ExaQUte

Exascale **Q**uantification of **U**ncertainties for **Te**chnology and Science Simulation

D6.5 Report on stochastic optimisation for wind engineering

Document information table

Contract number:	800898
Project acronym:	ExaQUte
Project Coordinator:	CIMNE
Document Responsible Partner:	EPFL
Deliverable Type:	Report
Dissemination Level:	PUblic
Related WP & Task:	WP6 task 6.5
Status:	Final version

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No **800898**

Authoring

Prepared by EPFL, CIMNE, TUM				
Authors	Partner	Modified	Version	Comments
Fabio Nobile	EPFL			Expert and reviewer
Quentin Ayoul-Guilmard				co-author, coordinator
Sundar Ganesh				co-authors
Marc Núñez	CIMNE			
Anoop Kodakkal	TUM			

Change log

Versions	Modified pages/sections	Comments
1.0	All	Submitted version

Approval

Approved by EPFL and CIMNE				
	Name	Partner	Date	OK
Task leader	Fabio Nobile	EPFL	2021-11-29	✓
Reviewer	Javier Principe	UPC	2021-11-30	✓

Executive summary

This report presents the latest methods of optimisation under uncertainties investigated in the ExaQUte project, and their applications to problems related to civil and wind engineering. The measure of risk throughout the report is the conditional value at risk.

First, the reference method is presented: the derivation of sensitivities of the risk measure; their accurate computation; and lastly, a practical optimisation algorithm with adaptive statistical estimation. Second, this method is directly applied to a non-linear relaxation oscillator (FitzHugh–Nagumo model) with numerical experiments to demonstrate its performance. Third, the optimisation method is adapted to the shape optimisation of an airfoil and illustrated by a large-scale experiment on a computing cluster. Finally, the benchmark of the shape optimisation of a tall building under a turbulent flow is presented, followed by an adaptation of the optimisation method.

All numerical experiments showcase the open-source software stack of the ExaQUte project for large-scale computing in a distributed environment.

Contents

1 Introduction

1.1 ExaQUte context

This report presents research outcomes from the ExaQUte project ('exascale quantification of uncertainties for technology and science simulation'), part of the European Union's research programme Horizon 2020[1]. This project researches methods for the optimisation of robust optimisation of the shape of tall structure to withstand uncertain wind, which are then implemented in tools apt to leverage large-scale distributed computing. This endeavour encompasses diverse scientific fields: uncertainty quantification (UQ), optimisation under uncertainties (OUU), computational fluid dynamics (CFD), high-performance computing (HPC), et cætera (etc.). The work presented here belongs to the work package 'Optimisation under uncertainties', id est (i.e.) in this case on the search for an engineering design robust against the risk presented by uncertain parameters. This report presents the latest research developments in this direction with a focus on the application to problems from civil and wind engineering. It concludes a series of four successive reports.

deliverable 6.2 described first-order optimality conditions for the type of problems of OUU considered in ExaQUte. The computation of sensitivities by adjoint calculus was presented for several risk measures of interest, along with their challenges and possible solutions.

deliverable 6.3 proposed various gradient-descent method and stochastic approximations for the optimisation of different risk measures. Their possible combinations with multi-level Monte Carlo (MLMC) methods were presented. For the case of the conditional value at risk (CVaR), the issue of lack of regularity was discussed and two regularisation strategies were considered.

deliverable 6.4 introduced several benchmarks, with a focus on unsteady problems and engineering applications. The adaptation of the optimisation method identified in the previous deliverable was discussed for each benchmark, along with the associated challenges and recommended solutions. That report laid the foundation of the work present in the current document.

Furthermore, two concurrent reports relate closely to the methods and results presented here: deliverable 5.5 reports on UQ for the same kind of problems, including methods necessary to the optimisation discussed here; deliverable 7.4 presents a major benchmark of OUU which motivates much of the research in this report.

[1]Funding received under grant agreement 800898.

As part of its mission, the ExaQUte project maintains several tools in which the methods developed are implemented. These constitute an interoperable software stack whose source code is publicly available under open licenses, as are the scientific reports. It is summarised below, and described in more details in deliverable 1.4.

MMG+ParMMG is a set of tools for simplicial remeshing developed by INRIA. The adaptive mesh refinement is performed with control of the interpolation error based on a given metric (C. Dapogny et al. 2014). ParMMG is an extension developed during the ExaQUte project by Cirrottola and Froehly 2019 to perform this remeshing across several processors in parallel, via the Message Passing Interface (MPI) standard. The concurrent report deliverable 2.5 presents the latest development on this tool.

Kratos Multiphysics is the solver for the CFD simulation, whose development is lead by CIMNE (Pooyan Dadvand et al. 2010; P. Dadvand et al. 2013). It also supports parallel computation through MPI. Its latest release is reported in deliverable 1.4.

COMPSs and Quake are the two alternative frameworks for distributed computing, accessible via a common application programming interface (API) developed for the ExaQUte project (see Böhm and Ejarque 2021). COMPSs (Lordan et al. 2013; Rosa M. Badia et al. 2015; Tejedor et al. 2017) is developed by the Barcelona Supercomputing Center while Quake is from IT4Innovations. Their latest progress and benchmarks are in the concurrent report deliverable 4.5.

XMC is a library of hierarchical Monte Carlo (MC) methods for UQ. It was created during the ExaQUte project as deliverable 5.1 and deliverable 5.2, and has been developed as a collaborative effort within ExaQUte lead by EPFL.

1.2 Structure of the report

This report is split into four parts, which relate to and follow in direct succession § 2–5 from deliverable 6.4, respectively. They are followed by a conclusion of the report and of the whole work package ‘Optimisation under uncertainties’.

Section 2 discusses the optimisation of the chosen risk measure, the CVaR. It presents the derivation of its sensitivities for use in a gradient-based optimisation method. A MLMC estimator is proposed for this task, along with associated error estimators and the calibration thereof. This eventually leads to a practical optimisation algorithm.

Section 3 demonstrates the application of the method from section 2 to the FitzHugh–Nagumo oscillator: an extension of the Van der Pol oscillator previously used as a simple benchmark in ExaQUte. Its adjoint problem is formulated, from which the parametric sensitivities of the primal solution are derived. The reliability and performance of the algorithm are assessed and compared to theoretical expectations.

Section 4 discusses the constrained shape optimisation of the CVaR of the lift coefficient of an airfoil under uncertain boundary conditions – an application announced in deliverable 6.4. A potential-flow model is used for the flow problem and its adjoint is presented in order to derive the sensitivities of the lift coefficient with respect to the shape of the airfoil. Then the optimisation itself is demonstrated on a computing cluster, using adaptive mesh refinement for the evolving geometry.

Section 5 addresses the case of turbulent fluid flows. It presents an important benchmark of ExaQUte whose numerical experiments are reported in deliverable 7.4: the shape optimisation of a tall building under turbulent wind to minimise the CVaR of the mechanical moment at its base. Given the challenges raised by the chaotic behaviour of the turbulent flow, a different methodology from the previous sections is proposed.

Finally, section 6 gives some conclusions and perspective on the research presented here and on the whole package.

1.3 Common notions and notation

The field OUU is broad and varied. To focus the research effort, the ExaQUte project decided from its inception to target robust shape optimisation constrained by a partial differential equation (PDE) with uncertain loading or boundary conditions. The first methodological decision was to research gradient-based methods. Given that shape optimisation is prone to high-dimensional design spaces, adjoint calculus is favoured to compute the gradient whenever available. The quantification of the risk associated with a given design is a significant choice from both the engineering and the mathematical point of views. The CVaR has been selected for its ability to control failure states and its suitable mathematical properties, exempli gratia (e.g.) *coherence* (see deliverable 6.3, § 3); we refer to R. T. Rockafellar and Royset 2015 for a detailed review of risk measures pertinent to engineering risk-averse decisions.

We introduce below several notions pertinent to this context and used throughout the report. In all that follows, $(\Omega, \mathcal{F}, \mathbb{P})$ will denote a probability space.

DEFINITION 1 (Conditional value at risk). Let $X \in \mathrm{L}^1(\Omega, \mathbb{R})$ an integrable, real-valued, random variable. Let $\tau \in]0,1[$. The value at risk (VaR) of significance[2] τ of X is defined as

$$\mathrm{VaR}_\tau(X) := \inf\{t \in \mathbb{R} : \mathbb{P}(X \leqslant t) \geqslant \tau\} \tag{1.1}$$

[2]Sometimes called 'value at risk $1-\tau$'.

In this report, we assume that $\mathbb{P}(X = \text{VaR}_\tau(X)) = 0$; see Uryasev et al. 2010 for a more general context. We define the conditional value at risk (CVaR) of significance τ of X as the following expectation, conditional on the value at risk (VaR):

$$\text{CVaR}_\tau(X) := \mathbb{E}(X|X \geqslant \text{VaR}_\tau(X)) = \inf\{\mathbb{E}(\phi(X,s)) : s \in \mathbb{R}\}. \tag{1.2}$$

$$\text{with} \quad \phi(X,s) := s + \frac{(X-s)^+}{1-\tau} \qquad \text{and} \quad \forall r \in \mathbb{R},\ (r)^+ := \max(r,0).$$

Assuming that the cumulative distribution function (CDF) of X is continuous, the infima in (1.1)–(1.2) are minima and $\text{CVaR}_\tau(X) = \mathbb{E}(\phi(X, \text{VaR}_\tau(X)))$, from T. R. Rockafellar and Uryasev 2000, theorem 1. ♢

Let Z be a design space and $Q : Z \to \mathbb{R}$ be a quantity of interest (QoI) whose risk is to be minimised. Using the CVaR of significance $\tau \in]0,1[$ as risk measure, the optimal design $z_\star$ is defined as

$$z_\star := \operatorname{argmin}\{\text{CVaR}_\tau(Q(z))\}. \tag{1.3}$$

Recalling (1.2), (1.3) appears as a nested optimisation problem. Therefore we consider the more practical formulation given in problem 1 below, which ibid., theorem 2 proved to yield a solution of (1.3).

Problem 1 (General CVaR optimisation with PDE constraint). We define the optimal design $z_\star$ as part of the optimal solution

$$(z_\star, s_\star) := \operatorname{argmin}\{J(z,s) := \mathbb{E}(\phi(Q(z),s)) : (z,s) \in Z \times \mathbb{R}\},$$

where $Q(z)$ implicitly depends on the solution $u(\omega,z)$ of a PDE affected by random effects, expressed in abstract form as the constraint

$$P(u(\omega,z),z,\omega) = 0 \qquad \text{for almost every (a.e.) } \omega \in \Omega. \tag{1.4}$$

We assume that the random solution $u(\omega,z)$ is unique for a given design z, hence the notation $Q(z)$. However, computing a sample of $Q(z)$ generally requires to solve (1.4) for a given $\omega \in \Omega$. ♢

In the following sections, P will generally be a time-dependent PDE and the QoI sought will typically be temporal averages as defined next.

Definition 2 (Temporal average). Let $T \in \mathbb{R}$, $d \in \mathbb{N}$ and $\boldsymbol{f} : [0,T] \to \mathbb{R}^d$, a function of time. Let $a \in [0,T[$ and $b \in]a,T[$. The temporal average (or 'time average') of the function $\boldsymbol{f}$ from a to b is defined as

$$\langle \boldsymbol{f} \rangle_{a,b} := \frac{1}{b-a}\int_a^b \boldsymbol{f}.$$

This notation may be simplified as $\langle \boldsymbol{f} \rangle_b := \langle \boldsymbol{f} \rangle_{0,b}$ and $\langle \boldsymbol{f} \rangle := \langle \boldsymbol{f} \rangle_{0,T}$. ♢

Finally, for any pair $(a, b) \in \mathbb{R}^2$, we will note $[\![a, b]\!] := [a, b] \cap \mathbb{Z}$ the set of integers between a and b.

2 Sensitivity estimation and optimisation for the conditional value-at-risk

In deliverable 6.2, § 3.4, we presented a framework for the gradient-based optimisation for the CVaR. However, the resultant expression for the gradient required computing the expected value of a discontinuous function using MLMC methods. If applied directly, the performance of MLMC methods is severely degraded.

In this section, we utilise the idea of parametric expectations to overcome this issue and estimate the sensitivities necessary to carry out gradient-based optimisation of the CVaR. We recall here briefly some concepts around the estimation of parametric expectations using MLMC estimators. Parametric expectations are expectations of the form:

$$\Phi(\theta) := \mathbb{E}(\phi(Q,\theta)), \quad \theta \in \Theta \subset \mathbb{R},$$

where Q denotes the output QoI which, in the setting of this report, is the quantity to be optimised. For the remainder of this work, we define ϕ as in definition 1:

$$\phi(Q,\theta) := \theta + \frac{(Q-\theta)^+}{1-\tau},$$

where τ denotes a significance parameter. This form has the advantage that after estimating the function Φ and its derivatives

$$\Phi^{(d)}(\theta) := \frac{\partial^d}{\partial\theta^d}\mathbb{E}(\phi(Q,\theta)), \quad d \in \{0,1,2\},$$

the CDF $F_Q(\theta) = \mathbb{E}\big(\mathbb{1}_{Q\leqslant\theta}\big)$ and the probability density function (PDF) $f_Q(\theta) = F_Q^{(1)}(\theta)$, as well as the VaR q_τ and the CVaR c_τ of significance τ, can be obtained by simple post-processing.

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

The reader is referred to deliverable 5.5, § 2.1 for more details.

We presented in ibid. an MLMC estimator $\hat{\Phi}_L^{(d)}$ for $\Phi^{(d)}$. The estimator works by first constructing point-wise MLMC estimates on a uniform grid $\boldsymbol{\theta} = \{\theta_1, \dots, \theta_n\}$ in the interval Θ. The MLMC estimator $\hat{\Phi}_L$ for Φ at a point θ_j reads:

$$\hat{\Phi}_L(\theta_j) := \frac{1}{m_0}\sum_{i=1}^{m_0} \phi(Q_0^{(i,0)}, \theta_j) + \sum_{l=1}^{L}\frac{1}{m_l}\sum_{i=1}^{m_l}\left[\phi(Q_l^{(i,l)},\theta_j) - \phi(Q_{l-1}^{(i,l)},\theta_j)\right],$$

where $\{Q_l : l \in [\![0, L]\!]\}$ denote a sequence of approximations of Q with increasing accuracy and cost and $Q_l^{(i,l)}$ and $Q_{l-1}^{(i,l)}$ are realisations at two consecutive accuracy levels obtained for the same random event. In addition, $Q_l^{(i,l)}$ and $Q_{l'}^{(i',l')}$ are statistically independent whenever $i \neq i'$ or $l \neq l'$.

We then construct a MLMC estimator of the whole function $\Phi(\theta), \theta \in \Theta$ by interpolating over the point-wise estimates as below.

$$\hat{\Phi}_L = \mathcal{S}_n\Big(\hat{\Phi}_L(\boldsymbol{\theta})\Big),$$

where $\mathcal{S}_n$ denotes the cubic spline interpolation operator and $\hat{\Phi}_L(\boldsymbol{\theta})$ denotes the set of point-wise MLMC estimates:

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

The function derivative estimate denoted by $\hat{\Phi}_L^{(d)}$ is then obtained by computing the derivative of the resultant interpolated function:

$$\hat{\Phi}_L^{(d)} := \mathcal{S}_n^{(d)}\Big(\hat{\Phi}_L(\boldsymbol{\theta})\Big),$$

where the superscript d denotes the order of the derivative.

The error of the estimator is quantified using an estimator for the mean squared error (MSE) defined as follows:

$$\mathrm{MSE}\Big(\hat{\Phi}_L^{(d)}\Big) := \mathbb{E}\left(\left\|\Phi^{(d)} - \hat{\Phi}_L^{(d)}\right\|^2_{\mathrm{L}^\infty(\Theta)}\right).$$

We presented in deliverable 5.5 and Ayoul-Guilmard, Ganesh, Krumscheid et al. 2021, an adaptive strategy to select the parameters of the hierarchy, namely the number of interpolation points n, the number of levels L and the level-wise samples sizes m_l, such that the MLMC satisfies a given tolerance in a cost optimal manner. When optimally tuned, the complexity behaviour is dramatically improved compared to a simple Monte Carlo estimator.

2.1 Sensitivity estimation and gradient error

To demonstrate the relation between parametric expectations and sensitivity estimation, we first consider a CVaR minimisation problem as follows. Let $\omega \in \Omega$ denote an elementary random event and $z \in \mathcal{A} \subset \mathbb{R}^p$ denote the design variable. Lastly we denote by $Q(z, \omega) \in \mathbb{R}$ the value of the QoI to be minimised for a given design z and random event ω. We formulate the following OUU problem:

$$\begin{aligned} J^* &= \min_{z \in \mathcal{A} \subseteq Z}\left\{\min_{\theta \in \Theta}\left(\theta + \frac{\mathbb{E}((Q(z,\cdot) - \theta)^+)}{1-\tau}\right) + \kappa\|z - z_0\|^2_{\ell^2}\right\} \\ &= \min_{\substack{z \in \mathcal{A} \\ \theta \in \Theta}}\left\{J(z,\theta) := \Phi(\theta; z) + \kappa\|z - z_0\|^2_{\ell^2}\right\}, \end{aligned} \tag{2.1}$$

where z_0 denotes a preferred design and κ denotes a regularisation parameter. We have also combined the minimisation problem of the CVaR with the minimisation problem of the design as in deliverable 6.3 and deliverable 6.4. Since we aim to use gradient-based algorithms to solve the OUU problem, we compute the sensitivities with respect to z and θ as follows:

$$J_\theta(z,\theta) = 1 - \frac{\mathbb{E}\left(\mathbb{1}_{Q(z,\cdot)\geqslant\theta}\right)}{1-\tau},$$

and $\forall j \in [\![1,p]\!]$,

$$J_{z_j}(z,\theta) = \frac{\mathbb{E}\left(Q_{z_j}(z,\cdot)\mathbb{1}_{Q(z,\cdot)\geqslant\theta}\right)}{1-\tau} + 2\kappa(z_j - z_{0,j})$$

where $Q_{z_j}(z,\omega) \in \mathbb{R}$ denotes the sensitivity of the QoI with respect to the j^{th} design parameter z_j. One can directly estimate the above expectations using MLMC estimators. However, using MLMC to estimate the expected value of a discontinuous function can lead to sub-optimal performance of the MLMC method, as shown in Krumscheid and Nobile 2018.

We propose the following alternative, which consists of interpreting J_θ and J_{z_j} as derivatives of parametric expectations, videlicet (viz.):

$$J_\theta(z,\theta) = \Phi^{(1)}(\theta;z),$$

and $\forall j \in [\![1,d]\!]$,

$$J_{z_j}(z,\theta) = \Psi_j^{(1)}(\theta;z) + 2\kappa(z_j - z_{0,j}),$$

where we introduce the parametric expectations $\Phi, \Psi_j : \Theta \times \mathcal{A} \to \mathbb{R}$:

$$\Phi(\theta;z) := \theta + \frac{\mathbb{E}((Q(z,\cdot)-\theta)^+)}{1-\tau} =: \mathbb{E}(\phi(Q(z,\cdot),\theta))$$

and $\forall j \in [\![1,p]\!]$

$$\Psi_j(\theta;z) := \mathbb{E}\left(\frac{(Q(z,\cdot)-\theta)^+ Q_{z_j}(z,\cdot)}{1-\tau}\right) =: \mathbb{E}\left(\psi(\theta, Q(z,\cdot), Q_{z_j}(z,\cdot))\right).$$

This form has several advantages for both the MLMC algorithm and gradient-based optimisation algorithms, which are described in detail in deliverable 5.5 and in Ayoul-Guilmard, Ganesh, Krumscheid et al. 2021.

The challenge then lies in the accurate estimation of $\Phi^{(1)}$ and $\Psi_j^{(1)}$, since these are directly related to accurately estimating the gradient $\nabla J = (J_\theta, J_{z_1}, \ldots, J_{z_p})$.

More precisely, we denote by $\hat{\Phi}^{(1)}$ and $\hat{\Psi}_j^{(1)}$ the estimates of $\Phi^{(1)}$ and $\Psi_j^{(1)}$ obtained through a MC or MLMC estimator. We denote by $\nabla \hat{J}$ the approximation to ∇J obtained from $\hat{\Phi}^{(1)}$ and $\hat{\Psi}_j^{(1)}$. We have that:

$$\begin{aligned}\left\|\nabla J(z,\theta) - \nabla \hat{J}(z,\theta)\right\|_{\ell^2}^2 &= \left(\Phi^{(1)}(\theta;z) - \hat{\Phi}^{(1)}(\theta;z)\right)^2 \\ &\quad + \sum_{j=1}^{d}\left(\Psi_j^{(1)}(\theta;z) - \hat{\Psi}_j^{(1)}(\theta;z)\right)^2 \\ &\leqslant \|\Phi^{(1)}(\cdot;z) - \hat{\Phi}^{(1)}(\cdot;z)\|_{\mathrm{L}^\infty(\Theta)}^2 \\ &\quad + \sum_{j=1}^{d}\|\Psi_j^{(1)}(\cdot;z)) - \hat{\Psi}_j^{(1)}(\cdot;z)\|_{\mathrm{L}^\infty(\Theta)}^2\end{aligned}$$

We can then define a MSE on the gradient and then bound it as follows:

$$\begin{aligned}\mathrm{MSE}\big(\nabla \hat{J}(z,\theta)\big) &:= \mathbb{E}\left(\left\|\nabla J(z,\theta) - \nabla \hat{J}(z,\theta)\right\|_{\ell^2}^2\right) \\ &\leqslant \mathbb{E}\Big(\|\Phi^{(1)}(\cdot;z) - \hat{\Phi}^{(1)}(\cdot;z)\|_{\mathrm{L}^\infty(\Theta)}^2\Big) \\ &\quad + \sum_{j=1}^{d}\mathbb{E}\Big(\|\Psi_j^{(1)}(\cdot;z) - \hat{\Psi}_j^{(1)}(\cdot;z)\|_{\mathrm{L}^\infty(\Theta)}^2\Big).\end{aligned} \tag{2.2}$$

Hence, the MSE on the gradient can be bounded by the sum of the MSEs on each of the functions $\Psi_j^{(1)}$ and $\Phi^{(1)}$. This allows us to estimate the parametric expectation using the MLMC procedure for parametric expectations described in deliverable 5.5. The procedure can be applied to a linear combination of MSEs on parametric expectations, as in (2.2), trivially. The reader is referred to ibid. and Ayoul-Guilmard, Ganesh, Krumscheid et al. 2021 for a detailed explanation of the procedure.

2.2 Optimisation algorithm

We seek to find the optimal design $z^* \in \mathcal{A} \subset \mathbb{R}^d$ that minimises an objective function of the form shown in (2.1). As seen earlier, we can estimate the gradient $\nabla \hat{J}$ up to a prescribed tolerance by using the framework of parametric expectations developed in deliverable 5.5. We propose in algorithm 1 a procedure for a gradient-based algorithm to solve the CVaR minimisation problem in (2.1). The algorithm estimates the gradient at each iteration using optimally calibrated MLMC estimators, calibrated using the continuation multi-level Monte Carlo (CMLMC) algorithm described in ibid., to attain a tolerance that is a fraction of the gradient magnitude obtained in the previous iteration. During each optimisation iteration, the function is first minimised in θ and as a result, $J_\theta(z,\theta^*) = 0$ at the minimising value θ^*. An estimate of J_z is then computed at this minimising value of θ and a gradient step is taken forward in the design variable z, with a fixed step size. The algorithm is terminated once the gradient magnitude drops to a specified fraction of the initial value.

ALGORITHM 1: Optimisation algorithm proposal

```
1  DATA: Start design $z^0$, tolerance $0 < \epsilon < 1$, step size $\alpha > 0$ and $\eta > 0$
2  RESULT: Optimal design $z^*$
3  Set residual $r_{opt} > \epsilon$ and $k = 0$
4  WHILE $r_{opt} > \epsilon$ DO
5      IF $k = 0$ THEN
6          Simulate screening hierarchy
7      ELSE
8          Initialise CMLMC from the optimal hierarchy for $\mathrm{MSE}\big(\nabla \hat{J}_k\big)$
9          Simulate CMLMC adapting $\mathrm{MSE}\big(\nabla \hat{J}_{k+1}\big)$ to tolerance $\eta\|\nabla \hat{J}_k\|_{\ell^2}^2$
10     Estimate parametric expectation $\hat{\Phi}_L$ and quantile $q_\tau^{k+1} = \operatorname{argmin} \hat{\Phi}_L$
11     Compute $\nabla \hat{J}(z^k, q_\tau^{k+1})$ and $\|\nabla \hat{J}_{k+1}\|_{\ell^2}^2$
12     Compute $z_j^{k+1} = z_j^k - \alpha J_{z_j}(z^k, q_\tau^{k+1})$
13     Set $r_{opt} = \|\nabla \hat{J}_{k+1}\|_{\ell^2}^2 / \|\nabla \hat{J}_0\|_{\ell^2}^2$
14     Update $k \leftarrow k + 1$
```

We note here that this algorithm is a variation from deliverable 6.4, algorithm 2. The main difference is that instead of calibrating the MLMC hierarchy parameters to achieve a tolerance on the VaR and computing an estimate of the gradient using the same hierarchy, we instead calibrate the hierarchy to achieve a tolerance directly on the gradient. In addition, we proposed in ibid. an algorithm for adaptively selecting the step size using interpolation. However, for the demonstrations considered in this work, we utilise fixed step sizes.

3 Optimisation of the conditional value-at-risk of an oscillator

To demonstrate the optimisation framework, we use the FitzHugh–Nagumo system described in FitzHugh 1961. The FitzHugh–Nagumo model is a bidimensional simplification of the Hodgkin–Huxley model introduced by Hodgkin and Huxley 1952, which is used extensively in the field of neuroscience to model the phenomenon of spiking neurons. It is also an extension of the Van der Pol oscillator introduced in deliverable 6.4, § 3. The dynamical equations formulated over the interval $[0, T]$ read as follows:

$$\begin{pmatrix} \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} v - \frac{v^3}{3} - w + I \\ \epsilon(v + a - bw) \end{pmatrix}, \quad \begin{pmatrix} v(0) \\ w(0) \end{pmatrix} = \begin{pmatrix} v^0 \\ w^0 \end{pmatrix},$$

where $(v(t), w(t))$ in $\mathbb{R}^2$ denotes the state variables and a, b, ϵ and I denote system parameters. Figure 1 shows a phase-space plot containing the v and w-nullclines for a nominal value of the system parameters. As long as the intersection of the two nullclines lies in the interval $v \in [-1, 1]$, indicated by the black lines, the oscillator enters a limit cycle. 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.

In particular, we study the forced FitzHugh–Nagumo system over a time interval $[0, T]$:

$$\begin{pmatrix} \dot{v} \\ \dot{w} \end{pmatrix} = \begin{pmatrix} v - \frac{v^3}{3} - w + I + \sigma\dot{W}_1 \\ \epsilon(v + a - bw) + \sigma\dot{W}_2 \end{pmatrix}, \quad \begin{pmatrix} v(0) \\ w(0) \end{pmatrix} = \begin{pmatrix} v^0 \\ w^0 \end{pmatrix},$$

where $\dot{W}_1$ and $\dot{W}_2$ are standard Brownian paths and $\sigma = 0.01$ controls the noise strength. The problem shows several characteristics that make it a surrogate problem for fluid dynamics optimisation problems. The first is that the problem is time-dependent with limit cycle oscillations, resembling the dynamics of low-Reynolds' number vortex shedding. The second is that it is a multi-parameter system, allowing us to test our optimisation algorithm for multiple system parameters, similar to what one can expect in civil engineering applications for shape optimisation.

We denote by $z = (a, b, \epsilon, I)$ the vector of design parameters with respect to which we want to minimise the system. The OUU problem which we wish to solve is given by (2.1) with $z_0 = [0.7, 0.8, 0.08, 1.0]$, with the significance of the CVaR set to $\tau = 0.7$. We use the approach detailed in section 2.2, namely the use of algorithm 1 with the CMLMC algorithm detailed in deliverable 5.5. We discretise the interval $[0, T]$ using a hierarchy of uniform grids $t_j = j\,\Delta t_l, j \in [\![0, N_{T,l}]\!]$, with

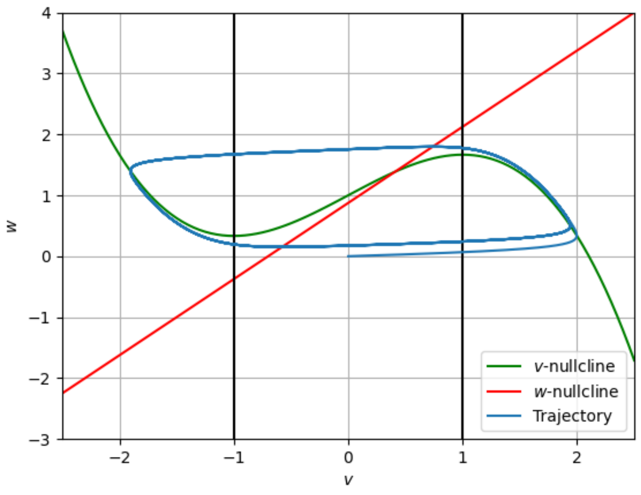


Figure 1: FitzHugh–Nagumo oscillator dynamics

$\Delta t_l = T/N_{T,l}$ and $N_{T,l} = N_{T,0}2^l$. We set $T = 10$ and $N_{T,0} = 40$. Using the notation v_n^l to denote the approximation of $v(t_n)$ at level l, the discretised system then reads:

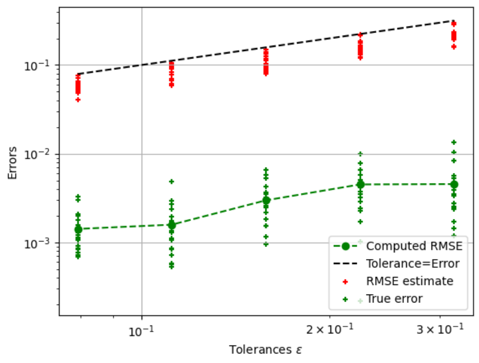
$$\begin{pmatrix} v_{n+1}^l \\ w_{n+1}^l \end{pmatrix} = \begin{pmatrix} v_n^l \\ w_n^l \end{pmatrix} + \Delta t_l \begin{pmatrix} v_n^l - \frac{(v_n^l)^3}{3} - w_n^l + I \\ \epsilon(v_n^l + a - bw_n^l) \end{pmatrix} + \sigma\sqrt{\Delta t_l}\begin{pmatrix} \xi_{1,n}^l \\ \xi_{2,n}^l \end{pmatrix},$$
$$\begin{pmatrix} v_0^l \\ w_0^l \end{pmatrix} = \begin{pmatrix} v^0 \\ w^0 \end{pmatrix}, \quad n \in [\![0, N_{T,l} - 1]\!],$$

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:

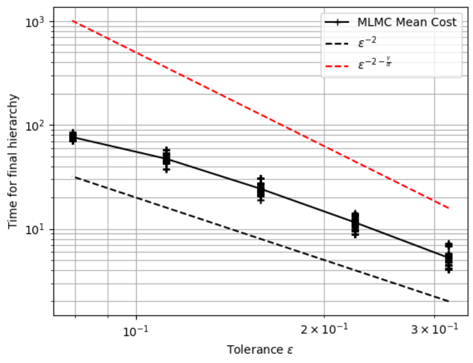
$$Q = \frac{1}{T}\int_0^T v^2(t)\,\mathrm{d}t \approx \sum_{n=0}^{N_{T,l}-1} \left(\frac{(v_n^l)^2 + (v_{n+1}^l)^2}{2} \right) \Delta t_l =: Q_l.$$

We consider the corresponding adjoint variables λ_n^l and μ_n^l corresponding to v_n^l and w_n^l, $n \in [\![0, N_{T,l}]\!]$ respectively. The adjoint equation reads as follows:

$$\begin{pmatrix} \lambda_n^l \\ \mu_n^l \end{pmatrix} = \begin{pmatrix} \lambda_{n+1}^l \\ \mu_{n+1}^l \end{pmatrix} + \Delta t_l \left(\begin{pmatrix} (1 - (v_n^l)^2) & \epsilon \\ -1 & -\epsilon b \end{pmatrix} \begin{pmatrix} \lambda_{n+1}^l \\ \mu_{n+1}^l \end{pmatrix} + \begin{pmatrix} \frac{2v_n^l}{T} \\ 0 \end{pmatrix} \right),$$
$$\begin{pmatrix} \lambda_{N_{T,l}}^l \\ \mu_{N_{T,l}}^l \end{pmatrix} = \Delta t_l \begin{pmatrix} \frac{v_n^l}{T} \\ 0 \end{pmatrix}, \quad n \in [\![1, N_{T,l} - 1]\!].$$



(a) Reliability of error estimator



(b) Complexity behaviour

Figure 2: Summary of results for the FitzHugh–Nagumo system

The reader is referred to deliverable 6.4, appendix A for the details of the derivations.

Once the adjoint equation is solved backwards in time, the approximation $Q_{z,l}$ of the sensitivities Q_z at level l can then be obtained as follows:

$$Q_{a,l} = \sum_{n=0}^{N_{T,l}-1} \Delta t_l \epsilon \mu_{n+1}^l \qquad Q_{b,l} = -\sum_{n=0}^{N_{T,l}-1} \Delta t_l \epsilon w_n^l \mu_{n+1}^l$$
$$Q_{I,l} = \sum_{n=0}^{N_{T,l}-1} \Delta t_l \lambda_{n+1}^l, \qquad Q_{\epsilon,l} = \sum_{n=0}^{N_{T,l}-1} \Delta t_l (v_n^l + a - b w_n^l) \mu_{n+1}^l.$$

To demonstrate the performance of the algorithm, we first assess the performance of the CMLMC algorithm and adaptive strategy. We estimate the gradient $\nabla \hat{J}$ for $z = z^0$. The gradient and gradient error are estimated using the MLMC procedure described in section 2.1. To assess the reliability of the error bound derived in section 2.1, we run a reliability study wherein we adapt the parameters of the MLMC hierarchy to attain a tolerance on the MSE. 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 10^4 samples. The resultant plot is shown in figure 2a. As can be seen from the figure, the error estimates are significantly more conservative than the true errors. This is a result of bounding the error of the gradient at a point (z, θ) using the L^∞-norm over the entire interval Θ, leading to conservative bounds. However, since the error estimates produce practically computable hierarchies, we use them for the studies in this work.

We present in Fig 2b the complexity behaviour of the CMLMC estimator. We compute the cost required to obtain the final optimal hierarchy for a given tolerance.

As can be seen from the figure, the cost grows as ϵ^{-2} where ϵ^2 is the tolerance we impose on the MSE of the gradient, which is the theoretically predicated best case performance for the MLMC estimator. For comparison, we plot the expected cost growth rate for a Monte Carlo estimator as well.

We now examine the performance of the gradient descent algorithm. We plot in figure 3 the difference between the objective function value at the current iteration and the final iteration, which we treat as a reference value. We observe exponential convergence in the number of iterations, as is predicted by theoretical consideration for the full gradient descent algorithm for convex functions. In addition, we plot in figure 4 the CDF computed for different iterations of the optimisation algorithm using the design z^k and a reference Monte Carlo simulation with 1000 simulations. We observe that the CDF moves left, reducing the mass in the tail of the distribution. This is since we are minimising the CVaR and thereby the tail of the distribution.

figure 5 shows the optimal hierarchy obtained from the CMLMC algorithm for each iteration of the optimisation. We observe that since the tolerance supplied to the MLMC algorithm is a fraction of the gradient magnitude, the optimally tuned hierarchy becomes larger for later iterations of the optimisation. In the limit of zero gradient, the hierarchy parameters N_l, n and L all go to infinity, hence ensuring consistency with the original problem. figure 6 shows the cumulative cost required for the optimisation algorithm to reach a given gradient magnitude. The cumulative cost is computed as $\sum_{i=0}^{k}\sum_{l=0}^{L} m_l^{(i)}(€(Q_l) + €(Q_{l-1}))$, where $\{m_l^{(k)}\}_{l=0}^{L}$ denotes the optimal level-wise sample sizes for the k^{th} optimisation iteration $€(Q_l)$ denotes the average cost of simulating one sample of Q_l. The cumulative cost at a given optimisation iteration is defined as the sum of costs of all optimal hierarchies until the current optimisation iteration. This cost is plotted versus the gradient magnitude. We observe that the cumulative cost is proportional to the cost of the optimally tuned MLMC hierarchy, since we adaptively select the parameters of the MLMC hierarchy to achieve a tolerance proportional to the gradient.

The data set of the results of this numerical experiments is publically available as a part of Ayoul-Guilmard, Rosa M. Badia et al. 2021.

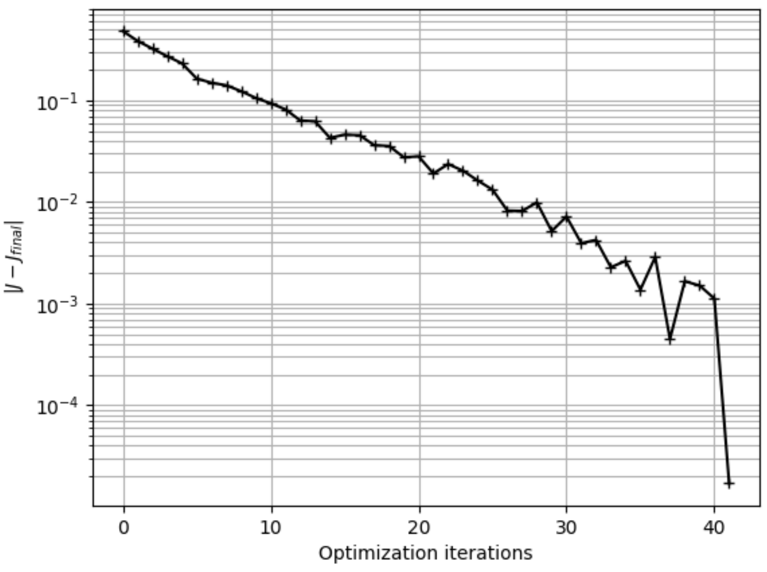


Figure 3: Decay of objective function towards final value

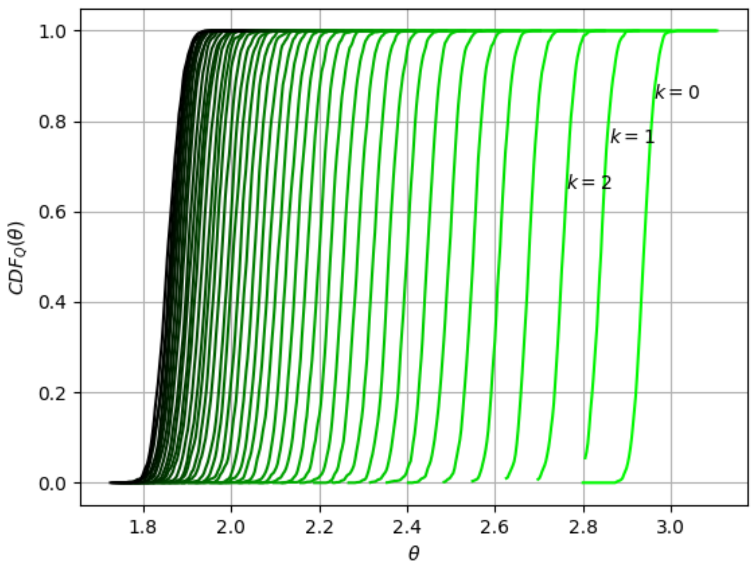


Figure 4: CDF for different optimisation iterations. Colour corresponds to optimisation iteration number, with darker colours indicating later iterations.

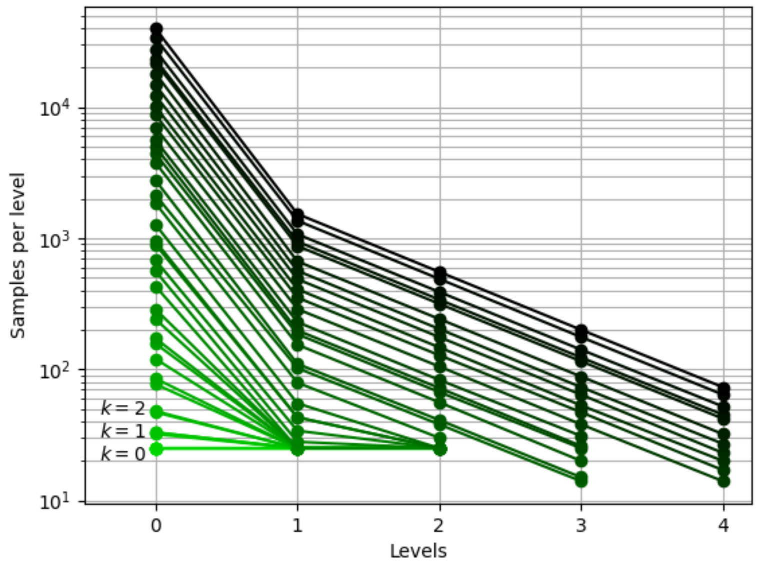


Figure 5: Level-wise sample sizes for different optimisation iterations. Colour corresponds to optimisation iteration number, with darker colours indicating later iterations.

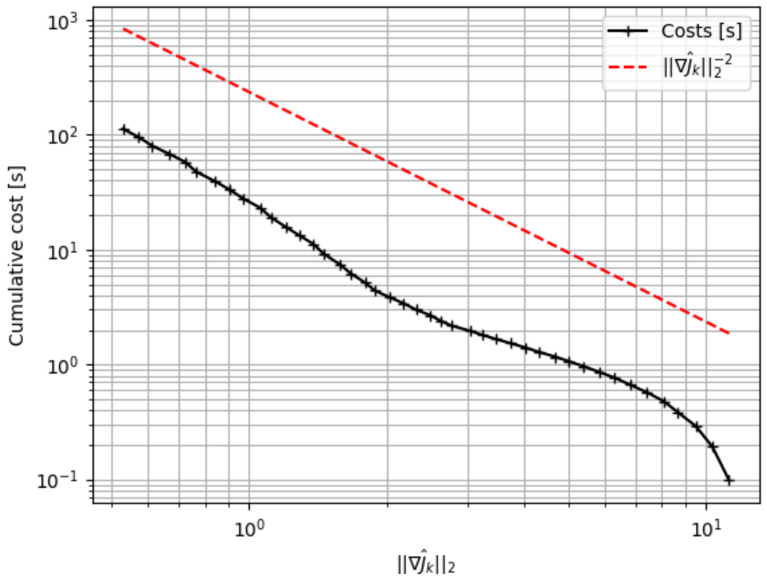


Figure 6: Cumulative optimisation cost to reach a given gradient magnitude

4 Stochastic optimisation of an airfoil

4.1 Method

In order to demonstrate the optimisation workflow using the CVaR as risk measure in the objective function, the potential flow over an airfoil is considered. To this end, the aim is to find the optimal shape of an airfoil that minimises the chance of obtaining low values of lift. More precisely, we aim at minimising the $\mathrm{CVaR}_\tau(-C_l)$ for a given significance level τ.

The method used to solve the full potential equation was presented by the authors in Davari et al. 2019 which features the solution of the problem on body-fitted meshes with an embedded wake. In Núñez et al. 2022, the solution was extended to a full embedded approach. This same method and the discussion of its sensitivity analysis was also introduced in deliverable 6.4. In this document, the deterministic adjoint equations presented in ibid. are extended to the stochastic case with the CVaR as risk measure.

The optimisation problem to be solved is detailed next. Let $(M_\infty, \alpha_\infty)$ be uncertain parameters, following a known distribution, as discussed in deliverable 5.5. The objective function presented in deliverable 6.4, problem 4, is rewritten including the CVaR as risk measure. Note that the problem is set up using the negative of the lift coefficient, in order to define the problem as a minimisation problem, so the shape that minimises the CVaR is the one that robustly maximises the lift.

PROBLEM 2 (Airfoil shape for CVaR-optimal lift). The lift coefficient C_l is now a random variable, and we wish to minimise $\mathrm{CVaR}_\tau(-C_l)$. We define the optimal airfoil as $z_\star$ such that

$$\begin{cases} (z_\star, s_\star) := \operatorname{argmin}\{J(z,s) := \phi(Q(u(\omega,z)),s) : (z,s) \in Z \times \mathbb{R}\} \\ \qquad\qquad \text{s.t.} \quad F_\omega(u(\omega,z),z) = 0 \quad \text{a.e} \quad \omega \in \Omega \\ \qquad\qquad\qquad\qquad \text{and} \quad c(z) = 0. \end{cases}$$

where $Q = -C_l$ is the quantity of interest, defined as the negative of the lift coefficient. The residual F_ω corresponds to the residual of the primal problem, which is set as a constraint on the optimisation problem. Notice that here $u(\omega, z)$ denotes the fluid potential for a given design z, not to be confused with the parametric expectation defined in section 2. The derivation of this residual was discussed in detail in ibid., and it is reformulated for a stochastic scenario in deliverable 5.5. Lastly, a set of equality constraints of the type $c(z) = 0$ are added, which are introduced in equations (4.2), (4.3) and (4.4). $\Diamond$

In order to find the optimal update of the airfoil shape, the derivative of the objective function for each of the geometrical parameters is computed. In deliverable 6.4, the deterministic discrete adjoint equations were presented, where the change of the quantity of interest Q with respect to the geometry parameters z is computed as:

$$\frac{\mathrm{d}Q}{\mathrm{d}z} = \frac{\partial Q}{\partial z} + \lambda^\top \frac{\partial F_\omega}{\partial z},$$

where λ represent the *adjoint* variables. The adjoint variables are computed solving the so-called adjoint equation:

$$\frac{\partial Q}{\partial u} = -\left(\frac{\partial F_\omega}{\partial u}\right)^\top \lambda.$$

For the stochastic case, a similar approach is used to find the update on the geometry parameters, considering the effect of the risk measure $\mathcal{R}(Q) = \textsc{cVaR}_\tau(Q)$. The objective function is replaced by the Lagrangian:

$$\mathcal{L}(u, z, \lambda) = \mathcal{R}(Q(u)) + \mathbb{E}(\lambda^\top F_\omega(u, z)),$$

where $\mathcal{R}(Q)$ represents the risk measure of the distribution of the quantity of interest. This expression is equivalent to the one introduced in deliverable 6.2, Section 3.1, without the penalisation term. Now, each of the Gateaux derivatives with respect to u, z and λ in the directions δu, δz and $\delta\lambda$ are computed. In λ, it reads:

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\lambda}(\delta\lambda) = \mathbb{E}(F_\omega(u, z)\delta\lambda) = 0, \quad \forall \delta\lambda,$$

which corresponds to the primal problem being satisfied for almost every realisation ω, i.e, that $u = u(\omega, z)$ is a solution of the problem $F_\omega(u, z) = 0$ for a given ω and z. The derivative in u reads:

$$\begin{aligned}
&\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}u}(\delta u) = \frac{\mathrm{d}\mathcal{R}(Q(u))}{\mathrm{d}u}(\delta u) + \mathbb{E}\left(\lambda^\top \frac{\mathrm{d}F_\omega(u, z)}{\mathrm{d}u}(\delta u)\right) = 0, \quad \forall \delta u, \\
\Leftrightarrow\ &\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}u}(\delta u) = \mathbb{E}\left(\frac{\partial \mathcal{R}(Q)}{\partial Q}\frac{\partial Q(u)}{\partial u}\partial u + \lambda^\top \frac{\partial F_\omega(u, z)}{\partial u}\partial u\right) = 0, \quad \forall \delta u \qquad (4.1)
\end{aligned}$$

where the theorem introduced in ibid. from (Shapiro et al. 2009, theorem 6.10) was employed to write the Gateaux derivative $\frac{\mathrm{d}\mathcal{R}}{\mathrm{d}u}$ as the expectation of a pointwise derivative. Equation (4.1) implies that the adjoint equation is satisfied for almost every realisation ω, which allows computing independently the adjoint variables $\lambda(\omega)$ for every realisation. Notice that the adjoint equations can be equivalently written as:

$$\frac{\partial Q}{\partial u} + \hat{\lambda}^\top \frac{\partial F_\omega(u, z)}{\partial u} = 0, \quad \hat{\lambda} = \frac{\partial \mathcal{R}}{\partial Q}\lambda.$$

Combining this with the derivative of the Lagrangian in z, the sensitivities can be computed as:

$$\frac{\partial \mathcal{L}}{\partial z} = \mathbb{E}\left(\frac{\partial \mathcal{R}}{\partial Q}\left(\frac{\partial Q}{\partial z} + \hat{\lambda}^{\top}\frac{\partial F_{\omega}(u,z)}{\partial z}\right)\right).$$

This has the advantage that the modified adjoint solution $\hat{\lambda}$ corresponds to the standard deterministic adjoint solution, computed for each random event. The sensitivity of the risk measure enters only in the computation of the sensitivity $\frac{\partial \mathcal{L}}{\partial z}$. The reader is referred to deliverable 6.2 for the details of this derivation.

The constraints applied in 2 are the same as the ones presented in ibid., Section 4.2.3 and 4.2.4, and we refer to those section for further details. The main functions used in this deliverable to define the constraints are summarised next:

Perimeter The perimeter is the sum of the lengths of the edges of the airfoil:

$$P = \int_{\Gamma_g} \mathrm{d}\Gamma_g,$$

where Γ_g refers to the boundary enclosing the geometry of the airfoil. The gradient of the perimeter is computed by finite difference for every point of the airfoil.

$$\frac{\partial P(z)}{\partial z} \approx \frac{P(z+\varepsilon) - P(z)}{\varepsilon}.$$

Angle of attack The angle of attack is defined as the angle between the inflow velocity and the chord line, using $\boldsymbol{v}$ as the inflow velocity and $\boldsymbol{c}$ as the chord vector. The chord vector is defined as the position of the leading edge and the trailing edge, with the trailing edge as origin:

$$\alpha = \arccos\left(\frac{\boldsymbol{v}\cdot\boldsymbol{c}}{\|\boldsymbol{v}\|\|\boldsymbol{c}\|}\right).$$

The gradient of the angle of attack will only depend on the position of the trailing edge and the leading edge nodes. It can easily be computed by finite differences.

$$\frac{\partial \alpha}{\partial z} \approx \frac{\alpha(z+\varepsilon) - \alpha(z)}{\varepsilon}.$$

Chord line length Another feature that is important to preserve across the design space is the chord length, that can be defined as the norm of the chord vector, represented by the leading and trailing edge nodes:

$$l_c = \|\boldsymbol{c}\|,$$

and its gradient is easily computed by finite differences:

$$\frac{\partial l_c}{\partial z} \approx \frac{l_c(z+\varepsilon) - l_c(z)}{\varepsilon}.$$

These functions are applied as equality constraints to enforce that they do not drift from their initial values P_0, α_0 and l_{c0}:

$$c_1(z) = P_0 - \int_{\Gamma_g} \mathrm{d}\Gamma_g = 0, \tag{4.2}$$

$$c_2(z) = \alpha_0 - \arccos\left(\frac{v \cdot \boldsymbol{c}}{\|v\|\|\boldsymbol{c}\|}\right) = 0, \tag{4.3}$$

$$c_3(z) = l_{c0} - \|\boldsymbol{c}\| = 0. \tag{4.4}$$

A trust region algorithm (Yuan 1999) is used to solve the constrained optimisation problem, as done in deliverable 6.4.

In order to model the uncertainty in the problem the approach used in Tosi, Amela et al. 2021 is considered. This approach is also used in deliverable 5.5. The Dirichlet condition on Γ_D is transformed to a stochastic condition by considering some variability in the free stream velocity $\boldsymbol{v}_\infty$, which can be expressed in terms of the Mach number and the angle of attack:

$$\boldsymbol{v}_\infty = M_\infty a_\infty \begin{pmatrix} \cos(\alpha) \\ \sin(\alpha) \end{pmatrix}$$

where the free stream sound velocity is set as $a_\infty = 340\,\mathrm{m\,s^{-1}}$. The Mach number and the angle of attack are therefore two independent random variables that model the stochastic problem. The probability distribution of the two stochastic variables are $M_\infty \sim \mathcal{N}(0.3, 0.1)$ and $\alpha \sim \mathcal{N}(5.0°, 0.02)$, respectively. $\mathcal{N}(\mu, \sigma^2)$ denotes a normal distribution of mean μ and variance σ^2.

The main quantity of interest is the lift coefficient, which is computed using the Kutta–Joukowski theorem:

$$C_l = \frac{\Gamma}{\frac{1}{2} v_\infty c},$$

where Γ is the circulation of the flow and the subscript, ρ_∞ is the free stream density and v_∞ the free stream velocity. The circulation Γ can be expressed as the potential jump across the domain $\Gamma = u^+ - u^-$, shown by Nishida and Drela 1995. This leads to the definition of the lift coefficient in terms of the potential jump across the wake:

$$C_l = \frac{2}{v_\infty c}(u^+ - u^-).$$

4.2 Numerical application

The optimisation problem introduced in problem 2 is solved using the MLMC method using as initial shape the symmetric NACA0012 airfoil at an angle of attack of

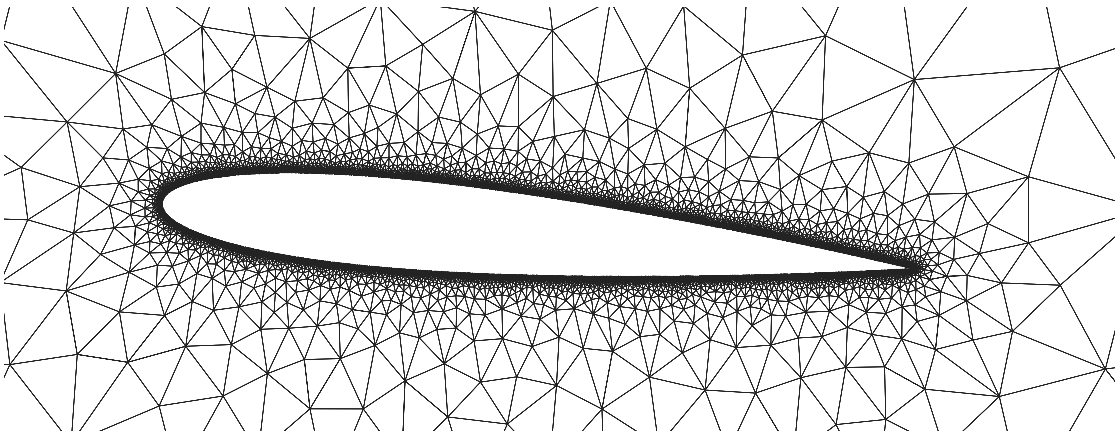


Figure 7: Starting mesh of the optimisation problem

$\alpha = 5°$. The airfoil is discretised with $n_{\text{nodes}} = 10\,198$ nodes, all of which are considered geometry parameters in the optimisation problem, denoted with z. The starting mesh for the algorithm is shown in figure 7.

Considering the number of parameters in the optimisation problem, the adaptive MLMC approach used in section 3 is replaced by a fixed hierarchy, as the cost of estimating the error of the sensitivity on every geometry parameter to adapt the hierarchy was too high. The fixed hierarchy is constructed with 4 levels ($L := 3$) and $N = 256$ samples at each level, whose cost at every optimisation step is expected to be constant. The hierarchy of meshes to solve the MLMC algorithm at every optimisation step was computed on the fly using the adaptive refinement strategy described in deliverable 5.3, § 2.2.2 and deliverable 5.5, § 2.2, where a metric-based refinement following the method from C. Dapogny et al. 2014 is considered using MMG software. The metric is defined as:

$$\mathcal{M} = \mathcal{A}\hat{\Lambda}^{\top}\mathcal{A}, \qquad \text{where} \qquad \hat{\Lambda} = \operatorname{diag}(\hat{\lambda}_i)$$
$$\hat{\lambda}_i = \min\left(\max\left(\frac{c_d|\lambda_i|}{\varepsilon}, h_{\max}^{-2}\right), h_{\min}^{-2}\right),$$

where λ_i are the eigenvalues and $\mathcal{A}$ the matrix of eigenvectors of the Hessian $\mathcal{H}_u$ of a given variable u. The eigenvalues are truncated for some user-defined minimal and maximal sizes, $h_{\min}$ and $h_{\max}$. The metric depends on a constant c_d and the interpolation error ε. This means that the refinement can be driven by the value of ε.

The testing and production runs to solve this problem were run both in MareNostrum 4, whose architecture is described in deliverable 4.5, and Karolina, whose architecture is described in deliverable 5.5, § 3.2.3.

The total number of samples solved in the optimisation problem is therefore $N_{\text{total}} = (2L - 1)N$, as shown in table 1. Each of the samples drawn are distributed

Table 1: Hierarchy used to solve the MLMC problem at every optimisation step. A fixed number of samples $N = 256$ is used at each level. For every level $L > 0$, each realisation evaluates the lift coefficient on each level L and $L-1$. The interpolation error ε used to generate each level is also shown.

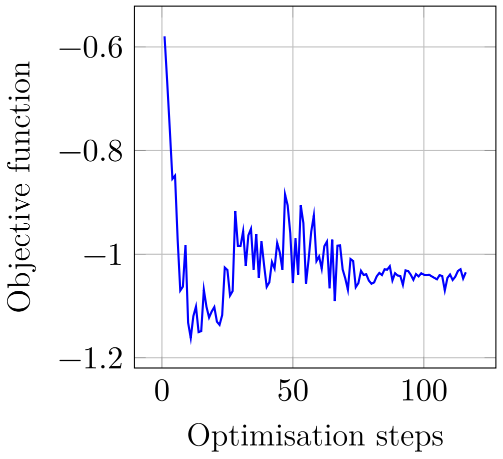
Level l	Samples		ε
	mesh l	mesh $l-1$	
0	256		
1	256	256	0.1
2	256	256	0.05
3	256	256	0.025

over the available computing nodes. For these runs, COMPSs programming model (described in deliverable 4.5) was used. Each sample was run with a single computing unit. After running the hierarchy, the estimation of the CVaR and the estimation of the sensitivities is performed by XMC.

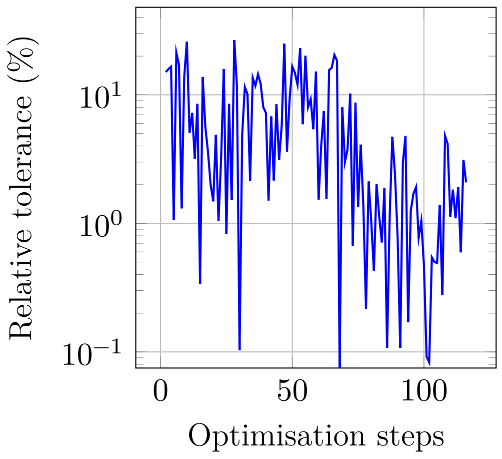
The results obtained after solving problem 2 for $\tau = 0.7$ are discussed next. The problem was run during 24 hours, after which the analysis was stopped. Figure 8a shows the change of the objective function through the optimisation steps. It can be seen that the objective function approaches an optimal value, but not smoothly. The objective function reaches a minimum value early in the optimisation loop, but then increases as the geometrical constraints are applied.

The stopping criteria of the algorithm was set in terms of the relative change of the objective function, with a limit of 0.001 %, which was not reached within the wall-time of 24 hours. The evolution of the stopping criteria is illustrated in figure 8b. This is partially due to the fact that the MLMC hierarchy was not adaptive, and thus the error of the MLMC algorithm was not under control. This does not only affect the accuracy of the estimated objective function, but also the values of the estimated sensitivities. It is expected that using an adaptive hierarchy would improve the convergence of the optimisation, although the cost of every step would also increase. Nonetheless, the variability of the objective function observed after solving the 24 hours can be considered acceptable, even if the initial stopping criteria was not met.

A comparison of the shape design obtained is presented in figure 9, where the resulting shape from the stochastic optimisation problem is compared with the initial configuration. Also, the shape resulting from solving a classical deterministic optimisation problem with the mean conditions of the stochastic problem is shown. It must be noted that both problems are optimising the shape on different objective functions, so it would not be expected to obtain the exact same shape.



(a) Evolution of the objective function with the number of optimisation steps. The objective function is the $\text{CVaR}_{0.7}$ of the negative of the lift coefficient.



(b) Evolution of the relative change of the objective function through the optimisation steps

Figure 8: Evolution of the objective function and the relative tolerance in the optimisation problem showcased

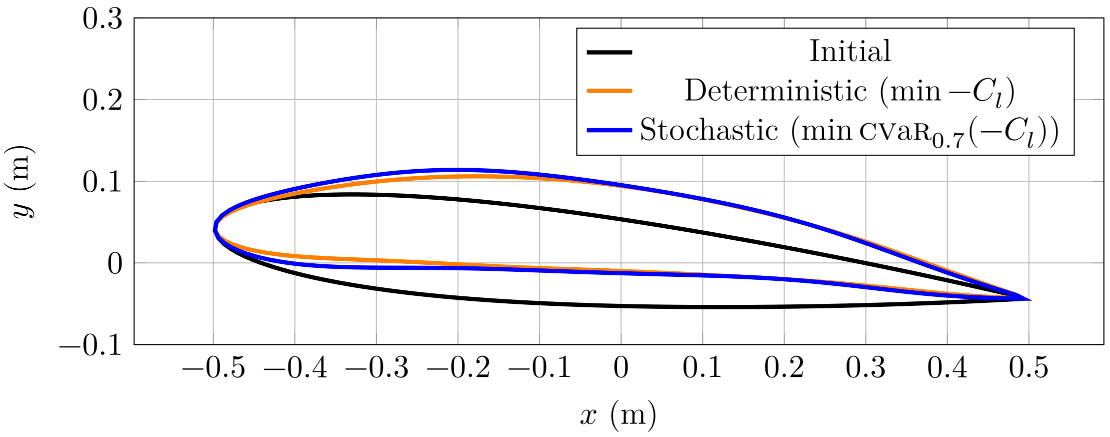


Figure 9: Shape comparison of the initial and final shapes after solving a stochastic optimisation problem with $\text{CVaR}_{0.7}(-C_l)$ minimisation and after solving a deterministic optimisation problem where the lift coefficient is maximised, as introduced in deliverable 6.4

As the optimisation loop advances, the distribution of the lift coefficient changes, updating the CVaR value. Figure 10 shows the evolution of the CDF of the negative of the lift coefficient over the first optimisation steps. Since the optimisation problem is written as a minimisation problem, the distribution moves towards the left side. Figure 11a shows the PDF of the lift coefficient for the first optimisation step, alongside with the mean, VaR, and CVaR values. Also, the distribution of the mean, VaR and CVaR of the pressure coefficient over the airfoil is presented in figure 12. The results on the initial step are equivalent to those obtained in deliverable 5.5. Figure 11b shows the PDF of the lift coefficient for the final optimisation step, and the same statistical quantities as before, and figure 13 shows the distribution of the pressure coefficient statistics for the final step over the airfoil. For this shape, a local effect caused by the mesh deformation across the steps is visible on the top surface in figure 13a, but it is not big enough to have a significant effect on the final integrated quantity.

The results obtained are a showcase of an engineering application with many design parameters for which the optimisation under uncertainties workflow is applicable. A state-of-the-art method to estimate relevant risk measures for the application was applied. This was possible thanks to the combination of the different utilities and software packages released during the project (deliverable 5.2; deliverable 4.5; deliverable 1.4) including: the remeshing software MMG; the solver Kratos Multiphysics; the programming models COMPSs and Quake; and the uncertainty quantification library XMC.

The data set of raw results of these numerical experiments is publically available as a part of Ayoul-Guilmard, Rosa M. Badia et al. 2021.

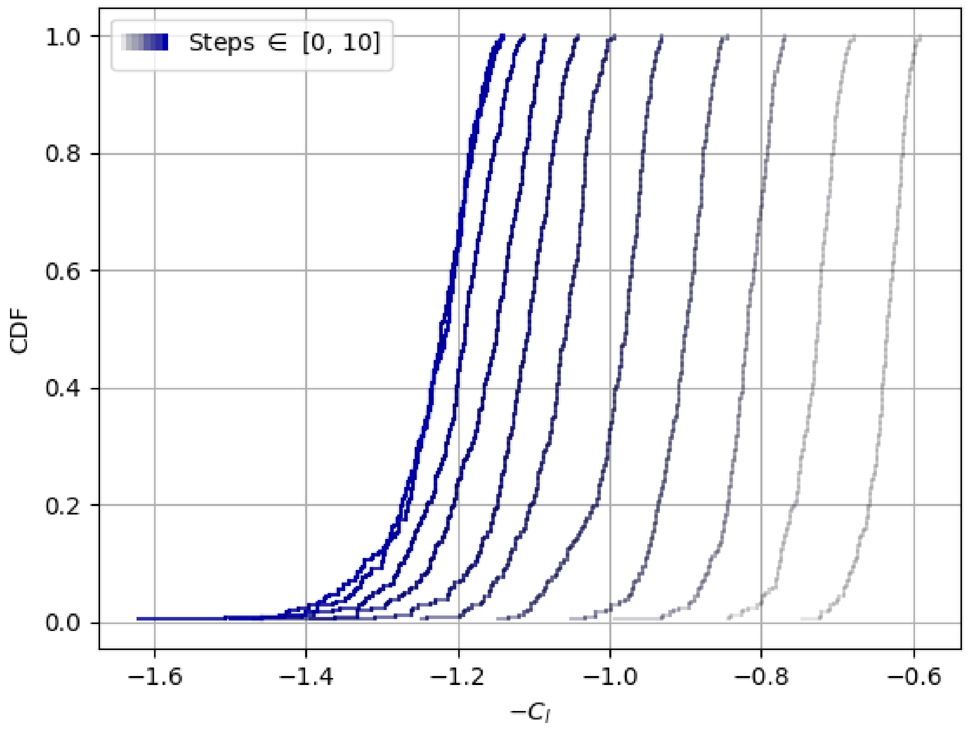
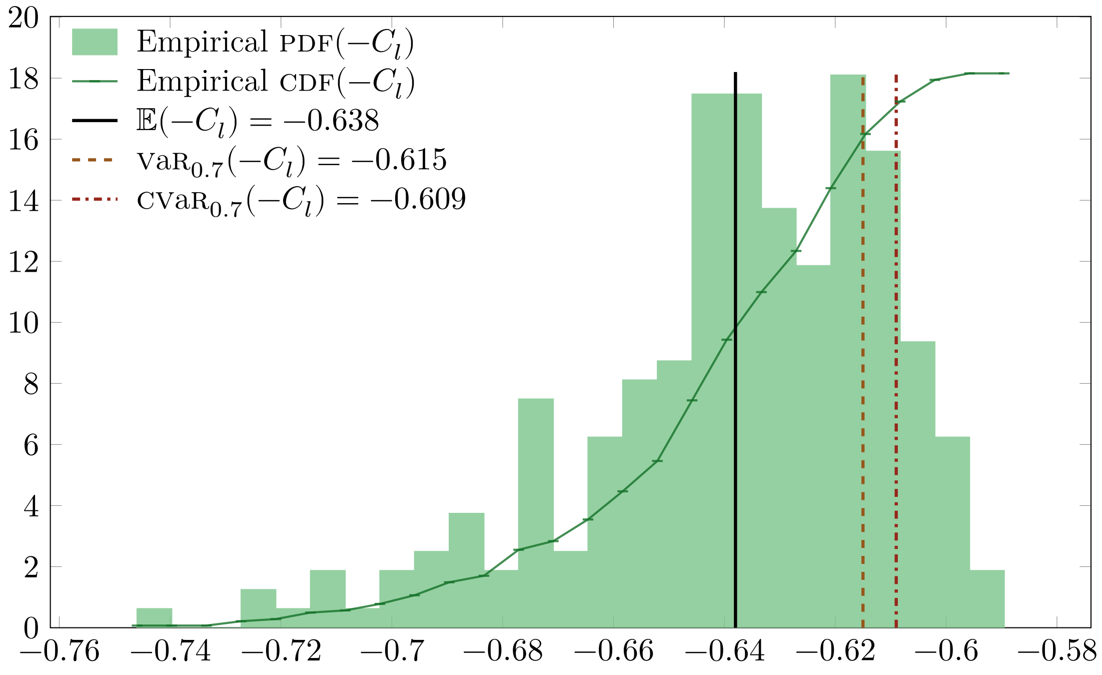
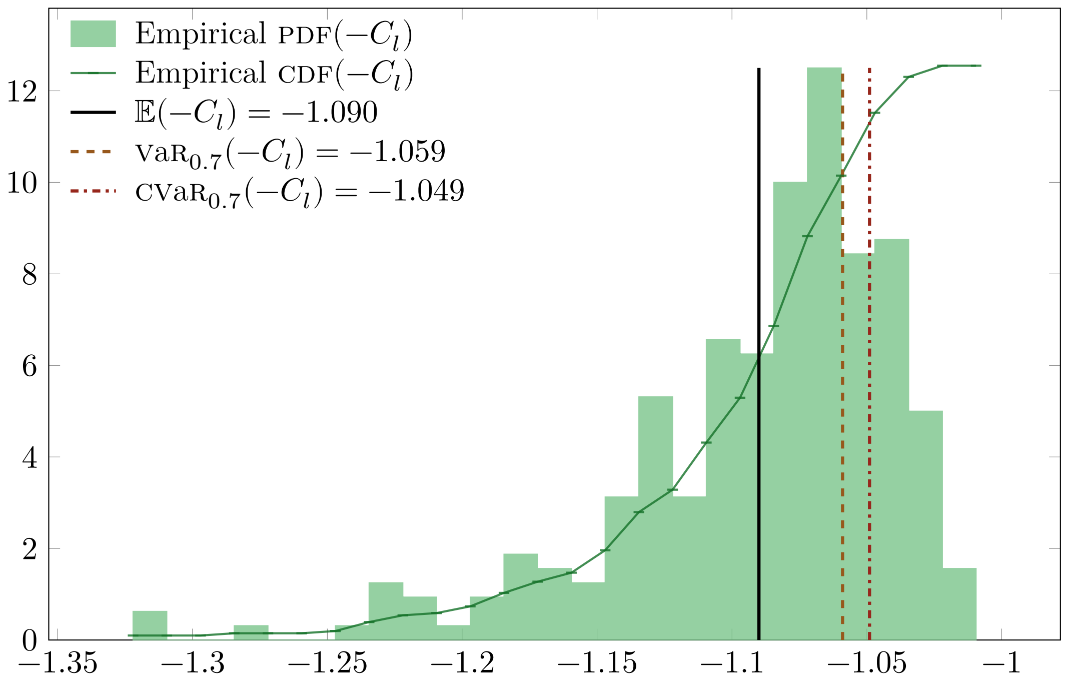


Figure 10: CDF of the negative of the lift coefficient for the first 10 iterations of the optimisation problem, where the curve gets darker as the iteration increases

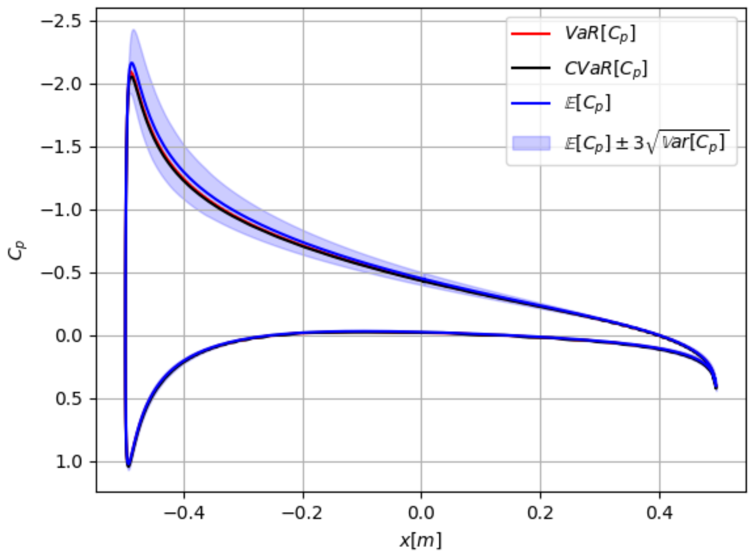


(a) Initial shape

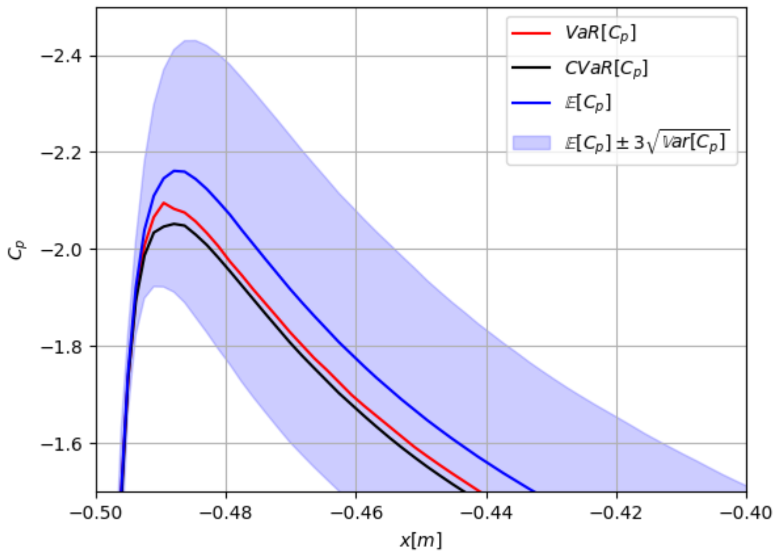


(b) Final shape

Figure 11: Distribution of the opposite of the lift coefficient for the initial and final optimisation steps. Constructed from the 256 samples from the finest level from the hierarchy in table 1.

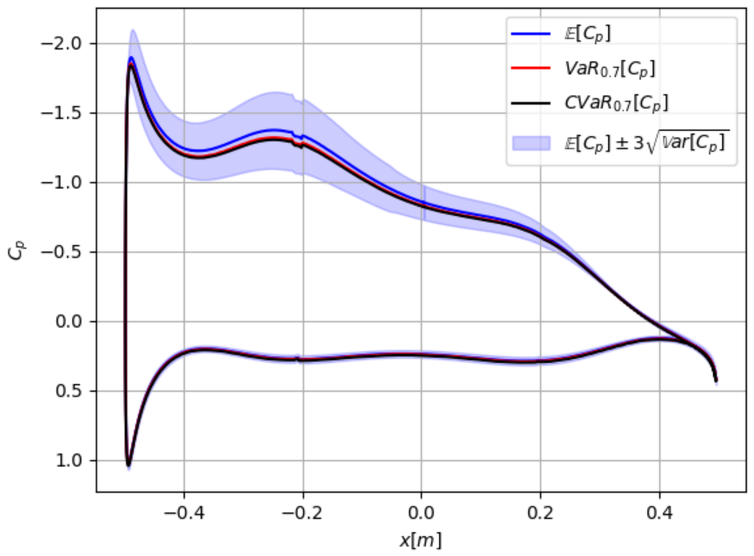


(a) Pressure distribution along the airfoil shape

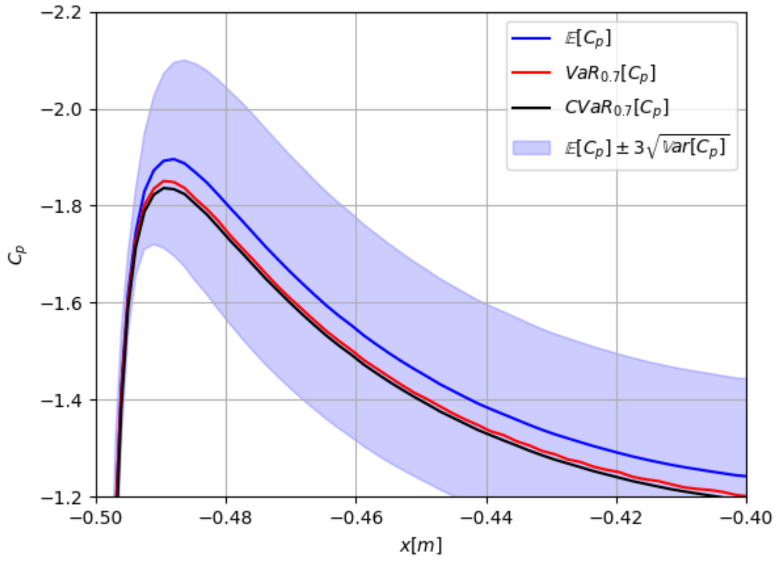


(b) Zoom in at the trailing edge

Figure 12: Distribution of statistical quantities for the pressure coefficient over the initial shape of the NACA0012 airfoil at 5° (significance $\tau = 0.7$)



(a) Pressure distribution along the airfoil shape



(b) Zoom in at the trailing edge

Figure 13: Distribution of statistical quantities for the pressure field over the final shape of the NACA0012 airfoil at 5° (significance $\tau = 0.7$)

5 Shape optimisation of a tall building under uncertain wind conditions

This is the most ambitious problem considered by the ExaQUte project: a three-dimensional robust optimisation of the shape of a building under turbulent, uncertain wind conditions. As far as optimisation is concerned, its most challenging feature is the turbulence: the chaotic behaviour of the fluid flow precludes the use of MLMC methods, as demonstrated in deliverable 5.4, § 2. For this reason an alternative multi-fidelity Monte Carlo (MFMC) method is studied in deliverable 5.5, § 3. However, further work would be required to use this method in a gradient-descent algorithm such as presented in section 3. In particular, the accurate estimation of parametric sensitivities such as described in section 2 would have to be researched for the MFMC estimator selected. For these reasons, a different optimisation method is used to address this particular problem of interest to wind engineering.

In this report we will describe the optimisation algorithm as well as the method used to estimate the sensitivity of the objective function to the optimisation parameters. We will also present the target application, insofar as it pertains to this discussion. The application itself and its results are available in detail in deliverable 7.4. The reader may also be interested deliverable 6.4, § 5 which previously presented this fluid simulation and part of the method hereafter.

5.1 Formulation of the fluid flow problem

This fluid-flow problem is similar to the one described in ibid. and deliverable 5.5, § 3.2.1: a tall building under turbulent wind, with uncertain inflow. The horizontal section of the building is an ellipse, whose orientation and diameters change with the height. This shape is therefore parameterised by two real numbers: (I) the angle of rotation around the vertical axis of the top section with respect to the section at the base of the building; (II) the length of the minor axis of the section at the top of the building. The length of the major axis is a function of the length of the minor axis such that the section area remains constant. Consequently, the design space is $\mathbb{R}^2$.

PROBLEM 3 (Incompressible flow over a building). The flow is simulated for 200 s on the domain D represented in figure 14. The building is parameterised, and for a given design $z \in \mathbb{R}^2$ we note $B(z)$ its shape. The boundary of the domain is therefore partitioned as $\partial D = D_{\text{inlet}} \cup D_{\text{outlet}} \cup D_{\text{top}} \cup D_{\text{walls}} \cup D_{\text{bottom}} \cup B(\boldsymbol{z})$. The wind pressure p and velocity $\boldsymbol{u}$ are solution of the incompressible Navier–

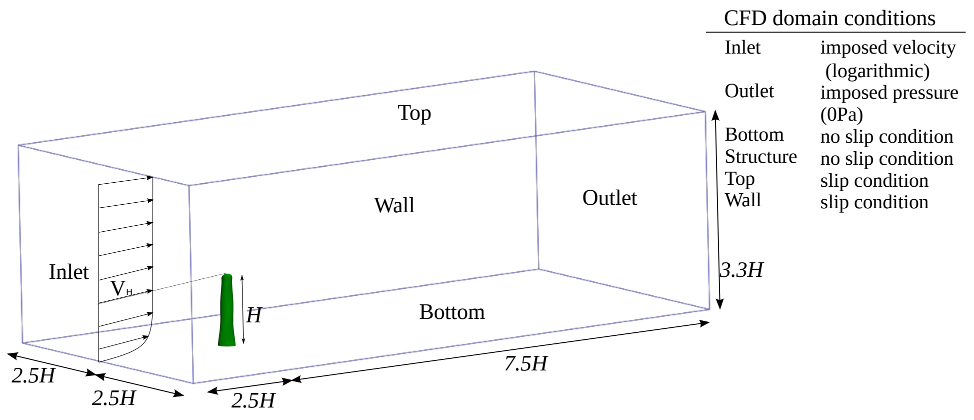


Figure 14: Details of the simulation domain and boundary conditions ($H = 180\,\mathrm{m}$)

Stokes equations with boundary conditions stated below.

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u} - \frac{\mu}{\rho}\Delta\boldsymbol{u} + \nabla p = -g\boldsymbol{e}_3 \qquad \text{in } [0,200]\times D, \tag{5.1a}$$

$$\nabla\cdot\boldsymbol{u} = 0 \qquad \text{in } [0,200]\times D, \tag{5.1b}$$

with the boundary conditions

$$\boldsymbol{u} = \boldsymbol{u}_{\text{inlet}} \qquad \text{in } [0,200]\times D_{\text{inlet}}, \tag{5.1c}$$

$$p = 0 \qquad \text{in } [0,200]\times D_{\text{outlet}}, \tag{5.1d}$$

$$\boldsymbol{u}\cdot\boldsymbol{n} = \boldsymbol{0} \qquad \text{in } [0,200]\times \left(D_{\text{walls}}\cup D_{\text{top}}\right), \tag{5.1e}$$

$$\boldsymbol{u} = \boldsymbol{0} \qquad \text{in } [0,200]\times (D_{\text{bottom}}\cup B(\boldsymbol{z})), \tag{5.1f}$$

$$\boldsymbol{u} = \boldsymbol{0} \qquad \text{in } \{0\}\times D. \tag{5.1g}$$

The vector normal to a surface at the point considered is noted generically $\boldsymbol{n}$. The properties of the fluid and dimensions of the building are given in table 2. The Reynolds number is calculated using the major axis at the base of the building as the characteristic length. The wind inflow at the inlet $\boldsymbol{u}_{\text{inlet}}$ is described in section 5.2. ◊

The mechanical moment created at the centre O of the base of the building $B(\boldsymbol{z})$ by the fluid pressure is defined as

$$\boldsymbol{M}(\boldsymbol{z}) = \int_{B(\boldsymbol{z})} (\boldsymbol{x}-O)\times p(\boldsymbol{x})\boldsymbol{n}(\boldsymbol{x})\,\mathrm{d}S(\boldsymbol{x}) \tag{5.2}$$

The details of the numerical resolution of problem 3 can be found in deliverable 7.4: spatial and temporal discretisation, solver, etc..

Table 2: Fluid and building properties

Quantity	Notation	Value	Unit
Air density	ρ	1.225	$\mathrm{kg\,m^{-3}}$
Air dynamic viscosity	μ	$1.846 \cdot 10^{-5}$	$\mathrm{kg\,m^{-1}\,s^{-1}}$
Gravity acceleration	g	9.81	$\mathrm{m\,s^{-2}}$
Height of the building	H	180	m
Major base axis		45	m
Minor base axis		30	m
Reynolds number	Re	$9.7 \cdot 10^{7}$	

5.2 Problem of optimisation under uncertainties

We define the QoI Q as the temporal average of the squared magnitude of the base moment:

$$Q(\boldsymbol{z}) := \langle \|\boldsymbol{M}(\boldsymbol{z})\|^2 \rangle_{50,200} \tag{5.3}$$

with $\boldsymbol{M}$ defined in (5.2). We seek the optimal design with respect to the CVaR of Q as the solution of problem 4.

PROBLEM 4 (CVaR-minimisation). The QoI here is the temporal average of the squared magnitude of the base moment defined in (5.3): Q. It is a real-valued random variable whose CDF is assumed to be continuous. The optimal design $\boldsymbol{z}_\star$ that we seek is such that

$$(\boldsymbol{z}_\star, s_\star) := \operatorname{argmin}\{\mathbb{E}(\phi(Q(\boldsymbol{z}), s)) : \boldsymbol{z} \in \mathcal{A};\ s \in \mathbb{R}\}$$

where ϕ is as in definition 1:

$$\phi(Q(\boldsymbol{z}), s) := s + \frac{(Q(\boldsymbol{z}) - s)^+}{1 - \tau},$$

and we define

$$J(\boldsymbol{z}) := \min \mathbb{E}(\phi(Q(\boldsymbol{z}), \cdot)) \tag{5.4}$$

with $\mathcal{A} := [0, 2\pi[\times]0, +\infty[\subset \mathbb{R}^2$ the set of admissible designs. $\Diamond$

The uncertainty on $Q(\boldsymbol{z})$ comes from the wind inflow at the inlet $\boldsymbol{u}_{\text{inlet}}$. It is modelled as the superposition of a time-independent profile $\overline{\boldsymbol{u}}_{\text{inlet}}$ and time-dependent fluctuations $\tilde{\boldsymbol{u}}_{\text{inlet}}$:

$$\boldsymbol{u}_{\text{inlet}} := \overline{\boldsymbol{u}}_{\text{inlet}} + \tilde{\boldsymbol{u}}_{\text{inlet}}.$$

The fluctuations follow the model from Mann 1998 and are generated by the method described in deliverable 7.3. The mean profile depends on the height x_3 as

$$\overline{\boldsymbol{u}}_{\text{inlet}}(x_3) = \frac{w}{\kappa} \ln\left(1 + \frac{x_3}{\overline{x}_3}\right) \begin{pmatrix} \cos\theta \\ \sin\theta \\ 0 \end{pmatrix}.$$

Besides the constant of von Karmán $\kappa \approx 0.41$, the model above features three parameters treated in this work as random variables: (I) the 'roughness height' $\overline{x}_3$; (II) the angle of incidence of the wind θ; (III) the reference wind velocity w. The roughness height affects the shape of the mean profile to represent the effect of obstacles close to the ground (relief, vegetation, buildings, etc.); it follows the uniform distribution $\mathcal{U}(0.01\,\text{m}, 0.1\,\text{m})$ which corresponds to an open country as per Joint committee on structural safety 2001. For this setting, w and θ are dependent. Their relation is modelled from historical measurements of mean wind velocities and directions $(\|\overline{u}(10)\|, \theta)$ collected at 10 m above the ground in Basel, Switzerland, from 2010-01-01 to 2015-12-31; from meteoblue 2015. This data is summarised as a wind rose in the left of figure 15. The reference velocity is calculated from these measurements as $w = \kappa\|\overline{u}(10)\|/\ln(10/\overline{x}_3 + 1)$. The joint distribution of w and θ is then fitter on this empirical data following a copula-based model (see Sklar 1959; Nelsen 2007), yielding the wind rose on the right of figure 15; more details are available in deliverable 7.4. It is visible in figure 15 that $\theta \in [0, 2\pi[$, however in the simulations D_{inlet} is fixed; the wind is kept blowing towards the building by rotating the building, instead of changing the origin of the wind inflow. This rotation is done for every realisation of θ, so that in practice θ parameterise the rotation of the building.

5.3 Optimisation method

The solution to problem 4 is sought by an algorithm alternating direct minimisation and gradient descent, similarly to the strategy adapted in algorithm 1 of section 2.2. For every design $\boldsymbol{z} \in \mathbb{R}^2$, the evaluation of $J(\boldsymbol{z})$ requires solving a unidimensional minimisation problem. Starting from a given initial design $\boldsymbol{z}_0 \in \mathcal{A}$, the new design at every iteration $k \in \mathbb{N}$ is defined as

$$\boldsymbol{z}_k := \boldsymbol{z}_{k-1} - \gamma_{k-1} \nabla J(\boldsymbol{z}_{k-1}).$$

As discussed in previous sections, the exact gradient $\mathbb{E}(\nabla J(\boldsymbol{z}_{k-1}))$ is generally not accessible and has to be estimated. In this section we present the estimation of $\nabla J(\boldsymbol{z}_{k-1})$, then the estimation of the expectation will be discussed in section 5.4.

Two limitations are imposed by the turbulent fluid application described in section 5.1: (I) the chaotic behaviour of the QoI with respect to mesh refinement

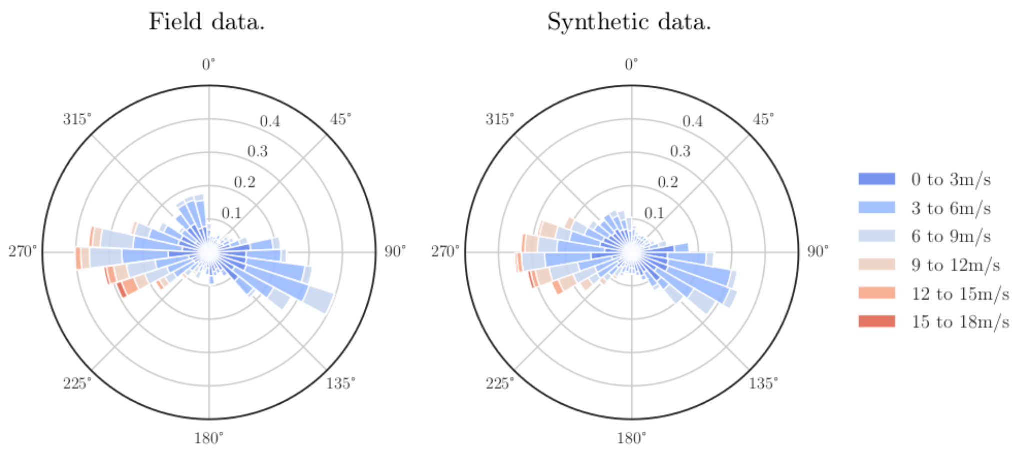


Figure 15: Wind rose based on data from meteoblue 2015

precludes the use of a MLMC method; (II) adjoint-based sensitivities are not available to compute samples of $\nabla J(\cdot)$. Limitation (I) will be addressed in section 5.4. Limitation (II) is addressed with a finite-difference estimation. We define the difference operator for any direction $\boldsymbol{d} \in \mathbb{R}^2 \ \{\boldsymbol{0}\}$,

$$\Delta_{\boldsymbol{d}} J(\boldsymbol{z}) := \frac{J(\boldsymbol{z}+\boldsymbol{d}) - J(\boldsymbol{z})}{\|\boldsymbol{d}\|}.$$

Given a stencil size $h \in \,]0, +\infty[$ we consider the first-order, forward difference scheme

$$\forall i \in \{1,2\}, \qquad \mathrm{D}\, J(\boldsymbol{z})(\boldsymbol{e}_i) \approx \Delta_{h\boldsymbol{e}_i} J(\boldsymbol{z}) = \Delta_{h\boldsymbol{e}_i} J(\boldsymbol{z}).$$

Applying this scheme in both directions $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ with respective stencil sizes $(h_1, h_2) =: \boldsymbol{h}$ yields an approximation of the gradient noted

$$\Delta_{\boldsymbol{h}} J(\boldsymbol{z}) := \boldsymbol{e}_1 \Delta_{h_1\boldsymbol{e}_1} J(\boldsymbol{z}) + \boldsymbol{e}_2 \Delta_{h_2\boldsymbol{e}_2} J(\boldsymbol{z}) \approx \nabla J(\boldsymbol{z}).$$

This entails to evaluate J for three different designs.

The sizes h_1 and h_2 define the stencil and parameterise the estimation $\Delta_{\boldsymbol{h}} J(\boldsymbol{z})$; they are kept fixed for the whole optimisation. These sizes are not straightforward to choose: although a small value should yield a more accurate estimation of the derivative, small step sizes were observed to lead to arbitrarily high and unstable gradient estimations due to the chaotic behaviour of Q. Moreover, h_i also affects the variance of sample estimations of $\Delta_{h_i\boldsymbol{e}_i} J(\boldsymbol{z})$; a higher variance makes an accurate estimation more expensive (see section 5.4).

5.4 Adaptive sample-average approximation

The discretisation of the probability measure – and approximation of the expectation thereof – is done by an adaptive sample-average approximation (SAA). The reader can find a description of the non-adaptive SAA in deliverable 6.3, §2.3.2. Here, the number of samples in each optimisation step is adapted with the goal to find a descent direction by estimating the gradient of the objective function. A relatively small set of events is chosen initially and, before each subsequent iteration, an assessment is made whether the computed gradient is likely to decrease the value of the objective function. If it is not, new events are added to the current set.

Due to limitation (I) mentioned in section 5.3, the expectation will be estimated using a single-level MC estimator. For a random variable $X \in \mathrm{L}^1(\Omega, \mathbb{R}^d)$ and a finite set of elementary events $S \subset \Omega$, the expectation of X is estimated as

$$\frac{1}{|S|}\sum_{\omega\in S} X(\omega) =: \mu_S(X) \approx \mathbb{E}(X),$$

where $|S|$ is the cardinal of the set. If $|S| \leqslant 2$, the variance of X is estimated as

$$\frac{1}{|S|-1}\sum_{\omega\in S}(X(\omega) - \mu_S(X))^2 =: \varsigma_S(X) \approx \mathbb{V}\mathrm{ar}(X).$$

We extend this notation to J defined in (5.4) as

$$\mu_S(J(z)) := \mu_S(\phi(Q(\boldsymbol{z}), t_z)) \tag{5.5}$$

and

$$\varsigma_S(J(z)) := \varsigma_S(\phi(Q(\boldsymbol{z}), t_z))$$

with

$$t_z := \operatorname{argmin}\{\mu_S(\phi(Q(\boldsymbol{z}), t)) : t \in \mathbb{R}\}.$$

Similarly, we will note $\forall i \in \{1, 2\}$

$$\mu_S\big(\Delta_{h_i \boldsymbol{e}_i} J(\boldsymbol{z})\big) := \frac{\mu_S(J(\boldsymbol{z} + h_i\boldsymbol{e}_i)) - \mu_S(J(\boldsymbol{z}))}{h},$$
$$\varsigma_S\big(\Delta_{h_i \boldsymbol{e}_i} J(\boldsymbol{z})\big) := \frac{\varsigma_S\big(\phi(Q(\boldsymbol{z}), t_{\boldsymbol{z}+h_i\boldsymbol{e}_i}) - \phi(Q(\boldsymbol{z}), t_z)\big)}{h};$$

and accordingly

$$\mu_S(\Delta_{\boldsymbol{h}} J(\boldsymbol{z})) := \boldsymbol{e}_1\mu_S\big(\Delta_{h_1\boldsymbol{e}_1} J(\boldsymbol{z})\big) + \boldsymbol{e}_2\mu_S\big(\Delta_{h_2\boldsymbol{e}_2} J(\boldsymbol{z})\big) \approx \Delta_{\boldsymbol{h}} J(\boldsymbol{z}) \approx \nabla J(\boldsymbol{z});$$
$$\varsigma_S(\Delta_{\boldsymbol{h}} J(\boldsymbol{z})) := \varsigma_S\big(\Delta_{h_1\boldsymbol{e}_1} J(\boldsymbol{z})\big) + \varsigma_S\big(\Delta_{h_2\boldsymbol{e}_2} J(\boldsymbol{z})\big).$$

As recommended in deliverable 6.4, § 5.3, the samples of Q share the same events S across the stencil $(\boldsymbol{z}, \boldsymbol{z}+h_1\boldsymbol{e}_1, \boldsymbol{z}+h_2\boldsymbol{e}_2)$. In other words, the pairs of random variables $(Q(\boldsymbol{z}), Q(\boldsymbol{z}+h_1\boldsymbol{e}_1))$ and $(Q(\boldsymbol{z}), Q(\boldsymbol{z}+h_2\boldsymbol{e}_2))$ are computed from the same realisations of the input uncertainty (i.e. wind inflow); thus $Q(\boldsymbol{z})$ is sampled $|S|$ times instead of $2|S|$.

The norm test delivers a posteriori control of the variance of the sample gradient $\mu_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z}))$, for a given design $\boldsymbol{z} \in \mathbb{R}^2$. The approach we follow to assess a potential descent direction is based on the 'norm test' introduced by Byrd et al. 2012. It is built around the observation that $\mu_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z}))$ is a descent direction for J at $\boldsymbol{z}$ if

$$\|\mu_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z})) - \mathbb{E}(\nabla\, J(\boldsymbol{z}))\|_2^2 \leqslant \eta^2 \|\mu_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z})\|_2^2. \tag{5.6}$$

where $\eta \in [0,1[$ is a constant chosen to control the pace of growth of the event set. As stated before, computing $\mathbb{E}(\nabla\, J(\boldsymbol{z}))$ is infeasible. However, the expectation of the left-hand side of (5.6) can be approximated as

$$\mathbb{E}(\|\mu_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z})) - \mathbb{E}(\nabla\, J(\boldsymbol{z}))\|_2^2) \approx \|\mathbb{V}\mathrm{ar}(\mu_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z})))\|_1 \approx \frac{\varsigma_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z}))}{|S|}$$

Using this expression, we arrive at the sample size condition

$$\frac{\varsigma_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z}))}{|S|} \leqslant \eta^2 \|\mu_S(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z}))\|_2^2. \tag{5.7}$$

Of course, (5.7) cannot be expected to always be satisfied. After an iteration k where it is violated for the event set S_k, the size of the next event set S_{k+1} is chosen as

$$|S_{k+1}| = \left\lceil \frac{\varsigma_{S_k}(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z}))}{\eta^2 \|\mu_{S_k}(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z}))\|_2^2} \right\rceil > |S_k|.$$

On the other hand, if (5.7) is satisfied, the event set remains unchanged.

The benefit of this strategy is that a small initial sample size allows for a fast progress towards the optimal design if the initial guess $\boldsymbol{z}_0$ is poor and the optimisation error dominates all other errors; then the sample size grows adaptively to keep the sampling error $\mathbb{V}\mathrm{ar}\big(\mu_{S_k}(\Delta_{\boldsymbol{h}}\, J(\boldsymbol{z}_k))\big)$ in line with the optimisation error[3] $|J(\boldsymbol{z}_\star) - J(\boldsymbol{z}_k)|$ (see Beck and Teboulle 2009). The strategy proposed in algorithm 1 of section 2.2 pursues the same objective with error estimators specific to the CVaR. It has been shown by Byrd et al. 2012 and Bollapragada et al. 2018 that the exact norm test (5.6) gives optimal convergence rates for convex objective functions; moreover it was found robust enough to efficiently deal with many non-convex problems, e.g. in applications to machine learning.

[3]We recall that, if J is strongly convex, then $\|\nabla\, J(\boldsymbol{z}_k)\|^2 \gtrsim |J(\boldsymbol{z}_k) - \min J|$.

5.5 Adaptive optimisation algorithm

The previous sections yield algorithm 2. It is similar to the stochastic optimisation method introduced more briefly in deliverable 6.4, § 5.3 for the minimisation of the expectation; the method described here minimises the CVaR and uses an adaptive SAA. Each evaluation of J in line 8 involves solving a unidimensional minimisation problem as per (5.5) and (5.5); this is performed with Brent's method from Brent 1972. In this application, the step size γ_k at each iteration k is set to a constant value γ. Its value was chosen based on previous experiments on the same problem, and is of the order of magnitude of $\|\nabla J(z_0)\|^{-1}$. Likewise, a specific study was carried out to choose the stencil sizes (h_1, h_2). The algorithm stops at the end of iteration k if the stagnation condition

$$\frac{J(\boldsymbol{z}_{k-1}) - J(\boldsymbol{z}_{k-2})}{J(\boldsymbol{z}_k) + \delta} < \varepsilon$$

is satisfied, with δ and ε set by the user.

Algorithm 2: Gradient descent with finite differences and SAA

```
1  INPUT: $\boldsymbol{z}_0$, $\gamma$, $\varepsilon$, $\delta$, $\eta$
2  INITIALISE: $k := 0$, $E := \varepsilon + 1$
3  WHILE $E > \varepsilon$ DO
4      FOR $\omega \in S_k$ DO
5          FOR $\zeta \in \{\boldsymbol{z}_k, \boldsymbol{z}_k + h_1\boldsymbol{e}_1, \boldsymbol{z}_k + h_2\boldsymbol{e}_2\}$ DO
6              Re-mesh for geometry $(\boldsymbol{\zeta}, \theta(\omega))$
7              Compute sample $Q(\boldsymbol{\zeta})(\omega)$
8      Compute gradient estimation $\boldsymbol{g} := \mu_{S_k}(\Delta_{\boldsymbol{h}} J(\boldsymbol{z}))$
9      IF $\varsigma_S(\Delta_{\boldsymbol{h}} J(\boldsymbol{z})) \leqslant \eta^2 |S| \left\|\mu_{S_k}(\Delta_{\boldsymbol{h}} J(\boldsymbol{z}))\right\|_2^2$ THEN $S_{k+1} := S_k$
10     ELSE
11         $n := \left\lceil \varsigma_{S_k}(\Delta_{\boldsymbol{h}} J(\boldsymbol{z}))\eta^{-2}\|\mu_{S_k}(\Delta_{\boldsymbol{h}} J(\boldsymbol{z}))\|_2^{-2}\right\rceil - |S_k|$
12         Draw events $S' \in \Omega^n$ and set $S_{k+1} := S_k \cup S'$
13     Update design $\boldsymbol{z}_{k+1} := \boldsymbol{z}_k - \gamma\boldsymbol{g}$
14     $k := k + 1$
15     Update stagnation
        $E := \mu_{S_{k-1}}(J(\boldsymbol{z}_{k-1})) - \mu_{S_{k-2}}(J(\boldsymbol{z}_{k-2}))/(\mu_{S_k}(J(\boldsymbol{z}_k)) + \delta)$
16 RESULT: $\boldsymbol{z}_k$, $J(z_k)$
```

The practical implementation, application and results of algorithm 2 for problem 4 are reported in detail in deliverable 7.4. As an example, this numerical experiment uses the following input values:

τ	h_1	h_2	η	δ	ε
0.9	$\frac{\pi}{60}$	0.1 m	0.5	10^{-8}	0.01

A notable limitation of the method presented here is its complexity with respect to the dimension of the design space. Due to the use of finite differences to estimate the gradient of the objective function, the cost of estimating J at every point of the stencil becomes quickly prohibitive as the dimension increases (see line 5 of algorithm 2). This curse of dimensionality could be solved by computing the sensitivities from the solution to an adjoint problem, as was done in sections 3 and 4. However, the computation of this adjoint solution remains a challenge for turbulent flows.

Another possible improvement mentioned in the conclusion of deliverable 6.4 is the use of a MFMC method to reduce the cost of estimating J. This approach has been investigated and applied to a fluid flow almost identical to problem 3 in deliverable 5.5, § 3, outside the context of OUU.

6 Conclusion

6.1 Stochastic optimisation methods for wind engineering

This report presented the latest research from ExaQUte for optimisation under uncertainties (OUU) of the conditional value at risk (CVaR) from the theory to a variety of applications. It began with a theoretical section combining a method of uncertainty quantification (UQ) from deliverable 5.5, § 2 with the methodology developed for OUU through deliverable 6.2, deliverable 6.3 and deliverable 6.4.

This led to a gradient-descent algorithm with an estimation of the gradient by multi-level Monte Carlo (MLMC) with controlled accuracy. The rest of the report consisted of three applications of increasing challenges. Each of these comprised numerical experiments leveraging distributed computing through the open-source software stack described in the introduction, in which the methods of the ExaQUte project are implemented.

The first application was a comparatively light stochastic differential equation (SDE): the FitzHugh–Nagumo oscillator with random forcing; an extension of a previous benchmark, the Van der Pol oscillator. This first problem, with readily available adjoint sensitivities, allowed to demonstrate a direct application of the algorithm proposed in the previous section. This algorithm evinced the predicted behaviour, with a complexity of the gradient estimation at least as good as expected and conservative but reliable error estimation.

The second application was the shape optimisation of an airfoil so as to maximise the CVaR of its lift coefficient under uncertain boundary conditions. The derivation of the adjoint sensitivities was described, so as to allow a high-dimensional parameterisation of the airfoil profile. Given the constrained nature of the optimisation problem, a trust-region algorithm was employed. The performance was then demonstrated in a numerical experiment on a supercomputer, with adaptive remeshing for the changing geometry. The behaviour of the optimisation algorithm was illustrated, including statistics of the output uncertainties before and after optimisation.

The third and last application presented was the optimisation of the shape of a twisted, tapered building under uncertain turbulent wind, so as to minimise the CVaR of the mechanical moment at its base. This is the most challenging benchmark of the ExaQUte project and was the object of a previous discussion in ibid. Its study, implementation, and numerical resolution constitute the major topic of deliverable 7.4. It was briefly described here so as to highlight the various challenges it raises for the OUU methods used. The method chosen to perform the optimisation was then presented.

6.2 Optimisation under uncertainties

The major challenges of this work package were the use of a risk measure informative on failure states, the difficulties inherent to the engineering applications considered, and the integration of methods and tools developed in other work packages of the project.

The focus on the CVaR as a risk measure began at the very inception of the project, which allowed to make considerable progress in that direction. The derivation of its sensitivities and their regularity issues were studied in deliverable 6.2 and deliverable 6.3. A suitable gradient-descent algorithm was then devised in ibid. and deliverable 6.4. Finally, the efficient and reliable estimation of its gradient has been a common thread from deliverable 6.3 to deliverable 6.5.

As far as OUU is concerned, the challenges of the target engineering applications of this project included a high-dimensional design space; unsteady, expensive computational fluid dynamics (CFD) simulations; lastly, a turbulent fluid flow. Unsteady applications were studied in both deliverable 6.4 and deliverable 6.5, and improvements in cost-efficiency were contributed by many other work packages of the projects. The most visible improvement is likely the UQ methods, well integrated in the optimisation method. The computation of shape sensitivities by adjoint solutions removes the complexity of a high-dimensional design space, as was shown in the optimisation of the airfoil. Unfortunately, these solutions were not available for all applications, which led to estimating the sensitivities by other methods more vulnerable to this curse of dimensionality. This limitation is related to the turbulence of the flow, whose chaotic behaviour curtailed or even cancelled the efficiency of several techniques used in this project. Such applications have been tackled in the current report and in deliverable 7.4, yet they could not benefit from some of the more advanced methods developed in the project. Several research directions have been pursued to improve on the methods used here, e.g. in deliverable 5.5 and deliverable 3.3; however, these developments are more recent and not yet integrated in this work package.

This points finally to the place of this work package in the workflow of the project: although the capstone is undeniably the work package 'Application to robust shape optimisation of structures under wind loads', the work package 'Optimisation under uncertainties' comes close second as the point of convergence of the research contributions. Particularly, deliverable 6.4 and deliverable 6.5 evince the integration of numerous beneficial developments from the other work packages of the project: adaptive mesh refinement (deliverable 2.5); parallel computing, task scheduling and resource management (deliverable 4.5); uncertainty quantification (deliverable 5.5); etc..

6.3 Beyond the ExaQUte project

As pointed out above, OUU was the work packages gathering synergistically most of the research efforts of ExaQUte. This integration of varied contributions into both methods and tools is a significant development in itself. Due to the concurrent developments and advancements of the different work packages, however, not all of the latest developments have been integrated into the work package on OUU. The most obvious is perhaps the study of multi-fidelity Monte Carlo (MFMC) in deliverable 5.5, which could improve on the efficiency of the Monte Carlo (MC) method currently used for turbulent flows. Another promising progress is the work on temporal discretisation and parallelisation in deliverable 3.3 and deliverable 3.4. These could constitute rewarding research collaborations in the future.

More generally, further time series analysis specific to UQ could bring various improvements to this work on OUU. For example, the progressive decorrelation of the solutions between two different spatial resolutions has been one of the challenges of turbulent flows which prompted time-correlation studies and encouraged to consider fluid simulations of shorter duration. Similarly, leveraging varying time resolutions and durations to accelerate statistical estimation has been mentioned as a promising lead at several stages of the project. Lastly, the CVaR is a risk measure defined for real-valued random variables, which were typically temporal averages in the studies reported here. A variation of this risk measure devised for a whole time series could be considered, following e.g. the idea from deliverable 6.4, § 3.3; its merit has yet to be investigated. This would require to adapt the UQ and OUU methodology accordingly but could provide better insights for engineering design.

Finally, to improve the performance of the optimisation algorithm itself one could look at faster gradient-based methods, e.g. Newton-like approaches. These typically make use of information on higher derivative of the objective function, which raises issues for non-smooth risk measure such as the CVaR. Two regularisation techniques for the CVaR were discussed in deliverable 6.3 for this very purpose.

References

General

Badia, Rosa M., J. Conejero, C. Diaz, J. Ejarque, D. Lezzi, F. Lordan, C. Ramon-Cortes and R. Sirvent (Dec. 2015). 'COMP Superscalar, an interoperable programming framework'. In: *SoftwareX* 3–4. DOI: 10.1016/j.softx.2015.10.004.

Beck, Amir and Marc Teboulle (2009). 'A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems'. In: *SIAM Journal on Imaging Sciences* 2.1, pp. 183–202. DOI: 10.1137/080716542.

Bollapragada, Raghu, Richard Byrd and Jorge Nocedal (2018). 'Adaptive sampling strategies for stochastic optimization'. In: *SIAM Journal on Optimization* 28.4, pp. 3312–3343.

Brent, R. P. (Dec. 1972). 'An Algorithm with Guaranteed Convergence for Finding a Zero of a Function'. In: *Algorithms for minimization without derivatives.* 1st ed. Englewood Cliffs, N.J: Prentice-Hall. Chap. 4. ISBN: 9780130223357.

Byrd, Richard H., Gillian M. Chin, Jorge Nocedal and Yuchen Wu (2012). 'Sample size selection in optimization methods for machine learning'. In: *Mathematical programming* 134.1, pp. 127–155.

Centre, Barcelona Supercomputing, *COMP Superscalar* version 2.9.0. URL: https://compss.bsc.es/.

Dadvand, P., R. Rossi, M. Gil, X. Martorell, J. Cotela, E. Juanpere, S.R. Idelsohn and E. Oñate (July 2013). 'Migration of a generic multi-physics framework to HPC environments'. In: *Computers & Fluids* 80, pp. 301–309. DOI: 10.1016/j.compfluid.2012.02.004.

Dadvand, Pooyan, Riccardo Rossi and Eugenio Oñate (July 2010). 'An Object-oriented Environment for Developing Finite Element Codes for Multi-disciplinary Applications'. In: *Archives of Computational Methods in Engineering* 17.3, pp. 253–297. DOI: 10.1007/s11831-010-9045-2.

Dapogny, C., C. Dobrzynski and P. Frey (Apr. 2014). 'Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems'. In: *Journal of Computational Physics* 262, pp. 358–378. ISSN: 10902716. DOI: 10.1016/j.jcp.2014.01.005.

Dapogny, Charles, Cécile Dobrzynski, Pascal Frey and Algiane Froehly, *MMG* version 5.6.0, 5th Nov. 2021. INRIA. LIC: GNU Lesser General Public License. VCS: https://github.com/MmgTools/mmg, SWHID: ⟨swh:1:rev:889d408419b5c48833c249695987cf6ec699d399⟩.

Davari, M., R. Rossi, P. Dadvand, I. López and R. Wüchner (May 2019). 'A cut finite element method for the solution of the full-potential equation with an embedded wake'. In: *Computational Mechanics* 63.5, pp. 821–833. ISSN: 01787675. DOI: 10.1007/s00466-018-1624-3.

FitzHugh, Richard (1961). 'Impulses and physiological states in theoretical models of nerve membrane'. In: *Biophysical journal* 1.6, pp. 445–466.

Hodgkin, Alan L and Andrew F Huxley (1952). 'A quantitative description of membrane current and its application to conduction and excitation in nerve'. In: *The Journal of physiology* 117.4, pp. 500–544.

Joint committee on structural safety (May 2001). 'Wind load'. In: *Probabilistic model code*. Vol. 2.13. ISBN: 978-3-909386-79-6.

Krumscheid, Sebastian and Fabio Nobile (2018). 'Multilevel Monte Carlo Approximation of Functions'. In: *SIAM/ASA Journal on Uncertainty Quantification* 6.3, pp. 1256–1293. DOI: 10.1137/17M1135566.

Lordan, Francesc, Enric Tejedor, Jorge Ejarque, Roger Rafanell, Javier Álvarez, Fabrizio Marozzo, Daniele Lezzi, Raül Sirvent, Domenico Talia and Rosa M. Badia (Sept. 2013). 'ServiceSs: An Interoperable Programming Framework for the Cloud'. In: *Journal of Grid Computing* 12.1, pp. 67–91. DOI: 10.1007/s10723-013-9272-5.

Mann, Jakob (1998). 'Wind field simulation'. In: *Probabilistic Engineering Mechanics* 13.4, pp. 269–282. ISSN: 0266-8920. DOI: 10.1016/S0266-8920(97)00036-2.

Mataix, Vicente et al., *Kratos Multiphysics* version 9.0, Nov. 2021. DOI: 10.5281/zenodo.3234644,

meteoblue (31st Dec. 2015). *Weather history. Basel, Switzerland.* 2010-01-01–2015-12-31. URL: https://www.meteoblue.com/en/weather/archive/export/basel_switzerland_2661604 (visited on 30/11/2021).

Nelsen, Roger B. (2007). *An introduction to copulas.* Springer Science & Business Media.

Nishida, Brian and Mark Drela (1995). 'Fully simultaneous coupling for three-dimensional viscous/inviscid flows'. In: *13th Applied Aerodynamics Conference*, pp. 355–361. DOI: 10.2514/6.1995-1806.

Rockafellar, R. Tyrrell and Johannes O. Royset (6th Mar. 2015). 'Engineering Decisions under Risk Averseness'. In: *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering* 1.2, p. 04015003. DOI: 10.1061/AJRUA6.0000816.

Rockafellar, Tyrrell R. and Stanislav Uryasev (1st Apr. 2000). 'Optimization of conditional value-at-risk'. In: *Journal of risk* 2.3, pp. 21–41. DOI: 10.21314/JOR.2000.038.

Shapiro, Alexander, Darinka Dentcheva and Andrzej Ruszczyński (2009). *Lectures on stochastic programming: modeling and theory.* SIAM.

Sklar, M. (1959). 'Fonctions de repartition a n dimensions et leurs marges'. In: *Publ. inst. statist. univ. Paris* 8, pp. 229–231.

Tejedor, Enric, Yolanda Becerra, Guillem Alomar, Anna Queralt, Rosa M. Badia, Jordi Torres, Toni Cortes and Jesús Labarta (2017). 'PyCOMPSs: Parallel computational workflows in Python'. In: *International Journal of High Performance Computing Applications* 31.1, pp. 66–82. DOI: 10.1177/1094342015594678.

Uryasev, Stanislav, Sergey Sarykalin, Gaia Serraino and Konstantin Kalinchenko (2010). *VaR vs CVaR in Risk Management and Optimization.* CARISMA conference.

Yuan, Ya-xiang (Sept. 1999). 'A Review of Trust Region Algorithms for Optimization'. In: *ICM99: Proceedings of the Fourth International Congress on Industrial and Applied Mathematics.*

ExaQUte

Amela, Ramon, Quentin Ayoul-Guilmard, Rosa M. Badia, Sundar Ganesh, Fabio Nobile, Riccardo Rossi and Riccardo Tosi, *XMC* version 2.0.0, 10th Nov. 2020. DOI: 10.5281/zenodo.3235832.

Amela, Ramon, Quentin Ayoul-Guilmard, Rosa Maria Badia, Sundar Ganesh, Fabio Nobile, Riccardo Rossi and Riccardo Tosi (30th May 2019). *Release of ExaQUte MLMC Python engine.* Deliverable 5.2. Version 1.0. ExaQUte consortium.

Ayoul-Guilmard, Quentin, Rosa M. Badia, Jorge Ejarque, Sundar Ganesh, Anoop Kodakkal, Fabio Nobile, Marc Núñez, Jordi Pons-Prats, Javier Principe, Riccardo Rossi, Cecilia Soriano and Riccardo Tosi (30th Nov. 2021). *ExaQUte Data.* Data set. Version 1. ExaQUte consortium. Zenodo. DOI: 10.5281/zenodo.5729258.

Ayoul-Guilmard, Quentin, Sundar Ganesh, Anoop Kodakkal, Fabio Nobile and Marc Núñez (14th Dec. 2020). *Report on stochastic optimisation for unsteady problems.* Deliverable 6.4. Version 1.0. ExaQUte consortium. 56 pp.

— (30th Nov. 2021). *Report on stochastic optimisation for wind engineering.* Deliverable 6.5. Version 1.0. ExaQUte consortium.

Ayoul-Guilmard, Quentin, Sundar Ganesh, Sebastian Krumscheid and Fabio Nobile (2021). 'Quantifying uncertain system outputs via the multilevel Monte Carlo method — distribution and robustness measures'. In preparation.

Ayoul-Guilmard, Quentin, Sundar Ganesh and Fabio Nobile (27th July 2020). *Report on stochastic optimisation for simple problems.* Deliverable 6.3. Version 1.3. ExaQUte consortium. 43 pp.

Ayoul-Guilmard, Quentin, Sundar Ganesh, Fabio Nobile, Marc Núñez, Riccardo Rossi and Riccardo Tosi (13th Nov. 2020a). *Report on MLMC for time-dependent problems.* Deliverable 5.4. Version 1.0. ExaQUte consortium. DOI: 10.23967/exaqute.2021.2.005.

— (30th May 2020b). *Report on theoretical work to allow the use of MLMC with adaptive mesh refinement.* Deliverable 5.3. Version 1.0. ExaQUte consortium. 31 pp.

Ayoul-Guilmard, Quentin, Sundar Ganesh, Fabio Nobile, Marc Núñez and Riccardo Tosi (30th Nov. 2021). *Report on the application of MLMC to wind engineering.* Deliverable 5.5. Version 1.0. ExaQUte consortium.

Badia, Rosa Maria, Stanislav Böhm and Jorge Ejarque (30th Nov. 2021). *Framework development and release.* Deliverable 4.5. Version 1.0. ExaQUte consortium.

Bidier, Sami, Ustim Khristenko, Riccardo Tosi, Roland Wuchner, Alexander Michalski, Riccardo Rossi and Cecilia Soriano (28th Feb. 2021). *Report on UQ results and overall user experience.* Deliverable 7.3. ExaQUte consortium. DOI: 10.23967/exaqute.2021.9.002.

Bidier, Sami, Anoop Kodakkal and Ustim Khristenko (30th Nov. 2021). *Final report on stochastic optimization results.* Deliverable 7.4. Version 1.0. ExaQUte consortium.

Böhm, Stanislav, Jakub Beránek and Martin Surkovsky, *Quake* 30th Nov. 2021. IT4I. URL: https://code.it4i.cz/boh126/quake, (visited on 30/11/2021).

Böhm, Stanislav and Jorge Ejarque, *ExaQUte API* 30th Nov. 2021. IT4I, BSC. URL: https://github.com/ExaQUte-project/exaqute-api, (visited on 30/11/2021).

Cirrottola, Luca and Algiane Froehly (Nov. 2019). *Parallel unstructured mesh adaptation using iterative remeshing and repartitioning.* Research Report RR-9307. INRIA Bordeaux, équipe CARDAMOM.

— (30th Nov. 2021a). *Final release of the mesh generation/adaption capaibilities.* Deliverable 2.5. Version 1.0. ExaQUte consortium.

Cirrottola, Luca and Algiane Froehly, *ParMMG* version 1.4.0, 5th Nov. 2021. INRIA. LIC: GNU Lesser General Public License. VCS: https://github.com/MmgTools/ParMmg, SWHID: ⟨swh:1:rev:be8d5242abaeeafe3d6fd70f5a35064b94b518e1⟩.

Drzisga, Daniel, Mario Teixeira Parete and Roland Wüchner (30th July 2018). *Release of ExaQUte API for MLMC.* Deliverable 5.1. ExaQUte consortium. DOI: 10.23967/exaqute.2021.2.026.

Ganesh, Sundar, Quentin Ayoul-Guilmard and Fabio Nobile (30th May 2019). *Report on the calculation of stochastic sensitivities.* Deliverable 6.2. Version 1.1. ExaQUte consortium. 18 pp.

Nobile, Fabio, Rosa Maria Badia, Jorge Ejarque, Luca Cirrottola, Algiane Froehly, Brendan Keith, Anoop Kodakkal, Marc Núñez, Carlos Roig, Riccardo Tosi, Riccardo Rossi, Cecilia Soriano, Sundar Ganesh and Quentin Ayoul-Guilmard (30th Nov. 2020). *Final public release of the solver.* Deliverable 1.4. Version 1.0. ExaQUte consortium. DOI: 10.23967/exaqute.2021.2.009.

Núñez, Marc, Iñigo López, Joan Baiges and Riccardo Rossi (Jan. 2022). 'An embedded approach for the solution of the full potential equation with finite elements'. In: *Computer Methods in Applied Mechanics and Engineering* 388, p. 114244. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2021.114244.

Tosi, Riccardo, Ramon Amela, Rosa M Badia and Riccardo Rossi (2021). 'A Parallel Dynamic Asynchronous Framework for Uncertainty Quantification by Hierarchical Monte Carlo Algorithms'. In: *Journal of Scientific Computing* 89.1, p. 28. ISSN: 1573-7691. DOI: 10.1007/s10915-021-01598-6.

Tosi, Riccardo, Marc Núñez, Ramon Codina, Jordi Pons-Prats, Javier Principe and Riccardo Rossi (30th Nov. 2021a). *Report of ensemble-based parallelism for turbulent flows and release of solvers*. Deliverable 3.3. ExaQUte consortium.

— (30th Nov. 2021b). *Report on the calibration of parallel methods for transient problems in wind engineering*. Deliverable 3.4. ExaQUte consortium.

Aliases

COMPSs	Barcelona Supercomputing Centre, *COMP Superscalar* version 2.9.0. URL: https://compss.bsc.es/.
deliverable 1.4	Fabio Nobile, Rosa Maria Badia, Jorge Ejarque, Luca Cirrottola, Algiane Froehly, Brendan Keith, Anoop Kodakkal, Marc Núñez, Carlos Roig, Riccardo Tosi, Riccardo Rossi, Cecilia Soriano, Sundar Ganesh and Quentin Ayoul-Guilmard (30th Nov. 2020). *Final public release of the solver*. Deliverable 1.4. Version 1.0. ExaQUte consortium. DOI: 10.23967/exaqute.2021.2.009.
deliverable 2.5	Luca Cirrottola and Algiane Froehly (30th Nov. 2021a). *Final release of the mesh generation/adaption capaibilities*. Deliverable 2.5. Version 1.0. ExaQUte consortium.
deliverable 3.3	Riccardo Tosi, Marc Núñez, Ramon Codina, Jordi Pons-Prats, Javier Principe and Riccardo Rossi (30th Nov. 2021a). *Report of ensemble-based parallelism for turbulent flows and release of solvers*. Deliverable 3.3. ExaQUte consortium.
deliverable 3.4	Riccardo Tosi, Marc Núñez, Ramon Codina, Jordi Pons-Prats, Javier Principe and Riccardo Rossi (30th Nov. 2021b). *Report on the calibration of parallel methods for transient problems in wind engineering*. Deliverable 3.4. ExaQUte consortium.
deliverable 4.5	Rosa Maria Badia, Stanislav Böhm and Jorge Ejarque (30th Nov. 2021). *Framework development and release*. Deliverable 4.5. Version 1.0. ExaQUte consortium.

deliverable 5.1	Daniel Drzisga, Mario Teixeira Parete and Roland Wüchner (30th July 2018). *Release of ExaQUte API for MLMC*. Deliverable 5.1. ExaQUte consortium. DOI: 10.23967/exaqute.2021.2.026.
deliverable 5.2	Ramon Amela, Quentin Ayoul-Guilmard, Rosa Maria Badia, Sundar Ganesh, Fabio Nobile, Riccardo Rossi and Riccardo Tosi (30th May 2019). *Release of ExaQUte MLMC Python engine*. Deliverable 5.2. Version 1.0. ExaQUte consortium.
deliverable 5.3	Quentin Ayoul-Guilmard, Sundar Ganesh, Fabio Nobile, Marc Núñez, Riccardo Rossi and Riccardo Tosi (30th May 2020a). *Report on theoretical work to allow the use of MLMC with adaptive mesh refinement*. Deliverable 5.3. Version 1.0. ExaQUte consortium. 31 pp.
deliverable 5.4	Quentin Ayoul-Guilmard, Sundar Ganesh, Fabio Nobile, Marc Núñez, Riccardo Rossi and Riccardo Tosi (13th Nov. 2020b). *Report on MLMC for time-dependent problems*. Deliverable 5.4. Version 1.0. ExaQUte consortium. DOI: 10.23967/exaqute.2021.2.005.
deliverable 5.5	Quentin Ayoul-Guilmard, Sundar Ganesh, Fabio Nobile, Marc Núñez and Riccardo Tosi (30th Nov. 2021). *Report on the application of MLMC to wind engineering*. Deliverable 5.5. Version 1.0. ExaQUte consortium.
deliverable 6.2	Sundar Ganesh, Quentin Ayoul-Guilmard and Fabio Nobile (30th May 2019). *Report on the calculation of stochastic sensitivities*. Deliverable 6.2. Version 1.1. ExaQUte consortium. 18 pp.
deliverable 6.3	Quentin Ayoul-Guilmard, Sundar Ganesh and Fabio Nobile (27th July 2020). *Report on stochastic optimisation for simple problems*. Deliverable 6.3. Version 1.3. ExaQUte consortium. 43 pp.
deliverable 6.4	Quentin Ayoul-Guilmard, Sundar Ganesh, Anoop Kodakkal, Fabio Nobile and Marc Núñez (14th Dec. 2020). *Report on stochastic optimisation for unsteady problems*. Deliverable 6.4. Version 1.0. ExaQUte consortium. 56 pp.
deliverable 6.5	Quentin Ayoul-Guilmard, Sundar Ganesh, Anoop Kodakkal, Fabio Nobile and Marc Núñez (30th Nov. 2021). *Report on stochastic optimisation for wind engineering*. Deliverable 6.5. Version 1.0. ExaQUte consortium.
deliverable 7.3	Sami Bidier, Ustim Khristenko, Riccardo Tosi, Roland Wuchner, Alexander Michalski, Riccardo Rossi and Cecilia Sori-

	ano (28th Feb. 2021). *Report on UQ results and overall user experience*. Deliverable 7.3. ExaQUte consortium. DOI: 10.23967/exaqute.2021.9.002.
deliverable 7.4	Sami Bidier, Anoop Kodakkal and Ustim Khristenko (30th Nov. 2021). *Final report on stochastic optimization results*. Deliverable 7.4. Version 1.0. ExaQUte consortium.
Kratos Multiphysics	Vicente Mataix et al., *Kratos Multiphysics* version 9.0, Nov. 2021. DOI: 10.5281/zenodo.3234644,
MMG	Charles Dapogny, Cécile Dobrzynski, Pascal Frey and Algiane Froehly, *MMG* version 5.6.0, 5th Nov. 2021. INRIA. LIC: GNU Lesser General Public License. VCS: https://github.com/MmgTools/mmg, SWHID: ⟨swh:1:rev:889d408419b5c48833c249695987cf6ec699d399⟩.
ParMMG	Luca Cirrottola and Algiane Froehly, *ParMMG* version 1.4.0, 5th Nov. 2021. INRIA. LIC: GNU Lesser General Public License. VCS: https://github.com/MmgTools/ParMmg, SWHID: ⟨swh:1:rev:be8d5242abaeeafe3d6fd70f5a35064b94b518e1⟩.
Quake	Stanislav Böhm, Jakub Beránek and Martin Surkovsky, *Quake* 30th Nov. 2021. IT4I. URL: https://code.it4i.cz/boh126/quake, (visited on 30/11/2021).
XMC	Ramon Amela, Quentin Ayoul-Guilmard, Rosa M. Badia, Sundar Ganesh, Fabio Nobile, Riccardo Rossi and Riccardo Tosi, *XMC* version 2.0.0, 10th Nov. 2020. DOI: 10.5281/zenodo.3235832.

List of Algorithms

List of Figures

List of Tables

Acronyms

API	application programming interface
CFD	computational fluid dynamics
CDF	cumulative distribution function
HPC	high-performance computing
MC	Monte Carlo
MFMC	multi-fidelity Monte Carlo
MLMC	multi-level Monte Carlo
MPI	Message Passing Interface
MLSG	multi-level stochastic gradient
MSE	mean squared error
OUU	optimisation under uncertainties
UQ	uncertainty quantification
PDE	partial differential equation
ODE	ordinary differential equation
SDE	stochastic differential equation
PDF	probability density function
QOI	quantity of interest
CMLMC	continuation multi-level Monte Carlo
SAA	sample-average approximation
VaR	value at risk
CVaR	conditional value at risk

Abbreviations

i.e.	id est
e.g.	exempli gratia
a.e.	almost every
a.e.	almost everywhere
a.s.	almost surely
i.i.d.	independent and identically-distributed
s.t.	such that
iff.	if and only if
ibid.	ibidem
et al.	et alii
etc.	et cætera
cf.	confer
viz.	videlicet
NB	nota bene