Fast Adjoint-assisted Multilevel Multifidelity Method for Uncertainty Quantification of the Aleatoric Kind

by

Pavanakumar Mohanamuraly

A thesis submitted to the University of London for the degree of
Doctor of Philosophy

School of Engineering and Material Science
Queen Mary, University of London
United Kingdom

July 2019

Acknowledgments

Firstly, I would like to thank Dr. Jens Müller and the European Union for giving me the opportunity to conduct this PhD research work under the Marie Curie IODA[1] Initial Training Network. I extend my thanks to Queen Mary University of London, for hosting me as a PhD student and staff under this project. I thank Dr. Shenren Xu, Associate Professor, Northwestern Polytechnical University (NPU), China for the tremendous support as a friend, colleague, and collaborator. The solver development was a joint effort with Dr. Xu, who implemented the rotational frame of reference and the time marching implicit scheme, which was implemented in parallel by the author. I also like to thank Prof. Tom Vertsraete, Associate Professor, VKI, Belgium for his constant encouragement and his persuasion to choose the LS89 as a test case in this thesis. The mesh for the Sajben diffuser has been provided by Dr. Vertsraete and I thankfully acknowledge it. I also would also like to thank the ESRs in the IODA project Rejish and Orest hosted at QMUL and ESRs at other external institutes for their support. I express my gratitude to Kitware, Lyon, ESI, Essen, and Rolls Royce, Berlin for hosting me in their premises during my secondment. Special thanks to Mitsubishi Heavy Industries (MHI), Japan, for their permission to present the computations of their Turbocharger compressor in the thesis and Mr. Tadashi Kanzaka, MHI for providing the geometry and meshes for the computations. Finally, this thesis is not possible without the support of my family, my mother Usha, wife Vaishanvi and our little boy Ranga, who had to put up with my *Constant Frustration and Disappointment (CFD)* bearing a constant smile. This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 642959.

[1] http://ioda.sems.qmul.ac.uk

Statement of Originality

I, Pavanakumar Mohanamuraly, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

Signature: Pavanakumar Mohanamuraly

Date: November 30, 2018

Abstract

In this thesis an adjoint-based multilevel multifidelity Monte Carlo (MLMF) method is proposed, analysed, and demonstrated using test problems. Firstly, a multifidelity framework using the approximate function evaluation [1] based on the adjoint error correction of Giles et al. [2] is employed as a low fidelity model. This multifidelity framework is analysed using the method proposed by Ng and Wilcox [3]. The computational cost reduction and accuracy is demonstrated using the viscous Burgers' equation subject to uncertain boundary condition.

The multifidelity framework is extended to include multilevel meshes using the MLMF of Geraci [4] called the FastUQ. Some insights on parameters affecting computational cost are shown. The implementation of FastUQ in Dakota toolkit is outlined. As a demonstration, FastUQ is used to quantify uncertainties in aerodynamic parameters due to surface variations caused by manufacturing process. A synthetic model for surface variations due to manufacturing process is proposed based on Gaussian process. The LS89 turbine cascade subject to this synthetic disturbance model at two off-design conditions is used as a test problem. Extraction of independent random modes and truncation using a goal-based principal component analysis is shown. The analysis includes truncation for problems involving multiple QoIs and test conditions.

The results from FastUQ are compared to the state-of-art SMLMC method and the approximate function evaluation using adjoint error correction called the inexpensive Monte Carlo method (IMC). About 70% reduction in computational cost compared to SMLMC is achieved without any loss of accuracy. The approximate model based on the IMC has high deviations for non-linear and sensitive QoI, namely the total-pressure loss. FastUQ control variate effectively balances the low fidelity model errors and additional high fidelity evaluations to yield accurate results comparable to the high fidelity model.

Table of Contents

Acknowledgments **i**

Statement of Originality **ii**

Abstract **iii**

Table of Contents **iv**

List of Figures **x**

List of Tables **xiv**

List of Abbreviations **xvi**

List of Symbols **xviii**

1 Introduction **1**

1.1 Motivation and Context 1

1.1.1 Why Uncertainty Quantification? 1

1.1.2 Challenges to Uncertainty Quantification 2

1.1.3 Some Recent Developments 3

1.1.4 Types of Uncertainties 4

1.1.5 Motivation for Choosing the Aleatoric Kind 5

1.2 Objectives of the Thesis 6

1.3 Thesis Plan 7

1.4 Contributions 8

2 Methods for Quantifying Aleatoric Uncertainties 10

2.1 Important Definitions and Concepts 10

2.1.1 Probability Space 10

2.1.2 Probability Distributions 11

2.1.3 Definition of Mean and Standard Deviation 12

2.1.4 Covariance, Cross-covariance, and Joint Probability 12

2.1.5 Statistical Independence and Covariance 13

2.1.6 Uncertainty Quantification and Model Response 14

2.1.7 Adjoint and Parameter Sensitivity 14

2.2 A Survey of Uncertainty Quantification Methods 17

2.3 The Monte Carlo Family of Sampling Methods 17

2.3.1 Randomly Sampled Monte Carlo 17

2.3.2 Improved and Adaptively Sampled Monte Carlo 19

2.3.3 Control Variate Monte Carlo 20

2.3.4 Multifidelity Control Variate of Ng and Wilcox 20

2.3.5 Multilevel Control Variate Monte Carlo 24

2.3.6 Multilevel Multifidelity Monte Carlo 27

2.3.7 Summary of Monte Carlo Methods 28

2.4 Generalised Polynomial Chaos Expansion 28

2.4.1 Intrusive vs. Non-intrusive Methods 30

2.4.2 Stochastic Collocation vs. gPCE Regression 31

2.4.3 Curse of Dimensionality : Some Cures and Limitations 32

2.5 Summary 33

3 Fast Adjoint-assisted Multifidelity Multilevel UQ 34

3.1 Low Fidelity Models in Multifidelity Monte Carlo 34

3.2 Adjoint Correction as a Low Fidelity Model 36

3.3 Uncertain Viscous Burgers' Equation 38

3.3.1 Analytical Solution to Deterministic Problem 38
3.3.2 Baseline Monte Carlo Results 41
3.3.3 Approximating Model Response using Adjoint Correction 42
3.3.4 Results of the Multifidelity Approach 46
3.4 Adjoint-assisted Multilevel Multifidelity Control Variate (FastUQ) 47
3.4.1 Optimal Sample Allocation between Levels and Models 49
3.4.2 An Insight on Model Correlation and Optimal Sampling 50
3.5 FastUQ implementation in Dakota toolkit 52
3.6 Summary 53

4 Aerodynamic Model and Implementation 55
4.1 Governing Equations in Rotational Frame[2] of Reference 56
4.2 Edge-based Second Order Finite Volume Spatial Discretisation 58
4.2.1 Convective Flux via Roe Approximate Riemann Solver 59
4.2.2 Viscous Flux via Green-Gauss Gradient Formula 59
4.3 Turbomachinery Boundary Conditions 60
4.3.1 Variable Swirl Inlet with Specified Total Conditions 60
4.3.2 Specified Exit Back Pressure 61
4.3.3 Rotating and Stationary No-slip Viscous Wall 61
4.4 Explicit Time Marching with Implicit Preconditioning 62
4.4.1 Preconditioned Runge-Kutta Time Marching Scheme 62
4.4.2 Preconditioner based on Newton-Krylov Method 63
4.4.3 Linearised System Solution using GMRES 64
4.5 Parallel Algorithms 66
4.5.1 Two-Level Halo Partitioning 67
4.5.2 Reordering and Scalable Assumed Partitioning 70
4.5.3 Parallel GMRES with Local ILU(0) Preconditioner 72
4.5.4 Implicit Solver Scalability Results 73

[2]Rotational frame of reference implemented by Xu and extended to parallel non-linear solver by the present author

4.6 Linearised Solvers and their Implementation 74
4.6.1 Discrete Adjoint Solver via Reverse Algorithmic Differentiation . . 74
4.6.2 Tangent Linear Solver via Forward Algorithmic Differentiation . . 76
4.6.3 Adjoint and Tangent Linear Sensitivity Assembly 77
4.7 Non-linear Flow Solver Validation 78
4.7.1 Laminar Flow Over Flat Plate with Zero-pressure Gradient 78
4.7.2 Turbulent Flow Over Flat Plate with Zero-pressure Gradient . . . 79
4.7.3 Sajben Transonic Diffuser 81
4.7.4 VKI LS89 Turbine Cascade 83
4.7.5 Turbocharger Compressor with Vaneless Diffuser[3] 85
4.8 Surface Sensitivity Verification 88
4.8.1 Point-wise Sensitivity Verification 89
4.8.2 Directional Derivative Verification 90
4.9 Summary 92

5 Modelling Geometric Uncertainties 93

5.1 Manufacturing Uncertainties in Turbomachines 93
5.2 State-of-art in Surface Imperfection Models 95
5.3 Background on Stochastic Processes 97
5.3.1 Definition of Stochastic Process 97
5.3.2 Stationarity, Mean, and Covariance Function 98
5.3.3 Gaussian Process 99
5.4 Stochastic Surface Variation Model 99
5.4.1 LS89 Geometry and Surface Variation Model 100
5.5 Model Improvements Using Dimensionality Reduction and Independence . 102
5.5.1 Truncation Using Partial Modal Fraction 103
5.5.2 LS89 Partial Modal Fraction Truncation Results 104
5.5.3 Goal-based Truncation for Multiple QoIs 105
5.5.4 LS89 Goal-based Truncation Results 106

[3] Joint work with Dr. Xu

5.6 FastUQ for Aerodynamic Uncertainties due to Surface Variations 108
5.7 Results and Discussion 110
5.7.1 MUR43 and MUR47 Flow Condition 110
5.7.2 Cost model parameters 112
5.7.3 Effects of G-PCA Truncation 113
5.7.4 Effect of Pilot Sampling 115
5.7.5 Optimal Resource Allocation and MSE Convergence (SMLMC) . . 115
5.7.6 Accuracy of Adjoint Correction 117
5.7.7 FastUQ Results 118
5.8 Summary 121

6 Summary and Recommendations **122**
6.1 Recommendations For Future Work 125
6.1.1 Model Improvements 125
6.1.2 MLMC improvement 126
6.1.3 Comparison with other UQ methods 126
6.1.4 Better than $\mathcal{O}(P^{-1})$ Convergence 126
6.1.5 Solver Improvements 126
6.1.6 New application areas 127

Appendix A Author's publications **128**

Appendix B Inverse Distance Weighted Interpolation Mesh Smoothing and Adjoint **129**
B.1 Mesh Smoothing Implementation 131
B.2 Adjoint of Mesh Smoother 134

Appendix C Algorithmic Differentiation and Seeding **137**
C.1 Forward and Reverse Mode of AD 138
C.2 Adjoining MPI Calls 140

Bibliography **142**

List of Figures

1.1 Double loop Monte Carlo simulation for epistemic uncertainty (from ref. [5]) 5

2.1 Gaussian and joint probability distribution example 12
2.2 Forward propagation of uncertainties 14
2.3 Overview of methods in UQ survey 17
2.4 Sobol sequence, pseudo-random, and latin hypercube points 19
2.5 Contours of ψ for given ratio of number of model evaluations r and control parameter β for (a) $\rho = 0.9$ and (b) $\rho = 0.95$ from ref. [3] 23
2.6 Comparison of total integration points 32

3.1 Classification of low fidelity models for multifidelity UQ [6] 34
3.2 Steady-state solution of viscous Burger‘s equation 39
3.3 Uniform probability distribution for input uncertainty δ 40
3.4 Block diagram of MC method 41
3.5 Convergence of relative error; (i) Random and (ii) LHS sampling 41
3.6 Monte Carlo solution to the viscous Burgers' equation 42
3.7 Comparison of transition point location 44
3.8 Contour plots of ψ for IMC $1-3$ models 45
3.9 Mean squared error (MSE) reduction for different LF models 47
3.10 Percentage reduction in computational cost of MFMC 48
3.11 Contour plot of $\phi(r^{opt})$ and $\Lambda(r^{opt})$ 51
3.12 UQ workflow using Dakota and STAMPS 52

4.1 Dual volume and mesh edges . . . 58
4.2 Sketch illustrating skewness correction stencil . . . 60
4.3 Wall boundary illustration . . . 60
4.4 Piecewise polynomials fit for inlet profile . . . 61
4.5 The (i) Zero-halo and (ii) one-layer halo based partitioning methods illustrated for a sample mesh split into two parts (grey and black); halo/shared nodes denoted by hollow circles and internal nodes denoted by filled circles 66
4.6 Periodic mesh and its graph representation . . . 67
4.7 Communication and computation schedule . . . 68
4.8 Average total communication volume comparison . . . 68
4.9 Increase in problem size and runtime profile data . . . 69
4.10 Actual and assumed partitioning of the rows of a vector . . . 71
4.11 Bandwidth reduction using RCM . . . 72
4.12 Non-linear implicit flow solver scaling . . . 74
4.13 Flat plate mesh, boundaries and partitions . . . 79
4.14 Laminar solution comparison with analytical solution . . . 79
4.15 Turbulent solution comparison with NASA FUN3D results . . . 80
4.16 Sajben diffuser geometry with grid . . . 81
4.17 Sajben diffuser Mach number contour . . . 81
4.18 Comparison of convergence; adjoint and primal solver . . . 82
4.19 Sajben diffuser surface pressure comparison with experiments . . . 83
4.20 Sajben diffuser sectional velocity comparison with experiments . . . 84
4.21 LS89 computational domain . . . 84
4.22 VKI LS89 surface isentropic Mach number distribution . . . 86
4.23 Centrifugal turbocharger compressor model . . . 87
4.24 Summary of results for a speed line efficiency and mass-flow rate . . . 88
4.25 Node perturbation and its volumetric displacement . . . 90
4.26 Verification of nodal sensitivity using finite-difference . . . 90
4.27 Modal perturbation and its volumetric displacement . . . 91

4.28 Verification of nodal sensitivity using finite-difference 92

5.1 Flank and point milling process and sources of noise in manufacturing . . 94
5.2 Measured deviations for integrally bladed rotor 94
5.3 Overview of models for surface imperfection due to manufacturing 95
5.4 Stochastic process example 98
5.5 Surface node distribution and spatial covariance function 101
5.6 Modal fraction for the LS89 turbine cascade first 20 PCA modes 104
5.7 Partial modal fraction for the LS89 turbine cascade first 20 PCA modes . 105
5.8 PCA vs. G-PCA modes comparison 107
5.9 PCA eigenmodes and surface sensitivity 107
5.10 Interpolation of PCA mode #4 from the fine to the medium and coarse meshes 109
5.11 FastUQ workflow for surface variations 109
5.12 Mach number contours for the LS89 cascade 111
5.13 Mean and standard deviation of surface isentropic Mach number 111
5.14 Computational cost comparison of non-linear solution 112
5.15 Computational cost comparison of adjoint solution 113
5.16 Computational cost comparison of tangent solution for mode #4 (fine mesh) 113
5.17 SMLMC results for the LS89 test case MUR47 116
5.18 HF/LF correlation across levels 119
5.19 Percentage reduction in computational cost compared to SMLMC results. 119
5.20 MUR47 mean and standard deviation variation with respect to SMLMC results 119
5.21 MUR43 mean and standard deviation variation with respect to SMLMC results 120
5.22 Computational resource allocation across levels and models 120

6.1 Simulation and experimentally determined tool cutting force (from ref. [7]) 125

2.1 Mesh smoothing test conditions 1 and 2 133

2.2 Mesh smoothing results test condition 1 134
2.3 Mesh smoothing results test condition 2 135

List of Tables

2-A gPCE weight function and orthogonal polynomial 29

3-A Mean, standard deviation and Pearson correlation (model vs. exact) . . . 43

4-A Blocking parallel transformation of GMRES kernels 73

4-B Test conditions for LS89 turbine cascade 85

4-C Adjoint and Tangent sensitivity at surface node P1 89

4-D Adjoint and Tangent sensitivity for surface mode 8 91

5-A VKI LS89 cascade geometry 101

5-B $MUR43$ and $MUR47$ test conditions 101

5-C Ranking of modes for the two QoIs and flow conditions 108

5-D Summary of combinations of UQ methods using multiple levels and models 110

5-E Effect of truncation; 7 and 25 G-PCA modes 114

5-F Effect of pilot sample size; 10 and 20 samples 115

5-G Results of SMLMC with different MSE 116

5-H Comparison of SMLMC and IMC 118

2-A Runtime comparison for different mesh smoother 133

List of Abbreviations

2-D	Two-Dimensional
3-D	Three-Dimensional
AD	Algorithmic Differentiation
AMG	Algebraic Multi-Grid
CDF	Cumulative Density Function
CFD	Computational Fluid Dynamics
CFL	Courant–Friedrichs–Lewy
CMLMC	Continuation Multi-level Monte Carlo
FM	Flank Milling
FOKI	First-Order Jacobian Krylov Implicit
FPI	Fixed-point Iteration
FVM	Finite Volume Method
G-PCA	Goal-based Principal Component Analysis
GF	Gaussian Field
GMRES	Generalized Minimal Residual Method
GP	Gaussian Process
gPCE	Generalised Polynomial Chaos Expansion
HF	High Fidelity
IMC	Inexpensive Monte Carlo
i.i.d	Independent and identically distributed

LF	Low Fidelity
LHS	Latin Hypercube Sampling
MC	Monte Carlo
MCM	Monte Carlo Method
MFMC	Multifidelity Monte Carlo
MFML	Multifidelity Multilevel Monte Carlo
MG	Muti-grid
MLMC	Multilevel Monte Carlo
MPI	Message Passing Interface
MSE	Mean Squared Error
MUSCL	Monotonic Upwind Scheme for Conservation Laws
NK	Newton-Krylov
PBM	Projection Based Method
PCA	Principal Component Analysis
PCE	Polynomial Chaos Expansion
pdf	Probability Density Function
PM	Point Milling
QoI	Quantity of Interest
RANS	Reynolds Averaged Navier Stokes
RK5	Five stage Runge-Kutta method
RMSE	Root mean square error
RV	Random variable
SA	Spalart-Allmaras
SC	Stochastic Collocation
SHL	Single-halo layer
SMLMC	Standard Multilevel Monte Carlo
STAMPS	Source Transformation Adjoint Multi-Purpose Solver
THL	Two-halo layer
UQ	Uncertainty Quantification
ZHL	Zero-halo layer

List of Symbols

Spaces

$\emptyset$	Null or empty set
$\mathcal{B}$	Event space
Ω	Sample space
$(\Omega, \mathcal{B}, P)$	Probability space
P	Probability measure
$\mathbb{R}$	Set of real scalar
$\mathbb{R}^n$	Set of n-dimensional real vectors

Probability

$\mathcal{A}_i$	Arbitrary event in $\mathcal{B}$
$\mathbf{C_x}$	Covariance of $\mathbf{x}$
$\mathbf{C_{xy}}$	Cross-covariance of $\mathbf{x}$ and $\mathbf{y}$
$\mathbb{E}[\mathbf{x}]$ or μ_x	Mean or expectation of x
$P(x_1 \leq x \leq x_2)$	Probability of finding x in range $[x_1, x_2]$
$p(x)$	Probability distribution function of RV x
$p(x, y)$	Joint probability distribution function of RV x and y
ρ	Pearson correlation or coefficient
σ	Standard deviation

$\mathbb{V}ar[\mathbf{x}]$	Variance of x
$X(\omega), Y(\omega)$	Scalar random variables
$\mathbf{X}(\boldsymbol{\omega}), \mathbf{Y}(\boldsymbol{\omega})$	Vector random variables
x, y	Realisation of scalar RV $X(\omega) and Y(\omega)$
$\mathbf{x}, \mathbf{y}$	Realisation of vector RV $\mathbf{X}(\boldsymbol{\omega}) and \mathbf{Y}(\boldsymbol{\omega})$

IMC & Adjoint

A	Linear system model
α	Design parameters
b	Linear forcing function
J	QoI
$L,\ M,\ N$	Dimensions of QoI, state, and design parameters
$R(u, \alpha)$	State equation
u	State variable
u_δ	Perturbed state variable
u_δ^*	Approximation to perturbed state variable
v	Adjoint variable
v_δ	Perturbed adjoint variable
v_δ^*	Approximation to perturbed adjoint variable

Burgers' model

A	Slope at transition location
$\dot{A}$	Adjoint solution
$\bar{A}$	Tangent-linear solution
$a,\ b$	Upper and lower limits of uniform distribution
δ	Uncertain boundary parameter
ν	Viscosity parameter in Burgers' equation

$u(x,t)$	Solution to Burgers' equation
x_0	Transition point location
x	Spatial coordinate

Flow solver

$\mathbf{A}$	Mesh deformation matrix
$\mathcal{B}$	Boundary surface area
CFL	CFL number
c_p	Specific heat capacity (constant pressure)
c_v	Specific heat capacity (constant volume)
$\mathbf{F}$	Flux function
γ	Adiabatic index
J	Cost-function or objective function
κ	Thermal conductivity
$\mathbf{M}$	Preconditioner matrix
M_{isen}	Isentropic Mach number
μ	Viscosity
$\mathbf{n}$	Normal vector
$\vec{\omega}$	Rotational velocity of the moving frame
Pr	Prandtl number
P_{01}	Total-pressure at inlet
P_s	Station static pressure
p	Static pressure
$\mathbf{R}$	Residual vector
R	Gas constant
ρ	Fluid density
$\mathbf{r}$	Position vector in rotational frame
$\mathcal{S}$	Surface area

T	Static temperature
τ	Pseudo-time
$\bar{\tau}, \tau_{ij}$	Fluid stress tensor
$\mathbf{U}$	Fluid state (conservative form)
$\mathbf{u}$	Fluid velocity vector
$\mathbf{u}_r$	Relative velocity in rotational frame
$\mathcal{V}$	Volume
v	Adjoint state
w	Tangent linear state

Geometric Model

$b,\ l$	Height and correlation length of surface perturbation
$\delta(\mathbf{x})$	Magnitude of normal perturbation of surface GP
η_{PCA}	Effectiveness factor of PCA mode
$\mathbf{\Lambda}$	Diagonal matrix with eigenvalues as entries
$\sqrt{\lambda_j}$	Eigen value of j^{th} PCA mode
$\hat{n}$	Normal vector of nominal surface (mean surface)
$\mathbf{T}$	Diagonal matrix with entires ± 1 based on parity of Λ
$\mathbf{U},\ \mathbf{\Sigma},\ \mathbf{V}$	Left, diagonal, and right singular matrix
$\mathbf{x},\ \mathbf{x}_\delta$	Unperturbed and GP perturbed surface
Y	Standard normal distribution with zero mean and unit variance
$\mathbf{z}_j$	Eigen vector corresponding to the j^{th} PCA mode

UQ Methods

$(\cdot)_l$	Level specific quantity
$(\cdot)^{LF}$	Low fidelity quantity
$(\cdot)^{HF}$	High fidelity quantity
$(\cdot)^{opt}$	Optimal value or quantity

a_i	Deterministic function in gPCE expansion
β	Control variate
c	Computational effort
c_l	Ratio of computational cost at level l and finest level N
C_l	Computational cost of level l
C_N^{ML}	Total computational cost of MLMC
d	Dimension of uncertainty parameter
ϵ	Error tolerance (upper bound for $\sqrt{MSE}$)
$f,\ \lambda$	Lagrangian and its multiplier (used in optimisation)
G	Approximate model in control variate
J_p^{MC}	Estimated mean of QoI J using p Monte Carlo samples
J_p^{CV}	Estimated mean of QoI J using p control variate samples
$J_{N,\ p}^{ML}$	Multilevel MC estimate of QoI J
$L,\ l$	Total number of levels and running level index
N	Number of coarse grid levels
N_p	Number of terms in gPCE expansion
n_s	Number of samples for regression gPCE
M	Total number of samples of G (value)
m	Order of chaos polynomial Ψ
q	Total number of samples of G (running index)
P	Total number of samples of J (value)
P_0	Total number of pilot (initial) samples of J (value)
P_l	Total number of samples of J at level l (value)
P_l^{opt}	Optimal number of samples per level
p	Total number of samples from J (running index)
$\Psi(\boldsymbol{\xi})$	Stochastic polynomial in gPCE expansion
ψ	Variance reduction estimator

r	Ratio between the number of high and low fidelity simulations
θ	Error splitting parameter
w	Runtime ratio b/w average computational cost of HF & LF model
$\boldsymbol{\xi}$	Uncertain vector in gPCE
ϕ	Ratio of classical MC and control variate MC MSE estimator
Y_l	Level difference function (high-fidelity)
Z_l	Level difference function (low-fidelity)

Gaussian Process

$\mathbf{C}_{ij}$	Covariance function
$f(x)$	Gaussian process in space x
$\{f_i(x_j)\}$	Samples of Gaussian process $f(x)$
$\mathcal{GP}(\mu_x, \mathbf{C}_{ij})$	Gaussian process with mean μ_x and covariance $\mathbf{C}_{ij}$
$\mu_f(x)$	Mean function
$\sigma_f(x)$	Variance function

Chapter 1

Introduction

1.1 Motivation and Context

1.1.1 Why Uncertainty Quantification?

The world we live in is filled with changes, climate change to technology change (machine learning); some are man-made and some are natural. A key to successful human existence has been the ability to adapt to uncertainty. Of course this is not possible if humans did not have the ability to predict changes or portend major events in an uncertain world.

On a more technical note, the growing need to compress time from design to production and availability of cheap compute power has pushed the reliability of simulation and modelling to depict reality. As a result hand calculations and experiments on prototypes are largely replaced with simulations. This has led to two major developments, (i) extensive use of simulations in design or simulation based designs and (ii) use of simulations to mitigate risk and uncertainty. In fact use of simulations to reduce risk is now considered as a part of the design failure modes and effects analysis (DFMEA) [8] used extensively by the industry. The structural engineering community is a classic example of how risk based design are being leveraged to develop large reliable and fail safe structures [9, 10, 11]. More than two decades of research has been devoted to risk and

reliability by the community. In fact, gas turbine structural design under uncertainty had become a standard practice a decade ago [12]. The same cannot be said about computational aerodynamics community, which according to the author has just started realising its potential use [13, 14, 15] due to ease of access to large compute resources and improvements to simulation tools and methods.

1.1.2 Challenges to Uncertainty Quantification

Three major challenges to uncertainty quantification (UQ) in aerodynamics are (i) computational cost, (ii) availability of fast, accurate and robust methods, and (iii) scalability of computational cost to large number of sources of uncertainty in the data.

Since uncertainty deals with the analysis of many possible outcomes, a large number of evaluations of the system (samples) is warranted and leads to an explosion of the computational cost. For example the classical Monte Carlo method has an error convergence rate of $\mathcal{O}(1/\sqrt{P})$ where, P is the number of samples [16]. This means that quadrupling the number of sampled points halves the error in the statistics.

During the exhaustive analysis of the various samples one has to frequently compute outliers (extreme or fringe cases), which are not usually encountered during regular analysis. The methods used for nominal scenarios can fail catastrophically when applied to such outliers [17]. For example, Mishra et al. [17] finds that the approximate Riemann solver used to solve balance laws are usually *benchmarked* against standard tests. In UQ one encounters extreme data scenarios, which in the eyes of an expert is seemingly impossible but physically plausible and the Riemann solver has to provide robust and accurate results.

Lastly, the third challenge to UQ is the scaling of computational cost to the size of uncertainties, i.e., an increase in the number of sources of uncertainty should not lead to a deterioration in the computational efficiency. The number of sources of uncertainty is mostly problem dependent, which the user usually does not have control over. For example Lange et al. [18] modelled performance variations in turbine blades due to man-

ufacturing variations using independent modes obtained from the principal component analysis of surface coordinate data from a large number of 3-D blade scan measurements. The authors found that in order to capture the standard deviation of the performance one needs to include a minimum of 50% of the principal modes. This gives an estimate of the order of 60 uncertainty parameters for 135 modes.

1.1.3 Some Recent Developments

The recent developments in multilevel [19, 20, 17, 21] multifidelity [3, 22, 23] and multilevel multifidelity [4, 24] Monte Carlo methods have made it possible to manage the cost of computations in UQ. In fact, research in the direction of reliability and risk in aerodynamic design have taken centre stage due to the reduction in computational cost [13, 14, 15]. Multilevel Monte Carlo methods are motivated from multi-grid methods used for solving deterministic partial differential equations. These methods assume the availability of coarse geometric multilevel meshes similar to that of a geometric multi-grid method [25].

Another key enabler to UQ is the adjoint solution. With the availability of Algorithmic Differentiation (AD) tools, automation of discrete adjoint solver development and maintenance has become less of an issue. Adjoints can provide valuable information such as sensitivity of the output to changes in input at a computational cost, which is independent of the number of input parameters [26]. This is an important property UQ can leverage to bring down the cost of computations. For instance goal-based truncation of parameters that have very little or nil effect on the measured output quantity [27, 28] can be employed to reduce the number of uncertain parameters. Ghate [1] proposed an adjoint-based inexpensive Monte Carlo method (IMC), which uses the adjoint error correction of Giles et al. [29], to estimate the output variations of integral quantities and avoids the expensive flow solution for the Monte Carlo samples. One of the IMC variants, IMC 1, can approximate the output using a single non-linear residual evaluation at the perturbed state and adjoint solution obtained at a mean state. IMC 1 method was

shown to be equivalent to a first order moment method [1].

The stability and robustness issues [30, 31, 32, 33] of the adjoint solver had prevented its wide spread use in UQ research. In fact, the aboutFlow[1] Initial Training Network (ITN) and its predecessor FlowHead[2] (a small to medium size focused research project) - both funded by the European Commission were aimed at tackling stability and robustness issues of adjoint solvers for automotive and industrial CFD and optimisation. The present thesis is part of the IODA[3] Initial Training Network (third in the series), which is aimed at application of adjoints to industrial workflows and uncertainty quantification. Recent research on stable non-linear flow and discrete adjoint solvers at QMUL by Xu and Müller [33] has made it possible to have stable convergence of the non-linear flow solver and its adjoint for mildly separated steady flows. Lastly, the QMUL adjoint optimisation group has a strong focus and expertise on multi-grid methods and have developed an in-house mesh coarsening tool HiP for generating coarse multilevel meshes from hybrid unstructured meshes using the element collapsing algorithm of Müller et al [25]. The present thesis leverages on these recent algorithmic and theoretical developments to make UQ practical and affordable.

1.1.4 Types of Uncertainties

Uncertainties are classified into two kinds; (i) *aleatoric* and (ii) *epistemic*. Aleatoric uncertainty is also called stochastic, static, or irreducible uncertainty. They are inherent to a problem or model and, in principle, cannot be reduced by adding more knowledge [34]. They are typically unbiased, i.e., the sample mean is representative of the true mean and with enough number of samples one can obtain a good approximation to the true variation. Therefore, they are readily parametrised using probabilistic models. On the other hand, epistemic uncertainties are due to simplifying model assumptions, missing physical models, or lack of basic knowledge of the system. They are often biased and less amenable to probabilistic modelling.

[1] `http://aboutflow.sems.qmul.ac.uk`
[2] `http://flowhead.sems.qmul.ac.uk`
[3] `http://ioda.sems.qmul.ac.uk`

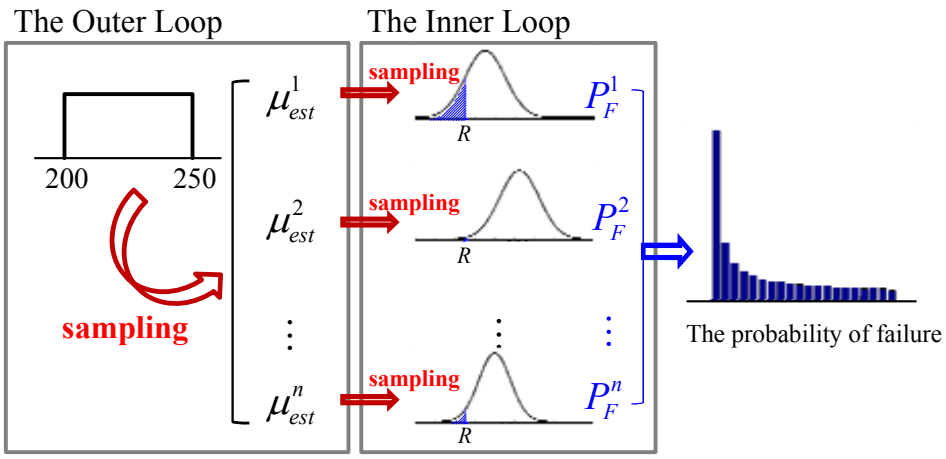


Figure 1.1: Double loop Monte Carlo simulation for epistemic uncertainty (from ref. [5])

Some examples of aleatoric uncertainties include free-stream or background disturbances, fluctuations in measuring instruments due to noise and vibration, small scale unsteadiness due to turbulence, etc. Examples of epistemic uncertainties include assumptions of linear eddy viscosity model [35], assumptions about the flow like parallel flow [36], inviscid/laminar/turbulent flow, constants used in the models, boundary conditions specification (back-pressure measurements can be slightly offset from the actual plane of measurement and averaged over the exit plane).

1.1.5 Motivation for Choosing the Aleatoric Kind

In this thesis, uncertainties of the aleatoric kind are considered and the motivation behind this decision is provided in this section. Aleatoric uncertainties can be represented as stochastic models and one has all the tools and methods of stochastic modelling at ones disposal. Epistemic uncertainties can usually be converted to aleatoric ones by better modelling and adding more knowledge about the system [34]. A real system comprises of both types of uncertainties [37] and modelling aleatoric uncertainties can serve as a first approximation to quantify the system behaviour.

In fact methods for epistemic uncertainties can equally benefit from improvements to aleatoric ones. For example, Yoo et al. [5] modelled the failure strength of material

as epistemic uncertainty and compared the results for the probability of failure using epistemic methods based on (i) probability (ii) interval analysis and (iii) evidence theory. In the probability method, the estimated failure strength was modelled as a distribution of distributions and the probability of failure was obtained using a double-loop Monte Carlo simulation (see fig. 1.1). In the outer loop the interval for the failure strength (epistemic) is modelled using an uniform distribution and in the inner loop the failure strength is modelled as a normal distribution (with estimated mean failure strength μ_{est} obtained from the outer loop). The inner loop represents aleatory uncertainty where aleatoric UQ methods are applicable. Therefore, the multilevel multifidelity ideas shown in the thesis for aleatoric type can be extended to methods for epistemic uncertainty quantification as well.

Finally, many existing tools in engineering practice are based on statistical and stochastic models. Therefore, engineers have greater insight and understanding of terms and models akin to their field for example mean and standard deviation.

1.2 Objectives of the Thesis

In this work, an adjoint based multifidelity multilevel Monte Carlo method for fast and accurate uncertainty quantification is proposed with emphasis on robustness of computational efficiency due to increase in number of uncertain parameters. Uncertainties are considered to be of the aleatoric kind. The multifidelity aspect of the method involves the development and use of low and high fidelity models and its analysis in the multifidelity framework.

The method is demonstrated on a model problem of quantifying uncertainties in turbomachinery aerodynamic performance due to manufacturing variations. A synthetic model is used to represent the surface variations due to manufacturing process. As a proof of concept the aerodynamic performance UQ of a 2-D turbine cascade whose surface is subject to the synthetic surface perturbation model is carried out using the

proposed UQ method.

1.3 Thesis Plan

A detailed survey of UQ methods is presented in ch. 2. The pros and cons of methods based on Monte Carlo and polynomial chaos expansion are discussed. Reasons for adopting the multilevel Monte Carlo methods are justified using some of its favourable properties.

The merits and demerits of low fidelity models for use in multifidelity control variate are discussed in ch. 3. A multifidelity control variate method based on Ghate's IMC [1] is proposed based on its favourable properties. The method is demonstrated on a viscous Burgers' problem with uncertain boundary condition. The multifidelity framework is analysed using the method of Ng and Wilcox [3, 22] closing with a discussion on the pros and cons. Then this multifidelity method is introduced into a multilevel Monte Carlo framework, which realises the proposed multifidelity multilevel uncertainty quantification framework called FastUQ. The implementation of FastUQ using the Dakota toolkit [38] is outlined.

In ch. 4, the implementation of the high fidelity model; a parallel flow solver based on the Reynolds Averaged Navier-Stokes (RANS) equation in rotational frame of reference, discrete adjoint, and the tangent-linear solver are presented. Validation results of the non-linear flow solver is presented and comparison between sensitivities obtained from discrete adjoint, tangent-linear and finite-difference is shown.

A brief introduction and survey of methods to model manufacturing variability in turbomachines is presented in ch. 5. A synthetic model for manufacturing variability is proposed based on the literature survey. A procedure for extracting uncertain surface modes from the model using goal-based principal component analysis (G-PCA) [39, 27] for multiple performance parameters is shown.

The proposed synthetic perturbation model and G-PCA technique along with the Fas-

tUQ method is used to quantify the uncertainty in two cost functions - total-pressure loss and exit mass flow rate, due to manufacturing variations on a VKI LS89 turbine cascade at two off-design conditions in ch. 5. In total 25 input uncertainties (surface perturbation modes) were considered in the UQ problem.

1.4 Contributions

The present work includes advances over the existing state of the art on several aspects, which are summarised below,

- The reduction factor ϕ to estimate the reduction of the Multifidelity Monte Carlo (MFMC) compared to the classical Monte Carlo was derived as an extension to the analysis of Ng et al. [3]. The contour plot of this reduction factor (a function of correlation and runtime ration) gives the feasible region of the MFMC.

- IMC as a low fidelity model in the MFMC framework of Ng and Wilcox [3] is proposed and demonstrated on the hypersensitive viscous Burgers' equation with uncertain boundary condition. Reduction estimate ϕ was verified with the results of the Burgers' problem. The multilevel multifidelity Monte Carlo (MLMF) framework proposed by Geraci et al. [4, 24] was derived using the cost model of Ng et al. [3]. A relation between the reduction factor Λ of the MLMF and ϕ of MFMC was derived, which proved the relation $\phi \geq \Lambda$, i.e., introducing the multilevel guarantees improvement of the MFMC.

- The non-linear flow solver shown in ch. 4 uses a two-halo layer partitioning in favour of the usual single halo layer. The trade-off between lower communication cost at the expense of additional redundant computations between the two partitioning is shown. In addition, the two-layer halo partitioning is shown to simplify the development of the discrete adjoint and tangent-linear solvers.

- Although the surface perturbation model introduced in ch. 5 and the goal-based principal component analysis to extract dominant surface uncertainty modes is

not new, application to problems involving multiple QoIs and the modifications necessary to the truncation procedure is shown with a practical example.

- In ch. 5, the FastUQ method is demonstrated using the 2-D VKI LS89 Turbine cascade subject to surface perturbations. The method is compared against the standard MLMC method of Giles [19] and IMC 1. The strengths and limitations of the method are discussed. The effects of PCA truncation and pilot sampling on the QoI statistics are shown.

The developments in this work have become part of the FORTRAN based flow solver STAMPS [40], which is an open-source in-house code of QMUL, maintained by the QMUL adjoint optimisation group of Dr. Jens-Dominik Müller.

Chapter 2

Methods for Quantifying Aleatoric Uncertainties

"Study the past if you would define the future."

— Confucius

2.1 Important Definitions and Concepts

In this section a brief review of widely used concepts and terms in probability and uncertainty quantification is presented followed by a brief introduction to adjoints and parameter sensitivity. Concepts such as joint probability, statistical independence, and adjoint sensitivity introduced in this chapter are also used extensively in ch. 3 and 5.

2.1.1 Probability Space

Probability space is usually denoted by $(\Omega, \mathcal{B}, P)$, where Ω is the sample space i.e. a set of all possible outcomes of a random event. For example, flipping a coin is an event that has a sample space $\Omega : \{head,\ tail\}$. $\emptyset \subset \Omega$ is a special set called the null or empty set. $\mathcal{B}$ is the event space, which is the set of all events that generates the outcomes. Finally P is called the probability measure (say probability of getting a tail).

2.1.2 Probability Distributions

Consider a continuous random function $X(\omega)$ in the probability space $(\Omega, \mathcal{B}, P)$, which maps each result $\omega \in \Omega$ to $x \in \mathbb{R}$, and for each event say $\mathcal{A}_i \in \mathcal{B} \subset \Omega$ in an interval $\mathcal{B}_i \subseteq \mathbb{R}$. $X(\omega)$ is called a *continuous random variable* (RV) and x is called its *realisation*. RV cannot be defined at specific values and takes an infinite number of possible values. Instead they are defined over an interval of values (say $[x_1, x_2]$) represented as integrals over a probability density function (*pdf*) denoted by $p(x)$.

This function satisfies the criteria (i) $p(x) \geq 0$ for all x and (ii) $\int_\Omega p(x)dx = 1$. The probability measure of finding $X(\omega)$ in $[x_1, x_2]$ is denoted by $P(x_1 \leq x \leq x_2)$ and it is evaluated using the integral in eq. (2.1). For example, the area under the curve shown in fig. 2.1(a) represents the probability in the interval $[x_1, x_2]$ for some distribution $p(x)$.

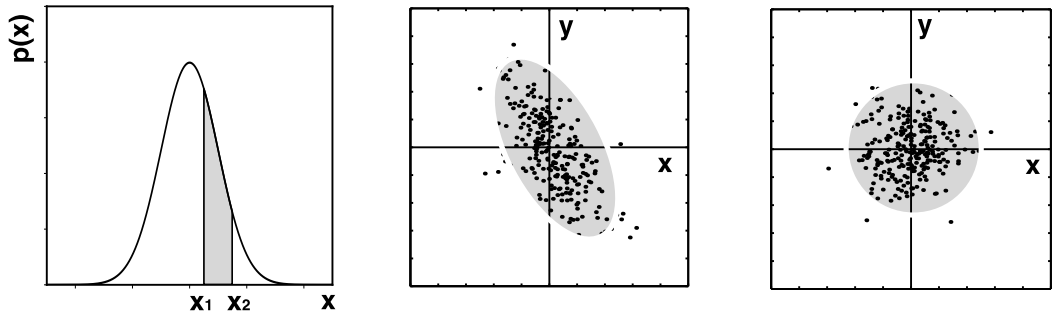
$$P(x_1 \leq x \leq x_2) = \int_{x_1}^{x_2} p(x)dx \tag{2.1}$$

The most widely used distribution in numerous applications is the *Gaussian* (or *normal*) *distribution* given in eq. (2.2) (see fig. 2.1(a)). The parameters μ and σ in eq. (2.2) are the mean and standard deviation of the distribution, which are defined in the next subsection 2.1.3.

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\left(\frac{x-\mu_x}{\sqrt{2}\sigma}\right)^2\right] \tag{2.2}$$

It is also possible to consider n-dimensional *random vectors* i.e., $\mathbf{X}(\boldsymbol{\omega})$ and its realisation (vector) $\mathbf{x} = [x_1 \;\; x_2 \;\; \ldots \;\; x_n]^T$, where x_1 to x_n are realisations of continuous scalar random variables. Here the probability measure for say $P(\mathbf{x} \leq \mathbf{x}_0)$ becomes an integral in $\mathbb{R}^n$ as

$$P(\mathbf{x} \leq \mathbf{x}_0) = \int_{-\infty}^{x_{0,1}} \int_{-\infty}^{x_{0,2}} \ldots \int_{-\infty}^{x_{0,n}} p(\mathbf{x})dx_1dx_2 \ldots dx_n = \int_{-\infty}^{\mathbf{x}_0} p(\mathbf{x})d\mathbf{x}. \tag{2.3}$$



(a) Gaussian Distribution (b) Joint Distribution (negative covariance) (c) Joint Distribution (zero covariance)

Figure 2.1: Gaussian and joint probability distribution example

2.1.3 Definition of Mean and Standard Deviation

The *mean* or the *expectation* of $\mathbf{x}$ is denoted by $\mathbb{E}[\mathbf{x}]$ and *variance* $\mathbb{V}ar[\mathbf{x}]$ (square of the *standard deviation* σ) of a random variable $\mathbf{x}$ are both defined using the integrals in eq. (2.4) and (2.5). Note that the mean is alternatively denoted using the symbols $\mu_{\mathbf{x}}$ in this thesis.

$$\mu_{\mathbf{x}} = \mathbb{E}[\mathbf{x}] = \int_{-\infty}^{+\infty} \mathbf{x} p(\mathbf{x}) d\mathbf{x} \tag{2.4}$$

$$\sigma^2 = \mathbb{V}ar[\mathbf{x}] = \int_{-\infty}^{+\infty} (\mathbf{x} - \mu_{\mathbf{x}})^2 f(\mathbf{x}) d\mathbf{x} \tag{2.5}$$

2.1.4 Covariance, Cross-covariance, and Joint Probability

The *covariance matrix* or simply *covariance* $\mathbf{C}_{\mathbf{x}}$ of a random variable $\mathbf{x}$ is defined as

$$\mathbf{C}_{\mathbf{x}} = \int_{-\infty}^{+\infty} (\mathbf{x} - \mu_{\mathbf{x}})(\mathbf{x} - \mu_{\mathbf{x}})^T p(\mathbf{x}) d\mathbf{x}. \tag{2.6}$$

Note that for a scalar random variable x, the covariance reduces to the variance i.e., $\mathbf{C}_x = \sigma^2$. The concept of covariance can be extended to more than one random variable, say random (vector) variables $\mathbf{X}$ and $\mathbf{Y}$, using the *cross-correlation* matrix $\mathbf{C}_{\mathbf{xy}}$ defined in eq. (2.7). This motivates the concept of a *joint probability distribution* $p(\mathbf{x}, \mathbf{y})$, which

is a distribution that yields the probability that each $\mathbf{x}$, $\mathbf{y}$ falls in any particular specified range. Examples of negative and zero covariance joint distributions are shown in fig. 2.1(b-c).

$$\mathbf{C_{xy}} = \int_{-\infty}^{+\infty} (\mathbf{x} - \mu_{\mathbf{x}})(\mathbf{y} - \mu_{\mathbf{y}})^T p(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y} \tag{2.7}$$

In the field of statistics, the *Pearson correlation* ρ is often used instead of the cross-covariance or covariance and it is useful to define it below:

$$\rho = \frac{\mathbf{C_{xy}}}{\sqrt{\mathbb{V}ar[\mathbf{x}]\mathbb{V}ar[\mathbf{y}]}} \tag{2.8}$$

The value of ρ is bound between $[0, 1]$, where $\rho = 0$ indicates no-correlation and $\rho = 1$ indicates maximum correlation.

2.1.5 Statistical Independence and Covariance

Two distributions $\mathbf{x}$ and $\mathbf{y}$ are *statistically independent* if their joint probability is the product of the individual probabilities (see eq. (2.9)), i.e., occurrence of an event in $\mathbf{x}$ does not affect the occurrence of $\mathbf{y}$. In addition, if $\mathbf{x}$ and $\mathbf{y}$ follow a joint Gaussian distribution then a *zero cross-covariance implies that they are statistically independent* [41] (see fig. 2.1(c) for an example of zero covariance). Note that any set of RVs having a Gaussian distribution and uncorrelated does not imply they are independent. The distributions have to be jointly Gaussian i.e., any non-trivial linear combination of them is a Gaussian RV.

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y}) \tag{2.9}$$

Two or more RV are independent identically distributed ($i.i.d$) if the RVs are statistically independent and belong to the same type of probability distribution function.

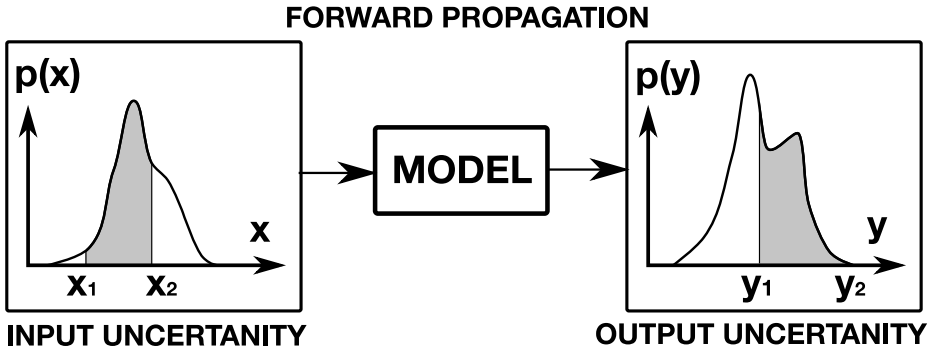


Figure 2.2: Forward propagation of uncertainties

2.1.6 Uncertainty Quantification and Model Response

Uncertainty quantification (UQ) can be broadly defined as the science of identifying, quantifying and reducing uncertainties associated with computational/analysis models, numerical algorithms, experiments and predicted outcomes of *quantities of interest* (QoI) [34]. QoI designate the output of a simulation or decision about the process. QoI is also referred by the names model-response or model-output. In this thesis QoI is mostly denoted by J.

Forward propagation of uncertainties deals with the propagation of the input (uncertainties) through the model to obtain the model-response. It is defined as the three step process [42] of input definition, propagation and output verification/analysis. In fig. 2.2 the input uncertainty $\mathbf{x}$ (with *pdf* $p(\mathbf{x})$), is propagated through the model to obtain the model response $\mathbf{y}$ having a *pdf* $p(\mathbf{y})$. Quantification of *risk* involves the estimation of the probability $P(\mathbf{y}_1 \leq \mathbf{y} \leq \mathbf{y}_2)$ of the model output by the forward propagation of the given input $\mathbf{x}$ [43] (possibly defined over a range $[\mathbf{x}_1, \mathbf{x}_2]$).

2.1.7 Adjoint and Parameter Sensitivity

Consider a QoI J, which depends on a state u and design parameters α. Let the state satisfy the solution to the state equation shown below:

$$R(u, \alpha) = 0 \tag{2.10}$$

This equation is a linear or non-linear model depending on the physical problem it describes. The parameter α is used to control the state of the system R to achieve a desired J. Let the dimension of the QoI vector be L, the state vector be N, and the design vector be M. Typically, the dimension of QoI is much smaller than that of the design variable ($M \gg L$). In practical engineering problems QoI are mostly scalar integral values [2] ($L = 1$), for example: efficiency, total pressure loss, cost of manufacturing, etc.

Let the QoI be optimised (minimised) subject to the state equation (shown in eq. (2.10)) and other additional constraints in the design space. This minimisation problem is defined as

$$\min_{\alpha} J(u, \alpha), \text{ s.t. } R(u) = 0. \tag{2.11}$$

The *sensitivity* ($\frac{dJ}{d\alpha}$) is an important quantity in design optimisation. Given an initial design α, the sensitivity can be used to find the search direction to minimise J. The computational difficulty involved in sensitivity computation is the construction of large matrices arising from the chain-rule (of calculus) as shown in eq. (2.12). Strang [16] describes the adjoint method as "a fast way to compute the sensitivity of J with respect to α when $M \gg L$".

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial \alpha} + \frac{\partial J}{\partial u}\frac{\partial u}{\partial \alpha} \tag{2.12}$$

The method is best described using the example of a linear state equation and linear QoI as shown in eq. (2.13) and eq. (2.14), where c is a constant vector of the same dimension as u and $b = b(\alpha)$ is the forcing term. Applying the chain-rule to eq. (2.12) and after simplification the sensitivity reduces to eq. (2.15). Computationally, the most expensive operation is the $N \times M$ matrix-multiplication of the term $A^{-1}\frac{\partial b}{\partial \alpha}$. The adjoint method pre-multiplies $c^T A^{-1}$ to avoid the expensive $N \times M$ matrix-multiplication operation. *The pre-multiplication is achieved by solving the adjoint equation shown in eq.* (2.16).

The adjoint variable v contains the product of pre-multiplication. Thus the sensitivity can be efficiently obtained using the adjoint value as shown in eq. (2.17). One can observe that the adjoint variable does not involve the forcing term b. Therefore, the adjoint

variable once calculated can be used to get the sensitivity of any arbitrary forcing b i.e., *adjoint solution gives the influence of an arbitrary source term on the QoI.*

$$J(u, \alpha) \equiv c^T u \tag{2.13}$$

$$R(u, \alpha) \equiv Au - b = 0 \tag{2.14}$$

$$\frac{dJ}{d\alpha} = c^T A^{-1} \frac{\partial b}{\partial \alpha} \tag{2.15}$$

$$A^T v = c \tag{2.16}$$

$$\frac{dJ}{d\alpha} = v^T \frac{\partial b}{\partial \alpha} \tag{2.17}$$

The method can be extended to non-linear state equation $R(u, \alpha) = 0$, for instance, the Reynolds Averaged Navier-Stokes Equation (RANS) shown in ch. 4. Using the linearised state eq. (2.18) the sensitivity for the non-linear case can be assembled as shown in eq. (2.19).

$$\frac{\partial u}{\partial \alpha} = -\left(\frac{\partial R}{\partial u}\right)^{-1} \frac{\partial R}{\partial \alpha} \tag{2.18}$$

$$\frac{dJ}{d\alpha} = \frac{\partial J}{\partial \alpha} - \frac{\partial J}{\partial u}\left(\frac{\partial R}{\partial u}\right)^{-1} \frac{\partial R}{\partial \alpha} \tag{2.19}$$

The adjoint method involving the non-linear state can be constructed using the same idea of pre-multiplication. Here the terms $\frac{\partial J}{\partial u}$ and $\left(\frac{\partial R}{\partial u}\right)^{-1}$ are pre-multiplied using the adjoint equation shown in eq. (2.20).

$$\left(\frac{\partial R}{\partial u}\right)^T v = \left(\frac{\partial J}{\partial u}\right)^T \tag{2.20}$$

In this thesis the discrete adjoint approach is used where the terms in the adjoint eq. (2.20) are obtained using *algorithmic differentiation* (AD). An introduction to AD and discussion on AD applied to parallel codes involving message passing communication calls is provided in Appendix C. The parallel implementation of the discrete adjoint solver for steady RANS equations is provided in ch. 4.

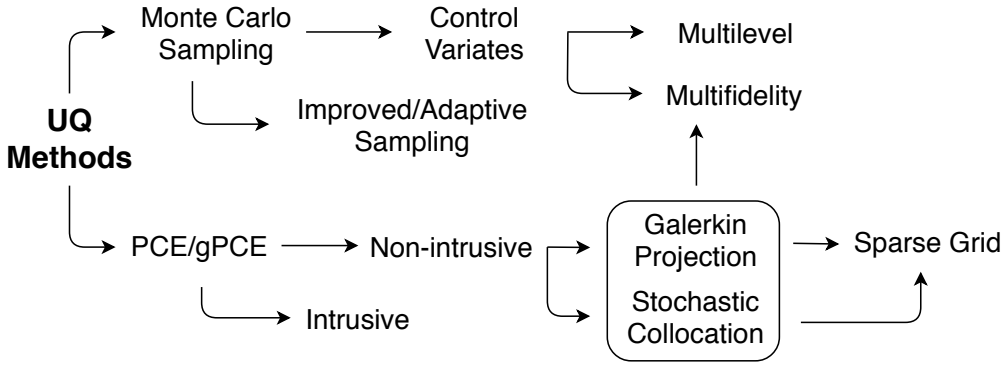


Figure 2.3: Overview of methods in UQ survey

2.2 A Survey of Uncertainty Quantification Methods

In this section a survey of existing UQ methods is shown. The strengths and weakness of each method is highlighted and finally justify the choice of multifidelity multilevel Monte Carlo method based on the survey. The methods are broadly classified into the Monte Carlo sampling and Polynomial Chaos Expansion. Improvements and variants within each method is then surveyed as shown in fig. 2.3.

2.3 The Monte Carlo Family of Sampling Methods

The Monte Carlo (MC) method is a popular method in UQ due to its simplicity and flexibility since the analysis model is treated like a black-box. These methods select input sample points and run the forward model to estimate the statistics of the QoI. In addition, MC has a provable convergence behaviour, which is *independent of the number of input (uncertain) parameters.* The methods differ mostly in the way the input samples or population are chosen.

2.3.1 Randomly Sampled Monte Carlo

The randomly sampled MC is the most general implementation, whose pseudo code is shown in *alg.* (2.1) on the following page. The algorithm finds an estimate for the mean and variance of QoI J with input uncertainty α. $J_p : \{p = 1, \ldots, P\}$ are the outputs of the P input samples.

input : Maximum mean squared error (ϵ) and maximum number of sampled P
output: Estimated output statistics ($\mathbb{E}[J], \mathbb{V}ar[J]$) and samples used P

1 *Set random seed*;
2 **for** $p \leftarrow 1$ *to* P **do**
3 *Generate random input sample* α_p;
4 *Evaluate model output* $J(\alpha_p)$;
5 $\mathbb{E}[J] \approx J_p^{MC} \leftarrow \frac{1}{p}\sum_{i=1}^{p} J_i$;
6 $\mathbb{V}ar[J] \approx \mathbb{V}ar[J]^{MC} \leftarrow \frac{1}{p^2}\sum_{i=1}^{p}(J_i - J_p^{MC})^2$;
7 **if** $\text{MSE}[J_p^{MC}] < \epsilon$ **then**
8 $P \leftarrow p$;
9 **exit;**
10 **end**
11 **end**

Algorithm 2.1: Monte Carlo Method Algorithm

In step 3 of the MC algorithm pseudo-random numbers are generated following the input *pdf*. Step 4 is computationally expensive since it involves model evaluation for every generated input sample. MC is embarrassingly parallel since each sample can be evaluated independently. The *mean squared error* (MSE) is an important quantity in MC. It gives an estimate of how far the computed statistics are from the true one. In fact, MSE was used as a termination criteria for the MC in step 7 of *alg.* (2.1). Note that the estimator J_p^{MC} is unbiased i.e., $\mathbb{E}[J] \equiv \mathbb{E}[J_p^{MC}]$ and if one assumes that the samples J_p are *i.i.d* then the MSE for the MC reduces to the simple expression:

$$\text{MSE}[J_p^{MC}] = \mathbb{E}\left[\left(J_p^{MC} - \mathbb{E}[J]\right)^2\right] = \underbrace{P^{-1}\mathbb{V}ar[J_p]}_{\text{Statistical error}} + \underbrace{(\mathbb{E}[J_p] - \mathbb{E}[J])}_{\text{Bias error}} \tag{2.21}$$

The second term in eq. (2.21) reflects the bias of the estimator. For large P, the estimator becomes unbiased ($\mathbb{E}[J_p] \to \mathbb{E}[J]$) [44] and this term can be neglected, which leads to the expression (for MSE) below:

$$\text{MSE}[J_p^{MC}] = P^{-1}\mathbb{V}ar[J_p] \tag{2.22}$$

For large number of samples P the MC estimator is unbiased and converges to the

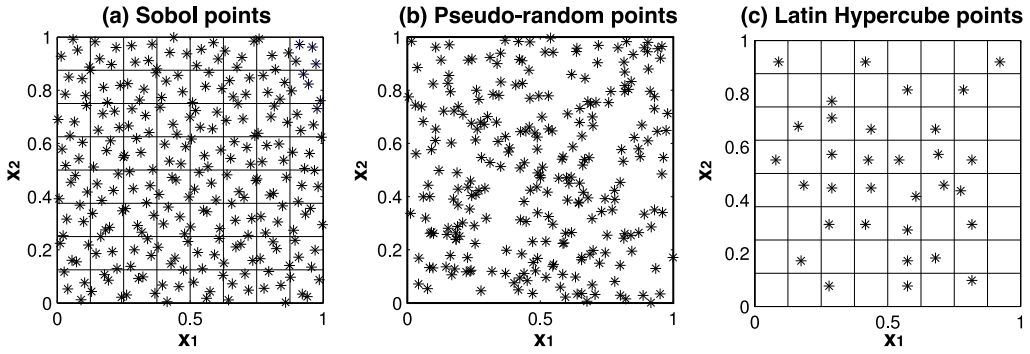


Figure 2.4: Sobol sequence, pseudo-random, and latin hypercube points

true statistics as $\mathcal{O}(P^{-\frac{1}{2}})$. *This slow rate of convergence can be improved by improved sampling and variance reduction techniques*, which are discussed next.

2.3.2 Improved and Adaptively Sampled Monte Carlo

Latin hypercube sampling (LHS) is a widely used sampling procedure for MC. Here the range of each input parameter is divided into equal probability segments. Samples are then picked randomly from the row and column intervals of this orthogonal array [45, 46] (see fig. 2.4(c)). *If the output pdf is square integrable then the error falls like* $\mathcal{O}(P^{-\frac{3}{2}})$ *with number of samples* P [44]. Quasi-Monte Carlo (QMC) methods [47] use points generated using the quasi-random Sobol sequences [48] (see fig. 2.4(a) for an example). *For input dimensions less than forty* ($M < 40$) *the Sobol sequence for the best case scenario gives* $\mathcal{O}\left(P^{-1}\right)$ *error convergence of QMC (randomised)* [49]. Bratley [50] provides FORTRAN subroutines to generate Sobol points for dimension $d < 40$. Problems where the QoI is a probability measure of the output, say rare events [51, 52, 53] near the tail of the *pdf*, even LHS or QMC might not have sufficient samples in such regions to obtain the probability. For such cases the importance sampling is used to reduce the total number of samples. Importance sampling [43, 15, 44, 54] relies on the importance function, which are distributions used in lieu of the output distribution and chosen a priori. Corrections are then applied using a posterior error estimator [44]. *The number of samples to estimate the output probability can be reduced significantly if the prior distribution are chosen optimally as shown in [55].*

2.3.3 Control Variate Monte Carlo

The control variate approach is based on the idea of replacing the QoI J with an approximation G, which is cheap to evaluate. In addition, G should have a similar expected value as the original objective, but, with a smaller variance. The goal here is to reduce the second term of eq. 2.22, which is the estimator variance, without using many samples. The control variate estimator J_p^{CV} including G can be defined as,

$$J_p^{CV} = J_p^{MC} - \beta \left(G_p^{MC} - \mathbb{E}[G] \right). \tag{2.23}$$

The estimator J_p^{CV} is unbiased irrespective of the value of the control variate parameter $\beta \in \mathbb{R}$, which is obtained from the variance estimate of G and the cross-covariance (see eq. (2.7)) between J and G is defined as,

$$\beta = \frac{\mathbf{C}_{JG}}{\mathbb{V}ar[G]}. \tag{2.24}$$

The variance of the control variate estimator is related to the MC estimator [4] via the Pearson correlation coefficient ρ (see eq. (2.8)) as,

$$\mathbb{V}ar[J_p^{CV}] = (1 - \rho^2)\mathbb{V}ar[J_p^{MC}] \tag{2.25}$$

$$\rho = \frac{\mathbf{C}_{JG}}{\sqrt{\mathbb{V}ar[J]\mathbb{V}ar[G]}}. \tag{2.26}$$

Success of control variate approach depends on how good a given low fidelity model (LF) correlates with a high fidelity model (HF), i.e. ρ should be close or equal to one.

2.3.4 Multifidelity Control Variate of Ng and Wilcox

Ng and Wilcox [3] proposed a practical multifidelity control variate implementation. In the context of robust aerospace design, the authors extended the same to an information reuse estimator [22] that reuses information from a previous design step.

In this approach, the number of high fidelity evaluations $J_p : \{p = 1, \ldots, P\}$ is typically

less than the number of low fidelity model evaluations $G_q : \{q = 1, \ldots, M\}$ and the mean of the approximate model $\mathbb{E}[G]$ in eq. (2.23) is replaced with the sample estimate $\mathbb{E}[G_q]$ to yield the realtion

$$J_p^{CV} = J_p^{MC} - \beta \left(G_q^{MC} - \mathbb{E}[G_q] \right). \tag{2.27}$$

Given a computational budget and the required MSE tolerance, the method optimises the allocation of resources between the high and low fidelity models. To benchmark the method with MC consider the computational effort c, which is a measure of the equivalent number of high-fidelity evaluations defined below:

$$c = P + \frac{M}{w} = P\left(1 + \frac{r}{w}\right) \tag{2.28}$$

In eq. (2.28), w is the runtime ratio between the average computational cost per high and low fidelity model evaluation and $r = \frac{M}{P} > 1$ is the ratio between the number of high and low fidelity simulations. For a fixed budget c one is free to choose the parameter r to vary the resource between high and low fidelity evaluations. The variance of the multifidelity estimator in terms of c, r, and w (obtained from ref [3]) is shown below:

$$\begin{aligned} \mathbb{V}ar[J_p]^{CV} &= \frac{1}{c}\left(1 + \frac{r}{w}\right)\left[\mathbb{V}ar[J] + \left(1 - \frac{1}{r}\right)\left(\beta^2 \mathbb{V}ar[G] - 2\beta\rho\sqrt{\mathbb{V}ar[G]\mathbb{V}ar[J]}\right)\right] \\ &= \left[1 + \left(1 - \frac{1}{r}\right)\left(\beta^2 \frac{\mathbb{V}ar[G]}{\mathbb{V}ar[J]} - 2\beta\rho\sqrt{\frac{\mathbb{V}ar[G]}{\mathbb{V}ar[J]}}\right)\right]\frac{\mathbb{V}ar[J]}{P} \end{aligned} \tag{2.29}$$

For optimal resource allocation, one minimises the variance in eq. 2.29 for both β and r to yield the optimality condition:

$$\mathrm{MSE}[J_p]^{CV} = \mathbb{V}ar[J_p]_{opt}^{CV} = \left(1 + \frac{r_{opt}}{w}\right)\left(1 - \frac{r_{opt} - 1}{r_{opt}}\rho^2\right)\frac{\mathbb{V}ar[J]}{c}. \tag{2.30}$$

Where, r_{opt} and β_{opt} are defined as

$$r_{opt} = \sqrt{\frac{w\rho^2}{1 - \rho^2}}, \quad \text{and} \quad \beta_{opt} = \rho\sqrt{\frac{\mathbb{V}ar[J]}{\mathbb{V}ar[G]}}. \tag{2.31}$$

When the cost of computing the low fidelity samples is almost free i.e., $w \to \infty$, then $r_{opt} \to \infty$ and one recovers the classical control variate estimator in eq. (2.25). When one has a perfect low fidelity model i.e., $\rho \to 1$ then $\text{MSE}[J_p]^{CV} = \frac{1}{w}\frac{\mathbb{V}ar[J]}{c}$. Therefore, a low fidelity model should not only correlate well but also should be cheaper to evaluate. If the condition $\rho > \frac{1}{1+w}$ is not satisfied then it is worthwhile to switch to the classical MC.

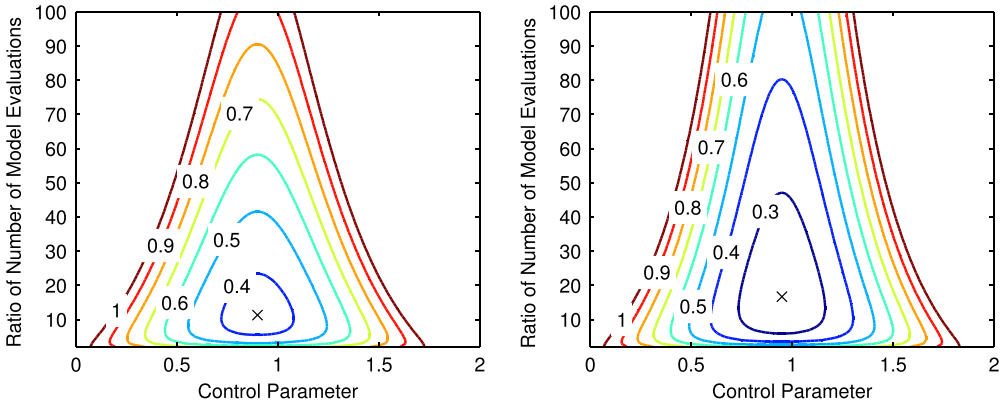
2.3.4.1 Effect of Under Sampling

Practically, it is not possible to compute $\mathbb{E}[J_p]^{CV}$ and $\mathbb{V}ar[J_p]^{CV}_{opt}$ using eq. 2.29 and eq. 2.30, because $\mathbb{E}[J]$, $\mathbb{V}ar[J]$, $\mathbb{V}ar[G]$, ρ and β are unknown quantities and the usual practice is to replace them with sample estimates. For a budget c, one has N samples of J_p, which can be used to estimate the necessary statistics. Replacing the exact values with approximate ones introduces errors. Therefore, it is necessary to check if the estimator still yields a reduction in variance. This is checked by plotting the ratio ψ shown in eq. 2.32 (from ref. [3]). In fig. 2.5 the sample contours for ψ is shown for different values of r and β for two values of correlation ρ (a) 0.9 and (b) 0.95. For small errors in the sample estimate of β and r (exact values indicated by $\times$) are acceptable but optimal variance reduction cannot be achieved [3].

$$\psi = \frac{\mathbb{V}ar[J_p]^{CV}}{\mathbb{V}ar[J_p]} = \left(1 + \frac{r}{w}\right)\left[1 + \left(1 - \frac{1}{r}\right)\left(\beta^2\frac{\mathbb{V}ar[G]}{\mathbb{V}ar[J]} - 2\beta\rho\sqrt{\frac{\mathbb{V}ar[G]}{\mathbb{V}ar[J]}}\right)\right] \tag{2.32}$$

2.3.4.2 Convergence Comparison with MC

In the original reference of Ng [3] a comparison with a classical MC estimator was not shown. In this section a comparison with the classic MC estimator is thus derived using the ratio of the respective MSE estimates in eq. (2.22) and (2.30) i.e., the ratio between the MSE of the optimal multifidelity and classical MC estimator. The final expression



(a) Correlation coefficient $\rho = 0.9$ (b) Correlation coefficient $\rho = 0.95$

Figure 2.5: Contours of ψ for given ratio of number of model evaluations r and control parameter β for (a) $\rho = 0.9$ and (b) $\rho = 0.95$ from ref. [3]

after replacing r_{opt} with $\sqrt{\frac{w\rho^2}{1-\rho^2}}$ in eq. (2.30) to yield:

$$\begin{aligned}\phi = \frac{\text{MSE}[J_p]^{CV}}{\text{MSE}[J_p]^{MC}} &= \left(1 + \frac{r_{opt}}{w}\right)\left(1 - \frac{r_{opt} - 1}{r_{opt}}\rho^2\right) \\ &= \left(1 + \sqrt{\frac{\rho^2}{w(1-\rho^2)}}\right)\left[1 - \left(1 - \sqrt{\frac{1-\rho^2}{w\rho^2}}\right)\rho^2\right]. \end{aligned} \tag{2.33}$$

The above equation neglects the bias error in MC and considers that the number of samples used in the MC estimator is the equivalent number of high fidelity samples of the multifidelity one (c). Note that there are no terms involving P in eq. (2.33). *Clearly the multifidelity MC has the same* $\mathcal{O}\left(P^{-\frac{1}{2}}\right)$ *convergence but the entire convergence curve is shifted (reduced) by the factor shown in eq.* (2.33)*. The shift is only a function of the correlation* ρ *and runtime ratio* w*.* To increase the rate of convergence one can combine the multifidelity method with a better sampling technique such as LHS.

2.3.4.3 Multifidelity Implementation

The algorithmic steps involved in the multifidelity approach is shown in *alg.* (2.2) on the following page. The algorithm starts with an initial pilot sample of P_0 HF and LF evaluations. An important aspect in step 6 of the algorithm is that the $M - P$ samples are run only for the low fidelity model (in addition to the P initial samples).

The pilot sample size is increased if MSE does not match the tolerance (see step 9). Peherstorfer[56] derived convergence estimates on multifidelity methods using multiple

input : Target error (ϵ), HF model (J), LF model (G), Pilot sample size (P_0)
output: Output statistics ($\mathbb{E}[J], \mathbb{V}ar[J]$) and equivalent HF samples (c)

1 $P \leftarrow P_0$;
2 **seed** $random$;
3 **draw** $samples(P, random)$;
4 **estimate** ρ, r, β, and $\mathbb{V}ar[J_p]$;
5 $M \leftarrow \lceil Pr \rceil$;
6 **draw** $samples(M - P, random)$;
7 **Estimate** $\text{MSE}[J_p]^{CV}$, $\mathbb{E}[J_p]^{CV}$ and $\mathbb{V}ar[J_p]^{CV}$;
8 **if** $\text{MSE}[J_p^{MC}] > \epsilon$ **then**
9 $\quad P \leftarrow P + P_0$;
10 $\quad$ **goto** 3;
11 **end**

Algorithm 2.2: Control variate Monte Carlo algorithm

low fidelity models. The coarse level solutions were considered as multiple low fidelity models. The multifidelity method was compared to the multilevel Monte Carlo method. The authors found that both approaches had similar runtimes and convergence. The multilevel Monte Carlo method is described in the next section.

2.3.5 Multilevel Control Variate Monte Carlo

The control variate approach can be recursively applied to solutions from coarse discretisation leading to the multilevel Monte Carlo (MLMC) method. MLMC methods were introduced by Heinrich for numerical quadrature [57, 58, 59]. Kebaier [60] proposed a two-level Monte Carlo method in which a coarse grid numerical approximation of a stochastic differential equation was used as a control variate to a fine grid numerical approximation. This reduced the number of samples needed on the fine grid and decreased the total computational burden. Giles later developed the method to enhance the efficiency of path simulations for Itô stochastic ordinary differential equations in refs. [61, 19]. The MLMC method due to Giles is usually called the Standard MLMC or SMLMC [62]. The method was extended to Finite Element Methods (FEM) for elliptic problems with stochastic coefficients by Barth in ref. [63]. Mishra proposed a

novel MLMC Finite Volume Method (MLMC-FVM) for scalar conservation laws [20] and non-linear systems of balance laws [17].

The numerical solution to PDE involves the tessellation of the domain into N degrees-of-freedom and the QoI J_N is obtained using this tessellation. Therefore, as $N \to \infty$, the truncation error vanishes and one recovers the exact solution $J_N \to J$ and expected values $\mathbb{E}[J_N] \to \mathbb{E}[J]$. Similar to multi-grid one can consider a sequence of discretisation levels $\{N_l : l = 0, \ldots, L\}$ with $N_0 < N_1 < \cdots < N_L \equiv N$. Using the linearity property of expectation, the expectation at the finest level can be written using the telescopic sum of the sequence of levels as shown in eq. (2.34). The difference function Y_l defined in eq. (2.35) is introduced to simplify the analysis.

$$\mathbb{E}[J_N] = \mathbb{E}[J_{N_0}] + \sum_{l=0}^{L} \mathbb{E}[J_{N_l} - J_{N_{l-1}}] = \sum_{l=0}^{L} \mathbb{E}[Y_l] \tag{2.34}$$

$$Y_l = \begin{cases} J_{N_0} & \text{if } l = 0 \\ J_{N_l} - J_{N_{l-1}} & \text{if } l \neq 0 \end{cases} \tag{2.35}$$

2.3.5.1 Optimal Sample Allocation Between Levels

Similar to the expectation telescopic sum in eq. (2.34), the MLMC estimator can be written as shown in eq. (2.36), where P_l is the number of MC samples for the discretisation level l and $Y_l^{(i)}$ is the difference function for the i^{th} sample point of level l. This estimator is unbiased, which reduces the MSE of the estimator to a simple expression shown in eq. (2.37). The overall computational cost C_N^{ML} can be estimated from the computational cost per-level C_l using eq. (2.38).

$$J_{N,\,p}^{ML} = \sum_{l=0}^{L} \frac{1}{P_l} \sum_{i=1}^{P_l} \mathbb{E}[Y_l^{(i)}] \tag{2.36}$$

$$\mathbb{E}\left[\left(J_{N,\,p}^{ML} - \mathbb{E}[J]\right)^2\right] = \sum_{l=0}^{L} \frac{\mathbb{V}ar[Y_l]}{P_l} \tag{2.37}$$

$$C_N^{ML} = \sum_{i=0}^{L} P_l C_l \tag{2.38}$$

From eq. (2.37) it is clear that the number of samples P_l required to resolve the variance at level l decreases with l. Giles [19] used this idea to derive a strategy to optimally allocate resources. The optimisation problem for resource allocation is defined in eq. (2.39) using the overall cost model in eq. (2.38) and MSE of MLMC in eq. (2.37), where λ is the Lagrangian multiplier and ϵ^2 gives an estimate of the upper bound of the MSE [19].

$$f(P_l, \lambda) = \sum_{l=0}^{L} P_l C_l + \lambda \left(\sum_{l=0}^{L} \frac{\mathbb{V}ar[Y_l]}{P_l} - \frac{\epsilon^2}{2} \right) \tag{2.39}$$

The Lagrangian f in eq. (2.39) balances the cost and evaluations per level to achieve a target MSE of the estimator. Minimisation of the Lagrangian f gives the optimal number of samples at each level as shown in eq. (2.40).

$$P_l^{opt} = \frac{2}{\epsilon^2} \left[\sum_{k=0}^{L} \sqrt{\mathbb{V}ar(Y_k) C_k} \right] \sqrt{\frac{\mathbb{V}ar(Y_l)}{C_l}} \tag{2.40}$$

2.3.5.2 Optimal Sample Allocation Considering Weighted Error Splitting

Note that in the derivation of eq. (2.40), the bias and statistical error terms (first and second terms in (2.21)) are given equal weighting. Pisaroni [64] derived the optimal allocation using a splitting parameter $\theta \in (0, 1)$ to offer the possibility of weighting the two MSE contributions differently as shown below,

$$\mathbb{E}\left[\left(J_{N,\,p}^{MC} - \mathbb{E}[J] \right)^2 \right] = \theta \underbrace{\left(P^{-1} \mathbb{V}ar[J_N] \right)}_{\text{Statistical error}} + (1-\theta) \underbrace{\left(\mathbb{E}\left[(J_N - J)^2 \right] \right)}_{\text{Bias error}} \tag{2.41}$$

to yield the following optimal allocation,

$$P_l^{opt} = \frac{1}{\theta \epsilon^2} \left[\sum_{k=0}^{L} \sqrt{\mathbb{V}ar(Y_k) C_k} \right] \sqrt{\frac{\mathbb{V}ar(Y_l)}{C_l}}. \tag{2.42}$$

Note that eq. (2.40) can be recovered from eq. (2.42) by setting $\theta = \frac{1}{2}$ (equal weighting).

2.3.5.3 Optimal Sample Allocation for Nested Mesh Levels

Mishra [17] considered a nested mesh tessellation with spatial width at level l satisfying the condition Δx_l is $\mathcal{O}\left(2^{-l}\Delta x_0\right)$ and obtains the following simple estimate for optimal number of samples per level for MLMC-FVM (second order spatial discretisation),

$$P_l^{opt} = P_L 4^{(L-l)} \tag{2.43}$$

For practical complex industrial geometries, it is not possible to obtain a nested tessellation of coarse levels satisfying the condition Δx_l is $\mathcal{O}\left(2^{-l}\Delta x_0\right)$ for spatial width. Hence, SMLMC is preferred in this work over the MLMC-FVM method.

The analysis in refs. [63, 20, 17] shows that the *MLMC can produce converged numerical approximations to statistics of uncertain solutions of partial differential equations in computational complexity comparable to that of one numerical solve of a single path*, i.e. a single realisation of the random input data. The MLMC algorithm is heuristic and is not guaranteed to achieve an MSE error that is $\mathcal{O}(\epsilon^2)$, where ϵ is the tolerance. The weakness in the heuristic algorithm lies in the bias estimation [19] and *accuracy of the variance estimate at each level depends on the pilot sample size* (or the initial sample set).

Collier et al. [62] introduce the Continuation MLMC (CMLMC), which uses a sequence of decreasing tolerances on the convergence of the analysis model in addition to the coarse mesh sequence in the control variate. CMLMC solves the problem of pilot sampling or screening phase of the SMLMC with an on-the-fly parameter set estimation. CMLMC was used to propagate operational and geometric uncertainties in internal and external aerodynamics by Pisaroni [64, 21].

2.3.6 Multilevel Multifidelity Monte Carlo

Geraci et al. [4] recently proposed a multifidelity control variate within each level of the MLMC and presented the multilevel multifidelity MC (MLMF). Two formulations

of MLMF were shown in [4] namely, Y and Q correlations. Both methods were shown to be equivalent and the Q form was slightly better than Y but adds more complexity. The method was used to solve a model problem of temperature evolution in a rod with uncertain initial condition and thermal diffusivity. *One order reduction in number of samples (required to attain the same level of MSE) compared to MLMC was reported using MLMF.*

2.3.7 Summary of Monte Carlo Methods

The MSE obtained using the various methods from the Monte Carlo family described in the previous sections can be summarised using the simple relation shown below:

$$\text{MSE} = constant \times P^{-(1+rate)} \tag{2.44}$$

Variance reduction methods such as control variates, MLMC, and MLMF target the reduction of the *constant* in eq. (2.44), while keeping $rate = 0$. Better sampling techniques such as LHS, QMC, etc. target the reduction of the *rate* (i.e. $rate > 0$). In fact methods that target the reduction of *rate* have a feature in common, namely the increased non-randomness or regularity of the sample distribution (see fig. 2.4(a, c)). For a pseudo-random MC sampling the MSE convergence is independent of the number of input uncertainties d. But when a better regularity of the sampling is enforced to improve the *rate* (for example QMC), the aforesaid property is violated and MSE convergence becomes a function of both P and d. The polynomial based methods described in the next section target the regularity of the sampling points to improve the convergence of MSE. The various improvements to the polynomial methods strive to balance the dependence on d and improvement of the *rate*.

2.4 Generalised Polynomial Chaos Expansion

Polynomial Chaos Expansion (PCE) was first introduced by Wiener [65] in the context of decomposition of Gaussian processes (refer to sec. 5.3 for an introduction to stochastic

Probability distribution	**gPC basis**	**Weighting function $w(\boldsymbol{\xi})$**	**Support**
Gaussian	Hermite	$\frac{1}{\sqrt{2\pi}} \exp\left(\frac{-\boldsymbol{\xi}^2}{2}\right)$	$[-\infty, \infty]$
Gamma	Laguerre	$\frac{\boldsymbol{\xi}^{\alpha} \exp(-\boldsymbol{\xi})}{\Gamma(\alpha+1)}$	$[0, \infty]$
Beta	Jacobi	$\frac{\Gamma(\alpha+\beta+2)}{2^{(\alpha+\beta+1)}\Gamma(\alpha+1)\Gamma(\beta+1)}(1-\boldsymbol{\xi})^{\alpha}(1+\boldsymbol{\xi})^{\beta}$	$[-1, 1]$
Uniform	Legendre	$\frac{1}{2}$	$[-1, 1]$

Table 2-A: gPCE orthogonal polynomial type and corresponding continuous random variable (from reference [69])

process). Xiu and Karniadakis [66] proposed the generalised Polynomial Chaos Expansion (gPCE) to handle non-Gaussian processes. The method handles different types of polynomial bases from the Askey scheme [67, 68] based on the type of input probability distribution (shown in tbl. 2-A). A review of orthogonal polynomials and the Askey scheme can be found in the work of Xiu [69]. In gPCE one expands the output y of the forward uncertainty propagation as a weighted sum of orthogonal polynomials Ψ (stochastic) and a deterministic function a_i as shown below:

$$J(\boldsymbol{\alpha}, \boldsymbol{\omega}) = \sum_{i=1}^{N_p} \underbrace{a_i(\boldsymbol{\alpha})}_{deterministic} \underbrace{\Psi_i(\boldsymbol{\xi}(\boldsymbol{\omega}))}_{stochastic} \tag{2.45}$$

For d-dimensional input uncertainty, Ψ becomes a multi-dimensional function of the uncertain vector $\boldsymbol{\xi}(\boldsymbol{\omega}) = [\xi_1,\ \xi_2,\ \ldots,\ \xi_d]^T$. The expansion is truncated to N_p terms determined by the number of uncertain variables d and order m of the polynomials Ψ using the total-order expansion formula in eq. (2.46). The computational cost of the total order expansion scales with the input dimension d as $\mathcal{O}(N_p^d)$.

$$N_p = 1 + \sum_{m}^{s=1} \frac{1}{s!} \prod_{r=0}^{s-1} (d+r) = \frac{m+d!}{m!\ d!} \tag{2.46}$$

The mean and variance of the output y are obtained using the integrals shown in eq. (2.47)-(2.48), where the deterministic function a_i can be obtained using orthogo-

nality condition and inner product $\langle . \rangle$ with Ψ_i as shown in eq. (2.49). The weighting function $w(\boldsymbol{\xi})$ for popular distributions are shown in tbl. 2-A.

$$\mathbb{E}[J] = \int_a^b J(\alpha, \boldsymbol{\omega}(\boldsymbol{\xi}))w(\boldsymbol{\xi})d\boldsymbol{\xi} = a_0(\alpha) \tag{2.47}$$

$$\mathbb{V}ar[y] = \int_a^b \left[J(\alpha, \boldsymbol{\omega}(\boldsymbol{\xi})) - a_0(\alpha)\right]^2 w(\boldsymbol{\xi})d\boldsymbol{\xi} = \sum_{i=1}^{N_p-1} a_i^2(\alpha) \left\langle \Psi_i, \Psi_i \right\rangle \tag{2.48}$$

$$a_i(\alpha) = \frac{\langle J, \Psi_i \rangle}{\langle \Psi_i, \Psi_i \rangle} = \frac{1}{\langle \Psi_i, \Psi_i \rangle} \int_a^b J(\alpha, \boldsymbol{\omega}(\boldsymbol{\xi}))\Psi_i(\boldsymbol{\xi})w(\boldsymbol{\xi})d\boldsymbol{\xi} \tag{2.49}$$

The total-order expansion shown above requires solution of N_p equations for N_p unknows (N_p evaluations of the model) and has an error convergence of $\mathcal{O}(N_p^{-2})$. Alternatively the expansion can be applied to individual dimensions of polynomial order m_i leading to the *tensor product expansion* and the number of terms N_p takes the form in eq. (2.50). *Montomoli [42] states that polynomial approaches are viable when the number of uncertain variables are small, typically around five.*

$$N_p = \prod_{i=0}^{d-1} (m_i + 1) \tag{2.50}$$

2.4.1 Intrusive vs. Non-intrusive Methods

The expansion shown in eq. (2.45) can be introduced directly into the governing equations of the model for every uncertain parameter and rederive the governing equation as explained in ref. [70]. This *intrusive approach* is tedious and requires implementation of the new set of governing equations in computer code. *But the method is very accurate and has spectral convergence [42].*

Non-intrusive methods approximate the integral in eq. (2.49) numerically using regression or by interpolation to obtain a_i (and output statistics of y). This approach does not require changing the governing equation and uses the model as a black-box. *Therefore, non-intrusive methods are more practical.* Onorato et al. [71] compared intrusive

and non-intrusive implementations for fluid flows. *The accuracy obtained between both methods were similar except at regions of high non-linearity* such as shocks. QoI such as mean lift and drag had small over and under-prediction as a result.

2.4.2 Stochastic Collocation vs. gPCE Regression

The total-order expansion (see eq. (2.46)) or the tensor product (see eq. (2.50)) requires solving N_p unknown from N_p equations. For robustness, twice the number of samples ($n_s = 2N_p$) are used to solve an over-determined system [42] shown in eq. (2.51). Least-squares regression is used to solve the resulting system below:

$$\begin{bmatrix} \Psi_0(\boldsymbol{\xi_1}) & \Psi_1(\boldsymbol{\xi_1}) & \dots & \Psi_{N_p}(\boldsymbol{\xi_1}) \\ \Psi_0(\boldsymbol{\xi_2}) & \Psi_1(\boldsymbol{\xi_2}) & \dots & \Psi_{N_p}(\boldsymbol{\xi_2}) \\ \vdots & \vdots & \ddots & \vdots \\ \Psi_0(\boldsymbol{\xi_{n_s}}) & \Psi_1(\boldsymbol{\xi_{n_s}}) & \dots & \Psi_{N_p}(\boldsymbol{\xi_{n_s}}) \end{bmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{N_p} \end{pmatrix} = \begin{pmatrix} J(\boldsymbol{\xi_1}) \\ J(\boldsymbol{\xi_1}) \\ \vdots \\ J(\boldsymbol{\xi_{n_s}}) \end{pmatrix} \tag{2.51}$$

Stochastic collocation (SC) methods use interpolation functions and employs structured collocations points to estimate the model response. Since interpolation functions have the sifting property i.e., a unit value at the evaluation node and zero elsewhere, the coefficients of the polynomial expansion are equal to the model response at the evaluation node (collocation node). Therefore, SC methods require only a definition of the collocation grid and the expansion coefficients are automatically obtained using the model response at the grid nodes.

On the contrary, gPCE not only requires the definition of an expansion function in eq. (2.45) but also requires an approach to estimate the coefficients of the polynomial expansion (see eq. (2.51)). A detailed comparison of the two approaches was carried out by Eldred et al. [72] using a series of computational experiments using analytical models namely, the cantilever beam, short column, the Rosenbrock and the log normal ratio function with uncertain coefficients. The gPCE regression and SC methods were found to be very similar in their computational performance and accuracy. When differences

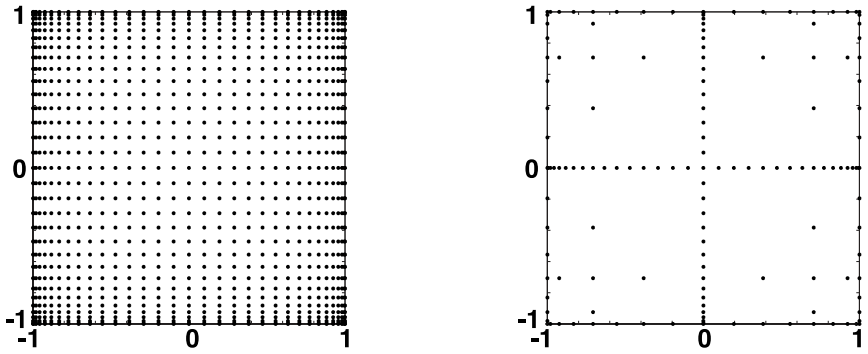


Figure 2.6: 2-D integration nodes for Chebyshev polynomials with 5 levels; Left: Tensor grid (1089 nodes) and Right: Equivalent Smolyak sparse grid (145 nodes)

arise *collocation was found to be better than the gPCE regression approach.*

2.4.3 Curse of Dimensionality : Some Cures and Limitations

When the number of input variables increases the multi-dimensional integral requires function evaluation on a tensorial grid. This leads to the *curse of dimensionality*, i.e., the number of function evaluations increases exponentially with the number of input variables $\mathcal{O}(N_p^d)$ (see eq. (2.46) and (2.50)).

Smolyak sparse grid [73] is inspired from the basic observation that the coefficients of the 1-D polynomial expansion decays rapidly. Therefore one can truncate nodes in the d-dimensional tensorial expansion (using the 1-D expansion) by neglecting terms with small coefficient values. This partly alleviate the curse of dimensionality and gives a cost complexity of $\mathcal{O}\left(N_p \log(N_p)^{d-1}\right)$ and error convergence of $\mathcal{O}\left(N_p^{-2} \log(N_p)^{d-1}\right)$. However it cannot eliminate the problem completely as it is still based on tensor grid [69]. The 2-D node distribution of points are compared between sparse grid and total-order tensor product in fig. 2.6 for Chebyshev polynomials with five levels. Recently, Ahlfeld et al. [14] extended Smolyak sparse approximation including anisotropy and adaptivity called the SAMBA framework. The framework is demonstrated to reduce the computational cost of problems with *input uncertainties as high as 20* using non-intrusive arbitrary polynomial chaos (aPC) [74, 75]. aPC targets problems of limited data availability by propagating the given information without making any assumptions. The method is quite flexible

since it propagates only the raw statistical moments. Therefore, raw test data can be used without any approximations.

Witteveen [76, 77] proposed the simplex stochastic-collocation approach on non-hypercube domains to robustly capture and accommodate non-hypercube probability spaces with an irregular shape of the parameter domain, which is common in engineering. The method is found to achieve *super-linear convergence* and a *linear increase of the initial number of samples with increasing dimensionality* and applications up to 15 *input parameters was demonstrated* in this work.

2.5 Summary

A survey of UQ methods for the forward propagation problem was presented in this chapter. One finds that there are two classes of methods, methods for higher (MC) and lower number of uncertainty variables (gPCE/SC). MC suffers from the problems of slow convergence and gPCE/SC from the curse of dimensionality. Improvements to MC mostly target convergence rate and computational cost using sampling, multifidelity and multilevel methods. Similarly for gPCE/SC improved methods target the curse of dimensionality and the scalability of computational cost to the number of input uncertainties.

Clearly MC is the best choice available when large number of uncertainties are involved and this thesis targets to improve its convergence by the use of multilevel and multifidelity control variates. But multifidelity methods depend heavily on the approximate model (G) used in the control variate approach. *The approximate model should not defeat MC's ability to handle large number of uncertainties.* This is the focus and motivation for the next chapter.

Chapter 3

Fast Adjoint-assisted Multifidelity Multilevel UQ

"Prediction is very difficult, especially if it's about the future."

— Niels Bohr

3.1 Low Fidelity Models in Multifidelity Monte Carlo

Low fidelity surrogate models are available abundantly in the literature. Smith [34] gives an exhaustive survey of these models and references. Peherstorfer [6] provides a survey of low fidelity models used in the context of multifidelity UQ. They are broadly classified as methods based on (1) Projection, (2) Regression/interpolation, and (3) Simplified

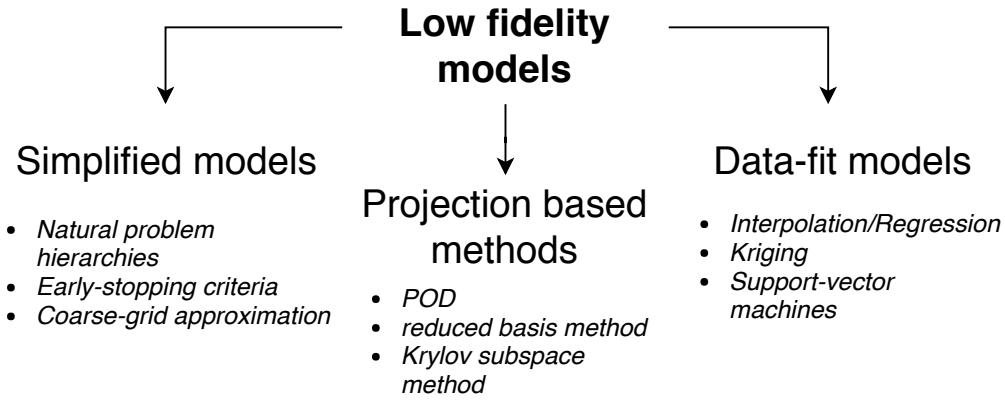


Figure 3.1: Classification of low fidelity models for multifidelity UQ [6]

models as shown in fig. 3.1.

Projection based methods construct a low fidelity model by projecting higher-dimensional state and parameters into a low-rank approximation. They are quite useful for time dependent problems and a popular method is the proper orthogonal decomposition (POD) [78]. POD constructs low-rank approximations of the system using spectral decomposition of snapshots in time. Therefore, construction of the reduced model takes significant computational effort and memory. POD carries too much information when one seeks to approximate only integral quantities of interest such as losses, drag, etc. Therefore, the runtime-ratio (w) is quite low and multifidelity approach becomes ineffective.

Data-fit models use interpolation (or extrapolation) methods to approximate the model response. A popular data-fit method is Kriging where one generates an estimated surface from a scattered set of points. In addition to the interpolation surface, Kriging also provides the confidence interval or uncertainty in the interpolation [79]. One needs to run design of experiments (DOE) to sample the response at multiple nodes to create the kriging model. Mostly the LHS sampling (see sec. 2.3.2) is used for sampling the response for DOE [80, 81]. Therefore, the computational cost of construction of the LF model is of the same order of the computational cost of the UQ problem. In addition, for multi-dimensional Kriging the computational cost scales with the number of dimensions [82]. *Therefore, data-fit method also suffers from the curse of dimensionality.*

Lastly, simplified models introduce approximations to the analysis model; for example, use of coarse meshes, lowering convergence tolerance for iterative solvers, etc. Note that MLMC and Continuation MLMC already use the aforementioned techniques. Recently, Ghate [1] proposed a MC method based on adjoint error correction of Giles et al. [2] called the inexpensive Monte Carlo (IMC). The method shows great potential as an approximate model since it is based on the adjoint solution, whose computational cost is independent of the input parameter size. *IMC as a low fidelity model retains the independence of computational cost to number of input uncertainties in the multifidelity*

control variate. The method is described in detail in the next section and analysed in a multifidelity MC framework using a model problem.

3.2 Adjoint Correction as a Low Fidelity Model

Motivated by the adjoint error correction of Giles and Pierce [83], Ghate [1] introduced an adjoint-based approximate output evaluation called the Inexpensive Monte Carlo method (IMC). The adjoint error correction is used to improve an approximation for the QoI $J(u^*_\delta, \alpha_\delta)$, where, u^*_δ is an approximation for the state u_δ due to perturbations in the design parameter α_δ as shown in eq. (3.1). The linearised state equation shown in eq. (3.2) is substituted in eq. (3.1) to simplify the adjoint correction as shown in eq. (3.3).

$$J(u_\delta, \alpha_\delta) \approx J(u^*_\delta, \alpha_\delta) - v^T_\delta \left.\frac{\partial R}{\partial u}\right|_{u_{\delta^*}} (u^*_\delta - u_\delta) + \mathcal{O}\left(||u^*_\delta - u_\delta||^2\right) \tag{3.1}$$

$$R(u_\delta, \alpha_\delta) \approx R(u^*_\delta, \alpha_\delta) - \left.\frac{\partial R}{\partial u}\right|_{u_{\delta^*}} (u^*_\delta - u_\delta) + \mathcal{O}\left(||u^*_\delta - u_\delta||^2\right) = 0 \tag{3.2}$$

$$J(u_\delta, \alpha_\delta) \approx J(u^*_\delta, \alpha_\delta) - v^T_\delta R(u^*_\delta, \alpha_\delta) + \mathcal{O}\left(||u^*_\delta - u_\delta||^2\right) \tag{3.3}$$

The adjoint v^T_δ in eq. (3.3) is evaluated at the perturbed state u_δ. But one can replace the adjoint with an approximate one v^{*T}_δ and the leading error term includes the error in the adjoint solution approximation as shown below:

$$J(u_\delta, \alpha_\delta) \approx J(u^*_\delta, \alpha_\delta) - v^{*T}_\delta R(u^*_\delta, \alpha_\delta) + \mathcal{O}\left(\max\left(||u^*_\delta - u_\delta||^2,\ ||v^*_\delta - v_\delta||^2\right)\right) \tag{3.4}$$

Although the IMC formulation in eq. (3.4) is equivalent to the moment method [1] a few major differences exists. Using IMC one can estimate the complete *pdf* of the QoI J but moment methods only provide an estimate of the statistical moments like mean, variance, etc. IMC requires the evaluation of the non-linear residual R at the perturbed state, which captures non-linearities in the QoI more effectively than the moment methods. Overall in IMC there are three possible combinations to the approximation of u^*_δ and v^{*T}_δ, which leads to the IMC 1–3 formulations.

IMC 1: Approximates the flow and adjoint values using the unperturbed baseline solution. This method has an overall leading error of second order and it is equivalent to the first order moment method.

$$\begin{aligned} u_\delta^* &= u \\ v_\delta^* &= v \end{aligned} \tag{3.5}$$

IMC 2: Approximates the flow values from the baseline tangent-linear solution and extrapolation to the perturbed state. The adjoint is approximated using the baseline solution similar to IMC 1. This method has an overall leading error of third order and it is equivalent to the second order moment method. For $\alpha \in \mathbb{R}^M$ IMC 2 requires M tangent-linear solutions in addition to the baseline non-linear and adjoint solution.

$$\begin{aligned} u_\delta^* &= u + \frac{du}{d\alpha}(\alpha_\delta - \alpha) \\ v_\delta^* &= v \end{aligned} \tag{3.6}$$

IMC 3: Approximates the flow and adjoint values using gradient extrapolation of the tangent-linear flow and adjoint solution. This is the most expensive, yet, most accurate method. The method has an overall leading error of order four. For $\alpha \in \mathbb{R}^M$, IMC 3 requires M tangent-linear flow and adjoint (Hessian) solutions. The tangent-linear adjoint solution is calculated using either the Tangent-on-Tangent or Tangent-on-Reverse strategy [84].

$$\begin{aligned} u_\delta^* &= u + \frac{du}{d\alpha}(\alpha_\delta - \alpha) \\ v_\delta^* &= v + \frac{dv}{d\alpha}(\alpha_\delta - \alpha) \end{aligned} \tag{3.7}$$

The adjoint correction has been employed as an approximate model evaluation in many

references. Engels-Putzka et al. [85] used the model to estimate the mass flow rate, total-pressure ratio and total-temperature ratio in the two-stage Darmstadt Transonic Compressor using the TRACE flow and adjoint solver developed at DLR. Luo et al. [86] used the adjoint and higher order sensitivities to approximately evaluate the aerodynamic performance of a turbine blade using Taylor expansion. The authors used a continuous adjoint approach based on inviscid Euler equations to obtain the adjoint sensitivity and Hessian solutions. Rumpfkeil et al. [87, 88] used the adjoint and Hessian solution to perform uncertainty quantification for NACA0012 airfoil subject to random surface perturbations using the IMC method. All methods in the literature only focused on using the adjoint correction for approximate QoI evaluation. But this is the first work to the authors knowledge which uses IMC as a LF model in a multifidelity control variate.

To assess the use of IMC as a LF model in multifidelity control variate it is first applied to a model problem of viscous Burgers' equation with uncertain boundary condition. Various measures such as model correlation and control variate parameter are estimates and finally used to validate the framework with numerical simulations. First the model problem is described in the next section followed by the analysis and validation.

3.3 Uncertain Viscous Burgers' Equation

3.3.1 Analytical Solution to Deterministic Problem

Consider the viscous Burgers' equation defined in the interval $x \in [-1, 1]$, with Dirichlet boundary conditions specified at the two end points $x = -1$ and $x = 1$ as shown in eq. 3.8-3.10. An uncertain parameter δ is specified at the boundary $x = -1$, which controls the perturbation from the unit value.

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}, \quad x \in [-1, 1] \tag{3.8}$$

$$u(-1, t) = u(-1) = 1 + \delta \tag{3.9}$$

$$u(1, t) = u(1) = -1 \tag{3.10}$$

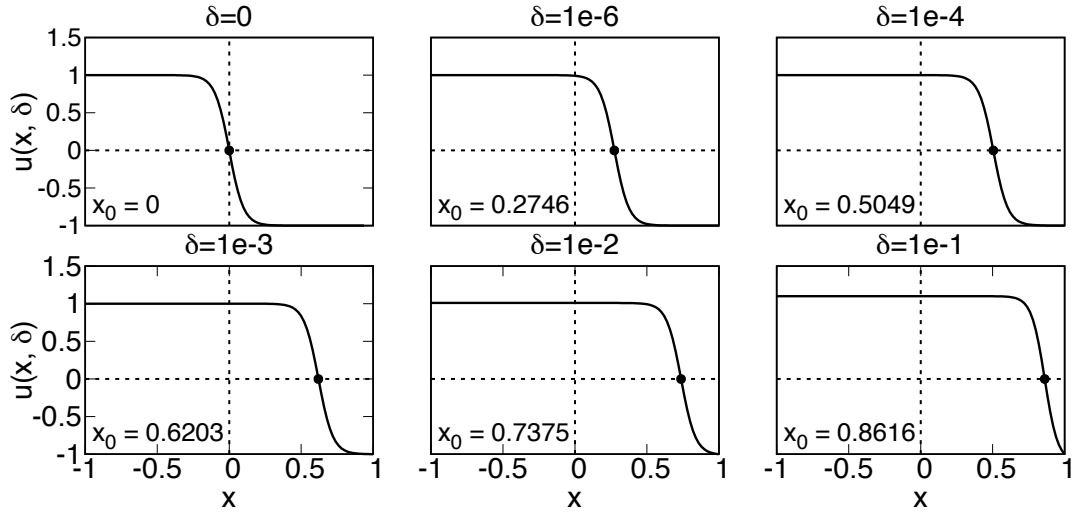


Figure 3.2: Steady-state solution of viscous Burger's equation for indicated input perturbation δ and transition point x_0 (denoted by a filled circle) and $\nu = \frac{1}{20}$

.

Xiu [89] gives the detailed analytic solution and procedure for this problem. The steady-state solution is shown in eq. (3.11), where x_0 is the location of the transition point ($u(x_0) = 0$), and A is the slope ($-A = \partial u/\partial x|_{x=x_0}$) at the transition point. The slope A requires the solution to eq. 3.12, which is solved iteratively using a root finding algorithm. Once A is determined x_0 is obtained by substituting A into any one of the eq. (3.9)-(3.10). Note that the viscosity ν determines the steepness of the shock at $x = x_0$ and a fixed value of viscosity $\nu = \frac{1}{20}$ was used in all simulations.

$$u(x) = -A \tanh\left[\frac{A}{2\nu}(x - x_0)\right] \tag{3.11}$$

$$f(A, \delta) : \left(1 + \delta + A^2\right) \tanh\left(\frac{A}{\nu}\right) - (2 + \delta)A = 0 \tag{3.12}$$

The steady state solution of the viscous Burgers equation is plotted in fig. 3.2 for various values of δ. The QoI considered in this model problem is the location of the transition point x_0 (shown using the black dot in fig. 3.2), which is readily obtained by substituting the value of A in,

$$x_0 = 1 - \frac{2\nu}{A} \tanh^{-1}\left(\frac{1}{A}\right). \tag{3.13}$$

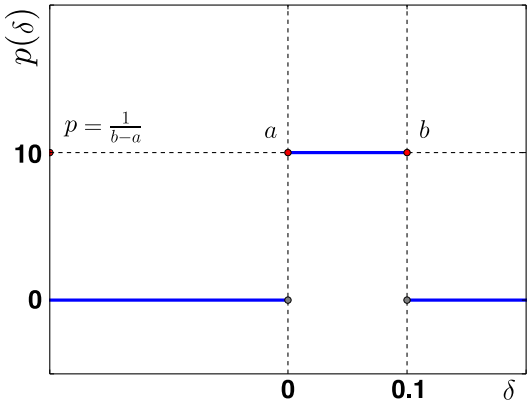


Figure 3.3: Uniform probability distribution for input uncertainty δ

Note that this is a hypersensitivity problem since a small variation of the input causes huge variations in the QoI x_0. The value of the input uncertainty δ is assumed to be equally likely in the given interval $[a, b]$. Therefore the occurrence of δ is modelled as a random variable with a uniform probability distribution $p(\delta)$, with mean $\mu_\delta = 2^{-1}(b-a)$ and variance $\mathbb{V}ar[\delta] = 12^{-1}(b-a)^2$, shown in eq. (3.14) (see fig. 3.3 for illustration of the *pdf*).

$$p(\delta) = \begin{cases} (b-a)^{-1} & \text{for } a \leq \delta \leq b \\ 0 & \text{otherwise} \end{cases} \tag{3.14}$$

For the UQ study in this work the values of $[a, b] \equiv [0,\ 0.1]$ is chosen. This model problem is simple, but sufficiently complex to demonstrate the proposed UQ methods. At the same time the viscous Burgers' equation mimics closely the compressible non-linear RANS model used in this work. Note that the finite viscosity in the viscous Burgers' equation actually yields a smooth solution instead of the the abrupt shock (jump) in the inviscid case, which ensures the validity of the adjoint correction [83]. But the regularity of the QoI x_0 is dictated by the slope $A = -\partial u/\partial x|_{x=x_0}$, whose regularity is less than that of u. This has been deliberately chosen because the higher order IMC corrections require high regularity of the QoI [1]. Therefore, they suffer from higher errors for this model problem. But the multifidelity framework is leveraged to accurately capture the QoI even under such severe circumstances with a reduced computational cost compared

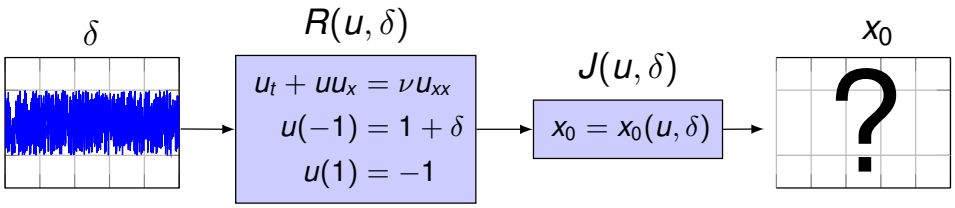


Figure 3.4: Block diagram of MC method to estimate QoI x_0 for uncertain input δ

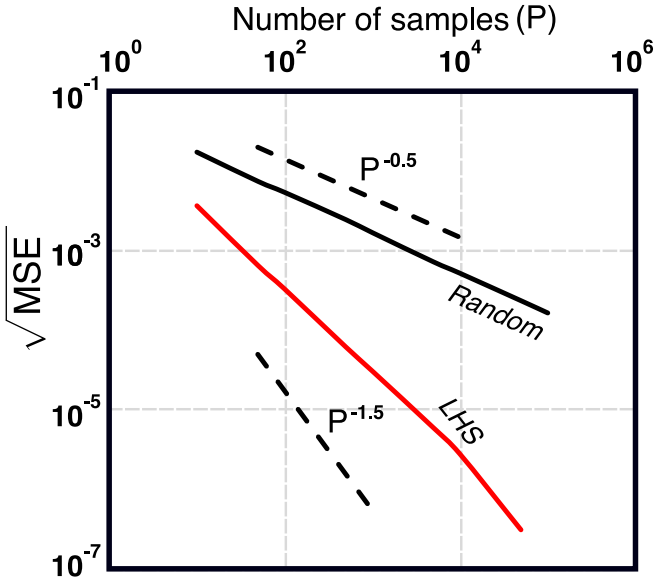


Figure 3.5: Convergence of root mean square error of QoI (transition point location x_0) with sample size between random and LHS sampling for the uncertain viscous Burgers' problem.

to a standard MC.

3.3.2 Baseline Monte Carlo Results

The overall scheme of the MC method applied to the viscous Burgers' problem with uncertain boundary parameter described in the previous section is illustrated in fig. 3.4. Random samples of input uncertainty parameter δ are generated and the analytical solution is evaluated for each sample. The output QoI x_0 is obtained for each sample (using eq. (3.13)) and one estimates the statistics such as mean and variance from the samples. The UQ model problem is simulated using random and LHS sampled MC (see

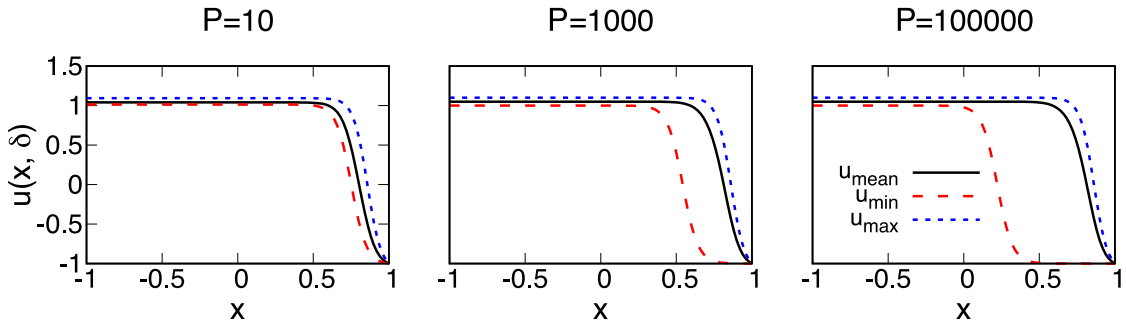


Figure 3.6: Monte Carlo solution to the viscous Burgers' equation; mean solution (u_{mean}) denoted by solid line and the minimum (u_{min}) and maximum (u_{max}) variation shown by the red and blue dotted lines for indicated number of samples (P)

sec. 2.3.1 and sec. 2.3.2 for random and LHS sampling). The reduction in root mean square error (RMSE = $\sqrt{\text{MSE}}$) with increasing number of MC samples P is compared between random and LHS sampling in fig. 3.5. The random sampling clearly exhibits the theoretical $\mathcal{O}(P^{-0.5})$ convergence rate. The LHS sampling initially has an error rate that is slightly less than the theoretical estimate of $\mathcal{O}(P^{-1.5})$ but for high accuracy, RMSE $\leq 10^{-6}$, the rate matches the theoretical limit.

The results of the mean solution $u(x, \delta)$ and the extremities obtained from MC applied to the uncertain viscous Burgers' problem are plotted in fig. 3.6 for various sample sizes. The mean is captured to a good approximation using $\mathcal{O}(100 - 1000)$ samples but it requires $\mathcal{O}(10^4 - 10^5)$ samples to capture the extremities. This shows the need for importance sampling [43] to capture the tail (extreme) events of the probability distribution. The output distribution of x_0 also exhibits high skewness (and skewed towards the boundary $x = 1$). Approximate QoI evaluation using the IMC method (described in sec. 3.2) is applied to the model problem and the approximation error is analysed in the next section.

3.3.3 Approximating Model Response using Adjoint Correction

The adjoint-based approximate QoI evaluation described in sec. 3.2 is demonstrated on the uncertain viscous Burgers' equation. To derive the adjoint for this model consider

the QoI J to be the transition point location x_0 and the non-linear state equation $R(A, \delta)$ (from eq. (3.12)) defined as defined below:

$$x_0(A, \delta) = 1 - \frac{2\nu}{A} \operatorname{arctanh}\left(\frac{1}{A}\right) \tag{3.15}$$

$$R(A, \delta) \equiv (2 + \delta)A - (1 + \delta + A^2) \tanh\left(\frac{A}{\nu}\right) = 0. \tag{3.16}$$

The adjoint, tangent-linear, and Hessian equations can be formulated as shown in eq. (3.17)-(3.19) based on the analysis shown in sec. 2.1.7. $\bar{A}$ and $\dot{A}$ in eq. (3.17)-(3.19) are the adjoint and tangent-linear solution variables.

$$\bar{A} = \frac{dx_0}{d\delta} = \left(\frac{\partial R}{\partial A}\right)^{-1} \frac{\partial x_0}{\partial A} \tag{3.17}$$

$$\dot{A} = \frac{dA}{d\delta} = \left(\frac{\partial R}{\partial A}\right)^{-1} \frac{\partial R}{\partial \delta} \tag{3.18}$$

$$\frac{d\bar{A}}{d\delta} = \frac{\partial \bar{A}}{\partial A} + \frac{\partial \bar{A}}{\partial \delta} = \frac{\partial}{\partial \delta}\left(\left(\frac{\partial R}{\partial A}\right)^{-1} \frac{\partial x_0}{\partial A}\right) + \frac{\partial}{\partial A}\left(\left(\frac{\partial R}{\partial A}\right)^{-1} \frac{\partial x_0}{\partial A}\right) \tag{3.19}$$

For the IMC estimator, one first linearises about the mean of the input disturbance $\delta \in [0, 0.1]$, which is $\mu_\delta = 0.05$. The perturbation about this mean can be evaluated from the adjoint and tangent-linear solution using any of the IMC schemes described in eq. (3.5)-(3.7).

Model	$\mathbb{E}[x_0]$ (% *error*)	$\sigma(x_0)$ (% *error*)	$\rho(x_0)$	$\beta(x_0)$
Exact	0.80728	5.28691×10^{-2}	-	-
IMC 1	0.82335 (+2%)	3.17072×10^{-2} (−40%)	0.8760	1.4603
IMC 2	0.78691 (−2.5%)	7.11764×10^{-2} (+35%)	0.9446	0.7015
IMC 3	0.71554 (−11%)	1.71379×10^{-1} (+225%)	0.9484	0.2925

Table 3-A: Comparison of mean and standard deviation between exact solution and IMC $1-3$ methods and Pearson correlation ρ and control variate β between the exact and model solution

In fig. 3.7 the QoI obtained from the various IMC estimators are plotted for a range of perturbations δ along with the exact value. The mean and standard deviation for the exact and IMC $1-3$ models and the Pearson correlation ρ between the exact and IMC

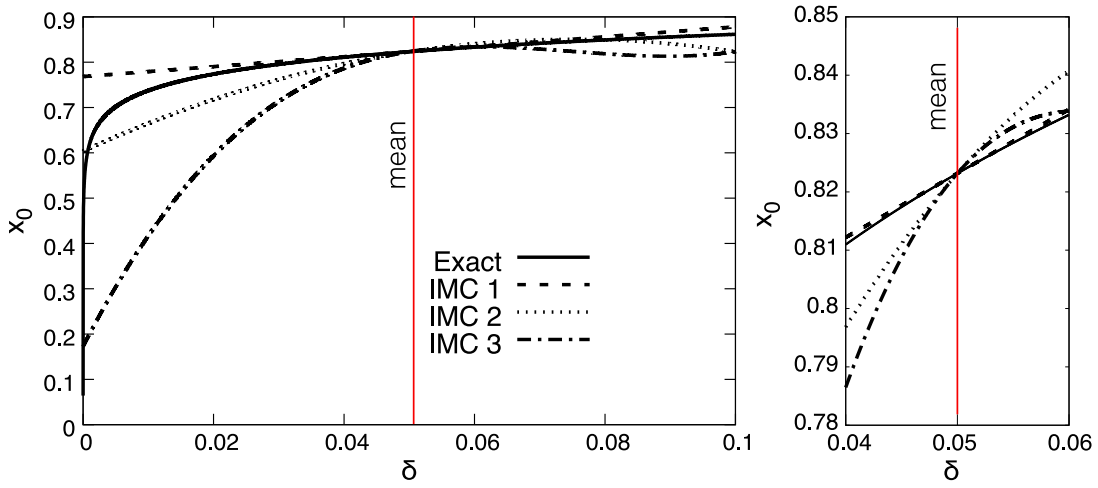


Figure 3.7: Comparison of transition point location x_0 between exact solution and approximate IMC $1-3$ models for a range of perturbation values $\delta \in (0, 0.1)$; zoomed view near the mean value shown on the right.

$1-3$ are shown in Table 3-A. An interesting aspect of IMC 2 and 3 for this problem is that they have *quite large errors in predicting the mean and variance* compared to the IMC 1 model. But they closely follow the trend of the exact solution much better than IMC 1, which is reflected in the *high correlation*. In fact, Ghate [1] notes a similar trend in IMC 1 and 3 approximation of lift for flow over an airfoil and proved that quadratic extrapolation in IMC 3 requires higher regularity of the function space (of the solution and QoI) than the adjoint corrected linear extrapolation of IMC 1. For the Burgers problem the QoI x_0 has a lower regularity than the solution u, which introduces errors in the higher order corrections in IMC 2/3. Interestingly the higher order corrections capture the trends better than the lower order ones for large input perturbations and the opposite is true for smaller perturbations (see fig. 3.7).

Ghate [1] was mainly interested in using the IMC to approximate the mean and variance of the QoIs. In contrast in this thesis they are used as low fidelity models in a multifidelity control variate. Therefore, the correlation is the key parameter that determines the performance. Even when the errors in the mean and standard deviation values of the QoI are high due to problems of regularity if the correlations are well approximated by the IMC model then the multifidelity Monte Carlo (MFMC) can still successfully capture

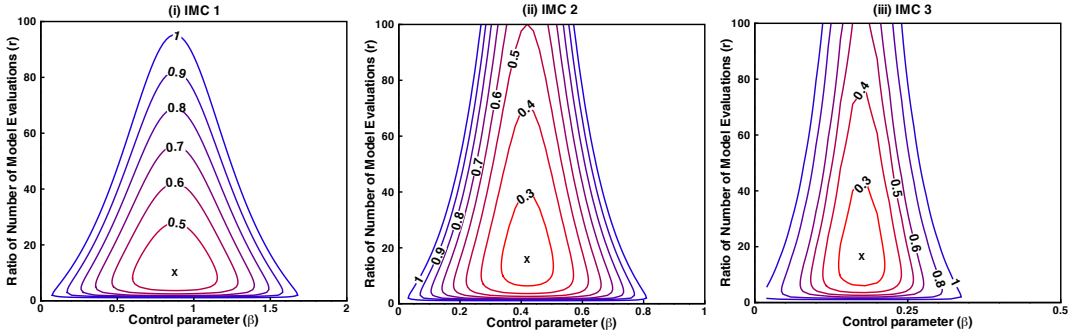


Figure 3.8: Contour plots of ψ for IMC $1-3$ LF models using runtime ratio $w = 30$ (see tab. 3-A for other details)

the variations in QoI.

IMC 2 and 3 have the highest Pearson coefficient, which makes them the best candidates amongst the IMC 1-3 models for the multifidelity control variate. When the Hessian and tangent-linear solutions are expensive to calculate or not available, IMC 1 is the next better choice in terms of computational cost and better correlation. Moreover, the computational cost of the adjoint solution is independent of the number of input parameters whereas the computational cost of the tangent-linear and Hessian solutions increases with increasing number of input parameters. *IMC* 1 *is the preferred method when the number of input parameters are high.*

To estimate the effect of under sampling, the ψ parameter from the analysis of Ng [3] in sec. 2.3.4.1 is plotted for the various IMC models in fig. 3.8. The runtime ratio between the IMC models and the exact Newton solution of the Burgers' equation is approximately $w \approx 30$. This value of w is used along with parameter ρ from Table 3-A to generate figs. 3.8(i)-(iii). Values of ψ less than unity indicate that the multifidelity estimator variance is lower than the MC estimate. Sample estimates around '$\times$' (in fig. 3.8) are acceptable although optimal variance reduction cannot be achieved when the estimate is not exactly at '$\times$'. The deterioration of variance reduction due to under-sampling is less severe for the IMC 2 and 3 compared to IMC 1. But the overall acceptable error band (over/under estimation) of the control variate parameter β is higher for IMC 1.

3.3.4 Results of the Multifidelity Approach

The root mean squared error (RMSE) reduction with sample size between the three different control variate MC (with IMC 1-3 as low fidelity models) is compared against the plain MC estimate in fig. 3.8. The relative reduction in RMSE is indicated in the plot. For the same number of samples the RMSE is reduced by 54% for IMC 2 and 3 and by 35% for the IMC 1 multifidelity control variate. In fact one can predict the reduction in computational cost with respect to the classical MC using the estimator ϕ derived in sec. 2.3.4.2 of ch. 2. In fig. 3.3.4, the contours of percentage reduction in RMSE compared to classical MC ($1-\sqrt{\phi}$ %) is plotted for different values of correlation ρ and runtime ratios w. The percentage reduction in RMSE for IMC 1-3 is indicated in the figure using the "$*$" symbol. The predicted values of reduction from the analysis of MFMC are consistent with the measured ones in fig. 3.3.4. Note that a region $1-\sqrt{\phi} < 0$ exists (greyed in fig. 3.3.4), where the cost of MFMC is higher than the classical MC. Therefore, use of MFMC does not provide any benefit and it is worthwhile to use classical MC in this region.

A major advantage of the proposed multifidelity method with IMC as an LF model is that the adjoint solution is already available in a gradient based design optimisation framework. Additional model reductions, samples, or design-of-experiments (DOE) need not be evaluated to fit surrogate models (see sec. 3.1). But the Hessian and tangent-linear solutions can be quite expensive to evaluate for IMC 2 and 3, especially for problems with large number of uncertainties. For the model problem involving the QoI x_0 one finds that IMC 1 is sufficiently accurate LF model that reduces the computational cost by 35% compared to MC.

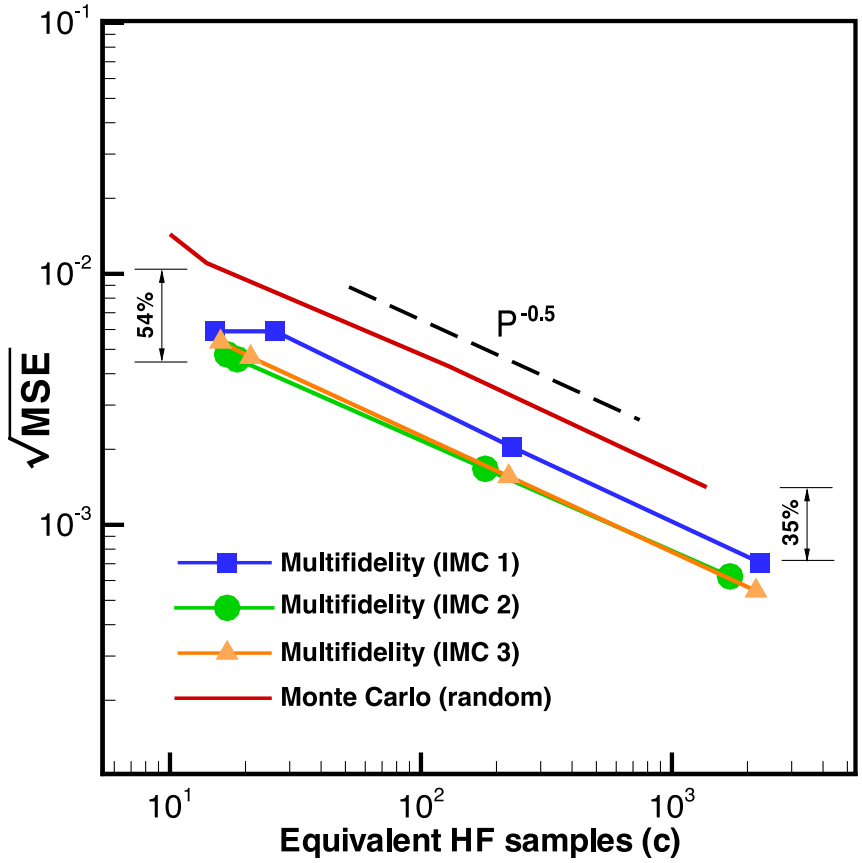


Figure 3.9: Mean squared error (MSE) reduction with sample size of the control variate MC using different LF models compared against MC with no LF model

3.4 Adjoint-assisted Multilevel Multifidelity Control Variate (FastUQ)

Recently Geraci et al. [4] proposed a multilevel multifidelity framework (MLMF) to achieve further reduction in computational cost by combining the multifidelity control variate in MLMC. In the previous section the multifidelity control variate based on the IMC was shown to achieve significant reduction in computational cost using a model problem. In addition the computational cost of QoI evaluation using IMC 1 model is independent of the number of input uncertainties. Using the approximate QoI evaluation of IMC as the low-fidelity model in the multilevel multifidelity framework has the benefit that it reduces the number of samples and maintains the MC property; independence of convergence rate with respect to the number of input uncertainties. This combination is

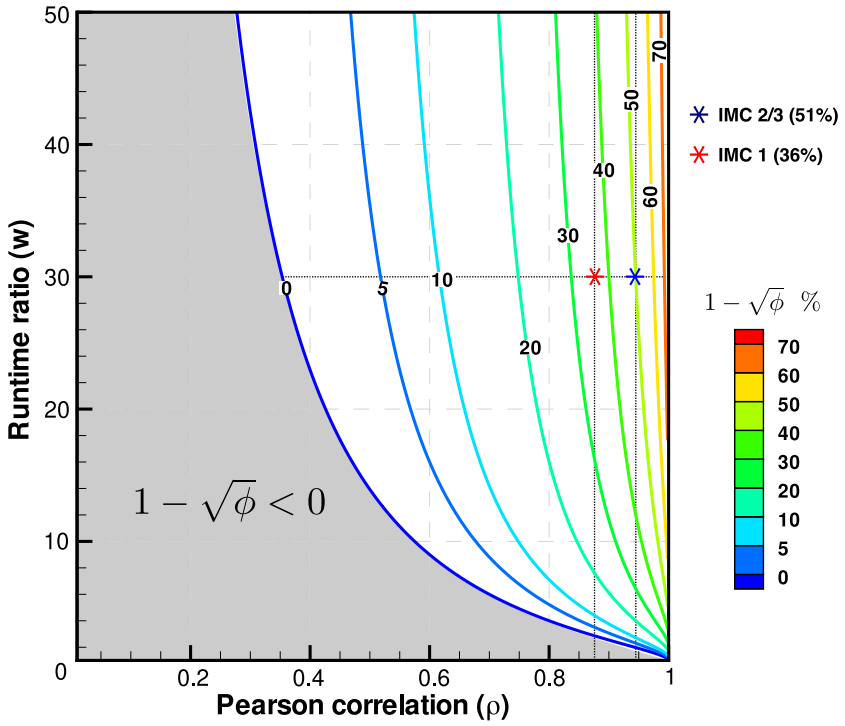


Figure 3.10: Percentage reduction in computational cost of MFMC with respect to the classical MC ("$*$" denotes the predicted reduction for IMC 1-3 models)

the proposed adjoint-assisted multilevel and multifidelity framework (FastUQ).

To analyse the properties of the MLMF, the Y correlation form of the multilevel multifidelity framework of Geraci et al. [4] is rederived using the multifidelity cost model proposed by Ng et al. [3]. This approach has been adopted in this work in order to show the connection between the two models. The final expressions in the present work slightly differ from the results of Geraci [4] due to the definition of the cost model. For simplicity the number of grid levels for HF and LF models is assumed to be the same (see ref. [4]). The MLMF estimator in eq. (3.20) is obtained by combining the SMLMC estimator of Giles [19] in eq. (2.36) and the multifidelity estimator of Ng [3] is shown in eq. (2.27), where P_l is the number of samples per level, L is the total number of levels and $J^{MLMF}_{N,\,p}$ is the MLMF estimator for the finest level N.

$$J^{MLMF}_{N,\,p} = \sum_{l=0}^{L} \frac{1}{P_l} \sum_{i=1}^{P_l} \mathbb{E}[\overline{Y}_l^{(i)}] \tag{3.20}$$

The difference function $\overline{Y}_l$ is obtained using the multifidelity estimator as shown in eq. (3.21), where Z_l is the difference function of the LF model G and Y_l is the function defined below:

$$\overline{Y}_l = Y_l - \beta_l \left(Z_l - \mathbb{E}[Z_l] \right) \tag{3.21}$$

$$Z_l = \begin{cases} G_{N_0} & \text{if } l = 0 \\ G_{N_l} - G_{N_{l-1}} & \text{if } l \neq 0 \end{cases} \tag{3.22}$$

3.4.1 Optimal Sample Allocation between Levels and Models

Let P_l and P_l^{LF} denote the number of sample evaluations of the HF and LF models at level l. Let r_l denote the fraction of HF evaluation to the LF evaluation $r_l = P_l^{LF}/P_l$. The cost of computation on a level is expressed in terms of the equivalent number of HF sample evaluations as shown in eq. (3.23), where c_l is the ratio of computational cost of evaluating a high fidelity sample at level l and the finest level N (note that $c_N = 1$ and $c_l \leq 1$). Therefore, C_l now contains both the relative computational cost between models and across levels.

$$C_l = c_l P_l \left(1 + \frac{r_l}{w_l} \right) \tag{3.23}$$

$$C_N^{MLMF} = \sum_{l=0}^{L} C_l = \sum_{l=0}^{L} c_l P_l \left(1 + \frac{r_l}{w_l} \right) \tag{3.24}$$

The total computational budget considering all levels (C_N^{MLMF}) is shown in eq. (3.24). The variance of this estimator can be optimised for β_l, the final optimal variance and the optimal number of HF samples per level is defined as,

$$\mathbb{V}ar\left[J_{N,\,p}^{MLMF} \right] = \sum_{l=0}^{L} \frac{1}{P_l} \mathbb{V}ar[Y_l] \Lambda(r_l) \tag{3.25}$$

$$P_l^{opt} = \frac{2}{\epsilon^2} \left[\sum_{k=0}^{L} \sqrt{\frac{\mathbb{V}ar(Y_k) c_k}{1 - \rho_k^2}} \; \Lambda_k \left(r_k^{opt} \right) \right] \sqrt{(1 - \rho_l^2) \frac{\mathbb{V}ar(Y_l)}{c_l}}. \tag{3.26}$$

The values of r_l^{opt} and Λ_l are obtained from eq. (3.27)-(3.28) respectively and ρ_l is the Pearson correlation of the HF and LF models per level.

$$r_l^{opt} = \sqrt{\frac{\rho_l^2}{1-\rho_l^2} w_l} \tag{3.27}$$

$$\Lambda_l(r_l) = \left(1 - \frac{r_l - 1}{r_l}\rho_l^2\right) \tag{3.28}$$

3.4.2 An Insight on Model Correlation and Optimal Sampling

The optimal ratio of HF to LF samples r^{opt} in eq. (3.27) obtained for the MLMF is the same as the one obtained for the multifidelity MC in eq. (2.31). Moreover, the factor $\Lambda(r^{opt})$ is related the reduction factor ϕ (see eq. (2.33)) of the multifidelity estimator using the relation shown below:

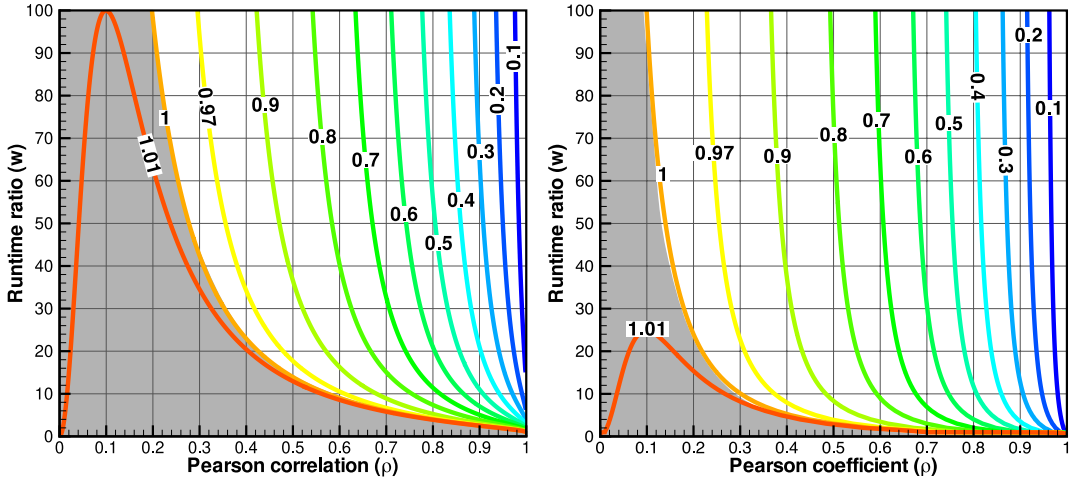
$$\phi(r^{opt}) = \left(1 + \frac{r^{opt}}{w}\right)\Lambda(r^{opt}) \tag{3.29}$$

Note that since $\frac{r^{opt}}{w} \geq 0$ is a positive quantity the identity $\phi(r^{opt}) \geq \Lambda(r^{opt})$ holds. *Therefore eq.* (3.29) *guarantees improvement of the multifidelity control variate when introduced in a multilevel framework.* Eq. (3.29) succinctly summarise the effect of introducing the multifidelity control variate in a multilevel framework, which is an important result obtained in this thesis.

To simplify the analysis consider the correlation between the HF and LF model across levels to be the same ($\rho_l = \rho$). Then the optimal number of samples in eq. (3.26) reduces to the simple expression:

$$P_l^{opt} = \frac{2}{\epsilon^2}\sum_{k=0}^{L}\left[\sqrt{\mathbb{V}ar(Y_k)c_k}\ \Lambda_k\left(r_k^{opt}\right)\right]\sqrt{\frac{\mathbb{V}ar(Y_l)}{c_l}} \tag{3.30}$$

The optimal number of samples per level P_l^{opt} in eq. (3.30) is exactly similar to the MLMC results (in eq. (2.40)) except for the extra Λ term. In ref. [4] Λ was simply referred to as an additional penalty term to the SMLMC due to the additional evaluation of the LF



(a) Contour of $\phi(w, \rho)$ (b) Contour of $\Lambda(w, \rho)$

Figure 3.11: Contour plot of $\phi(r^{opt})$ and $\Lambda(r^{opt})$ for various values of runtime ratio w and Pearson coefficient ρ (region where $\phi(r^{opt}) \geq 1$ and $\Lambda(r^{opt}) \geq 1$ is indicated in grey)

model.

In this thesis a different interpretation is given to Λ based on the optimal number of samples. The contour plot comparing $\phi(r^{opt})$ and $\Lambda(r^{opt})$ for a range of values of correlation ρ and runtime ratio w is shown in fig. 3.11(a-b). The value of $\Lambda(r^{opt})$ is mostly ≤ 1 indicating a reduction in the number of samples compared to the standard MLMC to achieve the same MSE. Therefore, $\Lambda(r^{opt})$ is an *additional sample reduction factor due to the multifidelity control variate.* Note that the multilevel actually *improves the multifidelity reduction factor* ϕ and the relation is shown in eq. (3.29).

A small region (indicated by the grey area in fig. 3.30(b)) exists, where $\Lambda(r^{opt}) > 1$ indicating that the MLMF requires more samples to achieve the same MSE compared to the SMLMC. From fig. 3.11(b) one can infer that the runtime ratio w should be ≥ 10 and correlation ρ should be ≥ 0.6 for the MLMF to be effective. P_l^{opt} is directly proportional to $\Lambda_l(r_l^{opt})$, which is a strong function of the correlation. For large runtime ratios w, $\Lambda(r^{opt}) \propto \rho$ and is almost independent of runtime ratio. *The success of the MLMF is strongly dependent on how well the LF model correlates with the HF model.* In addition $P_l^{opt} \propto \epsilon^{-2}$ so lowering the tolerance by half quadruples the number of samples per level

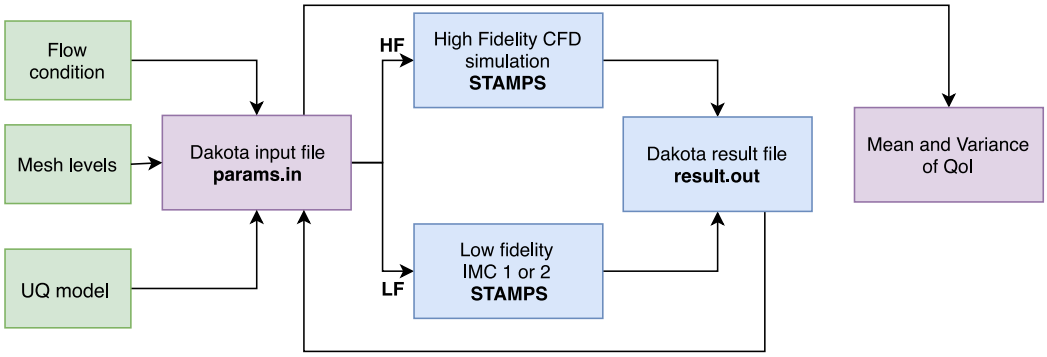


Figure 3.12: UQ workflow using Dakota and STAMPS

(similar to the classical MC and SMLMC convergence).

3.5 FastUQ implementation in Dakota toolkit

In this work the Dakota [38] toolkit was used to implement the FastUQ method for quantifying aerodynamic uncertainties arising in turbomachinery and external flow problems. The parallel version of Source Transformation Adjoint Multi-Purpose Solver (STAMPS) [40] developed as part of the thesis work was used as the high fidelity aerodynamic model. Parallel discrete adjoint and tangent-linear solver also developed in the present work (as part of STAMPS) was used to implement the IMC low fidelity model.

The overall block diagram of the implementation using Dakota and STAMPS is presented in fig. 3.12. A flexible but loose coupling of the STAMPS flow solver and the Dakota toolkit was implemented using input files. The multilevel meshes, flow and boundary conditions and the UQ model are used to generate the Dakota input parameters file and the input files for the HF/LF solver. The existing multilevel multifidelity simulation setup in Dakota was used and appropriate parameters are set in the Dakota parameter file (params.in). Unix shell scripts are then used to run the type of model (HF/LF) and to generate input files for STAMPS for the given sample (input uncertainty) generated by Dakota. The HF/LF simulation is then carried out for the given sample and the QoI is output to a results file, which is then read by Dakota. Dakota determines if more

sample evaluations are required or the simulation should be terminated based on the MSE tolerance specified by the user.

In this work only the synchronous mode of Dakota was used i.e., the samples are evaluated one after another serially. But the individual HF samples are run in parallel using multiple MPI ranks (or processor cores). Since the asynchronous or MPI mode in Dakota requires more control over the computing environment and infrastructure, it was not attempted in the present study. Future studies involving UQ of large 3-D cases can benefit from using asynchronous and parallel execution mode of Dakota. The STAMPS solver theory, parallel implementation and validation are presented in the next chapter. The adjoint and tangent-linear solution required for the IMC is also outlined in the next chapter.

3.6 Summary

A brief survey of low fidelity models used for multifidelity MC was presented. The IMC model was chosen due to its favourable properties such as low computational cost, which is independent of the number of input uncertainties. IMC as a low fidelity model in the multifidelity framework of Ng [3] was analysed using a model problem of viscous Burgers' equation with uncertain boundary condition. All variants of the IMC model showed good correlation with the HF results.

A good correlation does not translate to better prediction of the QoI statistics. It merely indicates that the LF model captures the trends in the HF model faithfully. For the Burgers‘ problem, IMC 1 had the best predictions for mean and standard deviation of the QoI x_0 but had a lower correlation with the HF model compared to IMC 2 and 3. The regularity of the QoI x_0 was lesser than the regularity of the solution u. Since higher order IMC corrections require higher regularity of the QoI [1] one finds large errors in IMC 2 and 3 for this problem. But the general trends in the function are still captured well by the IMC model even in this circumstance. This shows the superiority

of the multifidelity IMC proposed in this thesis. By using the IMC as a LF model in the control variate one can improve the range of its applicability to problems having lesser QoI regularity and to larger range of input perturbations.

The multilevel multifidelity framework of Geraci et al. [4] was re-derived using a cost model proposed by Ng [3]. Some useful insights on the performance of the MLMF was shown. For large runtime ratios ($w \gg 20$) the MLMF performance is purely dictated by the correlation between the LF and HF models and is independent of the runtime ratio. Introducing the multilevel guarantees improvement of the multifidelity MC. The FastUQ framework was proposed using the IMC model in the MLMF. The overall implementation of FastUQ using Dakota toolkit was outlined. STAMPS aerodynamic solver was used as the HF model and the parallel discrete adjoint and tangent-linear solver in STAMPS (developed as part of this thesis) was used to implement the IMC model, which is discussed in the next chapter.

Chapter 4

Aerodynamic Model and Implementation

"CFD is Constant Frustration and Disappointment"

— Bram Van Leer

The HF model used to estimate the variations in the aerodynamic QoI due to geometric variations is described in detail in this chapter. In addition, the implementation details of the adjoint and tangent linear solver used in the LF IMC model is described. The parallelisation techniques used to reduce computational time is also presented.

The non-linear flow solver used in this thesis is based on the steady Reynolds Averaged Navier Stokes (RANS) equation. Source Transformation Adjoint Multi-Purpose Solver (STAMPS), the in-house code developed in QMUL by Mueller et. al. [40] was used in this thesis. Capabilities for turbomachinery applications such as periodic boundaries, variable swirl inlet, new objective functions and complete parallelisation of the flow, adjoint, and tangent linear solvers using the Message Passing Interface (MPI) [90] was developed as part of this thesis by the present author. The overall solver development was a joint effort between the present author and Xu [91].

In addition, the non-linear flow solver, adjoint and tangent linear solvers were re-written to utilise the latest developments in the Algorithmic Differentiation (AD) tool Tape-

nade [92]. The tool is now capable of simultaneously differentiating the source code in the forward and reverse mode and automatically generates differentiated version of user-defined data structures common to both modes. This simplifies the automation of adjoint and tangent-linear solver development.

4.1 Governing Equations in Rotational Frame[1] of Reference

The governing RANS equations are solved in the relative (non-inertial) frame of reference using pseudo-time τ in conservative form as shown in eq. (4.1). Note that the relative frame of reference rotates with a constant angular velocity $|\vec{\omega}|$ along the axis $\vec{\omega}/|\vec{\omega}|$.

$$\frac{d}{d\tau}\int_{\mathcal{V}} \mathbf{U} d\mathcal{V} - \int_{\mathcal{S}} \left(\mathbf{F}_c^r - \mathbf{F}_\omega - \mathbf{F}_v\right) d\mathcal{S} = 0 \tag{4.1}$$

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho\mathbf{u} \\ \rho E \end{bmatrix}, \qquad \mathbf{F}_c^r = \begin{bmatrix} \rho\mathbf{u_r}\cdot\mathbf{n} \\ \rho\mathbf{u}\mathbf{u_r}\cdot\mathbf{n} + p\mathbf{n} \\ \rho(E+p)\mathbf{u_r}\cdot\mathbf{n} \end{bmatrix}, \qquad \mathbf{F}_{\vec{\omega}} = \begin{bmatrix} 0 \\ -\rho(\vec{\omega}\times\mathbf{u}) \\ -p(\vec{\omega}\times\mathbf{r}) \end{bmatrix}$$

$$\mathbf{F}_v = \begin{bmatrix} 0 \;\Big|\; \bar{\tau}\cdot\mathbf{n} \;\Big|\; (\mathbf{u}\cdot\bar{\tau})\cdot\mathbf{n} - \kappa(\mathbf{n}\cdot\nabla T) \end{bmatrix}^T \tag{4.2}$$

$$E = \frac{p}{\rho(\gamma-1)} + \frac{1}{2}\left(|\mathbf{u_r}|^2 - |\vec{\omega}\times\mathbf{r}|^2\right) \tag{4.3}$$

$$\bar{\tau} = \tau_{ij} = \mu\left(\nabla\mathbf{u} + \nabla\mathbf{u}^T\right) + \delta_{ij}\lambda\nabla\cdot\mathbf{u} \tag{4.4}$$

$$\mathbf{u_r} = \mathbf{u} - \vec{\omega}\times\mathbf{r} \tag{4.5}$$

This formulation can be used for the flow analysis of a single body rotating in isolation. For example, helicopter rotors in hover, propellers in forward flight (neglecting fuselage interactions) and single blade row turbomachinery calculations. In fact, complete simu-

[1] Rotational frame of reference implemented by Xu and extended to parallel non-linear solver by the present author

lation of radial compressor with vaneless diffuser is possible using this formulation. The particular form of the equations are derived in relative frame but using absolute velocities. The transformation from the relative to the absolute velocity is given in eq. (4.5). Agarwal [93] states that using the absolute-velocity formulation allows for more accurate evaluation of the fluxes in a finite-volume scheme and essential for obtaining accurate solutions on non-uniform grids.

The working fluid is assumed to be air (perfect gas) obeying the equation of state in eq. (4.6). The fluid viscosity is modelled as a sum of laminar and turbulent contributions using the eddy viscosity approach as shown in eq. (4.7). The laminar fluid viscosity is assumed to follow the Sutherland law in eq. (4.8). The thermal conductivity of air κ is a sum of the laminar (κ_L) and turbulence (κ_T) contributions as shown in eq. (4.9).

$$p = \rho R T \tag{4.6}$$

$$\text{where,} \quad R = c_p - c_v = 287.3 \text{ kg m}^{-3}, \quad \gamma = \frac{c_p}{c_v} = 1.4$$

$$\mu = \mu_L + \mu_T \tag{4.7}$$

$$\mu_L = \mu_L(T) = \frac{1.45 T^{3/2}}{T + 110} 10^{-6} \text{ kg m}^{-1} \text{ s}^{-1} \tag{4.8}$$

$$\kappa = \kappa_L + \kappa_T \tag{4.9}$$

$$\kappa_L = c_p \frac{\mu_L(T)}{Pr_L} \text{ and } \kappa_T = c_p \frac{\mu_T(T)}{Pr_T} \quad \text{where, } Pr_L = 0.72, Pr_T = 0.9 \quad \text{(for air)}$$

The RANS closure (via μ_T) is achieved using the standard one equation Spalart-Allmaras (SA) Turbulence model and the constants are obtained from ref. [94]. The SA model requires calculation of distance to wall for each mesh node. An exact wall distance algorithm based on "*Closest Point on Triangle to Point*" algorithm is implemented from sec 5.1.5 of ref. [95]. For quadrilateral surface elements one triangulates the mesh to obtain the wall distance.

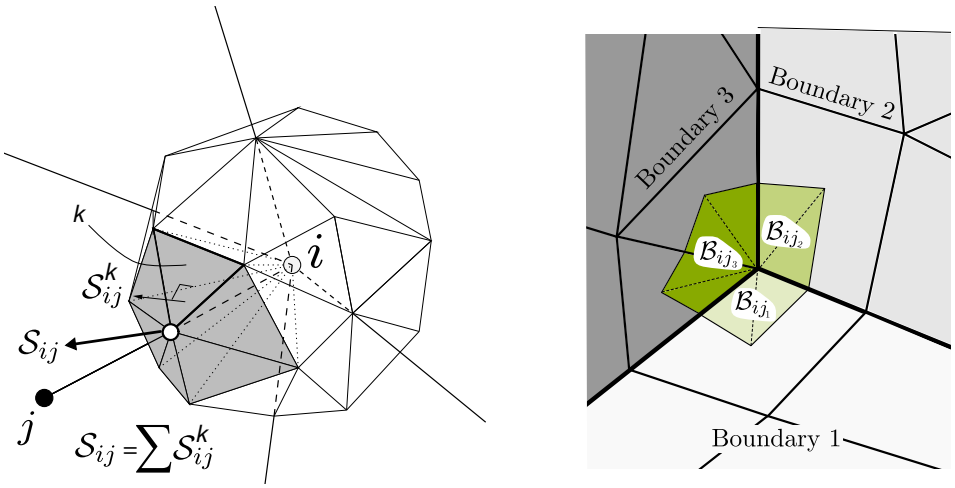


Figure 4.1: Dual volume illustrated with internal edge connecting node i to j and edge coefficient $\mathcal{S}_{ij}$ from [97] (left) and the boundary dual with edges $\mathcal{B}_{ij}$ (right)

4.2 Edge-based Second Order Finite Volume Spatial Discretisation

An edge-based node-centred second order finite volume method [96] is used in STAMPS to discretise the spatial fluxes. A medial dual approach is utilised to construct the dual elements from the primary mesh (see fig. 4.1). Details on the construction of the medial dual can be found in ref. [97].

$$\mathcal{V}_i \frac{d}{d\tau}(\mathbf{U}_i) - \sum_{j \in in} \left(\mathbf{F}_{c,ij}^r - \mathbf{F}_{\omega,j} - \mathbf{F}_{v,ij}\right) \mathcal{S}_{ij} - \sum_{j \in bc} \left(\mathbf{F}_{c,ij}^r - \mathbf{F}_{\omega,j} - \mathbf{F}_{v,ij}\right) \mathcal{B}_{ij} = 0$$

$$\mathcal{V} \frac{d\mathbf{U}}{dt} - R[\mathbf{U}] = 0 \quad (4.10)$$

The edges are segregated into internal (in) and boundary (bc) edges and their respective edge fluxes ($\mathbf{F}_{ij}$) are summed up using the internal and boundary edge coefficients $\mathcal{S}$ and $\mathcal{B}$ to yield the residual flux in eq. (4.10). But internally in the STAMPS the residual calculation is implemented as single loop over edges to reduce memory access and avoid storage of edge fluxes.

4.2.1 Convective Flux via Roe Approximate Riemann Solver

The Roe approximate Riemann solver is used to obtain the edge flux in the rotational frame of reference [98] as shown in eq. (4.11). The Harten and Hymen entropy fix [99] is applied to the acoustic part of the convective fluxes $|\mathbf{u}+c|$ and $|\mathbf{u}-c|$, where c is the speed of sound, but not to the entropic wave $|\mathbf{u}|$. A^r_{Roe} is the Roe dissipation matrix in the rotational frame. The left ($\mathbf{U}_L$) and right ($\mathbf{U}_R$) states of the edge are computed using a MUSCL [100] reconstruction technique as shown in eq. (4.12). To prevent over- and undershoots in the solution a differentiable Venkatakrishnan [101] gradient limiter (ϕ in eq. (4.12)) with the corrections proposed by Wang [102] and Michalak [103] was implemented in STAMPS. The gradients are obtained using the standard Green-Gauss formula [100].

$$\mathbf{F}_{face} = \frac{\mathbf{F}^r_c(\mathbf{U}_L) + \mathbf{F}^r_c(\mathbf{U}_R)}{2} + \frac{1}{2}\left|A^r_{Roe}\right|(\mathbf{U}_R - \mathbf{U}_L) \tag{4.11}$$

$$\mathbf{U}_L = \mathbf{U}_i + \phi_i \nabla\mathbf{U}_i \cdot (\mathbf{x}_j - \mathbf{x}_i), \qquad \mathbf{U}_R = \mathbf{U}_j + \phi_j \nabla\mathbf{U}_j \cdot (\mathbf{x}_i - \mathbf{x}_j) \tag{4.12}$$

4.2.2 Viscous Flux via Green-Gauss Gradient Formula

Viscous flux computation requires both the flow state $\mathbf{U}$ and its gradient $\nabla\mathbf{U}$ defined at a flux face A-B (see fig. 4.2). Let $\overline{\nabla\mathbf{U}}$ denote the arithmetic average of the flow state across an edge. Then a skew corrected gradient interpolation [104] at a flux face can be obtained using the formula below,

$$\overline{\nabla\mathbf{U}} = \frac{(\nabla\mathbf{U})_i + (\nabla\mathbf{U})_j}{2} \tag{4.13}$$

$$(\nabla\mathbf{U})_{face} = \overline{\nabla\mathbf{U}} - (\overline{\nabla\mathbf{U}} \cdot \mathbf{t})\frac{\mathbf{t}}{\mathbf{n}\cdot\mathbf{t}} + \frac{\mathbf{U}_j - \mathbf{U}_i}{\|\mathbf{x}_j - \mathbf{x}_i\|}\frac{\mathbf{t}}{\mathbf{n}\cdot\mathbf{t}}. \tag{4.14}$$

In the above formulation, $\mathbf{t}$ is the unit vector pointing from left to right node, and $\mathbf{n}$ is the unit normal vector of the flux face, as illustrated in Fig. 4.2.

For a skewed mesh (see fig. 4.2) $\mathbf{n}\cdot\mathbf{t}$ is small because the edge and face are highly non-orthogonal. Therefore, when $\mathbf{n}\cdot\mathbf{t}$ divides the correction term it has the equivalent effect

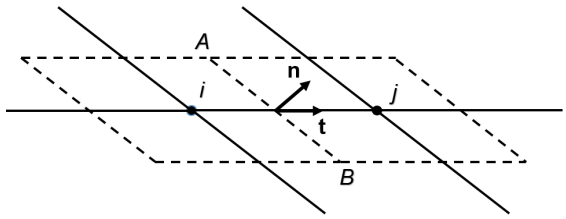


Figure 4.2: Sketch illustrating a flux face A-B between nodes i and j. The vectors $\mathbf{t}$ and $\mathbf{n}$ are the edge and the face normal vectors. The solid lines represent the primary mesh and the dashed lines represent its dual.

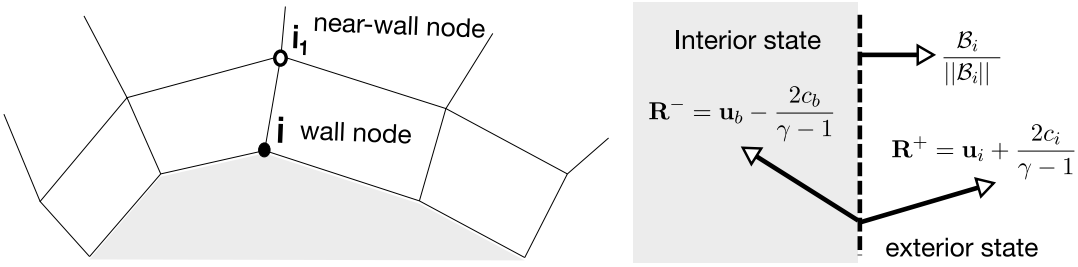


Figure 4.3: Wall boundary node i with near-wall node i_i (left) and characteristics at a boundary (right)

of increasing the resulting gradient on the face and thus adding more dissipation to the scheme. This added dissipation enhances the robustness of the solver in the presence of highly non-orthogonal meshes. The main application area of STAMPS is aerodynamic design, therefore robustness is given higher priority than accuracy.

4.3 Turbomachinery Boundary Conditions

The following three additional types of new boundary conditions have been implemented in STAMPS.

4.3.1 Variable Swirl Inlet with Specified Total Conditions

A variable swirl inlet with specified total-pressure and total-temperature has been implemented with option to specify piecewise-linear radial profiles for the same. The inlet values are obtained from the total conditions based on the characteristic analysis in ref. [105]. The profile is specified as a piecewise linear polynomial for the swirl angle,

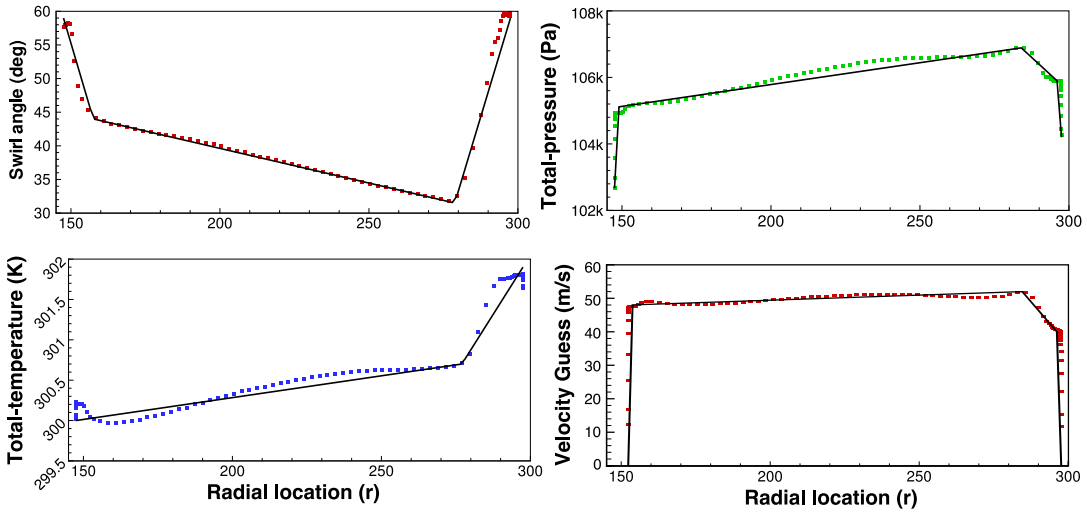


Figure 4.4: Piecewise polynomials fitted for experimentally determined inlet profile (indicated inlet parameters)

total-pressure, total-temperature, and initial guess velocity. A sample piecewise polynomial fitted for an experimentally obtained inlet profile is shown in fig. 4.3.1.

4.3.2 Specified Exit Back Pressure

The ghost state $\mathbf{U}_b$ is extrapolated from the interior and modified using a user-defined back-pressure p_{back}. The Riemann solver is applied at the boundary with the left state from the interior ($\mathbf{U}_i$) and the right ghost state ($\mathbf{U}_b$) to obtain the boundary flux. The characteristic Riemann invariants $\mathbf{R}^{+/-}$ at the boundary are shown schematically in fig. 4.3.

$$\mathbf{U}_b = \mathbf{U}_i + \begin{bmatrix} 0 \\ \vec{\mathbf{0}} \\ -p_i + p_{back} \end{bmatrix} \tag{4.15}$$

4.3.3 Rotating and Stationary No-slip Viscous Wall

The viscous wall nodes require an additional data structure, namely the near wall node. It is defined as the node within the first order stencil of the wall node, which is nearest to the wall (see fig. 4.3). The wall velocity is set to zero value and the pressure is

extrapolated from the near wall node. The density is calculated based on the formula shown in eq. (4.16) assuming an adiabatic wall condition $\partial T/\partial \mathbf{n} = 0$ (from ref. [106]).

$$\mathbf{U}_i^{static} = \begin{bmatrix} \rho_{i_1} p_i / p_{i_1} \\ \vec{\mathbf{0}} \\ p_{i_1} \end{bmatrix} \qquad \mathbf{U}_i^{rot} = \begin{bmatrix} \rho_{i_1} p_i / p_{i_1} \\ \vec{\omega} \times \mathbf{r} \\ p_{i_1} \end{bmatrix} \tag{4.16}$$

Moreover, for the rotating wall a non-zero velocity equal to $\vec{\omega} \times \mathbf{r}$ is specified for the wall node and the work term W_d shown below,

$$W_d = [(\vec{\omega} \times \mathbf{r}) \cdot \overline{\tau}] \cdot \mathcal{B}. \tag{4.17}$$

It is the work done to the fluid by moving walls and should be added to the wall node energy residual.

4.4 Explicit Time Marching with Implicit Preconditioning

4.4.1 Preconditioned Runge-Kutta Time Marching Scheme

The method of lines is used to discretise the temporal term. Once the spatial residual is obtained, a Newton-Krylov preconditioned Runge-Kutta explicit time integration of order 5 (RK5) is used to march in pseudo time as shown in eq. (4.18). The coefficients α_i^{rk} of RK5 are obtained from ref. [100]. The pseudo-time step $\delta\tau$ is obtained using a local time stepping procedure based on the Courant–Friedrichs–Lewy (CFL) number and maximum wave speed obtained based on the maximum eigenvalues of convective and viscous term as shown in eq. (4.21). A value of $CFL_{rk} = 2.5$ for first order and $CFL_{rk} = 1.15$ for second order spatial discretisation is fixed based on the stability

analysis of RK5 scheme [100].

$$\mathbf{P}\delta\mathbf{U}^{(i)} = -\alpha_i^{rk}\frac{\delta\tau}{\mathcal{V}}R[\mathbf{U}^{(i-1)}], \quad i = 1,\ldots,4 \tag{4.18}$$

$$\delta\mathbf{U}^{i} = \mathbf{U}^{(i)} - \mathbf{U}^{(i-1)}, \quad \mathbf{U}^{(0)} = \mathbf{U}^{n}$$

$$\delta\tau = CFL\frac{\mathcal{V}_i}{(\Lambda_c + 4\Lambda_v)_i}, \tag{4.19}$$

$$\text{where,} \quad (\Lambda_c)_i = \sum_j \left(|\mathbf{u}_{ij}\cdot\mathbf{n}_{ij} + c_{ij}|\right)\mathcal{S}_{ij} \tag{4.20}$$

$$(\Lambda_v)_i = \frac{1}{\mathcal{V}_i}\sum_j \left[\max\left(\frac{4}{3\rho_{ij}}, \frac{\gamma_{ij}}{\rho_{ij}}\right), \left(\frac{\mu_L}{Pr_L} + \frac{\mu_T}{Pr_T}\right)\right](\mathcal{S}_{ij})^2 \tag{4.21}$$

The time integration scheme is based on the JT-KIRK scheme of Xu et. al. [33] without the Multi-Grid (MG) acceleration. Since parallel MG is not implemented as part of this thesis it is not explained here. $\mathbf{P}$ in eq. (4.18) is the Newton-Krylov (NK) preconditioner for the flow equation, which is explained in the next section. A diagonal Jacobi preconditioner is used for the turbulence equation. Note that in STAMPS the turbulence equation is solved decoupled from the flow equations similar to the turbomachinery CFD solver Hydra [107].

4.4.2 Preconditioner based on Newton-Krylov Method

The implicit NK preconditioner is derived using the semi-discrete form of the governing equation shown in eq. (4.22). Note that the pseudo-time τ^* is different from the pseudo-time in the RK5 time integration. Similarly, the CFL number (CFL_{pre}) is defined separately for the implicit preconditioner and the local time step τ^* is defined using eq. (4.21).

$$\frac{\partial\mathbf{U}}{\partial\tau^*} = -R[\mathbf{U}] \tag{4.22}$$

The temporal term is discretised using backward Euler time integration. Linearising the residual (see eq. (4.23)), and rearranging one obtains the preconditioner in eq. (4.24).

$$\frac{\delta \mathbf{U}^n}{\delta \tau^*} = -R[\mathbf{U}^{n+1}] \approx -R[\mathbf{U}^n] + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \delta \mathbf{U}^n \tag{4.23}$$

$$\mathbf{P}\delta \mathbf{U}^n \equiv \left(\frac{\mathbf{I}}{\delta \tau^*} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \right) \delta \mathbf{U}^n = -R[\mathbf{U}^n] \tag{4.24}$$

The Jacobian matrix $\frac{\partial \mathbf{R}}{\partial \mathbf{U}}$ is obtained using the first order approximation for the convective fluxes and the gradients in the viscous fluxes are frozen to reduce the sparsity of the matrix to the first order stencil. This formulation is similar to the First-Order Jacobian Krylov Implicit (FOKI) scheme of Dwight [108], inspired from Cantariti et al. [109]. The present method closely follows the Jacobian assembly procedure in refs. [107, 108]. The Jacobian matrix is constructed by manually assembling the block matrix entry of every edge $\left(\frac{\partial \mathbf{R}_i}{\partial \mathbf{U}_j} \right)$ into a sparse block matrix by looping over all edges. The block matrix entry of the edges are obtained using algorithmic differentiation of the Roe approximate Riemann function in vector forward mode using Tapenade [92]. The residual on the right hand side is the full second order one without any approximations. The first-order Jacobian approximation is a standard practice in Newton-Krylov methods to reduce computational and memory cost [110].

4.4.3 Linearised System Solution using GMRES

The linear system in eq. (4.24) is solved using the left preconditioned restarted generalised minimal residual method (GMRES) method [111] described in *alg.* (4.1) on the next page. The left preconditioner $\mathbf{M}$ is an incomplete lower-upper factorisation of the first-order approximate Jacobian matrix (described in sec. 4.4.2) using zero fill-in (ILU0) [111]. The ILU0 matrix $\mathbf{M}$ is used to precondition the implicit update as shown in eq. (4.25). Since computing the ILU0 factorisation every iteration is computationally expensive, the preconditioner is lagged, similar to Jacobian lagging [112, 113], for every few (user defined) iterations.

$$\mathbf{M}^{-1}\mathbf{P}\delta \mathbf{U} = -\mathbf{M}^{-1} R[\mathbf{U}^n] \tag{4.25}$$

The three main numerical kernels in the GMRES algorithm are the (i) `ilu0` (ILU0 factorisation), (ii) `SpMV` (Sparse Matrix vector product), and (iii) `dot` (dot product) operators. The complete GMRES algorithm can be parallelised by simply parallelising the kernels (i)-(iii). The parallel GMRES implementation and the rank local ILU0 preconditioner is described in the next section.

input : Krylov dimension m, convergence tolerance ϵ, maximum restarts n
output: Update $\delta\mathbf{U}_m$

1 $\mathbf{M} \leftarrow \texttt{ilu0}\left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)$;
2 $k \leftarrow 0, \quad \delta\mathbf{U}_k \leftarrow [0]$;
3 $\mathbf{r}_0 \leftarrow \mathbf{M}^{-1}\left[R[\mathbf{U}] - \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)\delta\mathbf{U}_0\right]$;
4 $\beta \leftarrow \sqrt{\texttt{dot}(\mathbf{r}_0, \mathbf{r}_0)}$;
5 $\mathbf{v}_1 \leftarrow \mathbf{r}_0/\beta$;
6 **for** $j \leftarrow 1 \ldots m$ **do**
7 $\quad \mathbf{w} \leftarrow \texttt{SpMV}\left(\mathbf{M}^{-1}, \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right) \equiv \mathbf{M}^{-1}\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\mathbf{v}_j$;
8 $\quad$ **for** $j \leftarrow 1 \ldots m$ **do**
9 $\quad\quad h_{i,j} \leftarrow \texttt{dot}(\mathbf{w}, \mathbf{r}_i)$;
10 $\quad\quad \mathbf{w} \leftarrow \mathbf{w} - h_{i,j}\mathbf{v}_i$;
11 $\quad$ **end**
12 $\quad h_{j+1,j} \leftarrow ||\mathbf{w}||_2$;
13 $\quad \mathbf{v}_{j+1} \leftarrow \mathbf{w}/h_{j+1,j}$;
14 **end**
15 $\mathbf{V}_m \leftarrow [\mathbf{v}_1, \ldots, \mathbf{v}_m]$;
16 $\overline{\mathbf{H}}_m = \{h_{i,j}\}_{1 \le i \le j+1, 1 \le j \le m}$;
17 $\mathbf{y}_m \leftarrow \text{argmin}_{\mathbf{y}}||\beta\mathbf{e}_1 - \overline{\mathbf{H}}_m\mathbf{y}||_2$ (Least-squares minimisation) ;
18 $\gamma = e_{1,m+1}$ (error due to minimisation);
19 $\delta\mathbf{U}_m = \delta\mathbf{U}_0 + \mathbf{V}_m\mathbf{y}_m$;
20 **if** $\gamma > eps$ **then**
21 $\quad k = k + 1$;
22 $\quad$ **if** $k > n$ **then**
23 $\quad\quad$ **exit**;
24 $\quad$ **end**
25 $\quad$ **goto** 3;
26 **end**

Algorithm 4.1: ILU0 preconditioned restarted GMRES (reduction operations are shown in oval box and halo node exchange in matvec is shown in double box)

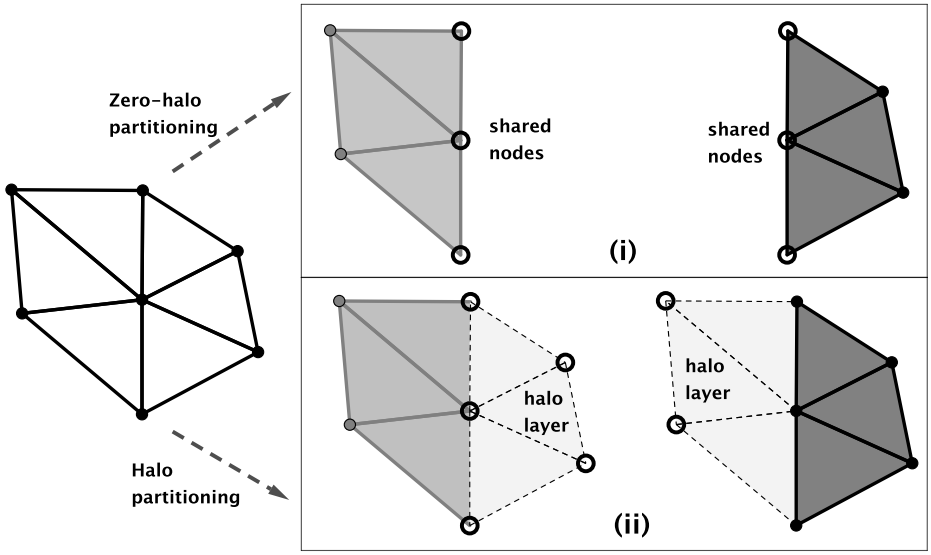


Figure 4.5: The (i) Zero-halo and (ii) one-layer halo based partitioning methods illustrated for a sample mesh split into two parts (grey and black); halo/shared nodes denoted by hollow circles and internal nodes denoted by filled circles

4.5 Parallel Algorithms

Partitioning methods employed for distributed parallel implementation of CFD codes can be broadly classified into two categories, namely partitioning with (i) zero-halo layer or with (ii) single or more than one halo layer. An illustration of the two strategies for partitioning is shown in fig. 4.5. Gicquel et al. [114] found that using a zero-halo layer (ZHL) approach one can completely eliminate redundant computations due to halo nodes and achieved good scalability. But adjoint seeding of ZHL fixed-point iteration was shown to be non-trivial by Mohanamuraly et al. [115] and requires manual seeding and differentiation of MPI calls. As a result this precludes the automation of the adjoint code generation using AD tools. Therefore, the zero-halo approach was abandoned in favour of the halo layer approach.

In this thesis, a two-halo layer (THL) approach is adopted instead of the more common single-halo layer (SHL). THL approach reduces overall communication and simplifies the algorithmic differentiation (AD) of the parallel fixed-point loop for the time marching adjoint solver compared to a SHL. Construction of the THL and demonstration of its

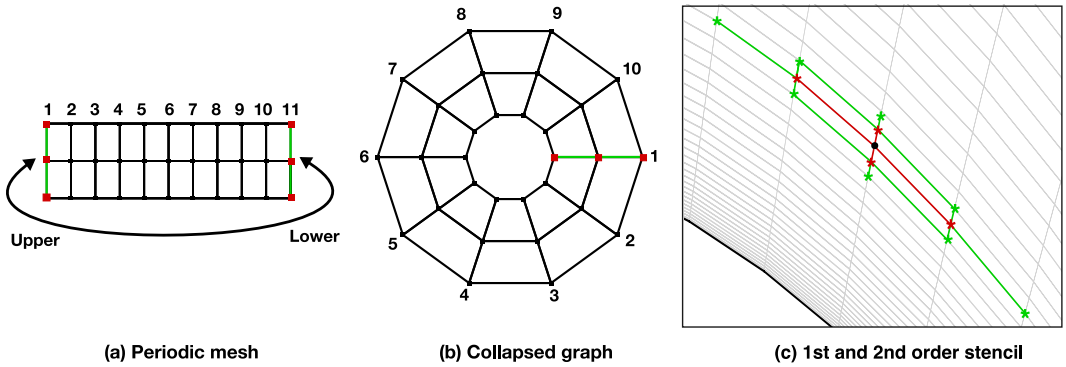


Figure 4.6: (a) Rectangular mesh with upper and lower periodic boundary, (b) its collapsed graph for partitioning and (c) first (red) and second (green) order stencil edges and nodes

aforesaid favourable properties are shown in the next section.

4.5.1 Two-Level Halo Partitioning

The METIS graph partitioning library [116] is used to partition the dependency graph of the first order computational stencil obtained from the dual mesh. For meshes with periodicity, vertices of the graph corresponding to the the upper and lower periodic nodes are collapsed and merged into a single graph vertex as shown in fig. 4.6(a-b). This technique ensures that periodic pairs are available in a local rank and no MPI communication is necessary. To ensure optimal load balancing the graph is assigned vertex weight equal to the total number of connected vertices in the stencil.

The THL partitioning is constructed such that the first and second order computational stencil shown in fig. 4.6(c) is completely available on a single partition rank for all nodes owned by the local partition. The typical computation and communication cycle in a fixed-point iterative flow solver is shown in fig 4.7 for SHL and THL. In THL approach one communicates smaller amounts of data less frequently compared to SHL resulting in *larger communication to computation ratio* between two update steps. Therefore, the communication in THL can be overlapped with computations more effectively compared to a SHL approach.

In fig. 4.8, the average communication volume induced by the THL approach is compared

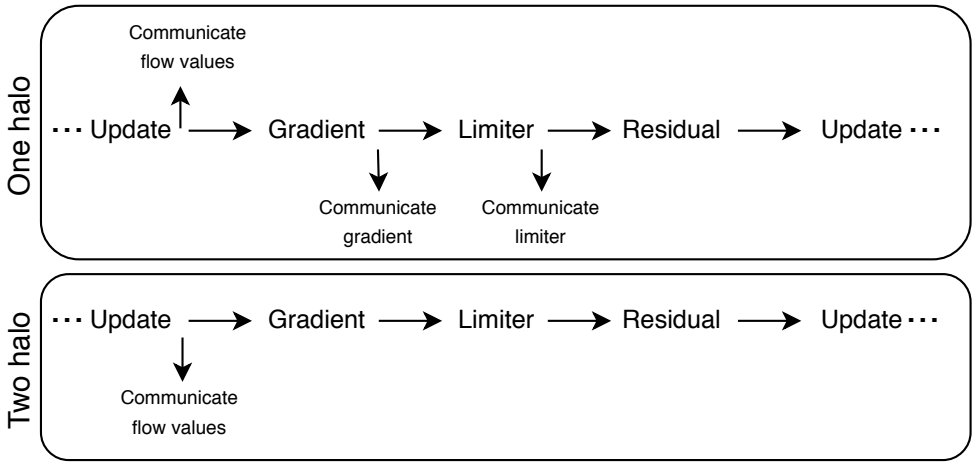


Figure 4.7: Communication and computation schedule in a typical fixed-point flow solver (one- and two-halo partitions)

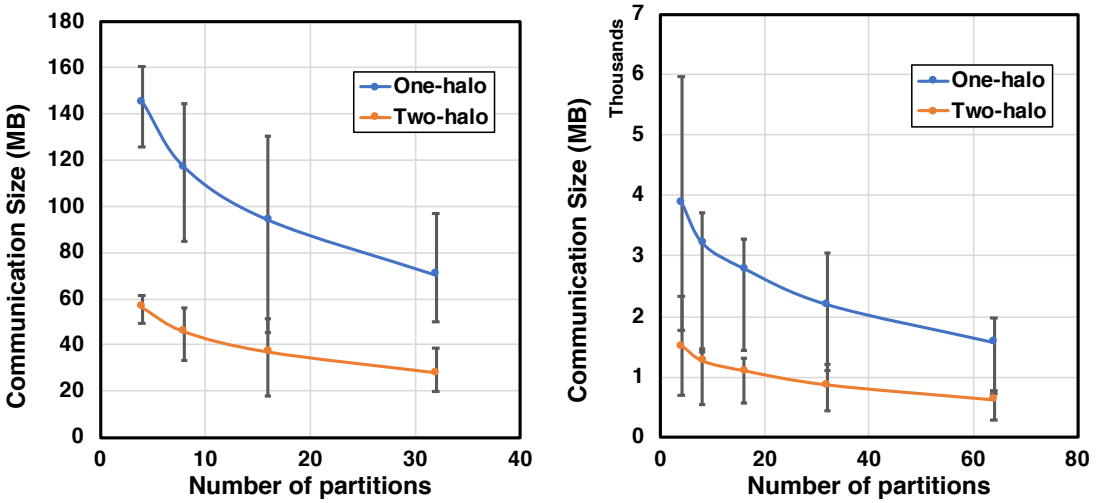


Figure 4.8: Comparison of average total communication volume (in MB) between the one-halo and two-halo partitioning for the (a) 2-D cascade (left) and (b) 3-D Turbocharger compressor fine mesh (right) for different partition sizes (error bars indicate the maximum and minimum deviation from average value).

against the SHL for a single update step of a 3-D Turbocharger compressor (see sec. 4.7.5) and 2-D cascade (see sec. 4.7.4). *THL approach reduces the total communication volume (compared to SHL) by half and the deviation in communication size between ranks is much lower.*

The reduction in communication comes at the cost of more redundant computations. Note that only the gradient calculation requires the THL data (second order stencil) and

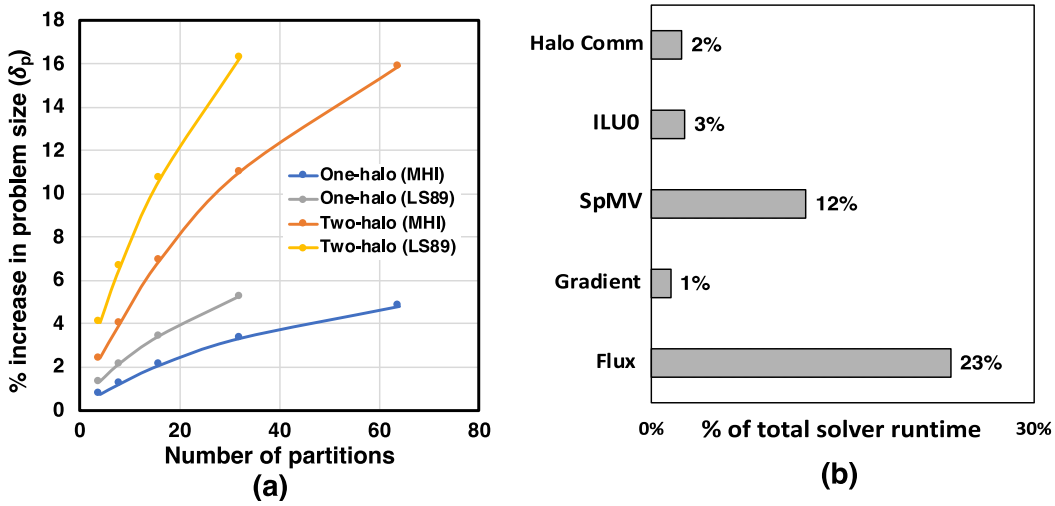


Figure 4.9: (a) Increase in problem size due to redundant computations in one- and two-halo partitioning and (b) profile of subroutines with high wall clock timing of the 3-D turbocharger compressor mesh 64 partitions.

all other computations such as viscous and inviscid flux calculations require values located at the SHL. In fig. 4.9(b) the relative timing for various subroutines in STAMPS is shown. The timings were obtained by running 10 update steps of the implicit flow solver on the 3-D Turbocharger mesh (64 partitions). The results show that the gradient calculation constitutes only 1% of the runtime (much smaller than the halo communication!). *Any additional increase in redundant computation due to THL is only going to affect the runtime of the low overhead gradient subroutine.*

The increase in the problem size δ_p (due to redundant edges and halo nodes) is plotted in fig. 4.9(a). The increase in problem size (δ_p) due to redundant computations can be estimated using the total number of edges e_i in a partition i, the total number of edges in the unpartitioned mesh e_p and total number of partitioned ranks N_{rank} using the expression:

$$\delta_p = \left(\sum_{i=1}^{N_{rank}} e_i \right) / e_p \quad \% \tag{4.26}$$

δ_p for THL approach is almost thrice that of SHL. But this additional cost is paid only for the gradient subroutine, which constitutes 1% of the total run-time of the solver.

In the THL approach (see fig. 4.7), only one communication call appears after update step and no communication is necessary between gradient, limiter, and flux residual evaluation. Christianson [117] showed that for a non-linear primal fixed-point iteration the adjoint gradients can be computed without storing intermediate stages of the primal loop. To have an efficient adjoint implementation in STAMPS the proof of Christianson is used to manually assemble the adjoint FPI which follows closely the work of Giering [118]. Note that the adjoint FPI is a one time effort and does not affect the automation of the adjoint code generation.

From an AD perspective the residual evaluation in a fixed point iteration (FPI) for the serial and parallel code remains the same in the THL approach since no MPI calls appear inside the residual evaluation. Only a single MPI call after the update step is necessary (see fig. 4.7). The reversal of MPI calls for the AD [119] are manually inserted in the FPI of the time-marching adjoint solver. This greatly simplifies the AD process by eliminating the need to differentiate MPI calls, which is not a trivial task for manually assembled FPI [115]. Appendix C outlines the adjoint differentiation of the two-halo parallel fixed-point iteration in STAMPS.

4.5.2 Reordering and Scalable Assumed Partitioning

The graph partitioner returns a non-contiguous partitioning of the rows of the Jacobian matrix. To accommodate the use of Algebraic Multi-Grid (AMG) preconditioners using parallel libraries such as Lis [120] and Hypre [121] (as future extension in STAMPS) one requires a contiguous partitioning of the rows of the Jacobian matrix (see fig. 4.11). Therefore, post-partitioning the global mesh is permuted to yield a contiguous partitioning of the Jacobian rows. This lends itself to the scalable assumed partitioning algorithm described in ref. [122].

A schematic of actual and assumed partitions of a (row) vector partitioned on five ranks is shown in fig. 4.5.2. In assumed partitioning one starts with a contiguous partitioning of rows and each rank only knows the range of contiguous rows it owns. In addition

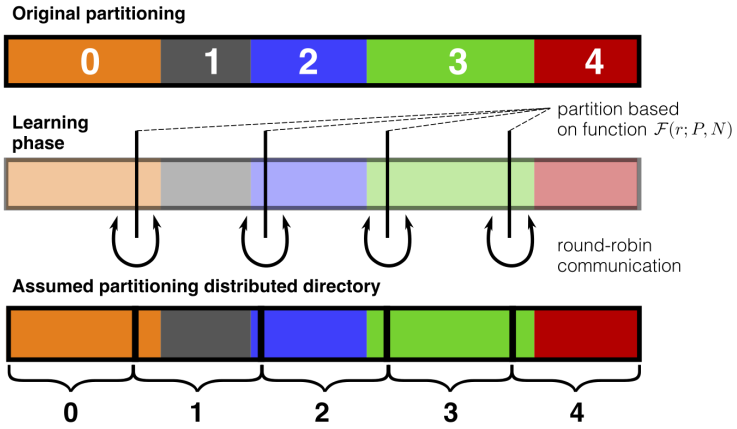


Figure 4.10: Actual and assumed partitioning of a vector (on five ranks); the arrow indicates the round-robin communication during the learning phase.

another contiguous partitioning of the rows is assumed based on a partition function $\mathcal{F}$ whose inverse $\mathcal{F}^{\dagger}$ is also known. A partition function $\mathcal{F}$ returns the partition rank (p) (assumed) given the row index (r). The inverse function $\mathcal{F}^{\dagger}$ returns the row index range (say $r \in [r_1, r_2]$) given a partition rank (p). For example, $\mathcal{F}$ can be defined using eq. (4.27), where P, N are the total number of partition ranks and rows, and $\lfloor \cdot \rfloor$ is the integer floor operator.

$$p = \mathcal{F}(r; P, N) = \lfloor (r \times P)/N \rfloor \tag{4.27}$$

Then each processor communicates in a round-robin fashion to learn the missing row information from the neighbouring ranks as shown in fig. 4.5.2. This assumed partitioning distributed directory (see fig. 4.5.2) can be used for scalable range queries on the Jacobian rows [122] and construct inter-processor communication schedule.

Local reordering of nodes has become a standard practice to improve performance of unstructured codes [123, 124, 125]. Ordering algorithms based on Cuthill-McKee/ Reverse Cuthill-McKee (RCM) and Space-filling curve (SFC) give good bandwidth reduction with low computational cost [125]. Local re-ordering of the nodes using Reverse Cuthill-McKee (RCM) algorithm is adopted in this work to improve memory access because the computational graph constructed for METIS partitioning can be reused for the RCM ordering. In fig. 4.11, the reduction in bandwidth achieved using the local

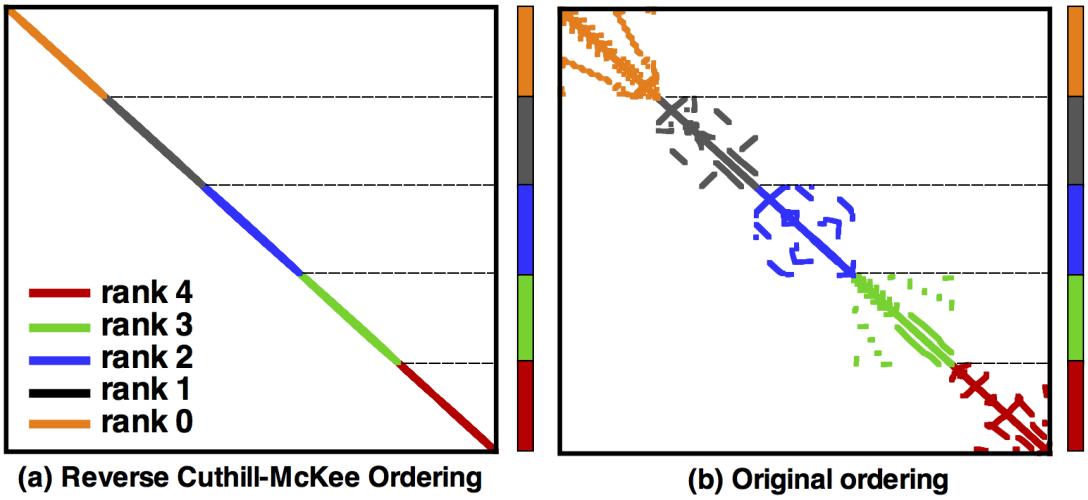


Figure 4.11: Bandwidth reduction achieved using (a) Reverse Cuthill-McKee (RCM) reordering of the local partition vertices vs. (b) the original ordering of the mesh nodes (note the contiguous row partitioning due to global permutation).

RCM is shown for the 2-D VKI LS89 cascade hexahedral mesh (note the contiguous partition of the rows).

4.5.3 Parallel GMRES with Local ILU(0) Preconditioner

Many parallel GMRES algorithms are available in literature for example, the s-step method [126], pipelined method [127] and hybrid derivatives thereof [128], and using different strategies for orthogonalisation such as iterative classical Gram-Schmidt (ICG) [129, 130]. Recently, the communication avoiding GMRES (CA-GMRES) algorithms have gained popularity due to its high scalability [126]. Grigori [131] recently proposed a communication avoiding version of ILU0 (CA-ILU0) preconditioner for use in a CA-GMRES framework. The CA-ILU0 when combined with the CA-GMRES shows great promise for a scalable implementation. Vannieuwenhoven[132], proposed an Incomplete Multi-Frontal (IMF) method, which uses the block elemental structure of the Jacobian matrix constructed element-by-element (or edge-by-edge in the present context). Since IMF retains the block nature of the Jacobian it can achieve better speed up by avoiding the Jacobian assembly into a sparse matrix format.

Sophisticated algorithms discussed above are postponed to a future study in STAMPS.

Operation	Serial	Parallel
dot product	$d = \texttt{dot}(\mathbf{x}, \mathbf{y})$	$d = \texttt{dot}(\mathbf{x}, \mathbf{y})$; $\texttt{allreduce}(d, +)$
matrix-vector product	$\mathbf{y} = \texttt{SpMV}(\mathbf{A}, \mathbf{x})$	$\mathbf{y} = \texttt{SpMV}(\mathbf{A}, \mathbf{x})$; $\texttt{halo\_comm}(\mathbf{y})$
ILU0 preconditioner	$\mathbf{M} = \texttt{ilu0}(\mathbf{A})$	$\mathbf{A}^{\dagger} = \texttt{nullify\_halo\_rc}(\mathbf{A})$; $\mathbf{M} = \texttt{ilu0}(\mathbf{A}^{\dagger})$

Table 4-A: Blocking parallel transformation of GMRES kernels

Only a simple blocking communication GMRES algorithm is adopted in this work. The transformation of the main numerical kernels of the GMRES algorithm are tabulated in tbl. 4-A. The $\texttt{allreduce}(\cdot, +)$ is the MPI all-reduce summation operation and `halo_comm` is the halo data exchange routine (implemented using MPI send/receive) in tbl. 4-A. The parallel ILU0 preconditioner is based on a rank local Jacobian matrix. Therefore, all halo row and column off-diagonal entries of the Jacobian are zeroed and a unit diagonal entry is enforced before the ILU0 factorisation step (using the `nullify_halo_rc` routine as shown in tbl. 4-A). For small partition sizes a local preconditioner can adversely affect solver convergence. But for large partition sizes $\mathcal{O}(10^4)$ Xu et al. [33] found no degradation in the convergence. Moreover, the local ILU0 does not require any parallel communication making it quite scalable. The performance of the ILU0 preconditioner is presented in sec. 4.7.3.

4.5.4 Implicit Solver Scalability Results

The parallel efficiency of the non-linear solver for a 4M hexahedral mesh computation of a centrifugal turbocharger compressor is shown in fig. 4.12 for varying number of decompositions. For details on the geometry and flow conditions the reader is referred to sec. 4.7.5. The average partition size is also indicated in the figure using bars. Each compute node in the cluster facility had 24 processor cores and as many partitions as the number of cores was created for the test. The full scalability run covered 4 compute nodes (or 96 processor cores) of the cluster.

Up to 2 nodes the efficiency does not deteriorate significantly. For the 4 node (96 processor cores) case the efficiency reduces to 50%. The all-reduce operation in the parallel dot-product of the GMRES is a strong parallel performance bottleneck [127]. Even small

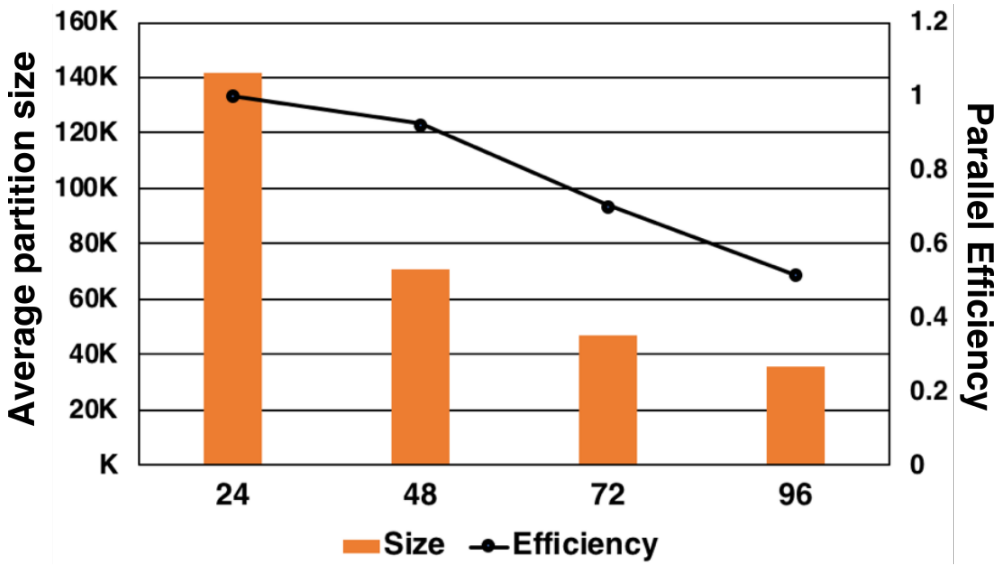


Figure 4.12: Non-linear implicit flow solver scaling for the turbocharger compressor test case (average partition size indicated by the bar chart).

imbalances within the ranks can be exaggerated due to the all-reduce operation. The focus of this thesis has been the initial parallel implement and correctness of the results from the parallel solver. Some possible ways to mitigate this problems were already discussed as future developments to the solver in sec. 4.5.3.

4.6 Linearised Solvers and their Implementation

4.6.1 Discrete Adjoint Solver via Reverse Algorithmic Differentiation

As explained in sec. 2.1.7 adjoint method provides an efficient way to compute the sensitivity $\frac{dJ}{d\alpha}$ of QoI J with respect to a set of input parameters α. The QoI J considered in the present work are (i) total-pressure loss [133] (J_{ploss}), (ii) exit mass flow rate (J_{mass}), and (iii) exit whirl angle (J_{whirl}) defined in eq. (4.28). The bar terms indicate mass-flow averaged quantities over a given boundary surface. The surface integrals are evaluated on the discrete boundary patches using a mid-point quadrature rule as shown

in eq. (4.29)-(4.30).

$$J_{ploss} = \frac{\bar{p}_0^{\,in} - \bar{p}_0^{\,out}}{\bar{p}_0^{\,out} - \bar{p}^{\,out}}, \qquad J_{mass} = \dot{m}_{out}, \qquad J_{whirl} = \bar{\alpha}_{out} \tag{4.28}$$

$$\bar{\Phi} = \int \Phi\rho(\mathbf{u} \cdot d\mathcal{S}) = \sum_i \Phi_i \rho_i (\mathbf{u}_i \cdot \mathcal{B}_i) \tag{4.29}$$

$$\dot{m} = \int \rho(\mathbf{u} \cdot d\mathcal{S}) = \sum_i \rho_i (\mathbf{u}_i \cdot \mathcal{B}_i) \tag{4.30}$$

The superscript *in* and *out* define the mass-flow averaged values over the inlet and outlet patches respectively. The whirl angle α is defined as (i) $\alpha = \arctan(u_y, u_x)$ for 2-D and (ii) $\alpha = \arctan(||\mathbf{u} - (\mathbf{u} \cdot \mathbf{n})\mathbf{n}||, \mathbf{u} \cdot \mathbf{n})$ for 3-D test cases ($\mathbf{n}$ is the boundary normal). Once the cost-functions are defined the adjoint equation can be derived for the RANS equation using eq (2.20) below:

$$\left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^T v = \left(\frac{\partial J}{\partial \mathbf{U}}\right)^T \tag{4.31}$$

In eq. (4.31), v is the adjoint variable and J is the QoI that depends on the state $\mathbf{U}$. The pseudo-time form to yield the time-marching adjoint is shown below:

$$\frac{\partial v}{\partial \tau^*} + \underbrace{\left[\left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^T v - \left(\frac{\partial J}{\partial \mathbf{U}}\right)^T\right]}_{\overline{R}[v,\mathbf{U}]} = 0 \tag{4.32}$$

The adjoint residual $\overline{R}[v, \mathbf{U}]$ (in eq. (4.32)) is obtained using the Algorithmic Differentiation (AD) of the discrete residual subroutine $R[\mathbf{U}]$ of the primal flow solver. Then the fixed-point iteration (FPI) implicit time marching loop is manually assembled using this differentiated residual. The implicit preconditioner for the adjoint system can be derived using the backward Euler time discretisation and linearisation of the residual $\overline{R}[v, \mathbf{U}]$. The final form is shown in eq. (4.33). Similar to the primal flow solution the preconditioned RK5 scheme is used to solve the adjoint system. The implicit preconditioner for the adjoint is the transpose of the primal system [33]. The two level halo partitioning used in this work enables the Jacobian matrix transposition without incurring any par-

allel communication because all the necessary information is contained within the local processor. Therefore, computing $\overline{R}$ is the only extra effort involved in constructing the adjoint FPI since other components can be reused from the primal FPI.

$$\mathbf{P}^T \delta v^n \equiv \left[\frac{\mathbf{T}}{\Delta \tau^*} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \right]^T \delta v^n = -\overline{R}[v^n] \tag{4.33}$$

4.6.2 Tangent Linear Solver via Forward Algorithmic Differentiation

In this work, the tangent linear solver is obtained by the linearisation of the state $\mathbf{R}$ with respect to the mesh nodes $\mathbf{x}$ along a volumetric direction vector $\delta\mathbf{x}$ define as:

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} \frac{d\mathbf{U}}{d\mathbf{x}} \delta\mathbf{x} = \frac{\partial \mathbf{R}}{\partial \mathbf{x}} \delta\mathbf{x} \tag{4.34}$$

To understand the volumetric directional derivative one should consider the mesh deformation algorithm described in appendix B. A given surface perturbation vector, say the PCA mode $\mathbf{z}_j$ described in ch. 5, can be injected into the volumetric mesh, via the mesh deformation operator $\mathbf{A}$ using the relation,

$$\delta\mathbf{x} = \mathbf{A}\mathbf{z}_j. \tag{4.35}$$

The directional derivative $\frac{\partial(.)}{\partial \mathbf{x}} \mathbf{A}\delta\mathbf{x}$ of the PCA mode can be condensed into $\frac{\partial(.)}{\partial \mathbf{z}_j}$. Following a similar procedure to the adjoint discretisation one obtains the implicit pseudo-time marching tangent linear form as shown below:

$$\mathbf{P}\delta w^n \equiv \left(\frac{\mathbf{I}}{\delta \tau^*} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \right) \delta w^n = -R^{\dagger}[w^n] \tag{4.36}$$

where, $R^{\dagger}[w^n] = \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right) w^n - \frac{\partial \mathbf{R}}{\partial \mathbf{z}_j}$ and $w = \frac{d\mathbf{U}}{d\mathbf{z}_j}$. The left hand side term of the tangent linear FPI is exactly similar to the primal FPI. But the construction of the right hand side term requires application of forward-mode of AD to the residual subroutine. Application of forward and reverse mode of AD to obtain the tangent linear and adjoint terms are explained in appendix B. Note that the IMC 2 LF model (see sec. 3.2) requires the tangent

linear solution for every dominant PCA mode. IMC 3 is not used in this work since the Hessian computation has not been implemented in the current version of STAMPS.

4.6.3 Adjoint and Tangent Linear Sensitivity Assembly

The sensitivity of the QoI J to surface perturbations can be obtained using either the forward (see eq. (4.37)) or reverse mode (see eq. (4.38)) of AD. Note that the term $\frac{\partial \mathbf{R}}{\partial \mathbf{z}_j}$ in eq. (4.38) requires differentiation of the mesh metric terms. In STAMPS all mesh metrics are pre-calculated and stored during the pre-processing or mesh conversion step. The computational stencil (and edges) are constructed, partitioned, and distributed across processors. Only this pre-processed data-structure is exposed to the solver. Therefore, the sensitivity assembly is performed as a serial computation in two stages. Firstly, the volumetric sensitivity is obtained with respect to $\mathcal{S}$, $\mathcal{B}$ and $\mathcal{V}$. These derivatives are then propagated into the differentiated metric routines to yield the final volumetric derivative with respect to $\mathbf{x}$. As a final step, volume to surface sensitivity projection is performed using the mesh deformation adjoint (see appendix B).

$$\frac{dJ}{d\mathbf{z}_j} = \frac{\partial J}{\partial \mathbf{z}_j} + \frac{\partial J}{\partial \mathbf{U}} w \tag{4.37}$$

$$\left(\frac{dJ}{d\mathbf{z}_j}\right)^T = \left(\frac{\partial J}{\partial \mathbf{z}_j}\right)^T + \left(\frac{\partial \mathbf{R}}{\partial \mathbf{z}_j}\right)^T v \tag{4.38}$$

An important aspect of the sensitivity is the adjoint differentiation of the wall distance function used for the SA model. Brute force differentiation using an AD tool results in higher runtimes and memory consumption for this routine. Therefore, the routine is manually differentiated and the various branches of the "*Closest Point on Triangle to Point*" algorithm [95] are handled manually using enumerated integer arrays. In this approach push/pop statements in the adjoint code are avoided using simpler branching using arrays. Müller et al. [134] found that push/pop statements in the discrete adjoint code are detrimental to its performance and provide techniques to avoid them in the context of AD.

4.7 Non-linear Flow Solver Validation

The validation of the primal solver was carried out using two external and three internal flow test cases. Since the new developments in the thesis focus on internal flows, more test cases are shown in this category, namely (i) the Sajben diffuser with a strong shock condition, (ii) VKI LS89 Turbine cascade and (iii) Turbocharger compressor with a vaneless diffuser. In the external flow category, the standard flat plate benchmark test case (both laminar and turbulent solutions) are shown.

4.7.1 Laminar Flow Over Flat Plate with Zero-pressure Gradient

The flow over a flat plate at zero-pressure gradient was simulated for a free-stream Mach number of 0.2 under standard atmospheric conditions (1 atm and 273.15 K) using the STAMPS flow solver. The mesh and overall boundary conditions are shown in fig. 4.13 and the flow is assumed to be laminar. The dimension of the computational domain is $(x \times y) \rightarrow (2.333 \times 1.0)$. The inviscid wall constitutes the first $0 \leq x \leq 0.333$ lengths of the domain and the viscous wall of the flat plate extents to the next $0.333 \leq x \leq 2.333$ lengths. The dynamic viscosity was adjusted to maintain the Reynolds number based on plate length $Re_x \approx 2000$ (laminar regime). A probe surface is introduced at the middle of the viscous wall (shown by the yellow line in fig. 4.13) to measure flow quantities for comparison with the analytical solution of Blasius[135] shown below,

$$2ff'' + f''' = 0, \quad \text{where,} \quad \eta = y\sqrt{\frac{\mathbf{u}_\infty}{\nu x}}, \quad f'(\eta) = \frac{\mathbf{u}}{\mathbf{u}_\infty}$$
$$\text{BC: } f(0) = f'(0) = 0, \quad \text{and} \quad f'(\infty) = 1 \tag{4.39}$$

The friction coefficient computed using the STAMPS solution is compared against the Blasius results given in eq. (4.40). The numerical values for both η and C_f are found to be in good agreement with the analytical results as shown in fig. 4.14.

$$C_f = \frac{0.664}{\sqrt{Re_x}}, \quad Re_x = \frac{\mathbf{u}_\infty x}{\nu} \tag{4.40}$$

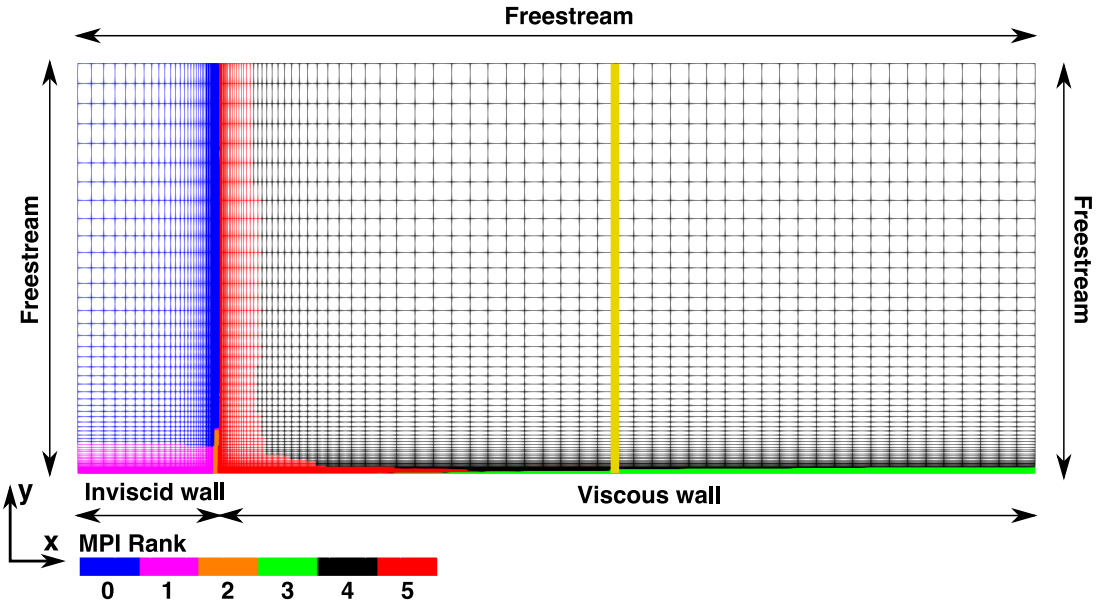


Figure 4.13: Flat plate mesh coloured by the partition rank (six ranks) with the boundary condition specification and the measurement probe surface is show in yellow (alternate mesh lines plotted for clarity).

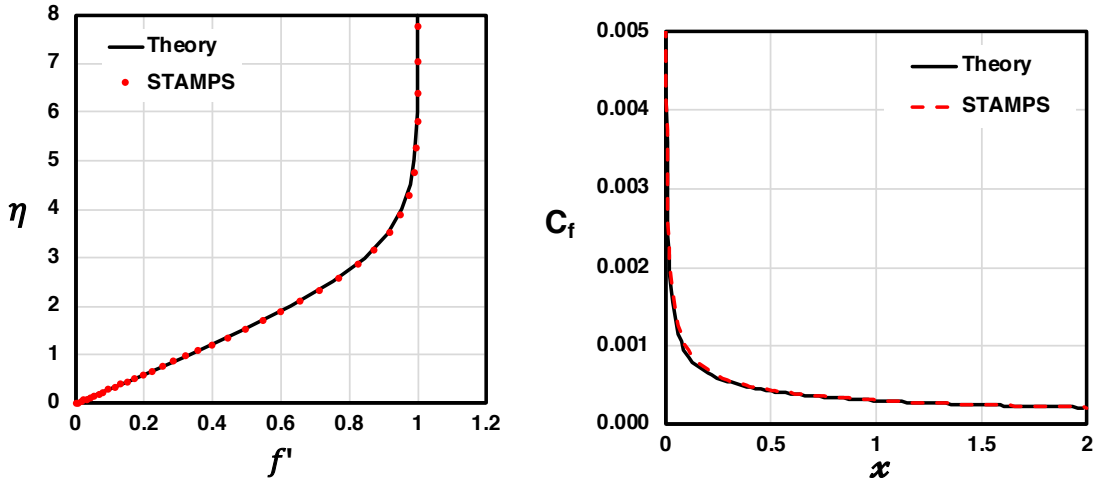


Figure 4.14: Laminar flow over a flat plate under zero pressure gradient (Blasius solution vs. STAMPS): (i) non-dimensional velocity η vs. f' (left) and (ii) friction coefficient C_f vs. wall location x

4.7.2 Turbulent Flow Over Flat Plate with Zero-pressure Gradient

The same setup from the previous laminar zero pressure gradient test case is used in this validation. But the viscosity is not adjusted and the working fluid is assumed to

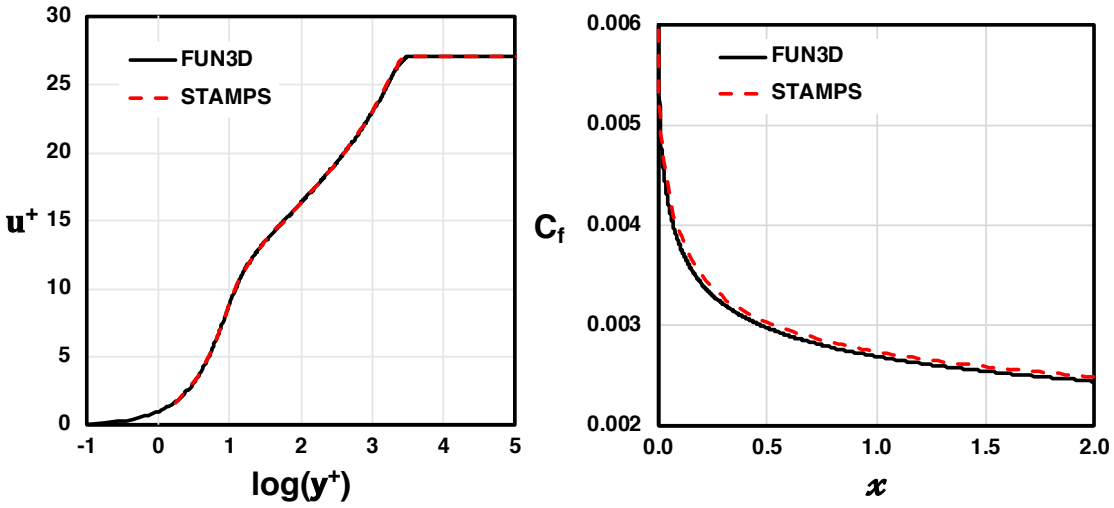


Figure 4.15: Turbulent flow over a flat plate under zero pressure gradient (NASA FUN3D vs. STAMPS): (i) non-dimensional velocity $\mathbf{u}^+$ vs. wall distance $\log(y^+)$ (left) and (ii) friction coefficient C_f vs. wall location x

be air obeying the Sutherland law for viscosity. The flow is assumed to be turbulent (solve the SA turbulence equation) with $Re_x \approx 9 \times 10^6$. The free-stream condition for the turbulent viscosity ν_T is set three times the free-stream laminar viscosity ν_∞ ($\nu_T = 3\nu_\infty$) and is set to zero-value at the viscous wall. The results from NASA FUN3D code provided in ref. [136] are used as reference solution to compare the results from STAMPS. The definitions for the quantities compared are shown in eq. (4.41)-(4.42) and the results are plotted in fig. 4.15.

$$\tau_{wall} = \mu_{wall}\left(\frac{\partial \mathbf{u}}{\partial y}\right)_{wall}, \quad C_f = \frac{\tau_{wall}}{\frac{1}{2}\rho_\infty \mathbf{u}_\infty^2} \tag{4.41}$$

$$\mathbf{u}^+ = \mathbf{u}\sqrt{\frac{\tau_{wall}}{\rho_{wall}}}, \quad y^+ = \frac{y}{\nu_{wall}}\sqrt{\frac{\tau_{wall}}{\rho_{wall}}} \tag{4.42}$$

Good agreement between the reference FUN3D results are obtained for both the non-dimensional velocity $\mathbf{u}^+$ and friction coefficient C_f. Both laminar and turbulent flat plate solutions were obtained after a overall residual convergence of 10^{-10}. No limit cycle oscillation or stalling of convergence was observed in either case.

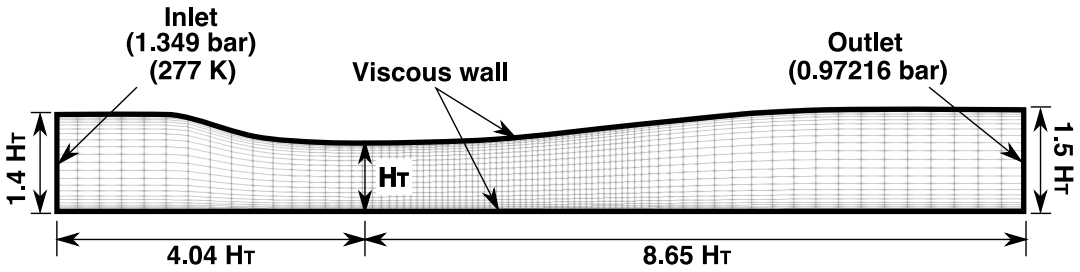


Figure 4.16: Sajben diffuser geometry with grid superimposed on top (alternate mesh line is plotted for clarity) : Throat height $H_T = 44.07$ mm.

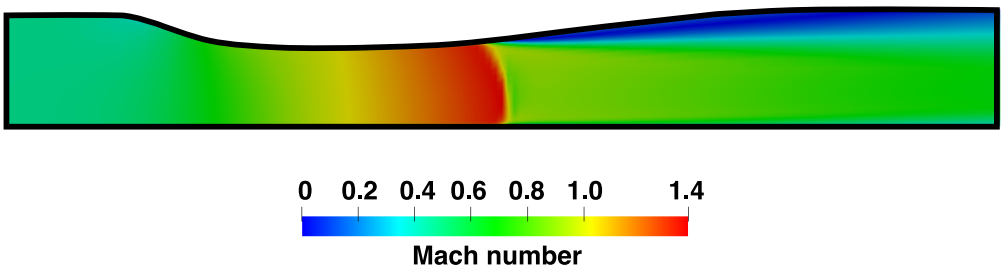


Figure 4.17: Sajben diffuser Mach number contour obtained using STAMPS: Shock aft of throat and the separated flow region are visible.

4.7.3 Sajben Transonic Diffuser

The Sajben transonic diffuser has been extensively studied experimentally by Sajben, Bogar, and co-workers. The results of the study have been published in a series of papers provided in ref. [137, 138, 139, 140]. In the so-called "*strong shock*" condition of the diffuser, a normal shock wave is present aft of the throat region and in its vicinity the solution is found to oscillate slightly. The cause for the small shock oscillation is the self-exited oscillation of this so-called *terminal shock* [141, 140]. Aft of the shock a region of subsonic separated flow exists (see fig. 4.17). This test case exhibits a rich variety of flow features, which makes it a good validation test case. The complete diffuser geometry is defined in fig. 4.16 in terms of throat height $H_T = 44.07$ mm. A maximum $y^+ \approx 0.7$ is used for the near wall grid spacing and a fully turbulent flow is assumed in the diffuser.

The convergence of the mass and turbulence residual is shown in fig. 4.18(i)-(ii) for a range of partition sizes. Note that 1000 explicit iterations were run to initialise the

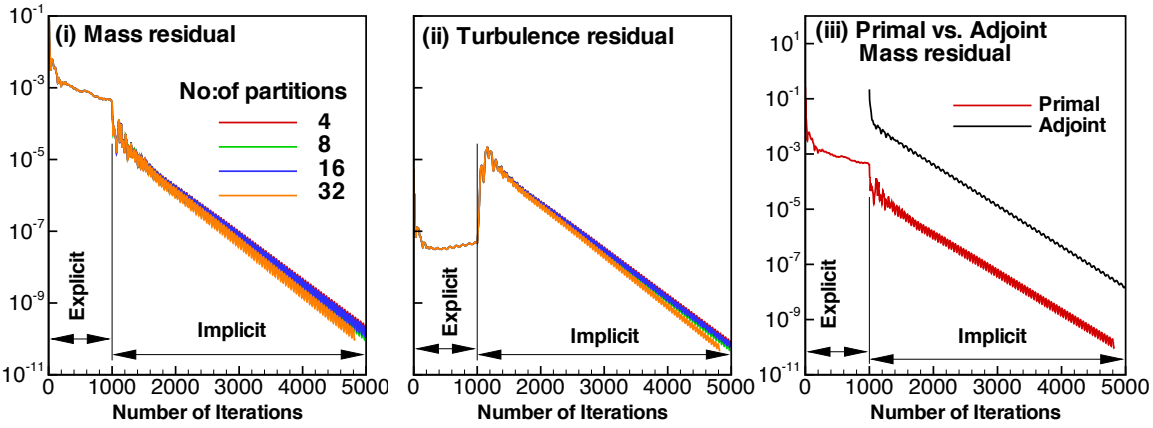


Figure 4.18: Convergence comparison of (i) primal solver mass residual and (ii) turbulence residual for indicated partition sizes and (ii) primal vs. adjoint mass convergence: CFL = 100

solution followed by hot-starting the implicit solver using a constant CFL value of 100. *The decoupled ILU0 preconditioner for the flow is quite robust and no significant deterioration in convergence was observed between the various partition sizes.* Moreover, the convergence to steady state was achieved for all partition sizes and the residuals reached machine zero without stalling.

The total-pressure loss (J_{ploss} in eq. (4.28)) adjoint convergence is plotted against the primal mass convergence of the diffuser in fig. 4.18(iii). The parallel implicit adjoint solver was started directly using a zero initialisation. Full convergence was achieved and the asymptotic rates between the primal and adjoint are found to be similar.

The surface pressure along the top and bottom walls of the diffuser are plotted against the experimental results from ref. [141] in fig. 4.19(i)-(ii). Good agreement is obtained for both top and bottom surfaces. The sectional velocity plot in fig. 4.20(i)-(iv) at stations $x/H_T = 2.882, 4.611, 6.43$ and $x/H_T = 7.493$ also show a good comparison with the experimental results. Georgiadis et al. [142] compared the solution of the strong shock case, computed using the NASA PARC code with five different turbulence models; comprising of algebraic and two-equation turbulence models. Similar under-prediction of the velocity in the separation zone (see fig. 4.20) and over-prediction of surface pressure

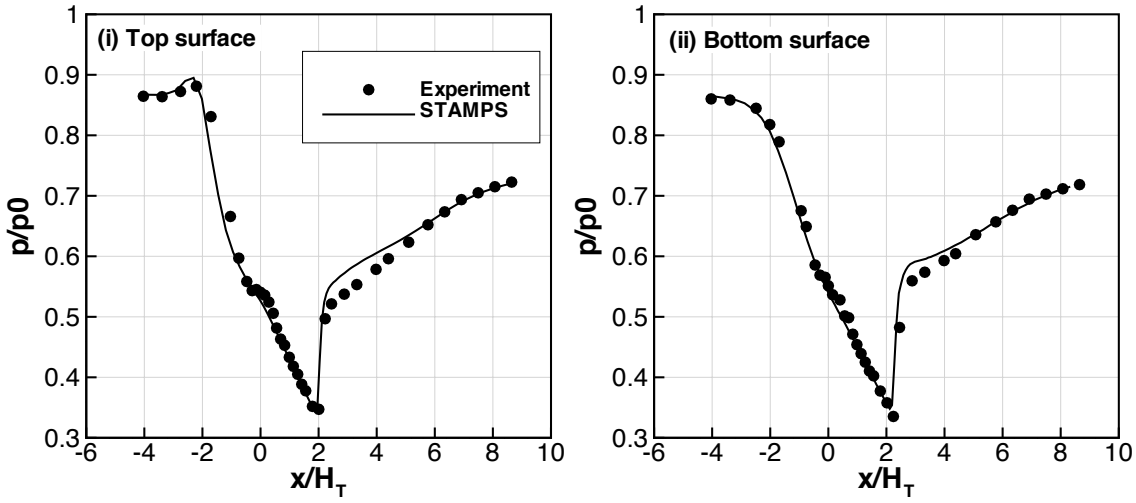


Figure 4.19: Surface pressure distribution comparison between STAMP and experimental results in ref. [141] (i) on the top and (ii) bottom surfaces of the Sajben diffuser

(see fig. 4.19) downstream of the diffuser was observed for all turbulence models.

4.7.4 VKI LS89 Turbine Cascade

The LS89 was originally designed and optimised at the Von Karman Institute for Fluid Dynamics (VKI) for a subsonic isentropic exit Mach number of 0.9 using an iterative inverse method that modifies a given blade profile such that it fits an input surface velocity distribution [143]. Experimental measurements consisting of surface pressure and wall heat transfer were conducted for a range of inlet total-pressure and exit back pressure ratios [144, 145]. The mesh and the computational domain used for the validation is shown in fig. 4.21.

The surface isentropic Mach number (M_{isen}) obtained using CFD is plotted against the experimental results of Arts [144] for the MUR43 to MUR48 test conditions in fig. 4.22, where M_{isen} is obtained from eq. (4.43), P_{01} is the total-pressure at the inlet and P_s is the static pressure at a measurement station.

Each condition MUR43 to MUR48 corresponds to a specific back pressure at the outlet and total temperature at the inlet, which is tabulate in tbl. 4-B. A constant total

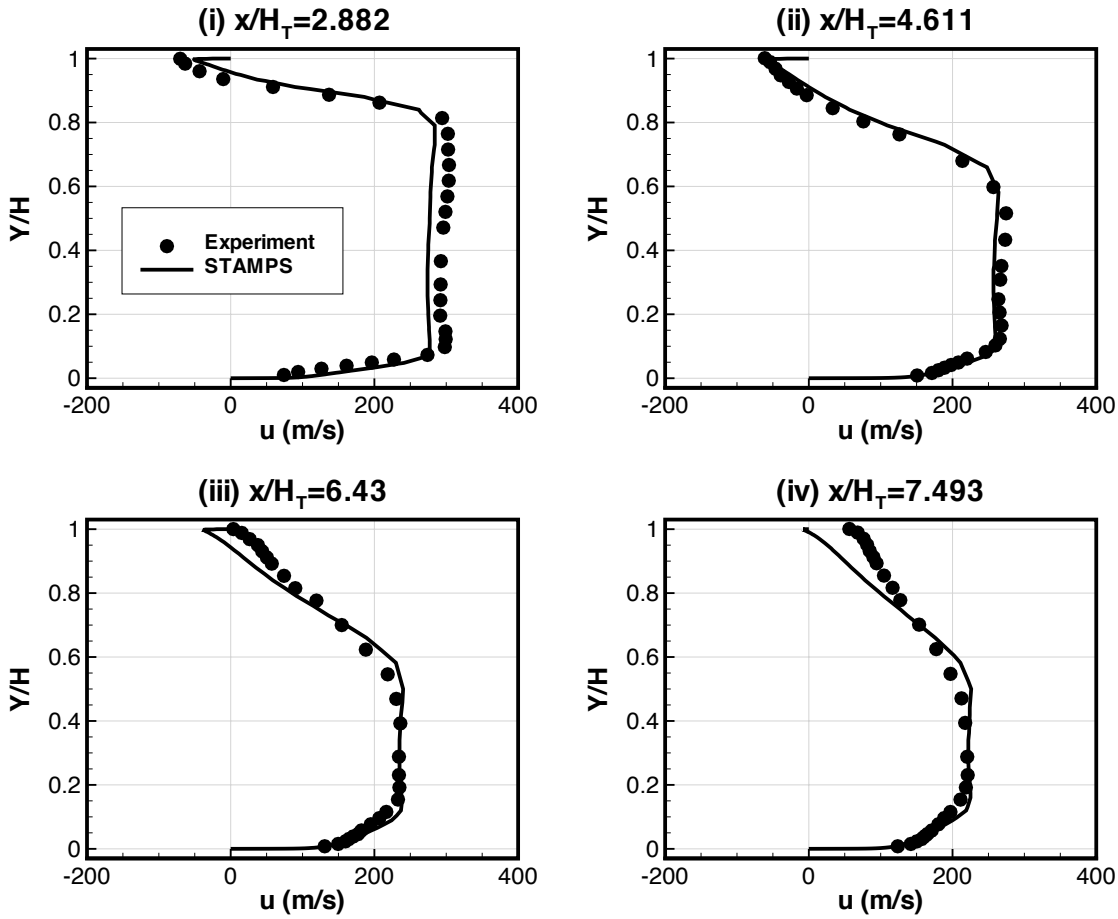


Figure 4.20: Sectional velocity comparison between experimental measurements [141] and STAMPS solution at indicated section.

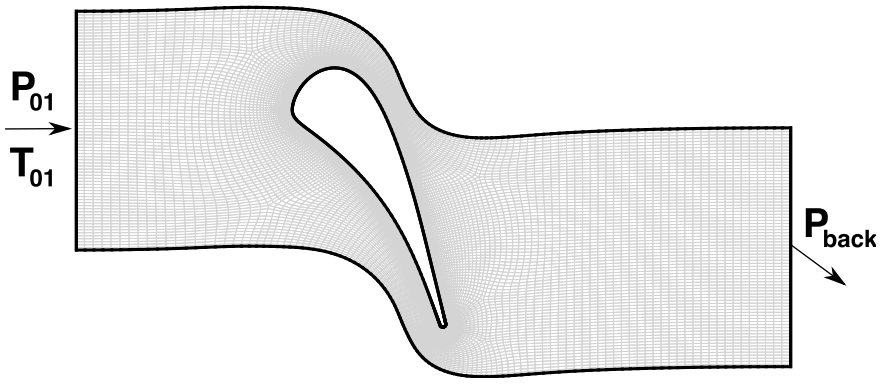


Figure 4.21: LS89 computational domain (every four mesh lines plotted for clarity)

temperature of 420 K was maintained at the inlet for all test conditions.

$$M_{isen} = \sqrt{\frac{2}{\gamma - 1}\left[\left(\frac{P_{01}}{P_s}\right)^{\frac{\gamma-1}{\gamma}} - 1\right]} \tag{4.43}$$

Condition	P_{01} (bar)	M_{isen}
MUR43	1.435	0.840
MUR44	1.433	0.840
MUR45	1.475	0.875
MUR46	1.478	0.875
MUR47	1.596	1.020
MUR48	1.605	1.020

Table 4-B: Test conditions for LS89 turbine cascade

Good overall agreement is obtained between the experimental results and the RANS simulation. One can observe that the RANS solution deviates from the experimental values near the aft portion ($x > 25$ mm) of the blade. A preliminary literature search shows that a similar trend has been observed by other researchers [146, 147, 148, 149]. In the experiments the downstream static pressure was measured along a plane parallel to the trailing edge, located at non-dimensional location $x/c = 1.433$ or 16.0 mm downstream of the trailing edge, which is outside the test section and hence outside the computation domain. Therefore, there is a level of uncertainty involved in the exit back-pressure specification for the LS89 cascade. In addition, STAMPS does not employ a non-reflecting exit boundary condition. Therefore, non-reflecting exit boundary with longer inlet and corrected back pressure may reduce this discrepancy [150]. This uncertainty in back pressure itself can be an interesting UQ study.

Full convergence of the RANS solver was achieved for all test conditions without any local limit cycle oscillations (LCO). This is highly essential to ensure the convergence of the adjoint [33] and tangent linear solution. The near wall spacing of $y^+ \approx 1$ was used for all test cases.

4.7.5 Turbocharger Compressor with Vaneless Diffuser[2]

A centrifugal turbocharger compressor with a vaneless diffuser provided by Mitsubishi Heavy Industries (MHI) was used in the present solver validation study. The impeller

[2] Joint work with Dr. Xu

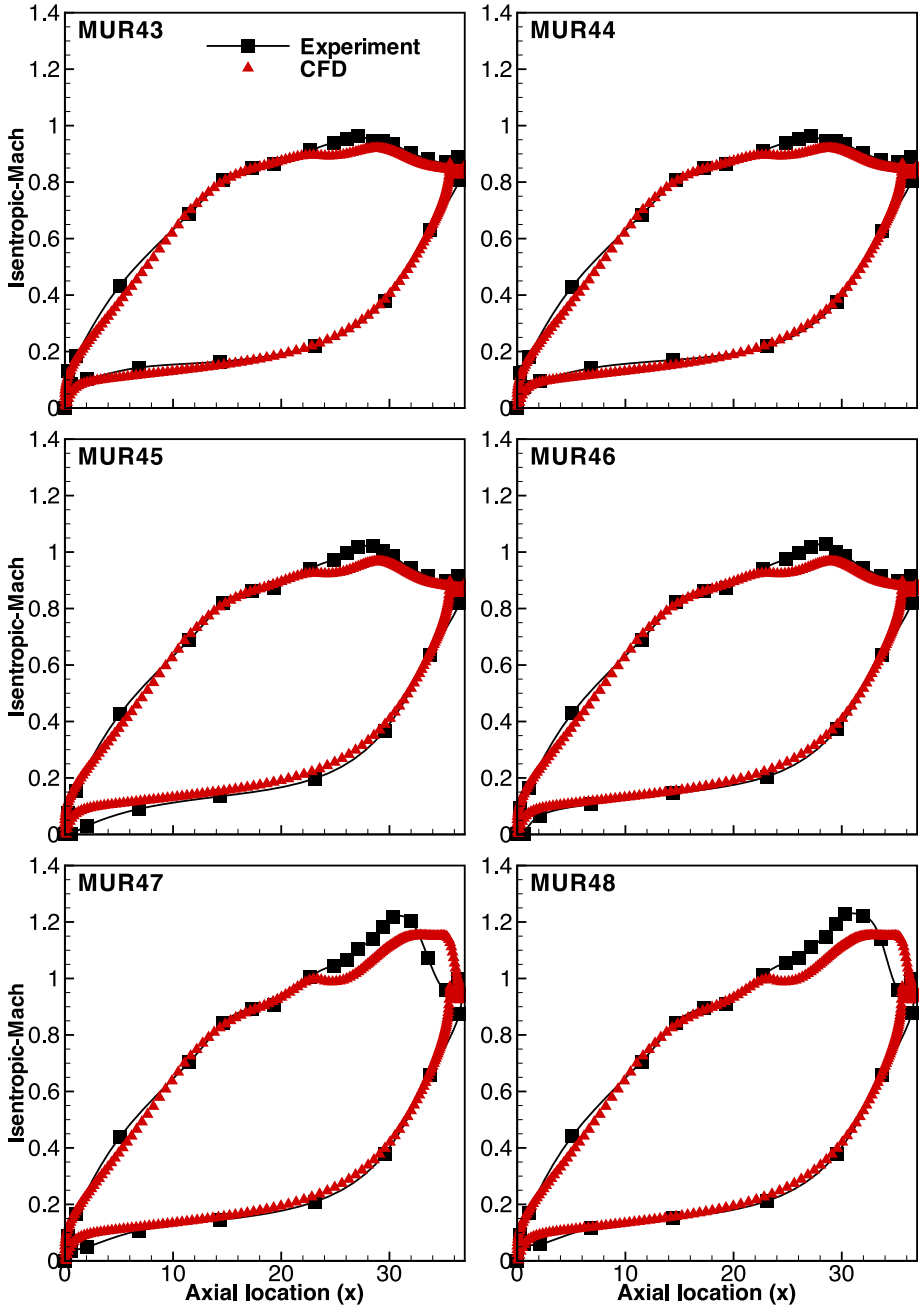


Figure 4.22: Surface distribution of isentropic Mach number for the VKI LS89 turbine cascade at indicated test conditions from ref. [144]

has a tip radius in the range of $25 - 30$ mm and has six main blades and corresponding number of splitter blades. The tip clearance is few tenths of a millimetre. One sixth of the full annulus was used for the steady state calculation. The computational domain comprises of (i) an inlet pipe, (ii) an impeller, and (iii) a vaneless diffuser as shown in

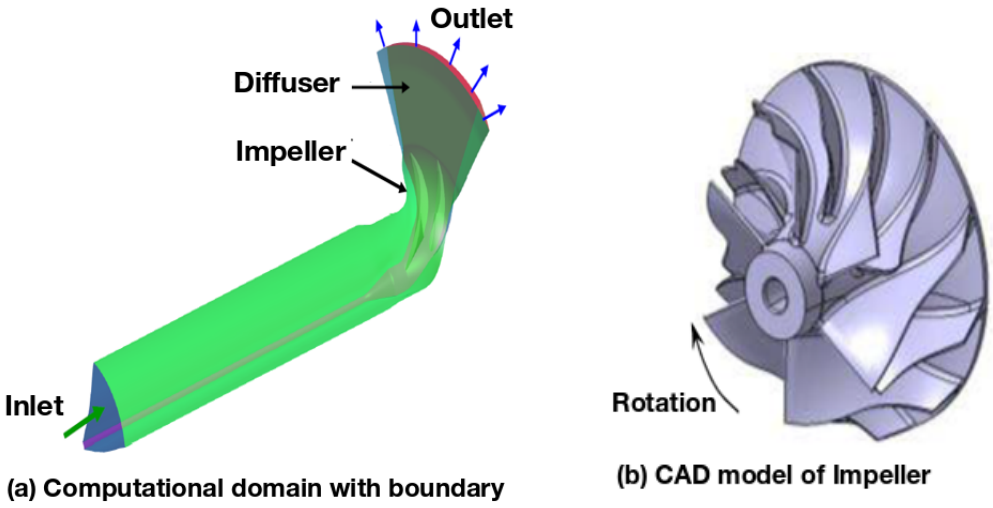


Figure 4.23: Centrifugal turbocharger compressor (a) computational domain and (b) impeller CAD model

fig. 4.23(a). The CAD model of the impeller wheel is shown in fig. 4.23(b).

Due to axisymmetry the entire configuration is considered a single rotating zone in order to avoid the use of an interface between the rotating zone and the diffuser. Standard atmospheric conditions were specified at the inlet where the flow enters axially (zero pre-swirl). The compressor rotates at a design rotation speed in the range of around $1.3 - 1.8 \times 10^5$ RPM, where RPM is revolutions per minute. The calculation was performed using three different mesh densities (i) coarse mesh with 76k nodes, (ii) medium mesh with 550k nodes, and (iii) fine mesh with 4M nodes. The meshes were provided by MHI, which were generated using the Ansys Turbogrid mesh generation software.

The compressor map of pressure ratio and efficiency for various back-pressures from choke to stall condition for the design RPM was simulated and the results are plotted in fig. 4.24. For comparison purpose fine mesh results from Ansys CFX commercial solver is plotted along with STAMPS results in fig. 4.24. A total-pressure and total-temperature inlet condition was imposed and a constant back-pressure was imposed at the outlet. To ensure accurate comparison: the Roe Riemann solver for convective fluxes, Green-Gauss gradient, SA turbulence model, and same inlet/outlet boundary conditions were used in

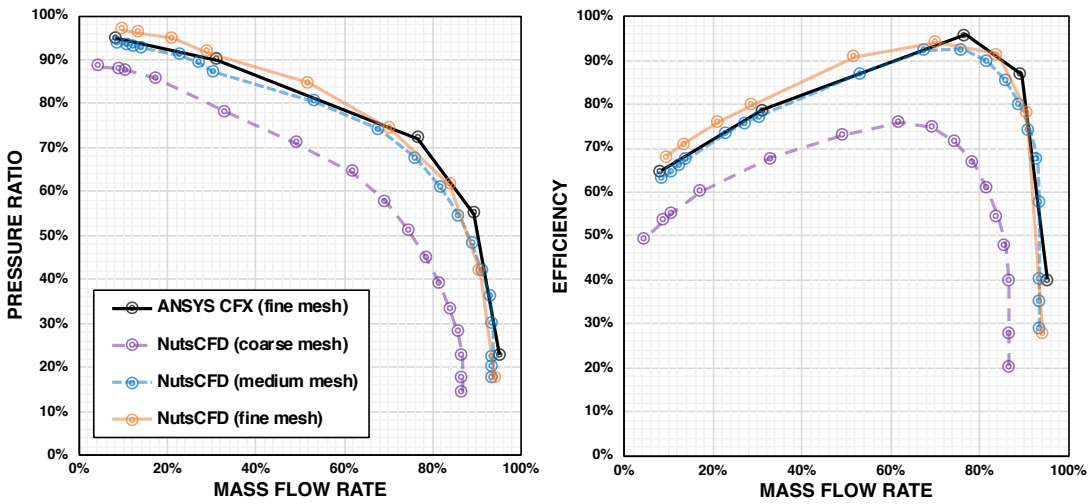


Figure 4.24: Map of pressure ratio (left) and efficiency (right) vs. mass flow rate at design RPM for the Turbocharger compressor (scales are non-dimensional due to confidentiality)

both solvers (namely Ansys CFX and STAMPS). The STAMPS fine mesh results (solid orange line in fig. 4.24) agree well with the ones obtained using Ansys CFX (solid black line in fig. 4.24). The solver had stable convergence for the range of back pressures used in this validation. In addition, both compressor map results converges well with grid refinement from coarse to fine meshes.

4.8 Surface Sensitivity Verification

The surface sensitivity verification is conducted using the LS89 turbine cascade test case. The MUR43 test condition is considered and the gradients of the QoI, namely exit mass flow rate and total-pressure loss are verified in this study. The gradient results obtained using adjoint, tangent-linear and finite-difference are compared using two types of verification, namely, (i) the stronger point-wise (Fréchet) and (ii) the weaker directional derivative (Gâteaux). The verification in each type of derivative is shown in the next sub-sections.

4.8.1 Point-wise Sensitivity Verification

As the name suggests, here one choses an arbitrary point on the surface of the cascade. Then the point is perturbed in either x or y direction with varying finite-difference step sizes. In fig. 4.25(a) the point P1 is perturbed along the x-axis by step size δx. The first order Taylor remainder convergence (forward difference) can be defined as shown in eq. (4.44).

$$|J(x + \delta x) - J(x)| \to 0, \quad \text{at} \quad \mathcal{O}(\delta x) \tag{4.44}$$

To evaluate the term $J(x+\delta x)$ the surface perturbation must be converted to a volumetric perturbation using the IDW mesh deformation (see appendix B). The flow is solved on the perturbed mesh to obtain the cost-function value $J(x+\delta x)$. The volume deformation field generated by the perturbation of surface node P1 is shown in fig. 4.25(b). Note that one can obtain the value of $\frac{dJ}{dx}$ using the tangent linear or the adjoint sensitivity. The adjoint sensitivity is simply the x-component of the surface sensitivity at node P1. But the tangent linear sensitivity requires the solution to the volumetric deformation field due to unit perturbation ($\delta x = 1$) at the surface node (fig. 4.25(b)). Gradients obtained from both sensitivities are tabulated in tab. 4-C.

Cost function	**Adjoint Sensitivity (at P1)**	**Tangent Sensitivity (at P1)**
Exit Mass flow	$+\mathbf{6.482}080$	$+\mathbf{6.482}012$
Total-pressure loss	$-\mathbf{3.465}840$	$-\mathbf{3.465}725$

Table 4-C: Adjoint and Tangent sensitivity at surface node P1 with respect to x direction

The first order Taylor remainder is plotted fig. 4.26(left) and the relative error in sensitivity between the adjoint and finite-difference results are plotted as a function of step size in fig. 4.26(right). For both cost-functions large step size has a large truncation error, which reduces with step size. But below a step size of 10^{-6} the round-off error dominates.

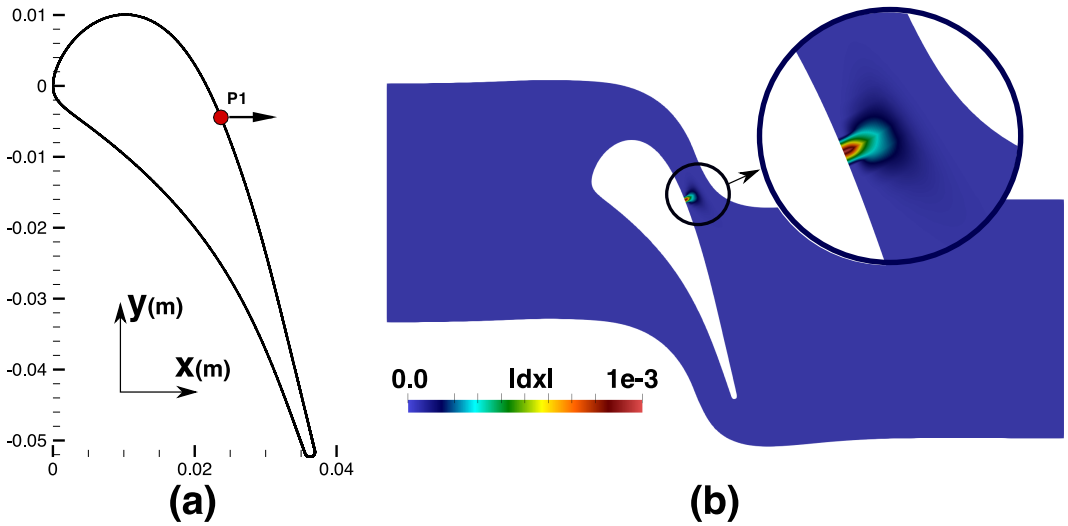


Figure 4.25: (a) Surface displacement of point P1 (x-axis) and (b) generated volumetric displacement after IDW mesh smoothing (for point-wise gradient verification)

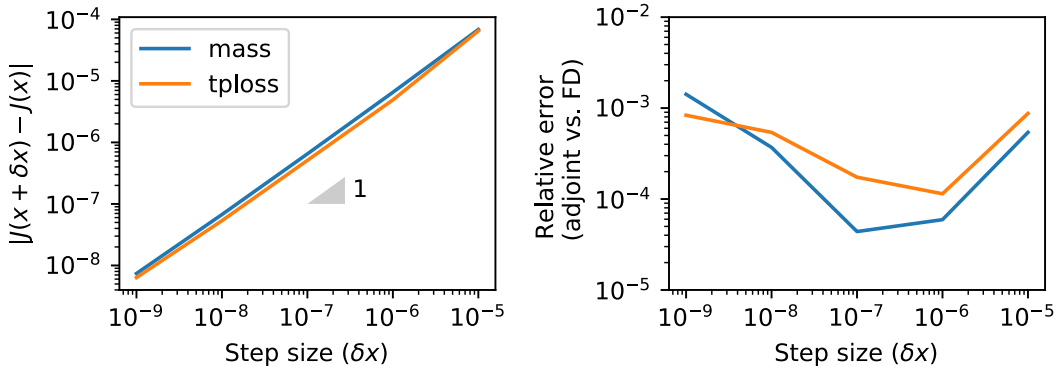


Figure 4.26: First order Taylor remainder $\mathcal{O}(h)$ vs. step size (left) and relative error between the finite-difference and adjoint sensitivity for surface point P1 (right) for the exit mass flow rate (blue) and total-pressure loss (orange) cost functions

4.8.2 Directional Derivative Verification

To obtain the directional derivative one needs to define a direction vector $\mathbf{h}$. Extraction of surface modes using principal component analysis (PCA) is shown in the next chapter in sec. 5.5.4. In this verification exercise the surface mode obtained using the PCA in sec. 5.5.4 is considered as the surface direction vector $\mathbf{h} \equiv \mathbf{z}_j$. Then using Taylor expansion one obtains the finite-difference form (forward difference) as shown in eq. (4.45),

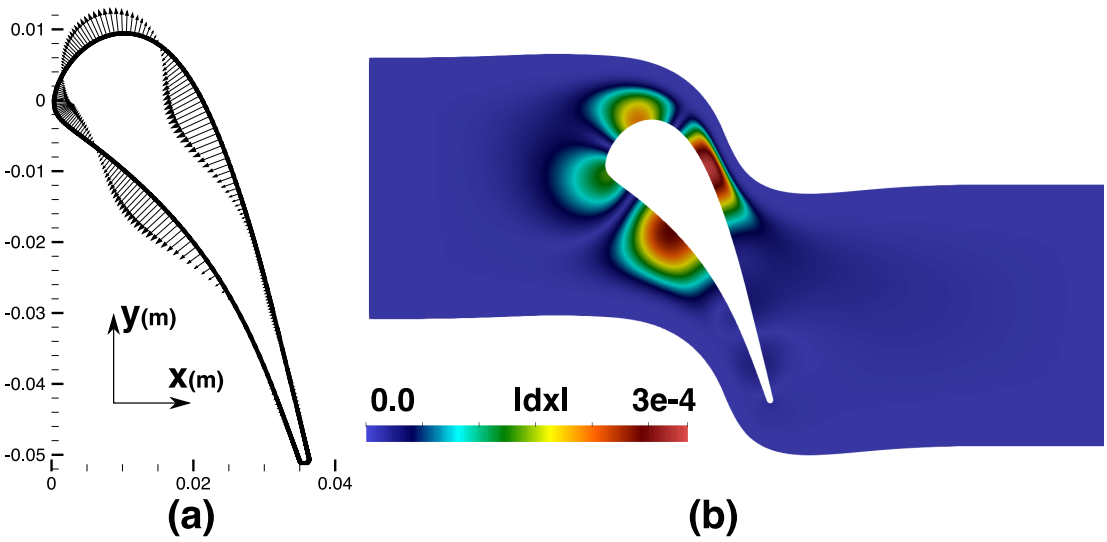


Figure 4.27: (a) Surface displacement of PCA mode 8 and (b) generated volumetric displacement after IDW mesh smoothing (for directional derivative verification)

The first order Taylor remainder convergence is defined as shown in eq. (4.46).

$$\frac{dJ}{d\mathbf{z}_j} = \frac{J(\mathbf{x} + \epsilon\mathbf{z}_j) - J(\mathbf{x})}{\epsilon} + \mathcal{O}(\epsilon) \tag{4.45}$$

$$|J(\mathbf{x} + \epsilon\mathbf{z}_j) - J(\mathbf{x})| \to 0, \quad \text{at} \quad \mathcal{O}(\epsilon) \tag{4.46}$$

The term $J(\mathbf{x} + \epsilon\mathbf{z}_j)$ is obtained from the flow solution at the perturbed mesh due to the surface deformation $\mathbf{x} + \epsilon\mathbf{z}_j$. PCA mode 8 from sec. 5.5.4 is used for the present gradient verification exercise. The adjoint and tangent linear sensitivities obtained for this surface mode are tabulated in tab. 4-D. Overall the the variations in the computed sensitivities of the various methods (finite-difference, adjoint, and tangent linear) are within a tolerance band of $< 0.1\%$ of the cost-function.

Cost function	**Adjoint Sensitivity**	**Tangent Sensitivity**
Exit Mass flow	$-\mathbf{6.717}423 \times 10^{-2}$	$-\mathbf{6.717}341 \times 10^{-2}$
Total-pressure loss	$+\mathbf{1.260}466 \times 10^{-2}$	$+\mathbf{1.260}385 \times 10^{-2}$

Table 4-D: Adjoint and Tangent sensitivity for surface mode 8

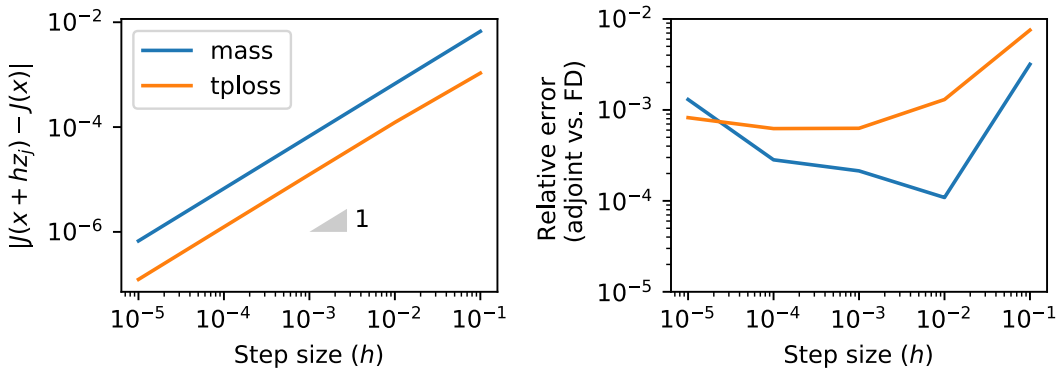


Figure 4.28: First order Taylor reminder $\mathcal{O}(h)$ vs. step size (left) and relative error between the finite-difference and adjoint sensitivity for surface PCA mode 8 (right) for the exit mass flow rate (blue) and total-pressure loss (orange) cost functions

4.9 Summary

The parallel aerodynamic primal, adjoint, and tangent linear solver implementation and performance improvements were discussed in this chapter. The two-halo partitioning scheme used in this work was shown to reduce the overall communication by half compared to a one-halo partition. In addition the adjoint and tangent-linear solver development was simplified by avoiding MPI calls inside the residual evaluation function. The time-implicit method was outlined and validation results from three preliminary test cases were shown. The implemented solver proved to be robust and provided reasonable solution accuracy for the chosen test cases and serves as the HF model for the FastUQ method.

Chapter 5

Modelling Geometric Uncertainties

"The strength of numbers bolstered by the power of images is enough to sustain in the public an irrational, quasi-mystical mind-set."
— Pierre-Giles de Gennes & Jacques Badoz, Fragile Objects

5.1 Manufacturing Uncertainties in Turbomachines

Various sources of geometric uncertainties exist in turbomachinery, for example, noisy manufacturing processes, wear, foreign object damage and deformations under loading. The variations due to manufacturing is considered in this work. In fig. 5.1, two types of blade manufacturing processes are shown namely, (i) Flank milling (FM) and (b) Point milling (PM). In FM, a conical tool is used to cut the entire surface of the blade from a blank material in a single pass [151]. In PM, a ball cutter removes material from a block based on the planned tool path. PM is disadvantageous because of slower manufacturing time and poor surface finish when compared with FM. Chattering occurs due to inherent vibration of the tool which also generates surface imperfections. Similarly, in FM, it is not possible for the tool to cut exactly along a ruled surface, thereby generating under and over-cuts (see fig. 5.1). The tool also wears out in due course of manufacturing and empirical tool compensations are applied, which add to the uncertainty. The measured

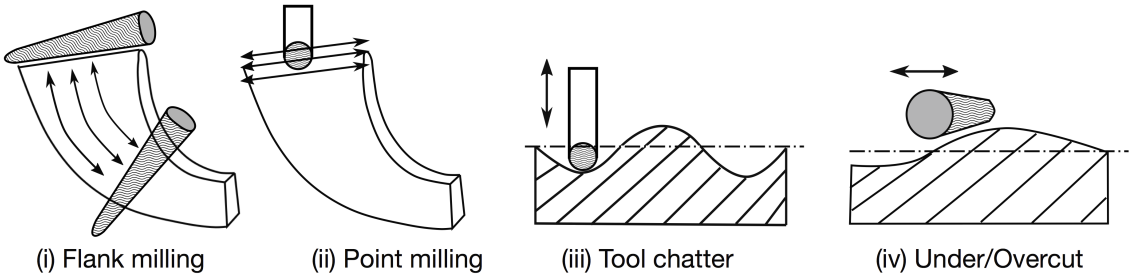


Figure 5.1: Flank and point milling process and sources of noise in manufacturing

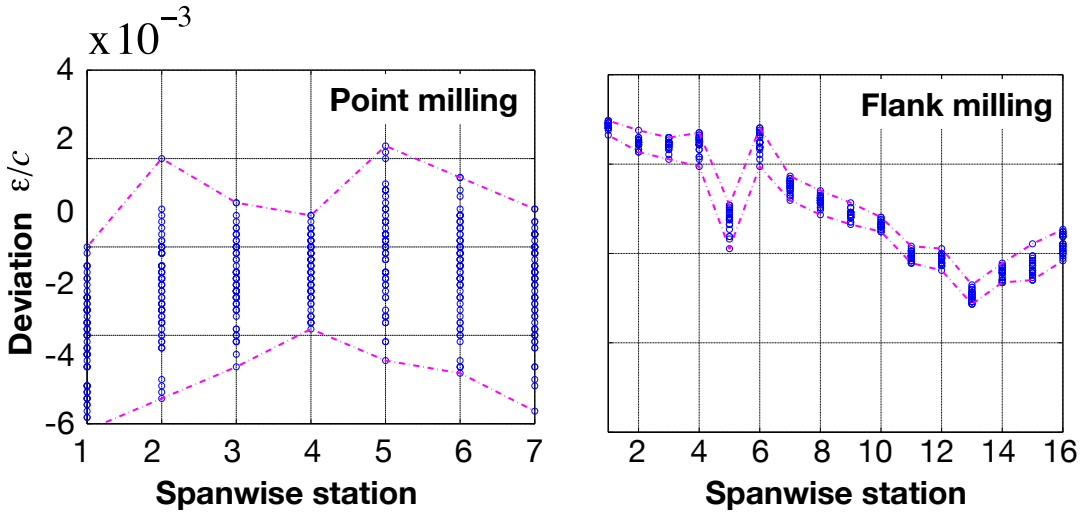


Figure 5.2: Measured deviation of leading edge thickness for sample point and flank-milled integrally bladed rotor (from ref. [152])

deviation in leading edge (LE) radius of an integrally-bladed rotor (IBR) between point and flank milling process in shown in fig. 5.1 (from Garzon [152]). An IBR is a type of rotor where the blade and the disk are manufactured as a single part. Large deviations from the design intent surface were observed for the leading edge thickness of the blade (see fig. 5.1), which is a important parameter affecting turbomachine performance. Garzon [152] estimates that the mean polytropic efficiency of a six stage axial compressor decreases by 1% due to manufacturing variations. In turbomachines even a gain of 1% in losses can create a big impact on the overall design.

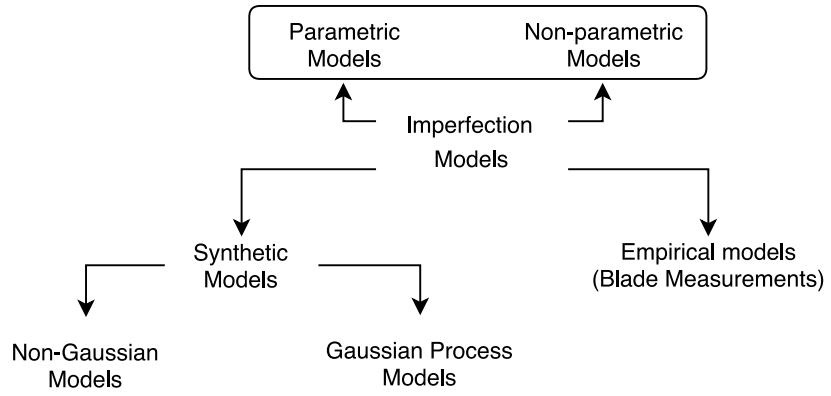


Figure 5.3: Overview of models for surface imperfection due to manufacturing

5.2 State-of-art in Surface Imperfection Models

An overview of surface imperfection models available in literature is summarised in fig. 5.3. Two main approaches to modelling surface variations due to manufacturing are adopted widely in the literature. In the first approach, a database of manufactured shapes is built by measuring real manufactured samples. Then this databased is used to create stochastic models for surface variations, whose effect on performance is quantified. For example, Lange [18] considered blade measurements from 3-D surface scan data of a high pressure compressor stage. The correlated coordinates from the 3-D scan were translated into mode shapes and uncorrelated modal amplitudes using principal component analysis (PCA) [153]. Garzon [152] used a similar PCA based approach to model measurements for a compressor blade geometry. The principal modes and amplitudes were used to model surface variations in his robust shape optimisation framework.

In the second approach, one resorts to synthetic, but, heuristically-based models for geometric variability. The blade geometries and associated measurement data are classified information and most manufacturers are unwilling to share them for academic or research purpose. Such synthetic models for variability are mostly modelled as Gaussian processes (GP) or Gaussian fields (GF) in the literature. Roughly speaking, a stochastic process is a generalisation of a probability distribution (which describes a finite-dimensional random variable) to functions [154], when the collection of distributions are jointly Gaussian

then the stochastic process is called a Gaussian process. One can also consider a general parameter space such that the stochastic process becomes a random function of more than one variable. This type of stochastic process is usually called a random field [155]. An introduction to stochastic process and GP is provided in sec. 5.3 for the interested reader.

Some examples of GP models include the work of Hacker [156] who used principal component analysis and other statistical techniques to produce reduced-order models of compressor blade performance. Hacker [156] used heuristically-based models for geometric variability using Gaussian distributions for the surface perturbations on turbine blades. Schillings [39] used a Gaussian random field with an analytically specified mean and covariance function to model surface variations of an airfoil. Schillings [39] avoided parametrisation of the uncertainties and the stochastic model considered the random perturbation of every point on the airfoil surface. This gives the maximum possible space of perturbed geometries or shapes. A similar approach to Schillings [39] has been adopted by Wang [157], and Dow [158] to model surface variations on compressor blades. In ref. [159], Dow and Wang modelled manufacturing tolerances as Gaussian random fields and obtain optimal tolerance distribution on the blade by associating tolerance to manufacturing cost. In addition, proper orthogonal decomposition was used to represent the random tolerance field as a spectral decomposition of its covariance function. An advantage of using GP is that the variability is modelled simply by defining a mean and covariance function. In addition, uncorrelated Gaussian fields (principal components) are statistically independent [154], which is an important property to have for an input random field.

The methods surveyed in the preceding paragraphs considered all nodes on the surface as an uncertain parameter without any parametrisation. An alternate approach is to model geometric variability using a parametric space [160, 161, 162]. Typically, in the parametric approaches a set of geometric parameters such as stagger angle, chord length, leading edge thickness, maximum camber, etc., are empirically or synthetically obtained

for a large sample set of blades. One fits probability distributions to model each parameter using this database of blade profiles. Ghate [1] used a parametric model of an airfoil geometry, randomly perturbed the parameters to model surface uncertainties and used independent component analysis [41] to account for non-Gaussianity. Pisaroni et al. [21] used PARSEC airfoil parametrisation [163] and modelled the PARSEC parameters using random distributions. An advantage of the parametric approach is that the geometric variability is described purely using design parameters. This aids in maintaining smaller input dimensions, which is required for certain UQ methods such as polynomial chaos expansion discussed ch. 2. However, one severely restricts the space of possible perturbed blade shapes neglecting shapes that can adversely impact the performance of the blade.

5.3 Background on Stochastic Processes

5.3.1 Definition of Stochastic Process

The stochastic or random process (SP) extends the concept of a random variable to functions of some independent random variable. Usually, they are used to model time varying random functions but the independent variable has no restriction on what it can represent. A stochastic process in this thesis is denoted by $f(x)$. It is best described using an example.

Consider a plane surface Γ whose surface coordinates are $\mathbf{x} \in \Gamma$, (denoted by the green line in fig. 5.4). Assume that the surface undergoes random perturbation modelled using the stochastic process $f(\mathbf{x})$. Sample measurements of this random surface $\mathbf{x}$ (denoted by red lines in fig. 5.4) yields the samples $\{f_i(\mathbf{x})\},\ i = 1, 2, \ldots$ etc. If one were to observe the sample measurements at a fixed spatial location $\mathbf{x} = \mathbf{x}_1$ then one obtains a random variable $\mathbf{x}_1$ with an associated probability distribution $p(\mathbf{x}_1)$. Similarly every sample $f_i(\mathbf{x})$ itself is a random variable. Therefore, to characterise the process $f(\mathbf{x})$ completely one needs to know the joint probability distributions of $\{f_i(\mathbf{x}_1)\},\ \{f_i(\mathbf{x}_2)\},\ \ldots$ etc. (see sec. 2.1.4 for definition of joint probability distribution).

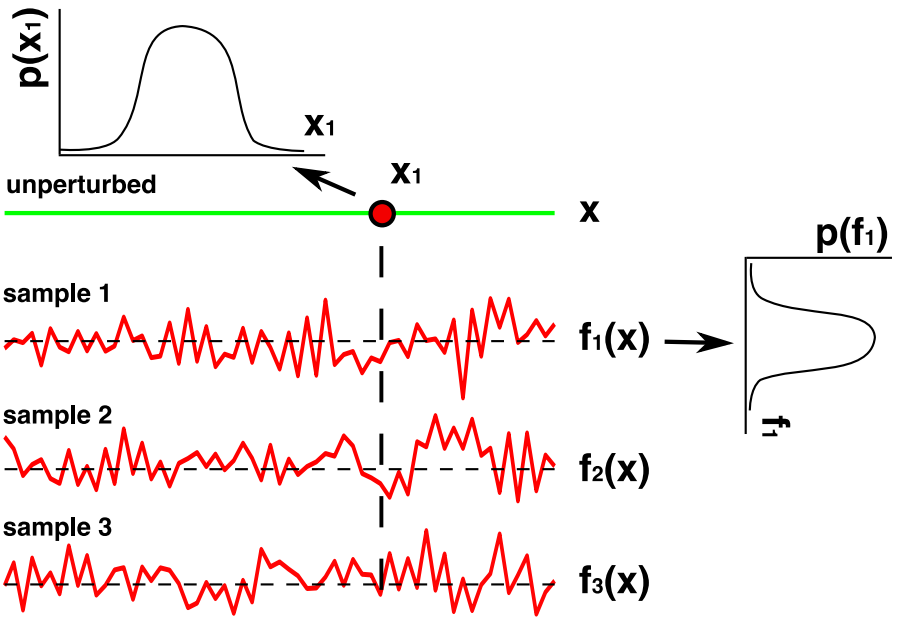


Figure 5.4: Stochastic process example: original unperturbed coordinate x shown using solid green line and three sample perturbed surfaces from the Gaussian model shown using solid red line.

5.3.2 Stationarity, Mean, and Covariance Function

A stochastic process is said to be *stationary* in the strict sense if its joint distribution is invariant under spatial shifts of origin i.e, the *pdf* only depends on $|\mathbf{x}_i - \mathbf{x}_j|$ (for some location $i,\ j$) and does not directly depend on individual locations $\mathbf{x}_1, \mathbf{x}_2, \ldots$ etc.

The mean function μ_f or $\mathbb{E}[f(\mathbf{x})]$ of a stochastic process $f(\mathbf{x})$ is defined using eq. (5.1). The mean is usually a function of space $\mathbf{x}$ but for a stationary process it is a constant (μ_f) for all random variables belonging to different spatial realisations with the same *pdf* defined by $p(f)$.

$$\mu_f = \mathbb{E}[f(\mathbf{x})] = \int_{-\infty}^{\infty} f(\mathbf{x}) p(f(\mathbf{x})) df(\mathbf{x}) \tag{5.1}$$

One can define two types of variance for a stochastic process. The usual variance definition is shown in eq. (5.2) and the covariance for two spatial locations $\mathbf{x}_i$ and $\mathbf{x}_j$ given in

eq. (5.2). Note that for $\delta\mathbf{x} = \mathbf{x}_j - \mathbf{x}_i = 0$ in eq. (5.3) one recovers eq. (5.2).

$$\sigma_f(\mathbf{x}) = \mathbb{E}\left[(f(\mathbf{x}) - \mu(\mathbf{x}))\ \ (f(\mathbf{x}) - \mu(\mathbf{x}))^T\right] \tag{5.2}$$

$$\mathbf{C}_{ij} = \mathbb{E}\left[\ (f(\mathbf{x}_i) - \mu(\mathbf{x}_i))\ \ (f(\mathbf{x}_j) - \mu(\mathbf{x}_j))^T\ \right] \tag{5.3}$$

Similar to the mean, the variance and covariance functions are spatially invariant constants for a *stationary process*. In addition, the covariance function is symmetric [154]. For a discrete spatial domain a covariance function becomes a covariance matrix (Gram matrix).

5.3.3 Gaussian Process

A Gaussian process (GP) is a special case of SP, where the collection of random variables $f_i(\mathbf{x}_1), f_i(\mathbf{x}_2), \ldots$, etc., any finite number of which have a joint Gaussian distribution [154]. A GP can be completely defined by its mean function μ_f and co-variance function $\mathbf{C}_{ij}$ and it is usually denoted using the short notation $f(\mathbf{x}) \sim \mathcal{GP}(\mu_f, \mathbf{C}_{ij})$.

5.4 Stochastic Surface Variation Model

In this work a synthetic surface perturbation model based on GP similar to the approach presented in refs. [39, 164, 156, 165] has been adopted due to a lack of access to empirical blade measurements. In addition, parametrisation of the perturbations is avoided since this would lead to a reduction in the space of possible perturbed geometries [166]. Therefore, a free node parametrisation [166] is used where every surface node is a random parameter defining the blade shape. The infinite dimensional probability space is approximated using a finite number of random variables, where every surface mesh node is a parameter [164].

Let $\mathbf{x} = \{x_1, x_2, ..., x_N\}$, where each $x_i \in \mathbb{R}^3$, be the set of coordinates defining the nominal surface of the blade and let $\hat{n}$ denote the normal vector of this nominal surface. A zero-mean Gaussian process $\delta(\mathbf{x})$ is imposed along the normal direction to this nominal

surface to generate the perturbed surface x_δ as,

$$\mathbf{x}_\delta = \mathbf{x} + \delta(\mathbf{x})\hat{n}. \tag{5.4}$$

In addition, the random variables $\delta_i = \delta(x_i)$ and $\delta_j = \delta(x_j)$ at any two arbitrary points i and j on the surface (see fig. 5.5) are assumed to have a squared exponential spatial covariance as shown in eq. (5.5). The squared exponential covariance has been extensively used in robust optimisation studies involving surface perturbations (see ref. [167, 28, 168]). This covariance function is infinitely differentiable, which means that the GP with this covariance function has derivatives of all orders (in the mean square sense), and is thus very smooth [154]. In eq. (5.5), the parameter b controls the height of the perturbations and l is the characteristic correlation length of the perturbation. A large value of l results in a larger spread of the disturbance around a given surface node. In this work, the perturbation height b and the correlation length l are assumed be a fraction of some characteristic dimension. The parameters b and l are illustrated for the surface perturbation over the LS89 cascade surface in fig. 5.5. The model is used to represent the surface variations over the LS89 turbine cascade, which is shown in the next section.

$$\mathbf{C}_{ij} = b^2 \exp\left(-\frac{||x_i - x_j||^2}{2l^2}\right) \tag{5.5}$$

5.4.1 LS89 Geometry and Surface Variation Model

The aerodynamic uncertainty quantification due to surface variation using FastUQ is demonstrated using the LS89 turbine cascade presented in sec. 4.7.4. The aim here is to estimate the statistics (mean and variance) of the QoIs mass flow rate (J_{mass}) and total pressure loss (J_{ploss}), subject to the surface perturbations on the cascade surface using the model described in the previous section. The surface disturbance model is a zero-mean Gaussian process imposed normal to the blade surface and follows a squared exponential spatial covariance function as discussed in the previous section. The height b is assumed to be approximately 10% of the trailing edge radius and the length l is

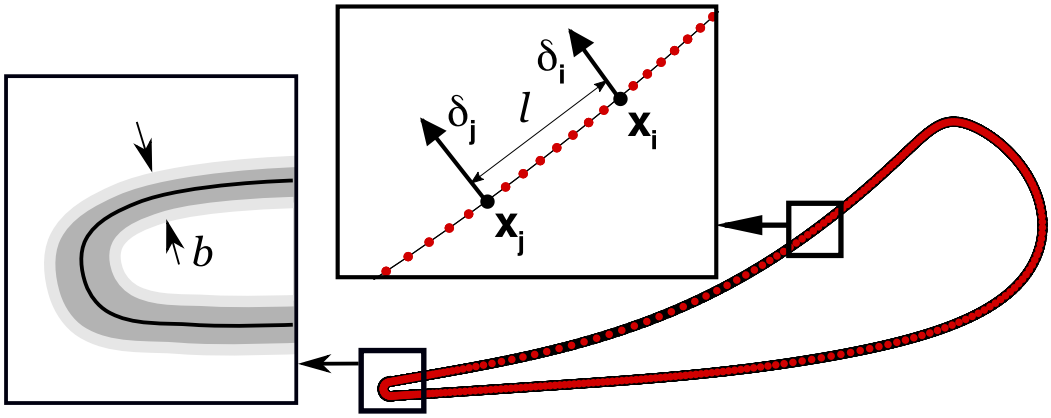


Figure 5.5: Surface node distribution on LS89 cascade surface along with the two arbitrary surface nodes i and j used to illustrate the spatial covariance function in eq. 5.5

Quantity	**Value**
Pitch	57.500 mm
Axial chord	36.985 mm
Leading-edge radius	4.127 mm
Trailing-edge radius	0.710 mm
Perturbation height (b)	0.1 mm
Perturbation width (l)	10.0 mm

Table 5-A: VKI LS89 cascade geometry

Quantity	**MUR43**	**MUR47**
Inlet total-pressure (P_{01})	1.435 bar	1.596 bar
Inlet total-temperature (T_{01})	420 K	420 K
Outlet back-pressure	0.904 bar	8.236 bar
Outlet isentropic Mach (M_{isen})	0.8	1.2

Table 5-B: $MUR43$ and $MUR47$ test conditions

approximately equal to the leading edge diameter of the cascade turbine cascade (see fig. 5.5). The various geometric parameters of the cascade are shown in tab. 5-A.

Two test conditions, namely the MUR47 (transonic flow) and MUR43 (subsonic flow) are used for the UQ study. The details of the test conditions are tabulated in Table 5-B and 5-A.

5.5 Model Improvements Using Dimensionality Reduction and Independence

The convergence properties of uncertainty quantification methods such as Monte Carlo, gPCE, and Stochastic Collocation (discussed in ch. 2) are valid under the assumption of independent identically distributed (*i.i.d*) input parameter space [44] (for definition see sec. 2.1.5). Therefore, statistical independence is an important property to have for the input variations. For a joint Gaussian distribution to be independent its covariance function must be uncorrelated (from sec. 2.1.4) i.e., the covariance matrix $\mathbf{C}_{ij}$ in eq. (5.5) should be transformed to a diagonal matrix $\mathbf{\Lambda}$ to make the input uncertainties independent. Principal components are linear combinations of random or statistical variables having special properties in terms of variances [169] and the first principal component is the normalised linear combination with maximum variance. The eigen decomposition of the Covariance matrix $\mathbf{C}_{ij}$ (shown in eq. (5.6)) yields the linear combinations with maximal variances. Estimation of the eigenvalues $\mathbf{\Lambda}$ and eigenvectors $\mathbf{Z}$ is also called the *principal component analysis* (PCA).

$$\mathbf{Z}\mathbf{C}_{ij} = \mathbf{Z}\mathbf{\Lambda} \tag{5.6}$$

In eq. (5.6), $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\}$ are the eigenvectors or PCA modes and $\mathbf{\Lambda} = diag(\lambda_1, \ldots, \lambda_n)$, $\lambda_1 \geq \ldots \geq \lambda_n$ are the eigenvalues. The principal components of $\mathbf{C}_{ij}$ can be shown to be equivalent to its Singular Value Decomposition (SVD) in reduced form [170]. Consider the SVD decomposition of the real symmetric matrix $\mathbf{C}_{ij}$ as shown in eq. (5.7), where $\mathbf{U}$ and $\mathbf{V}$ are the left and right singular vectors and $\mathbf{\Sigma}$ is the diagonal matrix whose diagonal entries are the singular values.

$$\mathbf{C}_{ij} = \mathbf{U}\mathbf{\Sigma}\mathbf{V} \tag{5.7}$$

For a symmetric matrix, (i) the eigenvectors and left singular vectors are the same i.e., $\mathbf{U} = \mathbf{Z}$, (ii) the entries of the singular value matrix $\mathbf{\Sigma}$ are nonnegative singular values or

nonnegative eigenvalues i.e., $\mathbf{\Sigma} = |\mathbf{\Lambda}|$ and (iii) the right singular vectors $\mathbf{V}$ are related to the eigenvectors using a matrix $\mathbf{T}$, whose diagonal entries are ± 1 corresponding to the parity/sign of the eigenvalues of $\mathbf{\Lambda}$ i.e., $\mathbf{V} = \mathbf{TZ}$ [170]. For a positive definite $\mathbf{C}_{ij}$ the parity matrix is the identity matrix ($\mathbf{T} \equiv \mathbf{I}$).

The SVD is a numerically stable algorithm to compute the eigen decomposition of $\mathbf{C}_{ij}$ because it can handle rank deficient matrices (near zero-diagonal entries). The expansion using the eigenvalues and vectors of the discrete zero-mean surface perturbation is shown in eq. (5.8), where, $Y_i(\omega)$'s are uncorrelated Gaussian random variables with zero mean and unit variance [171] (see eq. (2.2) for definition of Gaussian distribution). The definition of the random variable $Y_i(\omega)$ ($\omega \in \Omega$) follows the same convention for probability space as show in sec. 2.1.1.

$$\delta(x_i, \omega) = \sum_{i=1}^{n} \sqrt{\lambda_i}\mathbf{z}_i Y_i(\omega) \approx \sum_{i=1}^{m} \sqrt{\lambda_i}\mathbf{z}_i Y_i(\omega) \tag{5.8}$$

The expansion is usually truncated to the first m dominant eigenmodes for reducing the input dimension. Using Mercer's theorem one can show that the truncated expansion is a suitable approximation, if the eigenvalues decay sufficiently fast and m is sufficiently large [166]. For a Gaussian covariance function one can prove that the eigenvalues will exponentially decay towards zero [172]. Therefore, with a fraction of the total number of modes one can reliably approximate the variations.

5.5.1 Truncation Using Partial Modal Fraction

The modal fraction $\bar{\lambda}_i$ and the partial modal fraction $\bar{\Lambda}_i$ shown in eq. (5.9) are useful quantities to aid the truncation. Partial modal fraction can be used to estimate the value of m given a user specified threshold error. For example, if one truncates after the i^{th} PCA mode and $\bar{\Lambda}_i = 0.9$ then 90% of the total input variations (spectral content) are captured using the truncated set. A value of $\bar{\lambda}_i = 0.1$ denotes that the i^{th} PCA mode

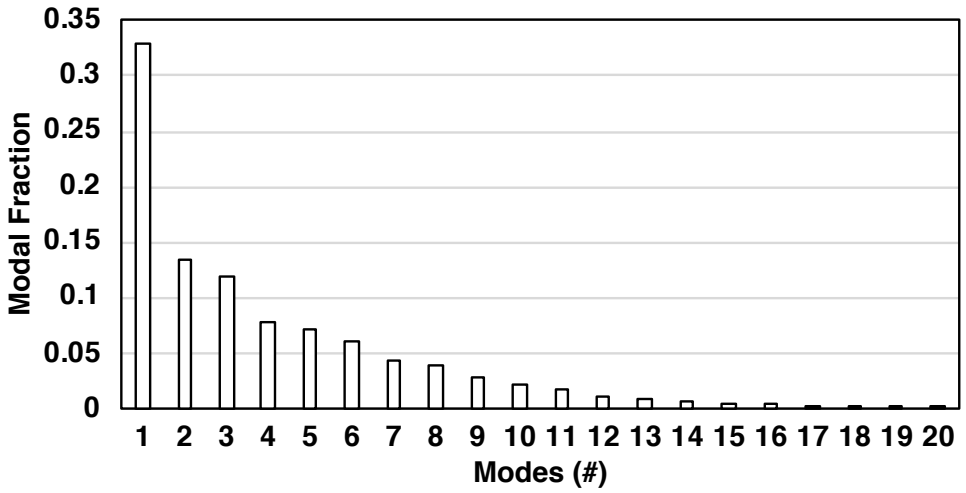


Figure 5.6: Modal fraction for the LS89 turbine cascade first 20 PCA modes

accounts for 10% of the total input variations (spectral content).

$$\bar{\lambda}_i = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}, \qquad \bar{\Lambda}_i = \frac{\sum_{j=1}^{i} \lambda_i}{\sum_{j=1}^{n} \lambda_j} \tag{5.9}$$

5.5.2 LS89 Partial Modal Fraction Truncation Results

The modal and partial modal fraction for the first 20 PCA modes computed for the LS89 turbine cascade subject to the GP surface disturbance model (see sec. 5.4.1) is shown in fig. 5.6 and 5.7. The exponential decay to zero of the mode strength can bee seen in fig. 5.6. The first 20 PCA modes are necessary to capture 99% of the spectral content of the input uncertainty Using the first 10 PCA modes one can capture slightly greater than 90% of the spectral content.

While PCA yields independent vectors having maximal variance, it does not contain the influence of these vectors on the function of interest. PCA modes cannot be truncated reliably unless this additional information is available. This motivates the goal-based PCA approach [168] using the adjoint sensitivity information presented in the next section.

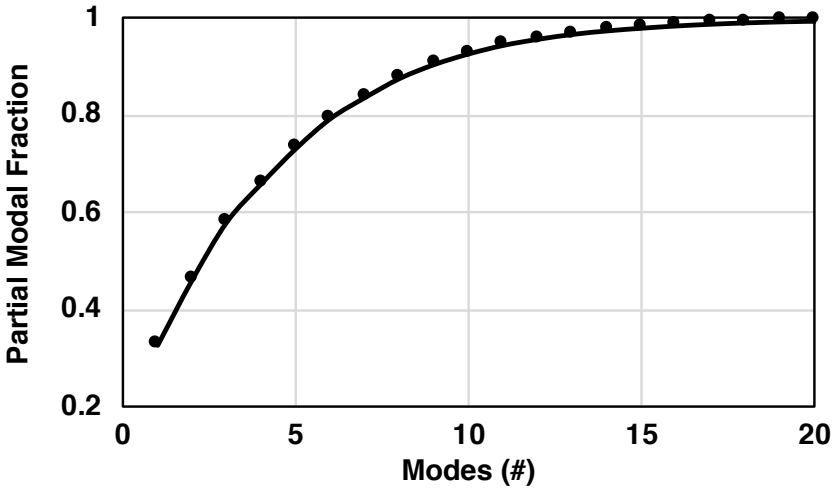


Figure 5.7: Partial modal fraction for the LS89 turbine cascade first 20 PCA modes

5.5.3 Goal-based Truncation for Multiple QoIs

The impact of an input variation α on the QoI J is encoded in the adjoint sensitivity. Therefore, by replacing α with the eigen modes of the PCA one can estimate the influence of the mode on QoI J. Since the adjoint sensitivity is independent of the number of input parameters, it is evaluated once and used for all the PCA modes.

$$\eta_{PCA_i} = \lambda_i \frac{dJ}{d\mathbf{z}_i} \approx \lambda_i \frac{dJ}{dx} \cdot \mathbf{z}_i \tag{5.10}$$

Once the PCA modes are obtained they are projected on to the sensitivity field to obtain the effectiveness factor (η_{PCA}) as shown in eq. (5.10). The PCA modes are sorted using η_{PCA} and truncated using a new modal fraction defined in eq. (5.11). Note that eq. (5.11) is obtained by replacing λ with the η_{PCA} in eq. (5.9). The projected PCA modes are called the goal-based PCA modes or simply G-PCA.

$$\bar{\lambda}_i = \frac{\eta_{PCA_i}}{\sum_{j=1}^{n} \eta_{PCA_j}}, \qquad \bar{\Lambda}_i = \frac{\sum_{j=1}^{i} \eta_{PCA_i}}{\sum_{j=1}^{n} \eta_{PCA_j}} \tag{5.11}$$

Most references on G-PCA applied to surface perturbation models consider only a single QoI or a single flow condition for the truncation, for example ref. [39]. In this work,

multiple QoIs at multiple flow conditions are considered for the G-PCA truncation.

For example, if one has multiple objectives, say J_1 and J_2, the ranking and truncation are performed individually for each objective and combined into a single set (union) of truncated G-PCA modes. A similar procedure is followed for multiple flow conditions.

5.5.4 LS89 Goal-based Truncation Results

The eigenvalues of the first ten eigenmodes of the PCA and corresponding G-PCA eigenvalue of the surface perturbation over the LS89 cascade are plotted in fig. 5.8 for both test conditions MUR43 and MUR47. The ranking of the first ten dominant mode numbers are tabulated in tab. 5-C for the exit mass flow G-PCA (EM-GPCA), and total-pressure loss G-PCA (TP-GPCA) eigenmodes. The G-PCA gives quite different ranking of the modes in comparison to PCA. In particular, the difference is quite dramatic for the MUR47 condition. Here PCA modes number 1 and 2 are in fact the least dominant in comparison to the TP-GPCA and the mode 3 is the most dominant TP-GPCA mode. The reason becomes clear when one plots the PCA modes against the surface sensitivity field as shown in fig. 5.9(a)-(l). PCA mode 2 in fig. 5.9(k) has deformations occurring only in regions of very low sensitivity. Similarly, PCA modes 2 and 3 are amongst the least dominant EM-GPCA and mode number 4 is the most dominant EM-GPCA mode. In particular, PCA mode 8 is highly dominant (considering EM-GPCA) for both test conditions. From fig. 5.9(a,d) one finds that mode 8 has a constructive superposition with the sensitivity field, which makes it quite effective.

Combining the (most dominant) five G-PCA modes in each QoI for the MUR43 test case i.e., $\{1, 4, 5, 6, 8\}$ one can capture 50% of the variations (in the G-PCA sense). Considering the 7 modes $\{1, 3, 4, 5, 6, 7, 8\}$ one can capture 50% of the spectral content of the input variations (G-PCA) considering both test conditions and QoIs. Clearly modes that are effective for a particular QoI and test condition need not be effective for another. *Therefore, when more QoIs and test conditions are considered the number of G-PCA modes necessary to capture the spectral content of the input variations increases.*

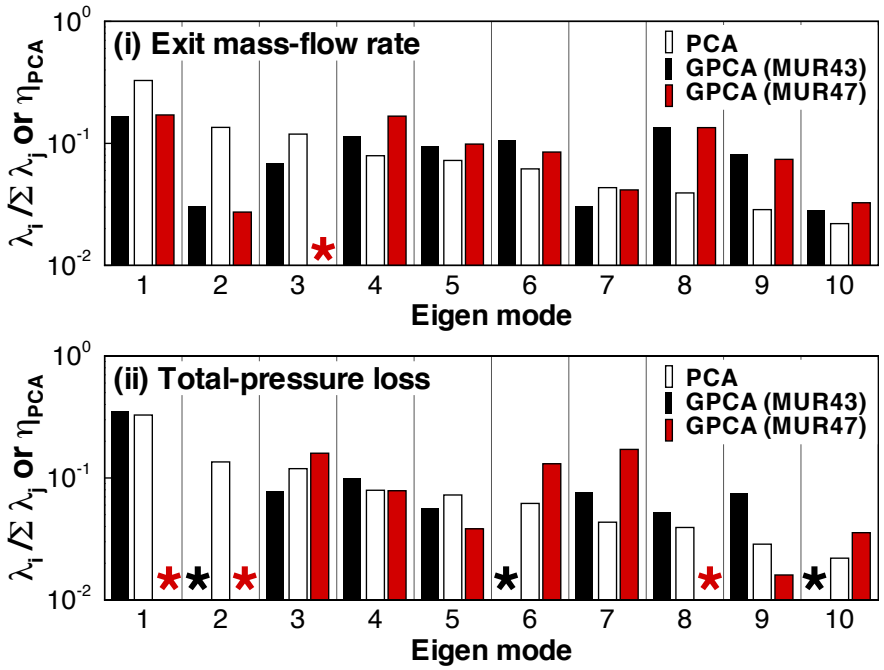


Figure 5.8: First ten PCA eigenmodes (along with the G-PCA counter-part) of the surface perturbation field over the LS89 cascade for indicated test condition and cost-function; modes with very low effectiveness ($< 10^{-2}$) are indicated by (*).

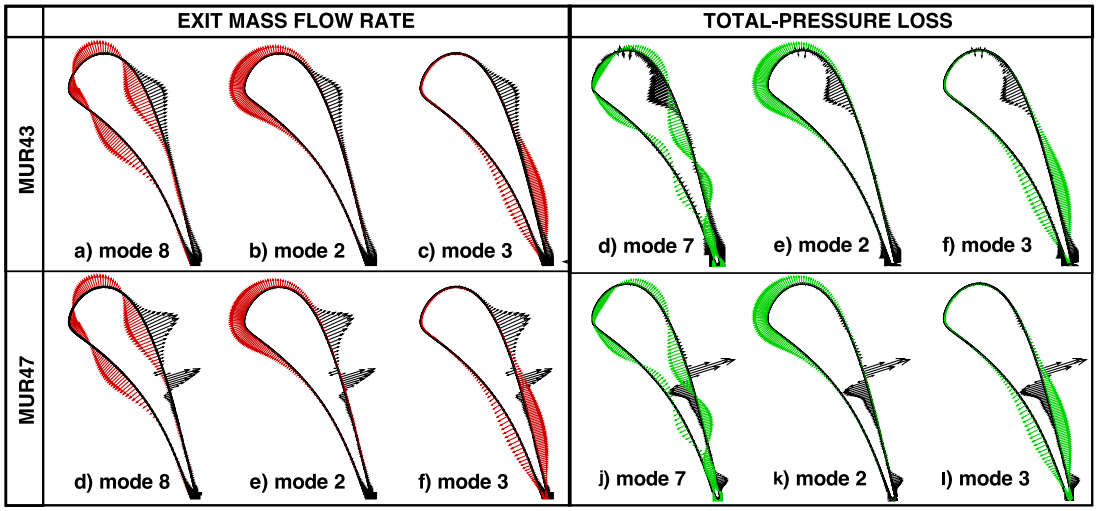


Figure 5.9: PCA eigenmode (coloured) and sensitivity (black) vector for the QoIs : *exit mass flow (red) and total-pressure loss (green)* plotted for test conditions MUR43 (top) (a)-(f) and MUR47 (bottom) (g)-(l) for indicated mode (trailing edge sensitivity clipped for better visualisation)

Table 5-C: Ranking of EM-GPCA and TP-GPCA modes w.r.t PCA for MUR43 and MUR47 test conditions: most-dominant (left) to least-dominant (right)

Condition	**Type**	**PCA mode #**									
MUR43	**EM-GPCA**	**1**	**4**	**8**	**6**	5	3	9	2	7	10
	$\bar{\Lambda}_i(\%)$	**17**	**30**	**42**	**52**	62	70	77	80	83	86
	TP-GPCA	**1**	**4**	**8**	**5**	6	9	2	7	10	13
	$\bar{\Lambda}_i(\%)$	**35**	**45**	**53**	**60**	68	73	78	83	87	89
MUR47	**EM-GPCA**	**1**	**4**	**3**	**7**	5	9	8	11	13	2
	$\bar{\Lambda}_i(\%)$	**17**	**33**	**47**	**57**	65	73	76	82	85	88
	TP-GPCA	**3**	**7**	**6**	**4**	5	11	10	15	14	1
	$\bar{\Lambda}_i(\%)$	**17**	**33**	**46**	**54**	61	68	73	77	81	84

5.6 FastUQ for Aerodynamic Uncertainties due to Surface Variations

To the authors knowledge this is the first work to use MLMF Monte Carlo method for the UQ of surface variations. Therefore, a practical implementation of this approach is provided for reference. The method is shown schematically in fig. 5.11. To begin with the method assumes that a surface disturbance model is available. In this work, the G-PCA modes and amplitudes are extracted from the GP model of the surface variations. The modes are typically obtained for the finest mesh (in the sequence of meshes) to maximise the space of possible perturbed geometries. The selected modes are interpolated from the surface mesh of the finest to all coarse levels. A piecewise-linear interpolation along the cascade surface (see fig. 5.10) is employed in this work to interpolate the PCA modes. The local parametric coordinate t along the cascade surface used for the piecewise-linear interpolation is shown in fig. 5.10. The interpolation ensures that *the samples across levels belong to the same random path* [19].

Independent Gaussian random samples with zero-mean and unit variance are generated for the selected G-PCA modes. One can reconstruct the surface disturbance sample using the weighted summation in eq. (5.8). A parallel fast mesh smoother is used to propagate the surface perturbations into the volume. In this work inverse distance-weighted interpolation (IDW) mesh smoothing is used for this purpose. The parallel

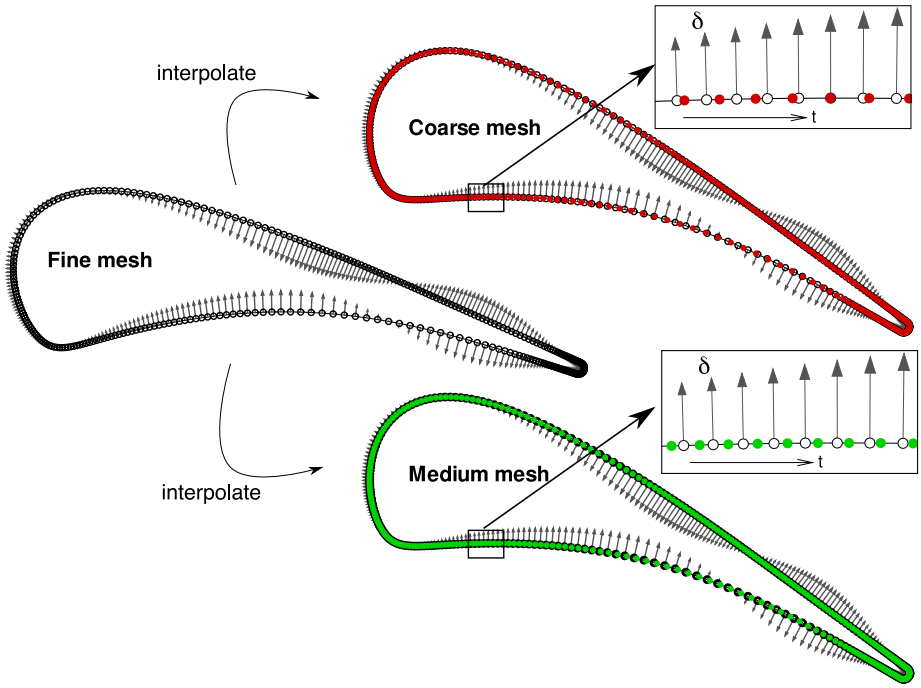


Figure 5.10: Interpolation of PCA mode #4 from the fine to the medium and coarse meshes

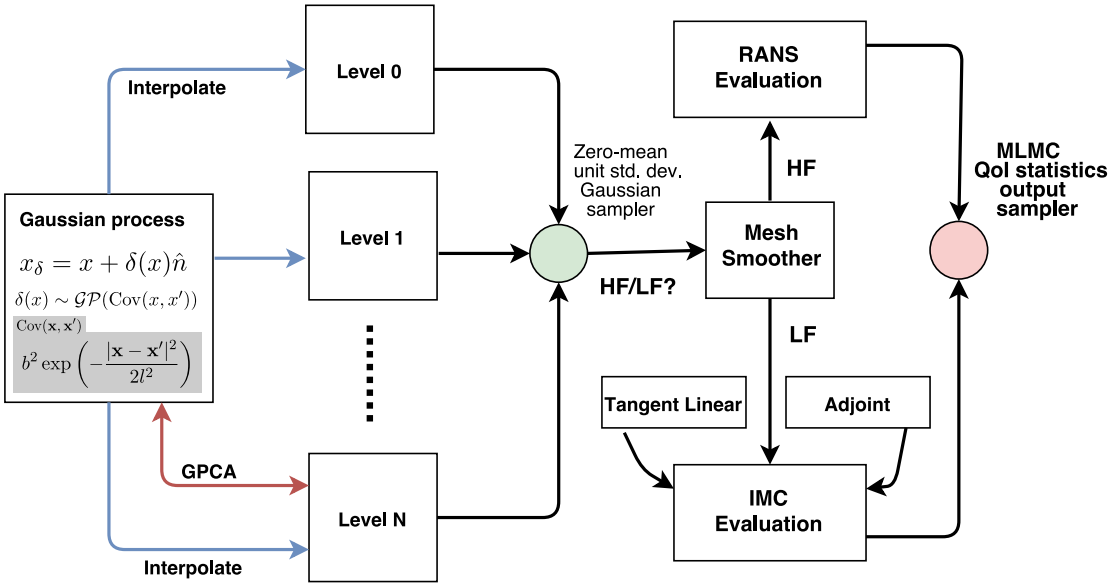


Figure 5.11: FastUQ workflow for surface variations

mesh smoother and its adjoint implementation are shown in Appendix B. The perturbed mesh is then used to run either the HF or LF model to obtain the output QoI. Therefore,

the total computational cost of a sample evaluation is a sum of the computational cost of mesh smoothing and QoI evaluation (using HF or LF model).

The FastUQ sampler runs the initial sampling for the LF and HF model at each level to construct the model correlations and multilevel parameters. Then based on the MSE tolerance the number of additional LF and HF samples are estimated at each level using eq. (3.26). Iteratively samples are added based on the MSE estimates derived in sec. 3.4.1. The implementation of FastUQ in the Dakota toolkit was already presented in sec. 3.5. The workflow presented in sec. 3.5 is adopted while choosing the surface UQ model for the input uncertainties.

5.7 Results and Discussion

Using the multilevel framework and the combination of HF and LF models one can generate a family of UQ methods. A summary of the methods is shown in tbl. 5-D. The IMC versions proposed by Ghate occupies the first row in tbl. 5-D. This is the fastest amongst the four methods summarised in tbl. 5-D but least accurate since it employs only a LF approximate QoI evaluation. The proposed FastUQ method employs both IMC LF (multifidelity) evaluation and multiple levels to accelerate the MC and is shown in the second row. The standard MLMC method (SMLMC) is used as the benchmark model to compared the FastUQ and IMC results. In terms of fidelity this has the highest fidelity amongst the three methods since it does not employ any LF model.

Method	**HF**	**LF**	**Levels**
Incomplete Monte Carlo (IMC)	-	IMC	Single (Finest)
Multilevel Multifidelity Monte Carlo (FastUQ)	RANS	IMC	Multiple
Standard Multilevel Monte Carlo (SMLMC)	RANS	-	Multiple

Table 5-D: Summary of combinations of UQ methods using multiple levels and models

5.7.1 MUR43 and MUR47 Flow Condition

The MUR43 and MUR47 flow conditions are chosen for quantifying the aerodynamic uncertainty due to surface variations. The Mach number contours for the two flow

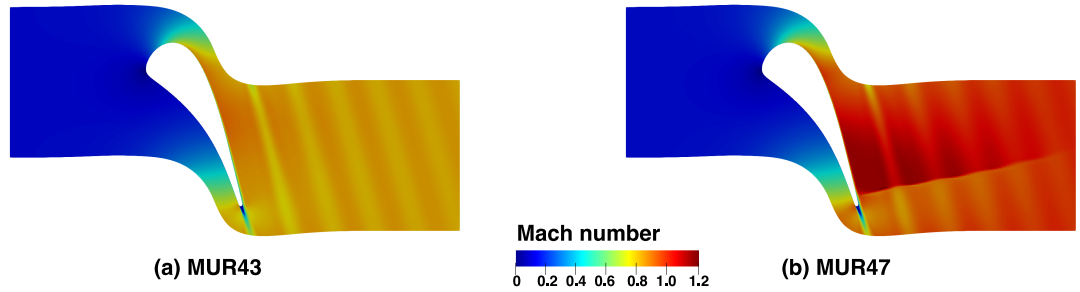


Figure 5.12: Mach number contours for the LS89 cascade at indicated test condition (for the unperturbed mesh)

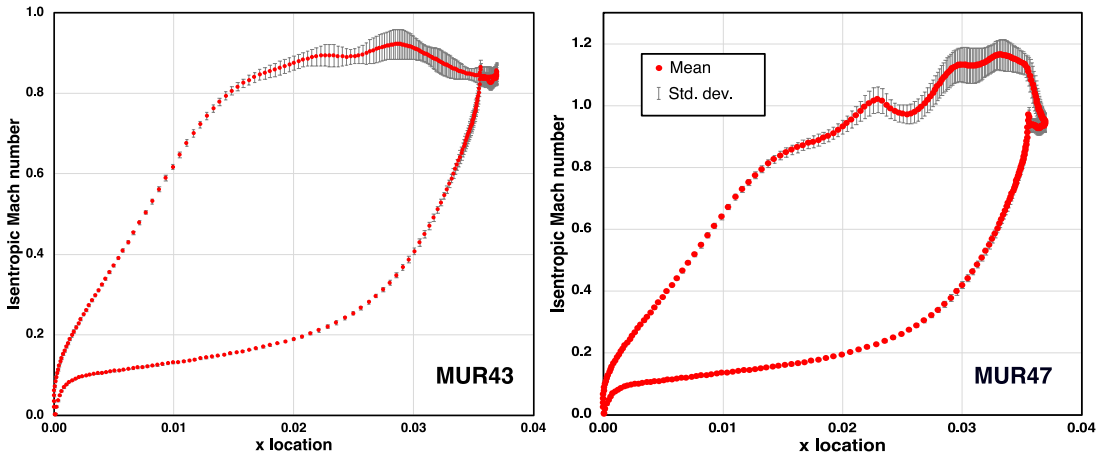


Figure 5.13: Mean and standard deviation of surface isentropic Mach number distribution at indicated flow condition ($\pm\sigma$ shown using error bars) obtained using 20 pilot samples from the surface GP.

conditions are shown in fig. 5.12. MUR47 has a supersonic exit isentropic Mach number of 1.02 and MUR43 has a subsonic exit isentropic Mach number of 0.84. One can notice the presence of a shock wave in MUR47 near the trailing edge of the cascade, which extends all the way into the exit. The total-pressure loss QoI is especially sensitive due to the presence of this shock. The mean and standard deviation in the surface isentropic Mach number is plotted in fig. 5.13. One finds that the surface isentropic Mach number becomes quite sensitive after mid-chord of the cascade geometry on the suction side. The variations are quite high for MUR47 due to the presence of the shock aft of the cascade.

		Coarse	Medium	Fine
Size		40k	60k	90k
Number of MPI ranks		32	32	32
Non-linear flow solver				
Computational Cost (minutes)	**MUR43**	15.0	28.6	85.1
	MUR47	31.8	60.7	171.6
Runtime ratio (w_l)	**MUR43**	0.18	0.34	1.00
	MUR47	0.19	0.35	1.00

Figure 5.14: Computational cost comparison of non-linear solution

5.7.2 Cost model parameters

The SMLMC and FastUQ adjoint-assisted MLMF requires a cost model for the multilevel non-linear flow solution. The computational cost for the baseline (unperturbed) mesh with a residual convergence tolerance of 10^{-11} is tabulated in fig. 5.14. The runtime ratio w_l for each level is also shown in fig. 5.14. Note that the runtime ratio for both MUR43 and MUR47 are quite similar (see fig. 5.14). Hence a fixed runtime ratio of $\{0.2 : 0.35 : 1.0\}$ was used throughout. The computational cost for the baseline geometry (nominal or unperturbed cascade surface) shown in fig. 5.14 is obtained using an uniform flow initialisation. Note that the flow field for every HF sample evaluation in the UQ analysis is initialised using this nominal solution to reduce the number of iterations required for convergence. In addition, the number of iterations to convergence varies depending on the size of the perturbation. If the perturbations are large then more iterations are necessary and vice versa. To simplify the analysis, the runtime ratio w_l is assumed fixed based on the nominal cost and one ignores the variations in computational cost due to difference in the perturbation size.

The cost of the adjoint solution and the relative cost ratio with respect to the HF solution at each level is shown in fig. 5.15. Note that the adjoint solution is significantly cheaper than the non-linear solution because the Jacobian matrix computation and its ILU0 factorisation are performed once during the first iteration and reused for all subsequent iterations. But the non-linear primal requires re-computation of the Jacobian matrix and its ILU0 factorisation for every iteration. The computational cost of the tangent solution

Adjoint (Exit mass flow)				
Computational Cost (minutes)	**MUR43**	3.8	6.9	32.0
	MUR47	3.3	6.5	28.3
Cost ratio (wrt non-linear solution)	**MUR43**	0.3	0.2	0.4
	MUR47	0.1	0.1	0.2
Adjoint (Total pressure loss)				
Computational Cost (minutes)	**MUR43**	2.9	5.5	30.6
	MUR47	2.8	5.1	29.2
Cost ratio (wrt non-linear solution)	**MUR43**	0.2	0.2	0.4
	MUR47	0.1	0.1	0.2

Figure 5.15: Computational cost comparison of adjoint solution

	Tangent solution	
	MUR43	**MUR47**
Computational Cost (minutes)	31.3	43.6
Cost ratio (wrt non-linear solution)	0.37	0.25

Figure 5.16: Computational cost comparison of tangent solution for mode #4 (fine mesh)

for G-PCA mode #4 for MUR43 and MUR47 flow condition is shown in fig. 5.16. Using the cost of the adjoint (see fig. 5.15) and tangent solution (see fig. 5.16) one can estimate the setup cost of the IMC 1/2 model using the eq. (5.12) shown below.

$$\begin{aligned} &\text{Setup cost of IMC 1} = \#\text{QoI} \times \text{AdjointCost} \\ &\text{Setup cost of IMC 2} = \#\text{QoI} \times \text{AdjointCost} + \#\text{Modes} \times \text{TangentCost} \end{aligned} \tag{5.12}$$

Note that the cost model shown in eq. 5.12 is for a single flow condition whose cost of adjoint and tangent linear solution is AdjointCost and TangentCost. #QoI and #Modes in eq. 5.12 denotes the number of QoIs and G-PCA mode included in the analysis.

5.7.3 Effects of G-PCA Truncation

To quantify the effect of G-PCA truncation the UQ of the LS89 cascade for the MUR47 flow condition was carried out using the SMLMC. Two sets of SMLMC simulations were run, (i) using 7 and (ii) 25 G-PCA modes (combining modes based on both QoIs and

flow conditions as shown in sec. 5.5.3). The 7 G-PCA modes capture 50% and the 25 G-PCA modes capture 99% of the total variations in the input uncertainty. Two levels of coarsening of the original mesh was performed maintaining a near wall spacing $y^+ \approx 1-2$. The fine, medium and coarse meshes have approximately 90k, 60k, and 40k nodes respectively. The cost model parameters from sec. 5.7.2 are used in the current SMLMC simulation. For both sets 10 pilot samples were used per level to obtain the initial model correlations. An MSE tolerance of 0.1 was specified for the simulations. The mean and standard deviation obtained from the simulation are tabulated in tab. 5-E. Inclusion of more modes does not dramatically alter the mean value of the exit mass flow rate but the mean total-pressure loss is quite sensitive. Although the variation in the mean of the QoI (between 7 and 20 modes) is $< 2\%$ the standard deviation has a high variation in the QoI (around 36%). Therefore, all simulations henceforth use 25 G-PCA modes since it captures 99% of G-PCA variation and the predicted variations with 7 modes differed significantly from the 25 G-PCA modes.

# G-PCA modes	**Exit mass flow**	**Total-pressure loss**
7 modes	$\mathbf{4.62}604\ (\pm\mathbf{2}.07639 \times \mathbf{10^{-2}})$	$\mathbf{1.8}5356\ (\pm\mathbf{1}.07648 \times \mathbf{10^{-1}})$
25 modes	$\mathbf{4.62}286\ (\pm\mathbf{2}.64565 \times \mathbf{10^{-2}})$	$\mathbf{1.8}9269\ (\pm\mathbf{1}.48167 \times \mathbf{10^{-1}})$

Table 5-E: SMLMC mean and standard deviation of indicated cost function obtained using 7 and 25 dominant (combined) G-PCA modes for the MUR47 test case (MSE tolerance 0.1)

Lange et al. [18] considered blade measurements from 3-D surface scan data of a high pressure compressor stage. The correlated coordinates from the 3-D scan were translated into mode shapes and uncorrelated modal amplitudes using PCA. Lange et al. included all 135 PCA modes in their perturbation model within the MC simulations and compared it with MC results considering 20 and 60 PCA modes. Lange [18] concludes that 60 PCA modes (half of the total number of PCA modes) were necessary to correctly model the manufacturing variability on the blades. Inclusion of 25 G-PCA modes of the LS89 cascade already increases the input uncertainty space quite dramatically. But a large number of input uncertainties can be handled using FastUQ with IMC 1 LF model (see

sec. 3.2) and the G-PCA can help truncate the statistically independent PCA modes that do not contribute to the change in QoI. Therefore, this suggests that FastUQ with IMC 1 LF model and G-PCA based truncation are well suited methods for the problem under consideration.

5.7.4 Effect of Pilot Sampling

The accuracy of the variance estimate at each level depends on the size of the initial samples set considered, which are also called the pilot samples. One needs to ensure that the bias error in the initial sampling is minimal because the estimation of the bias error is not considered in SMLMC [19]. Giles [19] suggests that the bias error be made proportional to ϵ^{-p}, where p is bound by $0 < p < 2 - \frac{1}{\alpha}$ and for a second order spatial discretisation $\alpha \approx \frac{1}{2}$. To test the bias error in the pilot sampling, the number of pilot samples was doubled from 10 to 20 and the change in QoI for an MSE tolerance (ϵ) of 0.1 was recorded for the SMLMC and is shown in tbl. 5-F. The results indicate that the change in the mean and standard deviation of QoI between the two pilot samples satisfies the aforementioned bound. Therefore, the 10 pilot sample run was used for all subsequent calculations. Note that the bias error and the initial sampling of MLMC demands further investigation. Adopting the CMLMC method can help resolve this problem but it has been postponed to a future study (see ch. 6).

# pilot samples	**Exit mass flow**	**Total-pressure loss**
10	$\mathbf{4.622}86\ (\pm\mathbf{2.64}565 \times \mathbf{10^{-2}})$	$\mathbf{1.892}69\ (\pm\mathbf{1.48}167 \times \mathbf{10^{-1}})$
20	$\mathbf{4.622}67\ (\pm\mathbf{2.64}368 \times \mathbf{10^{-2}})$	$\mathbf{1.892}74\ (\pm\mathbf{1.48}438 \times \mathbf{10^{-1}})$

Table 5-F: QoI statistics with different pilot samples in SMLMC (MUR47 condition with MSE tolerance 0.1)

5.7.5 Optimal Resource Allocation and MSE Convergence (SMLMC)

The SMLMC using 25 G-PCA modes was run for four MSE tolerances, 1.0, 0.1, 0.05, and 0.01 respectively for the MUR47 flow condition. The resource allocation between the three mesh levels (coarse, medium, and fine) to achieve the aforesaid MSE tolerances

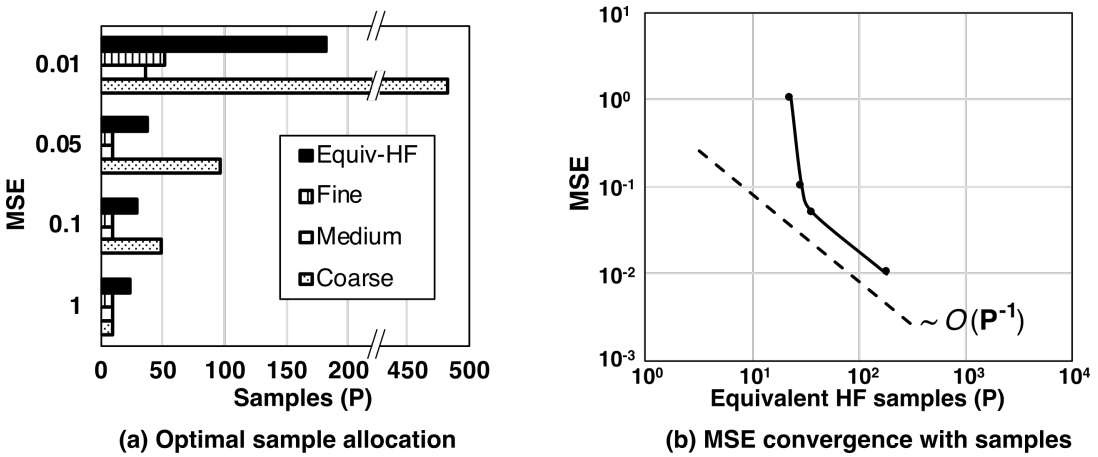


Figure 5.17: SMLMC results for the LS89 test case MUR47; (a) optimal sample allocation based on eq. (3.26) and (b) MSE convergence.

is shown in fig. 5.17 (a) (based on eq. (3.26) for optimal resource allocation). Results in fig. 5.17 (a) show that all computations required sample evaluation only on the coarsest mesh up to an MSE tolerance of 0.05 (excluding the 10 pilot samples at each level). This is primarily due to well correlated values of the QoI between levels i.e, the gross variations are captured by the coarse levels. This is an important requirement for multilevel methods [24]. The mean and standard deviation values obtained for the various MSE error tolerance are tabulated in tbl. 5-G. In tbl. 5-G one observes that for an MSE tolerance of 0.01 more equivalent HF samples are necessary to capture the finer details of the variations. The equivalent HF samples are still cheaper to evaluate because most of the samples are computed from the coarse level solution (see fig. 5.17(a)). The convergence of the MSE with equivalent HF samples P is plotted in fig. 5.17 (b), where the dotted trend-line indicates the $\mathcal{O}(P^{-1})$ convergence. The initial convergence is sharp for MSE<0.05 but for larger MSE>0.05 the convergence matches $\mathcal{O}(P^{-1})$.

MSE	**Exit mass flow**	**Total-pressure loss**	**Eqv. HF**
1.0	$\mathbf{4.62}043\ (\pm 3.31661 \times \mathbf{10^{-2}})$	$\mathbf{1}.90907\ (\pm \mathbf{1}.92868 \times \mathbf{10^{-1}})$	23
0.1	$\mathbf{4.62}286\ (\pm \mathbf{2}.64565 \times \mathbf{10^{-2}})$	$\mathbf{1.89}269\ (\pm \mathbf{1}.48167 \times \mathbf{10^{-1}})$	29
0.05	$\mathbf{4.62}339\ (\pm \mathbf{2}.43004 \times \mathbf{10^{-2}})$	$\mathbf{1.89}003\ (\pm \mathbf{1}.53320 \times \mathbf{10^{-1}})$	37
0.01	$\mathbf{4.62}408\ (\pm \mathbf{2}.42732 \times \mathbf{10^{-2}})$	$\mathbf{1.89}980\ (\pm \mathbf{1}.47926 \times \mathbf{10^{-1}})$	182

Table 5-G: SMLMC mean and standard deviation of indicated cost function obtained using 25 dominant (combined) G-PCA modes for the MUR47 test case

5.7.6 Accuracy of Adjoint Correction

To test the accuracy of the IMC LF model the aerodynamic UQ of LS89 cascade at the MUR47 condition for the two QoIs, namely, exit-mass flow and total-pressure loss was carried out. IMC 1 LF model requires only a single adjoint solution per QoI and its computational cost is independent of number of PCA modes (see sec. 3.2). For each level the IMC 2 LF model requires as many tangent linear solutions as the number of G-PCA modes (in addition to the cost of the adjoint solution). Note that the above estimates are the number of solution evaluations to construct the LF model per level in the case of FastUQ and it is a one time cost associated with the LF model setup.

The results presented in sec. 5.7.3 showed that 25 G-PCA modes were necessary to capture 99% of the input variations and using lower number of G-PCA modes gave large variations in the standard deviation of the QoI for the MUR47 test condition. The setup cost of the IMC 2 LF model for 25 modes is 17 equivalent HF samples (calculated based on eq. (5.12)). This setup cost is in fact greater than the computational cost in equivalent HF samples for the entire IMC 1 simulation for an MSE tolerance of 0.01 (see tbl. 5-H). Therefore, for large number of input uncertainties IMC 2 setup cost becomes quite high. In addition, the MUR47 condition has a shock discontinuity in the solution. This reduces the regularity of the solution, which introduces large errors in the higher order adjoint corrections [1, 29, 2] (also see sec. 3.3.3) Therefore, the analysis using IMC 2/3 models is postponed to a future study and only the IMC 1 LF model is considered in all simulations.

The results of the IMC 1 are compared against the SMLMC results in tab. 5-H for the same prescribed MSE tolerance of 0.01. Note that the sample allocation per level for SMLMC is shown in fig. 5.17(a) and the IMC 1 is run on the finest mesh level. The equivalent HF samples for the IMC 1 simulation is obtained by multiplying the total runtime of the IMC 1 simulation with the ratio of the SMLMC equivalent HF samples to the SMLMC runtime (for the prescribed MSE tolerance).

Type	**Exit mass flow**	**Total-pressure loss**	**Eqv. HF**
SMLMC	$\mathbf{4.62}408\ (\pm\mathbf{2.4}2732 \times 10^{-2})$	$\mathbf{1.}89980\ (\pm\mathbf{1.}47926 \times \mathbf{10^{-1}})$	182
IMC 1	$\mathbf{4.62}367\ (\pm\mathbf{2.4}1368 \times 10^{-2})$	$\mathbf{1.}77778\ (\pm\mathbf{1.}52438 \times \mathbf{10^{-1}})$	12

Table 5-H: Mean and standard deviation of indicated cost function obtained using 25 dominant (combined) G-PCA modes for the MUR47 test case (SMLMC and IMC 1)

Two important observations can be made from the results shown in tbl. 5-H. Firstly, the IMC 1 takes only a fraction of the cost of the SMLMC ($\approx \frac{1}{10}$) for the same MSE tolerance. The SMLMC took close to 27h to complete whereas the IMC took 1.7h. Secondly, the total-pressure loss (mean and variance) obtained using IMC 1 has large deviations from the SMLMC results but the deviation of the exit mass flow rate QoI is lower. *This clearly shows the shortcoming of the IMC 1 model for sensitive and non-linear QoIs, which can be remedied using the FastUQ method.*

5.7.7 FastUQ Results

The performance of the multilevel multifidelity FastUQ method described in sec. 5.6 is a strong function of the correlation between the HF and IMC LF model (see eq. (3.26)). Therefore, the correlation at each level for the MUR43 and MUR47 test case is plotted in fig. 5.18 (obtained from the 10 pilot samples). One finds that the correlation is quite high for both MUR47 and MUR43 test conditions (average correlation for both cases is ≥ 0.95). But the coarse level in MUR43 has a lower correlation value of 0.76. Therefore, one can expect more HF sample evaluation at the coarse level for MUR43 to compensate for the lower correlation. This is clearly reflected in the resource allocation shown in fig. 5.22, where 151 additional HF samples are evaluated for the MUR43.

The QoI statistics for both MUR43 and MUR47 test conditions were computed using all three UQ methods for an MSE tolerance of 0.01 and the results are summarised in fig. 5.19-5.21. The computational cost reduction in comparison to the runtime of the SMLMC achieved by FastUQ and IMC are shown in fig. 5.19. The percentage variation in the mean and standard deviation from SMLMC results for FastUQ and IMC are shown in fig. 5.20 and 5.21. The computational cost comparison in fig. 5.19 shows that

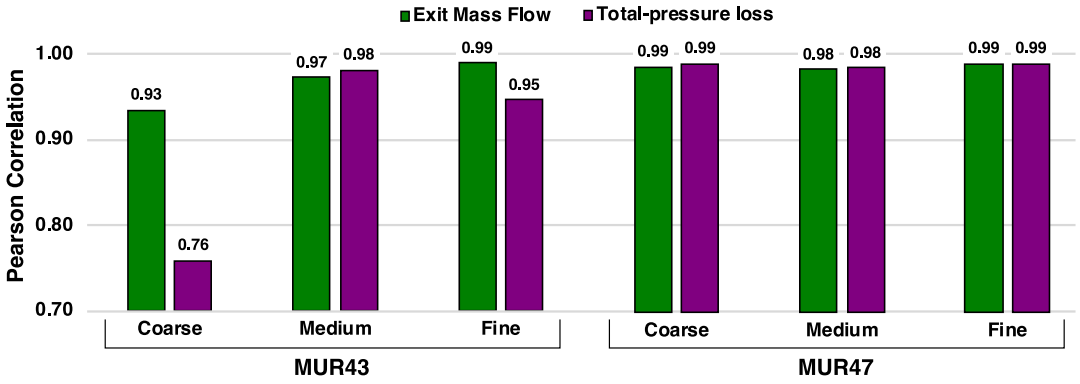


Figure 5.18: Pearson correlation between HF and LF model for each mesh level (MUR43 and MUR47 test condition and indicated QoI)

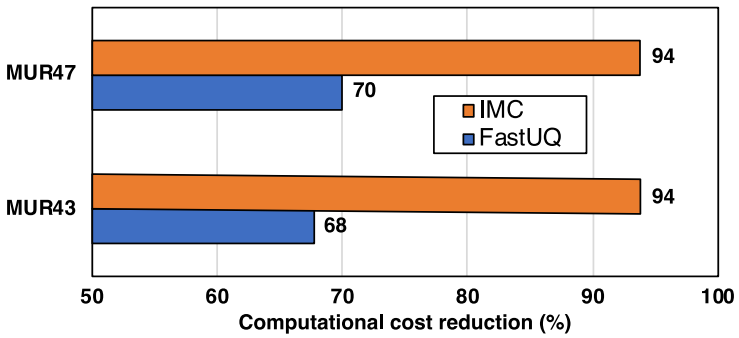


Figure 5.19: Percentage reduction in computational cost compared to SMLMC results.

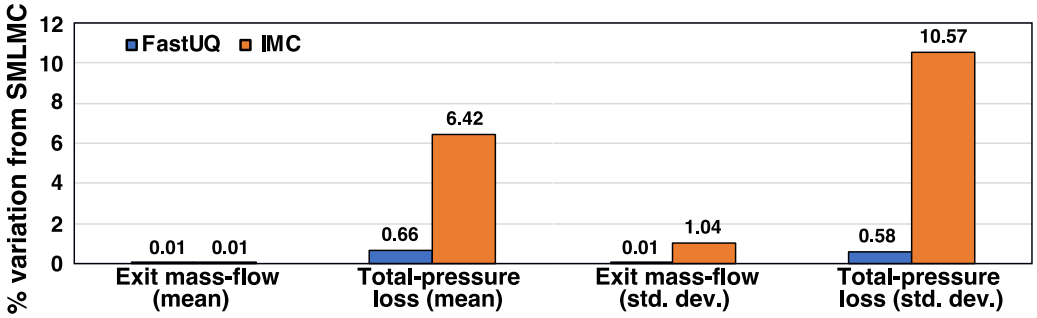


Figure 5.20: MUR47 mean and standard deviation variation with respect to SMLMC results

FastUQ gives $\approx 70\%$ reduction in computational cost over SMLMC and the variations in mean and standard deviation of the QoI are within 1% variations of the SMLMC results. IMC achieves an additional 25% reduction compared to FastUQ. Although the mean and standard deviation for the exit mass flow rate QoI is captured within 1% tolerance, IMC

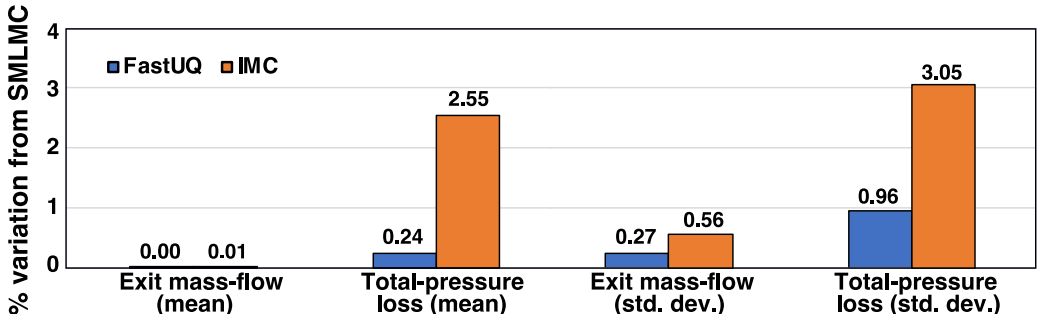


Figure 5.21: MUR43 mean and standard deviation variation with respect to SMLMC results

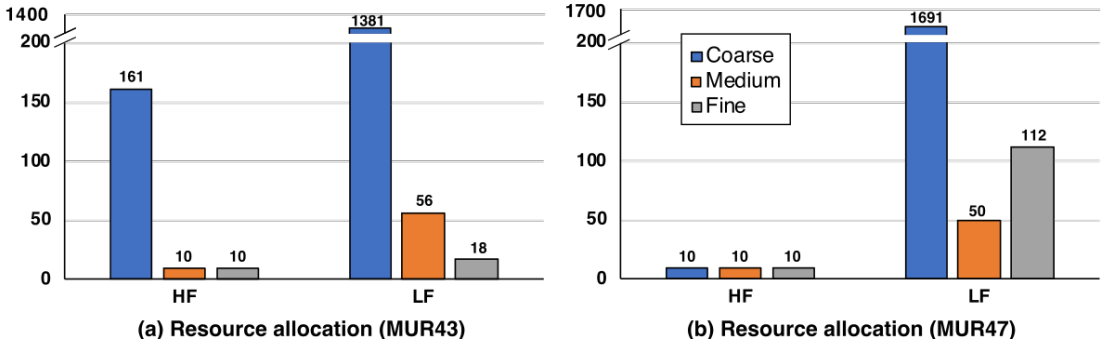


Figure 5.22: Computational resource allocation across levels and models

has significantly large deviations for the total-pressure loss QoI. Variations as large as 10% can be observed in the standard deviation and 6% in mean values.

FastUQ on the other hand uses the LF model only in regions where it is applicable and resorts to using the model correlation or running the HF model when the LF model has large deviations (lower correlation). This behaviour can be confirmed using the FastUQ resource allocation shown in fig. 5.22. Since the model correlation for MUR47 test case are higher (in the range 0.98-0.99 show in fig. 5.18) more LF samples are evaluated using the IMC 1 model and the correlation is used to correct the QoI. On the contrary the correlation for the coarse mesh is lower in MUR43 so additional HF samples are used to compensate this error.

5.8 Summary

In this chapter the FastUQ method was used to obtain the uncertainty in aerodynamic performance, namely total-pressure loss and exit mass flow, due to surface variations caused by manufacturing process. A Gaussian process model for surface variation was proposed and goal-based PCA was used to truncate and reduce the number of input surface uncertainties. The method was demonstrated on the LS89 turbine cascade for two sensitive flow conditions. When UQ analysis for multiple flow conditions and QoIs are desired the number of surface G-PCA modes required to capture the input variations can increase significantly. In this study, 25 G-PCA modes were necessary to capture the variations in the QoIs for the cascade. In additions, the goal-based truncation was necessary since not all dominant PCA modes were actually dominant in the G-PCA sense.

The performance and accuracy of FastUQ was compared with SMLMC and IMC 1 methods. FastUQ is similar in accuracy to SMLMC but the computational cost is 70% lower than SMLMC. FastUQ does not suffer from the loss of accuracy compared to the IMC 1 since the control variate corrects the approximate LF evaluations using model correlations and resorts to using HF samples when model correlations are low.

Chapter 6

Summary and Recommendations

> *"Oh! Arjuna surrender the fruits of your action to me and do not worry; you are neither the doer nor the cause ..."*
>
> — Shri Krishna, Bhagavathgeetha

In this thesis an adjoint-based multilevel multifidelity Monte Carlo method was proposed, analysed, and used to quantify uncertainties in aerodynamic parameters due to surface variations caused by manufacturing process. The following is a recapitulation of the thesis.

- A survey of uncertainty quantification methods was presented in ch. 2. Convergence of methods based on Monte Carlo (MC) sampling was found to be independent of number of input uncertainties while methods based on generalised polynomial chaos expansion (gPCE) and Stochastic Collocation (SC) was shown to suffer from the curse of dimensionality. But gPCE and SC gives $\mathcal{O}(P^{-2})$ rate of convergence (with samples P) but the MC suffers from the slow convergence rate $\mathcal{O}(P^{-0.5})$ of the Root Mean Square Error (RMSE). Improvements to both methods target these deficiencies. The multilevel and multifidelity Monte Carlo methods were motivated using the control variate framework. Control variate (CV) accelerates the MC by employing an approximate evaluation of the QoI (in addition to the original estimator also called the high-fidelity (HF) model). If the approximate or low-fidelity model (LF) is cheap to evaluate and has a reduced variance then one

can prove that a linear correction to the estimator (see eq. (2.23)) improves the predicted output with lower equivalent high-fidelity (HF) samples. In addition, high correlation between the HF and LF models was identified to be crucial for the success of CV. The reduction factor due to MFMC compared to MC ϕ was derived as an extension to the multifidelity Monte Carlo (MFMC) framework of Ng et al [3].

- The multifidelity control variate requires one to carefully choose the LF model. The LF model should not destroy the favourable property of MC i.e., a convergence independent of number of input uncertainties. The IMC approximate QoI evaluation using adjoint correction [1] was proposed as a LF model in ch. 3. The multifidelity framework was analysed based on the work of Ng et al. [3] and demonstrated on the viscous Burgers' equation analytical model with uncertain boundary condition. The multifidelity reduction factor ϕ derived in ch. 3 was validated using the Burgers' model problem. The framework was extended to the multilevel multifidelity (MLMF) framework of Geraci et al. [4]. The main result of the analysis presented in the thesis is the link between the multifidelity reduction factor ϕ and the the MLMF reduction factor Λ. The factor $\phi \geq \Lambda$ (see eq. (3.29)), which guarantees improvement of MF when introduced in a multilevel framework. Another insight from the analysis was the strong dependence of the computational reduction on the correlation between HF and LF model. Implementation of the proposed multilevel multifidelity FastUQ framework in the Dakota toolkit was presented.

- The high fidelity (HF) model used in this work is a parallel solver based on a second order edge based finite volume discretisation of the RANS equation. A two-halo layer (THL) partitioning algorithm was used to parallelise the flow solver. The THL reduced the total communication by half with negligible increase in the number of redundant computations compared to a single-halo layer partitioning (SHL) approach. Moreover the partitioning simplified the adjoint and tangent linear solver development by eliminating MPI call within the residual routine compared

to the zero-halo layer (ZHL) or the SHL approach. The non-linear solver was validated against external and internal flow test cases. The gradients obtained from adjoint and tangent linear solution compared well with the finite-difference counterpart.

- A probabilistic model for geometric perturbations based on Gaussian Process (GP) was proposed in ch. 5 using a square exponential correlation function. The correlation length and height were made problem-specific parameters, which a user has to specify. A procedure to separate statistically independent modes and to truncate them considering multiple QoIs using the adjoint effectiveness factor η_{PCA} was proposed. The FastUQ framework was used to obtain the statistics of the QoIs, namely total-pressure loss and exist mass flow for the VKI LS89 turbine test case subject to the proposed geometric disturbance model at two sensitivity flow conditions. The G-PCA truncation was performed and its effect on the QoI statistics was analysed. The results showed that modes dominant in PCA can be ineffective in goal-based measure and can differ greatly between QoIs. In addition, significant increases in G-PCA modes after truncation was observed when one considers multiple flow conditions and QoIs. The FastUQ results were compared against the SMLMC one for the LS89 cascade subject to the truncated G-PCA surface perturbation GP model. No significant loss of accuracy was observed but a 70% reduction in computational cost was achieved. Results from IMC 1 LF model (from sec. 3.2) was compared with FastUQ. Differences close to 10% in standard deviation and 6% in mean was observed for the IMC 1 due to its approximate nature. Overall the FastUQ has lower computational cost for maintaining similar solution accuracy as SMLMC. In addition the use of IMC 1 LF model in FastUQ retains the independence of convergence to number of input uncertainties.

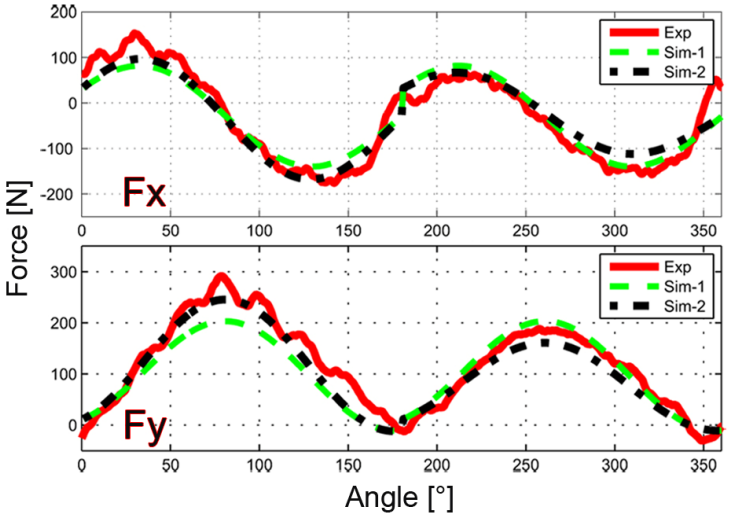


Figure 6.1: Simulation and experimentally determined tool cutting force (from ref. [7])

6.1 Recommendations For Future Work

6.1.1 Model Improvements

The proposed surface disturbance model for geometric variations is synthetic. A more realistic model should be based on empirical blade measurements [18]. Another approach to modelling surface perturbations due to machining that has not received much attention is the use of techniques already available in the field of CAD/CAM, namely tool path design and tolerancing [7, 173]. Any machining system works on removing or adding material using a tool to manufacture a given design. The characteristics like tool vibration, cutting forces, material property of the working material, etc. play a pivotal role in determining the final finished surface. For example the assumed and measured cutting forces in a steel ball-end tool from ref. [7] is shown in fig. 6.1. Therefore, there are many deterministic parameters that are ignored due to an aleatoric modelling of the uncertainty. Practically the problem consists of both epistemic and aleatoric uncertainties. Approaching the problem from the perspective of manufacturing and tolerancing [165] can bring more insight into the UQ analysis and on the final design optimisation of the blades. In fact one can link CAM and tool path via CAD into the design.

6.1.2 MLMC improvement

The continuation MLMC proposed by Pisaroni et al. [21] solves the problem of pilot sampling using a sequence of decreasing tolerances and an online algorithm to obtain the problem dependent rates and constants of the SMLMC. CMLMC reduces the ambiguities with bias error in the initial sampling of the SMLMC. Currently, the Dakota toolkit does not support CMLMC and this can be a useful addition if implemented in Dakota.

6.1.3 Comparison with other UQ methods

In the present work the full MC simulation of the LS89 turbine cascade was not carried out. Therefore, a comparative study with plain MC in combination with other sampling techniques like LHS and QMC is proposed. In addition, comparison of accuracy and computational cost with gPCE and SC is proposed to be carried out for the surface UQ problem.

6.1.4 Better than $\mathcal{O}(P^{-1})$ Convergence

Recently Peherstorfer [174] proposed an adaptive goal-based model selection framework for multifidelity control variates. Here multiple LF models are used to accelerate the control variate, where the coarse grid solutions are also considered as LF models. The author proves that the method has a convergence rate $\mathcal{O}(P^{-1-\alpha})$, where P is the number of samples and α is a positive quantity compared to the usual $\mathcal{O}(P^{-1})$ convergence of multilevel methods.

6.1.5 Solver Improvements

Although the present solver is quite robust it is not robust enough to be applied to large perturbations of the surface. A full Newton-Krylov solver with pre-conditioner based on the the second order Jacobian matrix has strong stability properties. Using AD and graph colouring approach [175] it is simple to construct the full second order matrix. Use of a scalable GMRES and pre-conditioner algorithm as recommend in ch. 4 should help the scalability of the solver to large test cases.

6.1.6 New application areas

The FastUQ method can be applied to new problem areas like robust optimisation and risk based design under manufacturing uncertainties by including importance sampling [43, 15, 176]. The computational gain can significantly improve such workflows. The IMC method can be used as an outline or approximate method in the robust optimisation. The FastUQ can then be used after a rough initial design has been obtained using IMC.

Appendix A

Author's publications

Journal papers

1. Mohanamuraly P., Hascöte L., and Müller J. D., "*Seeding and adjoining zero-halo partitioned parallel scientific codes*", Journal of Optimization Methods and Software, 2019.

2. Mykhaskiv O., Banovic M., Auriemma S., Mohanamuraly P., Walther A., Legrand H., Müller J., "*NURBS- based and Parametric-based Shape Optimisation with differentiated CAD Kernel*", Computer-Aided Design and Applications, 2017.

Conference papers

1. Mohanamuraly P., Müller J., "*Adjoint-assisted multi-level approach for quantifying geometry-induced uncertainties in robust aerodynamic shape optimization*", 2nd UNCECOMP, ECCOMACS congress, 2017.

2. Mykhaskiv O., Mohanamuraly P., Müller J., Xu S., Timme S., "CAD-based shape optimisation of the NASA CRM wing-body intersection using differentiated CAD-kernel", 35th AIAA applied aero. conf., 2017.

Appendix B

Inverse Distance Weighted Interpolation Mesh Smoothing and Adjoint

Mesh deformation, smoothing or morphing is a critical aspect of the present UQ analysis. Firstly, the quality of the mesh smoother affects the numerical solution. Secondly, the computational cost of mesh smoothing adds to the cost of UQ. Lastly, the IMC model requires the adjoint of the mesh smoothing algorithm. Therefore, an ideal mesh smoother should satisfy the following key criteria,

- Computationally efficient and easily parallelisable
- Zero or minimal deterioration of mesh quality post-smoothing
- Easily and efficiently adjoinable (transpose)

The aim of a mesh smoother is to generate a map $\mathbf{A}$ that projects the displacement field $\delta\mathbf{x}_b$ defined on a boundary $\partial\Omega$ onto a displacement field $\delta\mathbf{x}$ defined over the volumetric domain Ω as shown in eq. (B.1). The map $\mathbf{A}$ is expected to satisfy the aforementioned

criteria for an ideal smoother. The mesh smoothers available in literature can be classified into two broad categories, namely, (i) methods involving solution to linearised systems (SLS), and (ii) direct or algebraic methods (DAM).

$$\underbrace{\delta\mathbf{x}_b}_{\partial\Omega} \xrightarrow{\mathbf{A}} \underbrace{\delta\mathbf{x}}_{\Omega} \tag{B.1}$$

The methods based on the solution to linearised system usually involves the inversion of a linear system to obtain the final smoothed deformation field as shown in eq. (B.2). $\mathbf{M}$ is a non-square permutation matrix to match up the dimensions of $\partial\Omega$ and Ω.

$$\delta\mathbf{x} = \mathbf{A}^{-1}\mathbf{M}\delta\mathbf{x}_b \tag{B.2}$$

In contrast, DAM does not require inversion of a matrix system but involves a pure matrix multiplication as shown in eq. (B.3). But the matrix $\mathbf{A}$ in eq. (B.3) is usually a dense rectangular matrix and one in eq. (B.2) is a sparse square one.

$$\delta\mathbf{x} = \mathbf{A}\delta\mathbf{x}_b \tag{B.3}$$

Mesh smoothing based on linear or torsional spring model and linear elasticity are typical examples of smoothing base on SLS. Linear elasticity is a robust mesh smoothing algorithm for large deformation (see ref. [177, 178, 179, 180]). Yang and Mavriplis [179] derived the adjoint of the linear elasticity and used the same to optimise the Young's modulus to yield mesh with minimum skewness. But the authors report that the adjoint is computational expensive. Xu et al. [181] used linear elasticity for mesh smoothing and an inexpensive linear spring model to obtain the adjoint. Although this is an attractive alternative it brings inconsistencies and uncertainties into the workflow. The primary aim in the present work is to have a consistent mesh smoothing and sensitivity projection (adjoint mesh smoothing). In addition, the MLMC requires generation of a large number of deformed mesh samples. Therefore, it is necessary to keep the computational cost as low as possible.

Inverse Distance Weighted (IDW) interpolation is an explicit technique for multivariate interpolation of scattered data points. It was first introduced by Shepard within the framework of geographical information systems [182]. IDW as a DAM type mesh smoother was first proposed by Witteveen [183]. The author observed that to achieve good mesh quality it is necessary to also interpolate the rotation. But this destroys the extremum conserving property of IDW i.e, internal nodes could move outside of the domain. To prevent this situation various checks are usually added to the algorithm, which are summarised in ref. [184].

The implementation of the IDW smoother is fairly straightforward since no mesh connectivity information is necessary. Therefore, arbitrary mesh topologies, and hanging nodes, can be deformed without any modifications to the algorithm. In addition, the internal node in IDW only depends on the boundary node deformation which simplifies parallelisation. Luke and co-workers [185] used a fast multipole method (FMM) to speed-up the dense matrix-multiplication in IDW. In addition, the authors demonstrated the distributed parallelisation of the FMM based IDW smoother. Uyttersprot [184] gives a detailed comparison of IDW with other mesh smoothers for various types of meshes and deformation fields. The author concludes that IDW is comparable to linear elasticity and radial basis function (RBF) based mesh smoothers. In addition, the author provides an implementation of the IDW including local rotations and mesh sliding.

B.1 Mesh Smoothing Implementation

The basic kernel in IDW is eq. (B.4), where a volumetric field $\phi(\mathbf{x})$ is obtained from a boundary field $\phi_b(\mathbf{x})$ based on an inverse distance weighting kernel $w_b(\mathbf{x})$ defined in eq. (B.5). Luke [185] suggests a value for the power parameters $a = 3$ and $b = 5$ based on their numerical tests. The parameter L_{def} is chosen to be the furthest distance from any mesh node to the centroid of the mesh. The parameter α controls the weighting of near boundary nodes over more distant ones. Uyttersprot [184] recommends to use a value of $\alpha \geq 0.1$ to guarantee good near-boundary mesh quality. The parameter A

is the boundary face area weighting and Uyttersprot [184] recommends to ignore this parameter (i.e, $A = 1$). The OpenMP parallel C++ code of the IDW kernel is shown in listing 1. The `norm2` function is the $L2$ norm function and Cilk array notations [186] are used to concisely represent the code.

$$\phi(\mathbf{x}) = \frac{\sum_{b=0}^{n_b} w_b(\mathbf{x} - \mathbf{x}_b)\phi_b(\mathbf{x})}{\sum_{b=0}^{n_b} w_b(\mathbf{x} - \mathbf{x}_b)} = \frac{p}{q} \tag{B.4}$$

$$w_b(\mathbf{x} - \mathbf{x}_b) = A\left[\left(\frac{L_{def}}{||\mathbf{x} - \mathbf{x}_b||}\right)^a + \left(\frac{\alpha L_{def}}{||\mathbf{x} - \mathbf{x}_b||}\right)^b\right] \tag{B.5}$$

```
/* Constants A, a, b, LDEF, ALPHA assumed pre-defined */
void PrimalKernelIDW
( const int n,           const int nb,
  const float x[][3],    const float xb[][3],
  float dx[][3]/* in */, float dxb[][3]/* out */ )
{
  #pragma omp parallel for
  for ( unsigned i = 0; i < n; ++i ) {
    float p[1:3] = 0.0;
    for ( unsigned j = 0; j < nb; ++j ) {
      float temp = 1.0 / norm2( x[i][1:3] - xb[j][1:3] );
      float wb   = A * ( pow( LDEF * temp, a ) +
                         pow( ALPHA * LDEF * temp, b ) );
      p[1:3] += wb * dxb[j][1:3];
      q      += wb;
    }
    dxb[i][1:3] = p[1:3] / q;
  }
}
```

Listing 1: OpenMP parallel C++ code for the IDW kernel

Boundary deformation for two test cases is shown in fig. 2.1(a-b). In the first test case (fig. 2.1(a)) two types of deformation field are imposed on the square hole inside a plate (butterfly mesh topology). The deformation shown in red deforms the square to a circular hole and the one in green rotates the square by 10^o in anti-clockwise direction. Note that the mesh is stretched towards the square hole. The deformation results from linear

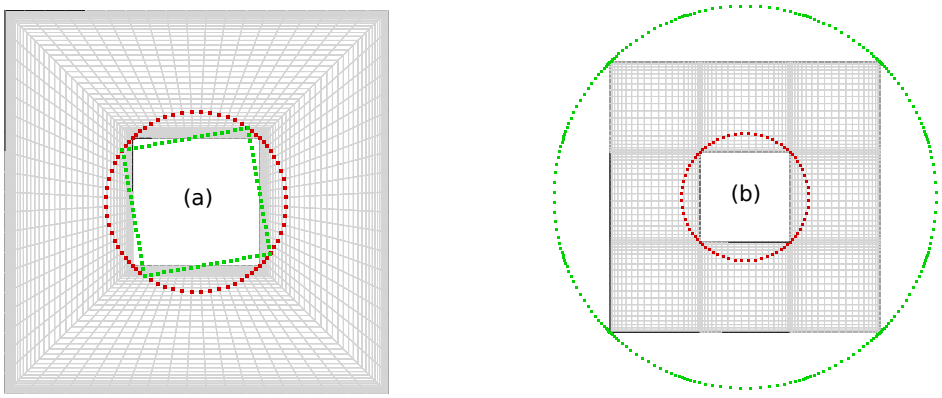


Figure 2.1: Two test cases for mesh smoothing (a) square hole in a butterfly mesh topology transforming to a circle (red) and rotation by 10^o anti-clockwise (green) (b) square hole in cartesian mesh topology transforming to a circle (red) and outer square boundary transforming to a circle (green)

spring, linear elasticity, and IDW mesh smoother is shown in fig. 2.2. The IDW performs on par with linear elasticity and the linear spring generates highly skewed elements near the hole. This is usually remedied by including rotations. Since no such rotations were used in the IDW it is fair to make this comparison.

A similar result is obtained for the second test case (fig. 2.1(b)), where the first deformation field (in red) transforms the square hole (in a cartesian mesh topology) into a circular one and the second (in green) transforms the outer plate boundary to a circle. The total serial runtime (of core numerical kernel) to generate the deformed meshes for the three mesh smoothing methods are tabulated in tab. 2-A. IDW outperforms the other two mesh smoothers and the quality of the deformed mesh is comparable to the linear elasticity results.

Linear Spring	Linear Elasticity	IDW
2.3s	57.1s	1.5s

Table 2-A: Serial runtime comparison between various mesh smoothers (linear solver convergence tolerance of linear elasticity and spring set to 10^{-10})

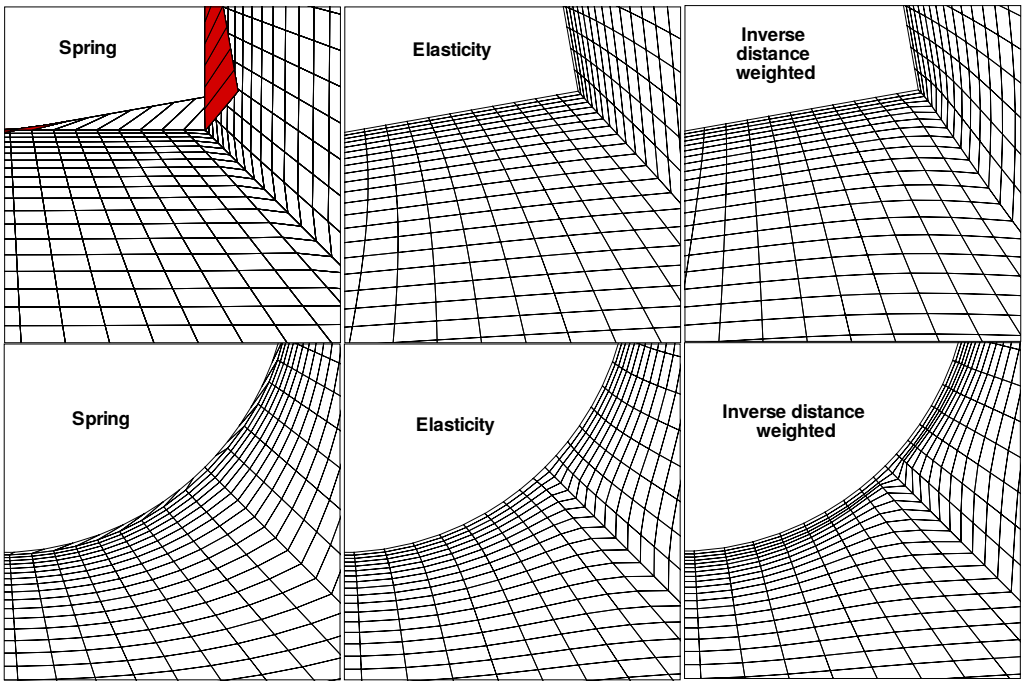


Figure 2.2: Results of the boundary deformation define in fig. 2.1(a) for the linear spring, linear elasticity and IDW (zoomed near the corner and inverted elements marked in red)

B.2 Adjoint of Mesh Smoother

The IDW is a simple mathematical kernel and hence it is simpler to manually derive the adjoint rather than using AD tools. Moreover, manual adjoining results in a more efficient code (free from push/pop). The form of IDW shown in eq. (B.5) is a matrix-free from and it is first necessary to bring that into matrix form. Let the superscript on $\mathbf{x}^j$ denote a boundary node and subscript on $\mathbf{x}_i$ denote an internal volume node and rewriting eq. (B.5) in this new notation one obtains eq. (B.6).

$$\delta\mathbf{x}_i = \frac{\sum\limits_{j\in\partial\Omega} w(\mathbf{x}_i - \mathbf{x}^j)\delta\mathbf{x}^j}{\sum\limits_{j\in\partial\Omega} w(\mathbf{x}_i - \mathbf{x}^j)} = \sum_{j\in\partial\Omega} a_i^j\,\delta\mathbf{x}^j = \underbrace{\left[a_i^j\right]_{n_b\times n}}_{\mathbf{A}}\delta\mathbf{x}^j \quad i\in\Omega \tag{B.6}$$

Note that the term $a_i^j = \frac{w(\mathbf{x}_i-\mathbf{x}^j)}{\sum\limits_{k\in\partial\Omega} w(\mathbf{x}_i-\mathbf{x}^k)} = \frac{p_i^j}{q_i}$. The adjoint model can be readily derived as shown in eq. (B.7), where the transpose IDW mesh smoothing operator $\mathbf{A}^T$ is given in eq. (B.8). Similar to the primal smoother, the adjoint can be implemented using a matrix-

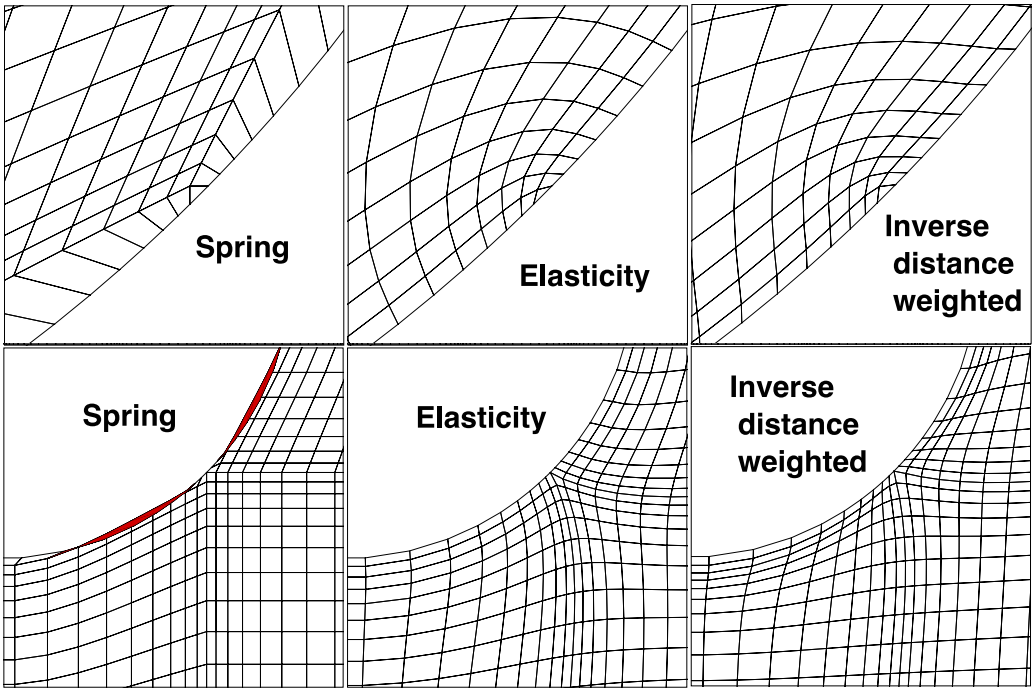


Figure 2.3: Results of the boundary deformation define in fig. 2.1(b) for the linear spring, linear elasticity and IDW (zoomed near the corner and inverted elements marked in red)

free approach. The parallel OpenMP implementation of the matrix-free adjoint IDW is shown in listing 2. In comparison to the primal mesh smoothing the adjoint performs less operations per loop count and involves OpenMP reduction operation. Therefore, the scalability of the adjoint is worser than the primal algorithm.

$$\delta\bar{\mathbf{x}}^j = \delta\bar{\mathbf{x}}^j + \mathbf{A}^T\delta\bar{\mathbf{x}}_i \tag{B.7}$$

$$\mathbf{A}^T\delta\bar{\mathbf{x}}_i = \begin{bmatrix} a_1^1 & \dots & \dots & \dots & a_n^1 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ a_1^{n_b} & \dots & \dots & \dots & a_n^{n_b} \end{bmatrix}_{n_b\times n} \begin{bmatrix} \delta\bar{\mathbf{x}}_1 \\ \vdots \\ \vdots \\ \vdots \\ \delta\bar{\mathbf{x}}_n \end{bmatrix}_{n\times 1} = \begin{bmatrix} \frac{p_1^1}{q_1}\delta\bar{\mathbf{x}}_1 + \ldots + \frac{p_n^1}{q_{n_b}}\delta\bar{\mathbf{x}}_n \\ \vdots \\ \frac{p_1^{n_b}}{q_1}\delta\bar{\mathbf{x}}_1 + \ldots + \frac{p_n^{n_b}}{q_{n_b}}\delta\bar{\mathbf{x}}_n \end{bmatrix}_{n_b\times 1} \tag{B.8}$$

```
/* Constants A, a, b, LDEF, ALPHA assumed pre-defined */
void AdjointKernelIDW
( const int n,           const int nb,
  const float x[][3],    const float xb[][3],
  float dx[][3]/* in */, float dxb[][3]/* out */ )
{
  for ( unsigned i = 0; i < n; ++i ) {
    float q = 0.0;
    #pragma omp parallel for reduction(+:q)
    for ( unsigned j = 0; j < nb; ++j ) {
      // Lambda function to obtain p^j
      p[j] = [&]()
      {
        float dr = 1.0 / norm2( x[i][1:3] - xb[j][1:3] );
        return A * ( pow( LDEF * dr, a ) +
               pow( ALPHA * LDEF * dr, b ) );
      }();
      q += p[j];
    }
    q = 1.0 / q;
    #pragma omp parallel for
    for ( unsigned j = 0; j < nb; ++j )
      dxb[j][1:3] += p[j] * q * dx[i][1:3];
  }
}
```

Listing 2: OpenMP parallel C++ code for the IDW adjoint kernel

Appendix C

Algorithmic Differentiation and Seeding

Algorithmic Differentiation (AD) is a technique to obtain derivatives of a computer code or program using the chain rule of calculus. This is in stark contrast to numerical differentiation (eg. finite-difference), which incurs truncation error. In AD the derivatives are exact to machine round-off. At the same time they are not symbolic differentiation because the final and intermediate results are always numerical values in finite precision.

In principle, AD takes a computer program or code and parses it into a computational graph of primitive mathematical operations. Then each primitive is individually differentiated and assembled using the chain rule to yield the final output. Two important quantities must be know a priori for the differentiation (i) input and (ii) output variables. The AD process is quite amenable to automation as the rules of differentiation for the primitives are simple to encode into an AD software tool. There are two competing ways to implement AD, namely, Operator overloading (OO) and Source Transformation (ST) [187, 92] and each have their own merits and shortcomings. In this work the ST AD using Tapenade [92] is used to obtain derivatives for the scientific code STAMPS.

C.1 Forward and Reverse Mode of AD

The program code whose derivative needs to be calculated using AD can be considered as a multivariate vector function F of an independent variable $\mathbf{x}$ as defined in eq. (C.1). The function F maps an input vector $\mathbf{x} \equiv (x_i),\ i = 1,\ \ldots,\ n \in \mathcal{R}^n$ onto an output vector $\mathbf{y} \equiv (y_i),\ i = 1,\ \ldots,\ m \in \mathcal{R}^m$.

$$\mathbf{y} = F(\mathbf{x}) \tag{C.1}$$

The entries of the Jacobian $\nabla F(\mathbf{x}) \in \mathcal{R}^{m \times n}$ exist if F is continuously differentiable in the neighbourhood of all arguments. In order to compute this Jacobian using AD, one has the tangent-linear model (forward mode) and the adjoint model (reverse mode) at his disposal. The computational complexity of the Jacobian accumulation based on the tangent-linear model depends on the dimension of the input n. In contrast, the complexity of the adjoint model depends on the output dimension m. The tangent and adjoint model of F are shown in eq. (C.2)-(C.3).

$$\dot{\mathbf{y}} = \dot{\mathbf{y}} + \nabla F(\mathbf{x}) \cdot \dot{\mathbf{x}} \tag{C.2}$$

$$\bar{\mathbf{x}} = \bar{\mathbf{x}} + \nabla F(\mathbf{x})^T \cdot \bar{\mathbf{y}} \tag{C.3}$$

Seeding is best described as the process of retrieval of the directional derivative in eq. (C.3) or (C.2) by initialising appropriate values for $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$ or $\dot{\mathbf{x}}$, $\dot{\mathbf{y}}$. For example, the Jacobian matrix can be constructed row-by-row in $\bar{\mathbf{x}}$ by initialising or seeding $\bar{\mathbf{x}} \equiv (\bar{x}_i),\ i = 1,\ \ldots,\ n = 0$ followed by seeding $\bar{\mathbf{y}} = (\mathbf{e}_i),\ i = 1,\ \ldots,\ m$ over the range of Cartesian unit basis vectors $\mathbf{e}_i \in \mathcal{R}^m$. Note that m evaluations are necessary to assemble the complete Jacobian matrix. Moreover, when $m = 1$ only one evaluation is necessary to assemble all the entries of the gradient in contrast to n evaluations in the tangent-linear mode.

Consider the FORTRAN residual subroutine signature shown in listing 3(a). The function takes the state Q as input and outputs the residual R. The signature of the function

```
! Original function call
subroutine residual(Q, R)
  implicit none
  real, intent(in)  :: Q(:,:)
  real, intent(out) :: R(:,:)
  ...
  ...
  ...
end subroutine residual
```

(a) Residual subroutine

```
! Generated tangent code
subroutine residual_d(Q, Qd, R, Rd)
  implicit none
  real, intent(in)    :: Q(:,:)
  real, intent(in)    :: Qd(:,:)
  real, intent(out)   :: R(:,:)
  real, intent(inout) :: Rd(:,:)
  ...
end subroutine residual_d
```

(b) Residual tangent AD generated code

Listing 3: Residual subroutine and its generated Tapenade forward AD code

```
! Original function call
subroutine residual(Q, R)
  implicit none
  real, intent(in)  :: Q(:,:)
  real, intent(out) :: R(:,:)
  ...
  ...
  ...
end subroutine residual
```

(a) Residual subroutine

```
! Generated adjoint code
subroutine residual_d(Q, Qb, R, Rb)
  implicit none
  real, intent(in)    :: Q(:,:)
  real, intent(inout) :: Qb(:,:)
  real, intent(out)   :: R(:,:)
  real, intent(in)    :: Rb(:,:)
  ...
end subroutine residual_b
```

(b) Residual reverse AD generated code

Listing 4: Residual subroutine and its generated Tapenade reverse AD code

returned by Tapenade after forward and reverse mode differentiation is shown in listing 3(b) and listing 4(b). The signatures are consistent with the forward and adjoint model shown in eq. (C.2)-(C.3). By properly seeding the inputs to the subroutine it is possible to assemble the vairous terms of the tangent linear and adjoint solver. For example, seeding `Rd` $= -\frac{\partial R}{\partial Z}$ and `Qd` $= w^n$ one can assemble the right hand side term of the tangent linear solver (see eq (4.36)). Similarly, if `Qb` is seeded with $-\left(\frac{\partial J}{\partial \mathbf{Q}}\right)^T$ and `Rb` with v one obtains the right hand side term in eq. (4.32) directly in `Qb`. In conclusion, seeding is a key ingredient in AD practice and the models in eq. (C.2)-(C.3) help to understand the seeding process.

```
do iter = 1, nIter
  ...
  ...
  ...
  ...
  call mpi_send(Q)
  call mpi_recv(Q_halo)
  call residual(Q, R)
  call update(Q, R)
  call costFunction(Q, J)
end do
```

(a) Primal fixed point iteration (FPI)

```
do iter = 1, nIter
  Jb = 1
  Qb = 0
  call costFun_b(Q, Qb, J, Jb)
  Rb = v
  call residualLocal_b(Q, Qb, R, Rb)
  call mpi_send(Qb_halo)
  call mpi_recv(t)
  Qb = Qb + t
  call update(v, Qb)
end do
```

(b) Adjoint fixed point iteration (FPI)

Listing 5: Fixed point iteration of primal and adjoint solver with manually differentiated MPI calls

C.2 Adjoining MPI Calls

Application of AD to MPI parallel codes especially in reverse mode remains a challenge. A major step forward has been made with the development of the AMPI library [119] which, under certain restrictions, allows to automatically derive the required MPI communications for the reverse and forward-differentiated code. AMPI focuses on the correct differentiation of an individual piece of code, but does not automate the integration of the differentiated elements into a driver code. In particular, the user has to provide appropriate seeding for the relevant inputs in order to obtain correct gradients. But this is far from trivial in practical application.

The two level halo partitioning scheme used in this work simplifies the differentiation of the parallel fixed-point iterative solver. In fact no MPI calls are actually present in the `residual` subroutine. The blocking halo data exchange is pushed outside the residual function and can be manually differentiated and inserted in the fixed point iteration loop. Using the analysis in ref. [119] the only change required for the adjoint MPI code is the reversal of the send/recv and accumulation of the values to the received buffer (see listing 5). In the forward mode, the MPI calls are simplified because every primal MPI call of `Q` has a corresponding pair for `Qb` as shown in listing 6.

```
! primal mpi call
call mpi_send (Q)
call mpi_recv (Q_halo)
...
...
```

```
! tangent call
call mpi_send (Q)
call mpi_recv (Q_halo)
call mpi_send (Qd)
call mpi_recv (Qd_halo)
```

Listing 6: Pairing of MPI calls in forward mode

Bibliography

[1] Devendra Ghate. *Inexpensive Uncertainty Analysis for CFD Applications*. PhD thesis, Universtiy of Oxford, 2013.

[2] Michael B. Giles and Niles A. Pierce. Adjoint Error Correction for Integral Outputs. In *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, pages 47–95. 2003.

[3] Leo W. T. Ng and Karen E. Willcox. Multifidelity approaches for optimization under uncertainty. *International Journal for Numerical Methods in Engineering*, 100(10):746–772, December 2014.

[4] G. Geraci, Michael S. Eldred, and Gianluca Iaccarino. A multifidelity control variate approach for the multilevel Monte Carlo technique. Technical report, Center for Turbulence Research, Stanford University, 2015.

[5] Min Young Yoo, Nam Ho Kim, and Joo Ho Choi. Comparison among Methods of Modeling Epistemic Uncertainty in Reliability Estimation. *Journal of the Computational Structural Engineering Institute of Korea*, 27(6):605–613, December 2014.

[6] Benjamin Peherstorfer, Karen E. Willcox, and Max Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *ArXiv*, June 2018.

[7] Goran Mladenovic, Ljubodrag Tanovic, and Kornel Ehmann. Tool path generation

for milling of free form surfaces with feed rate scheduling. *FME Transaction*, 43(1):9–15, 2015.

[8] D. H. Stamatis. *Failure Mode And Effect Analysis: FMEA from Theory to Execution.* ASQ Quality Press, Wisconsin, 2nd edition, 2003.

[9] J.N. Yang, S.D. Manning, J.L. Rudd, and M.E. Artley. Probabilistic durability analysis methods for metallic airframes. *Probabilistic Engineering Mechanics*, 2(1):9–15, March 1987.

[10] Robert Ryan and John Townsend. Application of probabilistic analysis/design methods in space programs - The approaches, the status, and the needs. In *34th Structures, Structural Dynamics and Materials Conference*, Reston, Virigina, April 1993. American Institute of Aeronautics and Astronautics.

[11] T.A. Cruse, O.H. Burnside, Y.-T. Wu, E.Z. Polch, and J.B. Dias. Probabilistic structural analysis methods for select space propulsion system structural components (PSAM). *Computers & Structures*, 29(5):891–901, January 1988.

[12] Chris Lykins and Dan Thomson. The Air Force's application of probabilistics to gas turbine engines. In *35th Structures, Structural Dynamics, and Materials Conference*, Reston, Virigina, April 1994. American Institute of Aeronautics and Astronautics.

[13] Pranay Seshadri, Paul Constantine, Gianluca Iaccarino, and Geoffrey Thomas Parks. A density-matching approach for optimization under uncertainty. *Computer Methods in Applied Mechanics and Engineering*, 305:562–578, June 2016.

[14] Richard Ahlfeld, B. Belkouchi, and Francesco Montomoli. SAMBA: Sparse Approximation of Moment-Based Arbitrary Polynomial Chaos. *Journal of Computational Physics*, 320:1–16, 2016.

[15] Benjamin Peherstorfer, Tiangang Cui, Youssef Marzouk, and Karen E. Willcox.

Multifidelity importance sampling. *Computer Methods in Applied Mechanics and Engineering*, 300:490–509, March 2016.

[16] Gilbert Strang. *Computational Science and Engineering*. Wellesley-Cambridge Press, 2007.

[17] Siddhartha Mishra, Christoph Schwab, and Jonas Šukys. Multi-level Monte Carlo Finite Volume Methods for Uncertainty Quantification in Nonlinear Systems of Balance Laws. In Schwab C. Bijl H., Lucor D., Mishra S., editor, *Uncertainty Quantification in Computational Fluid Dynamics. Lecture Notes in Computational Science and Engineering, vol 92*, pages 225–294. 2013.

[18] Alexander Lange, Matthias Voigt, Konrad Vogeler, and Erik Johann. Principal component analysis on 3D scanned compressor blades for probabilistic CFD simulation. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA*, Reston, Virigina, April 2012. American Institute of Aeronautics and Astronautics.

[19] Michael B. Giles. Multilevel Monte Carlo Path Simulation. *Operations Research*, 56(3):607–617, June 2008.

[20] Siddhartha Mishra and Christoph Schwab. Sparse tensor multi-level Monte Carlo finite volume methods for hyperbolic conservation laws with random initial data. *Mathematics of Computation*, 81(280):1979–2018, 2012.

[21] Michele Pisaroni, Fabio Nobile, and Penelope Leyland. Continuation Multi-Level Monte-Carlo method for Uncertainty Quantification in Turbulent Compressible Aerodynamics Problems modeled by RANS. Technical report, Nr. 10.2017, MATH-ICSE Technical Report, EPFL, 2017.

[22] Leo W. T. Ng and Karen E. Willcox. Monte Carlo Information-Reuse Approach to Aircraft Conceptual Design Optimization Under Uncertainty. *Journal of Aircraft*,

53(2):427–438, March 2016.

[23] Pramudita Satria Palar, Takeshi Tsuchiya, and Geoffrey Thomas Parks. Multi-fidelity non-intrusive polynomial chaos based on regression. *Computer Methods in Applied Mechanics and Engineering*, 305:579–606, June 2016.

[24] Gianluca Geraci, Michael S. Eldred, and Gianluca Iaccarino. A multifidelity multi-level Monte Carlo method for uncertainty propagation in aerospace applications. In *19th AIAA Non-Deterministic Approaches Conference*, Reston, Virginia, January 2017. American Institute of Aeronautics and Astronautics.

[25] Jens-Dominik Müller and Michael B. Giles. Edge-based Multigrid schemes for Hybrid Grids. In *Numerical Methods for Fluid Dynamics, VI, ICFD*, pages 425–432, 1998.

[26] Michael B. Giles, Mihai C. Duta, Jens-Dominik M-uacute, Ller, and Niles A. Pierce. Algorithm Developments for Discrete Adjoint Methods. *AIAA Journal*, 41(2):198–205, February 2003.

[27] M. Meyer and H. G. Matthies. Efficient model reduction in non-linear dynamics using the Karhunen-Loeve expansion and dual-weighted-residual methods. *Computational Mechanics*, 31(1-2):179–191, May 2003.

[28] Claudia Schillings, Stephan Schmidt, and Volker Schulz. Efficient shape optimization for certain and uncertain aerodynamic design. *Computers & Fluids*, 46(1):78–87, July 2011.

[29] Michael B. Giles. On adjoint equations for error analysis and optimal grid adaptation in CFD. In D A Caughey and M M Hafez, editors, *Frontiers of Computational Fluid Dynamics 2006*, pages 155–170. World scientific, December 2005.

[30] Michele Sergio Campobasso and Michael B. Giles. Stabilization of a Linear Flow Solver for Turbomachinery Aeroelasticity Using Recursive Projection Method.

AIAA Journal, 42(9):1765–1774, September 2004.

[31] Richard P. Dwight and Joël Brezillon. Efficient and robust algorithms for solution of the adjoint compressible Navier-Stokes equations with applications. *International Journal for Numerical Methods in Fluids*, 60(4):365–389, June 2009.

[32] Joshua A. Krakos and David L. Darmofal. Effect of Small-Scale Output Unsteadiness on Adjoint-Based Sensitivity. *AIAA Journal*, 48(11):2611–2623, November 2010.

[33] Shenren Xu, David Radford, Marcus Meyer, and Jens-Dominik Müller. Stabilisation of discrete steady adjoint solvers. *Journal of Computational Physics*, 299:175–195, October 2015.

[34] Ralph C. Smith. *Uncertainty Quantification: Theory, Implementation, and Applications*. SIAM, 2014.

[35] Michael Emory, Johan Larsson, and Gianluca Iaccarino. Modeling of structural uncertainties in Reynolds-averaged Navier-Stokes closures. *Physics of Fluids*, 25(11):110822, November 2013.

[36] C. Gorlé, Johan Larsson, Michael Emory, and Gianluca Iaccarino. The deviation from parallel shear flow as an indicator of linear eddy-viscosity model inaccuracy. *Physics of Fluids*, 26(5):051702, May 2014.

[37] Kesari Mishra. *Uncertainty Propagation through Dependability Models*. PhD thesis, Duke University, 2001.

[38] Brian M. Adams, Mohamed S. Ebeida, Michael S. Eldred, John D. Jakeman, Laura P. Swiler, Adam J. Stephens, Dena M. Vigil, Timothy M. Wildey, William J. Bohnhoff, Keith R. Dalbey, John P. Eddy, Kenneth T. Hu, Lara E. Bauman, and Patricia D. Hough. Dakota, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and

Sensitivity Analysis: Version 6.7 User's Manual. Technical report, Sandia National Laboratory, 2017.

[39] Claudia Schillings. *Optimal Aerodynamic Design under Uncertainties*. PhD thesis, Universität Trier, 2010.

[40] Jens-Dominik Müller, Jan-Christian Huckelheim, and Orest Mykhaskiv. STAMPS: a Finite-Volume Solver Framework for Adjoint Codes Derived with Source-Transformation AD. In *2018 Multidisciplinary Analysis and Optimization Conference*, Reston, Virginia, June 2018. American Institute of Aeronautics and Astronautics.

[41] Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent Component Analysis*. Adaptive and Learning Systems for Signal Processing, Communications, and Control. John Wiley & Sons, Inc., New York, USA, May 2001.

[42] Francesco Montomoli, Mauro Carnevale, Antonio D'Ammaro, Michela Massini, and Simone Salvadori. *Uncertainty Quantification in Computational Fluid Dynamics and Aircraft Engines*. Springer Briefs in Applied Sciences and Technology. Springer International Publishing, 2015.

[43] Achintya Haldar and Sankaran Mahadevan. *Probability, Reliability and Statistical Methods in Engineering Design*. Wiley, 1st edition, 1999.

[44] Christian Robert and George Casella. *Introducing Monte Carlo Methods with R*. Springer New York, New York, NY, 2010.

[45] Zsolt Sándor and Péter András. Alternative sampling methods for estimating multivariate normal probabilities. *Journal of Econometrics*, 120(2):207–234, June 2004.

[46] J.C. Helton and F.J. Davis. Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering & System*

Safety, 81(1):23–69, July 2003.

[47] Pierre L'Ecuyer and Christiane Lemieux. Recent Advances in Randomized Quasi-Monte Carlo Methods. In Dror M., L'Ecuyer P., and Szidarovszky F., editors, *Modeling Uncertainty : International Series in Operations Research & Management Science, vol 46*, pages 419–474. Kluwer Academic Publishers, Boston, 2005.

[48] I. M. Sobol. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, January 1967.

[49] Roger G. Ghanem, David Higdon, and Houman Owhadi. *Handbook of Uncertainty Quantification*. Springer International Publishing, Cham, 2017.

[50] Paul Bratley and Bennett L. Fox. ALGORITHM 659: implementing Sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software*, 14(1):88–100, March 1988.

[51] Richard Ahlfeld, Francesco Montomoli, E. Scalas, and Shahrokh Shahpar. Uncertainty Quantification for Fat-Tailed Probability Distributions in Aircraft Engine Simulations. *Journal of Propulsion and Power*, 33(4):881–890, July 2017.

[52] Richard Ahlfeld and Francesco Montomoli. A Single Formulation for Uncertainty Propagation in Turbomachinery: SAMBA PC. *Journal of Turbomachinery*, 139(11), August 2017.

[53] Benjamin Peherstorfer, Boris Kramer, and Karen E. Willcox. Multifidelity preconditioning of the cross-entropy method for rare event simulation and failure probability estimation. 2017.

[54] Rajan Srinivasan. *Importance Sampling*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[55] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo*

Method. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., Hoboken, NJ, USA, November 2016.

[56] Benjamin Peherstorfer, Max Gunzburger, and Karen Willcox. Convergence analysis of multifidelity Monte Carlo estimation. *Numerische Mathematik*, 139(3):683–707, July 2018.

[57] Stefan Heinrich. Multilevel Monte Carlo Methods. In *LSSC 2001. Lecture Notes in Computer Science, vol 2179*, pages 58–67. Springer, Berlin, 2001.

[58] Stefan Heinrich. Monte Carlo Complexity of Global Solution of Integral Equations. *Journal of Complexity*, 14(2):151–175, June 1998.

[59] Stefan Heinrich and Eugène Sindambiwe. Monte Carlo Complexity of Parametric Integration. *Journal of Complexity*, 15(3):317–341, September 1999.

[60] Ahmed Kebaier. Statistical Romberg Extrapolation: A New Variance Reduction Method and Applications to Option Pricing. *The Annals of Applied Probability*, 15(4):2681–2705, 2005.

[61] Michael B. Giles. Improved multilevel Monte Carlo convergence using the Milstein scheme. Technical report, Preprint NA-06/22, Oxford computing lab, Oxford, UK, 2006.

[62] Nathan Collier, Abdul-Lateef Haji-Ali, Fabio Nobile, Erik von Schwerin, and Raúl Tempone. A continuation multilevel Monte Carlo algorithm. *BIT Numerical Mathematics*, 55(2):399–432, June 2015.

[63] Andrea Barth, Christoph Schwab, and Nathaniel Zollinger. Multi-level Monte Carlo Finite Element method for elliptic PDEs with stochastic coefficients. *Numerische Mathematik*, 119(1):123–161, September 2011.

[64] Michele Pisaroni, Sebastian Krumscheid, and Fabio Nobile. Quantifying uncertain system outputs via the multilevel Monte Carlo method - Part I: Central moment

estimation. Technical report, MATHICSE technical report no. 23.2017, 2017.

[65] Norbert Wiener. The Homogeneous Chaos. *American Journal of Mathematics*, 60(4):897, October 1938.

[66] Dongbin Xiu and George Em Karniadakis. The Wiener–Askey Polynomial Chaos for Stochastic Differential Equations. *SIAM Journal on Scientific Computing*, 24(2):619–644, January 2002.

[67] Roelof Koekoek and René F. Swarttouw. The Askey-scheme of hypergeometric orthogonal polynomials and its q-analogue. *arXiv.org*, February 1996.

[68] Wim Schoutens. *Stochastic Processes and Orthogonal Polynomials*, volume 146 of *Lecture Notes in Statistics*. Springer, New York, 2000.

[69] Dongbin Xiu. *Numerical Methods for Stochastic Computations*. Princeton University Press, 2010.

[70] J. Tryoen, Olivier P. Le Maitre, M. Ndjinga, and A. Ern. Intrusive Galerkin methods with upwinding for uncertain nonlinear hyperbolic systems. *Journal of Computational Physics*, 229(18):6485–6511, September 2010.

[71] G. Onorato, G. J. A. Loeven, G. Ghorbaniasl, Hester Bijl, and C. Lacor. Comparison of intrusive and non-intrusive polynomial chaos methods for CFD applications in aeronautics. In *5th European Conference on Computational Fluid Dynamics, ECCOMAS CFD 2010*, 2010.

[72] Michael S. Eldred and John Burkardt. Comparison of Non-Intrusive Polynomial Chaos and Stochastic Collocation Methods for Uncertainty Quantification. In *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, Reston, Virigina, January 2009. American Institute of Aeronautics and Astronautics.

[73] S. A. Smolyak. Quadrature and interpolation formulas for tensor products of

certain classes of functions. *Dokl. Akad. Nauk SSSR*, 148(5):1042–1045, 1963.

[74] S. Oladyshkin and W. Nowak. Data-driven uncertainty quantification using the arbitrary polynomial chaos expansion. *Reliability Engineering & System Safety*, 106(11-14):179–190, October 2012.

[75] Jeroen A. S. Witteveen, Sunetra Sarkar, and Hester Bijl. Modeling physical uncertainties in dynamic stall induced fluid–structure interaction of turbine blades using arbitrary polynomial chaos. *Computers & Structures*, 85(11-14):866–878, June 2007.

[76] Jeroen A. S. Witteveen and Gianluca Iaccarino. Simplex Stochastic Collocation with Random Sampling and Extrapolation for Nonhypercube Probability Spaces. *SIAM Journal on Scientific Computing*, 34(2):814–838, January 2012.

[77] Jeroen A. S. Witteveen and Gianluca Iaccarino. Simplex stochastic collocation with ENO-type stencil selection for robust uncertainty quantification. *Journal of Computational Physics*, 239:1–21, April 2013.

[78] Kaushik Kumar Nagarajan, Laurent Cordier, and Christophe Airiau. Development and Application of a Reduced Order Model for the Control of Self-Sustained Instabilities in Cavity Flows. *Communications in Computational Physics*, 14(01):186–218, July 2013.

[79] Michael L. Stein. *Interpolation of Spatial Data : Some Theory for Kriging*. Springer Series in Statistics. Springer New York, New York, NY, 1999.

[80] Zhenhua Zhang, Liang Xu, Paulo Flores, and Hamid M. Lankarani. A DOE- and Kriging-Based Model for Studying on the Dynamics of Multibody Mechanical Systems With Revolute Joint Clearance. In *Volume 4A: Dynamics, Vibration and Control*. ASME, November 2013.

[81] Daniel Fulcoly, Nirav Shah, Adam Ross, and Donna Rhodes. Exploiting Multi-

dimensional Design of Experiments and Kriging Methods: An Application to a Satellite Radar System Tradespace and Orbital Transfer Vehicle Tradespace. In *AIAA SPACE 2012 Conference & Exposition*, Reston, Virigina, September 2012. American Institute of Aeronautics and Astronautics.

[82] Mohamed Amine Bouhlel, Nathalie Bartoli, Abdelkader Otsmane, and Joseph Morlier. Improving kriging surrogates of high-dimensional design models by Partial Least Squares dimension reduction. *Structural and Multidisciplinary Optimization*, 53(5):935–952, May 2016.

[83] Niles A. Pierce and Michael B. Giles. Adjoint Recovery of Superconvergent Functionals from PDE Approximations. *SIAM Review*, 42(2):247–264, 2000.

[84] Faidon Christakopoulos. *Sensitivity computation and shape optimisation in aerodynamics using the adjoint methodology and Automatic Differentiation.* PhD thesis, Queen Mary University of London, 2012.

[85] Anna Engels-Putzka and Jan Backhaus. Adjoint-based Shape Sensitivities for Turbomachinery Design Optimizations. In *10th International Conference on Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems, EUROGEN 2013*, Las Palmas de Gran Canaria, Spain, 2013.

[86] Jiaqi Luo and Feng Liu. Statistical evaluation of performance impact of manufacturing variability by an adjoint method. *Aerospace Science and Technology*, 77:471–484, June 2018.

[87] Markus P. Rumpfkeil and Dimitri J. Mavriplis. Efficient Hessian Calculations Using Automatic Differentiation and the Adjoint Method with Applications. *AIAA Journal*, 48(10):2406–2417, October 2010.

[88] Markus P. Rumpfkeil. Optimizations Under Uncertainty Using Gradients, Hessians, and Surrogate Models. *AIAA Journal*, 51(2):444–451, February 2013.

[89] Dongbin Xiu and George Em Karniadakis. Supersensitivity due to uncertain boundary conditions. *International Journal for Numerical Methods in Engineering*, 61(12):2114–2138, November 2004.

[90] MPI-Forum. MPI: A Message-Passing Interface Standard. Technical report, University of Tennessee Knoxville, TN, 1994.

[91] Shenren Xu, Pavanakumar Mohanamuraly, and Jens-Dominik Müller. A Newton Krylov RANS solver for effcient and robust turbomachinery aerodynamics analysis. *AIAA Journal of Propulsion and Power (submitted)*, 2019.

[92] Laurent Hascoet and Valérie Pascual. The Tapenade automatic differentiation tool. *ACM Transactions on Mathematical Software*, 39(3):1–43, April 2013.

[93] R. K. Agarwal and J. E. Deese. Euler calculations for flowfield of a helicopter rotor in hover. *Journal of Aircraft*, 24(4):231–238, April 1987.

[94] P. Spalart and S. Allmaras. A one-equation turbulence model for aerodynamic flows. In *30th Aerospace Sciences Meeting and Exhibit*, Reston, Virigina, January 1992. American Institute of Aeronautics and Astronautics.

[95] Christer Ericson. *Real-Time Collision Detection*. CRC Press, 1st edition, 2004.

[96] Rainald Lohner. *Applied Computational Fluid Dynamics Techniques*. John Wiley & Sons, Ltd, Chichester, UK, March 2008.

[97] Kaare Albert Sorensen. *A multigrid accelerated procedure for the solution of compressible fluid flows on unstructured hybrid meshes*. PhD thesis, University of Wales Swansea, 2001.

[98] C. L. Chen, W. J. Mccroskey, and S. Obayashi. Numerical solutions of forward-flight rotor flow using an upwind method. *Journal of Aircraft*, 28(6):374–380, June 1991.

[99] Ami Harten and James M Hyman. Self adjusting grid methods for one-dimensional hyperbolic conservation laws. *Journal of Computational Physics*, 50(2):235–269, May 1983.

[100] Jiri Blazek. *Computational Fluid Dynamics: Principles and Applications*. Elsevier, 2015.

[101] V. Venkatakrishnan. Convergence to Steady State Solutions of the Euler Equations on Unstructured Grids with Limiters. *Journal of Computational Physics*, 118(1):120–130, April 1995.

[102] Z. J. Wang. A fast nested multi-grid viscous flow solver for adaptive Cartesian/Quad grids. *International Journal for Numerical Methods in Fluids*, 33(5):657–680, July 2000.

[103] Christopher Michalak and Carl Ollivier-Gooch. Accuracy preserving limiter for the high-order accurate solution of the Euler equations. *Journal of Computational Physics*, 228(23):8693–8711, December 2009.

[104] Andreas Haselbacher and Jiri Blazek. Accurate and Efficient Discretization of Navier-Stokes Equations on Mixed Grids. *AIAA Journal*, 38(11):2094–2102, November 2000.

[105] Jan-Reneé Carlson. Inflow/Outflow Boundary Conditions with Application to FUN3D. Technical report, National Aeronautics and Space Administration, NASA/TM–2011-217181, 2011.

[106] Peter Eliasson, Sofia Eriksson, and Jan Nordström. Influence of Weak and Strong Solid Wall Boundary Conditions on the Convergence to Steady-State of the Navier-Stokes Equations. In *19th AIAA Computational Fluid Dynamics*, AIAA 2009-3551, Reston, Virigina, June 2009. American Institute of Aeronautics and Astronautics.

[107] Pierre Moinier, Jens-Dominik Müller, and Michael B. Giles. Edge-Based Multigrid

and Preconditioning for Hybrid Grids. *AIAA Journal*, 40(10):1954–1960, October 2002.

[108] Richard P. Dwight. *Efficiency Improvements of RANS-based Analysis and Optimization using Implicit and Adjoint Methods on Unstructured Grids*. PhD thesis, University of Manchester, 2006.

[109] K. Badcock, M. Woodgate, F. Cantariti, and B. Richards. Solution of the unsteady Euler equations in three dimensions using a fully unfactored method. In *38th Aerospace Sciences Meeting and Exhibit*, Reston, Virigina, January 2000. American Institute of Aeronautics and Astronautics.

[110] C. T. Kelley. *Solving Nonlinear Equations with Newton's Method*. Society for Industrial and Applied Mathematics, January 2003.

[111] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, January 2003.

[112] Jed Brown and Peter Brune. Low-rank Quasi-Newton updates for robust Jacobian lagging in Newton methods. In *Proceedings of the 2013 International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, pages 2554–2565, Sun Valley, ID, 2013.

[113] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Society for Industrial and Applied Mathematics, January 2000.

[114] Laurent Y.M. Gicquel, N Gourdain, J.-F. Boussuge, H. Deniau, G Staffelbach, P. Wolf, and Thierry Poinsot. High Performance Parallel Computing of Flows in Complex Geometries. *Comptes Rendus Mécanique*, 339(2-3):104–124, February 2011.

[115] Pavanakumar Mohanamuraly, Laurent Hascoet, and Jens-Dominik Müller. Seeding and adjoining zero-halo partitioned parallel scientific codes. *OMS Journal*

(available online), 2018.

[116] George Karypis and Vipin Kumar. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, January 1998.

[117] Bruce Christianson. Reverse Accumulation and Attractive Fixed Points. *Optimization Methods and Software*, 3(4):311–326, 1994.

[118] Ralf Giering and Thomas Kaminski. Recipes for Adjoint Code Construction. *ACM Transactions on Mathematical Software*, 24(4):437–474, December 1998.

[119] Jean Utke, Laurent Hascoet, Patrick Heimbach, Chris Hill, Paul Hovland, and Uwe Naumann. Toward adjoinable MPI. In *2009 IEEE International Symposium on Parallel & Distributed Processing*, pages 1–8. IEEE, May 2009.

[120] Akihiro Pujii, Akira Nishida, and Yoshio Oyanagi. Evaluation of Parallel Aggregate Creation Orders: Smoothed Aggregation Algebraic Multigrid Method. In *High Performance Computational Science and Engineering*, pages 99–122. Springer-Verlag, New York, 2005.

[121] Allison H. Baker, Robert D. Falgout, Tzanio V. Kolev, and Ulrike Meier Yang. Scaling Hypre's Multigrid Solvers to 100,000 Cores. In *High-Performance Scientific Computing*, pages 261–279. Springer London, London, 2012.

[122] Robert D. Falgout, Tzanio V. Kolev, and Ulrike Meier Yang. An assumed partition algorithm for determining processor inter-communication. *Parallel Computing*, 32(5-6):394–414, June 2006.

[123] R. Das, D. J. Mavriplis, J. Saltz, S. Gupta, and R. Ponnysamy. Design and implementation of a parallel unstructured Euler solver using software primitives. *AIAA Journal*, 32(3):489–496, March 1994.

[124] Alvaro L. G. A. Coutinho, Marcos A. D. Martins, Rubens M. Sydenstricker, and

Renato N. Elias. Performance comparison of data-reordering algorithms for sparse matrix–vector multiplication in edge-based unstructured grid computations. *International Journal for Numerical Methods in Engineering*, 66(3):431–460, April 2006.

[125] J. Mellor-Crummey, D. Whalley, and K Kennedy. Improving Memory Hierarchy Performance for Irregular Applications Using Data and Computation Reorderings. *International Journal of Parallel Programming*, 29(3):217–247, 2001.

[126] Mark Frederick Hoemmen. Communication-avoiding Krylov subspace methods. Technical report, Electrical Engineering and Computer Sciences, University of California at Berkeley, 2010.

[127] P. Ghysels, T. J. Ashby, K. Meerbergen, and W. Vanroose. Hiding Global Communication Latency in the GMRES Algorithm on Massively Parallel Machines. *SIAM Journal on Scientific Computing*, 35(1):48–71, January 2013.

[128] Ichitaro Yamazaki, Mark Frederick Hoemmen, Piotr Luszczek, and Jack Dongarra. Improving Performance of GMRES by Reducing Communication and Pipelining Global Collectives. In *2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 1118–1127. IEEE, May 2017.

[129] Valérie Frayssé, Luc Giraud, and Hatim Kharraz-Aroussi. On the influence of the orthogonalization scheme on the parallel performance of GMRES. In Pritchard D. and Reeve J., editors, *Euro-Par'98 Parallel Processing. Lecture Notes in Computer Science, vol 1470*, pages 751–762. Springer, 1998.

[130] Jocelyne Erhel. A Parallel GMRES Version For General Sparse Matrices. *Electronic Transactions on Numerical Analysis*, 3:160–176, 1995.

[131] Laura Grigori and Sophie Moufawad. Communication Avoiding ILU0 Preconditioner. *SIAM Journal on Scientific Computing*, 37(2):C217–C246, January 2015.

[132] Nick Vannieuwenhoven and Karl Meerbergen. IMF: An Incomplete Multifrontal

LU-Factorization for Element-Structured Sparse Linear Systems. *SIAM Journal on Scientific Computing*, 35(1):270–293, January 2013.

[133] J. D. Denton. Loss Mechanisms in Turbomachines. In *The 1993 IGTI Scholar Lecture*. ASME, May 1993.

[134] J.-D. Müller and P. Cusdin. On the performance of discrete adjoint CFD codes using automatic differentiation. *International Journal for Numerical Methods in Fluids*, 47(8-9):939–945, March 2005.

[135] I. G. Currie. *Fundamental Mechanics of Fluids*. CRC Press, 4 edition, 2012.

[136] Christopher Rumsey. Langley Research Center: Turbulence Modeling Resource, 2018.

[137] M. Sajben, T. Bogar, and J. Kroutil. Forced oscillation experiments in supercritical diffuser flows with application to ramjet instabilities. In *17th Joint Propulsion Conference*, Reston, Virigina, July 1981. American Institute of Aeronautics and Astronautics.

[138] T. J. Bogar, M. Sajben, and J. C. Kroutil. Characteristic frequencies of transonic diffuser flow oscillations. *AIAA Journal*, 21(9):1232–1240, September 1983.

[139] J. T. Salmon, T. J. Bogar, and M. Sajben. Laser Doppler velocimeter measurements in unsteady, separated, transonic diffuser flows. *AIAA Journal*, 21(12):1690–1697, December 1983.

[140] T. J. Bogar. Structure of self-excited oscillations in transonic diffuser flows. *AIAA Journal*, 24(1):54–61, January 1986.

[141] T. Hsieh, A. B. Wardlaw, P. Collins, and T. Coakley. Numerical investigation of unsteady inlet flowfields. *AIAA Journal*, 25(1):75–81, January 1987.

[142] N. Georgiadis, J. Drummond, and B. Leonard. Evaluation of turbulence models in

the PARC code for transonic diffuser flows. In *32nd Aerospace Sciences Meeting and Exhibit*, Reston, Virigina, January 1994. American Institute of Aeronautics and Astronautics.

[143] R. A Vandebraembussche, O. Leonard, and L. Nekmouche. Subsonic and Transonic Blade Design by Means of Analysis Codes. Technical report, AGARD, Computational Methods for Aerodynamic Design (Inverse) and Optimization, 1990.

[144] Tony Arts, M. Lambert de Rouvroit, and A. W. Rutherford. Aero-thermal investigation of a highly loaded transonic linear turbine guide vane cascade: A test case for inviscid and viscous flow computations. Technical report, Von Karman Institute for Fluid Dynamics, Belgium, 1990.

[145] Tony Arts and M. Lambert de Rouvroit. Aero-Thermal Performance of a Two-Dimensional Highly Loaded Transonic Turbine Nozzle Guide Vane: A Test Case for Inviscid and Viscous Flow Computations. *Journal of Turbomachinery*, 114(1):147, 1992.

[146] Arnaud Chatel, Tom Verstraete, Grégory Coussement, and Lasse Mueller. Single-Point Optimization of the LS89 Turbine Cascade Using a Hybrid Algorithm. In *ASME Turbo Expo 2018: Turbomachinery Technical Conference and Exposition, Volume 2D: Turbomachinery*. ASME, June 2018.

[147] Ananthakrishnan Kaliyaperumal and M. Govardhan. Secondary Flows and Losses in a Highly Loaded Low Aspect Ratio Transonic Axial Flow Turbine Stage. In *52nd AIAA/SAE/ASEE Joint Propulsion Conference*, Reston, Virginia, July 2016. American Institute of Aeronautics and Astronautics.

[148] E. Collado Morata, N. Gourdain, F. Duchaine, and L.Y.M. Gicquel. Effects of free-stream turbulence on high pressure turbine blade heat transfer predicted by structured and unstructured LES. *International Journal of Heat and Mass Transfer*, 55(21-22):5754–5768, October 2012.

[149] Anirban Garai, Laslo T. Diosady, Scott M. Murman, and Nateri K. Madavan. Scale-Resolving Simulations of Bypass Transition in a High-Pressure Turbine Cascade Using a Spectral Element Discontinuous Galerkin Method. *Journal of Turbomachinery*, 140(3), December 2017.

[150] Tom Verstraete. Private communication, 2018.

[151] Chung Y Wu. Arbitrary Surface Flank Milling and Flank SAM in the Design and Manufacturing of Jet Engine Fan and Compressor Airfoils. In *ASME Turbo Expo 2012: Turbine Technical Conference and Exposition*, 2012.

[152] Victor E. Garzon. *Probabilistic Aerothermal Design of Compressor Airfoils*. PhD thesis, Massachusetts Institute of Technology, 2003.

[153] Rudolph W Preisendorfer and Curtis Mobley. *Principal Component Analysis in Meteorology and Oceanography*. Elsevier, 1988.

[154] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2005.

[155] K Grill. Stochastic process and Random Fields. In Reinhard Viertl, editor, *PROBABILITY AND STATISTICS - Volume I: Probability Theory Stochastic Processes and Random Fields*, pages 1–410. EOLSS Publications, 1st edition, 2009.

[156] J. Hacker. *Statistical Analysis of Manufacturing Deviations and Classification Methods for Probabilistic Aerothermal Design of Turbine Blades*. Diplomarbeit, University of Stuttgart, Stuttgart, Germany, 2000.

[157] Qiqi Wang, Han Chen, Rui Hu, and Paul Constantine. Conditional sampling and experiment design for quantifying manufacturing error of transonic airfoil. In *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, Reston, Virigina, January 2011. American Institute of Aeronautics and Astronautics.

[158] Eric A. Dow and Qiqi Wang. The Implications of Tolerance Optimization on Compressor Blade Design. *Journal of Turbomachinery*, 137(10):7, October 2015.

[159] Eric A. Dow and Qiqi Wang. Optimization of Gaussian Random Fields. *SIAM Journal on Scientific Computing*, 37(4):1685–1704, January 2015.

[160] Apurva Kumar, P. B. Nair, A. J. Keane, and Shahrokh Shahpar. Probabilistic Performance Analysis of Eroded Compressor Blades. In *ASME 2005 Power Conference*, pages 1175–1183. ASME, 2005.

[161] A. Javed, R. Pecnik, M. Olivero, and J. P. van Buijtenen. Effects of Manufacturing Noise on Microturbine Centrifugal Impeller Performance. *Journal of Engineering for Gas Turbines and Power*, 134(10):102302, 2012.

[162] Kay Heinze, Winfried-Hagen Friedl, Konrad Vogeler, and Matthias Voigt. Probabilistic HCF-Investigation of Compressor Blades. In *Volume 6: Structures and Dynamics, Parts A and B*, pages 1103–1112. ASME, 2009.

[163] Helmut Sobieczky. Parametric Airfoils and Wings. In Fujii K. and Dulikravich G.S., editors, *Recent Development of Aerodynamic Design Methodologies. Notes on Numerical Fluid Mechanics (NNFM), vol. 65*, pages 71–87. Vieweg Teubner Verlag, 1999.

[164] Volker Schulz and Claudia Schillings. Problem Formulations and Treatment of Uncertainties in Aerodynamic Design. *AIAA Journal*, 47(3):646–654, March 2009.

[165] Eric A. Dow and Qiqi Wang. Optimal Design and Tolerancing of Compressor Blades Subject to Manufacturing Variability. In *16th AIAA Non-Deterministic Approaches Conference*, pages 1–11, Reston, Virginia, January 2014. American Institute of Aeronautics and Astronautics.

[166] Volker Schulz and Claudia Schillings. Optimal Aerodynamic Design under Uncertainty. In *Management and Minimisation of Uncertainties and Errors in Numerical*

Aerodynamics, pages 297–338. Springer, 2013.

[167] Eric A. Dow and Qiqi Wang. Simultaneous Robust Design and Tolerancing of Compressor Blades. In *Proceedings of ASME Turbo Expo 2014: Turbine Technical Conference and Exposition Volume 2B: Turbomachinery*, page 12, Düsseldorf, Germany, June 2014. ASME.

[168] Claudia Schillings and Volker Schulz. On the influence of robustness measures on shape optimization with stochastic uncertainties. *Optimization and Engineering*, 16(2):347–386, June 2015.

[169] T. W. Anderson. *An Introduction to Multivariate Stastical Analysis.* Wiley, 3rd edition, 2003.

[170] Llloyd N. Trefethen and David Bau. *Numerical Linear Algebra.* SIAM, 1997.

[171] A. Borzì and G. von Winckel. Multigrid Methods and Sparse-Grid Collocation Techniques for Parabolic Optimal Control Problems with Random Coefficients. *SIAM Journal on Scientific Computing*, 31(3):2172–2192, January 2009.

[172] Philipp Frauenfelder, Christoph Schwab, and Radu Alexandru Todor. Finite elements for elliptic problems with stochastic coefficients. *Computer Methods in Applied Mechanics and Engineering*, 194(2-5):205–228, February 2005.

[173] Young-Keun Choi and A. Banerjee. Tool path generation and tolerance analysis for free-form surfaces. *International Journal of Machine Tools and Manufacture*, 47(3-4):689–696, March 2007.

[174] Benjamin Peherstorfer. Multifidelity Monte Carlo estimation with adaptive low-fidelity models. *online pre-print*, 2017.

[175] Assefaw Hadish Gebremedhin, Fredrik Manne, and Alex Pothen. What Color Is Your Jacobian? Graph Coloring for Computing Derivatives. *SIAM Review*, 47(4):629–705, 2005.

[176] Michael B. Giles and Abdul-Lateef Haji-Ali. Multilevel nested simulation for efficient risk estimation. *arXiv e-prints*, February 2018.

[177] Robert Biedron and James Thomas. Recent Enhancements to the FUN3D Flow Solver for Moving-Mesh Applications. In *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, Reston, Virigina, January 2009. American Institute of Aeronautics and Astronautics.

[178] K. Stein, T. Tezduyar, and R. Benney. Mesh Moving Techniques for Fluid-Structure Interactions With Large Displacements. *Journal of Applied Mechanics*, 70(1):58, 2003.

[179] Zhi Yang and Dimitri J. Mavriplis. Mesh Deformation Strategy Optimized by the Adjoint Method on Unstructured Meshes. *AIAA Journal*, 45(12):2885–2896, December 2007.

[180] Konstantin Kovalev. *Unstructured Hexahedral Non-conformal Mesh Generation.* Phd thesis, Vrije Universiteit Brussel, Belgium, 2005.

[181] Shenren Xu, David Radford, Marcus Meyer, and Jens-Dominik Müller. CAD-Based Adjoint Shape Optimisation of a One-Stage Turbine With Geometric Constraints. In *ASME Turbo Expo 2015: Turbine Technical Conference and Exposition, Volume 2C: Turbomachinery*, pages 1–14. ASME, June 2015.

[182] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pages 517–524, New York, USA, 1968. ACM Press.

[183] Jeroen A. S. Witteveen. Explicit and Robust Inverse Distance Weighting Mesh Deformation for CFD. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, Reston, Virigina, January 2010. American Institute of Aeronautics and Astronautics.

[184] Laura Uyttersprot. *Inverse Distance Weighting Mesh Deformation: A Robust and Efficient Method for Unstructured Meshes*. Masters thesis, Delft University of Technology, 2014.

[185] Edward Luke, Eric Collins, and Eric Blades. A fast mesh deformation method using explicit interpolation. *Journal of Computational Physics*, 231(2):586–601, January 2012.

[186] Olaf Krzikalla and Georg Zitzlsberger. Code vectorization using Intel Array Notation. In *Proceedings of the 3rd Workshop on Programming Models for SIMD/Vector Processing - WPMVP '16*, pages 1–8, New York, USA, 2016. ACM Press.

[187] Andreas Griewank and Andrea Walther. *Evaluating Derivatives*. Society for Industrial and Applied Mathematics, January 2008.