallel environment, in addition to developing the necessary MLMC framework for risk-estimation. WP6, the central goal of the ExaQUte project, aimed at developing gradient-based optimisation algorithms for risk-averse shape design, that use the aforementioned MLMC framework developed in WP5 for gradient estimation.

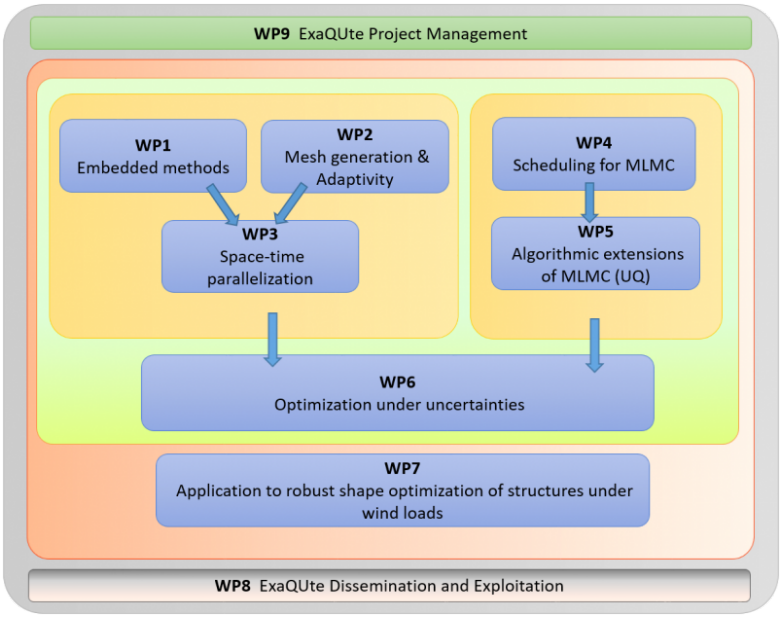


Figure 1.1: ExaQUte project work packages and their interdependencies

The project began in June 2018, with a scheduled end date of May 2021, with a total duration of 36 months. However, due to delays related to the COVID-19 pandemic, the end of the project was postponed to November 2021, extending the duration of the project to 42 months. Subject to this timeline, each of the work packages WP5 and WP6 were divided further into tasks. The description of the tasks, as well as their timeline proposed for the total 42 month-long project, are shown in the Gantt plot in Fig. 1.2. The research work conducted within this thesis begins at month 7 of the project, and encompasses tasks 5.2-5.5 and 6.2-6.5.

Each of the tasks in Fig. 1.2 required the time-bound submission of a comprehensive deliv-

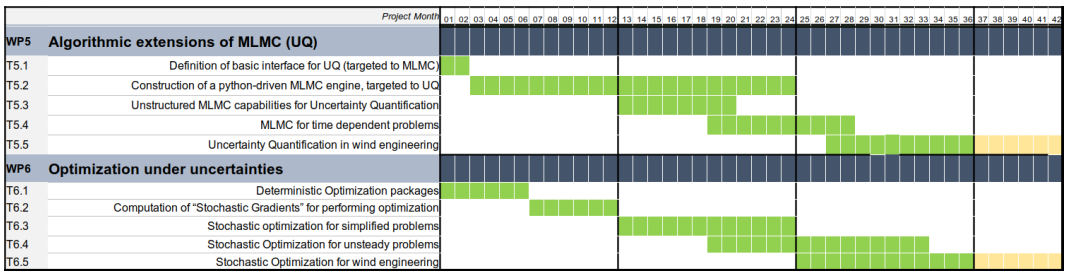


Figure 1.2: Timeline of the ExaQUte project for the tasks specific to this thesis

erable report, describing the steps taken towards accomplishing the corresponding task. The reports were due at the end of the corresponding period indicated in Fig. 1.2. In addition to the submission of periodic reports, the deliverables also consisted of periodic releases of a novel Python MLMC engine, titled X-Monte Carlo (XMC), initially developed as a part of task 5.2 and used extensively within the consortium thereafter [5]. The time-line and topics of research of this thesis have hence been selected to primarily prioritize the timely completion and submission of the deliverable reports and software releases to the European Union. Table 1.2 provides a brief description of each deliverable report headed by EPFL. The author was a co-author of all of the deliverables, and was a main contributor to deliverables D5.2 to 5.5, D6.2 and D6.5, with comparatively less contribution to D6.3 and D6.4.

Deliverable	**Reference**	**Content**
D5.2	[6]	Description of the XMC Python engine [5]
D5.3	[19]	Potential algorithms for the use of MLMC methods with adaptive mesh refinement
D5.4	[18]	Feasibility of MLMC methods for time-dependent problems
D5.5	[20]	MLMC estimators and algorithms for risk-measures applied to engineering problems
D6.2	[51]	OUU problem formulation and derivation of stochastic sensitivities
D6.3	[17]	Novel gradient-based OUU algorithms for risk-measures
D6.4	[14]	Stochastic optimisation for time dependent problems
D6.5	[15]	Risk-averse design using MLMC estimators applied to engineering problems

Table 1.2: ExaQUte project list of partners

The ExaQUte project was successfully concluded in January 2022 with the presentation of the cumulative results of the project to a scientific committee appointed by the European Union, consisting of three academic and industrial experts. The project received praise for its ambitious goals and significant results, and for its successful completion within the planned time-frame.

1.2 Contributions made by the thesis

The ExaQUte project was structured from its conceptualisation to focus on the development of suitable MLMC methods for use within the UQ and OUU problems inherent to risk-averse shape optimisation for wind engineering problems. As will be seen in Chapter 2, MLMC methods have shown great success in reducing the cost of simulations wherein the underlying problem is a complex differential model with high-dimensional random inputs, which characterises our application area of wind engineering. As a result, it was decided to struc-

ture the ExaQUte project around the development of MLMC methods. In addition, it was also decided during the conceptualisation phase that the ExaQUte project would aim at quantifying and minimising risks through the use of the CVaR, a risk-measure that quantifies risks associated to unlikely scenarios in the upper tail of the distribution of the output QoI and possesses highly favourable properties for OUU algorithms. This meant that suitable MLMC methods would have to be developed to estimate the CVaR, as well as its sensitivities with respect to design parameters. This, in turn, would require addressing several challenges, each with a corresponding contribution made during this thesis research.

The first challenge is related to the theme of WP5; namely, the quantification of the risks associated to a given design using the MLMC method where the risk is described in terms of the Probability Density Function (PDF), the Cumulative Distribution Function (CDF), the Value-at-Risk (VaR) or the CVaR of the output QoI. We followed the approach of [85], which recast the estimation of the above quantities to the computation of parametric expectations. We developed novel computable error estimates for the estimation of such quantities, which are then used to optimally select the parameters of the corresponding MLMC estimator in a continuation type adaptive algorithm. The efficiency and robustness of our novel procedure was demonstrated on an array of numerical test cases of increasing complexity. These developments are presented in our work [16], as well as in the deliverable report [20].

The second challenge is related to the subject of WP6; namely, the development of novel OUU algorithms for risk-averse shape optimisation using the MLMC framework. To this end, we tackled the problem of minimising the CVaR using gradient-based approaches in combination with MLMC estimators. In particular, we considered the framework of MLMC for parametric expectations that we developed for the CVaR in [16], and proposed modifications of the MLMC estimator, error estimation procedure, and adaptive MLMC parameter selection to ensure the accurate estimation of the CVaR and its sensitivities for a given design with a prescribed accuracy. We then proposed combining the MLMC framework with an alternating inexact minimisation-gradient descent algorithm for which we prove exponential convergence in the optimisation iterations under plausible assumptions on the objective function and its gradients. We demonstrated the performance of our approach on two numerical examples of practical relevance, which evidenced the same optimal asymptotic cost-tolerance behaviour as standard MLMC methods for fixed design computations of output expectations. The developments are summarized in our work [52], as well as the deliverable reports [51, 17, 14, 15].

The third challenge is related to implementing the mathematical and algorithmic procedures developed in addressing the first two challenges into a software framework capable of scalable parallelism on high-performance hardware. A key advantage of MLMC methods is their vast potential for parallelism. During the ExaQUte project, the author was involved in the development of the ExaQUte software framework, a collection of software tools that were developed simultaneously and collaboratively by the partners of the consortium, aimed at conducting exascale simulations on high-performance hardware. Specifically, the author

contributed to the development of the aforementioned XMC Python library [5], a software library that was founded and developed during this thesis research. In addition, the author was also involved in the integration of the library with the other software tools of the ExaQUte software framework, namely the Kratos multi-physics engine [95] and the PyCOMPSs/Hyperloom task schedulers [125, 35], which allowed the parallelised implementation of several MLMC algorithms and estimators for QoI that were outputs of complex differential models such as fluid-flow equations. The public repository of the XMC library can be found at [5], and a summary of its structure and capabilities can be found in [6]. The ExaQUte software framework was also applied by the author, in collaboration with the members of the consortium, to simulate risk-estimation and risk-averse shape optimisation for a problem of practical interest on high-performance hardware, thereby demonstrating the successful integration of the algorthmic and mathematical developments with the software components. The simulation results are summarized in [20] and [15] respectively, and are reported in this thesis.

Lastly, the grand goal of the ExaQUte project was to apply the above-mentioned frameworks to the problem of risk-averse shape optimisation of a civil engineering structure subject to turbulent wind flow conditions. To this end, the author of this thesis, in collaboration with members of the consortium, explored in [19] and [18] the conditions under which such a complex problem could be treated using MLMC methods. Several small- and large-scale simulations were conducted, specifically in [18], to assess whether the conditions necessary for the optimal performance of the MLMC method could be attained for a turbulent wind flow problem. The simulation results of [18] demonstrated that while the MLMC method could be used for oscillatory problems such as vortex shedding at lower Reynolds' numbers, the hypotheses necessary for the optimal performance of MLMC would be challenging to attain for turbulent problems. The report also proposed alternative methods for solving the UQ problem for turbulent flows. These results are also presented within this thesis.

1.3 Structure of the thesis

This thesis is structured into three parts, each related to a contribution described in Section 1.2. The first part, consisting of Chapters 2 and 3, focuses on the MLMC estimation of risk-measures. Specifically, we first present an overview of the current literature on MLMC methods in Chapter 2, laying a foundation for the rest of the thesis. We then summarize in Chapter 3 the developments that we presented in [16] and [20] on MLMC estimators for parametric expectations.

The second part, consisting of Chapters 4 and 5, sheds light on the combination of the MLMC algorithms developed in the first part with novel gradient-based approaches to tackle the challenge of CVaR minimisation. We provide an overview in Chapter 4 of the current literature on Partial Differential Equation (PDE)-constrained OUU of the CVaR, and the challenges associated with the use of MLMC methods for gradient-based CVaR minimisation. We then

describe in Chapter 5 the extension of the parametric expectation approach described in Chapter 3 that allows us to use MLMC estimators to estimate the sensitivities of the CVaR. We also describe our novel gradient-based OUU framework that uses inexact gradients computed using MLMC to minimise the CVaR. The combination is demonstrated on two problems of practical relevance.

The third part, consisting of Chapters 6 and 7 delve deeper into the collaborative software development aspects of the ExaQUte project, focussing on the work completed during this thesis research towards the development of the XMC library, as well as its integration with the ExaQUte software framework. Chapter 6 outlines the structure of the XMC library, and describes how its design and its combination with the PyCOMPSs/Hyperloom scheduler can efficiently exploit the parallelism inherent to MLMC and other hierarchical Monte Carlo estimators. Chapter 7 then describes multiple large-scale numerical simulations that were conducted using the ExaQUte software framework and the XMC library; namely the feasibility studies conducted to explore the use of MLMC methods for turbulent problems, as well as the production simulations conducted in [20] and [15] to demonstrate the successful implementation of the algorithmic developments described in Chapters 3 and 5 within the ExaQUte software framework.

Lastly, we present a conclusion in Chapter 8, summarizing the results of this thesis, and provide an outlook for future research on the topic.

Multi-Level Monte Carlo methods Part I

2 Introduction to MLMC methods

Complex differential models are used in many disciplines across science and engineering as predictive or design tools. More often than not, however, some input parameters of these models are uncertain, either due to missing information, lack of proper characterization or intrinsic variability. It is hence of utmost interest to study and quantify the effects of these uncertainties on an output QoI of the model, or several QoIs, which are in turn used for prediction or design. When uncertainty is modelled as randomness in a probabilistic framework, each QoI becomes a random variable and its distribution is often inaccessible in closed form. We assume here that the QoI can be simulated in an approximate way, typically by sampling the random input parameters and computing the solution of a suitable discretisation of the underlying differential model. It is therefore of great practical interest to estimate by simulation, and with controlled accuracy, the distribution of the QoI or some summary statistics such as the mean, central moments of different orders, or quantiles of a given significance.

Solving the underlying differential model at a desired accuracy typically has a high computational cost, even for a single realisation of the random input. An accurate estimation of the summary statistics of a QoI by a direct Monte Carlo approach is often prohibitively expensive. MLMC methods, as introduced in the works [57] and [70], are a well established technology to compute the expected value of a random QoI that is an output quantity of a stochastic differential model. MLMC estimators exploit multiple discretisations of the underlying differental model and have shown significant performance improvements over standard Monte Carlo algorithms [57, 85, 68, 45] when the appropriate parameters are selected properly.

In this chapter, we recall the construction of the basic MLMC estimator for estimating the expected value of a random QoI, through which we review the relevant literature on the different aspects of MLMC methods. In particular, Section 2.1 introduces this MLMC estimator, reviewing some basic complexity results as well as presenting an overview of practical MLMC algorithms. Section 2.2 highlights several common elements of MLMC methods, as well as other hierarchical Monte Carlo methods such as the Multi-Index Monte Carlo (MIMC) method presented in Section 2.2.1 and the Multi-Fidelity Monte Carlo (MFMC) method presented in Section 2.2.2. The association between these common structures and the XMC

software library [5] that we developed during this thesis research are highlighted in this section. Lastly, Section 2.3 gives a brief overview of some extensions of MLMC to compute summary statistics other than the expected value, as well as sampling strategies other than Monte Carlo.

2.1 MLMC estimators and complexity behaviour

Let $(\Omega, \mathscr{F}, \mathbb{P})$ denote a complete probability space, $\omega \in \Omega$ an elementary random event and $Q : \Omega \to \mathbb{R}$ a real valued QoI. We address here the problem of estimating the expected value $\mathbb{E}[Q]$ of the random QoI Q. Sampling directly from Q is typically not possible for the applications of interest in this thesis., which involve solving complex differential models. Rather, one samples from an approximation Q_h to Q, where h denotes an appropriate approximation parameter. For example, h can be a mesh parameter associated to a discretisation of the underlying differential model of which Q is an output.

We first study a naive Monte Carlo estimator for $\mathbb{E}[Q]$:

$$\mathbb{E}[Q] \approx \hat{\mu}_{\mathrm{mc}} := \frac{1}{N} \sum_{i=1}^{N} \frac{Q_h^{(i)}}{N}, \tag{2.1}$$

where $\left\{Q_h^{(i)}\right\}_{i=1}^{N}$ are N independent identically distributed samples of Q_h. The accuracy of this estimator can be quantified by the MSE, defined as $\mathrm{MSE}\left(\hat{\mu}_{\mathrm{mc}}\right) := \mathbb{E}\left[(\hat{\mu}_{\mathrm{mc}} - \mathbb{E}[Q])^2\right]$. Thanks to the independence of the samples $Q_h^{(i)}$, it is easy to see that

$$\mathrm{MSE}\left(\hat{\mu}_{\mathrm{mc}}\right) = (\mathbb{E}[Q_h - Q])^2 + \frac{\mathbb{V}\mathrm{ar}(Q_h)}{N}. \tag{2.2}$$

The first term, called the squared bias error, describes the discretisation error in approximating Q using Q_h, and the second term, called the statistical error, is related to finite sampling using N samples. Both error contributions need to be balanced to obtain a good estimate.

We assume that there exist positive constants $C_\alpha, \alpha, C_\gamma, \gamma$ such that

$$|\mathbb{E}[Q_h - Q]| \leq C_\alpha h^\alpha, \tag{2.3a}$$

$$\mathrm{Cost}\left(Q_h^{(i)}\right) \leq C_\gamma h^{-\gamma}. \tag{2.3b}$$

We remark that $\mathrm{Cost}\left(Q_h^{(i)}\right)$ denotes the expected cost of computing one realisation of Q_h, since different realisations can possibly have different costs, for example, due to differences in the convergence of iterative methods used to solve the underlying PDE. We require the MSE in Eq. (2.2) to satisfy a tolerance of ϵ^2, split equally between the squared bias and the statistical error contributions. By selecting the number of samples $N \propto \epsilon^{-2}$ and the discretisation parameter $h \propto \epsilon^{1/\alpha}$, it can be shown that the cost to compute the estimator $\hat{\mu}_{\mathrm{mc}}$ scales

as

$$\mathrm{Cost}\left(\hat{\mu}_{\mathrm{mc}}\right) \lesssim \epsilon^{-2-\gamma/\alpha}. \tag{2.4}$$

The cost term in Eq. (2.4) consists of two parts, namely the cost to solve the underlying problem once, which scales as $\mathscr{O}\left(\epsilon^{-\gamma/\alpha}\right)$, and the cost of sampling, which scales as $\mathscr{O}\left(\epsilon^{-2}\right)$.

MLMC methods aim to "hide" the cost of solving the underlying problem such that complexity of a comparable MLMC estimator scales only as $\mathscr{O}\left(\epsilon^{-2}\right)$ in the best case. MLMC methods work by sampling from a set of approximations $\{Q_l\}_{l=0}^{L}$ to Q on a sequence of $L+1$ discretisations with different characteristic discretisation parameters, for example induced by mesh sizes $h_0 > h_1 > ... > h_L$, typically a geometric sequence $h_{l-1} = sh_l$ with $s > 1$. The MLMC estimator to estimate $\mathbb{E}[Q]$ is given by

$$\mathbb{E}[Q] \approx \hat{\mu} := \frac{1}{N_0}\sum_{i=1}^{N_0} Q_0^{(i,0)} + \sum_{l=1}^{L}\frac{1}{N_l}\sum_{i=1}^{N_l}\left[Q_l^{(i,l)} - Q_{l-1}^{(i,l)}\right], \tag{2.5}$$

where $Q_l^{(i,l)}$ and $Q_{l-1}^{(i,l)}$ are correlated realisations of the QoI computed with the same underlying realisation of the input parameters on meshes with discretisation parameters h_l and h_{l-1} respectively, $Q_{l,l-1}^{(i,l)}$ and $Q_{k,k-1}^{(j,k)}$ are otherwise independent if $i \neq j$ or $k \neq l$, and $\{N_l\}_{l=0}^{L}$ is a decreasing sequence of sample sizes. The sample sizes N_l and the number of levels L are commonly referred to as the MLMC "hierarchy". Notice that the sum over the discretisation levels $l = 0, ..., L$ telescopes in expectation:

$$\mathbb{E}\left[\hat{\mu}\right] = \mathbb{E}[Q_0] + \sum_{l=1}^{L}\mathbb{E}[Q_l - Q_{l-1}] = \mathbb{E}[Q_L] \approx \mathbb{E}[Q], \tag{2.6}$$

and hence, the bias or discretisation error $|\mathbb{E}\left[\hat{\mu} - Q\right]| = |\mathbb{E}[Q_L - Q]|$ of the MLMC estimator depends only on the finest discretisation level considered.

The MSE of the MLMC estimator, defined again as $\mathrm{MSE}\left(\hat{\mu}\right) := \mathbb{E}\left[(\hat{\mu} - \mathbb{E}[Q])^2\right]$, splits in this case as:

$$\mathrm{MSE}\left(\hat{\mu}\right) = (\mathbb{E}[Q_L - Q])^2 + \sum_{l=0}^{L}\frac{\mathbb{V}\mathrm{ar}(Q_l - Q_{l-1})}{N_l}, \tag{2.7}$$

where $Q_{-1} := 0$. We see that the bias term is analogous to the one in Eq. (2.2), whereas the statistical error term is now split over the $L+1$ levels, thanks to the fact that the level-wise Monte Carlo estimators $\hat{\mu}_{mc,l} = \sum_{i=1}^{N_l}(Q_l^{(i,l)} - Q_{l-1}^{(i,l)})/N_l$ are mutually independent. We again make the assumption that there exist positive constants $C_\alpha, \alpha, C_\beta, \beta, C_\gamma, \gamma$ such that the following

hold:

$$b_l := |\mathbb{E}[Q_l - Q]| \leq C_\alpha h_l^\alpha, \tag{2.8a}$$

$$V_l := \mathbb{V}\mathrm{ar}(Q_l - Q_{l-1}) \leq C_\beta h_l^\beta, \tag{2.8b}$$

$$\mathfrak{c}_l := \mathrm{Cost}\left((Q_l^{(i,l)}, Q_{l-1}^{(i,l)}\right) \leq C_\gamma h_l^{-\gamma}. \tag{2.8c}$$

We remark once again that $\mathfrak{c}_l$ denotes the expected cost of computing one realisation each of Q_l and Q_{l-1}. We require the MSE to satisfy a tolerance of ϵ^2 that is equally split between the bias and statistical error terms in Eq. (2.7). The bias error is controlled by the number of levels L. Assuming a geometric sequence of discretisations $h_l = h_0 s^{-l}$, $s > 1$, we wish to select L such that the bias error satisfies a tolerance of $\epsilon^2/2$:

$$C_\alpha h_L^\alpha \leq \frac{\epsilon}{\sqrt{2}} \implies L = \left\lceil \frac{1}{\alpha \log(s)} \log\left(\frac{\sqrt{2} C_\alpha h_0}{\epsilon}\right)\right\rceil. \tag{2.9}$$

Once L has been selected, we wish to select the level-wise sample sizes $\{N_l\}_{l=0}^L$ to minimise the cost of the estimator $\hat{\mu}$ subject to the constraint that the statistical error satisfies a tolerance of $\epsilon^2/2$:

$$\{N_l^*\}_{l=0}^L = \underset{\{N_l\}_{l=0}^L \in \mathbb{N}^L}{\operatorname{argmin}} \sum_{l=0}^L N_l \mathfrak{c}_l \quad \text{s.t.} \quad \sum_{l=0}^L \frac{V_l}{N_l} \leq \frac{\epsilon^2}{2}. \tag{2.10}$$

It has been shown in [57] that a nearly optimal solution to this problem is given by

$$N_l^* = \left\lceil \frac{2}{\epsilon^2} \sqrt{\frac{V_l}{C_l}} \left(\sum_{k=0}^L \sqrt{V_k \mathfrak{c}_k}\right)\right\rceil, \quad l \in \{0, 1, \ldots, L\}. \tag{2.11}$$

This process of selecting L and N_l using knowledge of the estimates b_l, V_l and $\mathfrak{c}_l$, as well as the model constants and rates in Eqs. (2.8), is usually referred to in the literature as "tuning". It was shown in [57] that by selecting the number of levels and sample sizes as in Eqs. (2.9) and (2.11), the cost of the MLMC simulation scales as

$$\mathrm{Cost}(\hat{\mu}) = \sum_{l=0}^L N_l \mathfrak{c}_l \lesssim \begin{cases} \epsilon^{-2}, & \beta > \gamma, \\ \epsilon^{-2}(\log \epsilon)^2, & \beta = \gamma, \\ \epsilon^{-2-(\gamma-\beta)/\alpha}, & \beta < \gamma. \end{cases} \tag{2.12}$$

For $\beta > \gamma$, the cost is dominated by Monte Carlo sampling on the coarsest levels and, hence, the cost to solve a single problem on the finest discretisation $\mathcal{O}\left(\epsilon^{-\gamma/\alpha}\right)$ does not appear here. For $\beta = \gamma$ the number of samples is distributed evenly across levels and for $\beta < \gamma$, the cost is primarily on the finest levels. Even in this worst case, MLMC estimators are an improvement over standard Monte Carlo in terms of complexity as can be seen when compared with (2.4).

2.1.1 Adaptive tuning of MLMC algorithms

Selecting the hierarchy parameters $L(\epsilon)$ and $N_l(\epsilon)$ in a cost optimal manner requires knowledge of the quantities b_l, V_l and $\mathfrak{c}_l$, as well as their corresponding decay rates. Knowledge of these quantities is typically not available prior to computing samples. It was proposed in [57] to obtain estimates of b_l, V_l and $\mathfrak{c}_l$ by running a "screening" phase with a few samples. These quantities were then estimated by sample average and sample variance estimators using the screening hierarchy. The optimal hierarchy for a given tolerance ϵ^2 could then be computed based on these estimates. However, since the screening phase typically contains very few samples and levels, the resulting estimates of b_l, V_l and $\mathfrak{c}_l$ may not be accurate.

To remedy this issue, one can begin with a screening phase, but incrementally improve the estimates of b_l, V_l and $\mathfrak{c}_l$ in an iterative manner. The authors of [56] proposed Algorithm 1, a heuristic algorithm wherein the number of levels in the MLMC hierarchy is incremented by one for every iteration of the algorithm.

Algorithm 1: Adaptive MLMC algorithm from [56]

```
Input: Target tolerance ε > 0.
Start with L = 2 and N_l = N^0 samples, l ∈ {0,1,2}.
while extra samples need to be evaluated do
    Evaluate extra samples on each level
    Compute estimates for V_l and c_l
    Compute optimal sample sizes N_l(ε) according to Eq. (2.11). Set N_L = N^0.
    if |E[Q_L − Q_{L−1}]| / (e^α − 1) < ε/√2 then
        Exit loop
    else
        Set L = L + 1
    end if
end while
```

Alternatively, the authors of [39] proposed an approach, named CMLMC, where, in contrast to Algorithm 1, the MLMC hierarchy is adapted to a sequence of decreasing tolerances $\epsilon_i = \epsilon\kappa^{N-i}, i \in \{0,...,N\}$, of which the target tolerance ϵ is the final one. Such a construction makes the optimal MLMC estimator for the final tolerance robust to inaccurate initial estimates of b_l, V_l and $\mathfrak{c}_l$ from the screening hierarchy. This algorithm is described in Algorithm 2. In this thesis research, we develop such continuation type algorithms, both for use in risk-estimation and for combination with risk-averse gradient-based optimisation techniques. The reader is referred to [56], in addition to the works of [39, 105], for detailed descriptions of various different MLMC algorithms to estimate b_l, V_l and $\mathfrak{c}_l$ and adaptively tune the parameters of the MLMC hierarchy based on them.

Algorithm 2: CMLMC Algorithm

Input: Target tolerance $\epsilon > 0$, Number of CMLMC iterations $d \in \mathbb{N}$, Tolerance refinement ratios $\lambda > \kappa > 1$. Set $j = 1$, $\epsilon_a = \epsilon_0$.
Launch screening hierarchy.
Compute estimates b_l, V_l and $\mathfrak{c}_l$ and model parameters $c_\alpha, c_\beta, c_\gamma, \alpha, \beta, \gamma$.
Compute $\mathrm{MSE}\left(\hat{\mu}\right)$ based on Eq. (2.7)
while $j \leq d$ **or** $\mathrm{MSE}\left(\hat{\mu}\right) \geq \epsilon^2$ **do**
  Launch hierarchy with $L^*(\epsilon_a)$, $\{N_l^*(\epsilon_a)\}_{l=0}^{L^*}$ computed based on Eqs. (2.9) and (2.11)
  if $j \leq d$ { Set $\epsilon_a = \epsilon\lambda^{(d-j)}$ } **else** { Set $\epsilon_a = \epsilon\kappa^{(d-j)}$ }
  Compute estimates b_l, V_l, $\mathfrak{c}_l$ and model parameters $c_\alpha, c_\beta, c_\gamma, \alpha, \beta, \gamma$
  Compute $\mathrm{MSE}\left(\hat{\mu}\right)$ based on Eq. (2.7)
  Update $j \leftarrow j + 1$
end while

2.2 Beyond MLMC: A common structure for hierarchical estimators and relation to XMC software

A key part of this thesis research is the development of the XMC software library [5]. The structure of the library was chosen to reflect the MLMC idea of sampling the hierarchical differences $Q_l - Q_{l-1}$ for different l. Additionally, the library structure was also designed to reflect the commonalities between Algorithm 1 and 2, as well as other commonly used MLMC algorithms. The common features are identified as follows:

- The estimator is the sum of independent estimators over a list of levels (or, more generally, indices) such that:

$$\hat{\mu} = \sum_{l \in \mathscr{L}} \hat{\mu}_l, \tag{2.13}$$

  where $\mathscr{L}$ denotes the list of levels/indices.

- Each $\hat{\mu}_l$ is a Monte Carlo type estimator, requiring multiple independent evaluations of the QoI using different discretisations or models, e.g., $Q_l - Q_{l-1}$ at level l

- Level/index-dependent quantities such as b_l, V_l and $\mathfrak{c}_l$ must be estimated for error control/adaptivity.

- A global coordination between levels/indices (adaptive algorithm) is needed to define the new hierarchy once all computations and level-wise error estimations have been completed.

As a result of this general structure, users are able to design their own hierarchical Monte Carlo estimator and corresponding adaptive algorithm, by appropriately selecting modules within the library. We elaborate on these features more extensively in Chapter 6. In addition to the MLMC estimator, we present in this section some other examples of hierarchical

Monte Carlo estimators in the literature that share this common structure and can/have been implemented within the XMC software library.

2.2.1 Multi-Index Monte Carlo estimators

MIMC estimators [68] generalise the notion of MLMC estimators to more than one discretisation parameter. For example, one may wish to exploit different combinations of space and time discretizations for complexity gains. The notion of levels $l \in \{0, 1, ..., L\}$ is extended to a set of multi-indices $s = (s_1, s_2, ..., s_d)$ where each s_i takes a value in $\{0, 1, ..., L_i\}$. Each s_i corresponds to one level of one discretization parameter. For a multi-index $s \in \mathbb{N}^d$, we denote by Q_s the approximation of Q obtained with a discretisation characterised by $s = (s_1, s_2, ..., s_d)$ and define the following mixed difference operators:

$$\Delta_i Q_s = \begin{cases} Q_s - Q_{s-e_i}, & \text{if } s_i > 0, \\ Q_s, & \text{if } s_i = 0, \end{cases} \tag{2.14}$$

$$\text{and} \quad \Delta Q_s = \Delta_1 \otimes \Delta_2 \otimes ... \otimes \Delta_d Q_s, \tag{2.15}$$

where e_i is the canonical vector whose components are given by $(e_i)_j = \delta_{ij}$. The MIMC estimator is then defined as

$$\hat{\mu}_{\text{mimc}} = \sum_{s \in \mathscr{I}} \frac{1}{N_s} \sum_{i=1}^{N_s} \Delta Q_s^{(i,s)}, \tag{2.16}$$

where $\mathscr{I}$ is a set of indices chosen based on notions of optimal error and cost. Each term $\Delta Q_s^{(i,s)}$ involves at most 2^d computations on different discretisation levels; namely,

$$\Delta Q_s^{(i,s)} = \sum_{j \in \{0,1\}^d} (-1)^{\|j\|_1} Q_{s-j}^{(i,s)}, \tag{2.17}$$

where in Eq. (2.17), we use the convention that $Q_{s-j}^{(i,s)} = 0$ if any of the entries of $s - j$ are negative. The key point in Eq. (2.17) is that all the terms $Q_{s-j}^{(i,s)}$ are computed with the same realisation of the input parameters, whereas the terms $\Delta Q_s^{(i,s)}$ and $\Delta Q_p^{(j,p)}$ are independent if $i \neq j$ or $s \neq p$. Assuming that the mixed differences satisfy the following properties,

$$|\mathbb{E}[\Delta Q_s]| \leq C_\alpha \prod_{i=1}^{d} e^{-\alpha_i s_i}, \tag{2.18a}$$

$$\mathbb{V}\text{ar}(\Delta Q_s) \leq C_\beta \prod_{i=1}^{d} e^{-\beta_i s_i}, \tag{2.18b}$$

$$Cost(\Delta Q_s) \leq C_\gamma \prod_{i=1}^{d} e^{\gamma_i s_i}, \tag{2.18c}$$

for some positive constants C_α, C_β, C_γ and rates $(\alpha_i, \beta_i, \gamma_i), i \in \{1, ..., d\}$, a complexity theorem was proven in [68] for the estimator $\hat{\mu}_{mimc}$, showing substantial improvement over an

MLMC estimator in which all discrete parameters are refined going from one level to another. In other words, the bounds in Eqs. (2.18) assume that the dependence of the above quantities on the discretization parameters factorizes into individual rates along each discretization parameter.

2.2.2 Multi-Fidelity Monte Carlo estimators

The MFMC method is an extension of the concept of levels within the MLMC framework to a more general notion of "fidelity". MFMC estimators have been extensively used as an alternative to MLMC estimators, wherein the hypotheses in Eqs. (2.8) cannot be guaranteed. Typically, MFMC estimators consist of a few simulations conducted on a highly accurate but expensive high-fidelity model, corrected with correlated simulations from several less accurate but relatively cheaper low-fidelity models. In contrast to MLMC estimators, general MFMC estimators do not make any assumptions on the relative accuracy of the high- and low-fidelity models. Instead, they purely use correlation information to compute the optimal variance reduction and corresponding cost-optimal sample allocation.

To demonstrate this concept further, we study the idea of control variates. The MFMC method can be thought of as a generalisation of the concept of control variates. The method of control variates seeks to minimise the variance of the Monte Carlo estimator in Eq. (2.1) as follows. We are once again interested in estimating $\mathbb{E}[Q]$ and, to this end, introduce another random variable Z with a bounded second moment. We define the new random variable $\hat{Q}_\alpha := Q - \alpha(Z - \mathbb{E}[Z])$. By construction, we have that $\mathbb{E}\left[\hat{Q}_\alpha\right] = \mathbb{E}[Q]$. In addition, we have that $\mathbb{V}\mathrm{ar}\left(\hat{Q}_\alpha\right) = \mathbb{V}\mathrm{ar}(Q) - 2\alpha\,\mathbb{C}\mathrm{ov}(Q,Z) + \alpha^2\,\mathbb{V}\mathrm{ar}(Z)$, and that this variance is minimised by choosing the parameter α as:

$$\alpha^* = \underset{\alpha\in\mathbb{R}}{\arg\min}\,\mathbb{V}\mathrm{ar}\left(\hat{Q}_\alpha\right) = \frac{\mathbb{C}\mathrm{ov}(Q,Z)}{\mathbb{V}\mathrm{ar}(Z)}. \tag{2.19}$$

As a result, we have that

$$\mathbb{V}\mathrm{ar}\left(\hat{Q}_\alpha\right) = \min_{\alpha\in\mathbb{R}}\mathbb{V}\mathrm{ar}\left(\hat{Q}_\alpha\right) = \mathbb{V}\mathrm{ar}(Q)\left(1 - \frac{\mathbb{C}\mathrm{ov}(Q,Z)}{\mathbb{V}\mathrm{ar}(Q)\,\mathbb{V}\mathrm{ar}(Z)}\right). \tag{2.20}$$

Once the optimal value α^* has been estimated, possibly through an initial screening phase, the control variate estimator reads:

$$\mathbb{E}[Q] \approx \hat{\mu}_{\mathrm{cv},1} = \frac{1}{N}\sum_{i=1}^{N}\left(Q^{(i)} - \alpha^* Z^{(i)}\right) + \alpha^*\,\mathbb{E}[Z]. \tag{2.21}$$

In the event that $\mathbb{E}[Z]$ is not known exactly, but is inexpensive to evaluate, one can estimate the second term using a richer Monte Carlo estimator as follows:

$$\mathbb{E}[Q] \approx \hat{\mu}_{\mathrm{cv},2} = \frac{1}{N}\sum_{i=1}^{N}\left(Q^{(i)} - \alpha^* Z^{(i)}\right) + \frac{\alpha^*}{M}\sum_{j=1}^{M} Z^{(j)}, \tag{2.22}$$

where $M \gg N$, and the first N samples $\{Z_j^{(i)}\}_{i=1}^N$ in the last term may or may not coincide with the N samples $\{Z_j^{(i)}\}_{i=1}^N\}$ of the control variates. This notion can be extended to multiple control variates $\{Z_j\}_{j=1}^d$ as well, yielding a variety of MFMC estimators. The authors of [103, 101] propose an MFMC estimator of the following form:

$$\mathbb{E}[Q] \approx \hat{\mu}_{\text{mfmc}} = \frac{1}{N}\sum_{i=1}^{N} Q^{(i)} + \sum_{j=1}^{d} \alpha_j \left(\frac{1}{M_j}\sum_{i=1}^{M_j} Z_j^{(i)} - \frac{1}{M_{j-1}}\sum_{i=1}^{M_{j-1}} Z_j^{(i)} \right), \tag{2.23}$$

similar in structure to the MLMC estimator in Eq. (2.5), and the sample sizes $\{M_j\}_{j=1}^d$ are such that $N = M_0 < M_1 < ... < M_d$. In both works, the variance of the MFMC estimators are minimised over the sample sizes M_j and coefficients α_j, constrained to a given computational budget. Closed form expressions are provided in both works for the optimal values of these parameters in terms of the correlation coefficients between the high-fidelity model Q and lower fidelity models Z_j. Additionally, convergence and complexity results were shown in [102] for the estimator $\hat{\mu}_{\text{mfmc}}$, demonstrating that it could replicate the same complexity behaviour of a comparable MLMC estimator. We note that the estimator (2.23) does not quite possess the general structure presented at the beginning of this section. We also highlight that closed form expressions for the optimal parameters of MFMC estimators such as (2.23) are, in general, not readily available in closed form, and require additional assumptions on cost/correlation information.

We present below a hierarchical MFMC estimator that was proposed in [20] for a hierarchy of approximations $\{Q_l\}_{l=0}^L$ to Q:

$$\mathbb{E}[Q] \approx \mathbb{E}[Q_L] \approx \sum_{l=0}^{L} \frac{1}{N_l}\sum_{i=1}^{N_l} \left(Q_l^{(i,l)} - \alpha_{l-1} Q_{l-1}^{(i,l)} \right) \prod_{k=l}^{L-1} \alpha_k. \tag{2.24}$$

Although this estimator exploits less the correlations between the models than the one in Eq. (2.23), and therefore has worse cost-complexity behaviour, it fit the general structure of hierarchical estimators given at the beginning of this section and is easier to calibrate since it relies mostly on level-wise calculations and error estimations. Various strategies were presented in [20] for the calibration of the above estimator. In addition, the estimator was implemented in [5] and applied to a problem of turbulent flow around a building.

MFMC estimators have recently been successfully combined with MLMC estimators and control variates to achieve superior performance. [54] proposed a combined multi-fidelity-multi-level estimator and derived expressions for the optimal values of sample sizes as well as the equivalent of the coefficients α. The resultant estimator was shown to perform better than a comparable MLMC estimator in terms of accuracy. The approach was further applied successfully in [55] to a problem of aerospace engineering. A generalised framework was also proposed in [63] for unifying different MFMC, MLMC and control variate approaches. Notably, the authors of [117, 116] introduced the concept of best linear unbiased estimators, a generalized estimator which combines evaluations from models with different fidelities in a

manner that guarantees the largest variance reduction.

2.3 Extensions and alternatives to the MLMC method

As was reviewed earlier, MLMC methods have been used successfully in literature for computing estimates of $\mathbb{E}[Q]$ where Q is the output of a complex differential model with high-dimensional random inputs. Additionally, multi-level or multi-index methods have also seen use in combination with other sampling strategies than Monte Carlo sampling, and also to estimate statistics other than $\mathbb{E}[Q]$. For completeness, we present here a brief overview in this section of such literature, although we highlight that these methods are external to the focus area of this thesis research and are not elaborated on hereafter.

For alternative statistics, [27] proposed MLMC estimators for higher order central moments of the form $\mathbb{E}[(Q-\mathbb{E}[Q])^p]$, $p \geq 2$, based on biased estimators for the level-wise contributions. Unbiased MLMC estimators were proposed in [86] to estimate higher-order moments, with analogous complexity results as in Eq. (2.12). For rare event estimation, importance sampling estimators have been successfully combined with a multi-level framework in [131]. Lastly, we note that MLMC methods have also been used to estimate distribution and robustness measures such as quantiles or the CDF, albeit to a lesser extent than moments. The relevant literature will be reviewed in Chapter 3, since the main focus of that chapter is on the MLMC estimation of risk-measures.

As an alternative to Monte Carlo sampling, several other sampling methods have also been proposed in combination with multi-level and multi-index estimators. Notably, stochastic collocation [21] has been combined successfully with both multi-level [124, 77, 88] and multi-index [67, 66, 25] strategies. Quasi-Monte Carlo sampling has also been explored within multi-level [61, 87, 71] and multi-index [110, 111] frameworks.

3 MLMC estimators for parametric expectations

As was described in Chapters 1 and 2, it is of great practical interest to estimate by simulation, and with controlled accuracy, the distribution of a random QoI that is the output of a complex differential model with high-dimensional random input noise. MLMC methods, as demonstrated in Chapter 2, have shown significant promise in this area, exhibiting dramatic performance improvements over naive Monte Carlo methods for estimating the expected value of a random QoI. However, since one of the key aims of this thesis is to quantify and estimate the tails of the distribution of the QoI, it is desired instead to quantify summary statistics other than the expected value; namely risk-measures. Risk-measures are commonly used in risk-estimation and risk-averse design applications. Examples include quantiles of a given significance, alternatively known as the VaR, or super quantiles such as the so-called conditional-value-at-risk, which is often used as a risk-measure in stochastic optimisation problems in finance [113, 128]. The application of MLMC methods to estimate such statistics is not as well-developed as for the expected value, as well as the problem of error estimation and adaptive tuning of the MLMC hierarchy. We discuss, in this chapter, the developments presented in [16] and [20] towards the MLMC estimation of parametric expectations, from which estimates of risk-measures such as the VaR and the CVaR could be derived.

Particularly, we follow the approach proposed in [85], which consists of introducing suitable parametric expectations, and deriving the sought after statistics as a post-processing step. Parametric expectations are expectations of the form

$$\Phi(\theta) := \mathbb{E}\left[\phi(\theta, Q)\right]. \tag{3.1}$$

In this work, we follow [85] and use the following particular form for the function ϕ:

$$\phi(\theta, Q) := \theta + \frac{1}{1-\tau}(Q-\theta)^+, \quad X^+ := \max(0, X), \quad \theta \in \Theta \subset \mathbb{R}, \tag{3.2}$$

where $\tau \in (0,1)$ denotes a significance parameter and Θ denotes a suitable interval of interest.

This form has the advantage that after estimating the function Φ and its derivatives

$$\Phi^{(m)}(\theta) := \frac{\partial^m}{\partial \theta^m} \mathbb{E}\left[\phi(\theta, Q)\right], \quad m \in \mathbb{N}, \tag{3.3}$$

the CDF $F_Q(\theta) = \mathbb{E}\left[\mathbb{1}_{Q \geq \theta}\right]$ and the PDF $f_Q(\theta) = F_Q^{(1)}(\theta)$ over the interval Θ, as well as the VaR q_τ and the CVaR c_τ of any significance τ for which $q_\tau \in \Theta$, can be obtained by simple post-processing:

$$\begin{aligned} F_Q(\theta) &= \tau + (1-\tau)\Phi^{(1)}, & q_\tau &= \arg\min_{\theta \in \Theta} \Phi(\theta), \\ f_Q(\theta) &= (1-\tau)\Phi^{(2)}, & c_\tau &= \min_{\theta \in \Theta} \Phi(\theta) = \Phi(q_\tau). \end{aligned} \tag{3.4}$$

On the notation, we comment that $\Phi^{(0)} = \Phi$, and that $\mathbb{1}_{Q \geq \theta}$ denotes the characteristic function which takes on a value of 1 in the interval denoted by the subscript and 0 everywhere else.

We remark that the CDF could also be estimated by direct MLMC estimation of the expectations $F_Q(\theta) = \mathbb{E}\left[\mathbb{1}_{Q \geq \theta}\right]$ for different values of θ. However, using MLMC to estimate the expected value of a discontinuous function can lead to two main issues. The first is that since the function is discontinuous, the rate of decay of the level-wise variances corresponding to Eq. (2.8) but for $\mathbb{E}\left[\mathbb{1}_{Q \geq \theta}\right]$ is significantly reduced in comparison to $\mathbb{E}[Q]$, and could potentially cause non-optimal MLMC performance [12, 59]. In addition, only correlated sample pairs that lie on either side of the discontinuity at θ will contribute to the corresponding MLMC estimator and to estimates of the analogous versions of b_l and V_l in Eq. (2.8) corresponding to $\mathbb{1}_{Q \geq \theta}$. Since this occurs increasingly rarely for finer levels, due to Eq. (2.8), obtaining stable estimates of these quantities may require a excessively large number of samples. The parametric expectation approach overcomes these problems, since the function ϕ in Eq. (3.2) is Lipschitz continuous in Q for all $\theta \in \Theta$.

We briefly review alternative approaches that have been proposed in the literature to use MLMC methods for estimating the distribution of a QoI. A MLMC estimator for the CDF was proposed and analysed in [60] wherein a smoothened approximation to the characteristic function $\mathbb{1}_{Q \geq \theta}$ was used. The MLMC method was also used in [58] for nested conditional expectations from which the VaR and CVaR could be derived. An alternative smoothing of the characteristic function based on the KDE method was proposed in [123], combined with an MLMC estimator wherein stratification based sampling was applied at each level. The authors of [24] combined an approach to locate the discontinuity using a root-finding algorithm, followed by numerical pre-integration. One can also derive the VaR and the CVaR from surrogate distributions derived from moments. For example, in the works [28, 64], a maximum entropy approach was used to estimate the PDF and the VaR using moment estimates from MLMC estimators. The use of MLMC estimators for parametric expectations is still an ongoing research area. The work in [85] built upon the ideas presented in [60], but generalises them further to approximate general parametric expectations. Furthermore, novel MLMC estimators for the characteristic function were presented based on the idea of

pointwise estimation combined with interpolation.

The current work builds upon the theoretical work in [85] and aims at deriving practical algorithms for the MLMC estimators proposed therein. This requires the derivation of reliable and possibly sharp error estimators that can be used to adaptively calibrate the hierarchy of the MLMC estimators to achieve optimal performance, i.e., a computational complexity aligned with the theoretical predictions. An error bound was already presented in [85] based on the use of inverse inequalities. However, this bound results in conservative error estimates that lead to MLMC hierarchies that are impractically expensive to compute. In this work, we propose novel error estimators which are much sharper than those reported previously and can be used for practical engineering purposes. These estimators improve on the large leading constants while preserving their optimal theoretical decay rates as the discretisation is refined. More precisely, we propose a bias error estimator based on a smoothened density as well as a statistical error estimator based on bootstrapping [126]. We then use our novel error estimators to design a continuation MLMC algorithm that successively improves the hierarchy to meet a target tolerance with optimal performance. We show on three numerical tests, including an option pricing problem in finance and a laminar fluid dynamics problem, that our methodology does indeed feature a computational complexity aligned with the theoretical rates presented in [85]. Furthermore, we demonstrate that the methodology is robust in the sense that the true MSE, computed with respect to a reference solution, is always smaller than the prescribed tolerance. We add that the novel MLMC contributions of this work have been implemented in the Python package XMC, available at [5].

The structure of this work is as follows. In Section 3.1, we present the MLMC estimator for the parametric expectation Φ in Eq. (3.1) and introduce a notion of the MSE for Φ and its derivatives. We briefly recall the results of [85] on error bounds for MLMC estimators of parametric expectations and present a simplified complexity result for an optimally tuned MLMC estimator. We also detail the practical aspects of implementing such an error estimator. In Section 3.3, we describe novel error estimators that provide tighter bounds on the true error. We compare the performance of these error estimators with the a priori ones presented in [85] on a simple case for which theoretical results are known. Section 3.4 details an adaptive strategy for selecting the parameters of the hierarchy such that a given tolerance can be achieved on the MSE of the MLMC estimator of $\Phi^{(m)}$, $m \in \{0,1,2\}$, as well as on the MSE of MLMC estimators of the VaR and the CVaR. In particular, Section 3.4.4 recalls a result from [85] to relate the error on derived quantities such as the VaR and the CVaR to the error on Φ and its derivatives. The novel error estimator, the adaptive strategy and the performance of the MLMC algorithm are demonstrated on an array of problems of increasing complexity in Section 5.4. Finally, Section 3.6 offers a conclusion and a discussion on the presented work.

3.1 Multi-Level Monte Carlo approximation of parametric expectations

As presented at the beginning of this chapter, we focus in this work on the problem of approximating parametric expectations of the form in Eqs. (3.2) and (3.3) using the MLMC method. The approach we follow is motivated by [85, 60]. We approximate the parametric expectation Φ and its derivatives $\Phi^{(m)}$ on an interval Θ via the MLMC method as follows: We first consider a set of $n \in \mathbb{N}$ nodes

$$\boldsymbol{\theta} := \{\theta_1, \theta_2, ..., \theta_n\}, \quad \theta_j \in \Theta \subset \mathbb{R}, \quad 1 \leq j \leq n, \quad \theta_j < \theta_{j+1}, \tag{3.5}$$

such that $\Theta = [\theta_1, \theta_n]$. The function Φ is then approximated pointwise at any point $\theta_j \in \Theta$ as

$$\Phi(\theta_j) \approx \mathbb{E}\left[\phi(\theta_j, Q_L)\right] = \mathbb{E}\left[\phi(\theta_j, Q_0)\right] + \sum_{l=1}^{L} \mathbb{E}\left[\phi(\theta_j, Q_l) - \phi(\theta_j, Q_{l-1})\right], \tag{3.6}$$

where each expected value is estimated using a Monte Carlo estimator. We then define the MLMC estimator $\hat{\Phi}_L(\theta_j)$ of $\Phi(\theta_j)$ as

$$\hat{\Phi}_L(\theta_j) := \frac{1}{N_0} \sum_{i=1}^{N_0} \phi(\theta_j, Q_0^{(i,0)}) + \sum_{l=1}^{L} \frac{1}{N_l} \sum_{i=1}^{N_l} \left[\phi(\theta_j, Q_l^{(i,l)}) - \phi(\theta_j, Q_{l-1}^{(i,l)})\right]. \tag{3.7}$$

It is important to note that the same set of random events is used to evaluate the estimator for all θ_j. Finally, we obtain a MLMC estimator $\hat{\Phi}_L$ of the whole function $\Phi : \Theta \to \mathbb{R}$ by interpolating over the pointwise estimates as below:

$$\hat{\Phi}_L = \mathscr{S}_n\left(\hat{\Phi}_L(\boldsymbol{\theta})\right), \tag{3.8}$$

where $\mathscr{S}_n$ denotes an appropriate interpolation operator and $\hat{\Phi}_L(\boldsymbol{\theta})$ denotes the set of pointwise MLMC estimates in Eq. (3.7), that is:

$$\hat{\Phi}_L(\boldsymbol{\theta}) = \{\hat{\Phi}_L(\theta_1), \hat{\Phi}_L(\theta_2), \ldots, \hat{\Phi}_L(\theta_n)\}. \tag{3.9}$$

An estimate of the function derivative of order $m \in \mathbb{N}$ denoted by $\hat{\Phi}_L^{(m)}$ is then obtained by computing the derivative of the resultant interpolated function:

$$\hat{\Phi}_L^{(m)} := \mathscr{S}_n^{(m)}\left(\hat{\Phi}_L(\boldsymbol{\theta})\right) := \frac{\partial^m}{\partial \theta^m} \mathscr{S}_n\left(\hat{\Phi}_L(\boldsymbol{\theta})\right), \tag{3.10}$$

provided that it exists. Throughout this work, cubic spline interpolation with equally spaced interpolation points is used. Hence, we restrict ourselves to $m \in \{0, 1, 2\}$, although other interpolant operators and interpolation points can be used as well [85].

We use the following MSE criterion to quantify the accuracy of the function derivative esti-

mate:

$$\mathrm{MSE}\left(\hat{\Phi}_L^{(m)}\right) := \mathbb{E}\left[\left\|\Phi^{(m)} - \hat{\Phi}_L^{(m)}\right\|_{\mathrm{L}^\infty(\Theta)}^2\right], \quad m \in \{0,1,2\}, \tag{3.11}$$

where the norm $\|f\|_{\mathrm{L}^\infty(\Theta)}$ of a function $f : \Theta \to \mathbb{R}$ is defined as

$$\|f\|_{\mathrm{L}^\infty(\Theta)} := \operatorname*{ess\,sup}_{\theta\in\Theta} |f(\theta)|. \tag{3.12}$$

By the triangle inequality, the MSE can be separated into three terms:

$$\begin{aligned}\mathrm{MSE}\left(\hat{\Phi}_L^{(m)}\right) \leq 3\Bigg\{ &\underbrace{\left\|\Phi^{(m)} - \mathscr{S}_n^{(m)}\left(\Phi(\boldsymbol{\theta})\right)\right\|_{\mathrm{L}^\infty(\Theta)}^2}_{\text{Squared interpolation error}} + \underbrace{\left\|\mathscr{S}_n^{(m)}\left(\Phi(\boldsymbol{\theta}) - \mathbb{E}\left[\hat{\Phi}_L(\boldsymbol{\theta})\right]\right)\right\|_{\mathrm{L}^\infty(\Theta)}^2}_{\text{Squared bias error}} \\ &+ \underbrace{\mathbb{E}\left[\left\|\mathscr{S}_n^{(m)}\left(\hat{\Phi}_L(\boldsymbol{\theta}) - \mathbb{E}\left[\hat{\Phi}_L(\boldsymbol{\theta})\right]\right)\right\|_{\mathrm{L}^\infty(\Theta)}^2\right]}_{\text{Squared statistical error}}\Bigg\} \\ =: 3&\left\{(\mathrm{e}_i^{(m)})^2 + (\mathrm{e}_b^{(m)})^2 + (\mathrm{e}_s^{(m)})^2\right\},\end{aligned} \tag{3.13}$$

where we have used the notation $\mathrm{e}_i^{(m)}$, $\mathrm{e}_b^{(m)}$ and $\mathrm{e}_s^{(m)}$ for the interpolation, bias and statistical errors respectively.

Both the computational cost and accuracy, and thus the complexity, of the MLMC estimator are determined by three different sets of parameters; namely the number of interpolation points n, the level-wise sample size N_l at each level l and the number of levels L. These should be chosen in a cost optimal way based on suitable a priori or a posteriori error estimates. In the next sections, we first review the a priori error estimates and the corresponding complexity analysis from [85], before presenting our new and refined error estimators in Section 3.3.

3.2 A priori error estimates on function derivatives and complexity analysis

We review in this section the a priori estimators derived in [85] for each of the error terms in the MSE bound presented in Eq. (3.13). We review as well the MLMC method described therein to adaptively select the parameters of the hierarchy based on a simplified cost model, for which we also state the corresponding complexity result. The main idea behind the error bounds introduced in [85] is to exploit the properties of the particular form of the function ϕ given in Eq. (3.2) in order to derive an upper bound for the MSE in Eq. (3.13). Since the function $\phi(\theta, Q)$ is uniformly Lipschitz continuous in Q for all $\theta \in \Theta$, we have that

$$\left|\phi(\theta, Q_l) - \phi(\theta, Q_{l-1})\right| \leq C_{lip}\,|Q_l - Q_{l-1}| \qquad \forall\theta \in \Theta, \tag{3.14}$$

with finite Lipschitz constant $C_{lip} = 1/(1-\tau),\ \tau \in (0,1)$. If one can control the decay rates of the expected value and variance of the difference $Q_l - Q_{l-1}$ with level l, then the corresponding statistics of the difference in the function ϕ evaluated at the two levels $\phi(\cdot, Q_l) - \phi(\cdot, Q_{l-1})$ decay as well with the same or better rates in the $\mathrm{L}^\infty(\Theta)$-norm. Consequently, complexity results analogous to those available for MLMC estimators of the simple expectation of Q can be obtained for $\hat{\Phi}_L$.

In [85], inverse inequalities were used to relate the MSE of $\hat{\Phi}_L^{(m)}, m \geq 0$ in Eq. (3.11) to pointwise errors on $\hat{\Phi}_L(\theta_j),\ j \in \{1, ..., n\}$. Particularizing the general a priori bound from [85] to the case of cubic spline interpolation, we obtain:

$$\mathrm{MSE}\left(\hat{\Phi}_L^{(m)}\right) \leq 3\left\{(\bar{\mathrm{e}}_i^{(m)})^2 + (\bar{\mathrm{e}}_b^{(m)})^2 + (\bar{\mathrm{e}}_s^{(m)})^2\right\}, \tag{3.15a}$$

$$\text{where } \bar{\mathrm{e}}_i^{(m)} := C_1(m)\left\|\Phi^{(4)}\right\|_{\mathrm{L}^\infty(\Theta)}\left(\frac{|\Theta|}{n}\right)^{(4-m)}, \tag{3.15b}$$

$$\bar{\mathrm{e}}_b^{(m)} := C_2(m)C_3(n-1)^m b_L, \tag{3.15c}$$

$$\bar{\mathrm{e}}_s^{(m)} := C_2(m)C_3(n-1)^m\sqrt{c(n)\sum_{l=0}^{L}\frac{V_l}{N_l}}, \tag{3.15d}$$

where $|\Theta|$ denotes the size of the domain Θ. Each of the three terms $\mathrm{e}_i^{(m)}, \mathrm{e}_b^{(m)}$ and $\mathrm{e}_s^{(m)}$ in Eq. (3.13) are bounded respectively by the corresponding term $\bar{\mathrm{e}}_i^{(m)}, \bar{\mathrm{e}}_b^{(m)}$ and $\bar{\mathrm{e}}_s^{(m)}$ in Eq. (3.15a) and the constants $C_1(m)$, $C_2(m)$ and C_3 are related to the properties of the cubic spline interpolation operator and are detailed in 3.A, together with some relevant properties of cubic splines. The constant $c(n)$ in Eq. (3.15d) is introduced in [90], further detailed in [62] and reads:

$$c(n) = 2\pi\left(\ln(n+1) + \sqrt{8/\pi}\sum_{k=2}^{n+1} k^{-2}\ln(k)^{-1/2}\right). \tag{3.16}$$

We have also introduced the notation b_l and V_l for the level-wise biases and variances respectively, which are defined as

$$b_l := \left\|\Phi - \mathbb{E}\left[\phi(\cdot, Q_l)\right]\right\|_{l^\infty(\boldsymbol{\theta})}, \quad \text{and} \quad V_l := \mathbb{E}\left[\left\|\phi(\cdot, Q_l) - \phi(\cdot, Q_{l-1})\right\|_{l^\infty(\boldsymbol{\theta})}^2\right], \tag{3.17}$$

where the norm $\|\cdot\|_{l^\infty(\boldsymbol{\theta})}$ is defined for a function $f\colon \Theta \to \mathbb{R}$ evaluated at a set of points $\boldsymbol{\theta} \equiv \{\theta_1, ..., \theta_n\}$ as follows:

$$\|f\|_{l^\infty(\boldsymbol{\theta})} := \max_{1\leq i\leq n}|f(\theta_i)|. \tag{3.18}$$

Note that b_l, V_l and $\left\|\Phi^{(4)}\right\|_{\mathrm{L}^\infty(\Theta)}$ are usually not directly computable in practice. However, it is possible to estimate them reliably from the MLMC samples themselves. This will be discussed later in this section.

Using the a priori bounds derived in Eqs. (3.15a)-(3.15d), we now describe how to select the

optimal values n^*, N_l^* and L^* such that the MSE on the function derivative satisfies a tolerance ϵ^2 split with positive weights w_i, w_b and w_s between the squared interpolation, bias and statistical error terms respectively. The weights are such that $w_i + w_b + w_s = 1$. We define the tolerances ϵ_i^2, ϵ_b^2 and ϵ_s^2 as follows and require each of the terms in Eq. (3.15a) to satisfy their respective tolerances:

$$(\bar{\mathrm{e}}_i^{(m)})^2 \leq \epsilon_i^2 := \frac{w_i \epsilon^2}{3}, \qquad (\bar{\mathrm{e}}_b^{(m)})^2 \leq \epsilon_b^2 := \frac{w_b \epsilon^2}{3}, \qquad (\bar{\mathrm{e}}_s^{(m)})^2 \leq \epsilon_s^2 := \frac{w_s \epsilon^2}{3}, \tag{3.19}$$

The interpolation error is controlled solely by the number of interpolation points, which is therefore selected first, namely as

$$n^* = \left\lceil \left[\frac{C_1(m) \left\| \Phi^{(4)} \right\|_{\mathrm{L}^\infty(\Theta)}}{\epsilon_i} \right]^{\frac{1}{(4-m)}} |\Theta| \right\rceil, \tag{3.20}$$

ensuring that the squared interpolation error is bounded by ϵ_i^2 once n is chosen as in Eq. (3.20). Given n^*, the optimal number of levels L^* is selected to be the smallest level such that the squared bias error satisfies a tolerance ϵ_b^2; namely that $C_2(m)C_3(n^*-1)^m b_{L^*} \leq \epsilon_b$, that is

$$L^* = \min \left\{ K \in \mathbb{N}_0 \colon b_K \leq \frac{\epsilon_b}{C_2(m)C_3(n^*-1)^m} \right\}. \tag{3.21}$$

Lastly, with n^* and L^* fixed, the level-wise sample size N_l at level l is selected to minimise the cost of computing the MLMC estimator

$$\mathrm{Cost}\left(\hat{\Phi}_L^{(m)}\right) \leq \sum_{l=0}^{L^*} N_l(\mathfrak{c}_l + n^* \mathfrak{c}_\phi) + n^* \mathfrak{c}_{\mathrm{int}}, \tag{3.22}$$

subject to the following constraint on the squared statistical error:

$$C_2^2(m)C_3^2(n^*-1)^{2m} c(n^*) \sum_{l=0}^{L^*} \frac{V_l}{N_l} \leq \epsilon_s^2. \tag{3.23}$$

Here, $\mathfrak{c}_l$ is the cost of computing one realisation of the correlated pair of approximations (Q_l, Q_{l-1}) at level l, $\mathfrak{c}_\phi$ is the constant that bounds the cost of evaluating the function $\phi(\theta, Q)$ for any $(\theta, Q) \in \Theta \times \mathbb{R}$ and $\mathfrak{c}_{\mathrm{int}}$ is the cost per interpolation point of constructing the cubic spline interpolant on a uniform grid. In [85], the level-wise sample sizes were selected under the assumption that $\mathfrak{c}_{\mathrm{int}}$ and $\mathfrak{c}_\phi$ were non-zero. However, for the applications addressed in this work, it was found that $\mathfrak{c}_{\mathrm{int}}$ and $\mathfrak{c}_\phi$ are usually negligible in comparison to $\mathfrak{c}_l$. Hence, we select the level-wise sample sizes $\{N_l\}_{l=0}^{L^*}$ based on the simplified cost model

$$\mathrm{Cost}\left(\hat{\Phi}_L^{(m)}\right) \approx \sum_{l=0}^{L^*} N_l \mathfrak{c}_l. \tag{3.24}$$

Consequently, the level-wise sample sizes are selected similar to [57] as follows:

$$N_l^* = \left\lceil \frac{C_2^2(m)C_3^2 c(n^*)(n^*-1)^{2m}}{\epsilon_s^2} \sqrt{\frac{V_l}{\mathfrak{c}_l}} \sum_{k=0}^{L^*} \sqrt{V_k \mathfrak{c}_k} \right\rceil, \quad 0 \le l \le L^*. \tag{3.25}$$

Below, we present a complexity result based on the simplified cost model in Eq. (3.24) using the a priori bounds in Eqs. (3.15a)-(3.15d). This result is a simplified version of the one presented in [85] and is tailored to the use of cubic spline interpolation. We give here the proof for completeness.

Proposition 3.2.1. *Suppose that there exist positive constants α, β, and γ such that $2\alpha \ge \min(\beta,\gamma)$ and that*

(i) b_l decays exponentially with order $\alpha > 0$ in l, in the sense that $b_l \le c_\alpha e^{-\alpha l}$ for some constant $c_\alpha > 0$,

(ii) V_l decays exponentially with order $\beta > 0$ in l, in the sense that $V_l \le c_\beta e^{-\beta l}$ for some constant $c_\beta > 0$,

(iii) the cost to compute each i.i.d. realisation of (Q_l, Q_{l-1}) increases exponentially with rate $\gamma > 0$ in l, in the sense that $\mathfrak{c}_l = Cost(Q_l, Q_{l-1}) \le c_\gamma e^{\gamma l}$ for some constant c_γ,

for all $l \in \mathbb{N}_0$, when $h_{l-1} = sh_l$ for some $s > 1$ and $m \in \{0,1,2\}$. For any $0 < \epsilon < e^{-1}$, the m-th derivative, of the MLMC estimator $\hat{\Phi}_L$ of $\Phi \in C^4(\Theta)$ with the number n of (uniform) nodes chosen according to Eq. (3.20), the maximum number of levels L as in Eq. (3.21), and level-wise sample sizes N_l given by Eq. (3.25), satisfies $MSE\left(\hat{\Phi}_L^{(m)}\right) \le \epsilon^2$ at a computational cost that is bounded by

$$Cost\left(\hat{\Phi}_L^{(m)}\right) \lesssim \log(\epsilon^{-1})\epsilon^{-2-\frac{2m}{4-m}} \begin{cases} 1, & \text{if } \beta > \gamma, \\ \log(\epsilon^{-1})^2, & \text{if } \beta = \gamma, \\ \epsilon^{\frac{\beta-\gamma}{\alpha}\frac{4}{4-m}}, & \text{if } \beta < \gamma. \end{cases}$$

Proof. We begin by considering the choice of the number of interpolation points given by Eq. (3.20). We have that $n = \mathcal{O}\left(\epsilon^{\frac{-1}{4-m}}\right)$. In the light of hypothesis (i), the optimal choice of L is given by

$$L = \left\lceil \frac{1}{\alpha} \log\left[\frac{\sqrt{3}c_\alpha C_2(m)C_3(n-1)^m}{\sqrt{w_b}\epsilon}\right]\right\rceil = \mathcal{O}\left(\log\left(\epsilon^{-\frac{4}{\alpha(4-m)}}\right)\right). \tag{3.26}$$

Using the expression for N_l in Eq. (3.25) in the simplified cost model gives

$$\text{Cost}\left(\hat{\Phi}_L^{(m)}\right) = \sum_{l=0}^{L} N_l \mathfrak{c}_l \le \sum_{l=0}^{L} \mathfrak{c}_l + \frac{3C_2^2(m)C_3^2 c(n)(n-1)^{2m}}{w_s \epsilon^2} \left[\sum_{l=0}^{L} \sqrt{V_l \mathfrak{c}_l}\right]^2, \tag{3.27}$$

where the first term is added to take into account the cost of computing at least one sample per level. Using the hypothesis on the cost $\mathfrak{c}_l$ at level l, it follows from Eq. (3.26) that

$$\sum_{l=0}^{L} \mathfrak{c}_l \leq c_\gamma \sum_{l=0}^{L} e^{\gamma l} = c_\gamma \left[\frac{e^{\gamma L} - e^{-\gamma}}{1 - e^{-\gamma}} \right] = \mathcal{O}\left(\epsilon^{-\frac{\gamma}{\alpha}\frac{4}{4-m}}\right). \tag{3.28}$$

In addition, we use the hypotheses on the variance V_l at level l to write

$$\sum_{l=0}^{L} \sqrt{V_l \mathfrak{c}_l} = \sqrt{c_\beta c_\gamma} \sum_{l=0}^{L} e^{\left(\frac{\gamma-\beta}{2}\right) l} \tag{3.29}$$

$$= \sqrt{c_\beta c_\gamma} \begin{cases} \left[\frac{e^{pL} - e^{-p}}{1 - e^{-p}} \right], \text{ if } \beta \neq \gamma \\ (L+1), \text{ if } \beta = \gamma \end{cases} \tag{3.30}$$

where $p = (\gamma - \beta)/2$. In the event that $\beta > \gamma$, we have $p < 0$. In combination with Eq. (3.26), we have that

$$\left[\frac{e^{pL} - e^{-p}}{1 - e^{-p}} \right] \leq \frac{e^{-p}}{e^{-p} - 1} = \mathcal{O}(1). \tag{3.31}$$

In the event that $\beta < \gamma$, we have that $p > 0$ and hence that

$$\left[\frac{e^{pL} - e^{-p}}{1 - e^{-p}} \right] = \mathcal{O}\left(\epsilon^{\frac{\beta-\gamma}{2\alpha}\frac{4}{4-m}}\right). \tag{3.32}$$

In summary, we can write that

$$\left[\sum_{l=0}^{L} \sqrt{V_l \mathfrak{c}_l} \right]^2 = \begin{cases} \mathcal{O}(1), & \text{if } \beta > \gamma, \\ \mathcal{O}\left(\log(\epsilon^{-1})^2\right), & \text{if } \beta = \gamma, \\ \mathcal{O}\left(\epsilon^{\frac{\beta-\gamma}{\alpha}\frac{4}{4-m}}\right) & \text{if } \beta < \gamma. \end{cases} \tag{3.33}$$

As a final step, we note that $c(n) = \mathcal{O}\left(\log(n)\right) \equiv \mathcal{O}\left(\log(\epsilon^{-1})\right)$ and that $(n-1)^{2m} = \mathcal{O}\left(\epsilon^{\frac{-2m}{4-m}}\right)$. Combining all the terms together, we have that

$$\text{Cost}\left(\hat{\Phi}_L^{(m)}\right) \lesssim \epsilon^{\frac{-\gamma}{\alpha}\frac{4}{4-m}} + \log(\epsilon^{-1})\epsilon^{-2-\frac{2m}{4-m}} \begin{cases} 1, & \text{if } \beta > \gamma, \\ \log(\epsilon^{-1})^2, & \text{if } \beta = \gamma, \\ \epsilon^{\frac{\beta-\gamma}{\alpha}\frac{4}{4-m}}, & \text{if } \beta < \gamma. \end{cases} \tag{3.34}$$

In addition, we require that $2\alpha \geq \min(\beta, \gamma)$ for the complexity to be dominated by the second term alone and not by the first term that quantifies the cost of a single simulation. This can be seen by considering each of the following two cases. In the first case $\beta \geq \gamma$, we have

$$\frac{4\gamma}{(4-m)\alpha} \leq \frac{8}{4-m} \iff 2\alpha \geq \gamma, \tag{3.35}$$

since $m \leq 2$. For the second case $\beta < \gamma$, we have that

$$\frac{4\gamma}{(4-m)\alpha} \leq \frac{8}{4-m} + \frac{4(\gamma-\beta)}{(4-m)\alpha} \iff 2\alpha \geq \beta. \tag{3.36}$$

This completes the proof. □

3.2.1 Practical aspects and tuning of hierarchy parameters

As pointed out earlier, the error bounds in Eqs. (3.15a)-(3.15d) are still not directly computable. To this end, we present below a possible way to estimate the level-wise terms b_l and V_l, as well as the term $\|\Phi^{(4)}\|_{\mathrm{L}^\infty(\Theta)}$ based on the available samples of the MLMC estimator. To estimate the level-wise bias terms b_l, we first note that with the help of Hypothesis (i) from Proposition 3.2.1, we have that

$$\lim_{l\to\infty} \mathbb{E}\left[\phi(\cdot, Q_l)\right] = \Phi \tag{3.37}$$

in $l^\infty(\boldsymbol{\theta})$ and hence, similar to the procedure in [56], one can obtain the heuristic estimate

$$\begin{aligned} b_l &= \left\|\Phi - \mathbb{E}\left[\phi(\cdot, Q_l)\right]\right\|_{l^\infty(\boldsymbol{\theta})} && (3.38)\\ &= \left\|\sum_{k=l+1}^{\infty} \mathbb{E}\left[\phi(\cdot, Q_k)\right] - \mathbb{E}\left[\phi(\cdot, Q_{k-1})\right]\right\|_{l^\infty(\boldsymbol{\theta})} && (3.39)\\ &\approx \frac{\left\|\mathbb{E}\left[\phi(\cdot, Q_l) - \phi(\cdot, Q_{l-1})\right]\right\|_{l^\infty(\boldsymbol{\theta})}}{(e^\alpha - 1)}. && (3.40) \end{aligned}$$

The expectation in Eq. (3.40) is then estimated with a Monte Carlo estimator over the N_l independent and identically distributed correlated sample pairs $\{Q_l^{(i,l)}, Q_{l-1}^{(i,l)}\}_{i=1}^{N_l}$, denoted by $\hat{b}_l$. The variance term V_l can also be computed by replacing the expectation in Eq. (3.17) with a similar sample average estimator $\hat{V}_l$, yielding the following:

$$\begin{aligned} \hat{b}_l &:= \frac{1}{N_l} \frac{\left\|\sum_{i=1}^{N_l} \phi(\cdot, Q_l^{(i,l)}) - \phi(\cdot, Q_{l-1}^{(i,l)})\right\|_{l^\infty(\boldsymbol{\theta})}}{(e^\alpha - 1)}, && (3.41)\\ \hat{V}_l &:= \frac{1}{N_l} \sum_{i=1}^{N_l} \left\|\phi(\cdot, Q_l^{(i,l)}) - \phi(\cdot, Q_{l-1}^{(i,l)})\right\|^2_{l^\infty(\boldsymbol{\theta})}. && (3.42) \end{aligned}$$

To start the estimation procedure, one typically computes a small number of QoI realisations on a pre-fixed small number of levels. Such a small initial hierarchy is called a "screening" hierarchy. The screening hierarchy is typically selected such that it is significantly smaller than the expected optimal hierarchy, so that the computational cost of the screening hierarchy is negligible in comparison to the optimal hierarchy. Using the screening hierarchy, one can then obtain initial estimates of $\hat{b}_l$ and $\hat{V}_l$, as well as their decay rates in the levels l, based on which the optimal number of interpolation points n^*, number of levels L^* and level-wise

sample sizes N_l^* can be selected for a prescribed tolerance ϵ^2 according to Eqs. (3.20), (3.21) and (3.25). A MLMC estimator can then be constructed with the estimated optimal hierarchy, upon which better estimates of $\hat{b}_l$, $\hat{V}_l$ and a better MLMC estimator can be produced in an iterative manner. Such an approach was pioneered in [57].

To compute the optimal hierarchy L^* and N_l^* from Eqs. (3.21) and (3.25), one may need values of $\hat{b}_l$ and $\hat{V}_l$ on levels $L < l \leq L^*$ beyond the current maximum level L used, for which no samples are available. To this end, we fit the theorized models $c_\alpha e^{-\alpha l}$ and $c_\beta e^{-\beta l}$ from Proposition 3.2.1 to $\hat{b}_l$ and $\hat{V}_l$ respectively, for the levels where these estimates are available, using a least squares fit. We then use the level-wise biases and variances predicted by these models instead of the actual estimates in Eqs. (3.21) and (3.25).

For computing the interpolation error bound in Eq. (3.15b), as well as for computing the optimal number of interpolation points in Eq. (3.20), we are required to estimate the norm of the fourth derivative of the function Φ. The estimate of the fourth derivative cannot be computed directly from the interpolant as $\mathscr{S}_n^{(4)}(\hat{\Phi}_L(\boldsymbol{\theta}))$ since $\mathscr{S}_n$ is a cubic spline, hence $\mathscr{S}_n\left(\hat{\Phi}_L(\boldsymbol{\theta})\right) \in C^2(\Theta)$ and the fourth derivative does not exist. We propose instead the use of KDE techniques to solve this issue. Such a KDE smoothing procedure is used extensively through this work and is described in detail in Section 3.2.2.

The procedure to estimate $\left\|\Phi^{(4)}\right\|_{\mathrm{L}^\infty(\Theta)}$ is as follows. We begin by selecting the level $\lceil L/2 \rceil$ from the hierarchy. This level is selected since $N_{\lceil L/2 \rceil}$ is sufficiently large to justify the KDE approach but $\hat{\Phi}_{\lceil L/2 \rceil}$ is also expected to be sufficiently close to Φ. Although there may exist an optimal choice for this level, this particular choice was found to suffice for the applications in this study. A KDE-smoothened function estimate $\Upsilon_{\lceil L/2 \rceil}(\theta) := \mathbb{E}^{kde}_{\lceil L/2 \rceil}\left[\phi(\theta, \cdot)\right]$ of the function $\hat{\Phi}_{\lceil L/2 \rceil}$ is produced according to the procedure described in Section 3.2.2, where $\mathbb{E}^{kde}_l$ is defined in Eq. (3.46) below. The fourth derivative $\Upsilon^{(4)}_{\lceil L/2 \rceil}$ is then computed using a second order central difference approximation where $\Upsilon_{\lceil L/2 \rceil}$ is evaluated on a uniform grid on Θ with $n' \gg n$ points. The norm is also approximated on the same grid:

$$\left\|\Upsilon^{(4)}_{\lceil L/2 \rceil}\right\|_{\mathrm{L}^\infty(\Theta)} \approx \max_{i \in \{1,\ldots,n'\}} \left|\Upsilon^{(4)}_{\lceil L/2 \rceil}(\theta_i)\right|. \tag{3.43}$$

We summarize below the fully computable a priori error estimators:

$$\bar{\mathrm{e}}_i^{(m)} \approx \hat{\mathrm{e}}_i^{(m)} := C_1(m) \left\|\Upsilon^{(4)}_{\lceil L/2 \rceil}\right\|_{\mathrm{L}^\infty(\Theta)} \left(\frac{|\Theta|}{n}\right)^{(4-m)}, \tag{3.44a}$$

$$\bar{\mathrm{e}}_b^{(m)} \approx \hat{\mathrm{e}}_b^{(m)} := C_2(m) C_3 (n-1)^m \hat{b}_L, \tag{3.44b}$$

$$\bar{\mathrm{e}}_s^{(m)} \approx \hat{\mathrm{e}}_s^{(m)} := C_2(m) C_3 (n-1)^m \sqrt{c(n) \sum_{l=0}^{L} \frac{\hat{V}_l}{N_l}}. \tag{3.44c}$$

In Section 3.3.4, we will compare the a priori error estimators described here to the newly developed error estimators introduced in Section 3.3. As will be seen in Section 3.3.4, the a priori error estimators may prove to be too conservative and lead to hierarchies with large values of

L and N_l when selecting these parameters to attain practical tolerances on the MSE. These hierarchies become impractically expensive to simulate. The main advantage of the a priori error estimators is that the bias and variance terms $\hat{b}_l$ and $\hat{V}_l$ computed in the manner described in this section decay exponentially in the levels with the same rate as the underlying QoI Q (see [85]). However, the inequalities used to achieve this favourable property produce large leading constants. We will introduce new error estimators in Section 3.3 that preserve the exponential decay property, and consequently the complexity result in Proposition 3.2.1, while reducing or eliminating these leading constants.

3.2.2 Function derivative estimation by KDE based smoothing

The error estimator Eq. (3.44a) requires estimating the fourth derivative of the function $\Phi_l(\theta) = \mathbb{E}\left[\phi(\theta, Q_l)\right]$. For this, we could first estimate the expected value with a Monte Carlo sample average estimator. As can be seen easily from Eq. (3.1), this estimate produces a piecewise linear function in θ. This, in turn, implies that the first derivative of such a function is piecewise constant, and that second and higher order derivatives do not exist. Using an empirical direct Monte Carlo approach to estimating by Monte Carlo quantities such as $\left\|\Phi^{(4)}\right\|_{\mathrm{L}^\infty(\Theta)}$, which are important to the error estimation and to the procedure of adaptively selecting the hierarchy parameters, is hence not viable.

We propose the use of KDE techniques to remedy this issue. The KDE procedure for constructing derivatives of Φ_l is presented here. An appropriately smoothed probability density function p_l^{kde} of Q_l is constructed using a one dimensional Gaussian kernel centred on each of the N_l fine samples $\{Q_l^{(i,l)}\}_{i=1}^{N_l}$ at level l. The function Φ_l is then approximated as follows:

$$\Phi_l(\theta) = \int \phi(\theta, q) p_l(q) dq \tag{3.45}$$

$$\approx \int \phi(\theta, q) p_l^{kde}(q) dq =: \mathbb{E}_l^{kde}\left[\phi(\theta, Q_l)\right] \tag{3.46}$$

$$\text{where } p_l^{kde}(q) = \frac{1}{N_l} \sum_{i=1}^{N_l} K_{\delta_l}\left(q, Q_l^{(i,l)}\right), \tag{3.47}$$

$K_{\delta_l}(\cdot, \mu)$ denotes the Gaussian kernel with mean μ and bandwidth parameter $\delta_l > 0$. The bandwidth parameter δ_l controls the "width" of the kernel and is related to the covariance of the underlying data. It can in principle be a function of the level l and the sample size N_l. Here, it is chosen according to Scott's rule [121], which ensures that $\delta_l \to 0$ as $N_l \to \infty$. Since the expressions for K_δ and ϕ are known, a closed form expression can be computed for $\mathbb{E}_l^{kde}\left[\phi(\theta, Q_l)\right]$, which is a C^∞ function in θ due to the smoothness of the Gaussian kernel. The smoothed expression can then be evaluated on a fine grid in Θ with $n' \gg n$ points and derivatives can be evaluated exactly, or more conveniently, estimated by finite different formulas.

The KDE procedure will also used in the novel bias estimator proposed in the next section.

In particular, the novel bias estimator requires the estimation of the level-wise quantities $\left\| \mathscr{S}_n^{(m)} \left(\mathbb{E}\left[\phi(\boldsymbol{\theta}, Q_l) - \phi(\boldsymbol{\theta}, Q_{l-1}) \right] \right) \right\|_{\mathrm{L}^\infty(\Theta)}$. For estimating such quantities, which require the computation of derivatives of Φ, the KDE smoothing follows a similar procedure. However, the density is now bivariate, namely characterising the distribution of the two correlated random variables Q_l and Q_{l-1}:

$$\mathbb{E}\left[\phi(\theta, Q_l) - \phi(\theta, Q_{l-1})\right] = \int\int \left[\phi(\theta, q_l) - \phi(\theta, q_{l-1})\right] p_{l,l-1}(q_l, q_{l-1}) dq_l dq_{l-1} \tag{3.48}$$

$$\approx \int\int \left[\phi(\theta, q_l) - \phi(\theta, q_{l-1})\right] p^{kde}_{l,l-1}(q_l, q_{l-1}) dq_l dq_{l-1} \tag{3.49}$$

$$=: \mathbb{E}^{kde}_{l,l-1}\left[\phi(\theta, Q_l) - \phi(\theta, Q_{l-1})\right] \tag{3.50}$$

$$\text{where } p^{kde}_{l,l-1}(q_l, q_{l-1}) = \frac{1}{N_l} \sum_{i=1}^{N_l} K_{\delta_l}\left(q_l, Q_l^{(i,l)}\right) K_{\delta_{l-1}}\left(q_{l-1}, Q_{l-1}^{(i,l)}\right). \tag{3.51}$$

The bandwidth parameters δ_l and δ_{l-1} are chosen according to Scott's rule based on the sample sets $\{Q_l^{(i,l)}\}_{i=1}^{N_l}$ and $\{Q_{l-1}^{(i,l)}\}_{i=1}^{N_l}$, respectively. One consequence of this method of bandwidth parameter selection is that the parameter δ_l will be larger on fine levels where N_l is typically small. Although the joint density $p_{l,l-1}$ will tend to concentrate around the diagonal as l increases, the KDE density may include a significant off-diagonal mass for large values of δ_l and δ_{l-1}. This may induce a larger variance of the estimator in Eq. (3.50) with respect to naively estimating the expectation in Eq. (3.48) using Monte Carlo. The advantage of using Eq. (3.50), over a Monte Carlo estimate of Eq. (3.48), is that one can differentiate the approximation in Eq. (3.50) with respect to θ, since the resulting expression is smooth with respect to θ.

One can also use anisotropic or more complex choices for the kernel, which may require numerical integration or special quadrature. However, for the purposes of this work, the isotropic Gaussian kernel was found to suffice. In addition to being able to compute higher order derivatives of function estimates, the KDE based smoothing approach also provides other important benefits to the error estimation and adaptivity that will be demonstrated in Section 3.3.4.

3.3 Novel error estimators for function derivatives

As will be demonstrated numerically in Section 3.3.4 below, the a priori interpolation error estimator $\hat{\mathrm{e}}_i^{(m)}$ in Eq. (3.44a), provides a satisfactory error bound in practice. Moreover, the interpolation error is often much smaller than the bias and statistical error terms, at least for the cases explored in this work. As a result, we primarily target the accurate estimation of the bias and statistical error terms. In fact, we propose here new estimators for these quantities that provide tighter bounds on the corresponding true errors, while also preserving the same decay rates with respect to l as the a priori level-wise bias and variance contributions b_l and V_l and lead eventually to the complexity bound of Proposition 3.2.1.

3.3.1 Bias term

We begin with the bias term $\mathrm{e}_b^{(m)}$ in the expression for the MSE on $\hat{\Phi}_L^{(m)}$ in Eq. (3.13). The a priori bound $\bar{\mathrm{e}}_b^{(m)}$ from Eq. (3.15c), together with the hypotheses from Proposition 3.2.1, implies that $\mathrm{e}_b^{(m)}$ can be bounded from above as follows,

$$\mathrm{e}_b^{(m)} := \left\| \mathscr{S}_n^{(m)} \left(\Phi(\boldsymbol{\theta}) - \mathbb{E}\left[\hat{\Phi}_L(\boldsymbol{\theta})\right] \right) \right\|_{\mathrm{L}^\infty(\Theta)} \lesssim e^{-\alpha L}, \tag{3.52}$$

where the symbol $\lesssim$ denotes an inequality with a hidden constant, possibly depending on the derivative order m and the number of interpolation points n. Combining this with the implication from Eq. (3.37) implies that the differences $\left\| \mathscr{S}_n^{(m)} \left(\mathbb{E}\left[\hat{\Phi}_l(\boldsymbol{\theta}) - \hat{\Phi}_{l-1}(\boldsymbol{\theta})\right] \right) \right\|_{\mathrm{L}^\infty(\Theta)}$ decay with at least a rate α in the levels l. Then, proceeding as in Eq. (3.40), we can reasonably estimate the bias error as follows:

$$\mathrm{e}_b^{(m)} \approx \frac{\left\| \mathscr{S}_n^{(m)} \left(\mathbb{E}\left[\hat{\Phi}_L(\boldsymbol{\theta}) - \hat{\Phi}_{L-1}(\boldsymbol{\theta})\right] \right) \right\|_{\mathrm{L}^\infty(\Theta)}}{(e^\alpha - 1)}. \tag{3.53}$$

The expectation on the right-hand side could in practice be estimated using a sample average Monte Carlo estimator with the N_L samples on the finest level L, for example:

$$\begin{aligned}
\mathbb{E}\left[\hat{\Phi}_L - \hat{\Phi}_{L-1}\right] &= \mathbb{E}\left[\phi(\cdot, Q_L) - \phi(\cdot, Q_{L-1})\right] \\
&= \int \left[\phi(\cdot, q_L) - \phi(\cdot, q_{L-1})\right] p_{L,L-1}(q_L, q_{L-1}) dq_L dq_{L-1} && (3.54) \\
&\approx \frac{1}{N_L} \sum_{i=0}^{N_L} \phi(\cdot, Q_L^{(i,L)}) - \phi(\cdot, Q_{L-1}^{(i,L)}), && (3.55)
\end{aligned}$$

where we denote the true joint probability density of the bivariate random variable (Q_L, Q_{L-1}) as $p_{L,L-1}$ and have replaced it with the empirical measure induced by the Monte Carlo estimator. As was seen in Section 3.2.2, this approximation causes issues when computing quantities that depend on derivatives of such a Monte Carlo estimator; namely that the first derivative is piecewise constant and that the second and higher derivatives do not exist for such an estimator. This results in level-wise bias estimators that no longer satisfy the decay hypotheses of Proposition 3.2.1.

This problem was solved in the a priori estimator $\hat{\mathrm{e}}_b^{(m)}$ in Eq. (3.44b) by using spline inverse inequalities as follows:

$$\left\| \mathscr{S}_n^{(m)} \left(\mathbb{E}\left[\hat{\Phi}_L(\boldsymbol{\theta}) - \hat{\Phi}_{L-1}(\boldsymbol{\theta})\right] \right) \right\|_{\mathrm{L}^\infty(\Theta)} \leq C_2(m) C_3 (n-1)^m \left\| \mathbb{E}\left[\hat{\Phi}_L - \hat{\Phi}_{L-1}\right] \right\|_{\mathrm{L}^\infty(\Theta)}. \tag{3.56}$$

However, this procedure leads to unacceptably large constants. Here we propose a new estimator that avoids using inverse inequalities and, instead, directly estimates the term $\mathscr{S}_n^{(m)} \left(\mathbb{E}\left[\hat{\Phi}_l - \hat{\Phi}_{l-1}\right] \right)$ using the KDE technique described in Section 3.2.2 to smooth the empirical measure; that is, approximating the unknown joint density $p_{L,L-1}$ by a bivariate KDE smoothed density $p_{L,L-1}^{kde}$ as described in Section 3.2.2.

The resultant novel estimator for $\mathrm{e}_b^{(m)}$ is hence given by

$$\mathrm{e}_b^{(m)} \approx \hat{\mathrm{e}}_{b,new}^{(m)} := \frac{\left\| \mathscr{S}_n^{(m)} \left(\mathbb{E}_{L,L-1}^{kde} \left[\phi(\boldsymbol{\theta}, Q_L) - \phi(\boldsymbol{\theta}, Q_{L-1}) \right] \right) \right\|_{\mathrm{L}^\infty(\Theta)}}{(e^\alpha - 1)} =: \frac{\hat{b}_{L,new}^{(m)}}{(e^\alpha - 1)}, \tag{3.57}$$

where we have defined the level-wise bias terms $\hat{b}_{l,new}^{(m)}, l \in \{0, ..., L\}$. In practice, the decay rate α is estimated by fitting the model $c_\alpha e^{-l\alpha}$ by least squares on the estimates $\hat{b}_{L,new}^{(m)}$ for $l \in \{1, .., L\}$.

3.3.2 Statistical error term

The squared statistical error term in Eq. (3.13) has the form

$$(\mathrm{e}_s^{(m)})^2 = \mathbb{E}\left[\left\| \mathscr{S}_n^{(m)} \left(\hat{\Phi}_L(\boldsymbol{\theta}) - \mathbb{E}\left[\hat{\Phi}_L(\boldsymbol{\theta}) \right] \right) \right\|_{\mathrm{L}^\infty(\Theta)}^2 \right]. \tag{3.58}$$

The a priori bound described in Section 3.2 for the statistical error also suffers from a possibly large leading constant that results in conservative statistical error estimates, as will be highlighted below. As an alternative, we propose the use of a bootstrapping technique [126] to estimate this term as follows. First, observe that a MLMC estimator $\hat{\Phi}_L^{(m)}$ of $\Phi^{(m)}$ is defined through the hierarchy of samples denoted by

$$\mathscr{Q} \equiv \left\{ \{ Q_l^{(i,l)}, Q_{l-1}^{(i,l)} \}_{i=1}^{N_l} \right\}_{l=0}^{L}. \tag{3.59}$$

The idea behind bootstrapping is to create $N_{bs} \in \mathbb{N}$ new MLMC estimators of Φ denoted $\hat{\Psi}_1, \hat{\Psi}_2, ..., \hat{\Psi}_{N_{bs}}$, each defined by a hierarchy of samples of the same size as the original hierarchy $\mathscr{Q}$. For each $\hat{\Psi}_j$, this is done by randomly selecting N_l sample pairs $(\tilde{Q}_l^{(j;i,l)}, \tilde{Q}_{l-1}^{(j;i,l)}) = (Q_l^{(M_{ij},l)}, Q_{l-1}^{(M_{ij},l)})$, $i = \{1, ..., N_l\}$ with $M_{1j}, ..., M_{N_l j} \overset{\text{i.i.d}}{\sim} \mathscr{U}(\{1, ..., N_l\})$ and $j = \{1, ..., N_{bs}\}$ at each level l to define a resampled hierarchy $\mathscr{Q}_j$ by:

$$\mathscr{Q}_j \equiv \left\{ \{ \tilde{Q}_l^{(j;i,l)}, \tilde{Q}_{l-1}^{(j;i,l)} \}_{i=1}^{N_l} \right\}_{l=0}^{L}, \quad j \in \{1, ..., N_{bs}\}.$$

The bootstrapped MLMC estimate $\hat{\Psi}_j$ defined through $\mathscr{Q}_j$ then also provides an estimator of $\Phi^{(m)}$. Using the sample of N_{bs} bootstrapped MLMC estimators, one can approximate the expectations in Eq. (3.58) by sample averages over the bootstrapped MLMC estimators. That is, the statistical error $\mathrm{e}_s^{(m)}$ can be estimated by the bootstrapped estimate $\hat{\mathrm{e}}_{s,new}^{(m)}$ as

$$(\mathrm{e}_s^{(m)})^2 \approx (\hat{\mathrm{e}}_{s,new}^{(m)})^2 := \frac{1}{N_{bs}} \sum_{j=1}^{N_{bs}} \| \mathscr{S}_n^{(m)} \left(\hat{\Psi}_j(\boldsymbol{\theta}) - \bar{\Psi}(\boldsymbol{\theta}) \right) \|_{\mathrm{L}^\infty(\Theta)}^2, \tag{3.60}$$

where $\bar{\Psi}$ denotes the sample average of $\{\hat{\Psi}_j\}_{j=1}^{N_{bs}}$.

The choice of N_{bs} is made adaptively. First, it is set to an initial fixed value. Then, since Eq. (3.60) is a Monte Carlo estimator, the sample variance of the $\mathrm{L}^\infty(\Theta)$-norms is used to es-

timate the MSE of the statistical error estimate. If this MSE exceeds a fixed fraction of the statistical error tolerance ϵ_s^2, the number of bootstrapped samples N_{bs} is doubled, and the process is repeated until the tolerance is satisfied. In addition, since the cost of bootstrapping and interpolating is negligible in comparison to sample generation costs, N_{bs} can be arbitrarily large without a significant additional cost to computing the MLMC estimator $\hat{\Phi}_L^{(m)}$ itself.

3.3.3 Summary of novel error estimator

To summarise the developments above, we have proposed novel bias and statistical error estimators to improve on the properties of the a priori error estimator demonstrated in Section 3.2. The final estimator reads:

$$\begin{aligned}\mathrm{MSE}\left(\hat{\Phi}_L^{(m)}\right) \leq\, & 3C_1^2(m)\left\|\Upsilon^{(4)}_{\lceil L/2\rceil}\right\|^2_{\mathrm{L}^\infty(\Theta)}\left(\frac{|\Theta|}{n}\right)^{2(4-m)} \\ & + 3\frac{(\hat{b}^{(m)}_{L,new})^2}{(e^\alpha-1)^2} + \frac{3}{N_{bs}}\sum_{j=1}^{N_{bs}}\left\|\mathscr{S}_n^{(m)}\left(\hat{\Psi}_j(\boldsymbol{\theta})-\bar{\Psi}(\boldsymbol{\theta})\right)\right\|^2_{\mathrm{L}^\infty(\Theta)}.\end{aligned} \tag{3.61}$$

In fact, we will show in the following section that the new error estimator preserves decay rates of the underlying QoI while reducing or eliminating large leading constants and leading to a tighter error bound.

3.3.4 Demonstration and comparison of error estimators

To demonstrate the performance of the error estimators introduced in the previous sections, we introduce a simple toy problem. Specifically, we consider a random Poisson equation in two spatial dimensions,

$$-\Delta u = f\,, \quad \text{in } D=(0,1)^2\,, \tag{3.62}$$

with homogeneous Dirichlet boundary conditions. The forcing term f is given by

$$f(x) = -C\xi({x_1}^2+{x_2}^2-x_1-x_2)\,, \quad 0\leq x_1,x_2\leq 1\,, \tag{3.63}$$

with ξ being a random variable distributed according to the Beta$(2,6)$ distribution and $C>0$ a positive constant. This problem was also used as a demonstrative example in the companion paper [86] of this work, in order to demonstrate MLMC estimators for higher order central moments. For this forcing term, the solution to the PDE can be computed explicitly and reads

$$u(x_1,x_2) = C\xi x_1x_2(1-x_1)(1-x_2)/2. \tag{3.64}$$

The QoI we consider is the spatial average of the solution, that is

$$Q := \int_D u\,dx = \frac{C}{72}\xi\,. \tag{3.65}$$

For the remainder of the study, we set $C = 432$, leading to $Q = 6\xi$.

Since we have the explicit dependence of the QoI Q on the random input ξ, we can easily compute the exact distribution of Q given that we know the distribution of ξ. In particular, we can compute $\Phi(\theta) = \mathbb{E}\left[\phi(\theta, Q)\right]$ with ϕ as in Eq. (3.2) exactly for any $\tau \in (0,1)$. Indeed, the density p_η of a random variable $\eta = \kappa\xi$ with $\kappa > 0$ reads:

$$p_\eta(x) = \frac{42}{\kappa}\left(1 - \frac{x}{\kappa}\right)^5 \frac{x}{\kappa}, \quad x \in [0, \kappa]. \tag{3.66}$$

Setting $\kappa = 6$ leads to the following form for Φ based on the density p_Q of $Q = 6\xi$:

$$\begin{aligned}\Phi(\theta) &= \theta + \frac{1}{1-\tau}\int_0^6 (q-\theta)^+ p_Q(q)\,dq \\ &= \theta - \frac{(\theta-6)^7(\theta+2)}{373248(1-\tau)}.\end{aligned} \tag{3.67}$$

We plot the true CDF $F_Q(\theta) := \mathbb{P}(Q \le \theta)$ in Fig. 3.1. We also list the true values of the VaR q_τ and the CVaR c_τ for different significances τ in Table 3.1.

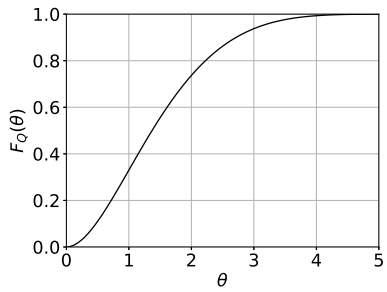


Figure 3.1: CDF F_Q of Q for the Poisson problem.

τ	$q_\tau = F_Q^{-1}(\tau)$	c_τ
0.6	1.611077	2.369803
0.7	1.885696	2.578204
0.8	2.225169	2.843327
0.9	2.715390	3.236473

Table 3.1: VaR and CVaR values for the QoI associated with Poisson problem.

For the numerical assessment of the error estimators, the Poisson problem in Eq. (3.62) is discretised using second order central finite differences on a hierarchy of uniform meshes, where the number of degrees of freedom at level l is given by $(5\times 2^l - 2)^2$. The resultant system is solved directly using sparse LU factorisation. The approximations $\{Q_l\}_{l=0}^L$ are also linear in the random variable ξ and hence, the solution for each discretisation can be precomputed for a fixed value $\xi = 1$ and simply multiplied afterwards by the random realization of ξ to yield the random QoI. In addition, this implies that the "true" function Φ_l is also known in closed-form for all levels.

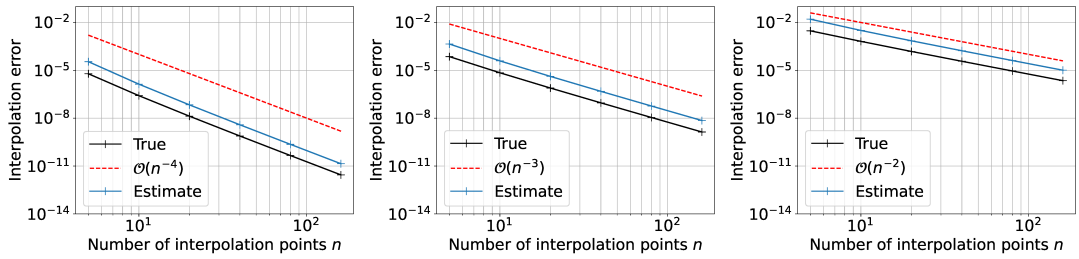
Assessment of the interpolation error estimator

We compute the true interpolation error by considering the true function Φ presented in Eq. (3.67) for $\tau = 0.7$. The interval of interest is selected to be $\Theta \equiv [1.5, 2.5]$, since we expect the 70%-VaR to be within this interval (cf. Table 3.1). The true interpolation error $e_{i,tru}^{(m)}$ of the mth derivative of Φ is given by

$$\mathrm{e}_{i,tru}^{(m)} = \left\| \mathscr{S}_n^{(m)}(\Phi(\boldsymbol{\theta})) - \Phi^{(m)} \right\|_{\mathrm{L}^\infty(\Theta)}.$$

We compare this with the fully a priori error estimate $\hat{e}_i^{(m)}$, introduced in Eq. (3.44a). Instead of the KDE method described in Section 3.2, the norm of the fourth derivative of Φ is estimated using the analytical form of $\Phi^{(4)}$ evaluated on a fine grid, in order to focus solely on the quality of the error estimator.

Fig. 3.2 compares the estimator $\hat{\mathrm{e}}_i^{(m)}$ with the true error for different values of the number of interpolation points n and for different derivatives $\Phi^{(m)}$ for $m \in \{0,1,2\}$. As can be seen from plots in that figure, $\hat{\mathrm{e}}_i^{(m)}$ produces a satisfactory bound on the true error for the range of interpolation points tested, and for all value of $m \in \{0,1,2\}$. The figure also shows that both the true error and the error estimate $\hat{\mathrm{e}}_i^{(m)}$ follow the expected decay, which is $\mathscr{O}(n^{4-m})$.



(a) $m = 0$ (b) $m = 1$ (c) $m = 2$

Figure 3.2: True interpolation error $\mathrm{e}_{i,tru}^{(m)}$ and interpolation error estimator $\hat{\mathrm{e}}_i^{(m)}$ for different numbers of interpolation points n and different order of derivatives m of Φ.

Assessment of bias estimators

To demonstrate the performance of the novel KDE-based bias estimator $\hat{\mathrm{e}}_{b,new}^{(m)}$ described in Section 3.3.1, we compare it with two alternative methods for bias error estimation. The first method is the fully a priori bias estimator $\hat{\mathrm{e}}_b^{(m)}$ given in Eq. (3.44b). The second method is to naively estimate the bias using an empirical mean without KDE smoothing, wherein the pointwise estimates obtained in Eq. (3.48) are directly interpolated. Note that the number of interpolation points is fixed during this study to $n = 10$, which ensures that the interpolation error is much smaller in comparison to the bias as can be inferred from Figs. 3.2 and 3.3. We also fix the level $l = 5$ for this study. The true error is given by

$$\mathrm{e}_{b,tru}^{(m)} = \left\| \Phi_l^{(m)} - \Phi^{(m)} \right\|_{\mathrm{L}^\infty(\Theta)}, \tag{3.68}$$

which we approximate accurately by evaluating the norm on a very fine grid using the known functions Φ_l and Φ and their derivatives. The a priori estimate, as was described in Section 3.2, is given by

$$\hat{\mathrm{e}}_b^{(m)} := \frac{C_2(m)C_3(n-1)^m}{(e^\alpha - 1)}\hat{b}_L, \tag{3.69}$$

where we used the N_l samples available on level l to estimate the expectation. The naive non-smoothened estimate is given instead by

$$\hat{\mathrm{e}}_{b,nai}^{(m)} := \frac{1}{(e^\alpha - 1)}\left\| \mathscr{S}_n^{(m)}\left(\frac{1}{N_l}\sum_{i=1}^{N_l}\phi(\boldsymbol{\theta}, Q_l^{(i,l)}) - \phi(\boldsymbol{\theta}, Q_{l-1}^{(i,l)})\right)\right\|_{\mathrm{L}^\infty(\Theta)}. \tag{3.70}$$

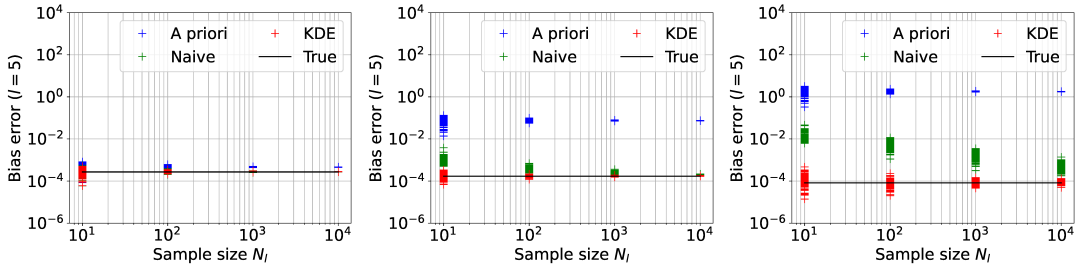
Lastly, the KDE-smoothened error estimator is given by

$$\hat{\mathrm{e}}_{b,new}^{(m)} = \frac{\hat{b}_{L,new}^{(m)}}{(e^\alpha - 1)}, \tag{3.71}$$

where $\hat{b}_{L,new}^{(m)}$ is defined in Eq. (3.57) and computed as described in Section 3.3.1. In both Eqs. (3.70) and (3.71), the norm is evaluated on a fine grid with $n' = 1000$ points. For the decay rate α, we use the theoretical result that for a hierarchy of meshes whose characteristic mesh size decays as 2^{-l} in the levels l, the second order central finite difference scheme yields a bias error decay rate of 2^{-2l}, giving $\alpha = 2\ln(2) \approx 1.39$.

Fig. 3.3 summarises the results on the performance of these bias estimators. We plot each of the error estimators as well as the true error for different sample-sizes N_l for fixed l and for different orders of derivatives of Φ. For each value of N_l, we create 20 independent realizations of N_l correlated sample pairs and for each set of N_l correlated sample pairs, we evaluate the different bias error estimators. We observe that the fully a priori error estimate becomes increasingly conservative for higher order derivatives m. The naive non-smoothened error estimator significantly improves on the a priori estimator but still overestimates the error for higher derivatives and small sample sizes. This is to be expected since we numerically differentiate a non-smooth function. The novel KDE-based approach clearly provides the tightest bound on the true error among the three estimators, consistently for all values of m.

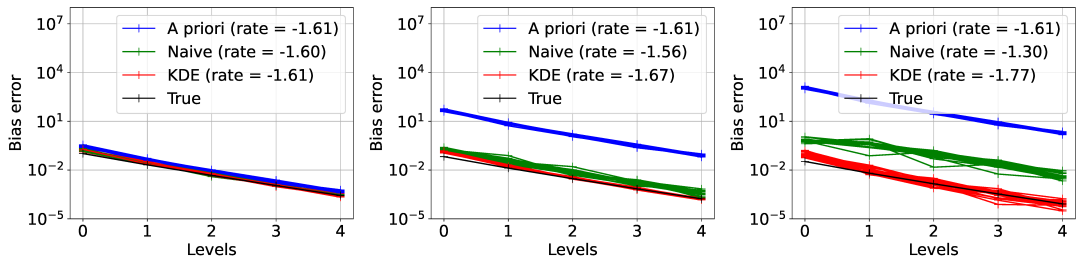
Moreover, the KDE-based approach preserves the underlying decay rate of the QoI with respect to the level l. To illustrate this, we compute a hierarchy that uses 100 samples per level for 5 levels. A hierarchy of this size is sampled independently 20 times and the resulting bias estimates are plotted against levels for each realisation of the hierarchy. These results are summarised in Fig. 3.4. The average least-squares-fit decay rate over these 20 simulations is computed and shown in the corresponding figure legend. We reiterate that theoretical considerations predict that the bias decays at a rate $\alpha = 2\log(2) \approx 1.39$, independent of the order m of the derivative. As can be seen from the figure, the a priori estimates capture the correct decay rate but are conservative on the true error. In addition, they become drastically more conservative for higher order derivatives. The naive approach provides a much tighter bound



(a) $m = 0$ (b) $m = 1$ (c) $m = 2$

Figure 3.3: Comparison of bias error estimators $\hat{e}_b^{(m)}$, $\hat{e}_{b,nai}^{(m)}$ and $\hat{e}_{b,new}^{(m)}$ for different sample sizes and for different derivatives of Φ.

than the a priori approach, but its decay rate deteriorates at least for the second order derivative. The KDE approach, on the other hand, provides the tightest bound while also preserving the correct underlying decay rate.



(a) $m = 0$ (b) $m = 1$ (c) $m = 2$

Figure 3.4: Comparison of bias decay over levels for different derivatives of Φ.

Statistical error comparison

The statistical error is controlled by the level-wise sample sizes. To assess the quality of the novel statistical error estimator that was introduced in Section 3.3.2, we consider three types of hierarchies, all of the general form

$$N_l = \left\lfloor N_0 2^{rl} \right\rfloor, \quad l \in \{0, 1, ..., L\}. \tag{3.72}$$

For the simulations performed in this section, we fix $L = 5$ and consider $r \in \{-1, 0, 1\}$. These values of r generate hierarchies where N_l decreases, stays the same, and increases with l, respectively. Although hierarchies where N_l increases with l do not occur in practice, they are nevertheless investigated here to assess the robustness of the error estimator. By selecting different values of N_0, one can determine the hierarchy fully.

We propose estimating the statistical error as follows. The true statistical error $e_{s,tru}^{(m)}$, which is not readily computable in this case, is estimated using a brute force strategy where we repeat

the MLMC procedure $N_{ref} = 10^4$ times. The squared true error is then estimated as:

$$(e_{s,tru}^{(m)})^2 \approx \frac{1}{N_{ref}} \sum_{i=1}^{N_{ref}} \left\| \mathscr{S}_n^{(m)} \left(\hat{\Phi}_{L,i} - \Phi_L \right) \right\|_{\mathrm{L}^\infty(\Theta)}^2, \tag{3.73}$$

where $\hat{\Phi}_{L,i}$ denotes the ith simulation of an MLMC hierarchy whose parameters are given by Eq. (3.72) for a given r and N_0. For the novel bootstrapped statistical error estimate introduced in Section 3.3.2, we create one realisation of the hierarchy in Eq. (3.72) and create $N_{bs} = 100$ bootstrapped realisations of the resulting MLMC estimator. The statistical error estimate $\hat{e}_{s,new}^{(m)}$ is then computed as described in Section 3.3.2 and in Eq. (3.60). We do not change N_{bs} adaptively in this demonstration and keep the value fixed. We perform the above study for the three different types of hierarchy in Eq. (3.72), and for the first two derivatives of Φ, as well as for Φ itself. For each hierarchy shape and derivative, we test for different values of the hierarchy size parameter N_0.

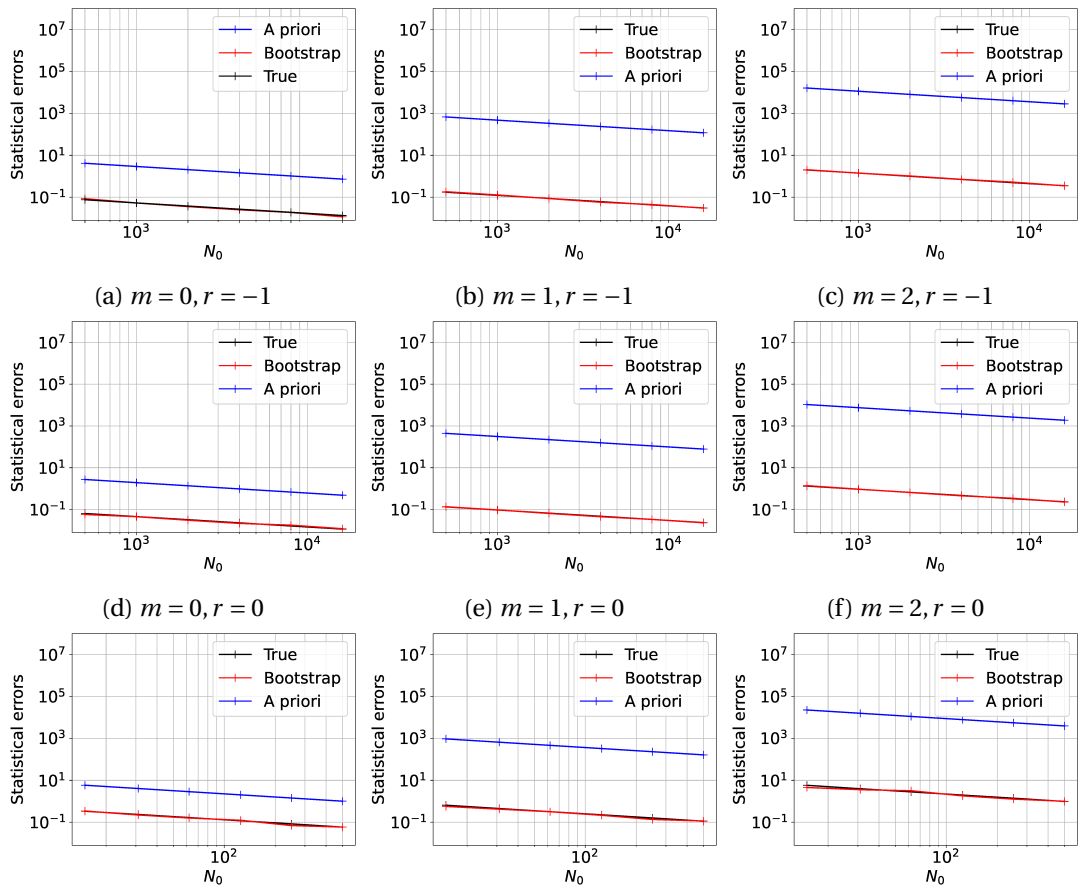
The results are shown in Fig. 3.5. The a priori statistical error estimate given by Eq. (3.44c) and the bootstrapped statistical error estimate given by Eq. (3.60) are plotted alongside the true statistical error for decreasing, uniform and increasing hierarchies and for different derivatives of Φ. As can be observed from the figure, the bootstrapped statistical error estimate provides a tight bound on the true error for the range of hierarchies and derivatives tested, whereas the a priori statistical error estimator defined in Eq. (3.44c) clearly provides overly conservative estimates for $m = 1, 2$.

3.4 Tuning the MLMC hierarchy using the novel error estimators

In the previous sections, we have presented effective error estimators for the MSE contributions of the MLMC estimator of $\Phi^{(m)}$. The next step will be to adapt the MLMC hierarchy based on such an error estimator to achieve a prescribed tolerance on the MSE in a cost-optimal way. This implies that one should choose adaptively the number of interpolation points n, the maximum discretisation level L and the level-wise sample sizes N_l. We discuss a possible way to do this for the MSE on $\hat{\Phi}_L^{(m)}$ and quantities derived from it, focussing on the PDF, the CDF, the VaR and the CVaR as defined in Eq. (3.4). The procedure is, of course, similar to the adaptive procedure described in Section 3.2 for the fully a priori error estimators, but tailored here to the current setting.

3.4.1 MLMC tuning procedure for linear combinations of MSEs

It can be seen from Eq. (3.4) that the MSE of the CDF and the PDF are directly proportional to the MSE of $\hat{\Phi}_L^{(1)}$ and $\hat{\Phi}_L^{(2)}$ respectively. In addition, as will be shown in Section 3.4.4, Lemma 3.4.2, the MSE of the VaR and of the CVaR can be bounded by a linear combination of the MSEs of $\hat{\Phi}_L$ and $\hat{\Phi}_L^{(1)}$. For these reasons, we first present a method to calibrate the MLMC estimator of any arbitrary quantity s_τ with corresponding estimate $\hat{s}_\tau$, whose MSE can be



(a) $m = 0, r = -1$ (b) $m = 1, r = -1$ (c) $m = 2, r = -1$

(d) $m = 0, r = 0$ (e) $m = 1, r = 0$ (f) $m = 2, r = 0$

(g) $m = 0, r = 1$ (h) $m = 1, r = 1$ (i) $m = 2, r = 1$

Figure 3.5: Statistical error estimator comparison of $\hat{\mathrm{e}}_s^{(m)}$ and $\hat{\mathrm{e}}_{s,new}^{(m)}$. From left to right, increasing order of derivative m of Φ. From top to bottom, decreasing, uniform and increasing hierarchies of different sizes.

bounded by linear combinations of the form

$$\mathrm{MSE}\left(\hat{s}_\tau\right) \leq k_0 \mathrm{MSE}\left(\hat{\Phi}_L\right) + k_1 \mathrm{MSE}\left(\hat{\Phi}_L^{(1)}\right) + k_2 \mathrm{MSE}\left(\hat{\Phi}_L^{(2)}\right),\ k_0, k_1, k_2 \geq 0. \tag{3.74}$$

Each of the three terms decomposes into its three respective error contributions, which can then be combined to yield global interpolation, bias and statistical error contributions:

$$\mathrm{MSE}\left(\hat{s}_\tau\right) \leq 3 \left\{ \underbrace{\left[\sum_{m=0}^{2} k_m (e_i^{(m)})^2\right]}_{\text{Squared interpolation error}} + \underbrace{\left[\sum_{m=0}^{2} k_m (e_b^{(m)})^2\right]}_{\text{Squared bias error}} + \underbrace{\left[\sum_{m=0}^{2} k_m (e_s^{(m)})^2\right]}_{\text{Squared statistical error}} \right\}. \tag{3.75}$$

We require the MSE to satisfy a tolerance ϵ^2 with each of the three squared error contributions on the right-hand side of Eq. (3.75) satisfying their corresponding tolerances ϵ_i^2, ϵ_b^2 and ϵ_s^2 as defined in Eq. (3.19). In addition, each of the terms $\mathrm{e}_i^{(m)}$, $\mathrm{e}_b^{(m)}$ and $\mathrm{e}_s^{(m)}$ is estimated using the

error estimators $\hat{\mathrm{e}}_i^{(m)}$, $\hat{\mathrm{e}}_{b,new}^{(m)}$ and $\hat{\mathrm{e}}_{s,new}^{(m)}$ defined in Eqs. (3.44a), (3.57) and (3.60) respectively.

As described in Section 3.2, the interpolation error is controlled solely by the number of interpolation points. To determine it, we require that the squared interpolation error term in Eq. (3.75) satisfies the condition:

$$\left\| \Upsilon^{(4)}_{\lceil L/2 \rceil} \right\|^2_{\mathrm{L}^\infty(\Theta)} \left[\sum_{m=0}^{2} k_m C_1^2(m) \left(\frac{|\Theta|}{n} \right)^{2(4-m)} \right] \leq \epsilon_i^2, \tag{3.76}$$

which results from each term $\mathrm{e}_i^{(m)}$ in the interpolation error contribution of Eq. (3.75) being bounded by its estimator $\hat{\mathrm{e}}_i^{(m)}$ in Eq. (3.44a). Determining n such that equality holds in Eq. (3.76) requires finding the roots of a polynomial of the form $a_8 n^{-8} + a_6 n^{-6} + a_4 n^{-4} + a_0 = 0$ for $a_8, a_6, a_4, a_0 \geq 0$. In case of multiple real roots, the smallest positive root is taken and the optimal number of interpolation points n^* is, in practice, taken to be the smallest integer larger than this root.

The bias error is controlled by the number of levels L. To determine it, we first estimate the level-wise bias terms $\hat{b}_{l,new}^{(m)}$, $l \in \{1, ..., L\}$ for $m \in \{0, 1, 2\}$. We then enforce the condition that the squared bias error term in Eq. (3.75), with each of the terms $\mathrm{e}_b^{(m)}$ replaced by the corresponding estimator $\hat{\mathrm{e}}_{b,new}^{(m)}$, satisfies the tolerance ϵ_b^2. However, we recall that bias estimates are unavailable for levels $l > L$ and are available only on levels where samples have already been computed. As a result, to determine the optimal choice of level L^*, the bias decay models $c_{\alpha_m} e^{-l\alpha_m}$ are first constructed by least squares fits respectively on the level-wise bias estimates $\hat{b}_{l,new}^{(m)}$, $l \in \{1, ..., L\}$ for each $m \in \{0, 1, 2\}$. We then require the squared bias error term, with the terms $\hat{b}_{l,new}^{(m)}$ in $\hat{\mathrm{e}}_{b,new}^{(m)}$ replaced by the corresponding model $c_{\alpha_m} e^{-l\alpha_m}$, to satisfy the following conditions:

$$\sum_{m=0}^{2} \frac{k_m c_{\alpha_m}^2 e^{-2L\alpha_m}}{(e^{\alpha_m} - 1)^2} \leq \epsilon_b^2, \tag{3.77}$$

on L. The appropriate choice of level L^* is selected to be the minimum level that satisfies the above condition in Eq. (3.77). Although all three rates α_m, $m \in \{0, 1, 2\}$ are expected to be the same in most cases, small differences can exist in practice due to estimation errors.

To select the appropriate level-wise sample sizes N_l, we are required to localize the bootstrapped statistical error estimator $\hat{\mathrm{e}}_{s,new}^{(m)}$ over the levels l. We propose an algorithm to accomplish this based on rescaling the level-wise variances $\hat{V}_l$ defined in Eq. (3.42), thus preserving the same squared statistical error splitting between levels as in the case of the a priori error estimator. We first discuss the case of a single term in Eq. (3.75) ($k_m = 1, k_j = 0, j \neq m$). Recall from Section 3.2 that the level-wise sample sizes N_l are selected to minimise the total cost subject to the constraint that the statistical error $\mathrm{e}_s^{(m)}$ satisfies a given tolerance. The a

priori error estimator $\hat{\mathrm{e}}_s^{(m)}$ in Eq. (3.44c) is naturally split over the levels as

$$(\hat{\mathrm{e}}_s^{(m)})^2 = \sum_{l=0}^{L} K(n,m)\frac{\hat{V}_l}{N_l}, \tag{3.78}$$

where $K(n,m) = C_2^2(m)C_3^2(n-1)^{2m}c(n)$, $\{N_l\}_{l=0}^L$ denotes the hierarchy based on which $\hat{\mathrm{e}}_s^{(m)}$ is computed and $\hat{V}_l$ is defined as in Eq. (3.42) and hence, is independent of m. On the other hand, the new error estimator $\hat{\mathrm{e}}_{s,new}^{(m)}$ based on bootstrapping does not provide such an immediate notion of how each level contributes to the overall statistical error and as such provides only a global statistical error estimate. To overcome this limitation, we propose using the error splitting structure of the a priori statistical error estimator in Eq. (3.78), however replacing the large constant $K(n,m)$, which is responsible for the overly conservative error bound of $\hat{\mathrm{e}}_s^{(m)}$ exemplified in Section 3.3.4, with a new one so that the total error matches the computable a posteriori estimator $\hat{\mathrm{e}}_{s,new}^{(m)}$. In particular, we introduce the redefined level-wise variances $\tilde{V}_l$, computed such that

$$(\hat{\mathrm{e}}_{s,new}^{(m)})^2 = \sum_{l=0}^{L} \frac{\tilde{V}_l}{N_l}, \tag{3.79}$$

where each $\tilde{V}_l$ is a rescaled version of $\hat{V}_l$ and the same scaling constant is used across levels. It follows that the rescaled variances $\tilde{V}_l$ decay at the same rate over levels as the a priori variances $\hat{V}_l$. Specifically, we define the rescaled variances $\tilde{V}_l$ as follows:

$$\tilde{V}_l = r_e\hat{V}_l, \quad \text{where } r_e := \frac{(\hat{\mathrm{e}}_{s,new}^{(m)})^2}{\sum_{k=0}^{L}\hat{V}_k/N_k}. \tag{3.80}$$

We refer to r_e as the rescaling ratio. The formulation of the cost optimisation problem then proceeds similarly to Section 3.2 with $\hat{V}_l$ replaced by $\tilde{V}_l$. As before, we neglect the evaluation and interpolation costs $\mathfrak{c}_\phi$ and $\mathfrak{c}_{\text{int}}$ since they are negligible in comparison to $\mathfrak{c}_l$ for the type of applications considered in this work. We require that the statistical error satisfies the prescribed tolerance ϵ_s^2 while minimising the total cost of the simulation. Similar to the constrained optimisation problem for the a priori estimators described in Section 3.2, the approach here yields the following optimal level-wise sample sizes:

$$N_l^* = \left\lceil \frac{1}{\epsilon_s^2}\sqrt{\frac{\tilde{V}_l}{\mathfrak{c}_l}}\sum_{k=0}^{L^*}\sqrt{\tilde{V}_k\mathfrak{c}_k}\right\rceil = \left\lceil \frac{r_e}{\epsilon_s^2}\sqrt{\frac{\hat{V}_l}{\mathfrak{c}_l}}\sum_{k=0}^{L^*}\sqrt{\hat{V}_k\mathfrak{c}_k}\right\rceil. \tag{3.81}$$

We note here that the hierarchy $\{N_l\}_{l=0}^L$ is used to compute the estimator $\hat{V}_l$, $\mathfrak{c}_l$ and $\hat{\mathrm{e}}_{s,new}^{(m)}$, whereas the hierarchy $\{N_l^*\}_{l=0}^L$ is the cost-optimal hierarchy computed based on these estimators that will achieve a tolerance of ϵ_s^2 on the statistical error. Finally, we also note that the variance rescaling proposed in Eq. (3.80) can be extended to a linear combination of errors

as follows:

$$\tilde{V}_l = r_e \hat{V}_l, \quad \text{where } r_e := \frac{\left[\sum_{m=0}^{2} k_m (\hat{\mathrm{e}}_{s,new}^{(m)})^2\right]}{\sum_{k=0}^{L} \hat{V}_k / N_k}, \quad 0 \le l \le L^*. \tag{3.82}$$

We now justify the choice of rescaling factor with the following theoretical result. We first establish upper and lower bounds on the true squared statistical error $\sum_{m=0}^{2} k_m (\mathrm{e}_s^{(m)})^2$ in terms of the true level-wise variances V_l defined in Eq. (3.17).

Lemma 3.4.1. *Let $\hat{\Phi}_L$ be the estimator defined in Eq.* (3.7) *to approximate $\Phi \in C^4(\Theta)$ and $\{V_l\}_{l=0}^{L}$ be the true corresponding level-wise variances as defined in Eq.* (3.17)*. Then there exist positive constants $\lambda(n)$ and $K_{low}(n,m)$ such that:*

$$\frac{\lambda(n)}{|\Theta|} \sum_{l=0}^{L} \frac{V_l}{N_l} \le \sum_{m=0}^{2} k_m (e_s^{(m)})^2 \le \left(\sum_{m=0}^{2} k_m K(n,m) \right) \sum_{l=0}^{L} \frac{V_l}{N_l}. \tag{3.83}$$

Proof. The upper bound follows directly from the a priori statistical error bound introduced in Eq. (3.15d) in Section 3.2. The lower bound is derived as follows. We first define the function $\Gamma := \hat{\Phi}_L - \mathbb{E}\left[\hat{\Phi}_L\right] : \Theta \to \mathbb{R}$. We then have:

$$(\mathrm{e}_s^{(m)})^2 = \mathbb{E}\left[\left\| \mathscr{S}_n^{(m)} (\Gamma(\boldsymbol{\theta})) \right\|_{\mathrm{L}^\infty(\Theta)}^2 \right] \ge \frac{1}{|\Theta|} \mathbb{E}\left[\left\| \mathscr{S}_n^{(m)} (\Gamma(\boldsymbol{\theta})) \right\|_{\mathrm{L}^2(\Theta)}^2 \right], \tag{3.84}$$

since Θ is bounded. The cubic spline interpolant $\mathscr{S}_n(\Gamma(\boldsymbol{\theta}))$ over a set of point evaluations $\Gamma(\boldsymbol{\theta}) \in \mathbb{R}^n$, $\boldsymbol{\theta} = \{\theta_1, ..., \theta_n\}$, can be written as a linear combination of suitable basis functions $\psi_i(\theta)$, $i \in \{1, ..., n\}$:

$$\mathscr{S}_n(\Gamma(\boldsymbol{\theta})) = \sum_{i=1}^{n} \Gamma(\theta_i) \psi_i(\theta). \tag{3.85}$$

This then implies that

$$\left\| \mathscr{S}_n^{(m)} (\Gamma(\boldsymbol{\theta})) \right\|_{\mathrm{L}^2(\Theta)}^2 = \sum_{i,j=1}^{n} \Gamma(\theta_i)\Gamma(\theta_j) \int_\Theta \psi_i^{(m)}(\theta) \psi_j^{(m)}(\theta) d\theta \tag{3.86}$$

$$= \Gamma(\boldsymbol{\theta})^T B^{(m)} \Gamma(\boldsymbol{\theta}), \tag{3.87}$$

where $B^{(m)} \in \mathbb{R}^{n \times n}$ is a matrix whose entries are given by $B_{ij}^{(m)} = \int_\Theta \psi_i^{(m)}(\theta) \psi_j^{(m)}(\theta) d\theta$. It then follows that:

$$\sum_{m=0}^{2} k_m \mathbb{E}\left[\left\| \mathscr{S}_n^{(m)} (\Gamma(\boldsymbol{\theta})) \right\|_{\mathrm{L}^2(\Theta)}^2 \right] = \mathbb{E}\left[\Gamma(\boldsymbol{\theta})^T \left(\sum_{m=0}^{2} k_m B^{(m)} \right) \Gamma(\boldsymbol{\theta}) \right] \ge \lambda \mathbb{E}\left[\|\Gamma(\boldsymbol{\theta})\|_{l^2}^2 \right], \tag{3.88}$$

where $\lambda = \lambda(n) > 0$ denotes the minimum eigenvalue of the positive definite matrix $B =$

$\sum_{m=0}^{2} k_m B^{(m)}$, which is non-zero since B is non-singular. We then finally have that

$$\sum_{m=0}^{2} k_m (\mathrm{e}_s^{(m)})^2 \geq \frac{\lambda(n)}{|\Theta|} \mathbb{E}\left[\left\|\hat{\Phi}_L(\boldsymbol{\theta}) - \mathbb{E}\left[\hat{\Phi}_L(\boldsymbol{\theta})\right]\right\|_{l^2}^2\right] \geq \frac{\lambda(n)}{|\Theta|} \sum_{l=0}^{L} \frac{V_l}{N_l}, \tag{3.89}$$

where in the final inequality, we have used the level-wise independence of samples of the MLMC estimator. This concludes the proof. □

Lemma 3.4.1 shows that the true global squared statistical error $\sum_{m=0}^{2} k_m (\mathrm{e}_s^{(m)})^2$ can be both lower and upper bounded by a constant times the quantity $\sum_{l=0}^{L} V_l / N_l$. Therefore, we expect the rescaling ratio

$$r_e = \frac{\sum_{m=0}^{2} k_m (\hat{\mathrm{e}}_{s,new}^{(m)})^2}{\sum_{l=0}^{L} \frac{\hat{V}_l}{N_l}} \approx \frac{\sum_{m=0}^{2} k_m (\mathrm{e}_s^{(m)})^2}{\sum_{l=0}^{L} \frac{V_l}{N_l}}$$

to remain bounded independent of the hierarchy $\{N_l\}_{l=0}^{L}$.

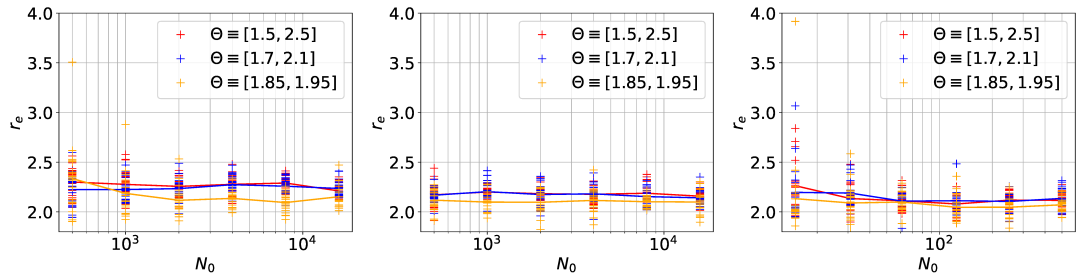
Lastly, since the variances $\hat{V}_l$ are estimated using Monte Carlo sampling, the estimates on finer levels typically have a larger error due to smaller sample-sizes. In addition, estimates of $\hat{V}_l$ and $\mathfrak{c}_l$ may not be available for unexplored levels l. To alleviate this problem, we fit the exponential models $c_\beta e^{-\beta l}$ and $c_\gamma e^{\gamma l}$ on the variances $\tilde{V}_l$ and costs $\mathfrak{c}_l$ respectively for $l \in \{1, ..., L\}$ using a least-squares fit, similar to the procedure described in Section 3.2.1. We use the costs and variances predicted by these models instead of $\tilde{V}_l$ and $\mathfrak{c}_l$ in Eq. (3.81) for the optimal level-wise sample sizes. This stabilises the estimates computed on finer levels, and the models can also be extrapolated for levels where estimates are not available yet. The expression for the optimal level-wise sample sizes is then given by

$$N_l^* = \left\lceil \frac{c_\beta}{\epsilon_s^2} \sqrt{\frac{e^{-\beta l}}{e^{\gamma l}} \sum_{k=0}^{L^*} \sqrt{e^{(\gamma-\beta)k}}} \right\rceil, \quad 0 \leq l \leq L^*. \tag{3.90}$$

3.4.2 Assessment of the stability and behaviour of the rescaling ratio

We now wish to numerically study the behaviour of the rescaling ratio r_e. We therefore consider once again the Poisson problem from Section 3.3.4 and focus instead on the computation of the 70%-CVaR. It will be shown in Section 3.4.4 that the MSE of the PDF, the CDF, the VaR and the CVaR can all be written in the form of Eq. (3.74) with appropriately chosen values of k_0, k_1 and k_2. Particularly, Lemma 3.4.2 in that section derives the values of k_0, k_1 and k_2 for the VaR and the CVaR. Fig. 3.6 shows the variation of r_e for different hierarchy shapes $N_l = N_0 2^{rl}$, $l \in \{0, ..., 5\}$, $r \in \{-1, 0, 1\}$ and for different interval sizes Θ centred approximately around the 70%-VaR. For each value of r, Θ and N_0, we simulate 20 independent random realizations of the hierarchy and plot the values of r_e along with the sample average over the 20 values of r_e. We observe that for nearly all choices of hierarchy, the rescaling ratio r_e is sta-

ble, in the sense that the realizations are clustered about a mean value with a relatively small variance. However, for hierarchies with very small sample sizes N_0 on coarser levels, which contribute proportionately more to the overall statistical error, and for cases with smaller intervals, we observe sporadic large values of r_e. These observations indicate that one needs to select an adequately large sample size and/or interval size in order for the rescaling ratio r_e to be numerically stable. It can be seen from Eq. (3.90) that larger values of r_e in practice lead to larger hierarchies and, hence, more conservative statistical error estimates. It is therefore important to select the interval Θ and sample sizes appropriately.



(a) $r = -1$ (b) $r = 0$ (c) $r = 1$

Figure 3.6: Behaviour of r_e for different hierarchy shapes and interval sizes for Poisson problem

3.4.3 Adaptive MLMC algorithm

It was shown earlier that the cost optimal number of interpolation points n, level-wise sample sizes N_l and the number of levels L could be calculated according to Eqs. (3.76), (3.77) and (3.90) respectively, with knowledge of the quantities $\hat{b}^{(m)}_{l,new}$, $\hat{V}_l$ and $\mathfrak{c}_l$, their corresponding decay rates and constants, as well as $\hat{\mathfrak{e}}^{(m)}_{s,new}$. Since estimates of these quantities are computed using a posteriori computations and require that samples have already been computed, we propose the use of a variation of the CMLMC algorithm introduced in [39] and further adapted for complex simulation problems in [105]. The CMLMC algorithm begins with a small pre-set initial hierarchy, typically called the "screening" hierarchy, and a geometrically decreasing sequence of tolerances $\epsilon_0^2 > \epsilon_1^2 > \cdots > \epsilon_d^2 = \epsilon^2$, where ϵ^2 is the target tolerance to be achieved on the MSE. The method then adapts for the tolerance ϵ_j^2, $j \in \{1, ..., d\}$ based on the estimates from the hierarchy tuned on ϵ_{j-1}^2. For ϵ_0, one uses the estimates from the screening hierarchy. The advantage of this method is that the estimators $b_l^{(m)}$, $\hat{V}_l$, $\hat{\mathfrak{e}}^{(m)}_{s,new}$ and $\mathfrak{c}_l$ are successively improved. This makes the algorithm more robust to inaccurate initial estimates from the screening hierarchy, since the screening hierarchy is typically selected to be much smaller than the optimal hierarchy. The algorithmic description of the CMLMC algorithm is presented in Alg. 3 for a general statistic s_τ whose MSE decomposes as in Eq. (3.75). The reader is referred to [39] for a more detailed exposition.

Algorithm 3: CMLMC Algorithm

Input: Target tolerance $\epsilon > 0$, Number of CMLMC iterations $d \in \mathbb{N}$, Tolerance refinement ratios $\lambda > \kappa > 1$, Error parameters k_0, k_1 and k_2. Set $j = 1$, $\epsilon_a = \epsilon_0$.
Launch screening hierarchy.
Compute estimators $\hat{b}_{l,new}^{(m)}, \hat{V}_l, \mathfrak{c}_l, \hat{\mathrm{e}}_{s,new}^{(m)}$ and model parameters $c_\alpha, c_\beta, c_\gamma, \alpha, \beta, \gamma$.
Compute $\mathrm{MSE}(\hat{s}_\tau)$ based on Eq. (3.75).
while $j \leq d$ **or** $\mathrm{MSE}(\hat{s}_\tau) \geq \epsilon^2$ **do**
    Launch hierarchy with $n^*(\epsilon_a)$, $L^*(\epsilon_a)$, $\{N_l^*(\epsilon_a)\}_{l=0}^{L^*}$ computed based on Eqs. (3.76), (3.77) and (3.90)
    if $j \leq d$ { Set $\epsilon_a = \epsilon\lambda^{(d-j)}$ } **else** { Set $\epsilon_a = \epsilon\kappa^{(d-j)}$ }
    Compute estimators $\hat{b}_{l,new}^{(m)}, \hat{V}_l, \mathfrak{c}_l, \hat{\mathrm{e}}_{s,new}^{(m)}$ and model parameters $c_\alpha, c_\beta, c_\gamma, \alpha, \beta, \gamma$
    Compute $\mathrm{MSE}(\hat{s}_\tau)$ based on Eq. (3.75)
    Update $j \leftarrow j + 1$
end while

3.4.4 Error bounds on the PDF, the CDF, the VaR and the CVaR

It follows directly from Eq. (3.74) that the MSE of the PDF $f_Q(\theta)$ and the CDF $F_Q(\theta)$ can be written as follows:

$$\mathrm{MSE}\left(\hat{F}_Q\right) = (1-\tau)^2 \mathrm{MSE}\left(\hat{\Phi}_L^{(1)}\right), \qquad \mathrm{MSE}\left(\hat{f}_Q\right) = (1-\tau)^2 \mathrm{MSE}\left(\hat{\Phi}_L^{(2)}\right), \tag{3.91}$$

where $\hat{F}_Q(\theta) := \tau + (1-\tau)\hat{\Phi}_L^{(1)}$ and $\hat{f}_Q(\theta) := (1-\tau)\hat{\Phi}_L^{(2)}$. As a result, Eq. (3.74) can be used to bound the error on these quantities by selecting k_1 and k_2 appropriately. We now present a simple result to demonstrate that the general form in Eq. (3.74) can also be used also to bound the MSE of the VaR and the CVaR.

Lemma 3.4.2. *Let $\hat{\Phi}_L$ be the multilevel Monte Carlo estimator defined in Eq.* (3.7) *to approximate Φ. If there exist $\hat{q}_\tau, q_\tau \in \Theta$ such that $\hat{\Phi}_L^{(1)}(\hat{q}_\tau) = \Phi^{(1)}(q_\tau) = 0$ for some given $\tau \in \mathbb{R}$, then it holds that*

$$\mathbb{E}\left[\left|\hat{q}_\tau - q_\tau\right|^2\right] \leq \left\| \frac{1}{\Phi^{(2)}} \right\|^2_{\mathrm{L}^\infty([\hat{q}_\tau, q_\tau])} MSE\left(\hat{\Phi}_L^{(1)}\right), \tag{3.92}$$

as well as that

$$\mathbb{E}\left[|\hat{c}_\tau - c_\tau|^2\right] \leq 2\left\|\Phi^{(1)}\right\|^2_{\mathrm{L}^\infty([\hat{q}_\tau, q_\tau])} \left\| \frac{1}{\Phi^{(2)}} \right\|^2_{\mathrm{L}^\infty([\hat{q}_\tau, q_\tau])} MSE\left(\hat{\Phi}_L^{(1)}\right) + 2MSE\left(\hat{\Phi}_L\right), \tag{3.93}$$

where $\hat{c}_\tau = \hat{\Phi}_L(\hat{q}_\tau)$ and $c_\tau = \Phi(q_\tau)$.

Proof. Let $\hat{q}_\tau, q_\tau \in \Theta$ be such that $\hat{\Phi}^{(1)}(\hat{q}_\tau) = \Phi^{(1)}(q_\tau) = 0$. It then follows from Taylor's theorem that

$$|\hat{q}_\tau - q_\tau||\Phi^{(2)}(\xi)| = |\hat{\Phi}_L^{(1)}(\hat{q}_\tau) - \Phi^{(1)}(\hat{q}_\tau)| \tag{3.94}$$

for some ξ between $\hat{q}_\tau$ and q_τ, so that the first claim follows. The second claim follows from

the first claim upon noting that

$$\begin{aligned}|\hat{c}_\tau - c_\tau| = \left|\hat{\Phi}_L(\hat{q}_\tau) - \Phi(q_\tau)\right| &\leq \left|\Phi(\hat{q}_\tau) - \Phi(q_\tau)\right| + \left|\hat{\Phi}_L(\hat{q}_\tau) - \Phi(\hat{q}_\tau)\right| \\ &\leq \left|\Phi^{(1)}(\zeta)\right||\hat{q}_\tau - q_\tau| + \left|\hat{\Phi}_L(\hat{q}_\tau) - \Phi(\hat{q}_\tau)\right| \end{aligned} \tag{3.95}$$

in view of Taylor's theorem for some ζ between $\hat{q}_\tau$ and q_τ. □

From Lemma 3.4.2, it is evident that q_τ and $\hat{q}_\tau$ represent the true and estimated VaR, while c_τ and $\hat{c}_\tau$ represent the true and estimated CVaR, respectively. We can then derive MSE bounds for the VaR and the CVaR by setting the constants k_0, k_1 and k_2 in Eq. (3.75) based on Lemma 3.4.2. A closed form expression for the solution of Eq. (3.76) for the number of interpolation points can be derived since Lemma 3.4.2 implies that some of the constants k_0, k_1 and k_2 are zero for each of the VaR and the CVaR. For the number of levels L and level-wise sample sizes N_l, the methods described earlier in this section can be directly used with the appropriate values of the constants k_0, k_1 and k_2. Since we expect the interval $[q_\tau, \hat{q}_\tau]$ to be small, we replace each of the constants $\left\|\Phi^{(1)}\right\|_{\mathrm{L}^\infty([q_\tau,\hat{q}_\tau])}$ with $|\hat{\Phi}_L^{(1)}(\hat{q}_\tau)|$ and $\left\|1/\Phi^{(2)}\right\|_{\mathrm{L}^\infty([q_\tau,\hat{q}_\tau])}$ with $|1/\hat{\Phi}_L^{(2)}(\hat{q}_\tau)|$ in practice. Lastly, we note that although k_0, k_1 and k_2 in Eq. (3.75) and in Algorithm 3 are constants, we use the function estimator $\hat{\Phi}_L^{(m)}$ to estimate and update them iteratively within the continuation framework in Algorithm 3.

3.5 Numerical Experiments

We now demonstrate the performance of the above combination of novel error estimators, adaptive strategy and CMLMC algorithm on a set of test cases. Firstly, we consider again the simple Poisson problem introduced in Section 3.3.4. We then study a problem of options contract pricing using the Black-Scholes Stochastic Differential Equation (SDE). Lastly, we study a case of laminar steady fluid flow over a cylinder placed in a channel governed by the Navier-Stokes equations, which demonstrates the methodology on a more applied problem.

3.5.1 Poisson Problem

We consider the same random Poisson equation in two spatial dimensions described in Section 3.3.4. We recall that we have an explicit dependence of Q on the random input ξ for this example and hence, it is straightforward to compute reference values for the VaR and CVaR of different significances (See Tab. 3.1).

The details of the input uncertainties, numerical scheme and discretisation hierarchy are described in Section 3.3.4. We are interested particularly in the estimation of the CVaR with significance $\tau = 0.7$, and hence consider the interval $\Theta = [1.5, 2.5]$ as before. The MLMC estimator proposed in Section 3.1 is used to estimate the parametric expectation Φ. The CVaR estimate is computed from $\hat{\Phi}_L$ as described in Eq. (3.4). The hierarchy is adaptively calibrated as described in Section 3.4 based on the novel error estimators described in Section 3.3. The

CMLMC algorithm described in Section 3.4.3 is then used to successively improve the estimates required to compute the optimal hierarchy with $\lambda = 1.5$ and $\kappa = 1.1$ in Algorithm 3. To compute the statistical error estimate, we initially use $N_{bs} = 100$ bootstrapped samples and then adapt N_{bs} according to the procedure described in Section 3.3.2 to obtain a bootstrap error smaller than 1% of the squared statistical error tolerance. The above combination of problem simulations, MLMC and error estimation have been implemented in the Python package XMC [3].

To assess the robustness of the novel error estimators, a reliability study is conducted. For a given tolerance, the entire MLMC simulation is repeated 20 times independently. For each simulation, an estimate of the CVaR and a corresponding estimate of the MSE are produced. Since the true value of the CVaR is known for this example, we compute the corresponding true squared errors. We expect the MSE estimates to be approximately equal to the sample average of the true squared errors, which we take here as the reference value for the true MSE. The results of this reliability study are shown in Fig. 3.7a. As can be seen from the figure, the error estimates bound the true error on the CVaR and lead to practically computable MLMC hierarchies for the CVaR. For all the tolerances tested, the squared error estimate is not larger than 10 times the squared true error, which we consider acceptable for practical applications (cf. Section 3.3.4).

To verify the predictions of Proposition 3.2.1, we also compute the cost of each MLMC simulation according to Eq. (3.24). The time taken to compute each of the N_l samples is measured, and $\mathfrak{c}_l$ is taken to be their average. The cost is computed using the level-wise sample sizes corresponding to the final iteration of the CMLMC that satisfies the target tolerance and averaged over the 20 repetitions of the algorithm. The results are summarised in Fig. 3.7b, where the average cost over all the simulations for each final CMLMC tolerance is plotted versus the final tolerance.

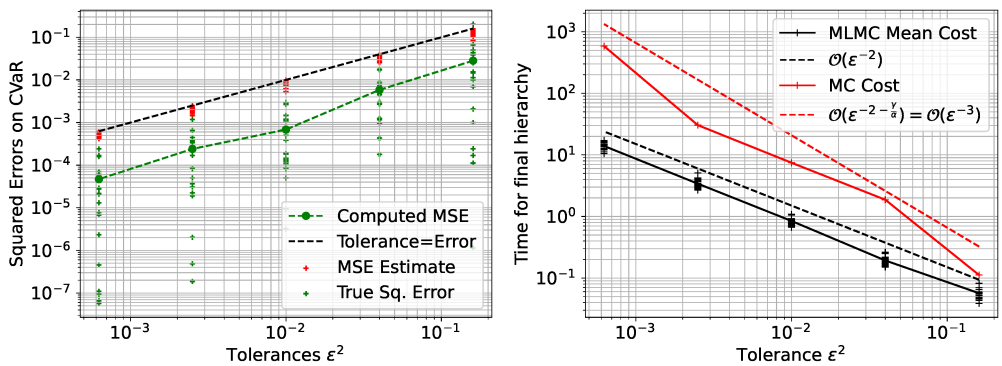
To compare the MLMC estimator with the Monte Carlo method, we propose the following Monte Carlo estimator:

$$\begin{aligned} \hat{\Phi}^{(m)}_{L,mc}(\theta) &:= \mathscr{S}^{(m)}_n\left(\hat{\Phi}_{L,mc}(\boldsymbol{\theta})\right), \quad \theta \in \Theta, \\ \text{where } \hat{\Phi}_{L,mc}(\theta_j) &:= \frac{1}{N}\sum_{i=1}^{N} \phi(\theta_j, Q_L^{(i)}). \end{aligned} \tag{3.96}$$

To estimate the MSE of the CVaR computed from the estimator in Eq. (3.96), we utilise the general MSE form in Eq. (3.75) but for an MLMC estimator with a single level, i.e., without the telescoping summation term in Eq. (3.7). The constants k_0, k_1 and k_2 are chosen according to the results of Lemma 3.4.2 for the CVaR. We now describe a procedure to select the parameters of this estimator such that a prescribed tolerance can be obtained on the corresponding MSE of the CVaR. The number of interpolation points n used in the Monte Carlo estimator is selected to be the same as for the MLMC estimator, since the CMLMC method leaves the number of interpolation points unchanged for all tested tolerances. The discreti-

sation level L is selected to be the same as the finest level predicted by the CMLMC algorithm for the MLMC estimator over all repetitions of the CMLMC algorithm for the given tolerance, although nearly all repetitions predict the same level L for a given tolerance. To predict the correct sample size N, we first note that the squared statistical error term of the CVaR for the Monte Carlo estimator in Eq. (3.96) contains only the single level contribution and hence, is inversely proportional to the sample size N, where the numerator is independent of N and can be estimated using a sample variance estimator. The sample size is then selected such that this statistical error term satisfies the same squared statistical error tolerance ϵ_s^2 as the MLMC estimator. The cost can then be computed in a straightforward manner from N and the cost of a single simulation at level L.

The estimated Monte Carlo cost is shown as well in Fig. 3.7b, together with a least squares fit rate over the estimated Monte Carlo cost. We observe that the predictions made by Proposition 3.2.1 are observed here, namely that the MLMC cost grows as $\mathcal{O}(\epsilon^{-2})$ and that the Monte Carlo cost grows as $\mathcal{O}(\epsilon^{-3})$ for a prescribed tolerance ϵ^2 on the MSE of the CVaR.

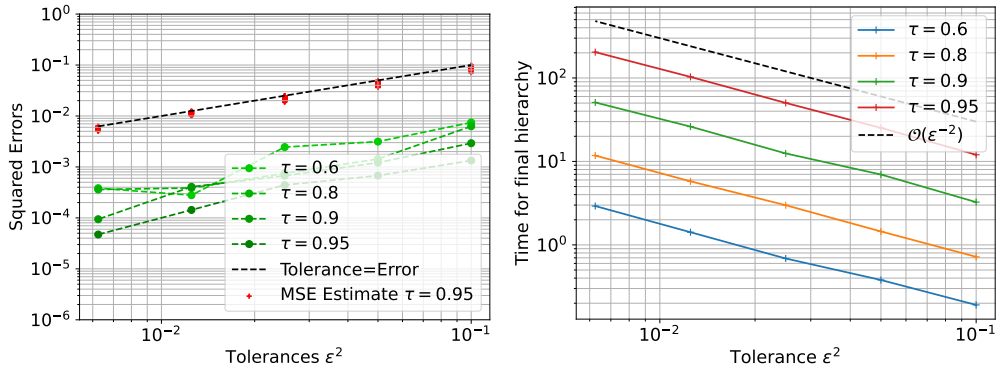


(a) Reliability of error estimator

(b) Complexity behaviour

Figure 3.7: Summary of results for the Poisson problem

In Fig. 3.8, we compare the performance of the error estimation procedure for difference values of the significance $\tau \in \{0.6, 0.8, 0.9, 0.95\}$, while keeping the interval Θ fixed for all tested significances. We perform a reliability study similar to the one in Fig 3.7a. The results are summarized in Fig. 3.8a, which shows the mean of the true squared errors for different significances, as well as the estimated MSEs for $\tau = 0.95$. The MSEs are not plotted for the other tested significances, since they are visually indistinguishable from the $\tau = 0.95$ case. We observe that the estimated MSE is more conservative for higher significances. It is possible that this issue can be ameliorated with an appropriately chosen selection procedure for Θ. In addition, Fig. 3.8b shows the mean cost to compute the optimal hierarchy for a given tolerance for each tested significance. For a given tolerance ϵ^2, the cost scales as $\mathcal{O}(\epsilon^{-2})$ for all significances tested. However, we observe that the constant increases for higher significances. Additionally, we observe that the constant scales approximately as $(1-\tau)^{-2}$, which is to be expected, since all three terms in Eq. (3.13) scale as $(1-\tau)^{-2}$.



(a) Reliability

(b) Complexity

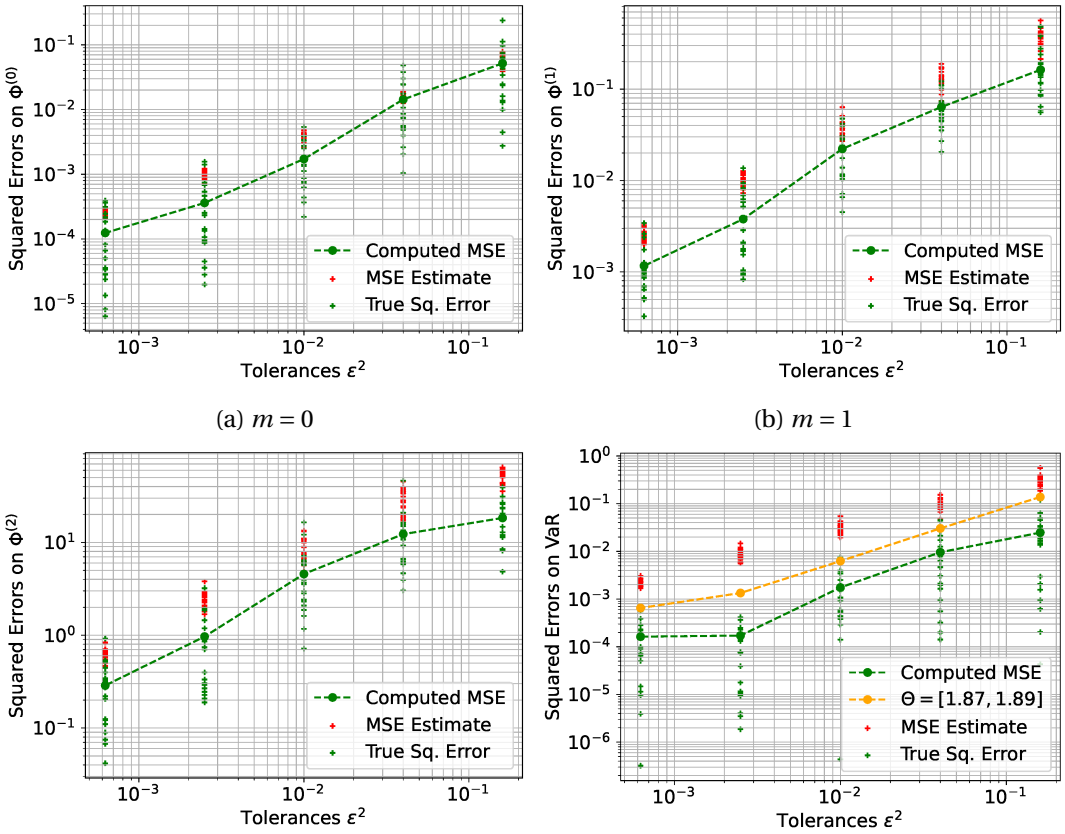
Figure 3.8: Error estimator performance for different significances τ

We now return the discussion to the case of $\tau = 0.7$. Fig. 3.9 compares the true and estimated squared errors on $\Phi^{(m)}$, $m \in \{0,1,2\}$ and the VaR for the same set of simulations as in Fig. 3.7 plotted against the prescribed tolerance used in the CVaR calculation. As can be seen in this figure, a tight bound is obtained on $\Phi^{(m)}$, $m \in \{0,1,2\}$, with a comparatively more conservative estimate on the VaR. The reason for this discrepancy can be explained with Lemma 3.4.2; although the equality in Eq. (3.94) holds true for the function derivative evaluated at the VaR, Eq. (3.92) in turn bounds this with the L^∞ norm over the entire interval Θ. Finally, Fig. 3.9d shows the MSE of the VaR estimated from the same QoI realizations corresponding to the optimal hierarchy for the interval $\Theta = [1.5, 2.5]$, but using a smaller interval $\Theta = [1.87, 1.89]$ such that the 70%-VaR is contained within the interval. It can be seen that choosing a smaller interval around the VaR results in a tighter bound on the true MSE. However, this choice needs to be balanced with the numerical stability of the rescaling ratio r_e in Eq. (3.80) (cf. discussion in Section 3.4.2). The choice of interval Θ hence may have an important effect on the tightness of the error bounds on the VaR and CVaR. In practical applications, however, one does not know a priori the location of the VaR. For the purposes of this study, we only explore fixed intervals Θ and find that the resultant hierarchies are practically computable, leading to effective MLMC estimators. In future works, we plan to explore algorithms that adaptively select Θ.

In Fig. 3.10, we conduct a similar complexity study as the one shown in Fig. 3.7b. We adapt the MLMC hierarchy to achieve a particular relative tolerance on the MSE of each of $\Phi^{(m)}$, $m \in \{0,1,2\}$, as well as the VaR and CVaR. The relative error $\mathrm{e}_r^{(m)}$ of $\hat{\Phi}_L^{(m)}$ and the relative errors $\mathrm{e}_r^{q_\tau}$ and $\mathrm{e}_r^{c_\tau}$ of the VaR and the CVaR respectively, are computed as follows:

$$(\mathrm{e}_r^{(m)})^2 = \frac{\mathrm{MSE}\left(\hat{\Phi}_L^{(m)}\right)}{\left\|\hat{\Phi}_L^{(m)}\right\|^2_{\mathrm{L}^\infty(\Theta)}}, \qquad (\mathrm{e}_r^{\hat{q}_\tau})^2 = \frac{\mathrm{MSE}\left(\hat{q}_\tau\right)}{\hat{q}_\tau^2}, \qquad (\mathrm{e}_r^{\hat{c}_\tau})^2 = \frac{\mathrm{MSE}\left(\hat{c}_\tau\right)}{\hat{c}_\tau^2}. \tag{3.97}$$

We run 20 independent runs of the CMLMC algorithm each for a given statistic and a given



(a) $m = 0$ (b) $m = 1$

(c) $m = 2$ (d) VaR

Figure 3.9: Reliability of error estimator for $\Phi^{(m)}$ and VaR for the Poisson problem

relative tolerance. We plot the average of the cost to compute the optimal hierarchy over these 20 simulations versus the corresponding relative tolerance in Fig. 3.10. As can be seen from Fig. 3.10, higher derivatives of Φ are more expensive to compute for a certain relative tolerance. In addition, for each simulation that was adapted on $\Phi^{(1)}$ for all tolerances considered, we plot the cost of the simulation versus the MSE estimate on the VaR. It can be observed from Fig. 3.10 that for a given budget, adapting the hierarchy on the VaR directly leads to a much lower relative error than adapting on $\Phi^{(1)}$ and computing the VaR as a post-processing step.

Lastly, Fig. 3.11 shows the optimal level-wise sample sizes computed for each intermediate tolerance of one simulation of the CMLMC algorithm aimed at estimating the 70%-CVaR to the finest tolerance simulated. This demonstrates the successive refinement strategy of the CMLMC algorithm, where the hierarchy is calibrated based on a decreasing sequence of tolerances. This can be seen in the increased level-wise sample sizes in the hierarchy with successive iterations. In addition, the red dashed line shows the expected decay rate of N_l over the levels l as predicted by Eq. (3.81), for the variance decay and cost growth exponents β and γ obtained from least squares fits on the estimates of $\tilde{V}_l$ and $\mathfrak{c}_l$ over the levels l, respectively.

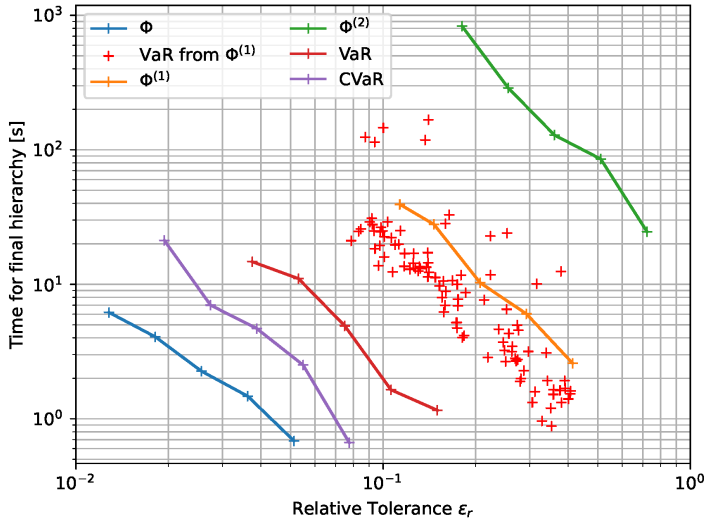


Figure 3.10: Complexity for different statistics

3.5.2 Black Scholes Stochastic Differential Equation

We consider in this section the simulation of the price of a financial asset in time using the Black-Scholes SDE. The price of the asset is modelled as a geometric Brownian motion:

$$dS = rS\,dt + \sigma S\,dW\,, \quad S(0) = S_0,\ t \in (0,T]. \tag{3.98}$$

with $r, \sigma, S_0 > 0$. The QoI for this example, whose distribution we wish to quantify, is the discounted European call option, defined as follows:

$$Q := e^{-rT} \max\big(S(T) - K, 0\big)\,, \tag{3.99}$$

where $K > 0$ denotes the agreed strike price and $T > 0$ the pre-defined expiration date. It is known that the solution $S(T)$ to Eq. (3.98) at time T is a log-normally distributed random variable with mean $S_0 e^{rT}$ and variance ${S_0}^2 e^{2rT}\left(e^{\sigma^2 T} - 1\right)$. Hence, the CDF of Q is:

$$F_Q(\theta) = \begin{cases} \frac{1}{2} + \frac{1}{2}\operatorname{erf}\left(\frac{\sqrt{2}\left(\sigma^2 T - 2rT + 2\ln\left(K + e^{rT}\theta\right) - 2\ln(S_0)\right)}{4\sigma\sqrt{T}}\right), & \theta \geq 0\,, \\ 0\,, & \theta < 0\,, \end{cases} \tag{3.100}$$

$$\text{where} \quad \operatorname{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-s^2}\,ds. \tag{3.101}$$

Using the CDF in Eq. (3.100), it is then straightforward to compute reference values for the VaR and CVaR. Table 3.2 lists the values of the VaR and CVaR for different significances τ and for $r = 0.05$, $\sigma = 0.2$, $T = 1$, $K = 10$, and $S_0 = 10$. The corresponding CDF is plotted in Fig. 3.12. We are interested in estimating the CVaR value corresponding to a significance of $\tau = 0.7$ while ensuring the numerical stability of the rescaling ratio r_e in Eq. (3.82) and hence,

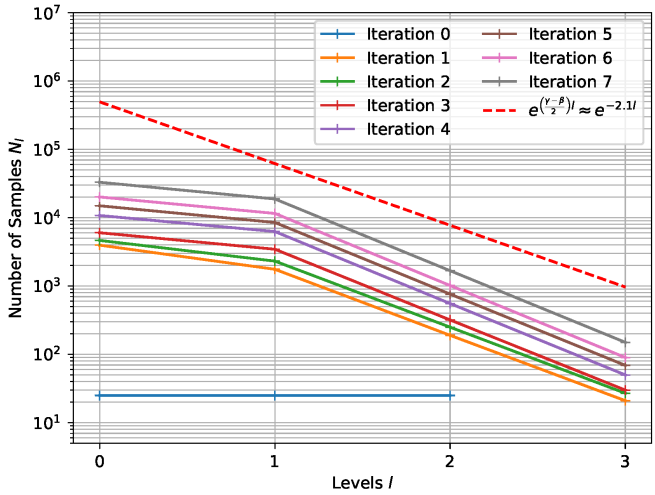


Figure 3.11: Hierarchy evolution over CMLMC iterations

select an interval $\Theta = [0.5, 2.0]$. We utilize the CMLMC algorithm coupled with the novel error estimators described in Section 3.3 in order to calibrate the MLMC estimator for the CVaR.

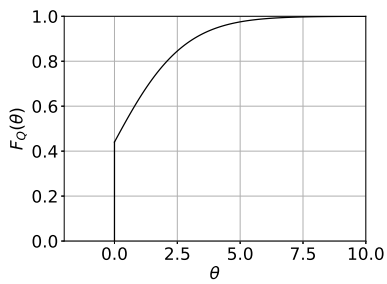


Figure 3.12: CDF F_Q of Q for the SDE problem.

τ	$q_\tau = F_Q^{-1}(\tau)$	c_τ
0.6	0.799151	2.455898
0.7	1.373571	2.914953
0.8	2.086595	3.515684
0.9	3.153379	4.460298

Table 3.2: VaR and CVaR values for the QoI associated with SDE problem.

For the numerical experiments, the SDE in Eq. (3.98) is discretised on a hierarchy of uniform grids given by $t_i^l = i\Delta t_l$, for level $l \in \{0, ..., L\}$, with $i \in \{0, 1, ..., N_T^l\}$ such that $N_T^l \Delta t_l = T$. The grid sizes Δt_l are selected such that $\Delta t_l = \Delta t_0 2^{-l}$, giving rise to a hierarchy of nested grids. Furthermore, we use the Euler Maruyama scheme to discretise the problem on this uniform grid. Denoting the discretised approximation of $S(t_i)$ on level l as $S_{t_i^l}$, the scheme for level l reads:

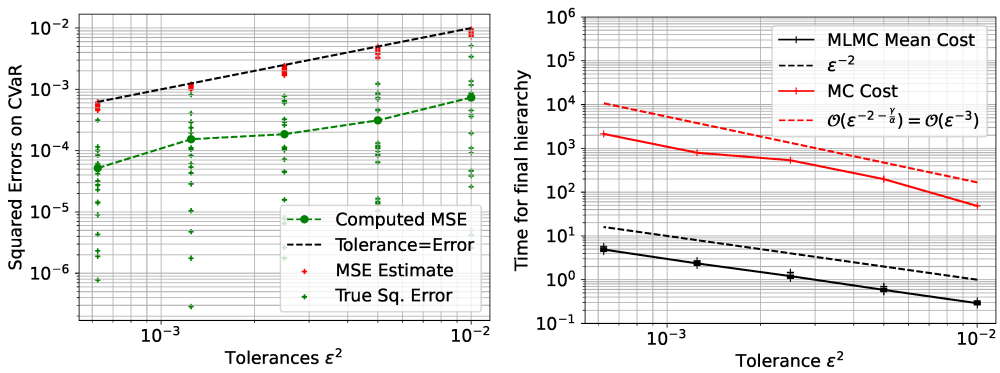
$$S_{t_{i+1}^l} = S_{t_i^l} + r\Delta t_l S_{t_i^l} + \sigma\sqrt{\Delta t_l} S_{t_i^l}\xi_i, \quad \xi_i \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,1), \quad i \in \{0, ..., N_T^l - 1\}, \tag{3.102}$$

with $S_{t_0^l} = S_0$. Correlated realizations are generated on a pair of levels l and $l-1$ by using the same realization of the Brownian path on both levels.

The performance of the MLMC method can be summarized as follows. Fig. 3.13a shows the results of a reliability study analogous to the one conducted in Section 3.5.1 for the Poisson

problem. For each CMLMC simulation, an estimate of the MSE of the CVaR is produced using the novel error estimation procedure described in Sections 3.3 and 3.4. The MSE estimates are compared with the corresponding true squared errors computed using the estimated value of the CVaR obtained from the CMLMC algorithm and the reference value in Table 3.2 corresponding to $\tau = 0.7$. As can be seen from Fig. 3.13a, the estimated MSE is larger than the "true" MSE by a factor of approximately 10, which we consider acceptable for practical applications. In addition, Fig. 3.13b shows the computational cost to compute the optimal hierarchy for a given tolerance on the MSE. The plot demonstrates that the complexity behaviour matches the best case scenario predicted by Proposition 3.2.1. It is also noteworthy that the MLMC estimator not only provides a significantly improved computational complexity of $\mathcal{O}(\epsilon^{-2})$ compared to $\mathcal{O}(\epsilon^{-3})$ for the Monte Carlo method, but also that the computational cost of the MLMC estimator is already smaller by one to two orders of magnitude even for the largest tolerance considered. For comparison, the Monte Carlo cost is plotted in Fig. 3.13b using the procedure as described in Section 3.5.1.

Lastly, we present in Fig. 3.14 a similar study as in Fig. 3.9. For the same set of reliability simulations as in Fig. 3.13a, we compute the true squared error and MSE estimate for $\Phi^{(m)}$, $m \in \{0,1,2\}$, as well as the VaR, and plot it against the tolerance on the MSE used in the CVaR calculation. In addition, for the VaR, shown in Fig. 3.14d, we estimate the MSE also on a smaller interval $\Theta = [1.35, 1.4]$ than the one used for estimating the CVaR for comparison. The results are comparable to the case of the Poisson problem in Section 3.5.1; although the novel error estimators provide accurate estimates of the MSE of $\Phi^{(m)}$, $m \in \{0,1,2\}$, the error estimates for the VaR are comparatively more conservative but improve with smaller interval size selection.



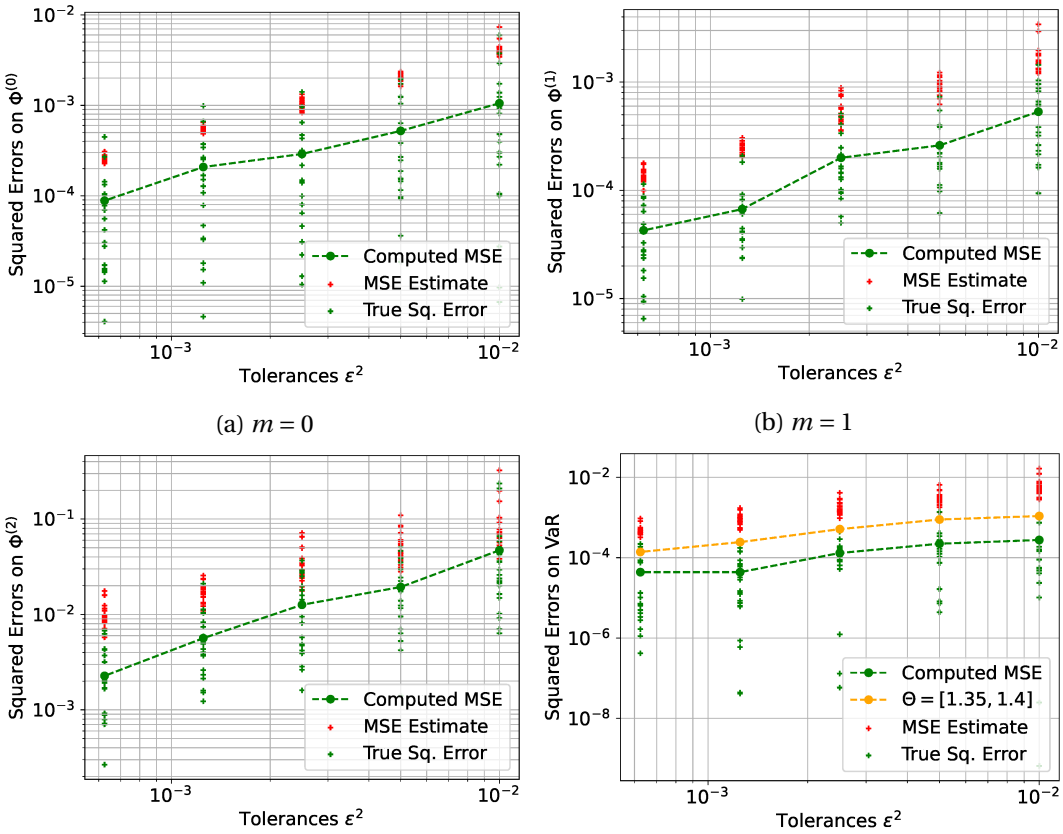
(a) Reliability of error estimator

(b) Complexity behaviour

Figure 3.13: Summary of results for the Black Scholes SDE

3.5.3 Navier-Stokes flow over a cylinder in a channel

To demonstrate the performance of the MLMC estimator and the novel error estimators on a more challenging problem, we consider a two-dimensional steady incompressible fluid flow



(a) $m = 0$ (b) $m = 1$

(c) $m = 2$ (d) VaR

Figure 3.14: Reliability of error estimator for $\Phi^{(m)}$ and VaR for Black Scholes problems

over a cylinder placed asymmetrically in a channel. The goal here is to study the effects of random inlet perturbations on the distributions of force and moment coefficients on the cylinder.

The domain of the problem is a rectangle with a circular cylinder removed and can be defined as $D = [0, 2.2] \times [0, 0.41] \backslash B_r(0.2, 0.2)$, $r = 0.05$, where $B_r(x, y)$ denotes a circle centred at the coordinate $(x, y) \in D \subset \mathbb{R}^2$ with radius $r > 0$. The flow is characterized by the velocity field $u : D \to \mathbb{R}^2$ and pressure field $p : D \to \mathbb{R}$. The velocity and pressure fields are governed by the steady incompressible Navier-Stokes equations:

$$(u \cdot \nabla) u - \nu \Delta u + \nabla p = 0, \tag{3.103}$$

$$\nabla \cdot u = 0, \quad \text{in } D, \tag{3.104}$$

where $\nu = 0.01$ denotes the kinematic viscosity. The boundary conditions are as follows. At the inflow boundary ($x = 0$), we consider a random inlet profile, which consists of a parabolic

mean profile on which harmonics with random amplitudes are added:

$$u(0,y) = \left(\frac{4Uy(0.41-y)}{0.41^2} + u_r, 0\right), \tag{3.105}$$

$$u_r(y) = \sigma \sum_{j=1}^{N_h} \xi_j e^{-j} \sin\left(\frac{j\pi y}{0.41}\right), \qquad \xi_j \overset{i.i.d}{\sim} \mathcal{N}(0,1), \tag{3.106}$$

where $U = 4.0$ is the peak velocity of the parabolic profile, $\sigma = 0.5$ denotes a strength parameter and $N_h = 8$ denotes the number of harmonics superimposed. Fig. 3.15 shows 10 different realizations of the inlet profile given in Eq. (3.106), plotted over the parabolic mean profile.

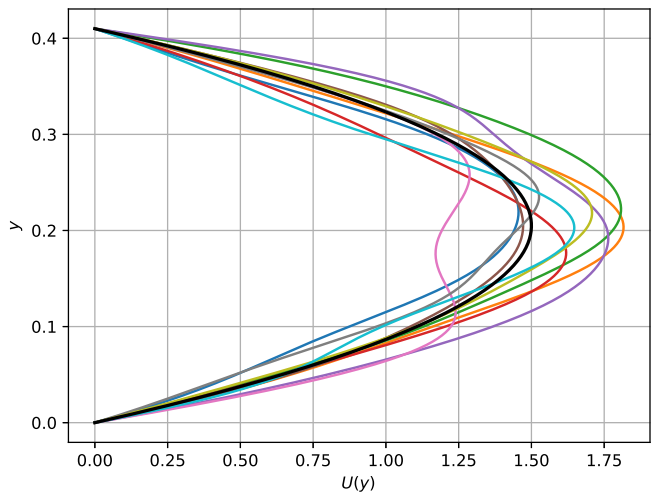


Figure 3.15: Inlet profile realizations (in color) plotted over parabolic mean profile (in black)

On the bottom and top channel walls ($y = 0$ and $y = 0.41$ respectively), no-slip boundary conditions are prescribed. On the outlet ($x = 2.2$), a zero-stress boundary condition is prescribed with the form

$$(\nu \nabla u - pI)n = 0, \tag{3.107}$$

where $n = [1,0]^T$ denotes the outward boundary normal vector and $I \in \mathbb{R}^{2\times 2}$ denotes the identity matrix.

For a peak velocity of $U = 4.0$, the area-weighted inlet velocity is $U_{in} = 2.667$. Taking the reference length to be the diameter of the cylinder, the Reynolds' number is

$$\mathrm{Re} = \frac{2U_{in}r}{\nu} = \frac{2.667 \times 0.1}{0.01} = 26.67. \tag{3.108}$$

The QoI considered is the drag coefficient C_d, whose value is computed as follows. First, we

compute the drag and lift forces F_d and F_l on the cylinder, which are given respectively by:

$$\begin{bmatrix} F_d \\ F_l \end{bmatrix} = \int_{\partial B_r} (\nu \nabla u - pI) n ds, \tag{3.109}$$

where ∂B_r denotes the surface of the circle over which the stress is integrated. The drag coefficient C_d is then computed from the drag force as:

$$C_d = \frac{F_d}{U_{in}^2 r}. \tag{3.110}$$

The domain D is discretised with a non-uniform triangulation. Reference mesh size values are prescribed on the surface of the circle, as well as at the corners of the domain. The coarsest two meshes, corresponding to levels $l = 0$ and $l = 1$ of those simulated, are shown in Fig. 3.16. Each finer level is produced by reducing the prescribed reference mesh sizes by a factor $\sqrt{2}$ from the previous level and re-applying the triangulation. The meshes computed as a result are non-nested. Table 3.3 shows the minimum and maximum mesh sizes h_{min} and h_{max}, as well as the number of vertices for each of the meshes considered in the hierarchy. As can be seen from the table, the number of vertices approximately doubles with every level.

The problem is implemented using the FEniCS finite element software [92]. P2-P1 Taylor-Hood elements are used for the velocity and pressure fields. The resulting non-linear problem is solved using Newton iterations with a relative tolerance of 10^{-10} on the residual. Linear systems are solved using a sparse direct solver [9, 8].

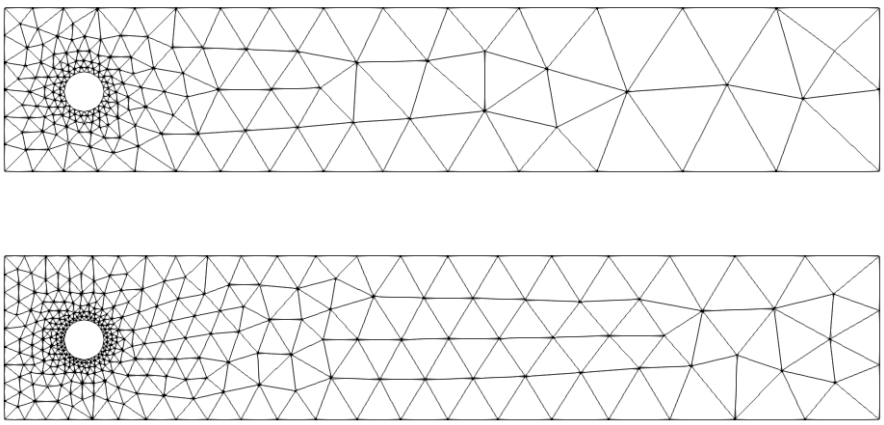


Figure 3.16: Meshes for cylinder problem for level $l = 0$ (top) and $l = 1$ (bottom)

As in previous sections, we aim to estimate the 70%-CVaR using the CMLMC method presented in this work, for which we select the interval $\Theta = [0.3, 0.8]$. We perform a reliability study identical to the ones conducted in Section 3.5.1 and Section 3.5.2 for the Poisson and Black Scholes problems, respectively. The reference value of the 70%-CVaR is, however, not available in this case. Instead, a numerical reference is computed as follows. We conduct 20 independent repetitions of the CMLMC algorithm, tuned to a tolerance that is one quarter

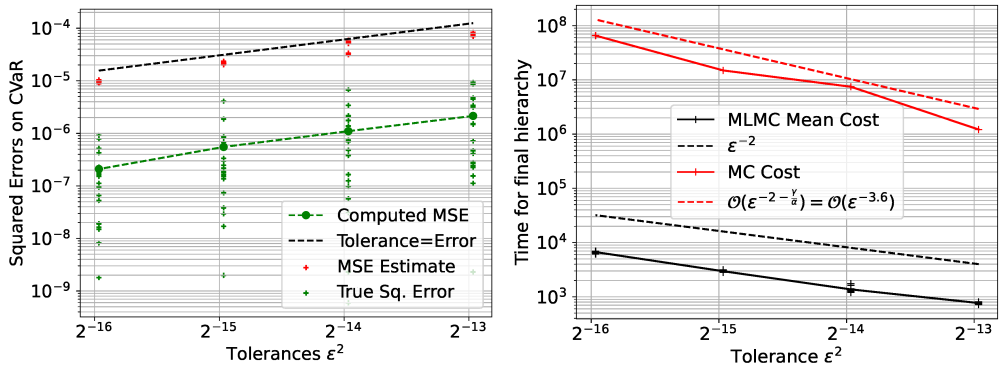
Level	h_{min}	h_{max}	Vertices
0	1.31×10^{-2}	2.65×10^{-1}	199
1	9.80×10^{-3}	1.87×10^{-1}	333
2	6.54×10^{-3}	1.44×10^{-1}	593
3	4.91×10^{-3}	9.82×10^{-2}	1073
4	2.91×10^{-3}	7.58×10^{-2}	2038
5	2.28×10^{-3}	5.62×10^{-2}	3857

Table 3.3: Mesh parameters for the Navier Stokes problem

of the finest tolerance tested for in what follows. The reference value is then taken to be the average over the 20 estimates of the CVaR produced by these simulations. The resultant reliability plot is shown in Fig. 3.17a. As can be seen in Fig. 3.17a, the estimated squared errors are approximately 1.5 orders conservative on the true MSE. Although considerably more conservative than for the Poisson and Black-Scholes problems, we deem the error estimator still acceptable and leading to practically computable hierarchies. In addition, the complexity plot is shown in Fig. 3.17b. The reference Monte Carlo cost is computed using the same procedure as described in Section 3.5.1 for the Poisson problem. The figure once again demonstrates that the complexity behaviour matches the best case scenario predicted by Proposition 3.2.1. The MLMC estimator shows a complexity of $\mathcal{O}(\epsilon^{-2})$ as compared to a complexity of $\mathcal{O}(\epsilon^{-3.6})$ in the Monte Carlo case. In addition, even for the largest tolerance considered, the MLMC estimator is three orders faster than the corresponding Monte Carlo estimator.

Fig. 3.18 shows the true and estimated MSEs on $\Phi^{(m)}$, $m \in \{0,1,2\}$, as well as the VaR for the same set of simulations as in Fig. 3.17a. In addition, MSE estimates are computed for the VaR case with a smaller interval $\Theta = [0.49, 0.51]$. We note that the results are similar to those seen for the Poisson and Black Scholes problems, although the error estimators for all four statistics are relatively more conservative when compared to those problems. In addition, although the reduction of interval size leads to a reduction in the MSE estimates predicted for the VaR, the reduction is not as significant as in the case of the Poisson and Black-Scholes problems.

Finally, we recall that the rescaling ratio for the Poisson problem was shown in Fig. 3.6 to be relatively stable with respect to different interval sizes and hierarchy shapes. However, we conducted a similar study for the Navier-Stokes problem and observed that the rescaling ratio was drastically more sensitive to the choice of these parameters than in the Poisson problem case. The study is summarized in Fig. 3.19; namely that we observe the behaviour of the variance rescaling ratio r_e for different hierarchy shapes and interval sizes around the 70%-VaR. We conduct the study only for hierarchies as in Eq. (3.72) with $r = 0$, that is, for a hierarchy with the same sample sizes across all levels. In contrast to the results of Fig. 3.6, we observe that the rescaling ratio very strongly depends on the hierarchy size N_0, as well as the interval Θ, and increases significantly for smaller values of N_0 and shorter intervals Θ. These observations demonstrate the imperative need for an adaptive selection algorithm for



(a) Reliability of error estimator

(b) Complexity behaviour

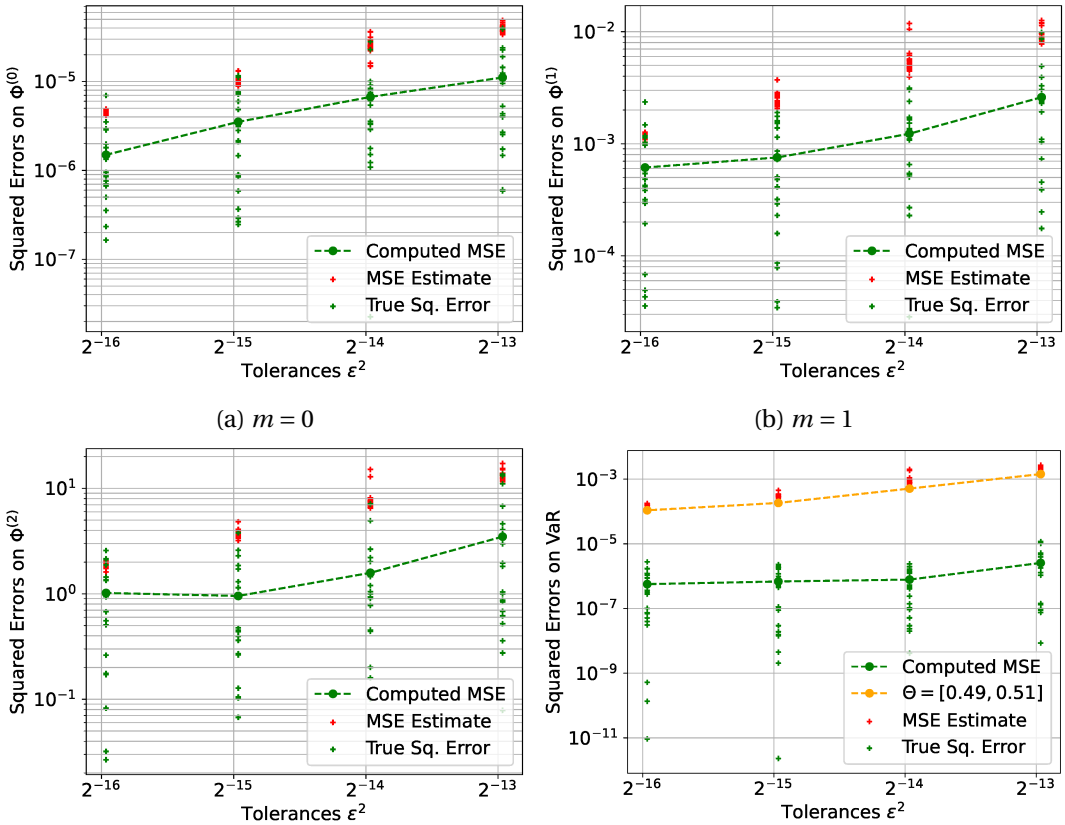
Figure 3.17: Summary of results for the Navier-Stokes problem

the interval Θ. We plan to explore this direction in a future work.

3.6 Conclusions

The aim of this work was to tackle the problem of estimating summary statistics of a random QoI which was an output of a complex differential model with random inputs. Namely, MLMC estimators for the VaR, the CVaR, the CDF and the PDF were proposed based on the concept of parametric expectations proposed in [85]. In this past theoretical work, a priori error estimates and complexity results were proposed for MLMC estimators of parametric expectations, laying the foundation for the current work. However, the a priori estimates previously proposed were found to be highly conservative due to the presence of large leading constants and hence, practically unusable.

A completely practical modification was presented in this work by developing novel error estimators combined with an adaptive strategy for selecting the MLMC hierarchy parameters and a CMLMC framework for these summary statistics. The novel developments entail the following. Novel error estimators were presented for parametric expectations of the form in Eqs. (3.1) and (3.2) in Section 3.3. In Section 3.4, we have subsequently derived novel error estimators on the VaR and the CVaR based on the novel error estimators on parametric expectations. The error estimators presented in this work are an important improvement from the error bounds presented in [85]; namely that they eliminate large leading constants that led to conservative error estimates while preserving decay rate properties important for the optimal performance of the MLMC algorithm. Novel practical methods were presented for estimating the bias and statistical error components; the bias error is estimated using a KDE-smoothened density and the statistical error is estimated using bootstrapping and localised using rescaled local variances. Adaptive strategies were also presented for selecting the parameters of the MLMC estimator for parametric expectations based on these error estimates. In particular, a CMLMC algorithm was described to successively calibrate the MLMC



(a) $m = 0$ (b) $m = 1$

(c) $m = 2$ (d) VaR

Figure 3.18: Reliability of error estimator for $\Phi^{(m)}$ and VaR for Navier Stokes problem

estimator on iteratively improved estimates of the errors. The above combination of error estimators, adaptive strategy and MLMC algorithm were demonstrated on a simple problem whose analytical solution was known. It was shown that the error estimators provided practical error bounds on the true error and resulted in practically computable hierarchies for the test problems, ranging from a simple Poisson problem to the steady Navier-Stokes equations for flow over a cylinder, demonstrated in this study.

The numerical examples considered here indicate that the performance of the novel approach sensitively depends on the choice of interval over which to construct the parametric expectation. It was shown that the choice of interval was important to the tightness of the novel error estimators for derived quantities such as the VaR and the CVaR. We plan to explore this and related improvements in future works.

3.A Spline Intepolator Property

We recall here basic results on error bounds for the use of cubic spline interpolation operators to approximate a function and its derivatives.

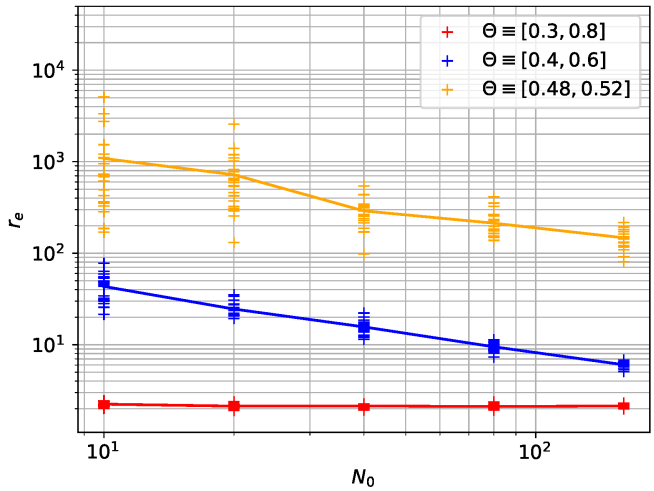


Figure 3.19: Behaviour of r_e for different hierarchy shapes and interval sizes for Navier Stokes problem

Lemma 3.A.1 (Cubic spline interpolation operator)**.** *Let $\mathscr{S}_n\left(\boldsymbol{f}(\boldsymbol{\theta})\right) \in C^2(\Theta)$ be the cubic spline interpolation operator acting on the function values $\boldsymbol{f}(\boldsymbol{\theta}) \in \mathbb{R}^n$ consisting of the function $f : \Theta \to \mathbb{R}$ evaluated at the n uniform nodes $\boldsymbol{\theta} = [\theta_1, ..., \theta_n]^T$ such $[\theta_1, \theta_n] = \Theta$. The interpolation operator satisfies*

(S.1) *for $m \in \{0,1,2\}$ and for any $f \in C^4(\Theta)$*

$$\left\| f^{(m)} - \frac{d^m}{d\theta^m}\mathscr{S}_n\left(\boldsymbol{f}(\boldsymbol{\theta})\right)\right\|_{\mathrm{L}^\infty(\Theta)} \leq C_1(m)\left\|f^{(4)}\right\|_{\mathrm{L}^\infty(\Theta)}\left(\frac{|\Theta|}{n}\right)^{(4-m)},$$

with $C_1(0) = 5/384$, $C_1(1) = 1/24$, and $C_1(2) = 3/8$,

(S.2) *for $m \in \{1,2\}$ and for any $\boldsymbol{x} \in \mathbb{R}^n$*

$$\left\| \frac{d^m}{d\theta^m}\mathscr{S}_n(\boldsymbol{x})\right\|_{\mathrm{L}^\infty(\Theta)} \leq C_2(m)(n-1)^m \left\|\mathscr{S}_n(\boldsymbol{x})\right\|_{\mathrm{L}^\infty(\Theta)},$$

with $C_2(1) = 18/|\Theta|$ and $C_2(2) = 48/|\Theta|^2$,

(S.3) $\|\mathscr{S}_n(\boldsymbol{x})\|_{\mathrm{L}^\infty(\Theta)} \leq C_3\|\boldsymbol{x}\|_{l^\infty}$ *for any $\boldsymbol{x} \in \mathbb{R}^n$, with $C_3 = \frac{7(2\sqrt{7}+1)}{27}$,*

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

Proof. The fact that $S := \mathscr{S}_n\left(\boldsymbol{f}(\boldsymbol{\theta})\right) \in C^2(\Theta)$ as well as the properties **(S.1)** and **(S.3)** are well known results in approximation theory [69, 44, 109]. To prove property **(S.2)**, we first note that

$$\|S\|_{\mathrm{L}^\infty(\Theta)} = \max_{1\leq j\leq n-1}\left\|P_j\right\|_{\mathrm{L}^\infty([\theta_j,\theta_{j+1}])} = \max_{1\leq j\leq n-1}\left\|P_j \circ g_j\right\|_{\mathrm{L}^\infty([-1,1])},$$

where $g_j(t) = \frac{\theta_j+\theta_{j+1}}{2} + \frac{\delta}{2}t$ with $\delta := \theta_{j+1} - \theta_j = \frac{|\Theta|}{n-1}$. Here, P_j denotes the cubic spline polynomial on the interval $[\theta_j, \theta_{j+1}]$. It then follows from the Markov type inequality result [65] that

$$\|S\|_{\mathrm{L}^\infty(\Theta)} \geq \frac{1}{9} \max_{1\leq j\leq n-1} \left\| \frac{d}{dt} P_j(g_j) \right\|_{\mathrm{L}^\infty([-1,1])} = \frac{\delta}{18} \max_{1\leq j\leq n-1} \left\| P_j^{(1)} \right\|_{\mathrm{L}^\infty([\theta_j,\theta_{j+1}])} ,$$

which shows that

$$\left\| S^{(1)} \right\|_{\mathrm{L}^\infty(\Theta)} \leq \frac{18}{\delta} \|S\|_{\mathrm{L}^\infty(\Theta)} = \frac{18(n-1)}{|\Theta|} \|S\|_{\mathrm{L}^\infty(\Theta)} .$$

An analogous analysis for the second derivative yields $\left\| S^{(2)} \right\|_{\mathrm{L}^\infty(\Theta)} \leq \frac{48}{\delta^2} \|S\|_{\mathrm{L}^\infty(\Theta)}$, which completes the proof. We recall that the constants $C_1(0)$ and $C_1(1)$ in **(S.1)** above are known to be optimal [69]. □

Optimisation Under Uncertainty Part II

4 Overview of Risk-Averse PDE-Constrained OUU

Optimisation algorithms play an important role across various scientific and engineering fields as valuable design tools. The key goal of optimisation is to find the best values of certain parameters (design variables) of a model, typically a differential model such as a PDE, used to predict the behaviour of a certain system, such that a desired output quantity of the model is optimised. Such differential models usually also include various other input parameters beside the design variable, which may or may not be fully characterised. There is an increasing interest in the computational science and engineering community to treat such parameters as random variables to reflect their uncertainty, either due to a lack of knowledge or to some intrinsic variability. As a result, the output quantity being optimised also becomes a random variable. Naively optimising the system for only one particular value of the random inputs can lead to a design that is not robust enough to the uncertainties in the system. A classical example is of designing civil engineering structures to minimise structural loads, the target application area of the ExaQUte project. Designing the structure for moderate or mean wind conditions may result in a design that is unable to withstand, for example, local wind gusts or storms. The field of PDE-constrained OUU seeks to characterise the randomness of the output QoI of the PDE using summary statistics such as moments, quantiles, etc., and optimise the summary statistic instead of the QoI directly. In particular, in risk-averse PDE-constrained optimisation, one aims at favouring designs with acceptable performance also in extreme conditions. In this case, the summary statistic, often called a risk-measure, should quantify the importance that is given in the design process to unfavourable scenarios. The choice of risk-measure, in turn, influences the choice of optimisation algorithm and sampling strategy used to solve the problem.

In this chapter, we present an overview of OUU problems, solution methodologies and sampling strategies. We begin by focusing on the problem formulation of risk-averse optimisation, in Section 4.1. In this thesis, we focus on coherent risk-measures, a class of risk-measures that demonstrates several favourable properties for use in risk-averse OUU. We briefly review the concept of coherent risk-measures in Section 4.2, with a focus on the CVaR, our coherent risk-measure of choice for the optimisation applications targeted within the

ExaQUte project and within this thesis in Section 4.2. Section 4.3 presents a brief overview of various optimisation algorithms and sampling strategies used to solve risk-averse design problems in literature.

We recall that the key aim of the ExaQUte project is to carry out risk-averse civil engineering design using gradient-based approaches. We also recall that we are primarily interested in using the MLMC method to do so, since the underlying differential model is typically a high-cost model. To this end, we review in Section 4.4 the relevant literature on risk-averse PDE-constrained minimisation of the CVaR, specifically using gradient-based approaches combined with the MLMC method, highlighting existing challenges. We also highlight the key features of our work in Chapter 5 that address these challenges, thereby laying the foundation for the developments presented in Chapter 5.

4.1 Risk-measures and optimisation formulation

Let $(\Omega, \mathscr{F}, \mathbb{P})$ denote a complete probability space and $\omega \in \Omega$ an elementary random event. We denote by $Q(z) \in \mathrm{L}^p(\Omega, \mathbb{R})$ a random QoI that is the output of the underlying differential model for a given set of design parameters $z \in \mathbb{R}^d$. A risk-measure $\mathscr{R} : \mathrm{L}^p(\Omega, \mathbb{R}) \to \mathbb{R}$ is a mapping that quantifies a notion of risk in the distribution of $Q(z)$. We focus in this work on optimisation problems of the type:

$$\mathscr{J}^* = \min_{z \in \mathbb{R}^d} \mathscr{J}(z) := \mathscr{R}(Q(z)). \tag{4.1}$$

The choice of $\mathscr{R}$ has important implications for the distribution of $Q(z)$. For example, a popular class of risk-measures, known as mean-risk models, are of the form $\mathscr{R}(Q(z)) := \mathbb{E}[Q(z)] + c\mathbb{D}[Q(z)], c > 0$, where $\mathbb{D}[Q(z)]$ quantifies the "dispersion" of the QoI. A common candidate for $\mathbb{D}$ is the standard deviation, leading to the mean-variance risk-measure $\mathscr{R}(Q(z)) = \mathbb{E}[Q(z)] + c\sqrt{\mathbb{V}\mathrm{ar}(Q(z))}$, which aims to minimise $\mathbb{E}[Q]$ as well as deviations from it, with the relative balance being controlled by the constant c. However, a common downside of this risk-measure is that it equally weighs deviations both above and below the mean. Another class of candidates for $\mathbb{D}$, which remedy this issue, are the central semi-deviations $\sigma_p^+[Q(z)] = \left(\mathbb{E}\left[(Q(z) - \mathbb{E}[Q(z)])_+^p\right]\right)^{1/p}$, where $p \in [1, \infty)$ and the subscript + denotes the maximum function. We refer the reader to [122, Chapter 6] for an extensive summary of different risk-measures and their corresponding properties.

[80] presented another class of risk-measures, corresponding to probabilistic optimisation, wherein the objective function $\mathscr{J}(z)$ has the form:

$$\mathscr{J}(z) = \mathbb{P}(Q(z) \geq Q_{ref}) = \mathbb{E}\left[\mathbb{1}_{Q(z) \geq Q_{ref}}\right], \tag{4.2}$$

which measures the probability of $Q(z)$ exceeding a prescribed threshold value Q_{ref}. A classical example, pertinent to the ExaQUte project, is when Q_{ref} corresponds to a critical structural moment or force in a structural engineering application. The work [112] also explored

the use of failure probabilities within constraints.

We also mention the concept of distributionally robust optimisation introduced in [10]. The measure of $Q(z)$ is treated as unknown, and it is desired to construct a surrogate measure from some known, possibly noisy, measurements of $Q(z)$. A possible formulation is then to minimise, with respect to the design z, the worst possible expectation over the space of possible surrogate measures as follows:

$$\mathscr{J}(z) := \sup_{P \in \mathscr{M}} \mathbb{E}_P \left[Q(z) \right], \tag{4.3}$$

where P denotes the surrogate measure and $\mathscr{M}$ denotes the set of surrogate probability measures. The interested reader is referred to [122] and [80] for a thorough overview of risk-measures and their use in risk-averse problem formulations. We remark that, in our work [51], we followed closely the analysis of [80] for a similar problem of PDE constrained penalised risk-measure minimisation, presenting optimality conditions and outlining the algorithmic implementation aspects of using a gradient-based algorithm for some common risk-measures including the CVaR.

4.2 Coherent risk-measures and the CVaR

Coherent risk-measures, introduced in [11], are risk-measures that satisfy several additional axioms that yield favourable properties for use in risk-averse OUU. These axioms are presented in Definition 1. Several coherent risk-measures have been proposed in literature, following the work of [11], including the entropic-value-at-risk [1], The tail-value-at-risk [104], and the CVaR [113].

Definition 1. *A coherent risk-measure is a risk-measure $\mathscr{R} : \mathrm{L}^p(\Omega, \mathbb{R}) \to \mathbb{R}$, $p \in [1, \infty)$ that satisfies the following properties for all $X, Y \in \mathrm{L}^p(\Omega, \mathbb{R})$:*

(i) ***Convexity:*** *For $t \in [0,1]$, $\mathscr{R}(tX + (1-t)Y) \leq t\mathscr{R}(X) + (1-t)\mathscr{R}(Y)$.*

(ii) ***Monotonicity:*** *If X, Y satisfy $X(\omega) \leq Y(\omega)$ for $\mathbb{P} - a.e. \omega \in \Omega$, then $\mathscr{R}(X) \leq \mathscr{R}(Y)$.*

(iii) ***Translation Equivariance:*** *For $t \in \mathbb{R}$, $\mathscr{R}(X + t) = \mathscr{R}(X) + t$.*

(iv) ***Positive Homogenity:*** *For $t \geq 0$, $\mathscr{R}(tX) = t\mathscr{R}(X)$.*

The work of [115] considered optimisation problems of the form in Eq. (4.1) containing coherent risk-measures and derived optimality conditions for the problem, along with several other results on the properties of $\mathscr{R}$. [122] extended the analysis of problem (4.1) to the case of PDE-constrained optimisation, deriving existence and optimality conditions. The authors of [83] derived optimality conditions for a penalised version of Eq. (4.1) with PDE constraints and with a coherent $\mathscr{R}$, where undesirable designs were penalised through a penalisation

term. A Gâteaux differentiability result was also shown for such an objective function. Notable, the authors of [53] formulated a modified version of Eq. (4.1), by considering a sequence of log-barrier approximations combined with an interior point optimisation algorithm.

In this thesis, we focus on the CVaR [113, 114], a widely used coherent risk-measure. We denote by $c_\tau(z)$ the CVaR of $Q(z)$ of significance $\tau \in [0,1]$, i.e., $c_\tau(z) := \mathbb{E}\left[Q(z)|Q(z) \geq q_\tau\right]$, where q_τ is the τ-quantile of $Q(z)$. We showed in Chapter 3 that, following the work in [113], $c_\tau(z)$ could be written in the following form under certain conditions on the distribution of $Q(z)$:

$$c_\tau(z) = \min_{\theta\in\mathbb{R}} \left\{\Phi(\theta; z) := \mathbb{E}\left[\phi(\theta, Q(z))\right]\right\}, \quad \phi(\theta, Q) := \theta + \frac{(Q-\theta)^+}{1-\tau}. \tag{4.4}$$

Such a formulation of the CVaR in terms of a minimisation problem was shown in [113] to possess several favourable properties for use with risk-averse optimisation.

4.3 Optimisation algorithms and sampling strategies

We now review the literature on OUU algorithms to solve problem (4.1). However, we restrict our discussion to PDE-constrained problems that use sample-based discretisations of the random space, since this allows us to easily re-purpose existing PDE solvers for sampling. The reader is referred to the introduction of [84] for a review of the alternative; namely, projection-based discretisations. Notably, we refer the reader to the works [120, 119, 118], who used an adaptive sparse-grid approach combined with a one-shot optimisation algorithm based on sequential quadratic programming.

We note that two broad approaches can be used to solve problem (4.1); namely, evolutionary algorithms and gradient-based methods. Evolutionary algorithms were used in combination with Monte Carlo estimators for PDE-constrained CVaR minimisation in [108, 107]. A genetic algorithm was also used in combination with MLMC estimators in [106]. Multiple different risk-measures, including the CVaR, were estimated, and the framework was applied to aerodynamic shape optimisation problems. However, evolutionary algorithms typically have slower rates of convergence in comparison to gradient-based methods and involve multiple expensive evaluations of the objective function.

Gradient-based algorithms have also been used to solve problems of the form in Eq. (4.1). For example, [89] proposed the use of non-smooth optimisation techniques combined with Monte Carlo estimation to solve a problem of CVaR minimisation. The works [77, 79, 78] combine stochastic collocation methods to estimate different risk measures with trust-region methods for solving the optimization problem. The authors of [84] propose to minimise a smoothed version of the CVaR, estimated using a Monte Carlo approach, with second order differentiability, using a trust region algorithm.

We also highlight that MLMC algorithms have been used successfully in several works to solve

PDE-constrained OUU problems, however, for quadratic (hence smooth) risk-measures. For instance, [130] uses a combination of MLMC estimators with an optimisation approach that combines a non-linear conjugate gradient method with local quadratic approximation of the objective function to solve a quadratic optimal control problem. [94] proposes a stochastic gradient algorithm that uses MLMC estimators for a similar quadratic control problem. The work in [2] solves a quadratic optimal control problem pathwise, generating and combining the pathwise optimal controls using an MLMC estimator.

4.4 Challenges in gradient-based CVaR minimisation with MLMC

We consider the following problem of penalised PDE-constrained CVaR minimisation for our work:

$$\mathscr{J}^* = \min_{z\in\mathbb{R}^d} \left\{ c_\tau(z) + \kappa \left\| z - z_{ref} \right\|_{l^2}^2 \right\}, \tag{4.5}$$

$$= \min_{\substack{z\in\mathbb{R}^d \\ \theta\in\mathbb{R}}} \left\{ \mathscr{J}(\theta, z) := \Phi(\theta; z) + \kappa \left\| z - z_{ref} \right\|_{l^2}^2 \right\}, \tag{4.6}$$

where we have added a term penalising deviations of the design z from a preferred design z_{ref} with strength parameter κ, and $\|\cdot\|_{l^2}$ denotes the Euclidean norm. Additionally, due to the high-cost of the underlying PDE, and the high-dimensionality of input uncertainties, we are interested in the use of the MLMC method. We address in this section the main challenges in using the MLMC methods, combined with gradient-based approaches, to solve problem (4.6). We highlight how our recent work [52], that will be summarized in Chapter 5, addresses these challenges by developing novel MLMC estimators that rely on the framework of parametric expectations [85], and extend the work in Chapter 3 to the computation of CVaR sensitivities.

The computation of the sensitivities of the CVaR $c_\tau(z)$ with respect to the extended design variables z and θ typically requires the estimation of expectations of the form $\mathbb{E}\left[(Q(z)-\theta)^+\right]$ and $\mathbb{E}\left[\mathbb{1}_{Q(z)\geq\theta} f(z)\right]$ for suitable design-dependent random variables $f(z)$. Although $\mathbb{E}\left[(Q(z)-\theta)^+\right]$ and $\mathbb{E}\left[\mathbb{1}_{Q(z)\geq\theta} f(z)\right]$ can be shown to be differentiable in θ and z [74, 73] under some conditions on the distribution of $Q(z)$, sample- or quadrature-based approximations of these expectations are typically not differentiable and may require some additional treatment. One possibility is to directly use the non-differentiable estimations in combination with non-smooth optimisation techniques that use sub-gradient information. For example, the work in [89] uses a combination of smooth and non-smooth optimisation techniques, using sub-gradients computed using Monte Carlo estimators, to minimise the CVaR. Alternatively, one could construct smoothed versions of the maximum/indicator functions, with sufficient regularity such that sample-based approximations are still differentiable. For example, a regularised version of the CVaR was constructed in [84], with second order differentiability, and optimised successfully using a trust-region method combined with Monte Carlo sampling. Additionally, a primal-dual algorithm was introduced in [81] for minimising

a similarly regularised version of the CVaR, smoothed using the epi-regularisation approach developed by the same authors in [82].

However, although regularised or smoothed versions of the CVaR can be constructed with adequate differentiability, this property is lost in the limit of vanishing smoothing, as is required when the algorithm is close to the optimum. The method proposed in [85] and extended in Chapter 3 offers an alternative to CVaR regularisation. In these works, the quantity $\mathbb{E}\left[(Q(z)-\theta)^+\right]$ is estimated directly using an MLMC estimator at a set of points in θ, all sharing the same realisations of $Q(z)$, followed by a cubic spline interpolation over the pointwise evaluations thus obtained. Derivatives such as $\mathbb{E}\left[\mathbb{1}_{Q(z)\geq\theta} f(z)\right]$ are then approximated using derivatives of the cubic spline. As was discussed in Chapter 3, directly using a naive MLMC estimator to estimate $\mathbb{E}\left[\mathbb{1}_{Q(z)\geq\theta} f(z)\right]$ may cause non-optimal MLMC complexity behaviour. By constructing an MLMC estimator of $\mathbb{E}\left[(Q(z)-\theta)^+ f(z)\right]$ and numerically differentiating in θ instead, our approach ameliorates this issue and preserves the same optimal complexity behaviour of the MLMC method as predicted for estimating $\mathbb{E}[Q(z)]$. Lastly, since the MLMC estimator proposed in [85, 16] automatically provides an approximation $\hat{\mathscr{J}}(\cdot, z)$ of the function $\theta \mapsto \mathscr{J}(\theta, z)$ at a given design z, we propose in this work to use an optimisation algorithm in which, at each iteration, gradient steps are taken only in the design variable z, whereas exact optimisation in θ is performed using the surrogate $\hat{\mathscr{J}}(\cdot, z)$. Such an algorithm, introduced in [17], was applied in combination with the Monte Carlo estimation of a regularised version of the CVaR in [26].

The computation of the sensitivities of the CVaR $c_\tau(z)$ with respect to the extended design variables z and θ typically requires the estimation of expectations of the form $\mathbb{E}\left[(Q(z)-\theta)^+\right]$ and $\mathbb{E}\left[\mathbb{1}_{Q(z)\geq\theta} f(z)\right]$ for suitable design-dependent random variables $f(z)$. Although $\mathbb{E}\left[(Q(z)-\theta)^+\right]$ and $\mathbb{E}\left[\mathbb{1}_{Q(z)\geq\theta} f(z)\right]$ can be shown to be differentiable in θ and z [74, 73] under some conditions on the distribution of $Q(z)$, such sample- or quadrature-based approximations of these expectations are typically not differentiable and may require some additional treatment. One possibility is to directly use the non-differentiable estimations in combination with non-smooth optimisation techniques that use sub-gradient information. For example, the work in [89] uses a combination of smooth and non-smooth optimisation techniques, using sub-gradients computed using Monte Carlo estimators, to minimise the CVaR. Alternatively, one could construct smoothed versions of the maximum/indicator functions, with sufficient regularity such that sample-based approximations are still differentiable. For example, a regularised version of the CVaR was constructed in [84], with second order differentiability, and optimised successfully using a trust-region method combined with Monte Carlo sampling. However, although regularised or smoothed versions of the CVaR can be constructed with adequate differentiability, this property is lost in the limit of vanishing smoothing, as is required when the algorithm is close to the optimum.

The method that was proposed in [85] and that we extended in Chapter 3 offers an alternative to CVaR regularisation. We proposed to estimate the quantity $\mathbb{E}\left[(Q(z)-\theta)^+\right]$ directly using an MLMC estimator at a set of points in θ, all sharing the same realisations of $Q(z)$, followed by

a cubic spline interpolation over the pointwise evaluations thus obtained. Derivatives such as $\mathbb{E}\left[\mathbb{1}_{Q(z)\geq\theta} f(z)\right]$ were then approximated using derivatives of the cubic spline. We will followed the above path in our work [52], and will summarize the same in Chapter 5, discussing how this approach can be extended to the computation of sensitivities of the CVaR.

Additionally, since the MLMC estimator that we proposed in Chapter 3 automatically provides an approximation $\hat{\mathscr{J}}$ of the function $\theta \mapsto \mathscr{J}(\theta, z)$ at a given design z, we propose in Chapter 5 to use an optimisation algorithm in which, at each iteration, gradient steps based on $\hat{\mathscr{J}}$ are taken only in the design variable z, whereas exact optimisation in θ is performed using the surrogate $\hat{\mathscr{J}}$. Such an algorithm, introduced in [17], was applied in combination with the Monte Carlo estimation of a regularised version of the CVaR in [26]. We instead propose to combine the this optimisation approach with a CMLMC approach, and introduce a novel CMLMC-Alternating Minimisation-Gradient Descent (AMGD) algorithm in Chapter 5.

5 Gradient-based minimisation of the CVaR using MLMC estimators

The main contributions of our work [52], which were briefly introduced in Section 4.4 and summarised here, are as follows. We propose novel expressions for the sensitivity of the objective function defined in Eq. (4.6) in terms of parametric expectations, thus allowing us to use and extend the framework we presented in Chapter 3 to build cost optimal adaptive MLMC estimators for those sensitivities with error control. We then propose to use MLMC sensitivity estimators within an alternating minimisation-gradient descent algorithm, analogous to the one proposed in [17, 26], where gradient steps are taken in the design variable z whereas exact optimisation is performed in θ using an MLMC-constructed surrogate $\hat{\mathscr{J}}$ of $\mathscr{J}$. The accuracy of the surrogate and sensitivity estimation is increased over the optimisation iterations and is set proportional to the gradient norm. Following closely the analysis in [26], we propose a convergence result for our algorithm under the assumption that the objective function $\mathscr{J}(\theta, z)$ is strongly convex with Lipschitz continuous gradients.

The structure of this chapter is as follows. We present the problem formulation in Section 5.1, for a problem of penalised CVaR minimisation of the form in Eq. (4.6). A novel expression for the gradients in terms of parametric expectations is also presented in Section 5.1. In Section 5.2, we propose the novel AMGD algorithm described at the beginning of this chapter with inexact gradient and objective function estimation and demonstrate its convergence. Section 5.3 discusses the novel MLMC estimators, error estimation procedure, and adaptive CMLMC-type hierarchy selection for the gradients of $\mathscr{J}(\theta, z)$. In addition, it presents a final CMLMC-AMGD algorithm. Lastly, in Section 5.4, we demonstrate the above optimisation algorithm and MLMC procedure on two problems of interest. The first is a two-dimensional oscillator, typically used to model oscillatory phenomena in excitable media. The second is a more applied problem of pollutant transport modelling. We demonstrate that the procedure proposed in this work performs well and reflects the theoretical results presented in Sections 5.1 and 5.2.

5.1 Problem formulation

Let $(\Omega, \mathscr{F}, \mathbb{P})$ denote a complete probability space, $\omega \in \Omega$ an elementary random event and $z \in \mathbb{R}^d$ the vector of design variables. We denote by $Q(z,\omega) \in \mathbb{R}$ the random QoI, typically a functional of the solution to an underlying differential model with random input ω and design z. We are interested in minimising the CVaR $c_\tau(z)$ of the random variable $Q(z,\cdot)$ over the designs $z \in \mathbb{R}^d$, as indicated in Eq. (4.5), following the formulation presented in [113]. To this end, we first introduce the following assumptions on the random variable $Q(z,\cdot)$.

Assumption 1. *For any $z \in \mathbb{R}^d$:*

(i) *$Q(z,\cdot)$ is a random variable in $\mathrm{L}^p(\Omega,\mathbb{R})$ for some $p \in [1,\infty)$.*

(ii) *The measure of $Q(z,\cdot)$ admits a probability density function, i.e., the measure of $Q(z,\cdot)$ is free of atoms. We denote by Γ the subset of random variables in $\mathrm{L}^p(\Omega,\mathbb{R})$ that are free of atoms, and hence, $Q(z,\cdot) \in \Gamma \subset \mathrm{L}^p(\Omega,\mathbb{R})$.*

(iii) *There exists a positive random variable K, possibly dependent on z, such that $\mathbb{E}[K] < \infty$ and*

$$|Q(z+\Delta z,\cdot) - Q(z,\cdot)| \leq K(\cdot)\,\|\Delta z\|_{l^2}\,, \tag{5.1}$$

for any $\Delta z \in \mathbb{R}^d$ close enough to 0 (restated here from [74, 73]).

(iv) *For almost every $\omega \in \Omega$, the mapping $z \mapsto Q(z,\omega)$ is differentiable in $\mathbb{R}^d$ and the corresponding vector of partial derivatives $Q_z(z,\cdot) = \left[Q_{z^1}(z,\cdot),\ldots,Q_{z^d}(z,\cdot)\right]^T$ of Q with respect to the components z^k of z, $k \in \{1,\ldots d\}$, is a random variable in $\mathrm{L}^p(\Omega,\mathbb{R}^d)$.*

To quantify the tails of $Q(z,\cdot)$, we first define the τ-VaR $q_\tau(z)$, alternatively known as the τ-quantile, of significance $\tau \in (0,1)$ as follows:

$$q_\tau(z) := \min\{\theta \in \mathbb{R} | \mathbb{E}\left[\mathbb{1}_{Q(z,\cdot)\leq\theta}\right] \geq \tau\}. \tag{5.2}$$

The τ-CVaR $c_\tau(z)$ is defined as the expected value of $Q(z,\cdot)$ in the tail above and including the τ-VaR $q_\tau(z)$:

$$c_\tau(z) := \mathbb{E}\left[Q(z,\cdot)|Q(z,\cdot) \geq q_\tau(z)\right]. \tag{5.3}$$

As was described in Chapter 4, [113] proposed that $c_\tau(z)$ could be written in the form in Eq. (4.4) for a random variable $Q(z,\cdot)$ satisfying Assumption 1.(ii).

In this chapter, we extensively use the concept of parametric expectations introduced in Chapter 3. In particular, let us re-introduce the function (parametric expectation) $\Phi : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$ as:

$$\Phi(\theta; z) := \mathbb{E}\left[\phi(\theta, Q(z,\cdot))\right], \quad \theta \in \mathbb{R},\ z \in \mathbb{R}^d, \tag{5.4}$$

with $\phi : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ given by Eq. (3.2). The introduction of the parametric expectation Φ has the advantage that the τ-VaR $q_\tau(z)$ and the τ-CVaR $c_\tau(z)$ of any significance τ can be obtained by simple post-processing of Φ as:

$$q_\tau(z) = \arg\min_{\theta \in \mathbb{R}} \Phi(\theta; z), \qquad c_\tau(z) = \min_{\theta \in \mathbb{R}} \Phi(\theta; z) = \Phi(q_\tau(z); z). \tag{5.5}$$

The framework of parametric expectations allows us to write the penalised CVaR minimisation problem in Eq. (4.5) as a combined minimisation over θ and z as in Eq. (4.6). The problem is restated below for reference:

$$\mathscr{J}^* = \min_{\substack{z \in \mathbb{R}^d \\ \theta \in \mathbb{R}}} \left\{ \mathscr{J}(\theta, z) := \Phi(\theta; z) + \kappa \left\| z - z_{ref} \right\|_{l^2}^2 \right\}. \tag{5.6}$$

For the remainder of this work, we address the challenge of solving problem (5.6). The combined objective function $\mathscr{J}(\theta, z)$ has several properties that, when combined with the properties of Q in Assumption 1, have useful implications for gradient based optimisation techniques. We first discuss the differentiability of $\mathscr{J}(\theta, z)$. Theorem 5.1.1 below gives a result on Fréchet differentiability of the CVaR.

Theorem 5.1.1. *Let $\mathfrak{L}(X, Y)$ denote the space of bounded linear operators between the normed vector spaces X and Y. We define the function $\mathscr{R} : \mathbb{R} \times \mathrm{L}^p(\Omega; \mathbb{R}) \to \mathbb{R}$ as follows:*

$$\mathscr{R}(\theta, Q) := \theta + \frac{\mathbb{E}\left[(Q - \theta)^+\right]}{1 - \tau} = \mathbb{E}\left[\phi(\theta, Q)\right]. \tag{5.7}$$

Then, $\mathscr{R}(\theta, Q)$ is jointly Fréchet differentiable in $\mathbb{R} \times \Gamma$, with Fréchet derivative $D\mathscr{R}(\theta, Q) \in \mathfrak{L}(\mathbb{R} \times \mathrm{L}^p(\Omega, \mathbb{R}), \mathbb{R})$ at the point $(\theta, Q) \in \mathbb{R} \times \Gamma$ in the direction $(\delta\theta, \delta Q) \in \mathbb{R} \times \mathrm{L}^p(\Omega, \mathbb{R})$ given by:

$$D\mathscr{R}(\theta, Q)(\delta\theta, \delta Q) = \left(1 - \frac{\mathbb{E}\left[\mathbb{1}_{Q \geq \theta}\right]}{1 - \tau}\right) \delta\theta + \frac{\mathbb{E}\left[\mathbb{1}_{\{Q \geq \theta\}} \delta Q\right]}{1 - \tau}. \tag{5.8}$$

Proof. The reader is referred to Appendix 5.A for the proof. □

This result, combined with Assumption 1 on Q, leads immediately to the differentiability of $\mathscr{J}(\theta, z)$.

Corollary 5.1.1. *The objective function $\mathscr{J}(\theta, z)$ is jointly Fréchet differentiable in $\mathbb{R} \times \mathbb{R}^d$, with Fréchet derivative $D\mathscr{J}(\theta, z) \in \mathfrak{L}(\mathbb{R} \times \mathbb{R}^d, \mathbb{R})$ at the point (θ, z) in the direction $(\delta\theta, \delta z) \in \mathbb{R} \times \mathbb{R}^d$ given by:*

$$D\mathscr{J}(\theta, z)(\delta\theta, \delta z) = \left(1 - \frac{\mathbb{E}\left[\mathbb{1}_{Q \geq \theta}\right]}{1 - \tau}\right) \delta\theta + \frac{\mathbb{E}\left[\mathbb{1}_{\{Q \geq \theta\}} Q_z^T \delta z\right]}{1 - \tau} + 2\kappa (z - z_{ref})^T \delta z. \tag{5.9}$$

A direct implication of Corollary 5.1.1 is that the gradient $\nabla \mathscr{J}(\theta, z) \in \mathbb{R}^{d+1}$ and partial derivatives $\mathscr{J}_z(\theta, z) = \left[\mathscr{J}_{z^1}(\theta, z), \ldots, \mathscr{J}_{z^d}(\theta, z)\right]^T \in \mathbb{R}^d$ and $\mathscr{J}_\theta(\theta, z) \in \mathbb{R}$ exist, and the partial deriva-

tives are given by the following expressions:

$$\mathscr{J}_\theta(\theta, z) = 1 - \frac{\mathbb{E}\left[\mathbb{1}_{Q(z,\cdot)\geq\theta}\right]}{1-\tau}, \tag{5.10}$$

$$\mathscr{J}_z(\theta, z) = \frac{\mathbb{E}\left[\mathbb{1}_{Q(z,\cdot)\geq\theta} Q_z(z,\cdot)\right]}{1-\tau} + 2\kappa(z - z_{ref}). \tag{5.11}$$

One of the main contributions of this work is the estimation of the sensitivities in Eqs. (5.10) and (5.11) using MLMC estimators. However, as discussed in Chapter 4, using MLMC to directly estimate the expectations in Eqs. (5.10) and (5.11) may require a large number of samples to achieve accurate estimates. Even if accurate estimates can be achieved, it may result in compromised or non-optimal MLMC performance. The reader is referred to [16, 85] for a detailed discussion on the topic. To ameliorate this issue, we propose the following alternative formulation of the gradients in terms of parametric expectations:

$$\mathscr{J}_\theta(\theta, z) = \Phi^{(1)}(\theta; z), \quad \text{with } \Phi \text{ as in Eqs. (3.1)-(3.2)}, \tag{5.12}$$

$$\mathscr{J}_z(\theta, z) = \Psi^{(1)}(\theta; z) + 2\kappa(z - z_{ref}), \tag{5.13}$$

$$\text{where } \Psi(\theta; z) := \mathbb{E}\left[-\frac{(Q(z,\cdot)-\theta)^+ Q_z(z,\cdot)}{1-\tau}\right] =: \left[\mathbb{E}\left[\psi(\theta, Q, Q_{z^1})\right], ..., \mathbb{E}\left[\psi(\theta, Q, Q_{z^d})\right]\right]^T. \tag{5.14}$$

The superscript of the parametric expectations in Eq. (5.12) and Eq. (5.13) denotes the derivative computed with respect to θ. In addition to $\Phi(\theta; z)$, we have introduced the parametric expectation $\Psi(\theta; z) \in \mathbb{R}^d$ and the function $\psi(\theta, Q, Q_{z^k}) \in \mathbb{R}$ where z^k and Q_{z^k} denote the kth components of z and Q_z respectively, $k \in \{1, ..., d\}$. The differentiability of $\Psi(\theta; z)$ in θ follows by the same arguments of Theorem 5.1.1 and Corollary 5.1.1, under Assumption 1. It was shown in [16] that since ϕ and ψ are Lipschitz continuous in their arguments, the corresponding MLMC estimators no longer suffer from the compromised performance due to discontinuities. The idea is then to build MLMC estimators $\hat{\Phi}(\cdot, z)$ and $\hat{\Psi}(\cdot, z)$ for the whole functions $\theta \mapsto \Phi(\theta; z)$ and $\theta \mapsto \Psi(\theta; z)$ respectively on a suitably chosen interval $\Theta \subset \mathbb{R}$, and then approximate $\mathscr{J}_\theta$ and $\mathscr{J}_z$ as $\hat{\mathscr{J}}_\theta(\theta, z) = \hat{\Phi}^{(1)}(\theta; z)$ and $\tilde{\mathscr{J}}_z(\theta, z) = \hat{\Psi}^{(1)}(\theta; z) + 2\kappa(z - z_{ref})$. As a by-product of this approach for estimating sensitivities, we construct an approximation $\theta \in \Theta \mapsto \hat{\mathscr{J}}(\theta, z) = \hat{\Phi}(\theta; z) + \kappa \left\| z - z_{ref} \right\|_{l^2}^2$ of the objective function itself for all $\theta \in \Theta$, at a given design $z \in \mathbb{R}^d$. This allows us to consider an optimisation problem in which exact minimisation in θ is performed at each iteration using the surrogate $\hat{\mathscr{J}}$, and gradient steps are performed only in z using the approximate gradient $\tilde{\mathscr{J}}_z$. Notice that the gradient approximation in z is inconsistent with the surrogate model $\hat{\mathscr{J}}$, i.e., $\tilde{\mathscr{J}}_z \neq \partial_z \hat{\mathscr{J}}$, in contrast to $\hat{\mathscr{J}}_\theta$. We will detail this approach in the next section.

5.2 Gradient based optimisation algorithm

In this section, we present a gradient-based iterative procedure to find a local minimiser (θ^*, z^*) of the OUU problem in Eq. (5.6), should it exist. The broad goal of a gradient based

algorithm is to define the iterates (θ_j, z_j), $j \in \mathbb{N}$ such that

$$\lim_{j\to\infty} (\theta_j, z_j) = (\theta^*, z^*), \tag{5.15}$$

where the iterates are computed using gradient information. Motivated by our interest in using MLMC estimators based on parametric expectations to estimate the objective function and its sensitivities, we consider in this section the general situation in which, at each iteration j of the gradient based algorithm, we build an approximation $\hat{\mathscr{J}}^j(\theta, z), \theta \in \Theta$ of the objective function at the design $z \in \mathbb{R}^d$ on a suitably chosen interval $\Theta \subset \mathbb{R}$ (which may depend on j, although to ease the notation, we do not highlight such dependence), as well as approximations $\hat{\mathscr{J}}^j_\theta(\theta, z)$ and $\tilde{\mathscr{J}}^j_z(\theta, z)$, $\theta \in \Theta$ where the approximation $\tilde{\mathscr{J}}^j_z$ may not coincide with the z-derivative of $\hat{\mathscr{J}}^j$. The approximations $\hat{\mathscr{J}}^j$, $\hat{\mathscr{J}}^j_\theta$ and $\tilde{\mathscr{J}}^j_z$ may be random, as will be the case for MLMC estimators. We the propose the following variation of the standard gradient descent algorithm, starting from an initial design z_0:

$$\theta_j \in \operatorname*{arg\,min}_{\theta\in\Theta} \hat{\mathscr{J}}^j(\theta, z_j), \tag{5.16}$$

$$z_{j+1} = z_j - \alpha \tilde{\mathscr{J}}^j_z(\theta_j, z_j), \tag{5.17}$$

where $\alpha > 0$ denotes a step size parameter. We note that according to the procedure in [16], the interval Θ can be freely selected and, hence, we can ensure that θ_j always belongs to the interior of Θ, so that $\hat{\mathscr{J}}^j_\theta(\theta_j, z_j) = 0 \; \forall j \in \mathbb{N}$.

In Theorem 5.2.1 in Section 5.2.1, we show that the iterates (θ_j, z_j) converge exponentially fast in the iteration counter j towards (θ^*, z^*) under additional assumptions on the objective function $\mathscr{J}$ and its approximations $\hat{\mathscr{J}}^j$. The results of Theorem 5.2.1, specifically the implications of Eq. (5.20), demonstrate that exponential convergence of the iterates z_j and θ_j in j can be obtained if the gradient approximation is accurate up to a tolerance that is a fraction η of the gradient magnitude at the previous iteration. The step size is selected sufficiently small, and remains fixed over all optimisation iterations, although variable step sizes and line search methods could be easily added. The algorithm is terminated once the gradient magnitude drops to a specified fraction of the initial value. We introduce here the notation $w = (\theta, z)$, $\nabla\mathscr{J} = (\mathscr{J}_\theta, \mathscr{J}_z)$ and $\tilde{\nabla}\hat{\mathscr{J}}^j = (\hat{\mathscr{J}}^j_\theta, \tilde{\mathscr{J}}^j_z)$ for convenience in the following.

Algorithm 4: Novel AMGD algorithm

1: Input: Initial design z_0, iteration counter $j = 0$, tolerance $0 < \epsilon < 1$, step size $\alpha > 0$ and tolerance fraction $\eta > 0$.
2: Set residual $r = \epsilon + 1$.
3: **while** $r > \epsilon$ **do**
4: **if** $j = 0$ { Compute $\hat{\mathscr{J}}^{)}(\cdot, z_j)$ and $\tilde{\mathscr{J}}_z^0(\cdot, z_j)$ up to a fixed tolerance.}
5: **else** $\Big\{$ Compute $\hat{\mathscr{J}}^j(\cdot, z_j)$ and $\tilde{\mathscr{J}}_z^j(\cdot, z_j)$ such that
$\mathrm{MSE}\big(\tilde{\nabla}\hat{\mathscr{J}}^j(\cdot, z_j)\big) \le \eta^2 \left\|\nabla \mathscr{J}(\theta_{j-1}, z_j)\right\|_{l^2}^2$ with $\mathrm{MSE}\big(\tilde{\nabla}\hat{\mathscr{J}}^j(\cdot, z_j)\big)$ defined as in Eq. (5.20). $\Big\}$
6: Compute a minimiser $\theta_j \in \arg\min_{\theta\in\Theta} \hat{\mathscr{J}}^j(\theta, z_j)$.
7: Compute gradient step $z_{j+1} = z_j - \alpha \tilde{\mathscr{J}}_z^j(\theta_j, z_j)$.
8: Set residual $r = \left\|\tilde{\nabla}\hat{\mathscr{J}}^j(w_j)\right\|_{l^2}^2 / \left\|\nabla\hat{\mathscr{J}}^0(w_0)\right\|_{l^2}^2$.
9: Update $j \leftarrow j + 1$.
10: **end while**

5.2.1 Convergence analysis

For the interested reader, we present a self-contained convergence analysis of the iterates (θ_j, z_j) in Theorem 5.2.1, under additional assumptions on $\mathscr{J}$ and $\hat{\mathscr{J}}^j$, based on the analysis presented in [26]. The key differences in the two analyses are related to the fact that the algorithm studied here is an AMGD algorithm instead of a pure gradient descent algorithm. We first note that the objective function $\mathscr{J}(\theta, z)$ is convex under the additional assumption that $Q(z, \cdot)$ is almost surely convex in z [113, Theorem 10]. When combined with the assumption that $\mathscr{J} \to \infty$ when $\|z\|_{l^2}, |\theta| \to \infty$, this ensures that a minimiser of $\mathscr{J}(\theta, z)$ exists in $\mathbb{R} \times \mathbb{R}^d$. However, we require additional assumptions on the objective function $\mathscr{J}$ to prove exponential convergence of the iterates θ_j and z_j towards such a minimiser; namely Assumptions 2 and 3 that it is both strongly convex and with Lipschitz continuous gradients, respectively. An immediate implication of Assumption 2 is that there exists a unique minimiser $(\theta^*, z^*) \in \mathbb{R} \times \mathbb{R}^d$ for the OUU problem in Eq. (5.6) such that $\mathscr{J}_z(\theta^*, z^*) = \mathscr{J}_\theta(\theta^*, z^*) = 0$.

In what follows, we denote by $\mathbb{E}_j[\cdot]$ the expectation conditional on all of the random variables used to define z_j (i.e., conditioned on the past up to iteration j), and by $\langle \cdot, \cdot \rangle$ the l^2 inner product. Readers interested in the implementation details of Algorithm 4 and its relation to the MLMC method can proceed directly to Section 5.3.

Assumption 2. *The objective function $\mathscr{J}$ is μ-strongly convex, i.e., there exists $\mu > 0$ such that, for all $w_a, w_b \in \mathbb{R} \times \mathbb{R}^d$, equivalently:*

(i) $\mathscr{J}(w_b) \ge \mathscr{J}(w_a) + \big\langle w_b - w_a, \nabla\mathscr{J}(w_a)\big\rangle + \frac{\mu}{2}\|w_b - w_a\|_{l^2}^2$,

(ii) $\big\langle \nabla\mathscr{J}(w_b) - \nabla\mathscr{J}(w_a), w_b - w_a\big\rangle \ge \mu\|w_b - w_a\|_{l^2}^2$.

Assumption 3. *The objective function $\mathscr{J}$ has Lipschitz continuous gradients, i.e., there exists $L > 0$ such that, for all $w_a, w_b \in \mathbb{R} \times \mathbb{R}^d$:*

$$\left\| \nabla \mathscr{J}(w_b) - \nabla \mathscr{J}(w_a) \right\|_{l^2} \leq L \| w_b - w_a \|_{l^2} . \tag{5.18}$$

Lemma 5.2.1. *Let $\mathscr{J}$ satisfy Assumptions 2 and Assumptions 3. Then we have that, for $0 < \alpha \leq 1/L$,*

$$\frac{\mu}{2} \| w - w^* \|_{l^2}^2 + \frac{\alpha}{2} \| \nabla \mathscr{J}(w) \|_{l^2}^2 \leq \langle \nabla \mathscr{J}(w), w - w^* \rangle . \tag{5.19}$$

The above result is restated here from [26, Lemma 2.1].

Theorem 5.2.1. *Let $\Theta \subset \mathbb{R}$ be a convex set. Let $\mathscr{J} : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$ satisfy Assumptions 2 and 3, and $\hat{\mathscr{J}}^j : \Theta \times \mathbb{R}^d \to \mathbb{R}$ satisfy the following condition:*

$$\begin{aligned} MSE\left(\tilde{\nabla} \hat{\mathscr{J}}^j(\cdot, z_j)\right) := \mathbb{E}_{\mathrm{j}} & \left[\left\| \hat{\mathscr{J}}_\theta^j(\cdot, z_j) - \mathscr{J}_\theta(\cdot, z_j) \right\|_{\mathrm{L}^\infty(\Theta)}^2 \right] \\ & + \sum_{k=1}^d \mathbb{E}_{\mathrm{j}} \left[\left\| \hat{\mathscr{J}}_{z^k}^j(\cdot, z_j) - \mathscr{J}_{z^k}(\cdot, z_j) \right\|_{\mathrm{L}^\infty(\Theta)}^2 \right] \leq \eta^2 \| \nabla \mathscr{J}(\theta_{j-1}, z_j) \|_{l^2}^2 , \end{aligned} \tag{5.20}$$

for some $\eta > 0$, where (θ_{j-1}, z_j) is the jth iterate produced by Algorithm 4 with step size α satisfying $0 < \alpha \leq 1/L$ and $\alpha\mu \leq 1$. Then, the following result holds true:

$$\mathbb{E}\left[\| z_{j+1} - z^* \|_{l^2}^2 + C_1(\theta_j - \theta^*)^2 \right] \leq \xi \, \mathbb{E}\left[\| z_j - z^* \|_{l^2}^2 + C_1(\theta_{j-1} - \theta^*)^2 \right], \tag{5.21}$$

for some constants $C_1 > 0$ and $0 < \xi < 1$.

Proof. From the definition of the iterate z_{j+1} in Eq. (5.17), we have:

$$\| z_{j+1} - z^* \|_{l^2}^2 = \left\| z_j - z^* - \alpha \tilde{\mathscr{J}}_z^j(\theta_j, z_j) \right\|_{l^2}^2 \tag{5.22}$$

$$= \| z_j - z^* \|_{l^2}^2 + \alpha^2 \left\| \tilde{\mathscr{J}}_z^j(\theta_j, z_j) \right\|^2 - 2\alpha \left\langle \tilde{\mathscr{J}}_z^j(\theta_j, z_j), z_j - z^* \right\rangle \tag{5.23}$$

$$\begin{aligned} &= \| z_j - z^* \|_{l^2}^2 + \underbrace{\alpha^2 \left(\left\| \tilde{\mathscr{J}}_z^j(\theta_j, z_j) \right\|^2 + \left(\hat{\mathscr{J}}_\theta^j(\theta_j, z_j) \right)^2 \right)}_{=:\hat{T}_1} \\ &\underbrace{-2\alpha \left(\langle \mathscr{J}_z(\theta_j, z_j), z_j - z^* \rangle + \langle \mathscr{J}_\theta(\theta_j, z_j), \theta_j - \theta^* \rangle \right)}_{=:\hat{T}_2} \\ &\underbrace{-2\alpha \left(\left\langle \tilde{\mathscr{J}}_z^j(\theta_j, z_j) - \mathscr{J}_z(\theta_j, z_j), z_j - z^* \right\rangle + \left\langle \hat{\mathscr{J}}_\theta^j(\theta_j, z_j) - \mathscr{J}_\theta(\theta_j, z_j), \theta_j - \theta^* \right\rangle \right)}_{=:\hat{T}_3}. \end{aligned} \tag{5.24}$$

The term $\hat{T}_1 = \alpha^2 \left\| \tilde{\nabla} \hat{\mathscr{J}}^j(w_j) \right\|_{l^2}^2$ can be bounded as follows:

$$\mathbb{E}_{\mathrm{j}}\left[\hat{T}_1\right] = \alpha^2 \mathbb{E}_{\mathrm{j}}\left[\left\| \tilde{\nabla} \hat{\mathscr{J}}^j(\theta_j, z_j) \right\|_{l^2}^2\right] \tag{5.25}$$

$$\leq \alpha^2 \mathbb{E}_{\mathrm{j}}\left[\left\| \tilde{\nabla} \hat{\mathscr{J}}^j(\theta_j, z_j) \pm \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2}^2\right] \tag{5.26}$$

$$\leq \alpha^2 \left[\mathbb{E}_{\mathrm{j}}\left[\left\| \tilde{\nabla} \hat{\mathscr{J}}^j(\theta_j, z_j) - \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2}^2\right]^{1/2} + \mathbb{E}_{\mathrm{j}}\left[\left\| \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2}^2\right]^{1/2}\right]^2 \tag{5.27}$$

$$\leq \alpha^2 \left[\eta \left\| \nabla \mathscr{J}(\theta_{j-1}, z_j) \right\|_{l^2} + \mathbb{E}_{\mathrm{j}}\left[\left\| \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2}^2\right]^{1/2}\right]^2 \tag{5.28}$$

$$\leq \alpha^2 \left[(\eta^2 + \eta) \left\| \nabla \mathscr{J}(\theta_{j-1}, z_j) \right\|_{l^2}^2 + (1+\eta) \mathbb{E}_{\mathrm{j}}\left[\left\| \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2}^2\right]\right], \tag{5.29}$$

The term $\hat{T}_2 = -2\alpha \left\langle \nabla \mathscr{J}(w_j), w_j - w^* \right\rangle$ can be bounded as follows:

$$\mathbb{E}_{\mathrm{j}}\left[\hat{T}_2\right] \leq -\alpha\mu \left(\left\| z_j - z^* \right\|_{l^2}^2 + \mathbb{E}_{\mathrm{j}}\left[(\theta_j - \theta^*)^2\right]\right) - \alpha^2 \mathbb{E}_{\mathrm{j}}\left[\left\| \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2}^2\right], \tag{5.30}$$

where we have used Lemma 5.2.1. Finally, the term $\hat{T}_3 = -2\alpha \left\langle \tilde{\nabla} \hat{\mathscr{J}}^j(w_j) - \nabla \mathscr{J}(w_j), w_j - w^* \right\rangle$ can be bounded as follows:

$$\mathbb{E}_{\mathrm{j}}\left[\hat{T}_3\right] \leq 2\alpha \mathbb{E}_{\mathrm{j}}\left[\left\| \tilde{\nabla} \hat{\mathscr{J}}^j(\theta_j, z_j) - \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2} \left\| w_j - w^* \right\|_{l^2}\right] \tag{5.31}$$

$$\leq 2\alpha \mathbb{E}_{\mathrm{j}}\left[\left\| \tilde{\nabla} \hat{\mathscr{J}}^j(\theta_j, z_j) - \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2}^2\right]^{1/2} \mathbb{E}_{\mathrm{j}}\left[\left\| w_j - w^* \right\|_{l^2}^2\right]^{1/2} \tag{5.32}$$

$$\leq 2\alpha\eta \left\| \nabla \mathscr{J}(\theta_{j-1}, z_j) \right\|_{l^2} \mathbb{E}_{\mathrm{j}}\left[\left\| w_j - w^* \right\|_{l^2}^2\right]^{1/2}. \tag{5.33}$$

Combining the bounds for $\hat{T}_1$, $\hat{T}_2$ and $\hat{T}_3$, we have the following:

$$\begin{aligned}\mathbb{E}_{\mathrm{j}}\left[\left\| z_{j+1} - z^* \right\|_{l^2}^2\right] \leq{}& (1 - \alpha\mu) \left\| z_j - z^* \right\|_{l^2}^2 - \alpha\mu \mathbb{E}_{\mathrm{j}}\left[(\theta_j - \theta^*)^2\right] \\ &+ \alpha^2(\eta^2 + \eta) \left\| \nabla \mathscr{J}(\theta_{j-1}, z_j) \right\|_{l^2}^2 + \alpha^2 \eta \mathbb{E}_{\mathrm{j}}\left[\left\| \nabla \mathscr{J}(\theta_j, z_j) \right\|_{l^2}^2\right] \\ &+ 2\alpha\eta \left\| \nabla \mathscr{J}(\theta_{j-1}, z_j) \right\|_{l^2} \mathbb{E}_{\mathrm{j}}\left[\left\| w_j - w^* \right\|_{l^2}^2\right]^{1/2}.\end{aligned} \tag{5.34}$$

We now utilise Lemma 5.2.1 once again, from which we have the following result:

$$\alpha \left\| \nabla \mathscr{J}(w) \right\|_{l^2} \leq (1 + \sqrt{1 - \alpha\mu}) \left\| w - w^* \right\|_{l^2} =: \tilde{L} \left\| w - w^* \right\|_{l^2}, \tag{5.35}$$

for $0 < \alpha \leq 1/L$ and $\alpha\mu \leq 1$. In addition, the last term of Eq. (5.34) can be rewritten as follows:

$$2\alpha\eta \left\| \nabla \mathscr{J}(\theta_{j-1}, z_j) \right\|_{l^2} \mathbb{E}_{\mathrm{j}}\left[\left\| w_j - w^* \right\|_{l^2}^2\right]^{1/2} \leq \eta \left(\frac{\alpha^2 \left\| \nabla \mathscr{J}(\theta_{j-1}, z_j) \right\|_{l^2}^2}{\tilde{L}} + \tilde{L} \mathbb{E}_{\mathrm{j}}\left[\left\| w_j - w^* \right\|_{l^2}^2\right]\right) \tag{5.36}$$

Applying Eqs. (5.35) and (5.36) to Eq. (5.34), we then have the following simplified bound:

$$\begin{aligned}\mathbb{E}_{\mathrm{j}}\left[\left\|z_{j+1}-z^*\right\|_{l^2}^2\right] &\leq \left(1-\alpha\mu+(\eta^2+2\eta)\tilde{L}^2+2\eta\tilde{L}\right)\left\|z_j-z^*\right\|_{l^2}^2 \\ &\quad+\left(-\alpha\mu+\eta\tilde{L}^2+\eta\tilde{L}\right)\mathbb{E}_{\mathrm{j}}\left[(\theta_j-\theta^*)^2\right] \\ &\quad+\left((\eta^2+\eta)\tilde{L}^2+\eta\tilde{L}\right)(\theta_{j-1}-\theta^*)^2, \qquad (5.37)\\ &= (1-C_1+C_2)\left\|z_j-z^*\right\|_{l^2}^2-C_1\,\mathbb{E}_{\mathrm{j}}\left[(\theta_j-\theta^*)^2\right]+C_2(\theta_{j-1}-\theta^*)^2, \qquad (5.38)\end{aligned}$$

where we have defined the constants $C_1 = \alpha\mu - \eta\tilde{L}^2 + \eta\tilde{L}$ and $C_2 = (\eta^2+\eta)\tilde{L}^2 + \eta\tilde{L}$. We then have the following:

$$\begin{aligned}\mathbb{E}_{\mathrm{j}}\left[\left\|z_{j+1}-z^*\right\|_{l^2}^2\right]+C_1\,\mathbb{E}_{\mathrm{j}}\left[(\theta_j-\theta^*)^2\right] &\leq (1-C_1+C_2)\left\|z_j-z^*\right\|_{l^2}^2+C_2(\theta_{j-1}-\theta^*)^2 \\ &\leq \max\left(1-C_1+C_2,\frac{C_2}{C_1}\right)\left(\left\|z_j-z^*\right\|_{l^2}^2+C_1(\theta_{j-1}-\theta^*)^2\right).\end{aligned} \qquad (5.39)$$

We note that the leading constant on the right hand side is less than 1 as long as $C_1 > C_2$, which holds true for $\eta < \sqrt{1+\alpha\mu/\tilde{L}^2}-1$. This in turn ensures contraction in the norm $\|z\|_{l^2}^2 + C_1\theta^2$ on the space $\mathbb{R}^d \times \mathbb{R}$. This completes the proof. □

Remark 1. We note that although the accuracy condition Eq. (5.20) is stated in the L^∞-norm for all θ, the proof of Theorem 5.2.1 uses this property only at θ_j. This condition is required since we do not know the quantile θ_j a priori, and seek to use the parametric expectation framework from [16] to do so. [16] requires that the error in the approximations $\hat{\mathscr{J}}^j$ be controlled at all θ, in order to estimate θ_j accurately.

Remark 2. In practical applications, it is difficult to determine whether Assumptions 2 and 3 are satisfied, since both are strongly dependent on the properties of the random QoI $Q(z,\cdot)$. These assumptions require stronger properties on $Q(z,\cdot)$ and its PDF than those presented in Assumption 1; for example, that the PDF remains both upper- and lower-bounded away from zero for all designs z, and that the random variable $Q(z,\cdot)$ is bounded, i.e., $Q(z,\cdot) \in \mathrm{L}^\infty(\Omega,\mathbb{R})$.

5.3 Gradient estimation and error control using MLMC methods

We note that the key assumption in the proof of Theorem 5.2.1 is Eq. (5.20); namely, that the gradient approximation is accurate up to a tolerance that is proportional to the magnitude of the true gradient. As stated earlier in Chapter 4, we are interested in utilising the framework of MLMC estimators for parametric expectations developed in [16] for the accurate estimation of the objective function $\mathscr{J}$ (risk-measure CVaR) and its gradient.

Expressing the gradients $\mathscr{J}_z$ and $\mathscr{J}_\theta$ in terms of the first derivatives of the parametric expectations $\Phi(\theta;z)$ and $\Psi(\theta;z)$ as in Eqs. (5.12) and (5.13) and estimating the latter using MLMC

estimators poses many key advantages. The first advantage was already seen earlier in Section 5.2; namely that $\tilde{\mathscr{J}}_z^j$ and $\hat{\mathscr{J}}_\theta^j$ can be estimated for all θ for a given design z in one shot. Secondly, as was demonstrated in [16], the analogous differences for $\Phi(\theta; z)$ decay at the same rate in the levels l as the differences $Q_l - Q_{l-1}$, in an appropriately selected norm over $\theta \in \mathbb{R}$. This ensures that if cost-optimal MLMC behaviour can be achieved for estimating $\mathbb{E}[Q]$, then it can be achieved also for MLMC estimators of $\Phi(\theta; z)$ and $\Psi(\theta; z)$, using a practically computable number of samples. The last key advantage is that, using the mechanism in [16], one can select the parameters of the MLMC estimator such that a prescribed tolerance can be attained on the MLMC approximation error on Φ and Ψ. By prescribing a tolerance proportional to the gradient magnitude, one can estimate the gradient using MLMC estimators that respect the condition in Eq. (5.20) as required by Algorithm 4.

Although the procedure used in this chapter to estimate Φ accurately is identical to the one described in Chapter 3, some important modifications are required to use the same procedure for accurately estimating Ψ. We present in this section the modifications of our work [16] presented in Chapter 3 that are required for the accurate estimation of Ψ, and consequently the gradients $\mathscr{J}_\theta$ and $\mathscr{J}_z$, using the MLMC method.

5.3.1 MLMC estimator for the gradients

We begin by recalling that the parametric expectation Ψ is defined as in Eq. (5.14). The proposed MLMC method relies on a sequence of approximations $\{Q_l(z)\}_{l=0}^L$ to $Q(z)$ on a sequence of $L+1$ discretisations with, for example, different mesh sizes $h_0 > h_1 > ... > h_L$, typically a geometric sequence $h_{l-1} = s h_l$ with $s > 1$. The MLMC estimator for the kth component $\Psi_k(\cdot; z) := \mathbb{E}\left[\psi(\cdot, Q(z), Q_{z^k}(z))\right]$ of Ψ on Θ, $k \in \{1, ..., d\}$ follows the same construction as that for Φ in [16]. The first step is to estimate $\Psi_k(\theta_r, z)$, $r \in \{1, ..., n\}$, on a set of n equidistant points $\boldsymbol{\theta} = \{\theta_1, ..., \theta_n\}$ such that $\Theta = [\theta_1, \theta_n]$, by a standard MLMC estimator $\hat{\Psi}_{L,k}(\theta_r; z)$, which reads:

$$\begin{aligned}\hat{\Psi}_{L,k}(\theta_r; z) := &\frac{1}{N_0}\sum_{i=1}^{N_0} \psi\left(\theta_r, Q_0^{(i,0)}(z), Q_{z^k,0}^{(i,0)}(z)\right) \\ &+ \sum_{l=1}^{L} \frac{1}{N_l}\sum_{i=1}^{N_l}\left[\psi\left(\theta_r, Q_l^{(i,l)}(z), Q_{z^k,l}^{(i,l)}(z)\right) - \psi\left(\theta_r, Q_{l-1}^{(i,l)}(z), Q_{z^k,l-1}^{(i,l)}(z)\right)\right],\end{aligned} \tag{5.40}$$

where $Q_l^{(i,l)}(z) \equiv Q_l(z; \omega^{(i,l)})$ and $Q_{l-1}^{(i,l)}(z) \equiv Q_{l-1}(z; \omega^{(i,l)})$ are correlated realisations of $Q_l(z)$ and $Q_{l-1}(z)$, respectively, typically obtained by solving the underlying differential problem on meshes with discretisation parameters h_l and h_{l-1}, driven by the same realisation $\omega^{(i,l)}$ of the random parameters for the fixed design z. On the other hand, $Q_l^{(i,l)}$ and $Q_k^{(j,k)}$ are independent if $i \neq j$ or $l \neq k$. Finally, $Q_{z^k,l}^{(i,l)}$ and $Q_{z^k,l-1}^{(i,l)}$ are the sensitivities of the realisations $Q_l^{(i,l)}$ and $Q_{l-1}^{(i,l)}$ respectively with respect to z^k. $\{N_l\}_{l=0}^L$ is a decreasing sequence of sample sizes. The MLMC hierarchy is hence defined by three parameters; namely the number of interpolation points n, the number of levels L and the level-wise sample sizes N_l.

We finally construct a MLMC estimator $\hat{\Psi}_{L,k}$ of the whole function $\Psi_k(\cdot;z):\Theta\to\mathbb{R}$ by interpolating over the pointwise estimates as below:

$$\hat{\Psi}_{L,k}(\cdot;z)=\mathscr{S}_n\left(\hat{\Psi}_{L,k}(\boldsymbol{\theta};z)\right), \tag{5.41}$$

where $\mathscr{S}_n$ denotes a uniform cubic spline interpolation operator and $\hat{\Psi}_{L,k}(\boldsymbol{\theta};z)$ denotes the set of pointwise MLMC estimates in Eq. (5.40), that is $\hat{\Psi}_{L,k}(\boldsymbol{\theta};z)=\{\hat{\Psi}_{L,k}(\theta_1;z),\hat{\Psi}_{L,k}(\theta_2;z),\ldots,\hat{\Psi}_{L,k}(\theta_n;z)\}$. An estimate of the first derivative $\Psi_k^{(1)}$ in θ is then obtained by computing the derivative of the resultant interpolated function, for each component $\hat{\Psi}_{L,k}^{(1)}$:

$$\hat{\Psi}_{L,k}^{(1)}(\cdot;z):=\mathscr{S}_n^{(1)}\left(\hat{\Psi}_{L,k}(\boldsymbol{\theta};z)\right):=\frac{\partial}{\partial\theta}\mathscr{S}_n\left(\hat{\Psi}_{L,k}(\boldsymbol{\theta};z)\right). \tag{5.42}$$

5.3.2 Estimation of the MSE of the gradient

Since we have assumed that the gradient estimate $\tilde{\nabla}\hat{\mathscr{J}}^j$ is a random vector in $\mathrm{L}^p(\Omega,\mathbb{R}^{d+1})$ with $p\geq 2$, we propose to quantify the error on the gradient in an MSE sense as follows:

$$\mathrm{MSE}\left(\tilde{\nabla}\hat{\mathscr{J}}^j(\cdot,z_j)\right):=\mathbb{E}\left[\left\|\hat{\mathscr{J}}_\theta^j(\cdot,z_j)-\mathscr{J}_\theta(\cdot,z_j)\right\|^2_{\mathrm{L}^\infty(\Theta)}\right]+\sum_{k=1}^d\mathbb{E}\left[\left\|\tilde{\mathscr{J}}_{z^k}^j(\cdot,z_j)-\mathscr{J}_{z^k}(\cdot,z_j)\right\|^2_{\mathrm{L}^\infty(\Theta)}\right], \tag{5.43}$$

where $\mathscr{J}_{z^k}$ and $\tilde{\mathscr{J}}_{z^k}^j$ denote the kth components of $\mathscr{J}_z$ and $\tilde{\mathscr{J}}_z^j$.

We now present a result relating $\mathrm{MSE}\left(\tilde{\nabla}\hat{\mathscr{J}}^j(\cdot,z_j)\right)$ to the MSE of the MLMC estimators $\hat{\Phi}_L^{(1)}$ and $\hat{\Psi}_L^{(1)}$.

Proposition 5.3.1. *Let $\hat{\Phi}_L(\cdot;z_j)$ and $\hat{\Psi}_L(\cdot;z_j)$ denote the MLMC estimators of $\Phi(\cdot;z_j)$ and $\Psi(\cdot;z_j)$ as defined in Eq. (3.7) and Eq. (5.40) respectively. Let $\tilde{\nabla}\hat{\mathscr{J}}^j(\cdot,z_j)$ be the approximation to the true gradient $\nabla\mathscr{J}(\cdot,z_j)$ computed using the estimates $\hat{\Phi}_L^{(1)}(\cdot;z_j)$ and $\hat{\Psi}_L^{(1)}(\cdot;z_j)$ at the jth optimisation iteration. Let Ψ_k and $\hat{\Psi}_{L,k}$ denote the kth component of Ψ and $\hat{\Psi}_L$ respectively, for $k\in\{1,\ldots,d\}$. Let the MSEs on $\hat{\Phi}_L^{(1)}$ and $\hat{\Psi}_{L,k}^{(1)}$ be defined as follows:*

$$MSE\left(\hat{\Phi}_L^{(1)}\right)(z_j):=\mathbb{E}\left[\left\|\hat{\Phi}_L^{(1)}(\cdot;z_j)-\Phi^{(1)}(\cdot;z_j)\right\|^2_{\mathrm{L}^\infty(\Theta)}\right], \tag{5.44}$$

$$MSE\left(\hat{\Psi}_{L,k}^{(1)}\right)(z_j):=\mathbb{E}\left[\left\|\hat{\Psi}_{L,k}^{(1)}(\cdot;z_j)-\Psi_k^{(1)}(\cdot;z_j)\right\|^2_{\mathrm{L}^\infty(\Theta)}\right], \tag{5.45}$$

for the design $z_j\in\mathbb{R}^d$. Then, we have that:

$$MSE\left(\tilde{\nabla}\hat{\mathscr{J}}^j(\cdot,z_j)\right)=MSE\left(\hat{\Phi}_L^{(1)}\right)(z_j)+\sum_{k=1}^d MSE\left(\hat{\Psi}_{L,k}^{(1)}\right)(z_j). \tag{5.46}$$

Proof. We first note that:

$$\left\| \hat{\mathscr{J}}_{\theta}^{j}(\cdot, z_j) - \mathscr{J}_{\theta}(\cdot, z_j) \right\|_{\mathrm{L}^{\infty}(\Theta)}^{2} = \left\| \hat{\Phi}^{(1)}(\cdot; z_j) - \Phi^{(1)}(\cdot; z_j) \right\|_{\mathrm{L}^{\infty}(\Theta)}^{2} \tag{5.47}$$

$$\left\| \tilde{\mathscr{J}}_{z^k}^{j}(\cdot, z_j) - \mathscr{J}_{z^k}(\cdot, z_j) \right\|_{\mathrm{L}^{\infty}(\Theta)}^{2} = \left\| \hat{\Psi}_{L,k}^{(1)}(\cdot; z_j) - \Psi_{k}^{(1)}(\cdot; z_j) \right\|_{\mathrm{L}^{\infty}(\Theta)}^{2} \tag{5.48}$$

Adding together each of the contribuitons and taking the expectation on both sides, we have that:

$$\mathrm{MSE}\left(\tilde{\nabla} \hat{\mathscr{J}}^{j}(\cdot, z_j)\right) = \mathrm{MSE}\left(\hat{\Phi}_{L}^{(1)}\right)(z_j) + \sum_{k=1}^{d} \mathrm{MSE}\left(\hat{\Psi}_{L,k}^{(1)}\right)(z_j). \tag{5.49}$$

□

As was described earlier in this section, we seek to use the error estimation and adaptivity procedure described in Chapter 3 to accurately estimate $\Phi^{(1)}$ and $\Psi_{k}^{(1)}$, and consequently, to accurately estimate the gradient $\nabla \mathscr{J}$. From Eq. (5.46), it is evident that if one can control the MSE of $\hat{\Phi}_{L}^{(1)}$ and $\hat{\Psi}_{L,k}^{(1)}$ in an L^{∞} sense, one can control the MSE on the gradient $\nabla \mathscr{J}$ as defined in Eq. (5.43). Specifically, the MSE of the gradient is equal to a simple sum of the MSEs of the parametric expectations. Eq. (5.46) hence allows us to use the work of [16] to accurately calibrate MLMC estimators for the parametric expectations $\Phi^{(1)}$ and $\Psi^{(1)}$ such that the resultant gradient estimate is accurate up to a prescribed tolerance.

5.3.3 Modified error estimation procedure

Since the error estimation procedure is independent of the design z, in the following, we drop the explicit dependence of Φ and Ψ on z, with the dependence being implied. We recall here that the error estimation procedure for estimating $\mathrm{MSE}\left(\hat{\Phi}_{L}^{(1)}\right)$ is identical to that presented in Chapter 3. The procedure for estimating $\mathrm{MSE}\left(\hat{\Psi}_{L}^{(1)}\right)$ however has several modifications from the procedure for $\hat{\Phi}_{L}^{(1)}$, that we detail in this section. We recall that $\mathrm{MSE}\left(\hat{\Psi}_{L}^{(1)}\right)$ was defined in Eq. (5.45). Proceeding similarly as in [16], we can bound $\mathrm{MSE}\left(\hat{\Psi}_{L}^{(1)}\right)$ as follows:

$$\mathrm{MSE}\left(\hat{\Psi}_{L,k}^{(1)}\right) \leq (\hat{\mathrm{e}}_{i}^{\Psi_k})^2 + (\hat{\mathrm{e}}_{b}^{\Psi_k})^2 + (\hat{\mathrm{e}}_{s}^{\Psi_k})^2, \tag{5.50}$$

where $\hat{\mathrm{e}}_{i}^{\Psi_k}$, $\hat{\mathrm{e}}_{b}^{\Psi_k}$ and $\hat{\mathrm{e}}_{s}^{\Psi_k}$ denote error estimators that estimate the error due to interpolation, the error due to approximation of the QoI (i.e. bias error), and the error due to finite sampling (i.e. statistical error) respectively on $\hat{\Psi}_{L,k}$.

The procedure for estimating the interpolation and bias errors requires the accurate estimation of θ-derivatives of the function $\Psi_{l,k}(\theta) = \mathbb{E}\left[\psi(\theta, Q_l, Q_{z^k,l})\right]$. Although the true function $\Psi_{l,k}$ is smooth, replacing the true probability density with an empirical probability density corresponding to a Monte Carlo estimator implies that the right hand side would be a linear

combination of piecewise linear functions. The first derivative of such a function would be piecewise constant, and high order derivatives would not exist in the discontinuity points, and would be zero otherwise. A MLMC hierarchy designed based on estimates obtained in this manner would lead to non-optimal complexity behaviour. In Section 3.2.2, we described a KDE based procedure for ameliorating this issue. Although the error estimation procedure is broadly the same for estimating Ψ as for Φ, an important distinction arises with respect to this KDE procedure, which we detail in this section.

Since the issue chiefly relates to the regularity of the empirical Monte Carlo probability density, we propose the use of a KDE based smoothing technique; namely, we replace the true joint density p_l of $(Q_l, Q_{z^k,l})$ with a KDE smoothed joint probability density p_l^{kde}, which consists of a linear combination of two-dimensional kernels composed of products of two one-dimensional Gaussian kernels centred on each of the N_l fine samples $\{(Q_l^{(i,l)}, Q_{z^k,l}^{(i,l)})\}_{i=1}^{N_l}$:

$$\Psi_{l,k}(\theta) = \int\int \psi(\theta, q, q_{z^k}) p_l(q, q_{z^k}) dq dq_{z^k} \tag{5.51}$$

$$\approx \int\int \psi(\theta, q, q_{z^k}) p_l^{kde}(q, q_{z^k}) dq dq_{z^k} \tag{5.52}$$

$$:= \frac{1}{N_l}\sum_{i=1}^{N_l}\int\int \psi(\theta, q, q_{z^k}) K_{\delta_l}(q, Q_l^{(i,l)}) K_{\delta_{z^k,l}}(q_{z^k}, Q_{z^k,l}^{(i,l)}) dq dq_{z^k}. \tag{5.53}$$

$$= -\frac{1}{N_l}\sum_{i=1}^{N_l}\int q_{z^k} K_{\delta_{z^k,l}}(q_{z^k}, Q_{z^k,l}^{(i,l)}) dq_{z^k} \int \frac{(q-\theta)^+}{1-\tau} K_{\delta_l}(q, Q_l^{(i,l)}) dq. \tag{5.54}$$

$$= -\frac{1}{N_l}\sum_{i=1}^{N_l} Q_{z^k,l}^{(i,l)} \int \frac{(q-\theta)^+}{1-\tau} K_{\delta_l}(q, Q_l^{(i,l)}) dq =: \mathbb{E}_{l,k}^{kde}\left[\psi(\theta,\cdot,\cdot)\right]. \tag{5.55}$$

Here, $K_{\delta_l}(\cdot,\mu)$ denotes a Gaussian kernel with mean μ and bandwidth parameter $\delta_l > 0$, which is selected according to Scott's rule [121] for the realisations $\{Q_l^{(i,l)}\}_{i=1}^{N_l}$ and controls the "width" of the kernel. A closed form expression can be computed for the integral in Eq. (5.55), leading to the KDE smoothed approximation $\mathbb{E}_{l,k}^{kde}\left[\psi(\theta,\cdot,\cdot)\right]$ for Ψ_k.

According to the procedure in Section 3.2.1, the interpolation error requires the estimation of the quantity $\left\|\Psi_k^{(4)}\right\|$, for which we use the KDE estimator described above. To this end, we first select a level $\lceil L/2 \rceil$ from the MLMC hierarchy; this choice of level is to ensure that $\hat{\Psi}_{\lceil L/2 \rceil,k}$ is sufficiently close to Ψ_k, and $N_{\lceil L/2 \rceil}$ is large enough for the KDE procedure to produce accurate estimates. We then construct the KDE approximation $\Upsilon_{\lceil L/2 \rceil,k}(\theta) := \mathbb{E}_{\lceil L/2 \rceil,k}^{kde}\left[\psi(\theta,\cdot,\cdot)\right]$. The fourth derivative $\Upsilon_k^{(4)}$ is then constructed using a second order central finite difference scheme on a uniform grid on Θ with $n' \gg n$ points. The norm is evaluated on the same grid as follows:

$$\left\|\Psi_k^{(4)}\right\|_{\mathrm{L}^\infty(\Theta)} \approx \max_{i\in\{1,\dots,n'\}} \left|\Upsilon_{\lfloor L/2 \rfloor,k}^{(4)}(\theta_i)\right| \tag{5.56}$$

For the bias error on $\hat{\Psi}_{L,k}$, we are required to estimate the quantity

$$\left\| \mathscr{S}_n^{(1)} \left(\mathbb{E}\left[\psi(\theta, Q_l, Q_{z^k,l}) - \psi(\theta, Q_{l-1}, Q_{z^k,l-1}) \right] \right) \right\|_{\mathrm{L}^\infty(\Theta)}. \tag{5.57}$$

Replacing the expectation by a Monte Carlo estimator leads to the same regularity issue as described earlier in this section. To smooth the empirical Monte Carlo density, we propose the use of a KDE smoothed approximation $p_{l,l-1}^{kde}$ to the true density $p_{l,l-1}$ of $(Q_l, Q_{z^k,l}, Q_{l-1}, Q_{z^k,l-1})$, consisting of products of four one-dimensional Gaussian kernels:

$$\begin{aligned}
&\mathbb{E}\left[\psi(\theta, Q_l, Q_{z^k,l}) - \psi(\theta, Q_{l-1}, Q_{z^k,l-1}) \right] && (5.58)\\
&= \int\int\int\int \left[\psi(\theta, q^f, q^f_{z^k}) - \psi(\theta, q^c, q^c_{z^k}) \right] p_{l,l-1}(q^f, q^f_{z^k}, q^c, q^c_{z^k}) dq^f dq^f_{z^k} dq^c dq^c_{z^k} && (5.59)\\
&\approx \frac{1}{N_l} \sum_{i=1}^{N_l} \int\int\int\int \left[\psi(\theta, q^f, q^f_{z^k}) - \psi(\theta, q^c, q^c_{z^k}) \right] \\
&\times K_{\delta_l}(q^f, Q_l^{(i,l)}) K_{\delta_{z^k,l}}(q^f_{z^k}, Q^{(i,l)}_{z^k,l}) K_{\delta_{l-1}}(q^c, Q_{l-1}^{(i,l)}) K_{\delta_{z^k,l-1}}(q^c_{z^k}, Q^{(i,l)}_{z^k,l-1}) dq^f dq^f_{z^k} dq^c dq^c_{z^k} && (5.60)\\
&= \frac{1}{N_l} \sum_{i=1}^{N_l} Q^{(i,l)}_{z^k,l-1} \int \frac{(q^c-\theta)^+}{1-\tau} K_{\delta_{l-1}}(q^c, Q_{l-1}^{(i,l)}) dq^c - Q^{(i,l)}_{z^k,l} \int \frac{(q^f-\theta)^+}{1-\tau} K_{\delta_l}(q^f, Q_l^{(i,l)}) dq^f && (5.61)\\
&=: \mathbb{E}^{kde}_{l,l-1,k}\left[\psi(\theta, Q_l, Q_{z^k,l}) - \psi(\theta, Q_{l-1}, Q_{z^k,l-1}) \right]. && (5.62)
\end{aligned}$$

The expectation in Eq. (5.57) can be replaced by the KDE smoothened expectation in Eq. (5.62), which can then be used in the bias error estimation procedure outlined in [16]. Lastly, the procedure for the statistical error follows the idea of bootstrapping developed in [16] identically without modification.

5.3.4 Adaptive hierarchy selection procedure and CMLMC-gradient descent algorithm

We discuss in this section how to select the parameters of the MLMC hierarchy; namely the number of interpolation points n, the level-wise sample sizes N_l and the number of levels L. The aim is to select these parameters such that a prescribed tolerance can be obtained on the gradient estimate $\tilde{\nabla}\hat{\mathscr{J}}^j$. In what follows, we drop the dependence on z for notational simplicity, with the dependence being implied. We propose here a minor variation of the framework presented in Section 3.4. An adaptive strategy was proposed therein for the selection of the hierarchy parameters n, L and N_l for any statistic s_τ, the MSE of whose estimator $\hat{s}_\tau$ could be bounded by a linear combination of MSEs on $\hat{\Phi}_L$ and its derivatives:

$$\mathrm{MSE}(\hat{s}_\tau) \leq c_0 \mathrm{MSE}\left(\hat{\Phi}_L\right) + c_1 \mathrm{MSE}\left(\hat{\Phi}_L^{(1)}\right) + c_2 \mathrm{MSE}\left(\hat{\Phi}_L^{(2)}\right), \quad c_0, c_1, c_2 > 0. \tag{5.63}$$

We first note that the same hierarchy adaptivity procedure extends trivially to any linear combination of MSEs of $\hat{\Phi}_L$, $\hat{\Psi}_{L,k}$, and their derivatives. Specifically, this includes the case of the MSE on the gradient $\tilde{\nabla}\hat{\mathscr{J}}^j$ in Eq. (5.46). In addition, each of the MSEs on the parametric expectations in Eq. (5.46) can be split into its three error contributions, similar to Eq. (5.50),

leading to the following error estimator for $\mathrm{MSE}\left(\tilde{\nabla}\hat{\mathscr{J}}^{j}(w)\right)$:

$$\begin{aligned}\mathrm{MSE}\left(\tilde{\nabla}\hat{\mathscr{J}}^{j}\right) &= \mathrm{MSE}\left(\hat{\Phi}_L^{(1)}\right) + \sum_{k=1}^{d}\mathrm{MSE}\left(\hat{\Psi}_{L,k}^{(1)}\right)\\ &\leq \underbrace{\left((\hat{\mathrm{e}}_i^{\Phi})^2 + \sum_{k=1}^{d}(\hat{\mathrm{e}}_i^{\Psi_k})^2\right)}_{\text{Squared interpolation error}} + \underbrace{\left((\hat{\mathrm{e}}_b^{\Phi})^2 + \sum_{k=1}^{d}(\hat{\mathrm{e}}_b^{\Psi_k})^2\right)}_{\text{Squared bias error}} + \underbrace{\left((\hat{\mathrm{e}}_s^{\Phi})^2 + \sum_{k=1}^{d}(\hat{\mathrm{e}}_s^{\Psi_k})^2\right)}_{\text{Squared statistical error}}. \end{aligned} \tag{5.64}$$

Here, $\hat{\mathrm{e}}_i^{\Phi}$, $\hat{\mathrm{e}}_b^{\Phi}$ and $\hat{\mathrm{e}}_s^{\Phi}$ denote the interpolation, bias and statistical error estimators corresponding to $\mathrm{MSE}\left(\hat{\Phi}_L^{(1)}\right)$. Once in the above form, the procedure described in Section 3.4 for adapting the hierarchy parameters n, L and N_l for linear combinations of MSEs can be extended trivially to the current case when combined with the modifications proposed in Section 5.3.3. Lastly, we comment that the above adaptive procedure is carried out within the framework of the CMLMC algorithm presented in Algorithm 3.

We now possess all the ingredients required to tailor Algorithm 4 to the specific case in which an MLMC procedure is combined with a CMLMC algorithm to estimate the gradient up to a prescribed tolerance. The algorithm is detailed below, and differs from Algorithm 4 in that the first estimate of the gradient is computed based on a screening hierarchy, and that successive gradients are computed such that the MSE on the gradient satisfies a tolerance equal to a fraction of the gradient magnitude from the previous iteration; namely, the right hand side of Eq. (5.20) is estimated using $\nabla\hat{\mathscr{J}}^{j-1}(w_{j-1})$. Another key difference to note is that in contrast to the CMLMC Algorithm 3, the screening hierarchy used to compute first estimates for the design z_j is the optimal hierarchy used to accurately estimate the gradient for the design z_{j-1}. In addition, the gradient at the first design point z_0 is estimated using an initial small fixed hierarchy.

Algorithm 5: CMLMC-gradient descent OUU algorithm

Input: Initial design z_0, iterate $j = 0$, tolerance $0 < \epsilon < 1$, step size $\alpha > 0$ and $\eta > 0$.
Set residual $r = \epsilon + 1$
while $r > \epsilon$ **do**
  if $j = 0$ { Simulate screening hierarchy }
  else { Start CMLMC from the optimal hierarchy for z_{j-1}; Simulate CMLMC adapting hierarchy such that $\mathrm{MSE}\left(\tilde{\nabla}\hat{\mathscr{J}}^{j}(\cdot, z_j)\right) \leq \eta \left\|\tilde{\nabla}\hat{\mathscr{J}}^{j-1}(w_{j-1})\right\|_{l^2}^2$ }
  Compute minimiser $\theta_j \in \arg\min_{\theta\in\Theta}\hat{\mathscr{J}}^{j}(\theta, z_j) = \hat{\Phi}_L(\theta, z_j)$
  Compute gradient $\tilde{\mathscr{J}}_z^{j}(\theta_j, z_j) = \hat{\Psi}_L^{(1)}(\theta_j; z_j) + 2\kappa(z_j - z_{ref})$
  Compute gradient step $z_{j+1} = z_j - \alpha\tilde{\mathscr{J}}_z^{j}(\theta_j, z_j)$ and
  $\tilde{\nabla}\hat{\mathscr{J}}^{j}(w_j) = \left(\hat{\mathscr{J}}_\theta^{j}(\theta_j, z_j) = 0, \tilde{\mathscr{J}}_z^{j}(\theta_j, z_j)\right)$
  Set residual $r = \left\|\tilde{\nabla}\hat{\mathscr{J}}^{j}(w_j)\right\|_{l^2}^2 / \left\|\nabla\hat{\mathscr{J}}^{0}(w_0)\right\|_{l^2}^2$
  Update $j \leftarrow j + 1$
end while

5.4 Numerical results

5.4.1 FitzHugh Nagumo oscillator

To demonstrate the optimisation framework, we use the FitzHugh–Nagumo system described in [49] and [97]. The FitzHugh–Nagumo model is a two dimensional simplification of the Hodgkin-Huxley model introduced by [72], which was originally proposed in the field of neuroscience to model the phenomenon of spiking neurons. The dynamical equations read as follows:

$$\begin{bmatrix} \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} v - \frac{v^3}{3} - w + I \\ \zeta\,(v + a - bw) \end{bmatrix}, \quad \begin{bmatrix} v(t=0) \\ w(t=0) \end{bmatrix} = \begin{bmatrix} v^0 \\ w^0 \end{bmatrix}, \quad t \in [0, T], \tag{5.65}$$

where $[v(t), w(t)]^T \in \mathbb{R}^2$ denotes the state variables and a, b, ζ and I denote system parameters. Fig. 5.1 shows a phase-space plot containing the v and w-nullclines for a nominal value of the system parameters. The oscillator enters a limit cycle for parameter values such that the intersection of the two nullclines lies in the interval $v \in [-1, 1]$, indicated by the black lines. If the intersection lies exterior to this interval, then the oscillator eventually reaches the intersection and remains at a constant value of v and w. Although initially proposed to model neuron behaviour, the FitzHugh–Nagumo model has seen widespread use in modelling wave phenomena in excitable media. Examples include blood coagulation [47, 91] and cardio-electrophysiological phenomena [33], wherein the optimal control of the model plays an important role in the application. The reader is referred to [129] for an overview of existing work on the modelling applications and optimal control of the FitzHugh–Nagumo system.

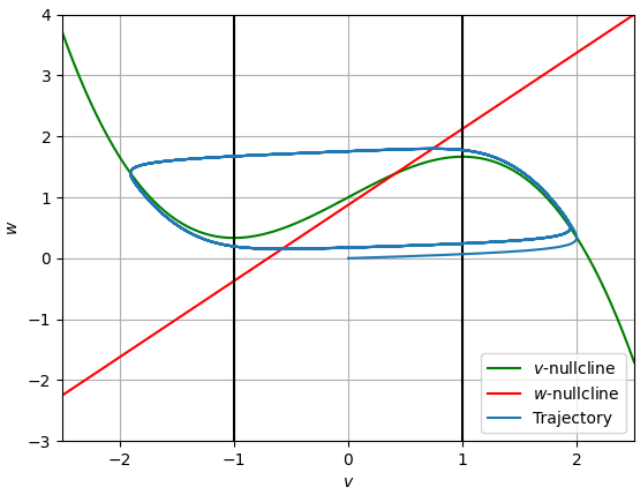


Figure 5.1: FitzHugh–Nagumo oscillator dynamics

In this work, we study the forced FitzHugh–Nagumo system:

$$\begin{bmatrix} \dot{v} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} v - \frac{v^3}{3} - w + I + \sigma \dot{W}_1 \\ \zeta\,(v + a - bw) + \sigma \dot{W}_2 \end{bmatrix}, \quad \begin{bmatrix} v(t=0) \\ w(t=0) \end{bmatrix} = \begin{bmatrix} v^0 \\ w^0 \end{bmatrix}, \quad t \in [0, T], \tag{5.66}$$

where $\dot{W}_1$ and $\dot{W}_2$ are "formal" derivatives of standard Brownian paths and $\sigma = 0.01$ controls the noise strength. To study the behaviour of the system, we propose the following QoI:

$$Q = \frac{1}{T} \int_0^T v^2(t)\,dt. \tag{5.67}$$

We are interested in minimising an objective function of the form in Eq. (5.6), where we seek to minimise the CVaR with significance $\tau = 0.7$. We denote by $z = [a, b, \zeta, I]^T$ the vector of design parameters with respect to which we want to carry out the optimisation, and seek to penalise deviations from the design $z_{ref} = [0.8, 0.7, 0.08, 1.0]$.

We discretise the interval $[0, T]$ using a hierarchy of uniform grids $t_j = j\Delta t_l$, $j \in \{0, 1, ..., N_{T,l}\}$, with $\Delta t_l = T/N_{T,l}$ and $N_{T,l} = N_{T,0} 2^l$. We set $T = 10$ and $N_{T,0} = 20$, and consider an Euler-Maruyama discretisation of Eq. (5.66). Using the notation v_n^l to denote the approximation of $v(t_n)$ at level l, the discretised system then reads:

$$\begin{bmatrix} v_{n+1}^l \\ w_{n+1}^l \end{bmatrix} = \begin{bmatrix} v_n^l \\ w_n^l \end{bmatrix} + \Delta t_l \begin{bmatrix} v_n^l - \frac{(v_n^l)^3}{3} - w_n^l + I \\ \zeta\left(v_n^l + a - b w_n^l\right) \end{bmatrix} + \sigma \sqrt{\Delta t_l} \begin{bmatrix} \xi_{1,n}^l \\ \xi_{2,n}^l \end{bmatrix}, \tag{5.68}$$

$$\begin{bmatrix} v_0^l \\ w_0^l \end{bmatrix} = \begin{bmatrix} v^0 \\ w^0 \end{bmatrix}, \quad n \in \{0, ..., N_{T,l} - 1\}, \tag{5.69}$$

where $\xi_{1,n}^l$ and $\xi_{2,n}^l$ are independently drawn realisations of standard normal random variables. The quantity of interest that we study is the following time average:

$$Q = \frac{1}{T} \int_0^T v^2(t)\,dt \approx \sum_{n=0}^{N_{T,l}-1} \left(\frac{(v_n^l)^2 + (v_{n+1}^l)^2}{2} \right) \frac{\Delta t_l}{T} =: Q_l. \tag{5.70}$$

To compute the sensitivities $Q_{z,l}$, we utilize the method of adjoints. We consider the corresponding adjoint variables λ_n^l and ν_n^l corresponding to v_n^l and w_n^l, $n \in \{1, ..., N_{T,l}\}$ respectively. The adjoint equation reads as follows:

$$\begin{bmatrix} \lambda_n^l \\ \nu_n^l \end{bmatrix} = \begin{bmatrix} \lambda_{n+1}^l \\ \nu_{n+1}^l \end{bmatrix} + \Delta t_l \left(\begin{bmatrix} (1 - (v_n^l)^2) & \zeta \\ -1 & -\zeta b \end{bmatrix} \begin{bmatrix} \lambda_{n+1}^l \\ \nu_{n+1}^l \end{bmatrix} + \begin{bmatrix} \frac{2 v_n^l}{T} \\ 0 \end{bmatrix} \right), \tag{5.71}$$

$$\begin{bmatrix} \lambda_{N_{T,l}}^l \\ \nu_{N_{T,l}}^l \end{bmatrix} = \Delta t_l \begin{bmatrix} \frac{v_n^l}{T} \\ 0 \end{bmatrix}, \quad n \in \{1, ..., N_{T,l} - 1\}. \tag{5.72}$$

The reader is referred to Appendix 5.B for the details of the derivation.

Once the adjoint equation is solved backwards in time, the approximation $Q_{z,l}$ of the sensi-

tivities Q_z at level l can then be obtained as follows:

$$\begin{aligned} Q_{a,l} &= \sum_{n=0}^{N_{T,l}-1} \Delta t_l \zeta v_{n+1}^l, \qquad & Q_{b,l} &= -\sum_{n=0}^{N_{T,l}-1} \Delta t_l \zeta w_n^l v_{n+1}^l, \\ Q_{I,l} &= \sum_{n=0}^{N_{T,l}-1} \Delta t_l \lambda_{n+1}^l, \qquad & Q_{\zeta,l} &= \sum_{n=0}^{N_{T,l}-1} \Delta t_l (v_n^l + a - b w_n^l) v_{n+1}^l. \end{aligned} \tag{5.73}$$

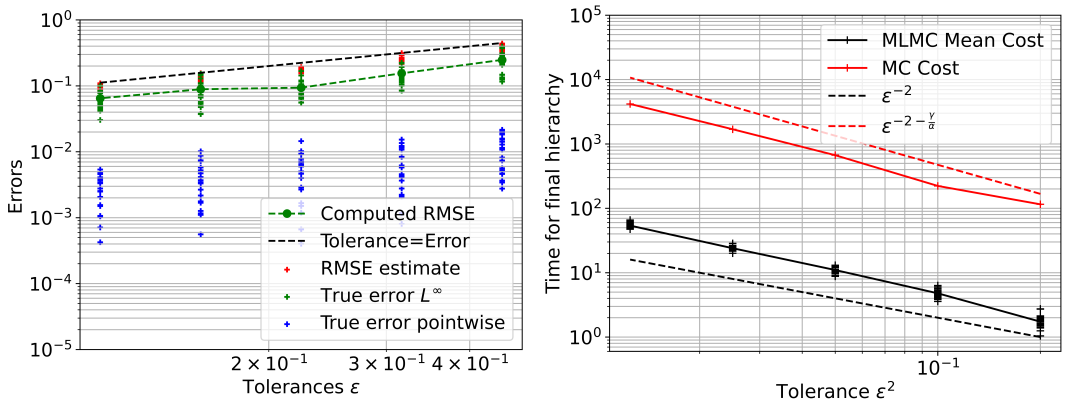
To demonstrate the performance of Algorithm 5, we assess the performance individually of its two components; firstly, the performance of the CMLMC algorithm, the error estimation procedure and the adaptive strategy described in Section 5.3 for accurately estimating the gradient for a given design, and secondly, the gradient based optimisation procedure described in Algorithm 5. We first assess the performance of the CMLMC algorithm and adaptive strategy. We remark that the solution of the forward and adjoint problems, as well as the CMLMC procedure, are implemented within the XMC software library [3], which we use for the simulations presented herein.

We seek to accurately estimate the gradient $\nabla \mathscr{J}(\cdot, z_0)$ using the estimator $\nabla \hat{\mathscr{J}}(\cdot, z_0)$, where $z_0 = [0.7, 0.8, 0.08, 1.0]$ and we set $\tau = 0.70$ for the significance of the CVaR. We set the parameters $z_r ef = z_0$ and $\kappa = 5.0$. The gradient and gradient error are estimated using the MLMC procedure described in Sections 5.3. To assess the reliability of the error bound derived in Proposition 5.3.1, we run a reliability study wherein we adapt the parameters of the MLMC hierarchy to attain a prescribed tolerance on $\mathrm{MSE}\left(\nabla \hat{\mathscr{J}}(\cdot, z_0)\right)$. We run the MLMC algorithm 20 times for each tolerance tested and compare the estimated error to the true error obtained using a reference gradient computed using a Monte Carlo estimator with 2×10^5 samples and 2×10^4 time steps. Specifically, we are interested in assessing the tightness of the inequality in Eq. (5.64).

The resultant plot is shown in Fig. 5.2a. Three errors are plotted in Fig. 5.2a; namely, the true error on the gradient, defined in the L^∞ sense, corresponding to the term on the leftmost side of Eq. (5.64), the square root of the MSE estimate on the gradient, produced by the optimally calibrated MLMC hierarchy, corresponding to the term on the rightmost side of Eq. (5.64), and the true error on the gradient evaluated at the point (θ_0, z_0), where θ_0 corresponds to the 70%-VaR for the design z_0. The true errors are computed with respect to a reference solution computed using 2×10^5 samples and 2×10^4 time steps. As can be seen from the figure, the MSE estimator provides a tight bound on the true error on the parametric expectations. However, the true error on the gradient in the L^∞ sense is much larger than the true pointwise error. This is a natural consequence of using the L^∞-norm over the entire interval Θ to define the MSE, as compared to using the pointwise error. Controlling the MSE error in an L^∞ sense, as defined in Eq. (5.43), is necessitated by the error accuracy condition in Eq. (5.20), in order to ensure exponential convergence of Algorithm 5.

Fig. 5.2b shows the complexity behaviour of the MLMC estimator calibrated using the CMLMC algorithm. We compute the cost required to obtain the final optimal hierarchy for a

given tolerance ϵ^2 on $\mathrm{MSE}\left(\nabla \hat{\mathscr{J}}(\cdot, z_0)\right)$. As can be seen from the figure, the cost grows as ϵ^{-2}, which is the theoretically predicated best case performance for the MLMC estimator. For comparison, we also plot the estimated cost of a comparable Monte Carlo estimator, as well as the expected cost growth rate for the case of the first order time discretisation used here. The Monte Carlo reference cost is computed as described in [16].



(a) Reliability of error estimator

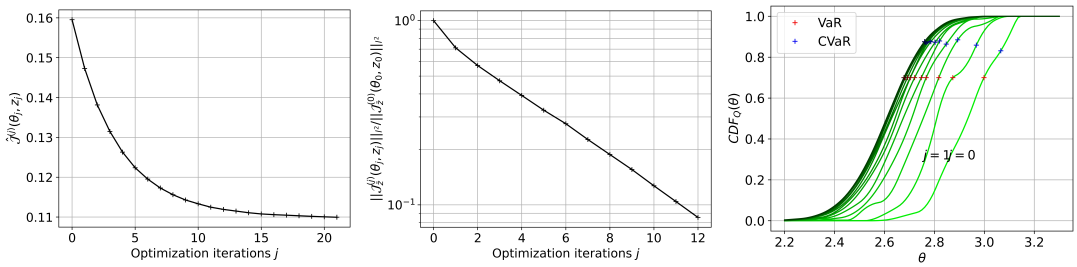
(b) Complexity behaviour

Figure 5.2: Error estimator performance for the CMLMC estimator of the gradient for the FitzHugh–Nagumo system

We now examine the performance of the gradient descent algorithm proposed in Section 5.2. We are interested in solving the minimisation problem given in Eq. (5.6), with $\tau = 0.7$. We utilize the framework of Algorithm 5, with a tolerance $\epsilon = 0.01$ on the gradient ratio. This implies that we stop the algorithm once the gradient magnitude has dropped to $1/100^{\text{th}}$ of its initial magnitude. As an initial guess, we begin with the design $z_0 = [0.7, 0.8, 0.08, 1.0]$. We also set $z_{ref} = [0.7, 0.8, 0.08, 1.0]$. We combine the above with the CMLMC algorithm detailed in [16] and detailed further in Section 5.3, with $\eta = 0.2$ on the relative error on the gradient.

We plot in Fig. 5.3a the value of the objective function for different iterations of the objective function. We observe exponential convergence in the number of iterations towards the final value, as predicted by Theorem 5.2.1, although we cannot guarantee that the hypotheses of Theorem 5.2.1 are satisfied for this problem. Fig. 5.3b shows the value of the gradient ratio r for different iterations of the optimisation algorithm. We also observe that the gradient decreases exponentially. Lastly, we plot in Fig. 5.3c the CDF of the output QoI $Q(z_j, \cdot)$ computed at different iterations of the optimisation algorithm, as well as the predicted VaR and CVaR values. We observe that the CDF, the VaR and the CVaR all move left, reducing the mass in the right tail of the distribution. Since we are minimising the CVaR, defined as the expectation of the random variable above the VaR, this translates to moving the right tail of the distribution as much as possible to the left.

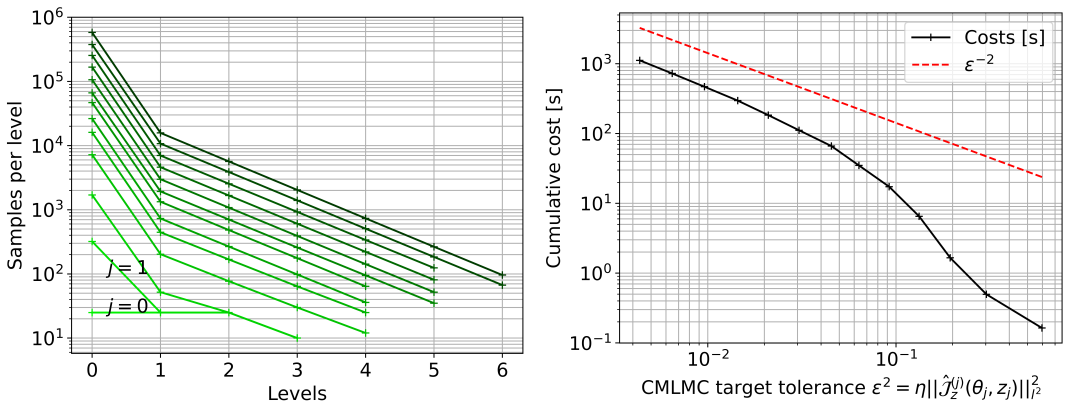
Fig. 5.4a shows the optimal hierarchy produced by the CMLMC algorithm at each iteration of the optimisation. We observe that since the tolerance supplied to the CMLMC algorithm



(a) Objective function decay. (b) Gradient ratio decay. (c) Change in CDF.

Figure 5.3: Performance of Algorithm 5 over different iterations for the FitzHugh–Nagumo system

is a fraction of the gradient magnitude, the optimally tuned hierarchy becomes larger for later iterations of the optimisation. In addition, Fig. 5.4b shows the cumulative cost required for the optimisation algorithm to reach a given gradient magnitude. The cumulative cost at a given optimisation iteration is defined as the sum of costs of all optimal hierarchies until the current optimisation iteration. Specifically, the cumulative cost is computed as $\sum_{i=0}^{j}\sum_{l=0}^{L} N_l^{(i)}(\text{Cost}(Q_l)+\text{Cost}(Q_{l-1}))$, where $\{N_l^{(i)}\}_{l=0}^{L}$ denote the optimal level-wise sample sizes for the ith optimisation iteration and $\text{Cost}(Q_l)$ denotes the average cost of simulating one sample of Q_l. This cost is plotted versus the gradient magnitude. We observe that after an initial pre-asymptotic regime, the cumulative cost grows as $\left\|\tilde{\nabla}\hat{\mathscr{J}}^{j}(w_j)\right\|_{l^2}^{-2}$, a rate comensurate with the use of an optimally tuned MLMC hierarchy at each iteration tuned to obtain a tolerance proportional to $\left\|\tilde{\nabla}\hat{\mathscr{J}}^{j}(w_j)\right\|_{l^2}$.



(a) Level-wise sample sizes (b) Cumulative cost

Figure 5.4: Hierarchy of CMLMC estimators and complexity behaviour of Algorithm 5 for different iterations for the FitzHugh–Nagumo system

5.4.2 Pollutant transport problem

We now apply the methodology to a more applied problem of practical relevance. We study a problem of pollutant transport, where the concentration of pollutant in a domain is modelled using a steady reaction-diffusion-advection equation. We consider a square domain $D = (0,1)\times(0,1)$, with boundary $\partial D := \Gamma_d \cup \Gamma_n$, where $\Gamma_d := \{0\}\times(0,1)$ and $\Gamma_n := \partial D \backslash \Gamma_d$. We denote by $u : D \times \mathbb{R}^9 \times \Omega \to \mathbb{R}$ the concentration of the pollutant. The concentration satisfies the following equation:

$$-\nabla\cdot(\epsilon\nabla u(x,z,\omega)) + \mathbb{V}(x,\omega)\cdot\nabla u(x,z,\omega) = f(x) - B(x,z), \quad x \in D, \tag{5.74}$$

subject to the following boundary conditions:

$$\epsilon\frac{\partial u}{\partial n}(x,z,\omega) = 0, \quad x \in \Gamma_n, \quad \text{for } \mathbb{P}-\text{a.e. } \omega \in \Omega \tag{5.75}$$

$$u(x,z,\omega) = 0, \quad x \in \Gamma_d, \quad \text{for } \mathbb{P}-\text{a.e. } \omega \in \Omega, \tag{5.76}$$

where $\epsilon > 0$ denotes a viscosity parameter. $\mathbb{V}(x,\omega)$ is a random divergence-free velocity field defined as follows:

$$\mathbb{V}(x,\omega) := \begin{bmatrix} b(\omega) - a(\omega)x_1 \\ a(\omega)x_2 \end{bmatrix}, \tag{5.77}$$

where $a \sim \mathcal{U}[4.95, 5.05]$ and $b \sim \mathcal{U}[3.95, 3.05]$ are uniformly distributed random variables, and x_1 and x_2 denote the components of x. The source $f(x)$ is the sum of five Gaussian source terms:

$$f(x) = \sum_{i=1}^{5} s_i \exp\left(-\frac{(x-\mu_i)^T(x-\mu_i)}{2\sigma_i^2}\right), \tag{5.78}$$

where the values of s_i, μ_i and σ_i are given in Table 5.1. The sink term $B(x,z)$ is defined as follows:

$$B(x,z) = \sum_{k=1}^{9} z^k \exp\left(-\frac{(x-p_k)^T(x-p_k)}{2\sigma^2}\right), \tag{5.79}$$

where the locations p_k are defined as $p_k = (0.25i, 0.25j)$, $i, j \in \{1,2,3\}$, $k = 3(i-1)+j$, $\sigma = 0.05$, and z^k denotes the kth component of $z \in \mathbb{R}^9$. We are interested in studying the distribution of the random QoI Q, defined as follows:

$$Q(z,\omega) := \frac{\kappa_s}{2}\int_D u^2(x,z,\omega)\,dx, \tag{5.80}$$

with $\kappa_s = 10^4$.

The problem is implemented using the FEniCS finite element software [92]. The domain is discretised using a uniform triangular mesh with piecewise linear finite elements. The

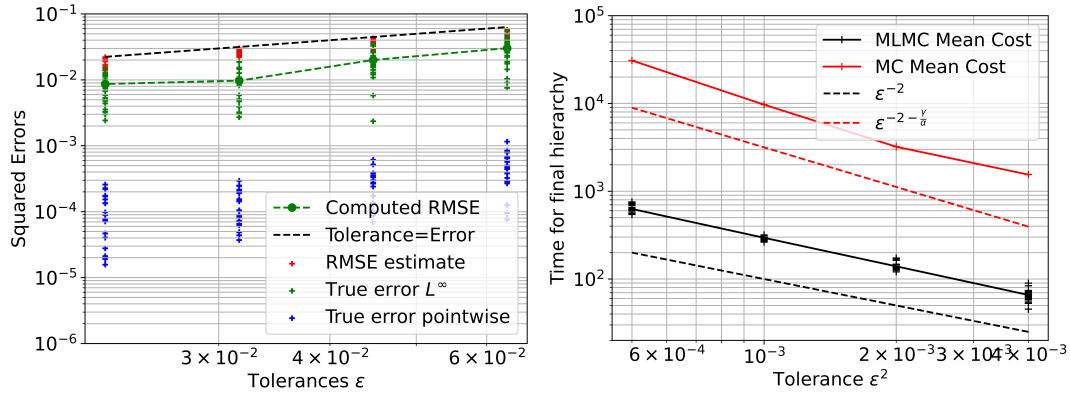
i	μ_i	σ_i	s_i
1	$[0.55205319, 0.65571641]^T$	0.0229487	2.3220339
2	$[0.49379544, 0.10950509]^T$	0.0205321	1.7931427
3	$[0.13032797, 0.57569277]^T$	0.0196891	2.3522452
4	$[0.33868732, 0.37971428]^T$	0.0212297	2.2850373
5	$[0.27670822, 0.15833522]^T$	0.0227373	2.3194400

Table 5.1: Source term parameters for the pollutant transport problem

resultant linear system is solved using a sparse direct solver [9, 8]. The number of elements per side of the square domain varies as $32 \times 2^{l/2}$, $l \in \{0, 1, ..., L\}$, leading to a mesh size h_l that varies as $h_l = h_0 \times 2^{-l}$. An in-built automatic differentiation module within the FEniCS library is used to compute the sensitivities of the QoI with respect to design parameters. Once again, the XMC software library [3] is used to implement the CMLMC procedure.

Similar to Section 5.4.1, we seek to examine both parts of the optimisation algorithm; namely the CMLMC and the gradient based OUU algorithm. For the CMLMC, we seek to accurately estimate $\nabla\hat{\mathscr{J}}(\cdot, z_0)$, where $z_0 = z_{ref} = [0.1]^9$, such that $\mathrm{MSE}\left(\nabla\hat{\mathscr{J}}(\cdot, z_0)\right)$ satisfies a prescribed tolerance. Fig. 5.5 shows the results of reliability and complexity studies conducted for the above parameters, similar to the one conducted for the FitzHugh–Nagumo system in Section 5.4.1. For studying the reliability of the error estimators, we conduct 20 independent CMLMC simulations for a given tolerance. For each simulation, we plot three errors; namely the true L^∞ error on the gradient, the square root of the MSE estimate produced by our error estimation procedure described in Section 5.3, and the true pointwise error on the gradient, computed by evaluating the parametric expectations $\nabla\hat{\mathscr{J}}(\cdot, z_0)$ for the gradient at θ_0, the VaR corresponding to the design z_0. The reference value of the gradient is computed by first running 20 simulations for a tolerance that is half of the finest tested tolerance, and averaging over the gradient estimates produced by these simulations. Similar to before, we find that although our novel error estimators provide a tight bound on the true L^∞ error of the gradient, the L^∞ error on the gradient is significantly larger than the error on the gradient evaluated at θ_0. Fig. 5.5b presents the complexity results of the CMLMC algorithm. The cost to compute the optimal hierarchy for a given tolerance ϵ^2 on $\mathrm{MSE}\left(\nabla\hat{\mathscr{J}}^j(\cdot, z_0)\right)$ is plotted versus the tolerance, for each of the 20 CMLMC simulations at a given tolerance, in addition to their sample average value. In addition, the theoretical cost growth rate of a comparable Monte Carlo estimator is shown, as well as the estimated cost of the estimator for reference and comparison. The Monte Carlo reference cost is computed as described in [16]. As can be seen from the figure, the complexity follows the theoretically predicted complexity ϵ^{-2}.

For the OUU, we wish to minimise an objective function of the form in Eq. (5.6), with $z_{ref} = [0.0]^9$, $\kappa = 1.0$, and for significance of $\tau = 0.7$. This implies that we seek to minimise the CVaR while also minimising the amplitude of the controlled sinks. We utilise Algorithm 5, starting from a design $z_0 = [0.1]^9$, and halt the optimisation once a gradient ratio of $r = 0.08$ has been

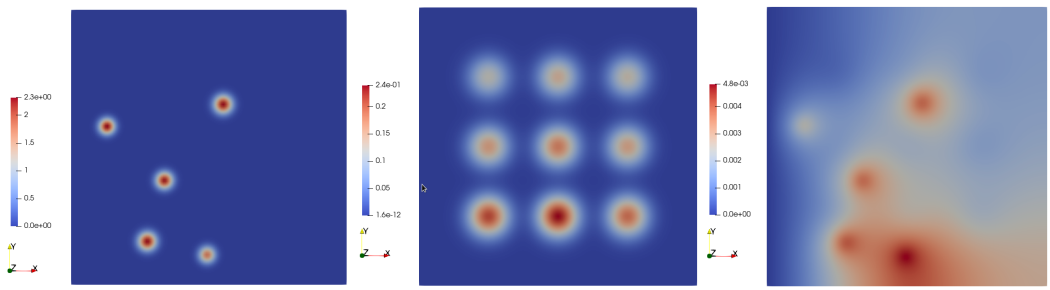


(a) Reliability of error estimator

(b) Complexity behaviour

Figure 5.5: Error estimator performance for the pollutant transport problem

achieved. In Fig. 5.6, we show the source field $f(x)$, the control field $B(x,z^*)$ and the solution $u(x,z^*,\omega)$ for the mean conditions $a(\omega)=4$ and $b(\omega)=5$ at the optimal control z^* obtained by solving problem (5.6).



(a) $f(x)$

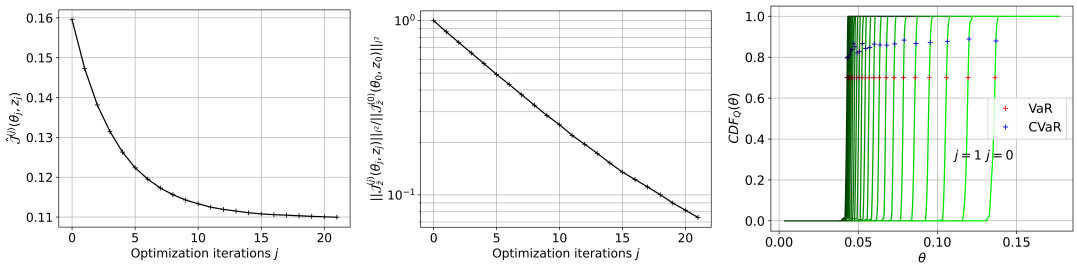
(b) $B(x,z^*)$

(c) $u(x,z^*,\omega)$

Figure 5.6: Source, control and solution fields for the pollutant transport problem for $a(\omega)=4$ and $b=5(\omega)$, and for $x\in D$

Fig. 5.7a shows the decay of the objective function towards its final value. We once again observe exponential convergence in the optimisation counter j, as predicted by Theorem 5.2.1. In addition, we plot in 5.3b the gradient ratio for different iterations of the optimisation, which also decreases exponentially in the iteration counter j. Fig. 5.7c shows the CDF of the output QoI $Q(z_j,\cdot)$ for different iterations j of the optimisation algorithm, along with the estimated VaR and CVaR. The CDF, the VaR and the CVaR all move left as before in Section 5.4.1, which translates to moving the right tail of the distribution as much as possible to the left.

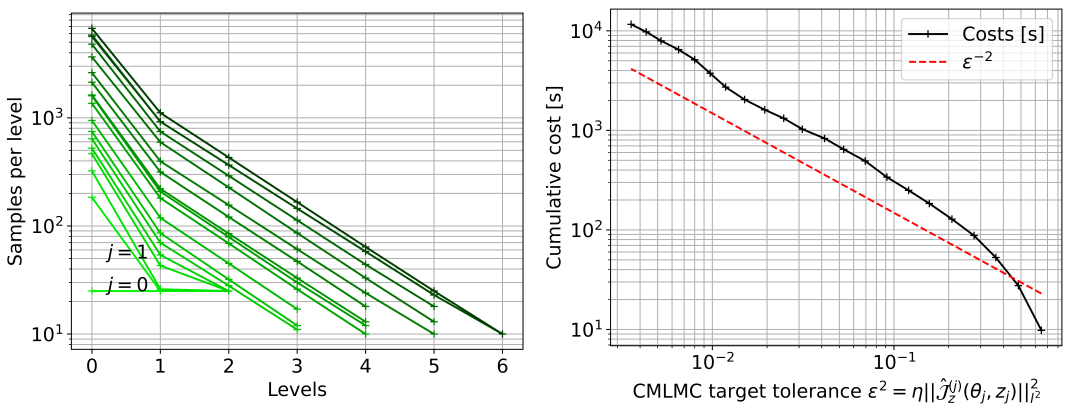
Fig. 5.9a shows the optimal hierarchy produced by the CMLMC algorithm at each iteration of the optimisation for a given tolerance. Similar to before, we observe that the optimally tuned hierarchy increases in size for later optimisation iterations since the tolerance supplied to the CMLMC is a fraction of the gradient magnitude. Fig. 5.5b shows the cumulative



(a) Objective function decay. (b) Gradient ratio decay. (c) Change in CDF.

Figure 5.7: Optimization performance over different iterations for the pollutant transport problem

cost, as defined in Section 5.4.1, for a given gradient magnitude. We observe once again that the cumulative cost grows as $\|\tilde{\nabla}\hat{\mathscr{J}}^j\|_{l^2}^{-2}$ after an initial pre-asymptotic regime, as is to be expected for the use of an optimally tuned MLMC hierarchy at each iteration, tuned to obtain a tolerance proportional to $\|\tilde{\nabla}\hat{\mathscr{J}}^j\|_{l^2}^{2}$.

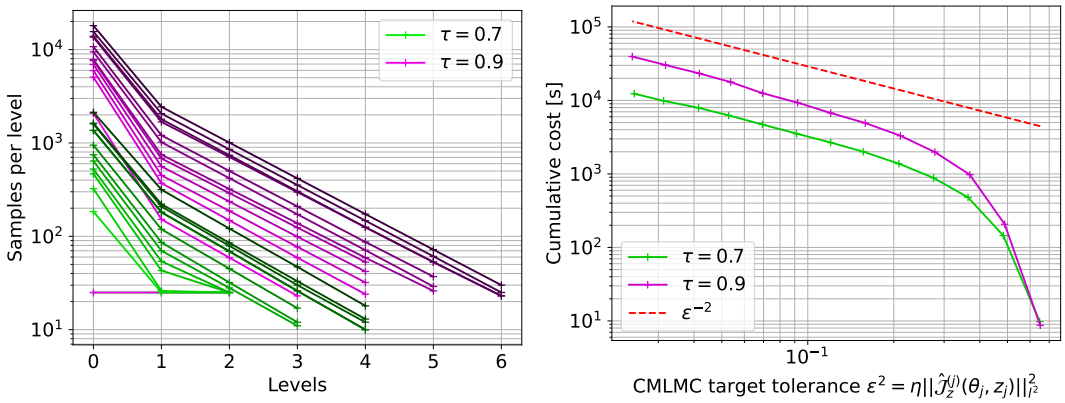


(a) Level-wise sample sizes (b) Cumulative cost

Figure 5.8: Hierarchy and complexity behaviour for different iterations for the pollutant transport problem

We now wish to study the performance of the AMGD algorithm for different significances τ. To this end, we compare the performance of the algorithm for $\tau = 0.7$ and $\tau = 0.9$. Since the performance of the algorithm in terms of objective function and gradient decay in the $\tau = 0.9$ case are nearly identical to the performance observed in Fig. 5.8 for the $\tau = 0.7$, the corresponding results are not presented here. Fig. 5.9a shows the optimal hierarchy produced by the CMLMC algorithm at each optimisation iteration for the two significances tested. We observe that the level-wise sample sizes N_l decay at the same rate in the levels l for both tested significances, however with a larger constant for the $\tau = 0.9$ case. Additionally, Fig. 5.9b shows the cumulative cost for a given gradient magnitude, for both significances. We observe that the cumulative cost grows as $\|\tilde{\nabla}\hat{\mathscr{J}}^j(w_j)\|_{l^2}^{-2}$ in both cases, following an initial pre-asymptotic regime. However, the $\tau = 0.9$ case shows a larger constant. In this case, the interval Θ is

changed with each optimisation iteration such that it is centered on the quantile estimate corresponding to the previous optimisation iteration. It can be shown, for the case of a simple Monte Carlo estimator, that the constant is expected to scale in this case as $(1-\tau)^{-1}$. We note that we observe a similar scaling in this case.



(a) Level-wise sample sizes

(b) Cumulative cost

Figure 5.9: Hierarchy and complexity behaviour for different significances for pollutant transport problem

5.5 Conclusions

The aim of this work was to tackle the challenge of minimising the CVaR of a random QoI, typically the output of a differential model with random inputs, over a suitable design space, using gradient-based optimisation techniques. A main challenge in utilising gradient-based techniques was the differentiability of the CVaR in terms of the design variables. A differentiability result was presented in Section 5.1, which was a generalisation of the one presented in [74], showing that gradient-based algorithms could still be used to directly minimise the CVaR without requiring smoothing.

The expression for the sensitivities of the CVaR with respect to design parameters required the computation of expectations of discontinuous functions of the QoI; namely, the indicator function. Estimating this expectation naively using MLMC estimators could become impractically expensive, and possibly result in non-optimal complexity behaviour of the corresponding MLMC estimator. A similar issue was discussed and tackled in [16], and an alternative was proposed using the framework of parametric expectations. We presented a modified expression for the sensitivities of the CVaR, based on derivatives of parametric expectations, thereby allowing us to use the work in [16]. Based on this modification, we also presented a novel optimisation algorithm consisting of an alternating minimisation-gradient procedure. We demonstrated a theoretical result that, under additional assumptions on the combined objective function in Eq. (5.6), the novel algorithm would achieve exponential convergence of the design iterates towards the optimal design in the optimisation iterations.

To enable the use of the work in [16], we presented modifications of the MLMC estimator, the error estimation procedure and adaptive hierarchy selection procedure specific to computing the sensitivities of the CVaR. Namely, a relation was derived between the MSE of the sensitivities and the MSE of the parametric expectations in Section 5.3. In addition, a modification of the KDE smoothing procedure presented in [16] was presented, specific to CVaR minimisation. The combination of the MSE relation and KDE modification allowed us to trivially extend the error estimation and hierarchy adaptivity procedure of [16] to the current application. Lastly, a minor modification of the CMLMC procedure of [16] was presented in Algorithm 5, wherein the CMLMC was restarted from the optimal hierarchy of the previous design iterate.

The combination of gradient-based optimisation and MLMC estimation of the sensitivities of the CVaR was tested on two problems of practical relevance; namely the FitzHugh–Nagumo oscillator and a more applied problem of advection-reaction-diffusion problem used to model pollutant transport. In both cases, it was observed that the novel error estimation procedure provided tight bounds on the MSE of the gradient as defined in Eq. (5.43). In addition, the CMLMC algorithm was shown to produce the best-case complexity behaviour for the MLMC estimators of the sensitivities. The OUU algorithm was shown to converge exponentially in the optimisation iterations, while also preserving the best case MLMC cost complexity.

The numerical examples considered in this work demonstrated that the AMGD procedure performs well for the cases presented here. However, one may wish to improve on the performance of the algorithm by considering alternatives to the AMGD algorithm. Such variations could, for example, include higher order optimisation methods such as the Newton method. It still remains to be seen whether higher order method can be used directly with objective functions of the type in problem (5.6), as well as whether the framework of parametric expecations can be combined with such an algorithm. The authors of [31, 32], for example, have introduced a novel stochastic adaptive BFGS algorithm. We plan to explore such questions in future works.

5.A Proof of Theorem 5.1.1

To prove Theorem 5.1.1 on the Fréchet differentiability of the objective function $\mathscr{J}(\theta, z)$, we first prove an important result in Lemma 5.A.1. We recall that $\Gamma \subset \mathrm{L}^p(\Omega, \mathbb{R})$ is the set of L^p-integrable random variables whose measures are atom-free.

Lemma 5.A.1. *Consider random variables $Y \in \Gamma \subset \mathrm{L}^p(\Omega, \mathbb{R})$ and $\delta Y \in \mathrm{L}^p(\Omega, \mathbb{R})$. We then have the following:*

$$\lim_{\|\delta Y\|_{\mathrm{L}^p} \to 0} \mathbb{E}\left[\mathbb{1}_{\{0 \le Y \le -\delta Y\}}\right] = 0, \tag{5.81}$$

$$\textit{and} \quad \lim_{\|\delta Y\|_{lp} \to 0} \mathbb{E}\left[\mathbb{1}_{\{-\delta Y \le Y \le 0\}}\right] = 0. \tag{5.82}$$

Proof. We begin with the proof for Eq. (5.81), since the proof for Eq. (5.82) follows from identical arguments. We make use of the following result; for any $X \in \mathrm{L}^p(\Omega, \mathbb{R})$, the following holds for any $\epsilon > 0$ and $p \geq 0$:

$$\mathbb{E}\left[\mathbb{1}_{\{|X| \geq \epsilon\}}\right] \leq \mathbb{E}\left[\frac{|X|^p}{\epsilon^p}\right] = \|X\|_{\mathrm{L}^p}^p \epsilon^{-p}. \tag{5.83}$$

Setting $\epsilon = \|X\|_{\mathrm{L}^p}^{\beta}$ for some $\beta \in [0, 1)$, we have that

$$\mathbb{E}\left[\mathbb{1}_{\{|X| \geq \|X\|_{\mathrm{L}^p}^{\beta}\}}\right] \leq \|X\|_{\mathrm{L}^p}^{p-\beta p} = \|X\|_{\mathrm{L}^p}^{\gamma}, \tag{5.84}$$

where $\gamma := p(1-\beta)$. We rewrite the term within the limit in Eq. (5.81) as follows:

$$\mathbb{E}\left[\mathbb{1}_{\{0 \leq Y \leq -\delta Y\}}\right] = \mathbb{E}\left[\mathbb{1}_{\{0 \leq Y \leq -\delta Y\}}\left(\mathbb{1}_{\{|\delta Y| < \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}} + \mathbb{1}_{\{|\delta Y| \geq \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right)\right] \tag{5.85}$$

$$= \mathbb{E}\left[\mathbb{1}_{\{0 \leq Y \leq -\delta Y\}} \mathbb{1}_{\{|\delta Y| < \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right] + \mathbb{E}\left[\mathbb{1}_{\{0 \leq Y \leq -\delta Y\}} \mathbb{1}_{\{|\delta Y| \geq \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right]. \tag{5.86}$$

The first term can be bounded as follows:

$$\mathbb{E}\left[\mathbb{1}_{\{0 \leq Y \leq -\delta Y\}} \mathbb{1}_{\{|\delta Y| < \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right] \leq \mathbb{E}\left[\mathbb{1}_{\{0 \leq Y \leq \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right] \tag{5.87}$$

Due to dominated convergence, we can pass the limit into the expectation, resulting in the following:

$$\lim_{\|\delta Y\|_{\mathrm{L}^p} \to 0} \mathbb{E}\left[\mathbb{1}_{\{0 \leq Y \leq \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right] = \mathbb{E}\left[\lim_{\|\delta Y\|_{\mathrm{L}^p} \to 0} \mathbb{1}_{\{0 \leq Y \leq \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right] = \mathbb{E}\left[\mathbb{1}_{\{Y=0\}}\right] = 0, \tag{5.88}$$

since $Y \in \Gamma$ is atom-free. The second term can be bounded as follows, where we use a Hölder inequality:

$$\mathbb{E}\left[\mathbb{1}_{\{0 \leq Y \leq -\delta Y\}} \mathbb{1}_{\{|\delta Y| \geq \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right] \leq \left\|\mathbb{1}_{\{0 \leq Y \leq -\delta Y\}}\right\|_{\mathrm{L}^\infty} \left\|\mathbb{1}_{\{|\delta Y| \geq \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right\|_{\mathrm{L}^1} \tag{5.89}$$

$$\leq \mathbb{E}\left[\mathbb{1}_{\{|\delta Y| \geq \|\delta Y\|_{\mathrm{L}^p}^{\beta}\}}\right] \tag{5.90}$$

$$\leq \|\delta Y\|_{\mathrm{L}^p}^{\gamma}. \tag{5.91}$$

Hence, we have that the second term in Eq. (5.86) goes to zero as well with the application of the limit, thus concluding the proof for Eq. (5.81). The proof for Eq. (5.82) follows from identical arguments. □

Proof of Theorem 5.1.1. We note that the function $\mathscr{R}(\theta, Q)$ is a composition of two functions.

We define the functions $l_1 : \Gamma \to \mathbb{R}$ and $l_2 : \mathbb{R} \times \Gamma \to \Gamma$ as follows:

$$l_1(Y) := -\mathbb{E}\left[Y^+\right] \tag{5.92}$$

$$l_2(\theta, Q) := Q - \theta, \tag{5.93}$$

$$\implies \mathscr{R}(\theta, Q) = \theta + \frac{l_1 \circ l_2(\theta, Q)}{1-\tau}. \tag{5.94}$$

Hence, to show that $\mathscr{R}$ is Fréchet differentiable, it suffices to show that each of the functions l_1 and l_2 are Fréchet differentiable.

It is straightforward to see that l_2 is Fréchet differentiable (being linear and bounded) with Fréchet derivative $Dl_2(\theta, Q)$ in the direction $(\delta\theta, \delta Q) \in \mathbb{R} \times \mathrm{L}^p(\Omega, \mathbb{R})$ given by:

$$Dl_2(\theta, Q)(\delta\theta, \delta Q) = \delta Q - \delta\theta. \tag{5.95}$$

The Fréchet derivative of l_1 however, requires some consideration. We argue that the Fréchet derivative of l_1 exists at any point $Y \in \Gamma$ and is given by $Dl_1(Y)(\delta Y) = \mathbb{E}\left[\mathbb{1}_{\{Y \geq 0\}}\delta Y\right]$. To prove this statement, we must verify the following limit:

$$\lim_{\|\delta Y\|_{\mathrm{L}^p} \to 0} \frac{\left|\mathbb{E}\left[(Y+\delta Y)^+\right] - \mathbb{E}\left[Y^+\right] - \mathbb{E}\left[\mathbb{1}_{\{Y \geq 0\}}\delta Y\right]\right|}{\|\delta Y\|_{\mathrm{L}^p}} = 0 \tag{5.96}$$

To show the above, we begin by re-writing the numerator as follows:

$$\begin{aligned}\mathbb{E}\left[(Y+\delta Y)^+ - Y^+ - \mathbb{1}_{\{Y \geq 0\}}\delta Y\right] = \mathbb{E}\left[\delta Y \mathbb{1}_{\{Y+\delta Y \geq 0, Y \geq 0\}} - \mathbb{1}_{\{Y \geq 0\}}\delta Y\right] \\ + \mathbb{E}\left[(Y+\delta Y)\mathbb{1}_{\{Y+\delta Y \geq 0, Y < 0\}}\right] \\ - \mathbb{E}\left[Y \mathbb{1}_{\{Y+\delta Y < 0, Y \geq 0\}}\right].\end{aligned} \tag{5.97}$$

Inserting Eq. (5.97) into Eq. (5.96), we have the following:

$$\frac{\left|\mathbb{E}\left[(Y+\delta Y)^+\right] - \mathbb{E}\left[Y^+\right] - \mathbb{E}\left[\mathbb{1}_{\{Y \geq 0}\delta Y\}\right]\right|}{\|\delta Y\|_{\mathrm{L}^p}} \leq \frac{T_1 + T_2 + T_3}{\|\delta Y\|_{\mathrm{L}^p}}, \tag{5.98}$$

with the terms T_1, T_2 and T_3 given by:

$$T_1 := \left|\mathbb{E}\left[\delta Y \mathbb{1}_{\{Y+\delta Y \geq 0, Y \geq 0\}} - \delta Y \mathbb{1}_{\{Y \geq 0\}}\right]\right|, \tag{5.99}$$

$$T_2 := \left|\mathbb{E}\left[(Y+\delta Y)\mathbb{1}_{\{Y+\delta Y \geq 0, Y < 0\}}\right]\right|, \tag{5.100}$$

$$T_3 := \left|\mathbb{E}\left[Y \mathbb{1}_{\{Y+\delta Y < 0, Y \geq 0\}}\right]\right|. \tag{5.101}$$

We then begin with the term T_1. We first note that T_1 can be rewritten in the following man-

ner:

$$T_1 = \left| -\mathbb{E}\left[\delta Y \mathbb{1}_{\{0 \le Y < -\delta Y\}}\right]\right| \le \mathbb{E}\left[|\delta Y| \, \mathbb{1}_{\{0 \le Y < -\delta Y\}}\right] \tag{5.102}$$

$$\le \|\delta Y\|_{\mathrm{L}^p} \left\| \mathbb{1}_{\{0 \le Y < -\delta Y\}} \right\|_{\mathrm{L}^q} = \|\delta Y\|_{\mathrm{L}^p} \, \mathbb{E}\left[\mathbb{1}_{\{0 \le Y < -\delta Y\}}\right]^{1/q}, \tag{5.103}$$

$$\le \|\delta Y\|_{\mathrm{L}^p} \, \mathbb{E}\left[\mathbb{1}_{\{0 \le Y \le -\delta Y\}}\right]^{1/q}. \tag{5.104}$$

The term T_2 can be bounded as follows:

$$T_2 = \left|\mathbb{E}\left[(Y + \delta Y) \mathbb{1}_{\{Y + \delta Y \ge 0\}} \mathbb{1}_{\{Y < 0\}}\right]\right| \le \mathbb{E}\left[|\delta Y| \mathbb{1}_{\{-\delta Y \le Y < 0\}}\right], \tag{5.105}$$

$$\le \|\delta Y\|_{\mathrm{L}^p} \left\| \mathbb{1}_{\{-\delta Y \le Y < 0\}} \right\|_{\mathrm{L}^q} = \|\delta Y\|_{\mathrm{L}^p} \, \mathbb{E}\left[\mathbb{1}_{\{-\delta Y \le Y \le 0\}}\right]^{1/q}. \tag{5.106}$$

Similarly, the term T_3 can be bounded as follows:

$$T_3 = \left|\mathbb{E}\left[Y \mathbb{1}_{\{Y + \delta Y < 0\}} \mathbb{1}_{\{Y \ge 0\}}\right]\right| \le \mathbb{E}\left[|\delta Y| \, \mathbb{1}_{\{0 \le Y < -\delta Y\}}\right] \tag{5.107}$$

$$\le \|\delta Y\|_{\mathrm{L}^p} \left\| \mathbb{1}_{\{0 \le Y \le -\delta Y\}} \right\|_{\mathrm{L}^q} = \|\delta Y\|_{\mathrm{L}^p} \, \mathbb{E}\left[\mathbb{1}_{\{0 \le Y \le -\delta Y\}}\right]^{1/q}. \tag{5.108}$$

Inserting Eqs. (5.104), (5.106) and (5.108) into Eq. (5.98), and applying the limit using Lemma 5.A.1, we have that:

$$\lim_{\|\delta Y\|_{\mathrm{L}^p} \to 0} \frac{\left|\mathbb{E}\left[(Y + \delta Y)^+\right] - \mathbb{E}\left[Y^+\right] - \mathbb{E}\left[\mathbb{1}_{\{Y \ge 0\}} \delta Y\right]\right|}{\|\delta Y\|_{\mathrm{L}^p}} = 0. \tag{5.109}$$

This concludes the proof. □

5.B Adjoint of first-order ODE with additive noise

We present here the derivation of the adjoints for a first-order Ordinary Differential Equation (ODE) with white noise forcing for an objective function containing the CVaR of a time-averaged quantity of the trajectory. Let $(\Omega, \mathscr{F}, \mathbb{P})$ be a complete probability space, $\omega \in \Omega$ denote an elementary random event, and $z \in \mathbb{R}^d$ the set of design variables. Let $u(t, z, \omega) \in U \subset \mathbb{R}^{N_u}$ be the state vector at time $t \in [0, T]$ for a given random input ω and design z. The state vector u is governed by the following ODE with additive noise.

$$\dot{u}(t, z, \omega) = g(u, z) + \tau \dot{W}(t, \omega) \quad \text{over } (0, T], \tag{5.110}$$

$$u(0, z, \omega) = u^0, \tag{5.111}$$

where $g : U \times \mathbb{R}^d \to \mathbb{R}^{N_u}$, and $W : [0, T] \times \Omega \to \mathbb{R}^{N_u}$ is a N_u-dimensional standard Wiener process.

We discretise the problem on a uniform temporal grid $\mathbb{T}$ where the interval $[0, T]$ is divided into $N \in \mathbb{N}$ segments of step size $\Delta t = T/N$, $\mathbb{T} := \{t_n := n\Delta t : n \in [\![0, N_l]\!]\}$. The ODE is discre-

tised using the Euler–Maruyama scheme, which reads as follows:

$$\begin{aligned} u^{n+1} &= u^n + \Delta t g(u^n, z) + \tau\sqrt{\Delta t}\xi^n, \\ u^0 &= u_0, \end{aligned}$$

where u^n denotes the approximation to $u(t_n, z, \omega)$, $\xi^n \in \mathbb{R}^{N_u}$ are N_u-dimensional random vectors whose components are independent identically distributed standard normal variables. We are interested in computing the statistics of time-averages of functions of the trajectory.

$$Q = \langle f(u) \rangle_T. \tag{5.112}$$

We approximate the time integral using the trapezoid rule on the aforementioned temporal grid, leading to

$$Q(z, \omega) \approx Q_h(z, \omega) := \sum_{n=0}^{N-1} \left(\frac{f(u^n) + f(u^{n+1})}{2} \right) \frac{\Delta t}{T}. \tag{5.113}$$

We are interested in minimising the CVaR of this quantity over the parameters z but use the combined formulation in Eq. (5.6). The corresponding Lagrangian for the problem reads

$$\mathscr{L}(\theta, z, \{u^n\}, \{\lambda^n\}) = \theta + \frac{\mathbb{E}\left[(Q(z,\cdot) - \theta)^+\right]}{1-\tau} + \mathbb{E}\left[\sum_{n=0}^{N-1} \lambda^{n+1} \left(u^n + \Delta t g^n + \tau\sqrt{\Delta t}\xi^n - u^{n+1} \right) - \lambda^0 (u^0 - u_0) \right], \tag{5.114}$$

where we use $g^n := g(u^n, z)$, and $\lambda^n \in \mathbb{R}^{N_u}$, $n \in [\![0, N]\!]$ denote the Lagrange multipliers for the initial condition and the steps of the discretised equations.

Differentiating with respect to z gives

$$\begin{aligned} \frac{\mathrm{d}\mathscr{L}}{\mathrm{d}z} &= \mathbb{E}\left[\frac{\mathbb{1}_{Q_h \geq \theta}}{(1-\tau)T} \sum_{n=0}^{N-1} \left(\frac{f_u^n u_z^n + f_u^{n+1} u_z^{n+1}}{2} \right) \Delta t \right] && (5.115) \\ &\quad + \mathbb{E}\left[\sum_{n=0}^{N-1} \lambda^{n+1} \left(u_z^n + \Delta t (g_u^n u_z^n + g_z^n) - u_z^{n+1} \right) \right] \\ &=: \mathbb{E}\left[\hat{\mathscr{L}}\right]. && (5.116) \end{aligned}$$

Re-arranging the terms leads to

$$\begin{aligned} \hat{\mathscr{L}} = u_z^0 \left[\lambda^1 (1 + \Delta t g_u^0) + \frac{\mathbb{1}_{Q_h \geq \theta}}{(1-\tau)T} \frac{f_u^0 \Delta t}{2} \right] + \Delta t \lambda^1 g_z^0 + u_z^N \left[\frac{\mathbb{1}_{Q_h \geq \theta}}{(1-\tau)T} \frac{f_u^N \Delta t}{2} - \lambda^N \right] \\ + \sum_{n=1}^{N-1} u_z^n \left[\lambda^{n+1} (1 + \Delta t g_u^n) - \lambda^n + \frac{\mathbb{1}_{Q_h \geq \theta}}{(1-\tau)T} \Delta t f_u^n \right] + \Delta t \lambda^{n+1} g_z^n, \end{aligned} \tag{5.117}$$

where we have used the subscript notation for partial derivatives.

We have in our case that $u_z^0 = 0$. To remove terms dependent on u_z^n, we set

$$\lambda^n = \lambda^{n+1}(1 + \Delta t g_u^n) + \frac{\mathbb{1}_{Q_h \geq \theta}}{(1-\tau)T} \Delta t f_u^n, \quad n = 1, \ldots, N-1 \tag{5.118}$$

$$\lambda^N = \frac{\mathbb{1}_{Q_h \geq \theta}}{(1-\tau)T} \frac{f_u^N \Delta t}{2}. \tag{5.119}$$

This gives us the adjoint equations which are solved backwards in time. It is noteworthy to mention that since Eq. (5.118) is linear, that it can be solved for $\{\lambda^n\}$ without the factor $\frac{\mathbb{1}_{Q_h \geq \theta}}{(1-\tau)T}$, and equivalently, the sensitivities can be computed as:

$$\frac{\mathrm{d}\mathscr{L}}{\mathrm{d}z} = \frac{\mathbb{1}_{Q_l \geq \theta}}{(1-\tau)T} \mathbb{E}\left[\sum_{n=0}^{N-1} \Delta t \lambda^{n+1} g_z^n\right]. \tag{5.120}$$

That is, setting

$$\mathscr{J}(\theta, z) = \theta + \frac{\mathbb{E}\left[(Q_h(z,\cdot) - \theta)^+\right]}{1-\tau}, \tag{5.121}$$

we have that $\mathscr{J}_z(\theta, z) = \mathbb{E}\left[\sum_{n=0}^{N-1} \Delta t \lambda^{n+1} g_z^n\right]$.

Software Development and Production Simulations

Part III

6 Software Development

One of the main aims of the ExaQUte project was to enable the simulation of risk-averse engineering design problems at an exascale computing speed. This translates to a speed of 10^{18} double precision operations per second. Many mathematical, algorithmic and software tools were developed within the project towards this goal. We summarize, in this chapter, the software goals of the ExaQUte project and detail the contributions made by this thesis in this context. We first present the overall ExaQUte software framework, a scalable parallel implementation of the algorithms developed within the ExaQUte project that consists of a combination of individual software tools concurrently and collaboratively developed by the members of the ExaQUte consortium. We then describe in detail our main contribution to this software framework; namely, the X-Monte Carlo (XMC) software library [5], a Python library developed during this thesis research for hierarchical Monte Carlo methods. We also discuss the interfacing of the XMC library with other software packages developed within the consortium.

In Section 6.1, we introduce the ExaQUte software framework, and describe each of the component software libraries, wherein we introduce the XMC package and how it interfaces with the other software tools in the framework. Section 6.2 describes the structure and function of the XMC library developed by EPFL and contributed to by the author of this thesis. We compare it to the MLMC estimator introduced in Chapter 2, and demonstrate how the structure of the estimator has motivated the structure of the library. In Section 6.3, we introduce some notable simulations, whose results are not presented in detail in this thesis, that were computed using the ExaQUte framework and using XMC. Furthermore, in Chapter 7, we will introduce the main production simulation results of the ExaQUte project wherein the research produced in this thesis, implemented within the XMC package and the ExaQUte software framework, was used for risk-estimation and risk-averse optimisation.

6.1 ExaQUte software framework

The development of the ExaQUte software framework was an important overall goal of the ExaQUte project, being integrated into the deliverable structure of the project. The reader is referred to the ExaQUte website [132], for a detailed list of software-related deliverables, each relating to a different software component of the ExaQUte software framework. A comprehensive overview of the ExaQUte software framework can be found in the report [23]. We present, in this section, a summary of the material therein.

The list of constituent libraries of the ExaQUte software framework, as well as their functionality within the overall algorithmic structure of the project, is given in Table 6.1, along with the responsible consortium partners from Table 1.1. The collective ExaQUte software framework structure is shown in Fig. 6.1, indicating a broad interlinking call-structure between the various software packages. We note here that the XMC library is a central component of the framework. We refer readers to [23] for references that offer further details on the individual software packages.

Software	**Partner**	**Functionality**
OUU scripts	EPFL, CIMNE, TUM	Control the overall gradient-based optimisation algorithm
XMC [5]	EPFL, CIMNE	Launch MLMC simulations to estimate a given statistic up to a tolerance
Kratos Multi-physics [95]	TUM, CIMNE, UPC	Time-dependent FEM PDE solver
ExaQUte API [30]	BSC, IT4I	Interface API between XMC and scheduling systems to enable agnostic selection
PyCOMPSs [125]	BSC	Job scheduling system to enable parallelism between MLMC simulations
Hyperloom, Quake [35, 29]	IT4I	Job scheduling system to enable parallelism between MLMC simulations
ParMMG [36]	Inria	Parallel adaptive mesh refinement

Table 6.1: Software of the ExaQUte project

We now wish to relate the structure of the MLMC estimator for the expected value $\mathbb{E}[Q]$ of a random QoI Q described in Eq. (2.5) to the software packages of the framework. The estimator in Eq. (2.5) uses a set of approximations $\{Q_l\}_{l=0}^{L}$ of increasing accuracy and cost. The estimator reads:

$$\mathbb{E}[Q] \approx \hat{\mu} := \frac{1}{N_0}\sum_{i=1}^{N_0} Q_0^{(i,0)} + \sum_{l=1}^{L}\frac{1}{N_l}\sum_{i=1}^{N_l}\left(Q_l^{(i,l)} - Q_{l-1}^{(i,l)}\right), \tag{6.1}$$

where $Q_l^{(i,l)}$ and $Q_{l-1}^{(i,l)}$ are correlated realisations of the QoI for the same random input on levels l and $l-1$. Using Eq. (6.1), we now describe in detail each of the component frameworks,

excluding the XMC package, which will be covered in detail in Section 6.2.

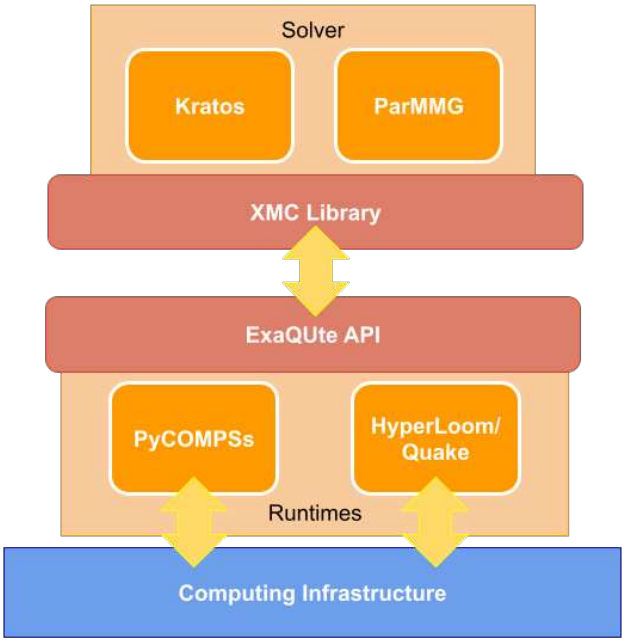


Figure 6.1: ExaQute software framework

6.1.1 Kratos Multiphysics

The Kratos Multiphysics solver [95] is a multi-physics software framework written in the C++ language for the numerical solution of engineering problems. Although the Kratos package was chiefly used for Computational Fluid Dynamics (CFD) applications within the ExaQUte project, the package is also capable of computational structural dynamics, thermal problems, fluid-structure interaction and particle methods. The package is also capable of thread-level and node-level parallelism. In the context of the ExaQUte project and the MLMC estimator presented in Eq. (6.1), the Kratos package is used to generate the realisations $Q_l^{(i,l)}$ and $Q_{l-1}^{(i,l)}$ for a given random input.

6.1.2 ParMMG

ParMMG [36] is a parallel three dimensional volume remesher, that is used extensively by the Kratos package in Section 6.1.1. The Kratos library first computes the solution field for a given mesh. It then computes mesh refinement information for the mesh based on the obtained solution field. For example, for the simulations that will be presented in Chapter 7, a metric field is computed by Kratos using Hessian information that is estimated using the solution field. This information, along with the mesh, is passed on to ParMMG, which applies the required refinements in parallel, and then repartitions the mesh between processors such that an equal workload is maintained between processors. An important implication of this

capability is that when it is tuned appropriately, it can be used to select and design meshes such that the MLMC hypotheses in Eq. (2.8) are respected.

6.1.3 PyCOMPSs and Hyperloom

We described in Chapter 2 that the hierarchical Monte Carlo estimators and algorithms presented therein possessed several common features that were exploited by the structure of the XMC library. Another highly important features of hierarchical Monte Carlo estimators is that each correlated sample pair $(Q_l^{(i,l)}, Q_{l-1}^{(i,l)})$ can be computed concurrently and independent of one-another. For scenarios where there are no data-dependencies between the two samples within the pair, even $Q_l^{(i,l)}$ and $Q_{l-1}^{(i,l)}$ can be computed independently and concurrently. Additionally, samples on different levels l can also be computed independently. Lastly, the estimation of the level-wise bias and variances, as defined in Eq. (2.8), can be computed independently between the levels. In total, hierarchical Monte Carlo estimators possess a large degree of latent parallelism that can be exploited by an appropriate software tool and parallelisation strategy. Some research exists on the optimal scheduling of MLMC estimators. Notably, the authors of [46] presented several different optimal scheduling strategies for MLMC estimators.

Within the ExaQUte project, task-based parallelism has been used to exploit this latent parallelism, through the PyCOMPSs [125] and Hyperloom/Quake [35, 29] task schedulers. The user provides the task schedulers with a list of "tasks" within their program. With respect to the hierarchical Monte Carlo approach, these can, for example, be the computation of a given realisation $Q_l^{(i,l)}$, or the computation of the level-wise estimates b_l, V_l and $\mathfrak{c}_l$ in Eq. (2.8). The task schedulers then construct a task-dependency graph, and, given a pool of available computing resources, allocate the tasks to different processors based on data-locality. Although the default strategy is one of data-locality, users can also select strategies such as first-in-first-out, last-in-first-out, etc. A key advantage of both of the task-scheduling software is that they are hardware-agnostic. This means that, other than supplying information of the pool of available resources and providing information on the list of tasks, the user does not have to provide any hardware-specific information to the code.

6.1.4 ExaQUte API

The ExaQUte Application Programming Interface (API) aims at providing a common Python API through which to interact with either of the runtime parallelism frameworks described in Section 6.1.3. Specifically, the ExaQUte API provides a set of Python decorators that are applied to functions by the user throughout their program. The decorators provide PyCOMPSs or Hyperloom with information about the inputs and outputs of the task, whether the task requires MPI or OpenMP parallelism, resource constraints and/or data-dependencies, etc. It also provides a synchronization API to allow users to synchronize remotely generated data and barriers to synchronize tasks' execution. More details about the ExaQUte API are pro-

vided in [30]. The ExaQUte API is the key link between the task-schedulers in Section 6.1.3 and the XMC package, which will be described in detail in Section 6.2.

6.2 XMC library

As a part of the ExaQUte project, EPFL was tasked with the development and maintenance of an MLMC Python engine. The Python engine, developed extensively during this thesis research, is a key ingredient in the ExaQUte software framework, as can be seen from Fig. 6.1. The XMC software library, short for X-Monte Carlo, is a software library for implementing various hierarchical Monte Carlo methods as described in Chapter 2. Along with the deliverable reports introduced in Chapter 1, it was required to release a version of the software at month 12 of the project [4], as well as at month 30 [5]. In addition to the features present in these releases, several experimental features are available as well in lesser stable versions of the code, available at the public repository information presented in [5] and [4]; for example, the parametric expectation framework of [16] and [52].

Users of the library are able to select one of many pre-programmed hierarchical Monte Carlo algorithms such as fixed or adaptive Monte Carlo/MLMC algorithms as well as CMLMC algorithms like Algorithm 2 and 3. They can also chose from pre-programmed interfaces with widely used numerical analysis packages. Current support exists for the Kratos package [95], developed within the ExaQUte consortium, as well as the Fenics library [92]. The library also offers parallelisation capabilities using the PyCOMPSs [125] and Hyperloom/Quake [35, 29] distributed computing frameworks, and is written using the ExaQUte API to interact with them [30].

The library is also programmed in a modular way that allows users to construct their own X-Monte Carlo algorithm from several building-block functions. Although current support exists for MLMC, MIMC and some MFMC estimators, the library was developed to allow future developers to extend the capabilities of XMC to wider classes of hierarchical Monte Carlo estimators as well. It also allows developers to write custom-interfaces easily for their own numerical analysis packages, such as custom-finite element analysis or computational fluid dynamics libraries.

6.2.1 Library structure and relation to modularity

As was discussed in Chapter 2, the XMC library uses a class structure that we designed based on the common properties of hierarchical Monte Carlo methods, specifically the MLMC estimator in Eq. (6.1), the Algorithms 1 and 2, as well as other hierarchical estimators and algorithms. To motivate the library structure, we first present a generic version of an adaptive hierarchical Monte Carlo algorithm in Algorithm 6. We observe that each of the algorithms 1 and 2, as well as the CMLMC algorithm presented in Chapter 3, fits into this common structure, with the specific definition of each generic step redefined according to the definition of

each algorithm.

Algorithm 6: Generic hierarchical Monte Carlo algorithm used in XMC

```
1:  Start: Fixed initial hierarchy. Inputs for stopping criteria.
2:  while Stopping criteria are not satisfied do
3:      if First iteration then
4:          Simulate screening hierarchy
5:      else
6:          Compute optimal hierarchy parameters
7:          Simulate optimal hierarchy
8:      end if
9:      Compute level/index-wise estimates
10:     Compute model fit on level/index-wise estimates if needed
11:     Compute global estimator and error estimates.
12: end while
```

During the early development stages of the library, we aimed to develop a modular framework that, although initially and primarily used for MLMC methods, could be extended to more generic hierarchical Monte Carlo methods. We decided to design the class structure of XMC to satisfy the needs of Algorithm 6. To this end, Table 6.2 shows the main classes of XMC, their related mathematical objects, and their intended function within the library. In what follows, we will relate each step of Algorithm 6 to the class structure presented in Table 6.2.

In relation to Algorithm 6, we have the following. The `XMCAlgorithm` class manages the overall execution of Algorithm 6. The tasks in lines 4 and 7 of computing a certain hierarchy, given the number of levels/indices and index-wise sample sizes, is managed by the `MonteCarloSampler` class, and its index-wise child hierarchy of classes `MonteCarloIndex`, `SampleGenerator`,`SolverWrapper` and `RandomGeneratorWrapper`. Once the samples are computed, the tasks in lines 9 of compute level-wise or index-wise estimates, such as the bias terms b_l and variance terms V_l, are handled by instances of the `StatisticalEstimator` class. Additionally, the task in line 10 of computing models for the variation of these level/index-wise quantities over the levels/indices is carried out by `ModelEstimator` class. The overall assembly of the hierarchical estimator and the computation of its corresponding error estimate in line 11, both computed using level/index-wise estimations, is carried out by the `EstimationAssembler` and `ErrorEstimator` classes respectively. The `HierarchyOptimiser` class carries out the task in line 6, which is to select the shape parameters of the novel hierarchy based on the estimates from the previously simulated hierarchy. Lastly, the execution of the stopping criterion in line 2 is managed by the `MultiCriterion`. A number of other minor classes exist within the library, satisfying more auxiliary functions. In addition to the primary classes described in the hierarchical nested structure in Table 6.2, we present in Fig. 6.2 the interdependence between the classes. At each level, the higher class contains one or more instances of the lower class.

A key feature of this library is that, due to the generic definition of Algorithm 6, and due to the modularity of the class structure matching each step of Algorithm 6, one can select from a variety of estimators and algorithms by adjusting the specific definition of each step. For example, one can choose between adaptively selecting the hierarchy parameters to attain a prescribed tolerance on the MSE, or simply doubling the current level-wise sample sizes, by changing the definition of routines within the `HierarchyOptimiser` class. Another example is of the switch between multi-index and multi-level estimators. The estimators differ by the number of discretisation parameters, which is inferred from the length of the multi-index defining a certain index. By simply editing the length of the multi-index provided to the `MonteCarloSampler` class by `XMCAlgorithm`, users can change between MIMC and MLMC estimators.

To support such modularity, we have created a set of demo configuration files that allow users to edit the specific routine definitions of each class. Users can select from these pre-made demo configuration files, or to create their own files based on them, to uniquely tailor Algorithm 6 to their needs. The configuration file mechanism goes hand-in-hand with the framework of function object instantiation in Python, which we have used to define function routines at run-time by selecting from a pre-defined list of options for each routine. This mechanism is covered in detail in Section 6.2.3.

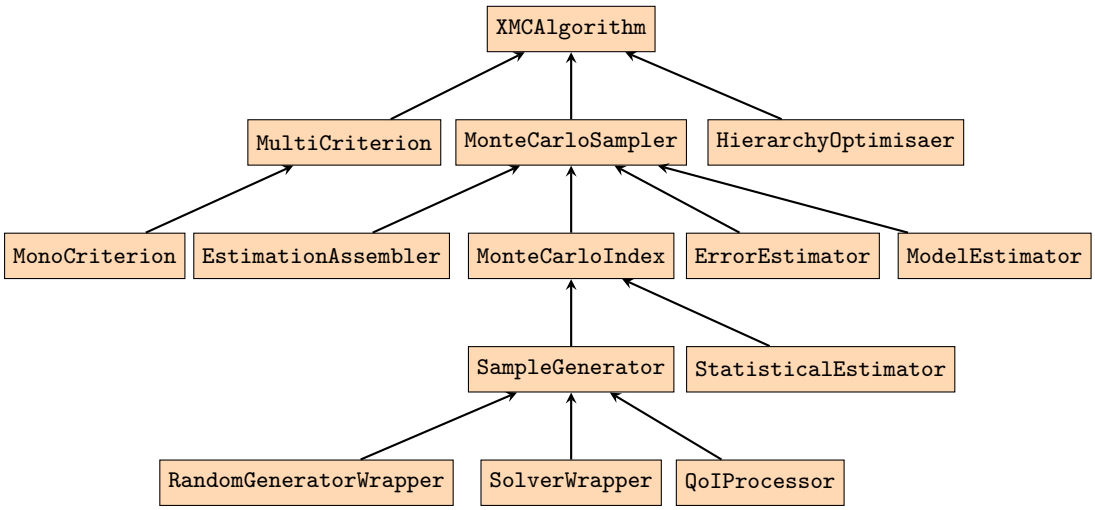


Figure 6.2: XMC class structure.

6.2.2 Parallelism using PyCOMPSs

As was discussed in Chapter 2, we designed the XMC library to reflect the hierarchical structure of various hierarchical Monte Carlo algorithms. In addition to this, we developed the hierarchical structure of XMC with the aim of exploiting any underlying potential for parallelism to the maximum extent. The discussion in Chapter 2 highlighted that there are two main degrees of parallelism that could be exploited.

The first is that since the level-wise estimators of $\mathbb{E}[Q_l - Q_{l-1}]$ are simple Monte Carlo estimators, the computation of the correlated sample pairs can all be conducted in parallel. As a result, the routines of the `SolverWrapper` class that generate each individual sample $Q_l^{(i,l)}$ is task-decorated using the ExaQUte API. The second degree of parallelism is that the computation of the level-wise estimates b_l, V_l and $\mathfrak{c}_l$ defined in Eq. (2.8) can be done independently for each level l. To this end, the routines of the `MonteCarloIndex` class that compute these estimates are also task-decorated ExaQUte API, with a synchronisation point at each level l to wait for all of the samples required to compute b_l, V_l and $\mathfrak{c}_l$ to finish simulating.

The global coordination between the levels, required for error control and adaptivity, is conducted serially. To this end, a synchronisation point is placed at the execution of the stopping criterion at each iteration of the generic XMC algorithm, since this requires the computation of an appropriate error, which depends on the computation of b_l, V_l and $\mathfrak{c}_l$, which in turn depend on the computation of the underlying QoI realisations. Once the synchronisation point is reached, the relatively inexpensive hierarchy adaptivity calculations are computed serially before launching the next batch of tasks. Such a paradigm is illustrated in Fig. 6.3 for one iteration of a generic MLMC algorithm. Tasks at the same horizontal point in Fig 6.3 are executed concurrently.

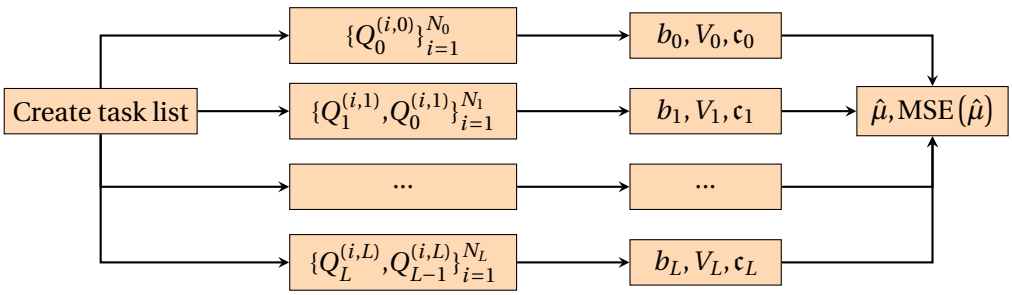


Figure 6.3: Parallelised iteration of MLMC algorithm

6.2.3 Function definition mechanism

The ExaQUte XMC library is organised as follows. The global directory `xmc` is a Python package that contains all the packages, sub-packages and modules of the library. Within this folder, each file contains the definition of one class. Every file is treated as a module that can be imported. If the class is called `ClassName`, the file is named `className.py` Each class contains multiple types of members; this includes class and function definition instances, variables and other data structures, as well as method definitions.

The key mechanism that allows for the multi-algorithmic capability of the XMC library is the instantiation of function objects; this is used to enable a function with a given name to have multiple definitions. For example, we require that the `optimalIndexSet` method of the `HierarchyOptimiser` class to be called as `optimalIndexSet()`, but to evaluate different expressions for the optimal number of levels based on the type of XMC algorithm used.

In this way the user can simply specify the type of algorithm they would like to run and the definition of `optimalIndexSet` automatically changes based on this, without any further intervention. This is achieved by instantiating `optimalIndexSet` dynamically to a specific function definition through the constructor of the `HierarchyOptimiser` class as follows:

```
# hierarchyOptimiser.py
class HierarchyOptimiser():
    def __init__(self, **keywordArgs):
        ...
        self.optimalIndexSet =
            dynamicImport(keywordArgs.get("optimalIndexSet"))
        ...
```

The `dynamicImport` method sets the definition of `optimalIndexSet` based on inputs to the constructor. In this way, every call to `optimalIndexSet` in the code is replaced by a call to the specific definition. It is hence implied that every "general" method such as `optimalIndexSet` will have a corresponding list of specific definitions, one of which is selected at runtime during the construction of the class containing the method.

To organise this, every `className.py` file has a corresponding folder `methodDefs_className`. Inside this folder, there are multiple files named `generalMethod.py`, one for each member of `ClassName` that is a function object instance of the class. Within each `generalMethod.py`, there are a list of definitions as follows:

```
# generalMethod.py
def specificDefinition1():
    ...

def specificDefinition2():
    ...
```

For the example in this section, the instantiation then occurs as follows.

```
# elsewhere.py
if (xmcAlgorithmType == "xmcAlgorithm1"):
    keywordArgs["optimalIndexSet"] = "xmc.methodDefs_hierarchyOptimiser.
      optimalIndexSet.specificDefinition1"

x = HierarchyOptimiser(**keywordArgs)
```

6.3 Examples of use

We briefly report here on simulations within the ExaQUte project that the XMC package has been used for. Firstly, the ExaQUte software framework was presented in totality in [99], applied to a case of fluid flow over a building. The case was intended to demonstrate efficient Central Processing Unit (CPU) usage. The XMC package was also used to simulate both

two dimensional and a three dimensional fluid flow cases in [23] using a simple adaptive Monte Carlo algorithm to estimate the expected value of a given random QoI. In this case the aim was to measure weak and strong scalability performance of the entire ExaQUte software framework on high performance computing hardware, while using either of the PyCOMPSs or Hyperloom schedulers. The combination showed good strong and weak scaling performance. Lastly, the XMC library was used to simulate a MFMC algorithm in [20].

Mathematical Object	Corresponding Class	Functionality
–	`XMCAlgorithm`	Main class. Handles routines corresponding to MLMC algorithm being implemented. Manages all other class instances and their interactions
$\hat{\mu}$	`MonteCarloSampler`	Handles routines corresponding to the estimator in consideration
$\frac{1}{N_l}\sum_{i=0}^{N_l}\left(Q_l^{(i,l)} - Q_{l-1}^{(i,l)}\right)$	`MonteCarloIndex`	Index/level-specific class that manages sample generation and level-wise estimations at the level/index l
$\left(Q_l^{(i,l)} - Q_{l-1}^{(i,l)}\right)$	`SampleGenerator`	Index/level-specific class that manages the correlated generation of one sample at all indices involved in a correlated difference at the level/index l, i.e., l and $l-1$. Contains 2^d `SolverWrapper` instances, where d is the number of discretisation parameters
Q_l	`SolverWrapper`	Handles the generation of one realisation for a given random input
$\omega^{(i,l)}$	`RandomGeneratorWrapper`	Generates the random inputs for use by the `SolverWrapper` instances
b_l, V_l, c_l	`StatisticalEstimator`	Handles the computation of level-wise statistics necessary for global estimations, error estimation, and hierarchy optimisation
C_α, α, C_β, β, C_γ, γ	`ModelEstimator`	Obtains the level-wise estimate data from all levels and constructs exponential/geometrical decay models on them
$\hat{\mu}$	`EstimationAssembler`	Obtains the level-wise estimate data from all levels and constructs the overall estimator
$\mathrm{MSE}\left(\hat{\mu}\right)$	`ErrorEstimator`	Obtains the level-wise estimate data from all levels and constructs error components such as the bias error, statistical error, etc. and also the MSE
$N_l(\epsilon), L(\epsilon)$	`HierarchyOptimiser`	Predicts the new hierarchy parameters for a given tolerance ϵ^2
–	`MultiCriterion`	Manages the stopping criterion/criteria of the algorithm

Table 6.2: XMC classes and relation to mathematical objects

7 Production Simulations

We recall here that the aim of the ExaQUte project was to apply MLMC methods to the problem of risk-averse shape optimisation for civil engineering applications; namely for minimising the CVaR of a random QoI, typically a structural force or moment coefficient, with respect to a set of design parameters. This required several novel mathematical, algorithmic and software-related developments. Chapters 3 and 5 presented the novel mathematical and algorithmic tools developed during the ExaQUte project and during this thesis research; namely the development of MLMC estimators for the CVaR, and the development of novel gradient-based OUU algorithms for the CVaR using the MLMC method to estimate the required sensitivities. Additionally, we presented in Chapter 6 the novel Python library XMC [5], which provides an implementation of these algorithms. Numerical results were presented in both Chapters 3 and 5 that demonstrated the efficacy of the novel algorithmic developments, as well as their implementation within XMC, for problems of practical relevance to the ExaQUte project. Specifically, in Chapter 3, we presented a problem of steady incompressible fluid flow over a cylinder placed in a channel and subject to inflow uncertainties, whereas, in Chapter 5, we tested our framework on an advection-reaction-diffusion problem used in pollutant transport applications. The above-mentioned simulations were conducted on a commercial desktop computer, and were simulated serially using only the XMC library. Although these simulations demonstrated the effectiveness of the serial implementation of these algorithms within the XMC library, it was also required of the ExaQUte project to demonstrate that the ExaQUte software framework presented in Chapter 6 could be used to solve complex multi-physics UQ and OUU problems on high-performance hardware with scalable parallelism.

In this chapter, we present the results of some of these large-scale simulations that were conducted on high-performance hardware using the ExaQUte software framework presented in Chapter 6. The production simulations were conducted with two broad aims in mind. The first was to assess whether the decay rate hypotheses in Eq. (2.8) could be fulfilled for the target application of the ExaQUte project; namely one of turbulent fluid flow over a civil engineering structure, or for more simplified versions of such a problem, thereby enabling the

use of MLMC estimators. The second, dependent on the results of the first set of feasibility studies, was to apply the CVaR estimation and CVaR sensitivity estimation frameworks developed in our works [16] and [52] to a problem of practical interest to the ExaQUte project that would satisfy the MLMC decay rate hypotheses.

The structure of this chapter reflects these two main aims, and can be divided into two broad parts. The first part, Section 7.1, summarises the material that we presented in [18] on the use of MLMC estimators for chaotic problems. We introduce the challenges related to MLMC estimators for chaotic problems in Section 7.1.1. We demonstrate these challenges on two simple oscillator problems in Sections 7.1.2, followed by a simplified problem representing wind flow over a building in Section 7.1.3. The second part, covered in Section 7.2, summarises the results from further collaborative simulations using the ExaQUte software framework that we presented in the final deliverable reports [20] and [15]. Specifically, in Section 7.2.2, we apply the parametric expectation framework of Chapter 3 to estimate the CVaR of the drag coefficient using the ExaQUte software framework. Additionally, in Section 7.2.3, we apply the MLMC procedure developed for estimating the sensitivities of the CVaR within a trust-region based optimisation algorithm to tackle a problem of shape optimisation, wherein we seek to minimise the CVaR of the negative of the lift-coefficient for a problem of potential flow over an airfoil subject to inlet uncertainties. Section 7.2.4 then presents an overall conclusion of the results presented in this chapter, as well as potential future research directions.

7.1 Feasibility of MLMC for time dependent problems

7.1.1 MLMC theory for time dependent problems

To illustrate the potential pitfalls in using the MLMC method for time-dependent problems, let us consider a system of SDEs with additive noise given by

$$dX(t) = f(t, X(t))dt + \sigma dW(t), \quad t > 0, \quad X(0) = X^0, \tag{7.1}$$

where $X(t) \in \mathbb{R}^n$, $\sigma \in \mathbb{R}^{n\times k}$, and $W(t)$ is a $\mathbb{R}^k$-valued standard Wiener process. Relevant to the ExaQUte project is the accurate computation of time averages of some output quantities $\tilde{Q}(X(t))$ with $\tilde{Q} : \mathbb{R}^n \to \mathbb{R}$ a smooth function. Namely, our goal is to compute

$$Q = \langle \tilde{Q}(X(t)) \rangle_T = \frac{1}{T}\int_0^T \tilde{Q}(X(t))dt. \tag{7.2}$$

We consider as well a discretised version of Eq. (7.2) by, for example, the Euler–Maruyama scheme.

$$X_{n+1} = X_n + hf(t_n, X_n) + \sigma\Delta W_n, \quad n = 0, 1, ..., \quad X_0 = X^0, \tag{7.3}$$

with $\Delta W_n \overset{i.i.d}{\sim} \mathcal{N}(0,h)$, where h is the step size, and a piecewise linear reconstruction of the solution

$$X_h(t) = \left(\frac{t_{n+1}-t}{h}\right) X_n + \left(\frac{t-t_n}{h}\right) X_{n+1}, \quad t \in (t_n, t_{n+1}], \quad n = 0,1,... \tag{7.4}$$

This leads to the following approximation of the time average.

$$Q_h = \langle \tilde{Q}(X_h(t)) \rangle_T = \frac{1}{T}\int_0^T \tilde{Q}(X_h(t))\,dt. \tag{7.5}$$

Under reasonable regularity assumptions on f and $\tilde{Q}$ [75], we have that:

$$\left|\mathbb{E}\left[\tilde{Q}(X_h(t))\right] - \mathbb{E}\left[\tilde{Q}(X(t))\right]\right| \leq c_1(t)h, \quad \forall t > 0, \tag{7.6}$$

$$\mathbb{E}\left[\left(\tilde{Q}(X_h(t)) - \tilde{Q}(X(t))\right)^2\right]^{1/2} \leq c_2(t)h, \quad \forall t > 0, \tag{7.7}$$

for the case of additive noise considered here. Eq. (7.6) follows from [75, Theorem 14.1.5], whereas Eq. (7.7) follows from [75], with both results requiring that f is Lipschitz continuous in the state and 1/2-Hölder continuous in time and $\tilde{Q}$ is Lipschitz continuous. The left hand side of Eq. (7.6) and Eq. (7.7) are called the weak error and strong (or pathwise) error, respectively.

[75, Theorem 4.5.4] showed that, under the additional assumption of linear growth on f, one could show that the constant $c_2(t)$ typically has the form $c_2(t) = \bar{c}_2 e^{Lt}, L \in \mathbb{R}$. For chaotic systems, the constant L is typically positive and large, meaning that the error estimate is meaningful only for a time horizon T of the order $\mathcal{O}\left(L^{-1}\right)$. On the other hand, under a dissipative condition [1] on f [96, Theorem 6.1], one has that $\mathbb{E}\left[\tilde{Q}(X(t))^2\right]^{1/2} \leq C$ for all $t > 0$, so that Eq. (7.7) can be replaced by

$$\mathbb{E}\left[\left(\tilde{Q}(X_h(t)) - \tilde{Q}(X(t))\right)^2\right]^{1/2} \leq \min\left\{\bar{c}_2 e^{Lt} h, 2C\right\}, \quad \forall t > 0. \tag{7.8}$$

Concerning the weak error, if both the SDE and its discretised form are ergodic [75, Section 17.2], the constant $c_1(t)$ is uniformly bounded in time so that Eq. (7.6) can be replaced by

$$\left|\mathbb{E}\left[\tilde{Q}(X_h(t))\right] - \mathbb{E}\left[\tilde{Q}(X(t))\right]\right| \leq \bar{c}_1 h, \quad \forall t > 0. \tag{7.9}$$

The estimates in Eqs. (7.8) and (7.9) can be used to estimate the decay of the bias and variance

[1] $X^\top f(t,X) \leq -k_1 |X|^2 + k_2$ for $|X| \geq R$ for suitable non-negative constants k_1, k_2 and R.

estimates b_l and V_l defined in Eqs. (2.8) for the time averaged QoI Q_{h_l}. We have that

$$b_l = \left|\mathbb{E}\left[Q_{h_l} - Q\right]\right| = \left|\mathbb{E}\left[\langle\tilde{Q}(X_{h_l}(t))\rangle_T - \langle\tilde{Q}(X(t))\rangle\right]\right| \leq \bar{c}_1 h_l = \hat{c}_1 s^{-l}, \tag{7.10}$$

$$V_l = \mathbb{V}\mathrm{ar}\left(Q_{h_l} - Q_{h_{l-1}}\right) \leq \mathbb{E}\left[\left(Q_{h_l} - Q_{h_{l-1}}\right)^2\right] \tag{7.11}$$

$$\leq 2\mathbb{E}\left[\left(Q_{h_l} - Q\right)^2\right] + 2\mathbb{E}\left[\left(Q_{h_{l-1}} - Q\right)^2\right] \tag{7.12}$$

$$\leq 4\min\left\{4C^2, \frac{\bar{c}_2^2 e^{2LT}}{2L} h_{l-1}^2\right\} \tag{7.13}$$

$$\leq \min\{\hat{c}_2, \hat{c}_3 e^{2LT} s^{-2l}\}, \tag{7.14}$$

for suitable time independent constants $\hat{c}_1, \hat{c}_2, \hat{c}_3$. We see from these estimates that the bias term always features an exponential decay with respect to the level l, whereas to observe a decay of the variance in the chaotic case, we have to either consider very large l or $T = \mathcal{O}(L^{-1})$ so that the second term in the minimum of Eq. (7.14) is smaller than the first one. In other words, variance decay can be expected only for short time intervals in which the two approximated outputs $\tilde{Q}(X_{h_l}(t))$ and $\tilde{Q}(X_{h_{l-1}}(t))$ remain correlated. For long time horizons, the two time series decorrelate completely and the variance term V_l, while remaining bounded, does not feature any decay with respect to l.

7.1.2 Oscillator problems

Van der Pol Oscillator

The Van der Pol Oscillator is an oscillator whose trajectory $x(t)$ is governed by the second-order differential equation

$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + x = 0. \tag{7.15}$$

The oscillator has the favourable property that it has a limit cycle to which it converges independent of the initial coordinates in the phase space. It is also a good proxy model for vortex shedding fluid flows. To assess the effectiveness of MLMC methods in this case, we consider a stochastic version of Eq. (7.15) where the system is forced by white noise (derivative of a Wiener process):

$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + x = \tau\dot{W}(t), \tag{7.16}$$

where $W(t)$ is a standard Wiener process and τ is the strength parameter for the forcing term. We rewrite this equation as a system of SDEs:

$$dx = y\,dt, \quad t \in (0,T] \tag{7.17}$$

$$dy = \left(\mu(1-x^2)y - x\right)dt + \tau\,dW, \quad t \in (0,T] \tag{7.18}$$

$$x(0) = x_0, \quad y(0) = y_0. \tag{7.19}$$

We discretize the system using the Euler-Maruyama scheme, which reads as follows:

$$\begin{bmatrix} y_{n+1} \\ x_{n+1} \end{bmatrix} = \begin{bmatrix} y_n \\ x_n \end{bmatrix} + h \begin{bmatrix} \mu(1-x_n^2)y_n - x_n + \frac{\tau}{\sqrt{h}}\xi_n \\ y_n \end{bmatrix}, \tag{7.20}$$

where $[y_n, x_n]^T$ denote the approximations to $[y(t_n), x(t_n)]$ at the time steps $t_n = nT/N =: nh$, ξ_n are independent identically distributed realizations of a standard normal random variable, and we set $\mu = 1$, $x_0 = 1$ and $y_0 = 1$. The strength parameter is chosen as $\tau = 1.0$ and the time horizon is chosen to be $T = 100$. The solution $x(t)$ is plotted versus t for the same realization of the white noise solved on both finest and coarsest meshes in Fig. 7.1. Pathwise correlation can clearly be observed in the plot.

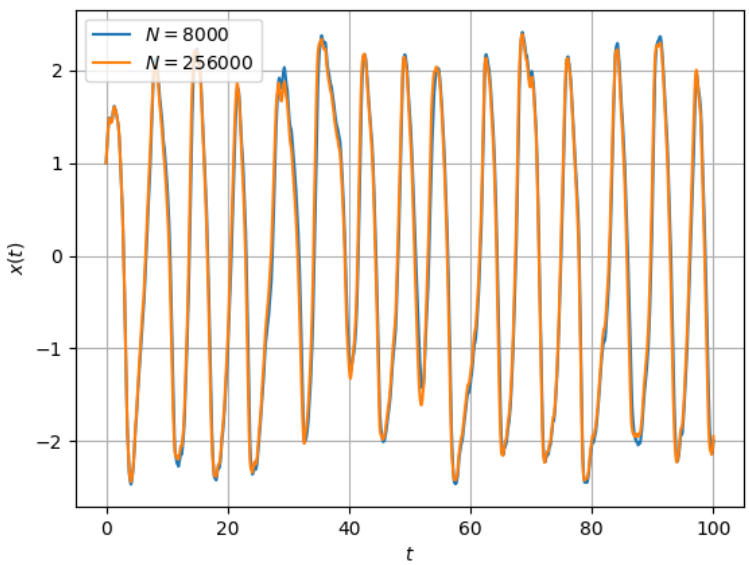


Figure 7.1: One realization of the stochastic Van der Pol oscillator solution on the finest and coarsest meshes

We then simulate the system for 10 independent realizations of the Brownian path. We plot the values $|\langle x\rangle_{h,T}^{(i)} - \langle x\rangle_{ref,T}^{(i)}|$ vs. h for $i = \{1,\ldots,10\}$, where for each realization of the Brownian path, the finest mesh is taken to be the reference solution. Each color corresponds to a different underlying Brownian path realization. The resultant plot is shown in Fig. 7.2. We observe

convergence rates of between 1.0 to 1.3 in the step size h based on least squares fits, which is consistent with the predicted strong convergence rate of the Euler-Maruyama scheme.

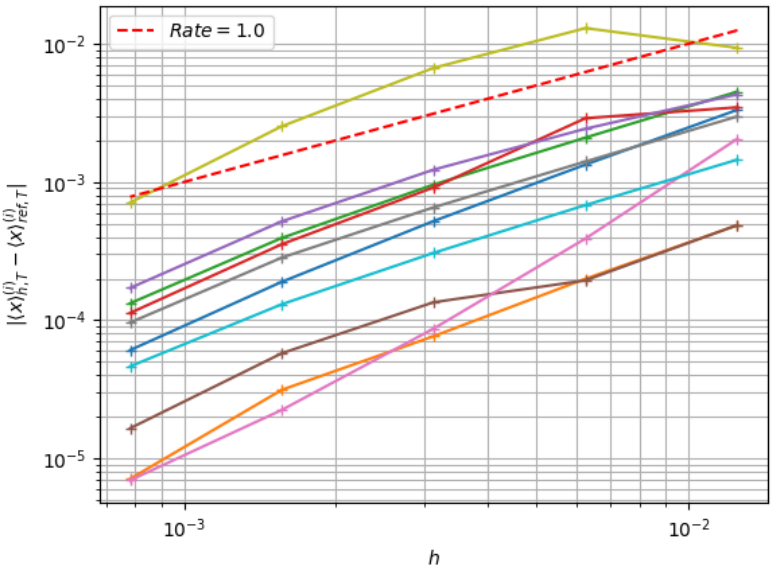


Figure 7.2: Pathwise mesh convergence in the stochastic case for the Van der Pol oscillator. Different colours correspond to different realisations.

We now wish to study the applicability of MLMC algorithms to this problem. For MLMC to produce the optimal complexity for a given tolerance, the underlying problem should satisfy the rate hypotheses of Eqs. (2.8). For the purposes of this study, we consider meshes with 16000, 32000, 64000 and 128000 points in time, indexed as $l = \{0, ..., 3\}$. We denote the QoI computed on mesh l as $\langle x\rangle_{l,T}$ and the corresponding step size as h_l. We study the convergence of the quantities

$$b_l := |\mathbb{E}\left[\langle x\rangle_{l,T} - \langle x\rangle_{ref,T}\right]|, \tag{7.21a}$$

$$V_l := \mathbb{V}\mathrm{ar}\left(\langle x\rangle_{l,T} - \langle x\rangle_{l-1,T}\right), \tag{7.21b}$$

with respect to h_l. For each l, we estimate b_l and V_l using sample average and sample variance estimators respectively, using 100 independent Brownian path realizations. For each Brownian path, the problem is solved on both the fine and coarse levels. The variation of b_l and V_l with levels l is shown in Fig. 7.3. We observe rates of approximately 1 and 2 in the step size h_l for the bias terms b_l and variance terms V_l respectively.

We observe that the stochastic Van der Pol oscillator possesses favourable properties in terms of retaining pathwise correlations. In addition, we have also shown that the Van der Pol oscillator satisfies the MLMC hypotheses of Eqs. (2.8) and, hence, can obtain the demonstrated complexity behaviour with an optimally selected hierarchy. To demonstrate this optimal

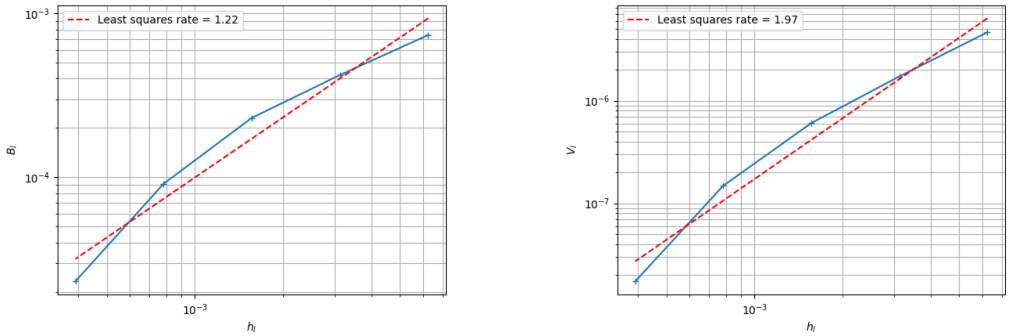


Figure 7.3: Bias (left) and variance (right) convergence for the stochastic Van der Pol oscillator.

complexity behaviour, we estimate the expectation $\mathbb{E}[\langle x \rangle_T]$ using an optimally tuned MLMC estimator $\hat{\mu}$. We prescribe a tolerance ϵ on the total error of the MLMC estimator defined as

$$\mathrm{TE}(\hat{\mu}) := |\mathbb{E}\left[\langle x \rangle_{L,T} - \langle x \rangle_{L-1,T}\right]| + C_\alpha \sqrt{\sum_{l=0}^{L} \frac{\mathbb{V}\mathrm{ar}\left(\langle x \rangle_{l,T} - \langle x \rangle_{l-1,T}\right)}{N_l}}, \tag{7.22}$$

where C_α corresponds to the inverse of the CDF of the standard normal distribution evaluated at a significance of $1 - \alpha/2$.

We use the CMLMC algorithm [39, 105] to tune the hierarchy optimally for a given tolerance on the total error. The cost of computing the optimally tuned hierarchy is then measured and plotted against the corresponding tolerance. For each tolerance tested, the entire MLMC simulation is repeated 15 times and the corresponding simulation time is noted for the optimally tuned hierarchy. The results are shown in Fig. 7.4. In addition, the estimated cost for a Monte Carlo simulation to reach the same tolerance is also shown. It can be seen that the cost grows as ϵ^{-2}, thus demonstrating optimal complexity behaviour for an MLMC estimator. It also demonstrates that MLMC estimators can be used with success for problems displaying periodic or oscillatory behaviour.

Lorenz Oscillator

We now wish to study the challenges faced in applying the MLMC framework to chaotic problems. To this end, we study the Lorenz oscillator. The Lorenz oscillator is a three dimensional

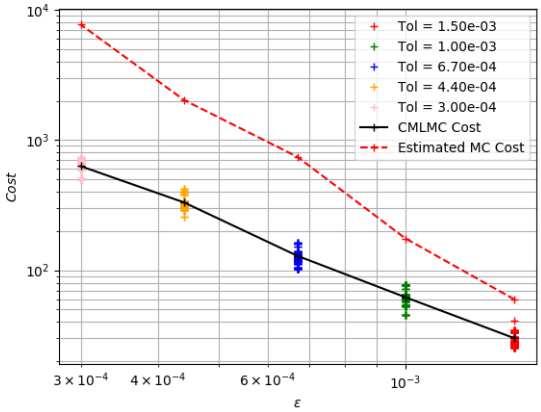


Figure 7.4: Complexity behaviour for CMLMC algorithm for the Van der Pol oscillator.

chaotic oscillator governed by the following system of ODEs:

$$\frac{dx}{dt} = \sigma(y - x), \tag{7.23a}$$

$$\frac{dy}{dt} = x(\rho - z) - y, \tag{7.23b}$$

$$\frac{dz}{dt} = xy - \beta z. \tag{7.23c}$$

For the purposes of this study, we select the parameter values to be $\sigma = 10$, $\rho = 28$ and $\beta = 8/3$. The Lorenz oscillator has the property that, for these parameter values, it is chaotic. This means that two trajectories with initial conditions differing by an arbitrarily small perturbation will eventually diverge. This poses a challenge for the MLMC method since pathwise convergence is important for the hypotheses in Eqs. (2.8) to be satisfied.

As for the Van der Pol oscillator, we study a stochastic version of the Lorenz Oscillator where the right hand sides of all three of the Eqs. (7.23) are forced with independent white noise terms as follows:

$$dx = \sigma(y - x)dt + \tau dW_1, \tag{7.24a}$$

$$dy = \big(x(\rho - z) - y\big)\,dt + \tau dW_2, \tag{7.24b}$$

$$dz = \big(xy - \beta z\big)\,dt + \tau dW_3, \tag{7.24c}$$

for $t \in (0, T]$, where $T = 400$ denotes the time horizon, W_1, W_2 and W_3 are independent Wiener processes and τ is the strength parameter whose value is chosen to be 1.0 for the purposes of this study.

We discretize Eqs. (7.24) using the Euler-Maruyama scheme. The discretized system reads

$$\begin{bmatrix} x_{n+1} \\ y_{n+1} \\ z_{n+1} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} + h \begin{bmatrix} \sigma(y_n - x_n) + \frac{\tau}{\sqrt{h}}\xi_{1,n} \\ x_n(\rho - z_n) - y_n + \frac{\tau}{\sqrt{h}}\xi_{2,n} \\ x_n y_n - \beta z_n + \frac{\tau}{\sqrt{h}}\xi_{3,n} \end{bmatrix}, \tag{7.25}$$

where $[x_n, y_n, z_n]^T$ denote approximations to $[x(t_n), y(t_n), z(t_n)]^T$ on the time grid $t_n = nT/N =: nh$, and $\xi_{1,n}, \xi_{2,n}$ and $\xi_{3,n}$ are independent standard normally distributed random variables. We wish to study the behaviour of the system for different step sizes; namely corresponding to $N = \{4, 8, 16, 32, 64\} \times 10^4$, and index the levels using $l \in \{0, ..., 4\}$ accordingly. Fig 7.5 shows one realization of the solution of the forced Lorenz oscillator computed on the finest and coarsest meshes with the same underlying white-noise realizations. We observe that the solutions very quickly decorrelate.

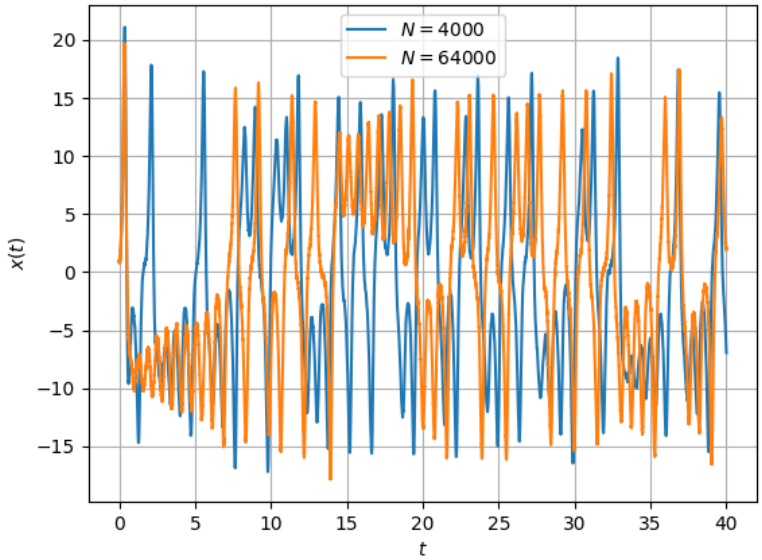


Figure 7.5: A realization of the stochastic Lorenz oscillator solution computed on the finest and coarsest meshes

We then simulate the system for 10 independent realizations of the Brownian paths. We plot the values $|\langle x\rangle^{(i)}_{h,T} - \langle x\rangle^{(i)}_{ref,T}|$ versus h for $i = \{1, ..., 10\}$, where, for each realization of the Brownian path, the finest mesh is taken to be the reference solution. The resultant plot is shown in Fig. 7.6 where different colors indicate the different realizations. It is evident from Fig. 7.6 that pathwise convergence cannot be expected for the Lorenz oscillator for the given parameters.

We now conduct a screening MLMC similar to the Van der Pol oscillator with 10^4 samples per level, to study the difference of the time averages. We wish study the decay of the level-wise biases b_l and variances V_l defined in Eqs. (7.21) for the QoI $\langle x\rangle_{h_l,T}$ for different levels l as

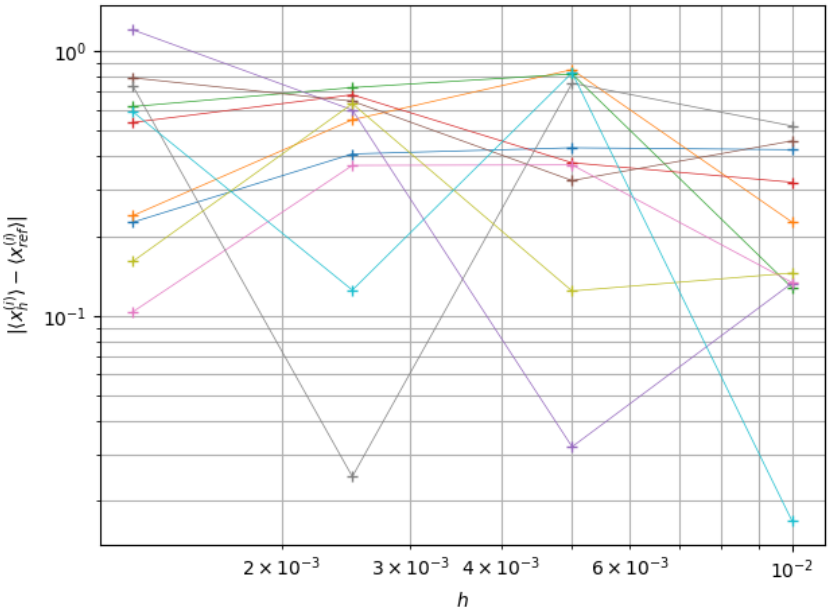


Figure 7.6: Convergence of time averages for the stochastic Lorenz oscillator. Different colours correspond to different realisations.

defined earlier. It is expected that the bias decays for a large enough time window T, but that variance decay cannot be guaranteed. The resultant behaviour is reported in Fig. 7.7. It can be observed that although the bias decays with rate better than 1 in the time step size h, the variance does not decay in a meaningful way. The framework of MLMC estimation cannot be applied to the Lorenz oscillator problem in this circumstance, since the MLMC hypotheses in Eq. (2.8) may not be fulfilled for chaotic problems unless very small time step sizes or time windows are considered for the analysis. We mention the work [48], who proposed a modification of the MLMC procedure and tested the same on the the Lorenz oscillator. A spring-like coupling between the fine and coarse levels was proposed and demonstrated to retain pathwise-correlation, and it was shown that Eq. (2.8) could be satisfied for significantly larger mesh sizes than compared to a naive MLMC approach.

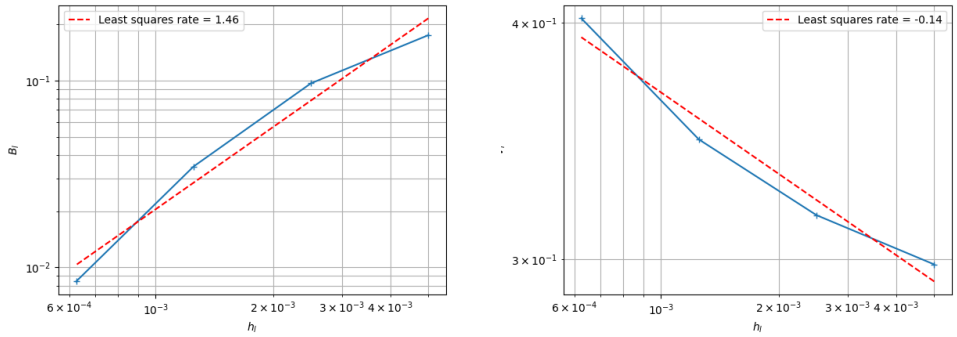


Figure 7.7: Bias (left) and variance (right) convergence for the stochastic Lorenz oscillator

7.1.3 Turbulent flow over a rectangle

The eventual goal of the ExaQUte project is to simulate fully turbulent three dimensional flow over a civil engineering structure such as a building. As an intermediate two dimensional step, a reduced problem is considered; namely, that of turbulent flow over a rectangle. A scheme of the problem is shown in Fig. 7.8 and Fig. 7.9.

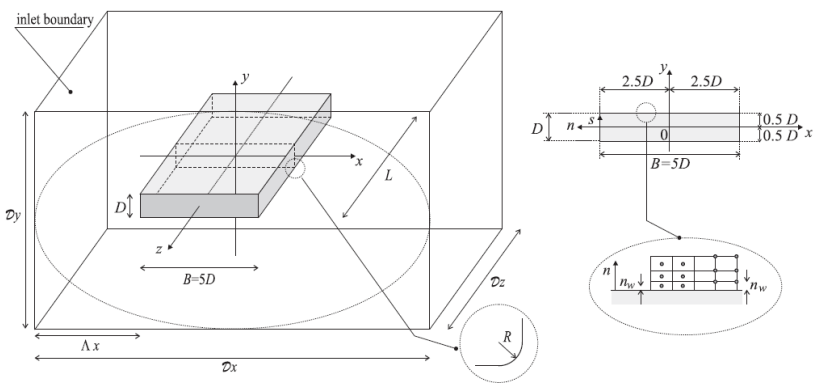


Figure 7.8: Problem description [34], $D = 1$

The incompressible Navier-Stokes Eqs. (7.26) and (7.27) are used to model the fluid flow:

$$\frac{\partial u}{\partial t} - \nu\Delta u + (u\cdot\nabla)u + \nabla p = 0, \tag{7.26}$$

$$\nabla\cdot u = 0. \tag{7.27}$$

Uncertainty is present in the inlet conditions. Specifically, a Dirichlet condition is applied at the inlet boundary, where the velocity is prescribed, with some randomness, in the horizontal direction and constant along the edge, whereas the pressure is prescribed to a constant value of zero. The inlet velocity is taken to be distributed as $v_{inlet} \sim \mathcal{N}(2.0, 0.02^2)$. This leads to a flow-through time of approximately 140 seconds based on the length of the domain. The viscosity value is adjusted to achieve the required Reynolds' number of $Re \approx 1.3\times10^5$. On the

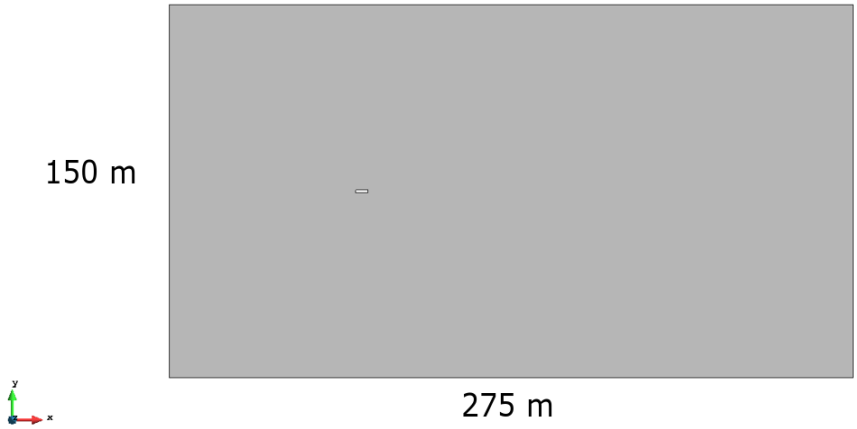


Figure 7.9: Rectangle problem dimensions. Inner rectangle 5 m × 1 m.

upper and lower boundaries, the outward normal component of velocity is set to zero. On the outlet, a zero stress condition is enforced. No-slip boundary conditions are enforced on the surface of the rectangle.

The problem is discretised using linear triangular elements for both pressure and velocity fields. Algebraic subgrid-scale stabilization is used to stabilize the problem [37, 38]. A second order fractional step method is used for time stepping that treats both pressure and velocity implicitly. For further details on the case set-up and numerical scheme used, the reader is referred to [19, 18] and the literature cited therein. A hierarchy of meshes was constructed using the adaptive procedure described in [19] for the mean conditions. The parameters of the resultant meshes are shown in Table 7.1, where h_{min} denotes the minimum mesh size and h denotes the time step size.

Interpolation Error	h_{min}	Nodes [×1000]	CFL	h
10^1	0.035	1.1	80	0.7
10^0	0.012	2	80	0.24
10^{-1}	0.0033	5	80	0.066
10^{-2}	0.0011	15	80	0.022
10^{-3}	0.00037	92	80	0.0075

Table 7.1: Mesh parameters for high Reynolds' number study

We are interested in studying the pathwise convergence of the time average of the drag force $\langle F_{D,l}\rangle$ over the interval $[140, 300]$, where l indexes the meshes in Table 7.1. The drag force is computed using a boundary integral formulation. Specifically, for each realisation of the random inlet velocity, indexed by $i \in \{1, ..., 50\}$, it is of interest to study the convergence of $\left|\langle F^{(i)}_{D,l}\rangle - \langle F^{(i)}_{D,l-1}\rangle\right|$, where $F^{(i)}_{D,l}$ denotes the drag force computed for the ith inlet velocity realisation on level l.

Fig. 7.10 shows the variation of $|\langle F_{D,l}^{(i)}\rangle - \langle F_{D,l-1}^{(i)}\rangle|$ versus the interpolation error for each of the different realizations, as well as $\mathbb{E}[\left|\langle F_{D,l}\rangle - \langle F_{D,l-1}\rangle\right|]$ estimated using a sample average over the 50 realizations. It can be seen from the plot that pathwise convergence of this quantity in the mesh is not observed. The variation of the variance of the differences $\mathbb{V}ar[\langle F_{D_l}\rangle - \langle F_{D_{l-1}}\rangle]$ with the interpolation error is also plotted in Fig. 7.11, where it is evident that this quantity does not decay with the mesh parameter either.

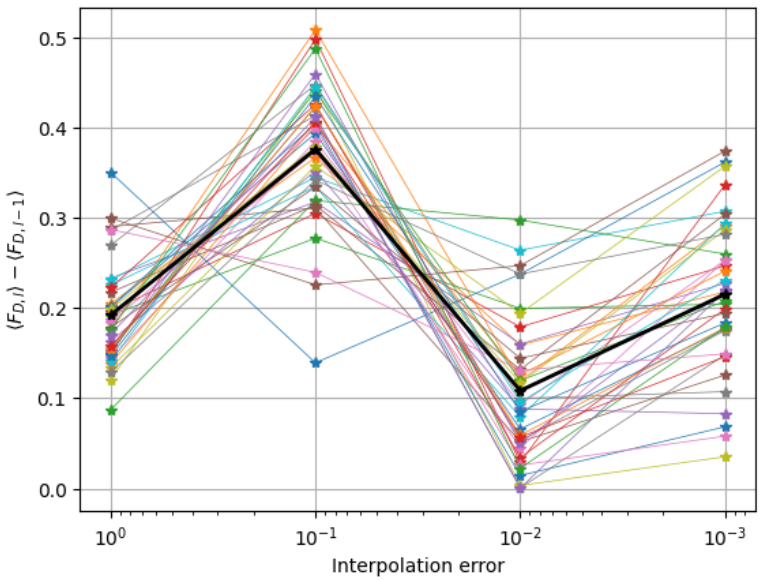


Figure 7.10: Pathwise convergence test for the flow problem with high Reynolds number. Same color for same realisation at different levels. Black line denotes sample average over the realisations. Levels defined by Table 7.1.

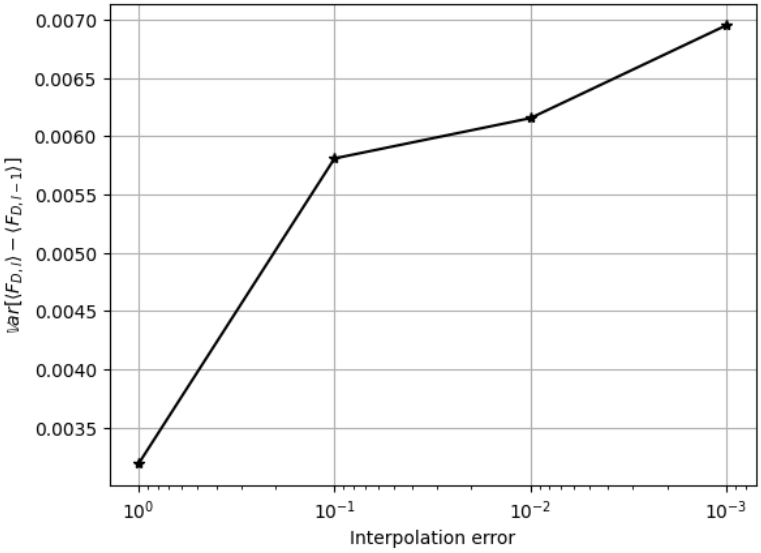


Figure 7.11: Variance decay plot for the flow problem with high Reynolds number. Levels defined by Table 7.1

It can be concluded that neither pathwise convergence, nor variance decay, may be possible to obtain at high Reynolds' numbers due to the chaotic nature of the flow. The turbulence indeed poses a significant challenge to retaining the pathwise correlation of both fine and coarse samples necessary for optimal MLMC performance. As a result, the MLMC hypotheses of Eqs. (2.8) are not likely to be fulfilled for this problem configuration. Although level-wise bias and variance decay were not obtained for the problem configuration, meaningful correlations were still observed between the various levels. The resultant correlations are shown in Table 7.2. We proposed in [18] the use of MFMC estimators to exploit these correlations, demonstrating that MFMC type estimators could provide a significant cost reduction compared to a naive Monte Carlo estimator at the finest considered level. The interested reader is referred to [18, 20] for further information.

Correlations	1	2	3	4
1	1			
2	0.586	1		
3	0.516	0.322	1	
4	0.652	0.490	0.357	1

Table 7.2: Correlation data for rectangle flow problem for levels indexed as in Table 7.1

We highlight at this point that, in addition to the numerical experiments in Sections 7.1.1 and 7.1.2, the author of this thesis contributed to the design of the simulation, as well as the analysis of the results and data used to produce Figs. 7.10 and 7.11. Although the simulations themselves were produced by our consortium partners at CIMNE using the Kratos software

package, it was decided to include the results in the content of this thesis to highlight the important negative result that MLMC methods were likely to perform sub-optimally for turbulent fluid flow problems.

7.2 CVaR estimation and minimisation using MLMC

As described earlier in this chapter, the aim of the ExaQUte project was to carry out risk-avere design of civil engineering structures. An example of the target application can be found in [76] and [20], wherein the fully three dimensional wind flow over a tall building was simulated. The shape parameters of the building were selected to minimise the CVaR of a force coefficient subject to random wind conditions. Simpler lower-dimensional surrogate problems were studied as intermediate steps towards achieving this goal in [19] and [18].

We present here the numerical results from [20] and [15]. The MLMC framework for the CVaR and its sensitivities with respect to design parameters, developed in Chapters 3 and 5, is applied to a practical problem; namely that of steady potential flow around an airfoil with uncertainties in the wind conditions [20, 100]. Such a problem was selected to demonstrate that the mathematical frameworks developed within this thesis, and within the ExaQUte project, could perform successfully for a problem that satisfied the MLMC decay rate hypotheses, but also possessed characteristics similar to the target application. We seek to estimate and minimise the CVaR corresponding to a force coefficient of the system. In Section 7.2.1, we describe the problem domain, governing equations, and boundary conditions, and briefly introduce the numerical method used to solve the problem. Section 7.2.2 presents results on the estimation of the CVaR of a force coefficient of the system using the parametric expectation framework developed in Chapter 3. In addition, Section 7.2.3 describes the results obtained for minimising the CVaR of the same force coefficient over the space of the airfoil design parameters, although using a constrained optimisation formulation and a trust-region algorithm that uses our MLMC procedure developed in Chapter 5 to estimate sensitivities of the CVaR. Section 7.2.4 then presents the overall conclusions obtained from these simulations, and proposes potential future directions of research.

7.2.1 Airfoil problem formulation

We study a problem of steady potential flow around a NACA0012 airfoil placed in a circular domain. The flow is governed by a simplified version of the Navier-Stokes equations; namely, the potential flow equation. The potential flow equation is typically used to model flows around streamlined bodies at high Reynolds' numbers and small angles of attack. For example, it is often used to model commercial airliners at cruise speed and altitude. The potential flow model allows us to reduce the Navier-Stokes system of PDEs to a single nonlinear PDE, thereby greatly reducing computational cost while preserving modelling accuracy. In this case, the steady compressible Navier-Stokes equations simplify to the steady full-potential

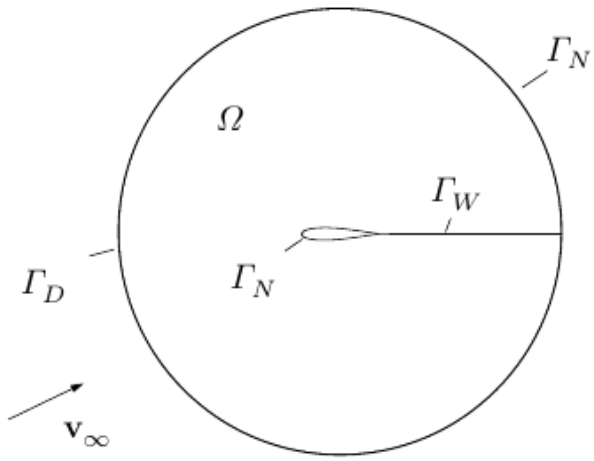


Figure 7.12: Simplified representation of the domain Ω and its boundary Γ. The wake Γ_W is modelled as a straight line. Extracted from [100].

equation:

$$\nabla \cdot \left(\rho \nabla u\right) = 0, \tag{7.28}$$

where u denotes the potential, ρ denotes the density of the fluid, and the velocity field v is given by $v = \nabla u$. The density ρ can be written in the following form using the isentropic flow hypothesis:

$$\frac{\rho}{\rho_\infty} = \left(1 + \frac{\gamma - 1}{2} \frac{u_\infty^2}{a_\infty^2} \left(1 - \frac{\nabla u \cdot \nabla u}{u_\infty^2}\right)\right)^{\frac{1}{\gamma - 1}}, \tag{7.29}$$

where u_∞, ρ_∞ and a_∞ denote the freestream velocity, density and speed of sound respectively, and γ denotes the ratio of specific heats. Hence, Eq. (7.28) is a nonlinear PDE for the potential u, requiring appropriate treatment.

The problem domain is denoted by Ω, and is shown in Fig. 7.12; its boundary is denoted by Γ. Γ is partitioned into two parts Γ_D and Γ_N. Let $\boldsymbol{n}$ denote the outward normal on the boundary and v_∞ denote the freestream velocity. Γ_D then denotes the part of Γ where $v_\infty \cdot \boldsymbol{n} < 0$ and Γ_N denotes the part of Γ where $v_\infty \cdot \boldsymbol{n} \geq 0$. The boundary conditions on u read as follows:

$$u(x) = v_\infty \cdot x + u_\infty, \quad \text{on } \Gamma_D, \tag{7.30}$$

$$\boldsymbol{n} \cdot \left(\rho \nabla u\right) = g, \quad \text{on } \Gamma_N, \tag{7.31}$$

$$\boldsymbol{n} \cdot \left(\rho^+ \nabla u^+ - \rho^- \nabla u^-\right) = 0, \quad \text{on } \Gamma_W, \tag{7.32}$$

$$\left|\nabla u^+\right|^2 - |\nabla u^-|^2 = 0, \quad \text{on } \Gamma_W, \tag{7.33}$$

where the + and − denote the upper and lower parts of the wake boundary Γ_W. Eq. (7.30)

prescribes a velocity on the inlet part of the domain based on a given Mach number M_∞ and angle of attack α_∞ as $v_\infty = a_\infty M_\infty(\cos\alpha_\infty, \sin\alpha_\infty)$. Eq. (7.31) imposes a Neumann boundary condition on Γ_N; namely by prescribing an appropriate mass flux on the "outlet" part of the domain boundary, as well as a zero flux condition on the airfoil surface. Since the naive finite element discretisation of Eq. (7.28) can lead to a non-lifting problem, the wake conditions Eqs. (7.32) and (7.33) need to be enforced in order for circulation to be generated around the boundary. Eq. (7.32) corresponds to mass conservation and Eq. (7.33) corresponds to the equality of pressures across the wake Γ_W. The wake Γ_W is modelled in this problem as a straight line. In both the UQ and OUU studies that will be presented in Sections 7.2.2 and 7.2.3, we study, as a QoI, the negative of the lift coefficient C_l, computed as was proposed in [98]:

$$C_l = \frac{u^+ - u^-}{\frac{1}{2}|v_\infty|c}, \tag{7.34}$$

where $|v_\infty|$ denotes the freestream speed and c denotes the chord length of the airfoil.

The problem is discretised using a linear triangular finite element discretisation. The numerical method used to solve the problem was presented in [43] which features the solution of the problem on body-fitted meshes with the wake boundary treated in an embedded manner. In [100], the solution was extended to a fully embedded approach. The meshes are generated differently, depending on whether they were used for the UQ problem or the OUU problem. As a result, the details of the meshing procedure will be presented in Sections 7.2.2 and 7.2.3.

7.2.2 CVaR estimation for the potential flow

We wish to study the behaviour of the lift coefficient C_l defined in Eq. (7.34) subject to uncertainties in the inlet boundary Γ_D. We model the freestream Mach number and angle of attack as random variables distributed as $M_\infty \sim \mathcal{N}(0.3, 0.1^2)$ and $\alpha_\infty \sim \mathcal{N}(5.0, 0.02^2)$ degrees. In Section 7.2.3, we minimise the CVaR of the negative of C_l, since this is equivalent to shifting the distribution of the lift coefficient towards higher values. To this end, we are interested in this section in estimating the 70%-CVaR of $-C_l$.

In this study, we consider two different approaches to design the MLMC hierarchy; namely a fixed hierarchy of meshes and an adaptive refinement approach. The fixed hierarchy consists of a set of classical body-fitted meshes. Some details of the meshes used in the analysis are shown in Table 7.3. The three first levels of the fixed hierarchy are shown in Fig. 7.13.

The adaptive refinement approach, on the other hand, uses a series of adaptive mesh refinement steps starting from a pre-generated level zero mesh, where the tolerance prescribed to each iteration of the refinement process forms a geometrically decreasing sequence. The MLMC hierarchy is defined by this same set of tolerances, which varies with the level l as $\epsilon_0 2^{-l}$, with $\epsilon_0 = 0.2$. When a set of correlated sample pairs is desired at the levels l and $l-1$, the adaptive mesh refinement process is started from the the solution on the fixed initial

Level	Nodes	Elements
0	7238	14230
1	18931	37054
2	35016	68364
3	51129	99672
4	68471	133430

Table 7.3: Information of the fixed hierarchy of meshes.

mesh, and terminated at level l. The solutions corresponding to the adaptive refinement steps at levels l and $l-1$ are then used to compute realisations of the QoI at these two levels. In this way, meshes can be generated on-the-fly for each sample pair. The mesh refinement process is based on the procedure proposed in [41] and implemented in [36]. The method in [41] constructs a metric based on an approximate Hessian of the solution, as described in [50]. This Hessian-based metric was previously described in [19, Section 2.2]; further details are available therein, such as the procedure to approximate the Hessian.

An important distinction exists between the fixed and adaptive approaches. Namely, the fixed meshes were constructed purely using airfoil geometry information. As a result, the hierarchy of meshes in Table 7.3 preserved the geometry shape of the airfoil. In contrast, the adaptive approach functions as follows. The Kratos software package is used to compute the solution and corresponding metric information for a given mesh and random realisation. It then passes this information to the MMG remesher for adaptive refinement, which treats the mesh representation of the geometry as the "true" geometry, as opposed to incorporating geometry information during the refinement process. For coarse initial meshes, this can mean that the solver interprets the airfoil geometry as its polygonal approximation on the coarse mesh and keeps the same polygonal shape through the adaptivity procedure, thus leading to sharp corners and multiple singularities in the corresponding pressure field, which, in turn, leads to a subsequent loss of convergence of the lift coefficient in the mesh parameters. To avoid this issue, the initial fixed mesh used for the adaptive refinement process is chosen to be relatively fine, with 7×10^4 nodes, to ensure that the airfoil shape is already adequately captured using the initial mesh.

We recall that we are interested in accurately estimating the CVaR $c_{0.7}(-C_l)$. We use the parametric expectation framework described in Chapter 3 for estimating the CVaR and adaptively calibrate the hierarchy based on the error estimation and adaptive hierarchy selection procedure described therein such that a prescribed tolerance is obtained on the CVaR. The simulations are conducted using the ExaQUte software framework described in Chapter 6. Specifically, the management of the CMLMC algorithm is conducted by the XMC package. The XMC package is interfaced with the Kratos multi-physics engine, which solves Eq. (7.28) for a given realisation of the input noise. The Kratos multi-physics engine interfaces, in turn, with the MMG adaptive re-mesher to conduct adaptive mesh refinement for the adaptive mesh

refinement case. Lastly, the PyCOMPSs scheduler is used to execute the large number of PDE solves required by the MLMC estimator in parallel. The ExaQUte software framework was implemented on the MareNostrum4 cluster [93], a supercomputer managed by the Barcelona Supercomputing Centre, a partner in the ExaQUte consortium.

The results can be summarised as follows. To demonstrate the physics of the problem, we conduct a set of 256 simulations on adaptively refined meshes, adapting each realisation to a tolerance corresponding to the finest level obtained during the MLMC simulations which will be presented later. We plot a histogram of the negative of the lift coefficient in Fig. 7.23. Additionally, we plot the mean, the 70%-VaR, the 70%-CVaR and a scaled version of the empirical CDF. As will be demonstrated in Section 7.2.3, minimising the right tail of this distribution and moving it left-ward corresponds to minimising the occurrences of small lift values.

For each of the above simulations, we compute the pressure coefficient C_p defined as follows:

$$C_p := \frac{p - p_\infty}{\frac{1}{2}\rho_\infty |v_\infty|^2}, \tag{7.35}$$

where p denotes the pressure field and p_∞ denotes the freestream pressure. The pressure field p can be computed from the density ρ using the isentropic relation:

$$\frac{p}{p_\infty} = \left(\frac{\rho}{\rho_\infty}\right)^\gamma, \tag{7.36}$$

where γ denotes the ratio of specific heats. Fig. 7.16 shows the physical distribution of several estimated statistics of the pressure coefficient field over the airfoil, including the mean, the VaR and the CVaR, for a significance value of $\tau = 0.7$. The mean and variance are estimated using their corresponding sample average estimators, whereas the VaR and the CVaR are estimated by constructing suitable parametric expectations at each mesh point using the procedure in Chapter 3. Other statistics introduced by [127], namely the worse-case scenario, denoted by $\mathrm{Sup}[C_p]$, and the safety-margin $\mathbb{E}\left[C_p\right] + \lambda_{0.7}\sqrt{\mathbb{V}ar[C_p]}$, where $\lambda_{0.7}$ denotes the inverse of the standard normal CDF at a significance of $\tau = 0.7$, are also plotted. The mean value is extended with a filling area, covering the mean values plus and minus three times the pointwise standard deviations. As can be seen from Fig. 7.16, the safety-margin statistic does not coincide with the 70%-VaR, indicating that the distribution of the pressure coefficient is also asymmetric, similar to the lift coefficient.

We show in Fig. 7.17 the decay of the quantities $\hat{b}^{(1)}_{l,new}$ and $\hat{V}_l$, defined in Eqs. (3.57) and (3.42), over the levels l. The quantities $\hat{b}^{(1)}_{l,new}$ and $\hat{V}_l$ are estimated using the final optimal hierarchy computed in one of the simulations corresponding to the finest tolerance tested in each of the fixed and adaptive mesh cases. As can be seen from Fig. 7.17, the problem exhibits ideal properties for the application of MLMC methods in both cases, showing an exponential decay in the levels l. We note however, that the levels are identified with different discretisation parameters in either of the cases; namely, the mesh parameters in the fixed case and the refinement error in the adaptive case. This makes the interpretation of the convergence rates

somewhat difficult.

In both the fixed and adaptive mesh cases, we conduct a reliability test similar to the tests conducted in Section 5.4 of Chapter 3; namely, we run 5 CMLMC simulations each for a set of decreasing tolerances prescribed on the MSE of the CVaR. These results are summarised in Fig. 7.18 for both fixed and adaptive mesh cases. Specifically, Figs. 7.18a and 7.18c demonstrate the reliability of the error estimation procedure. Two errors are plotted; namely the true squared error on the CVaR, computed using a reference CVaR that is the sample average over the 5 realisations of the CVaR produced at the finest tolerance tested, and the CVaR MSE estimate produced by the novel error estimation procedure of Chapter 3. As can be seen from both figures, the error estimates provide a sufficiently tight bound on the true error in the CVaR. However, neither the true nor the estimated errors decay smoothly with the tolerance, nor is the order of difference between them consistent for all tolerances, as was observed for the test cases in Chapter 3. In addition, Figs. 7.18b and 7.18d plot the estimated cost of simulating the optimal MLMC hierarchy versus the corresponding tolerance. In both fixed and adaptive cases, it can be seen that the cost to compute the MLMC hierarchy scales faster than the best-case MLMC performance, a result not predicted by Theorem 3.2.1.

Several factors could potentially contribute to the behaviours observed in Fig. 7.18, which demonstrates poorer performance than was observed in Chapter 3. The first is that due to computational budget and time constraints, it was possible to compute only 5 realisations for each tolerance, leading to poor estimates of the true MSE and the estimated mean cost. The second, possibly more significant, contributor could be the phenomenon of interval selection demonstrated in Section 3.4.2 of Chapter 3; namely that selecting too small an interval over which to construct the parametric expectation could lead to unstable rescaling ratio behaviour and, hence, unstable MSE estimation. Unfortunately, the airfoil simulations could not be re-run with a larger interval size to test this hypothesis, since the rescaling ratio phenomenon was discovered only after the loss of access to the MareNostrum4 supercomputer following the end of the ExaQUte project. It is hoped that in future research, this phenomenon can be accounted for and the test cases re-run with a larger number of CMLMC simulations for each prescribed tolerance.

The author of this thesis contributed to the simulations reported above in the following ways. The simulations were conducted using the CVaR estimation framework of Chapter 3, which the author implemented within the XMC library. Additionally, the author collaborated with the partners at CIMNE to integrate this XMC implementation of the work with the Kratos multi-physics software, to parallelise the implementation using the PyCOMPSs task scheduler, and to implement the join software framework on the MareNostrum 4 cluster. The data was then processed and analysed by the author to present the results on MLMC performance reported in this section.

The data set of results of these numerical experiments is publicly available as a part of [13].

7.2.3 CVaR optimisation for the potential flow

We now study a problem of CVaR minimisation for the airfoil problem introduced in Section 7.2.1. A NACA0012 airfoil is placed within a circular domain as shown in Fig. 7.12 and is subject to the same uncertainty in the inlet conditions as in Section 7.2.2; namely that the Mach number is distributed as $M_\infty \sim \mathcal{N}(0.3, 0.1^2)$ and the angle of attack is distributed as $\alpha_\infty \sim \mathcal{N}(5.0, 0.02^2)$ degrees. The flow around the airfoil is governed by the potential flow equation described in Eq. (7.28). We wish to minimise the 70%-CVaR of the negative of the lift coefficient over the space of possible airfoil designs, where the lift coefficient is computed as in Eq. (7.34). We chose to minimise the negative of the lift coefficient since, as was seen in Fig. 7.23, minimising the right tail of the distribution amounts to reducing the occurrences of small values of the lift coefficient.

We now describe the optimisation problem that was solved in [15]. We introduce the complete probability space $(\Omega, \mathscr{F}, \mathbb{P})$ and elementary random event $\omega \in \Omega$. We denote by $z \in \mathbb{R}^d$ the vector of design parameters that determine the shape of the airfoil, and by $u(\omega, z)$ the solution to Eq. (7.28) for a given random input ω and design z. In contrast to problem (5.6) in Chapter 5, an alternative constrained formulation was proposed in [15] as follows:

$$\begin{aligned} \mathscr{J}^* = \min_{\substack{z \in \mathbb{R}^d \\ \theta \in \mathbb{R}}} & \left\{ \mathscr{J}(\theta, z) := \Phi(\theta; z) \right\} \\ \text{s.t.} \quad & F(u(\omega, z), z) = 0 \quad \text{for} \quad \mathbb{P}\text{-a.e. } \omega \in \Omega, \\ & \text{and} \quad c(z) = 0, \end{aligned} \tag{7.37}$$

where Φ is as defined in Eq. (3.1) and corresponds to the CVaR statistic, $F(u(\omega, z), z) = 0$ denotes the residual operator corresponding to the underlying PDE, and $c(z)$ denotes the set of equality constraints on the design z.

To use gradient-based methods to solve problem (7.37), it is required to compute the sensitivities of the objective function $\mathscr{J}(\theta, z)$. The sensitivities, in turn, require the solution of the adjoint equation corresponding to problem (7.37). Additionally, three equality constraints are applied to the problem; namely that the chord length, the perimeter, and the angle of attack remain constant. The reader is referred to [14] for a detailed derivation of the adjoint equation corresponding to the "deterministic" version of problem (7.28). In addition, the reader is referred to [15] for a detailed extension of the deterministic adjoints to the case of CVaR minimisation, as well as a review of some literature on the trust-region algorithm used and the treatment of the constraints therein.

The underlying domain is discretised using triangular linear finite elements, and the forward and adjoint problems are solved as described in [100] and [15] respectively to produce realisations of the QoI $Q = -C_l$ and its sensitivity Q_z with respect to the design z. Although Algorithm 5 was not used in [15] to solve this problem, the estimation of the sensitivities of $\mathscr{J}(\theta, z)$ was carried out using the MLMC framework described in Chapter 5, wherein suitable

parametric expectations were used to compute the sensitivities. The adaptive remeshing procedure described in Section 7.2.2, as proposed by [41] and implemented in [42], is also used here to produce correlated samples pairs at a desired level l in the MLMC hierarchy; namely that each level is identified with a tolerance prescribed to the remeshing process, that exponentially decreases with levels. The solutions to Eq. (7.28) obtained in the final two steps of the refinement process are then used to compute correlated realisations of Q_l and Q_{l-1}.

However, there exists an important distinction in the meshing process in comparison to Section 7.2.2, with important consequences for the OUU procedure. We recall here the meshing issue that was described in Section 7.2.2; namely that the MMG remesher considered the geometry captured by the coarse mesh to be the "true" geometry, and did not incorporate geometry information in the remeshing process. As a result, it was required to define the shape of the airfoil through the coordinates of the mesh points on the surface of the initial airfoil design, and for this discretisation to remain unchanged throughout the optimisation. Since the refinement process generated a hierarchy of nested meshes for the MLMC hierarchy, this required the airfoil to already be well resolved by the level 0 mesh, and for the number of points on the surface of the mesh to remain unaffected by the refinement process across all MLMC levels. To this end, the airfoil was discretised with 10198 nodes, leading to twice as many design variables ($d = 20396$) due to the two dimensional nature of the problem. The corresponding starting mesh for the algorithm is shown in Fig. 7.19. Additional details are presented in [15] on smoothing the shape of the airfoil after performing a trust-region step to ensure smoothness of the design.

An important implication of the large number of design points was that the XMC implementation of the error estimation and MLMC adaptivity procedure described in Chapter 5 used to conduct the simulations became excessively expensive and unfeasible. Due to computational budget constraints, as a result, the adaptive hierarchy selection procedure was discarded and replaced by a fixed MLMC hierarchy. The fixed hierarchy is constructed with 4 levels and 256 samples per level.

Table 7.4: MLMC hierarchy used at every optimisation step.

Level l	Fine Samples	Coarse Samples	Refinement tolerance
0	256		
1	256	256	0.1
2	256	256	0.05
3	256	256	0.025

The ExaQUte software framework was utilised to simulate the optimisation problem. Specifically, the MLMC sensitivity estimation procedure was implemented using a parallelised verion of the XMC library, with the parallelism across samples and levels being exploited using the PyCOMPSs task scheduler. The underlying forward and adjoint problems were solved

using the Kratos multi-physics software, using the approaches described in [100] and [15] respectively. As described earlier, the MMG software package was used to carry out adaptive mesh refinement using the Hessian-based metric information computed by the Kratos multi-physics library. The simulations were conducted on the MareNostrum4 cluster of the Barcelona Supercomputing Centre [93]. The problem was run for a duration of 24 hours, after which the simulation was stopped due to wall clock time restrictions. The stopping criteria of the algorithm was set in terms of the relative change of the objective function, which was not reached within the wall clock time limit.

The results obtained from the simulation can be summarised as follows. We first note that since we use a fixed MLMC hierarchy, we do not eliminate or reduce the MLMC error as the optimisation progresses, in contrast to Algorithm 5 of Chapter 5. As a result, we can neither guarantee that we asymptotically reach the true optimum of the problem, nor can we make a claim on the decay rates of the objective function or the gradient in the optimisation steps similar to the results of Theorem 5.2.1. At best, since the MLMC decay rate hypotheses are satisfied for this problem (see Fig. 7.17), it may be expected that the objective function possibly decays for some initial iterations of the optimisation algorithm before saturating to a stable value when the error in the MLMC estimation exceeds the accuracy requirements of the optimisation algorithm. This behaviour can somewhat be observed in Fig. 7.20a, which shows the change of the objective function through the optimisation steps. It can be seen that the objective function eventually reaches a stable value, but not smoothly. The corresponding evolution of the relative change in the objective function is illustrated in Fig. 7.20b, and does not exhibit any meaningful systematic decay in value. A comparison of various airfoil designs is presented in Fig. 7.21; namely, we present the initial shape of the airfoil, the final design obtained from the OUU simulation described above, and an optimal design obtained from a deterministic minimisation conducted using the mean operating conditions of the uncertainties prescribed to the OUU problem.

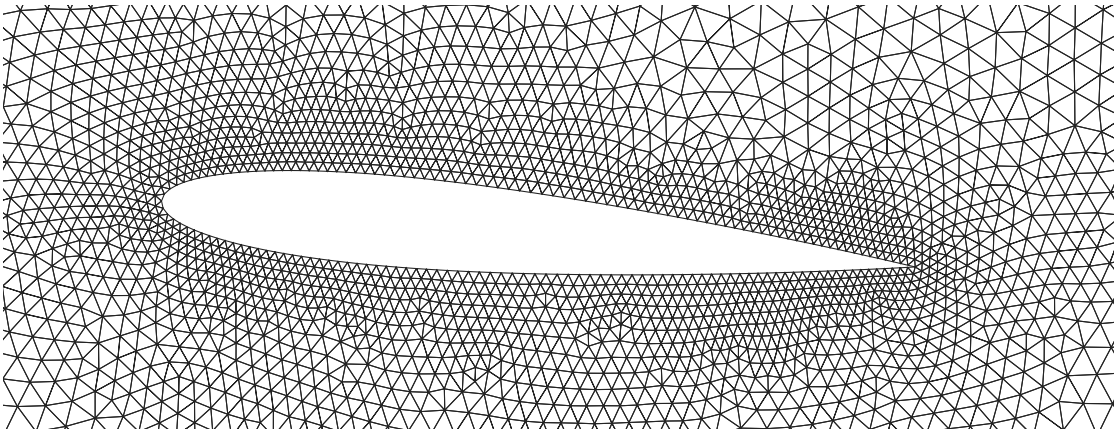
Fig. 7.22 shows the evolution of the empirical CDF of $-C_l$ computed using the 256 finest QoI realisations at each optimisation step. We observe that the distribution moves towards the left, minimising small values of the lift coefficient as is preferred for the airfoil design. In addition, Figs. 7.23a and 7.23b display histograms computed using the 256 finest samples of the MLMC hierarchy computed at the initial and final optimisation steps respectively. A scaled version of the empirical CDF is also shown, in addition to the mean, the estimated VaR and the estimated CVaR, for each of the two cases. We mention that, in addition to the distribution of the QoI moving to the left, as was also observed in Fig. 7.22, the final PDF is also observed to have a larger variance than the initial PDF, specifically with a longer left tail.

In addition to the lift coefficient, we present the behaviour of the pressure coefficient in Fig. 7.24 for the final optimisation iteration, computed according to Eq. (7.35). The pressure coefficient distribution of the initial shape is identical to the distribution shown in Fig. 7.16 for the UQ problem, and is hence, not repeated here. At each point on the surface of the airfoil, we utilise the MLMC samples of C_p to compute estimates of the mean, the VaR and the

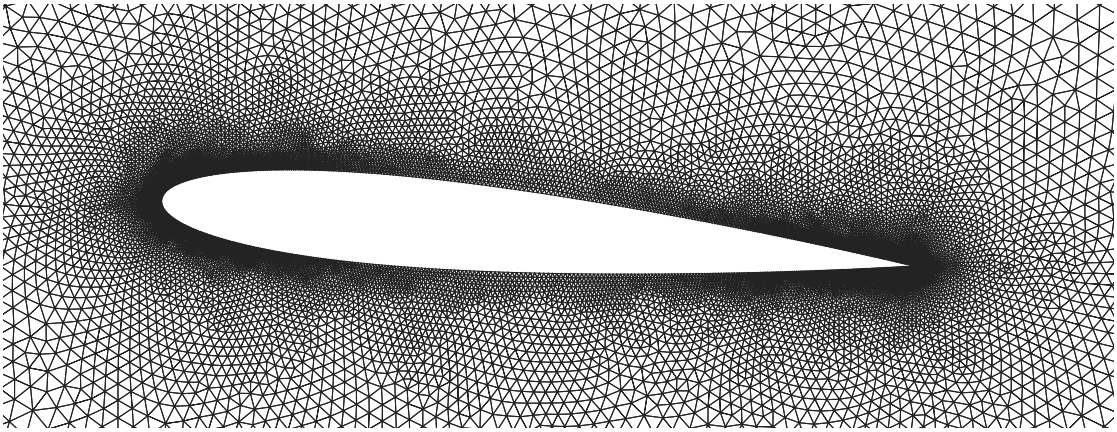
CVaR, in addition to the worst-case, safety-margin and three-sigma statistics introduced in Section 7.2.2 in Fig. 7.16. We note the presence of some numerical artefacts in the pressure coefficient distribution corresponding to the final iteration; namely the presence of a wave-like variation of the lift coefficient over both surfaces of the airfoil, as well as a sharp variation on the upper surface at $x = -0.2$ approximately. Several factors could be contributing to the presence of these artefacts, including the inaccurate MLMC estimation and its combination with the trust region algorithm, as well as the discrete representation of the airfoil. Unfortunately, these artefacts could not be rectified and the corresponding simulations could not be re-run due to the restriction of access to the MareNostrum4 cluster, following the completion of the ExaQUte project.

The results produced in [15] and presented here are a showcase of an engineering application with many design parameters for which the ExaQUte software framework has been applied successfully on a supercomputer, albeit with caveats on the mathematical consistency of the numerical approximations used. This was possible due to the combination of the different utilities and software packages released during the project [99, 6, 23]; namely, the remeshing software [36]; the solver [95]; the task schedulers [22] and [29]; and the uncertainty quantification library [5]. The author of this thesis contributed to these simulations in the following ways; namely by implementing the framework of MLMC estimation of the sensitivities in the XMC software package, and integrating this implementation with the PyCOMPSs scheduler to exploit parallelism, as well as the Kratos multi-physics package to solve the underlying forward and adjoint problems. The author was also involved in the design of the simulations, as well as in the processing and assessment of the data in order to produce the results presented herein.

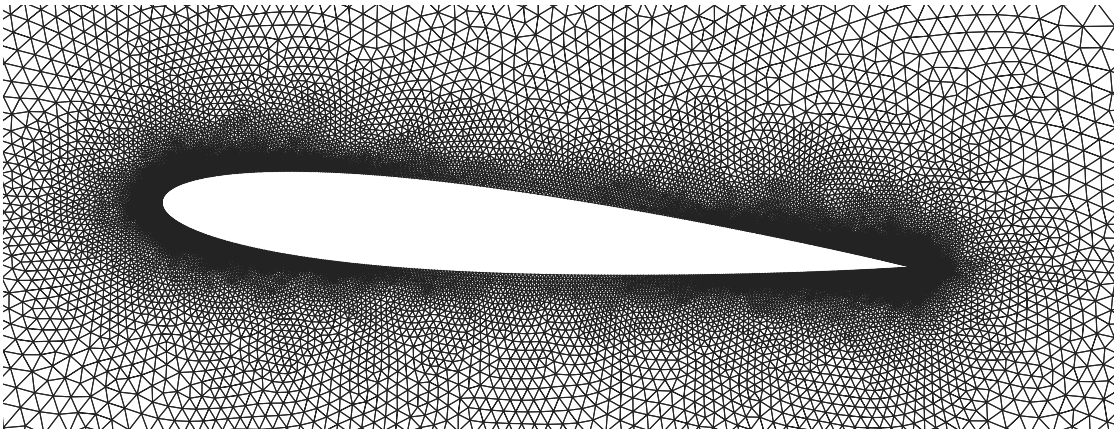
The data set of raw results of these numerical experiments is also available publicly at [13].



(a) Pre-generated level 0 mesh

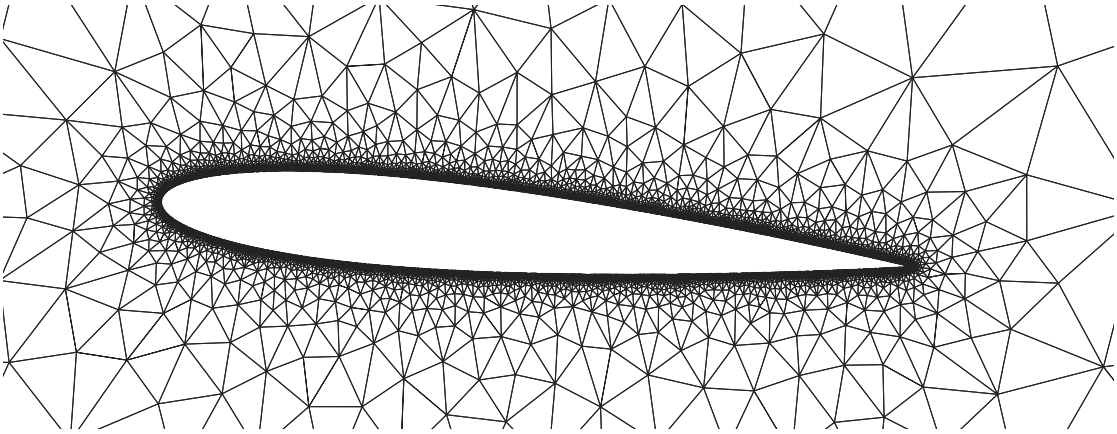


(b) Pre-generated level 1 mesh

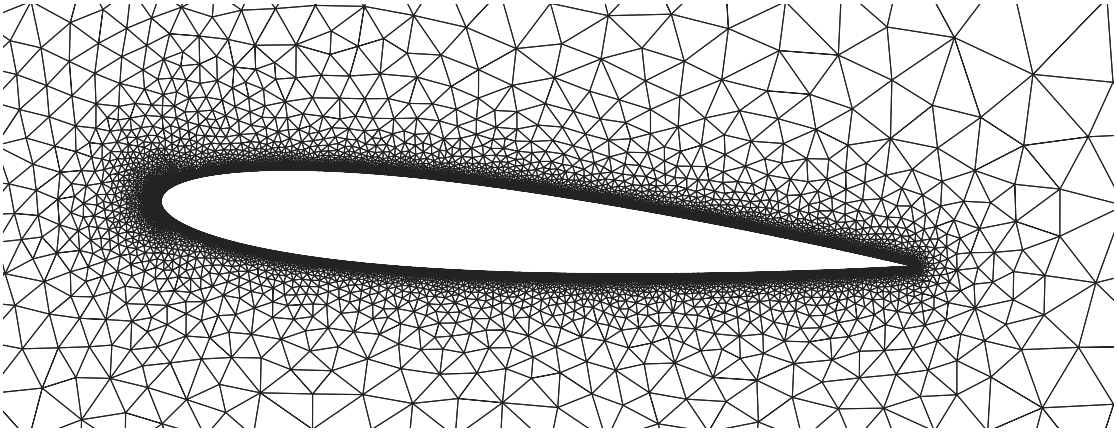


(c) Pre-generated level 2 mesh

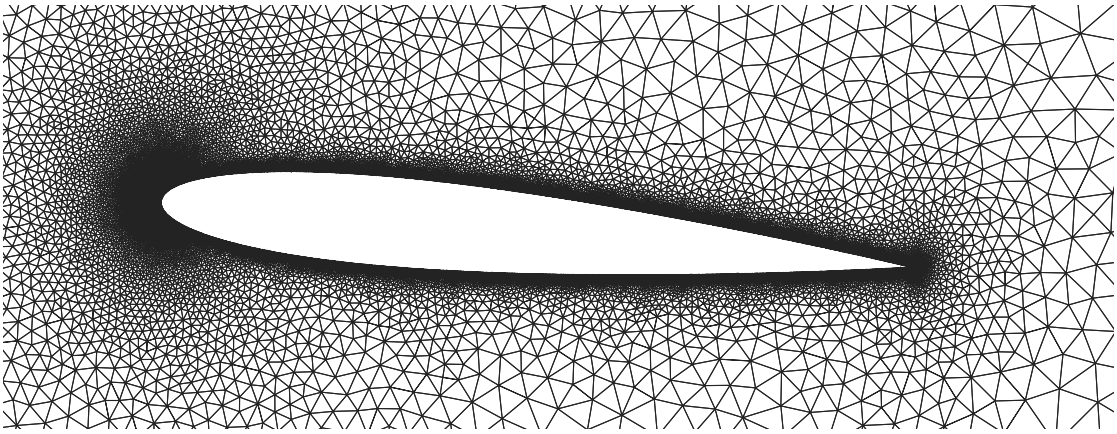
Figure 7.13: Meshes corresponding to the first three levels of the MLMC hierarchy in the fixed case.



(a) Level zero mesh from which adaptive refinment begins



(b) Adaptively refined mesh corresponding to level 1



(c) Adaptively refined mesh corresponding to level 2

Figure 7.14: Meshes corresponding to the first three levels of the MLMC hierarchy using adaptive refinement.

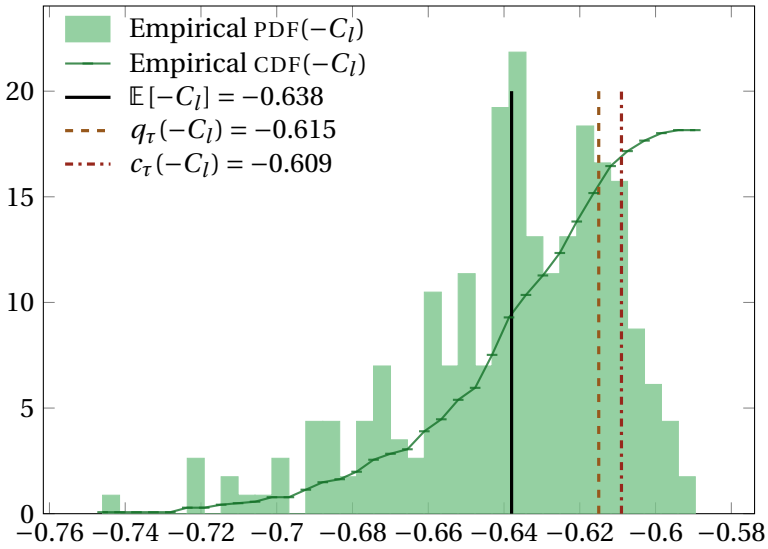
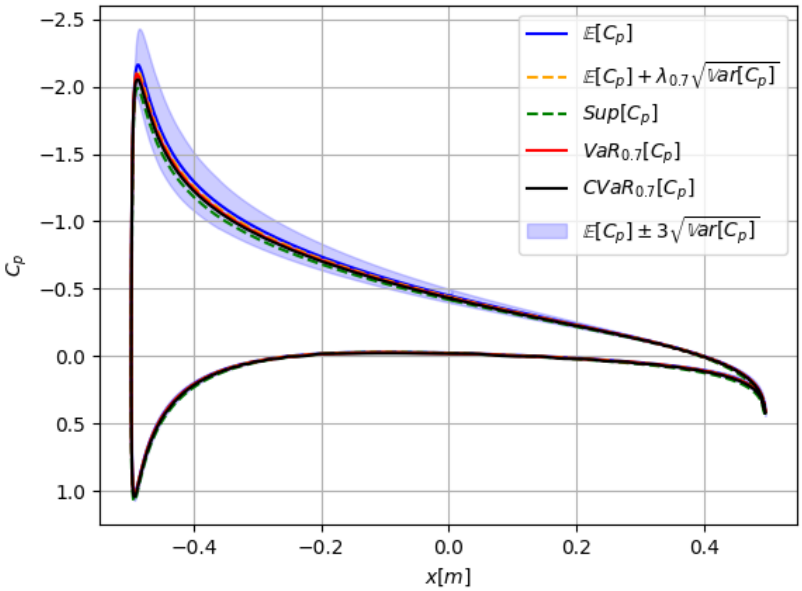
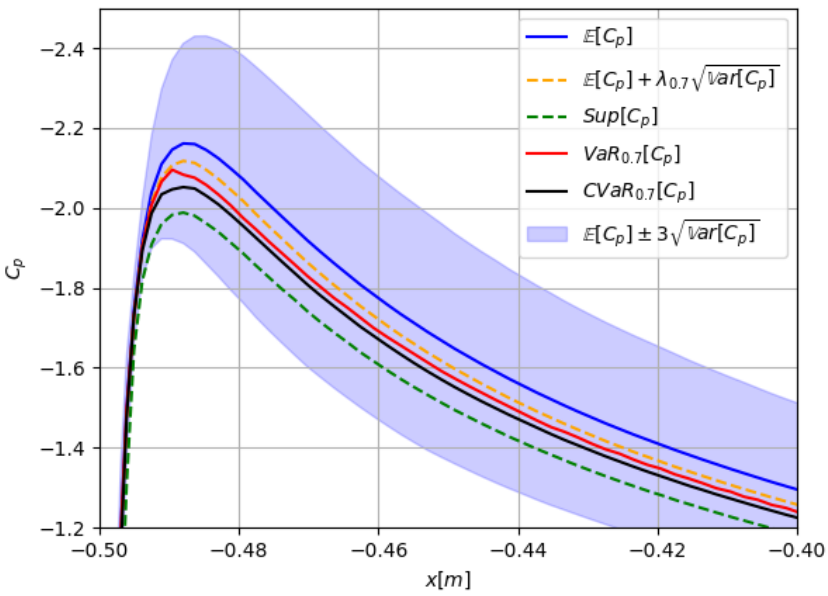


Figure 7.15: Distribution of $-C_l$.

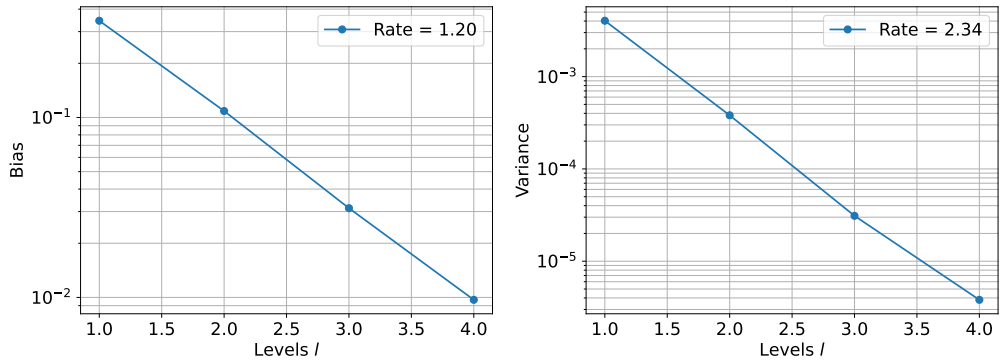


(a) Pressure distribution over the airfoil.



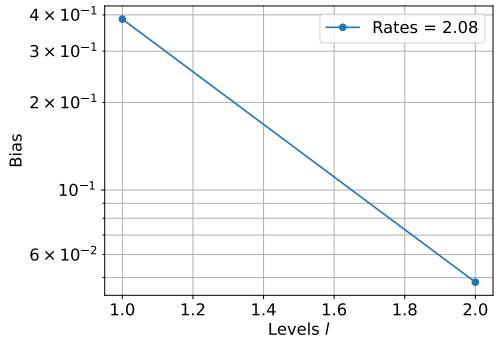
(b) Zoom in at the leading edge

Figure 7.16: Distribution of statistical quantities for the pressure field

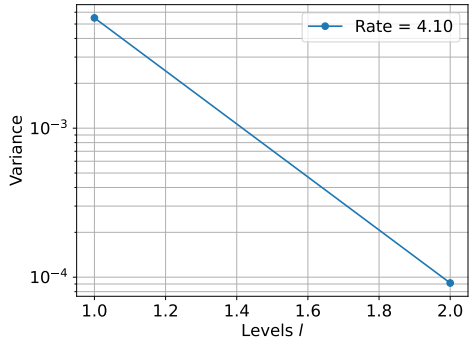


(a) Bias decay for fixed case

(b) Variance decay for fixed case

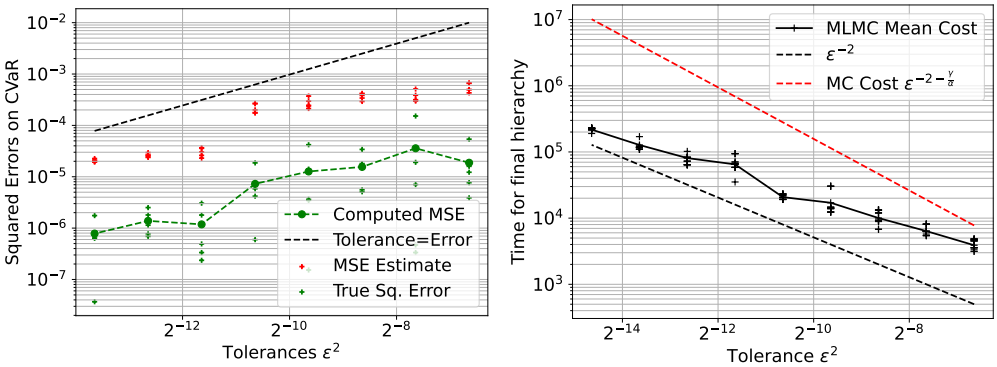


(c) Bias decay for adaptive case



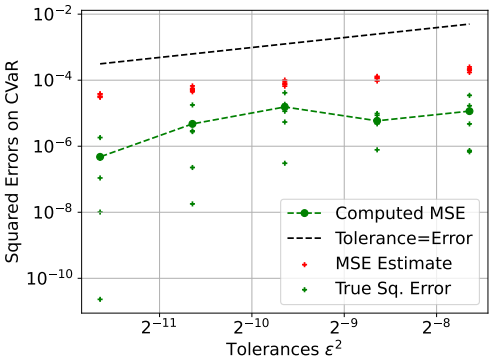
(d) Variance decay for adaptive case

Figure 7.17: Bias and variance decay for airfoil problem with fixed and adaptive meshes.

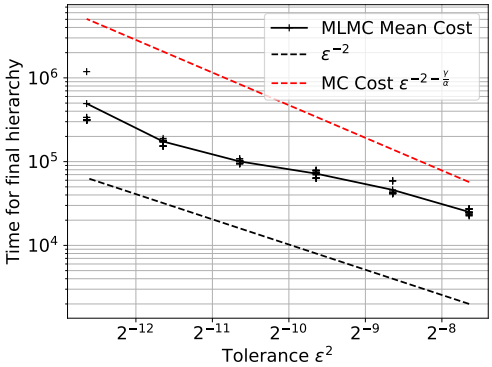


(a) Fixed case error estimator reliability

(b) Fixed case complexity behaviour



(c) Adaptive case error estimator reliability



(d) Adaptive case complexity behaviour

Figure 7.18: Summary of results for the airfoil problem

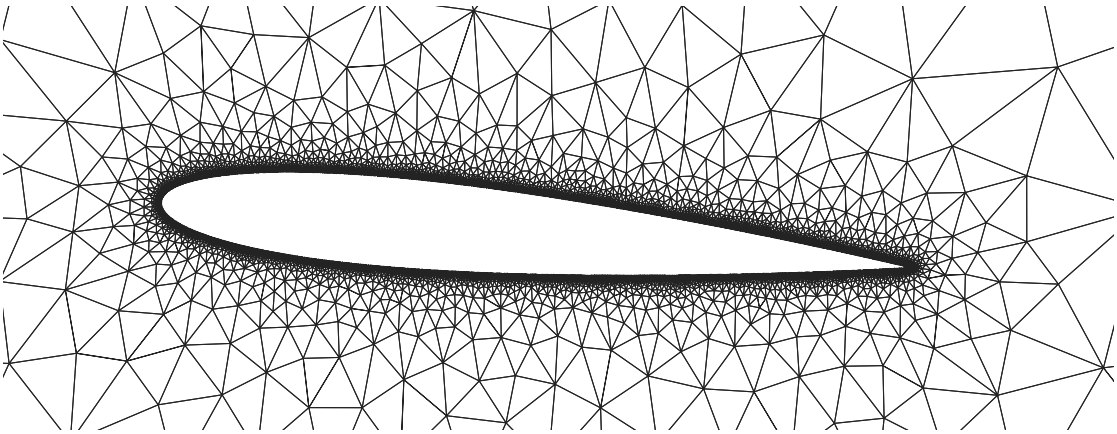
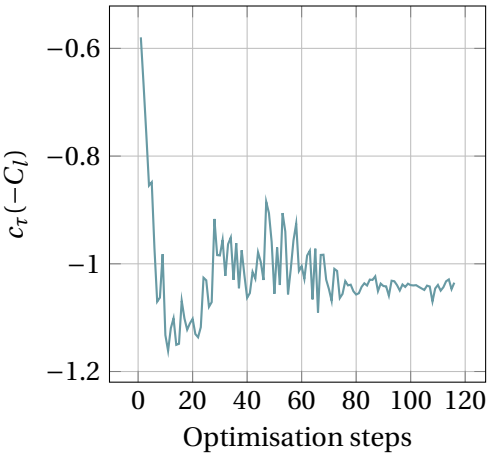
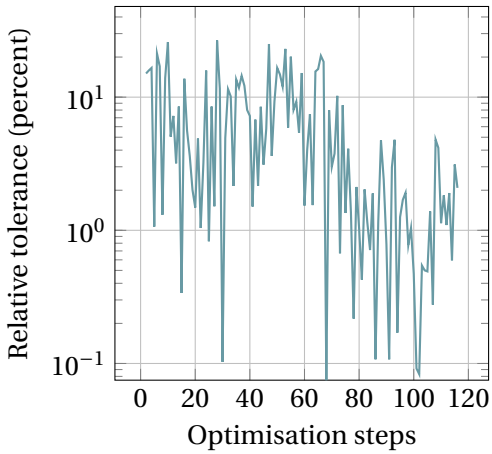


Figure 7.19: Starting mesh of the optimisation problem



(a) Objective function



(b) Relative change of the objective function

Figure 7.20: Evolution of the objective function over optimisation iterations

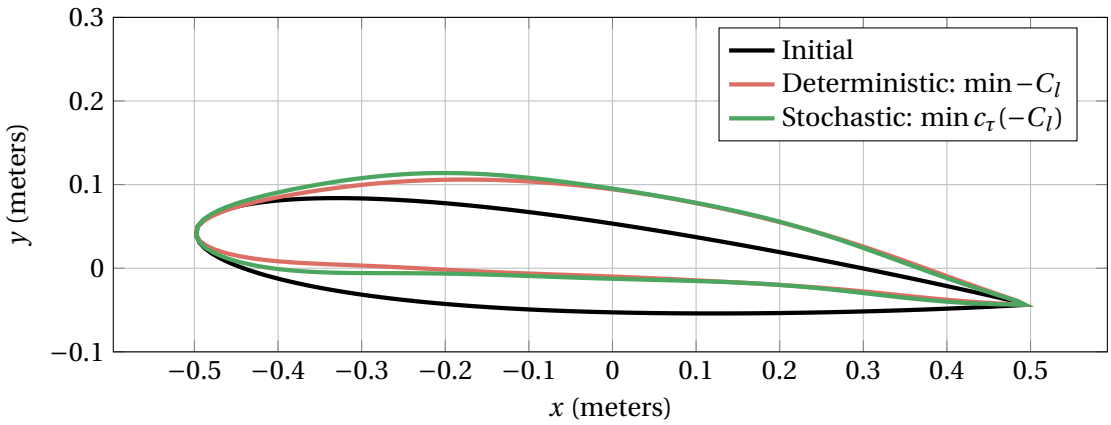


Figure 7.21: Comparison of the initial shape and various optimal designs

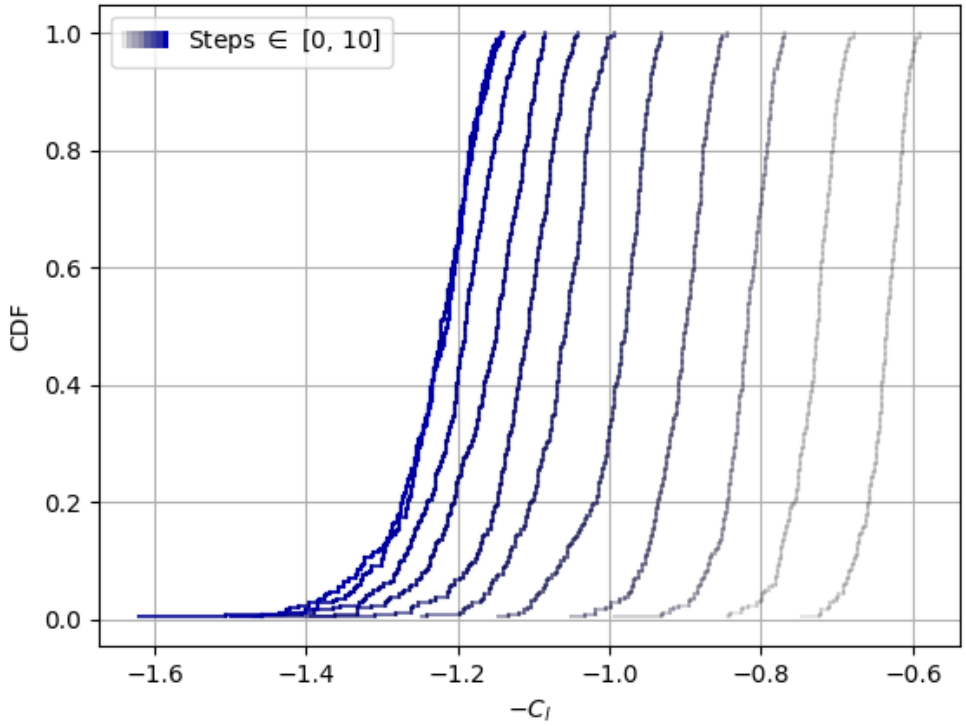
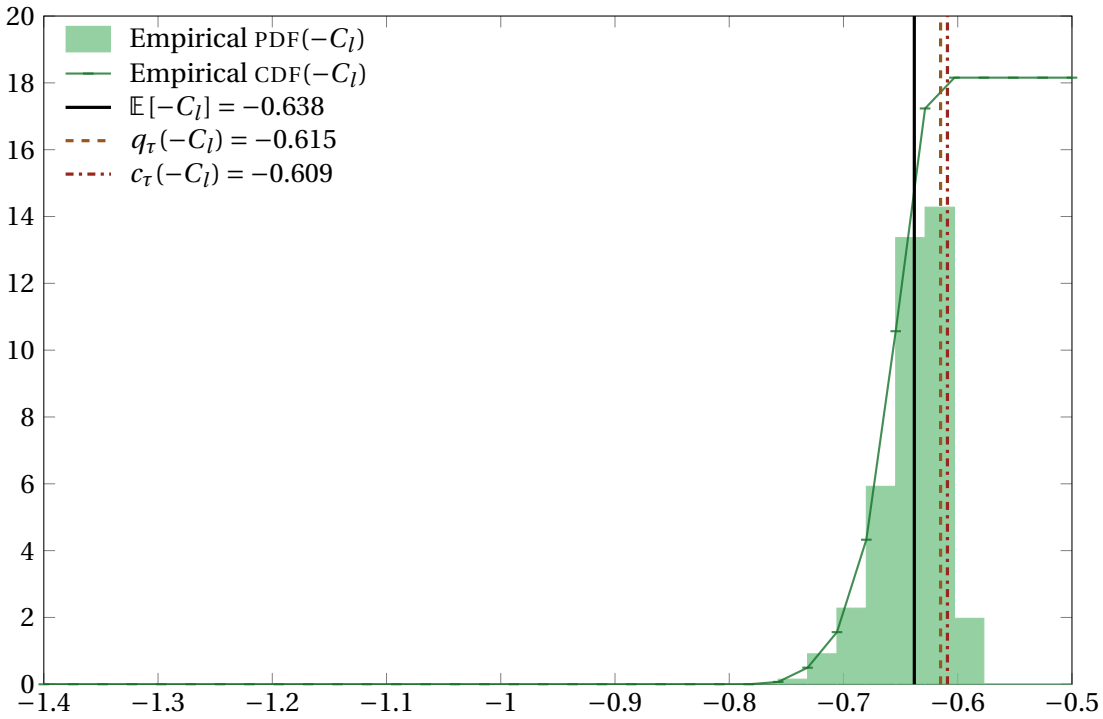
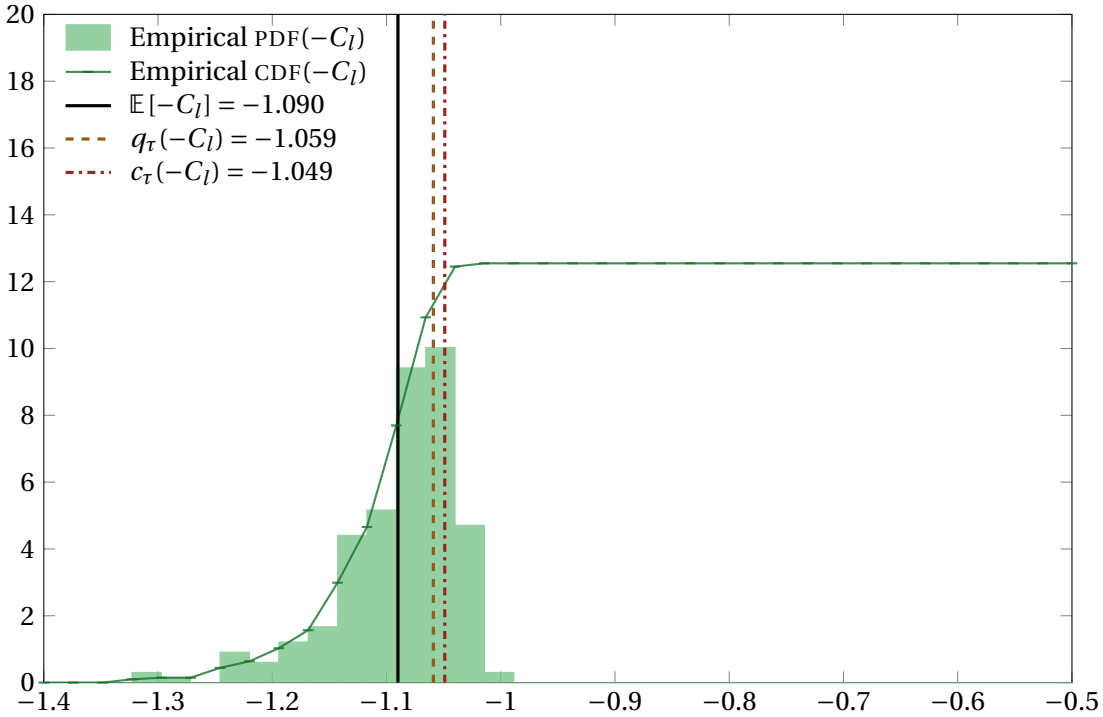


Figure 7.22: Empirical CDF for the first ten iterations of the optimisation algorithm. Darker colours represent later iterations.

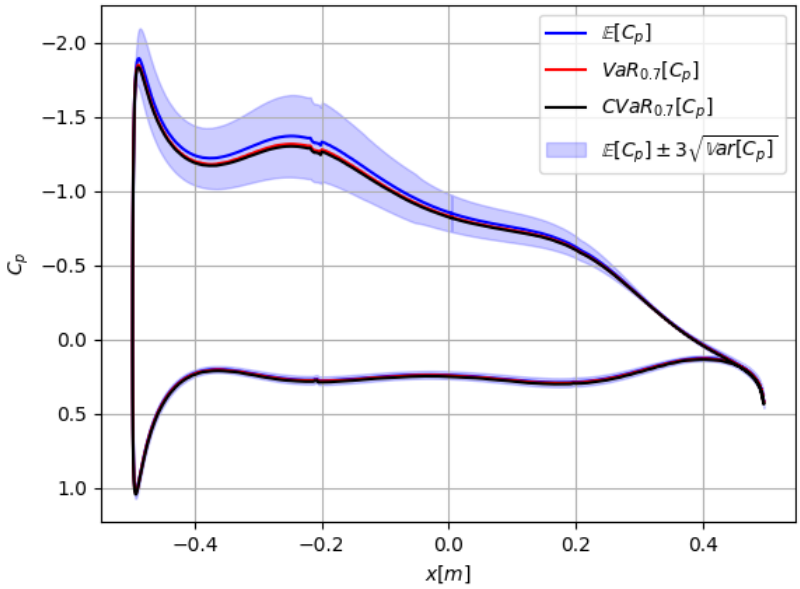


(a) Initial shape

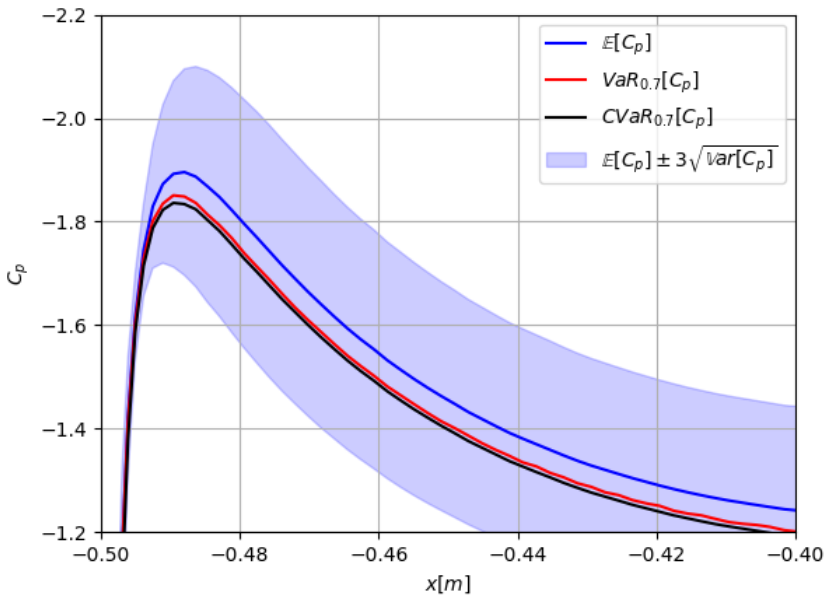


(b) Final shape

Figure 7.23: Distribution of QoI for initial and final optimisation steps.



(a) Pressure distribution along the airfoil shape



(b) Zoom in at the leading edge

Figure 7.24: Distribution of the pressure coefficient over the final airfoil shape.

7.2.4 Conclusion

We presented in this chapter the results of several collaborative production level simulations conducted during the ExaQUte project using the ExaQUte software framework. In Section 7.1, we presented results from [18] on the feasibility of using the MLMC method with time-dependent problems. It was shown that pathwise correlation of the time signals was required in order to ensure convergence in the discretisation parameter, which could only be ensured for very short time windows for chaotic problems. We demonstrated, through two simple oscillator problems and a more applied example of turbulent fluid flow over a rectangle, that although the MLMC decay rate hypotheses of Eqs. (2.8) could be satisfied for oscillatory problems, satisfying the hypotheses for chaotic problems still remains a challenge due to the difficulty in preserving pathwise correlation. Although the decay rate hypotheses were not satisfied for the turbulent flow problem, we observed that correlations still existed between the different discretisation levels. To exploit these correlations, we proposed the use of MFMC estimators in [18]. In addition, MFMC simulations were presented in the report [20] for an engineering application, which the author of this thesis was however not involved in and, therefore, have not been reported in this thesis.

In Section 7.2, we presented the results of collaborative production simulations that were conducted using the ExaQUte software framework for the reports [20] and [15]. The aim of the simulations was to demonstrate that the implementation of the MLMC procedures from Chapters 3 and 5 in the XMC library, integrated with the ExaQUte software framework, could be utilised successfully for an applied engineering problem, and to highlight the contributions of the author of this thesis towards this end. In Section 7.2.2, we estimated the 70%-CVaR for a problem of potential flow over an airfoil with inlet uncertainties. It was demonstrated that although the problem satisfied the hypotheses in Eq. (2.8), poor error estimator performance was observed, possibly due to the rescaling ratio issue discussed in Chapter 3. However, due to loss of access to the supercomputing hardware of the consortium, we were unable to rectify these issues and re-run the corresponding simulations. In terms of software however, the ExaQUte software framework was found to perform as expected, demonstrating the successful code integration efforts carried out by the members of the consortium and the author of this thesis.

In Section 7.2.3, we tackled a problem of CVaR minimisation for the potential flow problem over an airfoil. It was desired to minimise the CVaR of the negative of the lift coefficient, leading to higher values of lift, by controlling the shape of the airfoil, using a trust region algorithm. However, due to limitations in the capability of the MMG remeshing software that caused a dramatic increase in the computational cost of the XMC library, it was required to discard the adaptive MLMC procedure described in Chapter 5 for the accurate estimation of the gradient and, instead, utilise a fixed MLMC hierarchy. Although the corresponding numerical results demonstrated a minimisation of the CVaR and a corresponding favourable shift in the distribution of the lift coefficient, several numerical artefacts were found in the simulations. Once again, due to the unfortunate loss of access to computing hardware fol-

lowing the completion of the ExaQUte project, we were unable to rectify the artefacts, and presented the results here as they were at the time of simulation. We once again highlight that in terms of software, the ExaQUte software framework performed as expected, giving us the ability to simulate the MLMC hierarchy in a parallel and scalable manner.

For future research, we believe that the exploration of MFMC estimators to estimate sensitivities of the CVaR with respect to design parameters would be a highly promising research direction, enabling the use of MFMC methods for optimisation problems wherein the MLMC hypotheses (2.8) are not satisfied. It is also hoped that the numerical artefacts presented in Sections 7.2.2 and 7.2.3 can be rectified through adequate calibration of the MLMC estimation procedure through the selection of a more appropriate interpolation interval, or an automated interval selection procedure, and through the extension of the capabilities of the meshing software. Lastly, we briefly explored the concept of an asynchronus MLMC algorithm outisde of this thesis during the ExaQUte project, wherein estimates of the level-wise biases and variances required to estimate an optimal hierarchy could be updated on-the-fly in a continuous manner. An implementation of such an algorithm, however for a fixed hierarchy, was presented in [7], to which the author of this thesis has not, however, contributed. We believe that an adaptive version of this algorithm would prove highly beneficial for scalable high-performance implementations of the MLMC estimation procedure.

8 Conclusions and Outlook

8.1 Conclusions

As was described in Chapter 1, this thesis research was conducted as a part of the ExaQUte project [40], a European Union Horizon 2020 research project aimed at the development of technologies to enable exascale uncertainty quantification and risk-averse optimisation. The aim of the project was to carry out risk-estimation and risk-averse optimisation for a problem of civil engineering design. The project required the development of several novel mathematical and algorithmic technologies, as well as an integrated high-performance software framework capable of executing these technologies on high-performance hardware. The project was ambitious in its goals, and was deemed by a scientific review committee to have successfully satisfied the goals of the project in a timely manner. Several important contributions were made by this thesis towards the completion of the ExaQUte project goals.

The first key aim of the project was the development of novel MLMC estimators and algorithms for risk-estimation for use with complex differential models whose input uncertainties and output QoI were modelled as random variables. Our research work [16], reported in Chapter 3, tackled this challenge and aimed to develop an accurate MLMC estimation procedure for the CVaR, a risk-measure that quantifies the tails of the distribution of the random QoI and possesses properties favourable for use with risk-averse optimisation algorithms. Specifically, we proposed to use the framework of parametric expectations introduced in [85] to accurately estimate the CVaR. The developments, presented in Chapter 3, included novel error estimators, a novel adaptive MLMC hierarchy selection procedure, and a novel CMLMC algorithm to incrementally calibrate the MLMC hierarchy to obtain accurate estimates of robustness measures such as the PDF, the CDF, the VaR and the CVaR. We also proved that our novel procedure preserved the optimal cost complexity behaviour predicted for the MLMC estimation of the expected value in the case of the CVaR. We successfully demonstrated our approach on three problems of interest; namely, the Poisson problem, the Black-Scholes SDE, and the steady Navier-Stokes equations modelling the flow around a cylinder placed in a channel.

The second key aim of the project was the development of novel optimisation algorithms for the gradient-based minimisation of the CVaR, using the MLMC method to reduce the cost of simulating the underlying PDE. Our work [52] addressed this challenge, and presented a novel gradient-based approach to solve a problem of penalised CVaR estimation. The work built upon our earlier research on MLMC estimators for parametric expectations, extending the work in [16] to propose novel MLMC estimators for the sensitivities of the CVaR with respect to design parameters. This additionally allowed us to combine the MLMC approach with an AMGD algorithm, alternatively minimising in one coordinate of the objective function to obtain the VaR, and subsequently conducting a gradient step in the remaining design variables. Under suitable assumptions on the objective function, we proved exponential convergence in the optimisation iterations towards the minimiser. We demonstrated our framework on two examples of interest; namely the FitzHugh–Nagumo oscillator and an advection-reaction-diffusion equation used to model pollutant transport. We showed that our optimisation algorithm reproduces the same cost complexity behaviour as the underlying MLMC estimation procedure.

The third key aim of the ExaQute project was the development of an overarching software framework capable of integrating the different mathematical developments produced within the project together into a high-performance implementation capable of achieving scalable performance on high-performance hardware. To this end, we extensively developed the X-Monte Carlo (XMC) software library [5], a library capable of implementing several hierarchical Monte Carlo estimators and algorithms. As discussed in Chapters 2 and 6, the software library was designed to mirror the common structures observed in hierarchical Monte Carlo estimators; namely, the idea of successive differences between the various hierarchy levels, as well as the common pattern of adaptive Monte Carlo algorithms. As of the writing of this conclusion, the XMC library is capable of implementing MLMC, MIMC and some MFMC estimators, in addition to several adaptive algorithms. Extensive work has also been carried out on the integration of the XMC library with the other component libraries of the ExaQUte software framework [23]; namely, the author aided in the parallelised implementation of the MLMC estimators for parametric expectations and for the sensitivities of the CVaR within the XMC library, as well as the integration of these implementations with the Kratos multi-physics solver and the PyCOMPSs/Hyperloom task schedulers.

These integration efforts culminated in the ability to simulate several engineering application problems in parallel on high-performance hardware [93]. We presented in Chapter 7 the results of several of these simulations; namely to study the feasibility of the MLMC approach for turbulent and chaotic problems, and to demonstrate the implementation of the works in Chapters 3 and 5 within the ExaQUte software framework. Although the ExaQUte software framework performed well, we observed several challenges that we hope to tackle in future research. It was found that the MLMC decay rate hypotheses presented in Chapter 2 may not be satisfied for turbulent fluid flows, due to the chaotic nature of the flow. The MFMC method was proposed as a plausible alternative to the MLMC approach, and some results were presented in [20] on a civil engineering problem. Additionally, due to limitations in

the capability of the remeshing software utilised by Kratos, several numerical artefacts were observed in the results of Chapter 7.

8.2 Outlook and Future Scope

Our research into risk-estimation and risk-averse optimisation raised several important mathematical questions that we hope to tackle in future scope. In Chapter 3, we reported that the performance of our error estimation procedure was sensitive to the choice of interval over which we construct the parametric expectations; namely that larger intervals provided more conservative error estimates, and smaller intervals were likely to produce numerical instabilities. We hope to examine the issue further in future works, with the aim to adaptively select an interval such that optimal error estimator performance can be achieved.

In Chapter 5, we had demonstrated the successful performance of our approach using a gradient-based approach. It still remains to be seen whether the MLMC estimation of the sensitivities of the CVaR, as well as the AMGD procedure, can be extended to higher-order gradient-based methods such as the Newton based method. This would require a proof of the second order differentiability of the CVaR in the design parameters, which we are as yet unaware of in literature.

Although we presented a parallelisation of the XMC library using task-based parallelism in Chapter 6, the parallelisation strategy contained a natural synchronisation point, which could cause a loss of efficiency in the task-based paradigm due to idle processors. We briefly explored the concept of an asynchronous MLMC algorithm during the ExaQUte project, where estimates of the relevant level-wise errors would be updated continuously on-the-fly as more samples were computed. Mathematically, algorithmically and in terms of software capability, this would require several novel developments, but could potential lead to highly scalable implementations of MLMC estimators on high-performance hardware.

In Chapters 7, we explored briefly the use of adaptive mesh refinement strategies to refine the underlying discretised PDE such that the MLMC decay rate hypotheses would be respected. We demonstrated a strategy wherein the underlying discretisation is refined for each sample, and the levels of the MLMC hierarchy are identified with a geometrically decreasing set of tolerances provided to the adaptive refinement process. We proposed in [19] the use of an alternative strategy, wherein metric information from all of the samples at a given level could be used to refine a common mesh used for all samples at that level. The strategy is expected to fit well within a continuation-type MLMC algorithm. We believe that developing this strategy further is key to extending the MLMC method to more complex applications.

In Chapters 7 we observed several numerical artefacts that were caused due to the limitations described in Chapter 3, as well as limitations in the capabilities of the remesher used in the ExaQUte software framework. Due to computational budget and cluster access constraints, we were unable to rectify these artefacts as of the writing of this thesis. We hope to revisit

these simulations in future research, and hope to demonstrate the successful performance of our methodologies on additional challenging applications.

8.3 Funding acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programmed under grant agreement No. 800898.

Bibliography

[1] A. Ahmadi-Javid. "Entropic value-at-risk: A new coherent risk measure". In: *Journal of Optimization Theory and Applications* 155.3 (2012), pp. 1105–1123.

[2] A. A. Ali, E. Ullmann, and M. Hinze. "Multilevel Monte Carlo analysis for optimal control of elliptic PDEs with random coefficients". In: *SIAM/ASA Journal on Uncertainty Quantification* 5.1 (2017), pp. 466–492.

[3] R. Amela, Q. Ayoul-Guilmard, R. M. Badia, S. Ganesh, F. Nobile, R. Rossi, and R. Tosi. *ExaQUte XMC*. Comp. software. ExaQUte consortium, Oct. 2020. DOI: 10.5281/zenodo.4265429.

[4] R. Amela, Q. Ayoul-Guilmard, R. M. Badia, S. Ganesh, F. Nobile, R. Rossi, and R. Tosi. *XMC. Python library for hierarchical Monte Carlo algorithms*. Version 1.0.0. May 31, 2019. DOI: 10.5281/zenodo.4265429.

[5] R. Amela, Q. Ayoul-Guilmard, R. M. Badia, S. Ganesh, F. Nobile, R. Rossi, and R. Tosi. *XMC. Python library for hierarchical Monte Carlo algorithms*. Version 2.0.0. Nov. 10, 2020. DOI: 10.5281/zenodo.3235832.

[6] R. Amela, Q. Ayoul-Guilmard, R. M. Badia, S. Ganesh, F. Nobile, R. Rossi, and R. Tosi. *Release of ExaQUte MLMC Python engine*. Deliverable 5.2. Version 1.0. ExaQUte consortium, May 30, 2019. DOI: 10.23967/exaqute.2021.2.024.

[7] R. Amela, R. Badia, S. Böhm, and R. Tosi. *Profiling report of the partner's tools, complete with performance suggestions*. Deliverable 4.2. ExaQUte consortium. DOI: 10.23967/exaqute.2021.2.023.

[8] P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. "Performance and Scalability of the Block Low-Rank Multifrontal Factorization on Multicore Architectures". In: *ACM Transactions on Mathematical Software* 45 (1 2019), 2:1–2:26.

[9] P. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. "A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling". In: *SIAM Journal on Matrix Analysis and Applications* 23.1 (2001), pp. 15–41.

[10] H. Antil, D. P. Kouri, M.-D. Lacasse, and D. Ridzal. *Frontiers in PDE-constrained Optimization*. Vol. 163. The IMA Volumes in Mathematics and its Applications. Springer, 2018.

[11] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath. "Coherent measures of risk". In: *Mathematical finance* 9.3 (1999), pp. 203–228.

[12] R. Avikainen. "On irregular functionals of SDEs and the Euler scheme". In: *Finance and Stochastics* 13.3 (2009), pp. 381–401.

[13] Q. Ayoul-Guilmard, R. M. Badia, J. Ejarque, S. Ganesh, A. Kodakkal, F. Nobile, M. Núñez, J. Pons-Prats, J. Principe, R. Rossi, C. Soriano, and R. Tosi. *ExaQUte Data*. Data set. Version 1. ExaQUte consortium. Zenodo, Nov. 30, 2021. DOI: 10.5281/zenodo.5729258.

[14] Q. Ayoul-Guilmard, S. Ganesh, A. Kodakkal, F. Nobile, and M. Núñez. *Report on stochastic optimisation for unsteady problems*. Deliverable 6.4. Version 1.0. ExaQUte consortium, Dec. 14, 2020. 56 pp. DOI: 10.23967/exaqute.2021.2.003.

[15] Q. Ayoul-Guilmard, S. Ganesh, A. Kodakkal, F. Nobile, and M. Núñez. *Report on stochastic optimisation for wind engineering*. Deliverable 6.5. Version 1.0. ExaQUte consortium, Nov. 30, 2021.

[16] Q. Ayoul-Guilmard, S. Ganesh, S. Krumscheid, and F. Nobile. "Quantifying uncertain system outputs via the multilevel Monte Carlo method — distribution and robustness measures". In: *arXiv preprint arXiv:2208.07252* (2022).

[17] Q. Ayoul-Guilmard, S. Ganesh, and F. Nobile. *Report on stochastic optimisation for simple problems*. Deliverable 6.3. Version 1.3. ExaQUte consortium, July 27, 2020. 43 pp. DOI: 10.23967/exaqute.2021.2.001.

[18] Q. Ayoul-Guilmard, S. Ganesh, F. Nobile, M. Núñez, R. Rossi, and R. Tosi. *Report on MLMC for time-dependent problems*. Deliverable 5.4. Version 1.0. ExaQUte consortium, Nov. 13, 2020. DOI: 10.23967/exaqute.2021.2.005.

[19] Q. Ayoul-Guilmard, S. Ganesh, F. Nobile, M. Núñez, R. Rossi, and R. Tosi. *Report on theoretical work to allow the use of MLMC with adaptive mesh refinement*. Deliverable 5.3. Version 1.0. ExaQUte consortium, May 30, 2020. 31 pp. DOI: 10.23967/exaqute.2021.2.002.

[20] Q. Ayoul-Guilmard, S. Ganesh, F. Nobile, M. Núñez, and R. Tosi. *Report on the application of MLMC to wind engineering*. Deliverable 5.5. Version 1.0. ExaQUte consortium, Nov. 30, 2021. DOI: 10.23967/exaqute.2022.3.03.

[21] I. Babuška, F. Nobile, and R. Tempone. "A stochastic collocation method for elliptic partial differential equations with random input data". In: *SIAM Journal on Numerical Analysis* 45.3 (2007), pp. 1005–1034.

[22] R. M. Badia, J. Conejero, C. Diaz, J. Ejarque, D. Lezzi, F. Lordan, C. Ramon-Cortes, and R. Sirvent. "COMP Superscalar, an interoperable programming framework". In: *SoftwareX* 3–4 (Dec. 2015). DOI: 10.1016/j.softx.2015.10.004.

[23] R. M. Badia, S. Böhm, and J. Ejarque. *Framework development and release*. Deliverable 4.5. Version 1.0. ExaQUte consortium, Nov. 30, 2021.

[24] C. Bayer, C. B. Hammouda, and R. Tempone. "Multilevel Monte Carlo Combined with Numerical Smoothing for Robust and Efficient Option Pricing and Density Estimation". In: *arXiv preprint arXiv:2003.05708* (2020).

[25] J. Beck, L. Tamellini, and R. Tempone. "IGA-based Multi-Index Stochastic Collocation for random PDEs on arbitrary domains". In: *Computer Methods in Applied Mechanics and Engineering* 351 (2019), pp. 330–350.

[26] F. Beiser, B. Keith, S. Urbainczyk, and B. Wohlmuth. "Adaptive sampling strategies for risk-averse stochastic optimization with constraints". In: *arXiv preprint arXiv:2012.03844* (2020).

[27] C. Bierig and A. Chernov. "Estimation of arbitrary order central statistical moments by the multilevel Monte Carlo method". In: *Stoch. Partial Differ. Equ. Anal. Comput.* 4.1 (2016), pp. 3–40. DOI: 10.1007/s40072-015-0063-9.

[28] C. Bierig and A. Chernov. "Approximation of probability density functions by the multilevel Monte Carlo maximum entropy method". In: *Journal of Computational Physics* 314 (2016), pp. 661–681.

[29] S. Böhm, J. Beránek, and M. Surkovsky. *Quake*. https://code.it4i.cz/boh126/quake. IT4I, Nov. 30, 2021.

[30] S. Böhm and J. Ejarque. *ExaQUte API*. https://github.com/ExaQUte-project/exaqute-api. IT4I, BSC, Nov. 30, 2021.

[31] R. Bollapragada, R. Byrd, and J. Nocedal. "Adaptive sampling strategies for stochastic optimization". In: *SIAM Journal on Optimization* 28.4 (2018), pp. 3312–3343.

[32] R. Bollapragada, J. Nocedal, D. Mudigere, H.-J. Shi, and P. T. P. Tang. "A progressive batching L-BFGS method for machine learning". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 620–629.

[33] T. Breiten and K. Kunisch. "Riccati-based feedback control of the monodomain equations with the Fitzhugh–Nagumo model". In: *SIAM Journal on Control and Optimization* 52.6 (2014), pp. 4057–4081.

[34] L. Bruno, M. V. Salvetti, and F. Ricciardelli. "Benchmark on the aerodynamics of a rectangular 5: 1 cylinder: an overview after the first four years of activity". In: *Journal of Wind Engineering and Industrial Aerodynamics* 126 (2014), pp. 87–106.

[35] V. Cima, S. Böhm, J. Martinovič, J. Dvorsky, K. Janurová, T. V. Aa, T. J. Ashby, and V. Chupakhin. "HyperLoom: A Platform for Defining and Executing Scientific Pipelines in Distributed Environments". In: *Proceedings of the 9th Workshop and 7th Workshop on Parallel Programming and RunTime Management Techniques for Manycore Architectures and Design Tools and Architectures for Multicore Embedded Computing Platforms*. ACM. 2018, pp. 1–6.

[36] L. Cirrottola and A. Froehly. *ParMMG*. https://github.com/MmgTools/ParMmg. Version 1.4.0. Nov. 5, 2021.

[37] R. Codina. "Comparison of Some Finite Element Methods for Solving The Diffusion-Convection-Reaction Equation". In: *Computer methods in applied mechanics and engineering* 156.1-4 (1998), pp. 185–210. DOI: 10.1016/S0045-7825(97)00206-5.

[38] R. Codina. "Pressure stability in fractional step finite element methods for incompressible flows". In: *Journal of Computational Physics* 170.1 (2001), pp. 112–140. DOI: 10.1006/jcph.2001.6725.

[39] N. Collier, A.-L. Haji-Ali, F. Nobile, E. von Schwerin, and R. Tempone. "A continuation Multilevel Monte Carlo algorithm". In: *BIT Numerical Mathematics* (2014), pp. 1–34.

[40] *Cordis EU Research Results Horizon 2020 - ExaQUte*. https://cordis.europa.eu/project/id/800898. Accessed: 2022-09-12.

[41] C. Dapogny, C. Dobrzynski, and P. Frey. "Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems". In: *Journal of computational physics* 262 (2014), pp. 358–378.

[42] C. Dapogny, C. Dobrzynski, P. Frey, and A. Froehly. *MMG*. Version 5.6.0. Nov. 5, 2021.

[43] M. Davari, R. Rossi, P. Dadvand, I. Lopez, and R. Wüchner. "A cut finite element method for the solution of the full-potential equation with an embedded wake". In: *Computational Mechanics* 63.5 (2019), pp. 821–833.

[44] C. De Boor. *A Practical Guide to Splines*. Vol. 27. Applied Mathematical Sciences. Springer-Verlag New York, 1978.

[45] G. Detommaso, T. Dodwell, and R. Scheichl. "Continuous level Monte Carlo and sample-adaptive model hierarchies". In: *SIAM/ASA Journal on Uncertainty Quantification* 7.1 (2019), pp. 93–116.

[46] D. Drzisga, B. Gmeiner, U. Rude, R. Scheichl, and B. Wohlmuth. "Scheduling massively parallel multigrid for multilevel Monte Carlo methods". In: *SIAM Journal on Scientific Computing* 39.5 (2017), S873–S897.

[47] E. A. Ermakova, M. A. Panteleev, and E. E. Shnol. "Blood coagulation and propagation of autowaves in flow". In: *Pathophysiology of haemostasis and thrombosis* 34.2-3 (2005), pp. 135–142.

[48] W. Fang and M. B. Giles. "Multilevel Monte Carlo method for ergodic SDEs without contractivity". In: *Journal of Mathematical Analysis and Applications* 476.1 (2019), pp. 149–176.

[49] R. FitzHugh. "Impulses and physiological states in theoretical models of nerve membrane". In: *Biophysical journal* 1.6 (1961), pp. 445–466.

[50] P.-J. Frey and F. Alauzet. "Anisotropic mesh adaptation for CFD computations". In: *Computer methods in applied mechanics and engineering* 194.48-49 (2005), pp. 5068–5082.

[51] S. Ganesh, Q. Ayoul-Guilmard, and F. Nobile. *Report on the calculation of stochastic sensitivities*. Deliverable 6.2. Version 1.1. ExaQUte consortium, May 30, 2019. 18 pp. DOI: 10.23967/exaqute.2021.2.025.

[52] S. Ganesh and F. Nobile. “Gradient-based optimisation of the conditional-value-at-risk using the multi-level Monte Carlo method”. In: *arXiv preprint arXiv:2210.03485* (2022).

[53] S. Garreis, T. M. Surowiec, and M. Ulbrich. “An interior-point approach for solving risk-averse PDE-constrained optimization problems with coherent risk measures”. In: *SIAM Journal on Optimization* 31.1 (2021), pp. 1–29.

[54] G. Geraci, M. Eldred, and G. Iaccarino. “A multifidelity control variate approach for the multilevel Monte Carlo technique”. In: *Center for Turbulence Research Annual Research Briefs* (2015), pp. 169–181.

[55] G. Geraci, M. S. Eldred, and G. Iaccarino. “A multifidelity multilevel Monte Carlo method for uncertainty propagation in aerospace applications”. In: *19th AIAA non-deterministic approaches conference*. 2017, p. 1951.

[56] M. B. Giles. “Multilevel Monte Carlo methods”. In: *Acta Numerica* 24 (2015), pp. 259–328.

[57] M. B. Giles. “Multilevel Monte Carlo path simulation”. In: *Operations Research* 56.3 (2008), pp. 607–617.

[58] M. B. Giles and A.-L. Haji-Ali. “Multilevel nested simulation for efficient risk estimation”. In: *SIAM/ASA Journal on Uncertainty Quantification* 7.2 (2019), pp. 497–525.

[59] M. B. Giles, D. J. Higham, and X. Mao. “Analysing multi-level Monte Carlo for options with non-globally Lipschitz payoff”. In: *Finance and Stochastics* 13.3 (2009), pp. 403–413.

[60] M. B. Giles, T. Nagapetyan, and K. Ritter. “Multilevel Monte Carlo approximation of distribution functions and densities”. In: *SIAM/ASA Journal on Uncertainty Quantification* 3.1 (2015), pp. 267–295.

[61] M. B. Giles and B. J. Waterhouse. “Multilevel quasi-Monte Carlo path simulation”. In: *Advanced Financial Modelling, Radon Series on Computational and Applied Mathematics* 8 (2009), pp. 165–181.

[62] M. B. Giles, T. Nagapetyan, and K. Ritter. “Adaptive Multilevel Monte Carlo Approximation of Distribution Functions”. In: *arXiv preprint arXiv:1706.06869* (2017).

[63] A. A. Gorodetsky, G. Geraci, M. S. Eldred, and J. D. Jakeman. “A generalized approximate control variate framework for multifidelity uncertainty quantification”. In: *Journal of Computational Physics* 408 (2020), p. 109257.

[64] W. Gou. *Estimating value-at-risk using multilevel Monte Carlo maximum entropy method*. 2016.

[65] N. K. Govil and R. N. Mohapatra. “Markov and Bernstein type inequalities for polynomials”. In: *J. Inequal. Appl.* 3.4 (1999), pp. 349–387. DOI: 10.1155/S1025583499000247.

[66] A.-L. Haji-Ali, F. Nobile, L. Tamellini, and R. Tempone. “Multi-index stochastic collocation convergence rates for random PDEs with parametric regularity”. In: *Foundations of Computational Mathematics* 16.6 (2016), pp. 1555–1605.

[67] A.-L. Haji-Ali, F. Nobile, L. Tamellini, and R. Tempone. "Multi-index stochastic collocation for random PDEs". In: *Computer Methods in Applied Mechanics and Engineering* 306 (2016), pp. 95–122.

[68] A.-L. Haji-Ali, F. Nobile, and R. Tempone. "Multi-index Monte Carlo: when sparsity meets sampling". In: *Numerische Mathematik* 132.4 (2016), pp. 767–806.

[69] C. A. Hall and W. W. Meyer. "Optimal error bounds for cubic spline interpolation". In: *Journal of Approximation Theory* 16.2 (1976), pp. 105–122.

[70] S. Heinrich. "Multilevel Monte Carlo methods". In: *International Conference on Large-Scale Scientific Computing*. Springer. (2001), pp. 58–67.

[71] L. Herrmann and C. Schwab. "Multilevel quasi-Monte Carlo integration with product weights for elliptic PDEs with lognormal coefficients". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 53.5 (2019), pp. 1507–1552.

[72] A. L. Hodgkin and A. F. Huxley. "A quantitative description of membrane current and its application to conduction and excitation in nerve". In: *The Journal of physiology* 117.4 (1952), pp. 500–544.

[73] L. J. Hong and G. Liu. "Monte Carlo estimation of value-at-risk, conditional value-at-risk and their sensitivities". In: *Proceedings of the 2011 Winter Simulation Conference (WSC)*. IEEE. 2011, pp. 95–107.

[74] L. J. Hong and G. Liu. "Simulating sensitivities of conditional value at risk". In: *Management Science* 55.2 (2009), pp. 281–293.

[75] P. E. Kloeden and E. Platen. *Numerical solution of stochastic differential equations*. Vol. 23. Stochastic Modelling and Applied Probability. Springer, 1992.

[76] A. Kodakkal, B. Keith, U. Khristenko, A. Apostolatos, K.-U. Bletzinger, B. Wohlmuth, and R. Wuechner. "Risk-averse design of tall buildings for uncertain wind conditions". In: *arXiv preprint arXiv:2203.12060* (2022).

[77] D. P. Kouri. "A multilevel stochastic collocation algorithm for optimization of PDEs with uncertain coefficients". In: *SIAM/ASA Journal on Uncertainty Quantification* 2.1 (2014), pp. 55–81.

[78] D. P. Kouri, M. Heinkenschloss, D. Ridzal, and B. G. van Bloemen Waanders. "A trust-region algorithm with adaptive stochastic collocation for PDE optimization under uncertainty". In: *SIAM Journal on Scientific Computing* 35.4 (2013), A1847–A1879.

[79] D. P. Kouri, M. Heinkenschloss, D. Ridzal, and B. G. van Bloemen Waanders. "Inexact objective function evaluations in a trust-region algorithm for PDE-constrained optimization under uncertainty". In: *SIAM Journal on Scientific Computing* 36.6 (2014), A3011–A3029.

[80] D. P. Kouri and A. Shapiro. "Optimization of PDEs with Uncertain Inputs". In: vol. 163. The IMA Volumes in Mathematics and its Applications. Springer, 2018, pp. 41–81.

[81] D. P. Kouri and T. M. Surowiec. "A primal–dual algorithm for risk minimization". In: *Mathematical Programming* 193.1 (2022), pp. 337–363.

[82] D. P. Kouri and T. M. Surowiec. "Epi-regularization of risk measures". In: *Mathematics of Operations Research* 45.2 (2020), pp. 774–795.

[83] D. P. Kouri and T. M. Surowiec. "Existence and optimality conditions for risk-averse PDE-constrained optimization". In: *SIAM/ASA Journal on Uncertainty Quantification* 6.2 (2018), pp. 787–815.

[84] D. P. Kouri and T. M. Surowiec. "Risk-averse PDE-constrained optimization using the conditional value-at-risk". In: *SIAM Journal on Optimization* 26.1 (2016), pp. 365–396.

[85] S. Krumscheid and F. Nobile. "Multilevel Monte Carlo approximation of functions". In: *SIAM/ASA Journal on Uncertainty Quantification* 6.3 (2018), pp. 1256–1293.

[86] S. Krumscheid, F. Nobile, and M. Pisaroni. "Quantifying uncertain system outputs via the multilevel Monte Carlo method—Part I: Central moment estimation". In: *Journal of Computational Physics* 414 (2020), p. 109466.

[87] F. Kuo, R. Scheichl, C. Schwab, I. Sloan, and E. Ullmann. "Multilevel quasi-Monte Carlo methods for lognormal diffusion problems". In: *Mathematics of Computation* 86.308 (2017), pp. 2827–2860.

[88] J. Lang, R. Scheichl, and D. Silvester. "A fully adaptive multilevel stochastic collocation strategy for solving elliptic PDEs with random data". In: *Journal of Computational Physics* 419 (2020), p. 109692.

[89] C. Lim, H. D. Sherali, and S. Uryasev. "Portfolio optimization by minimizing conditional value-at-risk via nondifferentiable optimization". In: *Computational Optimization and Applications* 46.3 (2010), pp. 391–415.

[90] W. Linde and A. Pietsch. "Mappings of Gaussian cylindrical measures in Banach spaces". In: *Theory of Probability & Its Applications* 19.3 (1975), pp. 445–460.

[91] A. Lobanov and T. Starozhilova. "The effect of convective flows on blood coagulation processes". In: *Pathophysiology of haemostasis and thrombosis* 34.2-3 (2005), pp. 121–134.

[92] A. Logg, K.-A. Mardal, and G. Wells. *Automated solution of differential equations by the finite element method: The FEniCS book*. Vol. 84. Springer Science & Business Media, 2012.

[93] *MareNostrum4 (2017) System Architecture*. https://www.bsc.es/marenostrum/marenostrum/technical-information. Accessed: 2022-10-03.

[94] M. Martin, F. Nobile, and P. Tsilifis. "A multilevel stochastic gradient method for PDE-constrained optimal control problems with uncertain parameters". In: *arXiv preprint arXiv:1912.11900* (2019).

[95] V. Mataix, P. Bucher, R. Zorrilla, R. Rossi, J. Cotela, J. M. Carbonell, M. A. Celigueta, T. Teschemacher, A. Cornejo, C. Roig, G. Casas, M. Masó, S. Warnakulasuriya, M. Núñez, P. Dadvand, S. Latorre, I. Pouplana, J. Irazábal, F. Arrufat, R. Tosi, A. Ghantasala, P. Wilson, D. Baumgaertner, B. Chandra, A. Franci, A. Geiser, K. Bernd, I. Lopez, and J. Gárate. *Kratos Multiphysics*. Version 9.0. Nov. 2021. DOI: 10.5281/zenodo.3234644.

[96] S. P. Meyn and R. L. Tweedie. "Stability of Markovian processes III: Foster–Lyapunov criteria for continuous-time processes". In: *Advances in Applied Probability* 25.3 (1993), pp. 518–548.

[97] J. Nagumo, S. Arimoto, and S. Yoshizawa. "An active pulse transmission line simulating nerve axon". In: *Proceedings of the IRE* 50.10 (1962), pp. 2061–2070.

[98] B. Nishida and M. Drela. "Fully simultaneous coupling for three-dimensional viscous/inviscid flows". In: *Proceedings of the 13th Applied Aerodynamics Conference*. AIAA, 1995, pp. 355–361. DOI: 10.2514/6.1995-1806.

[99] F. Nobile, R. M. Badia, J. Ejarque, L. Cirrottola, A. Froehly, B. Keith, A. Kodakkal, M. Núñez, C. Roig, R. Tosi, R. Rossi, C. Soriano, S. Ganesh, and Q. Ayoul-Guilmard. *Final public release of the solver*. Deliverable 1.4. Version 1.0. ExaQUte consortium, Nov. 30, 2020. DOI: 10.23967/exaqute.2021.2.009.

[100] M. Núñez, I. López, J. Baiges, and R. Rossi. "An embedded approach for the solution of the full potential equation with finite elements". In: *Computer Methods in Applied Mechanics and Engineering* 388 (2022), p. 114244.

[101] B. Peherstorfer, P. S. Beran, and K. E. Willcox. "Multifidelity Monte Carlo estimation for large-scale uncertainty propagation". In: *2018 AIAA Non-Deterministic Approaches Conference*. 2018, p. 1660.

[102] B. Peherstorfer, M. Gunzburger, and K. Willcox. "Convergence analysis of multifidelity Monte Carlo estimation". In: *Numerische Mathematik* 139.3 (2018), pp. 683–707.

[103] B. Peherstorfer, K. Willcox, and M. Gunzburger. "Optimal model management for multifidelity Monte Carlo estimation". In: *SIAM Journal on Scientific Computing* 38.5 (2016), A3163–A3194.

[104] J. Peng. "Value at Risk and Tail Value at Risk in Uncertain Environment". In: *Proceedings of the 8th International Conference on Information and Management Sciences*. 2009, pp. 787–793.

[105] M. Pisaroni, F. Nobile, and P. Leyland. "A Continuation Multi Level Monte Carlo (C-MLMC) method for uncertainty quantification in compressible inviscid aerodynamics". In: *Computer Methods in Applied Mechanics and Engineering* 326 (2017), pp. 20–50.

[106] M. Pisaroni, F. Nobile, and P. Leyland. "Continuation Multilevel Monte Carlo Evolutionary Algorithm for Robust Aerodynamic Shape Design". In: *Journal of Aircraft* 56.2 (2019), pp. 771–786.

[107] D. Quagliarella. "Value-at-risk and conditional value-at-risk in optimization under uncertainty". In: *Uncertainty Management for Robust Industrial Design in Aeronautics*. Springer, 2019, pp. 541–565.

[108] D. Quagliarella, E. Morales Tirado, and A. Bornaccioni. "Risk measures applied to robust aerodynamic shape design optimization". In: *Flexible Engineering Toward Green Aircraft*. Springer, 2020, pp. 153–168.

[109] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical Mathematics*. Vol. 37. Texts in Applied Mathematics. Springer Science & Business Media, 2007.

[110] P. Robbe, D. Nuyens, and S. Vandewalle. "A multi-index quasi–Monte Carlo algorithm for lognormal diffusion problems". In: *SIAM Journal on Scientific Computing* 39.5 (2017), S851–S872.

[111] P. Robbe, D. Nuyens, and S. Vandewalle. "Multi-Index Quasi-Monte Carlo: an algorithm for simulating PDEs with random coefficients". In: *International Conference on Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing (MCQMC2016), Date: 2016/08/14-2016/08/19, Location: Stanford, USA*. 2016.

[112] R. T. Rockafellar and J. O. Royset. "On buffered failure probability in design and optimization of structures". In: *Reliability engineering & system safety* 95.5 (2010), pp. 499–510.

[113] R. T. Rockafellar and S. Uryasev. "Conditional value-at-risk for general loss distributions". In: *Journal of banking & finance* 26.7 (2002), pp. 1443–1471.

[114] R. T. Rockafellar, S. Uryasev, et al. "Optimization of conditional value-at-risk". In: *Journal of risk* 2 (2000), pp. 21–42.

[115] A. Ruszczyński and A. Shapiro. "Optimization of convex risk functions". In: *Mathematics of operations research* 31.3 (2006), pp. 433–452.

[116] D. Schaden and E. Ullmann. "Asymptotic analysis of multilevel best linear unbiased estimators". In: *SIAM/ASA Journal on Uncertainty Quantification* 9.3 (2021), pp. 953–978.

[117] D. Schaden and E. Ullmann. "On multilevel best linear unbiased estimators". In: *SIAM/ASA Journal on Uncertainty Quantification* 8.2 (2020), pp. 601–635.

[118] C. Schillings. "Optimal aerodynamic design under uncertainties". In: *Thesis* (2011).

[119] C. Schillings and V. Schulz. "On the influence of robustness measures on shape optimization with stochastic uncertainties". In: *Optimization and Engineering* 16.2 (2015), pp. 347–386.

[120] V. Schulz and C. Schillings. "Optimal aerodynamic design under uncertainty". In: *Management and Minimisation of Uncertainties and Errors in Numerical Aerodynamics*. Springer, 2013, pp. 297–338.

[121] D. W. Scott. "On optimal and data-based histograms". In: *Biometrika* 66.3 (1979), pp. 605–610.

[122] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on stochastic programming: modeling and theory*. SIAM, 2009.

[123] S. Taverniers and D. M. Tartakovsky. "Estimation of distributions via multilevel Monte Carlo with stratified sampling". In: *Journal of Computational Physics* 419 (2020), p. 109572.

[124] A. L. Teckentrup, P. Jantsch, C. G. Webster, and M. Gunzburger. "A multilevel stochastic collocation method for partial differential equations with random input data". In: *SIAM/ASA Journal on Uncertainty Quantification* 3.1 (2015), pp. 1046–1074.

[125] E. Tejedor, Y. Becerra, G. Alomar, A. Queralt, R. M. Badia, J. Torres, T. Cortes, and J. Labarta. "PyCOMPSs: Parallel computational workflows in Python". In: *International Journal of High Performance Computing Applications* 31.1 (2017), pp. 66–82. DOI: 10.1177/1094342015594678.

[126] R. J. Tibshirani and B. Efron. *An introduction to the bootstrap*. Vol. 57. Chapman and Hall New York, 1993, pp. 1–436.

[127] R. Tyrrell Rockafellar and J. O. Royset. "Engineering decisions under risk averseness". In: *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering* 1.2 (2015), p. 04015003.

[128] S. Uryasev and R. T. Rockafellar. "Conditional value-at-risk: optimization approach". In: *Stochastic optimization: algorithms and applications*. Springer, 2001, pp. 411–435.

[129] M. Uzunca, T. Küçükseyhan, H. Yücel, and B. Karasözen. "Optimal control of convective FitzHugh–Nagumo equation". In: *Computers and Mathematics with Applications* 73.9 (2017), pp. 2151–2169.

[130] A. Van Barel and S. Vandewalle. "Robust optimization of PDEs with random coefficients using a multilevel Monte Carlo method". In: *SIAM/ASA Journal on Uncertainty Quantification* 7.1 (2019), pp. 174–202.

[131] F. Wagner, J. Latz, I. Papaioannou, and E. Ullmann. "Multilevel sequential importance sampling for rare event estimation". In: *SIAM Journal on Scientific Computing* 42.4 (2020), A2062–A2087.

[132] *Website - Exascale Quantification of Uncertainties for Technology and Science Simulation*. exaqute.eu. Accessed: 2022-09-12.

Sundar **Ganesh**

Doctoral Candidate · EPFL

(+41) 779952523 | sundar.ganesh@protonmail.com | sganesh21 | sundar.ganesh93

Education

Ph. D. in Mathematics — *Lausanne, Switzerland*

École Polytechnique Fédérale de Lausanne (EPFL) — *Oct. 2018 - Nov. 2022 (Expected)*

- Thesis supervisor and group: Prof. Fabio Nobile, Chair for Scientific Computing and Uncertainty Quantification (Link)

M. Sc. (Hons.) in Simulation Sciences — *Aachen, Germany*

Rheinisch-Westfaelische Technische Hochschule (RWTH) — *Oct. 2016 - Sep. 2018*

- Cumulative GPA : 1.2/1.0. Graduated with honours
- Awarded the Springorum Medal for graduating top 10% of the faculty

M. Sc. (Hons.) in Physics and B. E. (Hons.) in Mechanical Engineering — *Goa, India*

Birla Institute of Technology and Science (BITS), Pilani — *Aug. 2011 - May. 2016*

- Cumulative GPA: 8.29/10.0. Graduated with honours in both degree programs

Skills

Programming Languages	Python, C, C++
Software	Latex, MPI, OpenMP, Matlab, OpenFOAM, Fenics, Paraview, ANSYS Fluent, ICEM, CATIA, Pro-E
Operating Systems	Ubuntu, Debian, Mac OS, Windows
Languages	English (Native), Tamil (Native), Hindi (B2), German (B1/B2), French (A2/B1)

Research Experience

Doctoral Thesis Research (See Publications) — *Lausanne, Switzerland*

Chair for Scientific Computing and Uncertainty Quantification, EPFL — *Oct. 2018 - Present*

- Doctoral thesis title : *Multi-Level Monte Carlo (MLMC) Methods for Uncertainty Quantification and Risk-Averse Optimization*
- Researched exascale uncertainty quantification as a part of the EU Horizon 2020 “ExaQUte” Project (Link), a highly applied and collaborative international research endeavour between 4 universities, 3 national research centers and 1 private company all based in Europe.
- Co-founded and developed the XMC library (Link, 456 views, 46 downloads), a Python library for hierarchical Monte Carlo methods
- Developed and applied novel Monte Carlo algorithms to problems in finance, climate modeling, civil and aerospace engineering
- Conducted high-performance parallelized simulations on the supercomputers of the Barcelona Supercomputing Center and the Czech National Supercomputing Center
- Supervised two master students on semester projects on MLMC methods
- Aided in the design of projects for a master-level course on stochastic simulations, Monte Carlo methods and Markov Chains
- Taught multiple courses on OpenMP, MPI, C++ and other numerical methods

Master Thesis Research — *Lausanne, Switzerland*

Chair of Computational Mathematics and Simulation Science, EPFL — *Apr. 2018 - Sep. 2018*

- Master thesis title : *Stochastic Sampling of Maximum Entropy Distribution*
- Implemented combination of stochastic gradient descent with Ito processes to generate maximum entropy samples directly without acceptance-rejection sampling

Research Assistant (See Publications) — *Aachen, Germany*

Center for Computational Engineering and Science, RWTH — *Jul. 2017 - Mar. 2018*

- Studied variance reduction for stochastic particle simulations
- Implemented and validated novel correlated particle method for the Direct Simulation Monte Carlo method

Master Thesis Research (See Publications) — *Goa, India*

Computational Fluid Dynamics Laboratory, BITS-Pilani — *Jan. 2016 - May. 2016*

- Master thesis title: *Passive Flow Control in Aggressive Inter-Turbine diffusers*
- Conducted parametric study simulations using OpenFOAM computational fluid dynamics code. Found critical flow-separation point for experimental duct found in literature

Honors & Awards

2019	**Awardee**, Springorum Commemorative Coin for Distinguished Academic Performance. (Link)	*Aachen, Germany*
2021	**Finisher**, La Tour Genève Full Olympic Triathlon	*Geneva, Switzerland*
2016	**2nd place**, My Thesis in 180 Seconds, BITS-Pilani	*Goa, India*

Selected Publications

Ganesh, S., Nobile, F. (2022). ***Gradient Based Optimisation of the Conditional-Value-at-Risk using the Multi-Level Monte Carlo Method.***

Manuscript in preparation.

Ayoul-Guilmard, Q., Ganesh, S., *et al.* (2022). ***Quantifying Uncertain System Outputs via The Multilevel Monte Carlo Method — Distribution and Robustness Measures.***

Available at arXiv:2208.07252. Submitted to the International Journal for Uncertainty Quantification.

Ganesh., S., *et al.* (2018-2022) ExaQUte list of peer-reviewed deliverable reports.

Open Access Repository of the ExaQUte project: Deliverables (Link)

Ganesh, S., & Gorji, H. (2018). ***A Variance Reduction Method for DSMC Using Correlated Process.***

Proceedings of the 3rd European Conference on Non-Equilibrium Gas Flows (pp. 443-446).

Selected Presentations and Conferences

Apr. 2022	SIAM Conference on Uncertainty Quantification	*Atlanta, USA*
Jul. 2021	US National Congress on Computational Mechanics	*Chicago, USA*
Apr. 2021	Platform for Advanced Scientific Computing Conference	*Geneva, Switzerland*
Mar. 2021	SIAM conference on Computational Science and Engineering	*Virtual*
Aug. 2020	International Conference in Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing	*Oxford, UK*
Feb. 2018	MicroFluidics and Non-Equilibrium Gas Flows Conference	*Strasbourg, France*

Workshops and Summer Schools

Jun. 2022	Management of Innovation and Technology Transfer, EPFL	*Lausanne, Switzerland*
May. 2022	Machine Learning Methods in Computational Finance, King Abdullah University of Science and Technology	*Virtual, Saudi Arabia*
Sep. 2019	Model Order Reduction Summer School, Eindhoven University of Technology	*Eindhoven, Netherlands*
Mar. 2019	Topics in the Theory of Markov Processes, EPFL	*Lausanne, Switzerland*

Society Memberships

Society for Industrial and Applied Mathematics (SIAM) *USA*

Member *Jan. 2017 - Present*

Extracurricular Activity

Jun 2020 - Present	**Rock climbing, mountaineering and alpinism**, Club Montagne, EPFL	*Lausanne, Switzerland*
Aug. 2021 - Present	**Independent music artist**, SoundCloud and Spotify	*Lausanne, Switzerland*
Aug. 2011 - May. 2016	**Music performance and concert organization**, Music Society, BITS-Pilani	*Goa, India*
Aug. 2011 - May. 2016	**Concert/competition organizer**, Department of Pro-Nights and SeaRock, BITS-Pilani	*Goa, India*
Aug. 2012 - May. 2013	**Radio-controlled flights and drones**, Aerodynamics Club, BITS-Pilani	*Goa, India*