

Robust Optimisation of Wing Aerostructural Response

Gustavo Pedro Krupa

Academic Year 2018–2022

A thesis presented for the degree of Doctor of
Philosophy

Supervisors: Dr. Timoleon Kipouros, Professor Mark Savill

May 2022



© Cranfield University 2022. All rights reserved.

No part of this publication may be reproduced without the written permission of
the copyright owner.

Abstract

The life cycle of an aeronautical product is characterised by an increasing design certainty and a decreasing in design freedom over time. The uncertainty in the design and in the manufacturing process may be quantified to assess its impact on the aircraft performance and on the likelihood of meeting the performance requirements, given the design margins and constraints. One of the major challenges in this process is the so-called *curse of dimensionality*, where the associated computational cost grows exponentially as a function of the number of random variables and intractable integrals. This challenge is addressed by exploring how recent advances in numerical methods and in machine learning might be used in uncertainty quantification and in robust optimisation.

This work investigates how to solve the following industrially relevant aeronautical problem: *How to perform the robust optimisation of wing twist (jig shape) to balance aerodynamic performance and risk of exceeding the design loads margins in the presence of structural uncertainties?* To address this problem, this thesis presents theoretical and algorithmic advances in probabilistic regression, uncertainty quantification and Bayesian optimisation, demonstrating and expanding its engineering applications. The primary aim is to formulate and solve the robust optimisation challenge using Gaussian processes and Bayesian optimisation of a wing subject to structural uncertainties and design load constraints. The problem is formulated as the evaluation of the impact of the wing jig shape twist angles, bending and torsional stiffness uncertainties on the overall lift-to-drag ratio, shear, moment and torque loads across the wingspan for different load cases and considering design load constraints. As secondary aim, a variational approach for inference is investigated since the required mathematical machinery is rarely tractable. It is concluded that probabilistic methods greatly expand the ability to learn and infer from data yielded by highly complex simulations.

Acknowledgements

First, I would like to express my gratitude to my supervisors Dr Timoleon Kipouros and my co-supervisor Professor Mark Savill, who have helped me in growing as a researcher and have given advice when needed, always providing me personal and professional guidance.

I would also like to offer greatly extended thanks to my two industrial supervisors, Simon Coggon and Sanjiv Sharma.

I must also thank Cranfield University staff for all the administrative support I have received over the course of my studies.

Lastly, I am grateful to God, my family and friends who have always supported me along this work.

This work is funded by a joint grant from Cranfield University and Airbus UK.

.

Contents

List of Figures	V
List of Tables	VII
List of Algorithms	VIII
1 Introduction	1
1.1 Background and Motivation	1
1.2 Industrial Problem Statement	5
1.3 Research Question and Thesis Contributions	9
1.4 Aims and Objectives	11
1.5 Thesis Outline	12
1.6 Appendices	14
2 Literature Review on Uncertainty Quantification and Robust Optimisation	15
2.1 Introduction to Robust Optimisation	17
2.2 Gaussian Process for Machine Learning	18
2.3 Robust Optimisation in the Bayesian Paradigm	25
2.4 Uncertainty Quantification and Management	30
2.5 Digital Transformation in the Aerospace Industry	39
2.6 Summary of Research Choices	48
3 Probabilistic Modelling in the Bayesian Paradigm	51
3.1 Introduction to Bayesian Paradigm	52

3.2	Bayesian Probabilistic Modelling	54
3.3	Bayesian Inference Engines	61
3.4	Validation and Verification	68
3.5	Model Checking	73
3.6	Formulation of the Robust Optimisation Problem	77
3.7	Summary and Conclusions	93
4	Flight Loads Evaluation Using Gaussian Process Emulators	94
4.1	Introduction	95
4.2	Gaussian Processes	100
4.3	Flight Physics	106
4.4	Results and Discussion	114
4.5	Summary and Conclusions	123
5	Flight Loads Evaluation Using Sparse Gaussian Process Emulators	125
5.1	Introduction	126
5.2	Sparse Gaussian Process Approximations	127
5.3	Results and Discussion	132
5.4	Summary and Conclusions	135
6	Robust Aerostructural Optimisation for Improved Lift-to-Drag under Design Load Margin Constraints	149
6.1	Introduction	150
6.2	Bayesian Optimisation	150
6.3	Acquisition Functions	152
6.4	Sensitivity Analysis	156
6.5	Results and Discussions	158
6.6	Summary and Conclusions	167
7	Conclusions and Recommendations	168
7.1	Conclusions	169

7.2	Recommendations for Future Work	171
A	Miscellaneous Material	179
A.1	Mathematical Background	180
A.2	Gaussian Process Latent Variable Model	183
A.3	Model-based Bayesian Framework and Graph-based Optimisation . . .	185
A.4	Software Engineering of the UQOPT Code	190

List of Figures

1.1	The process of wing assembly	6
1.2	The two possible scenarios for setting the tooling	6
1.3	Shear, Moment and Torsion unravelled by the Principle of Virtual Work [1]	7
1.4	Overview of the Dataset	9
2.1	Bayesian optimisation represented as a UML flowchart	26
2.2	Agile System Engineering process for aircraft conceptual design	43
2.3	A factor graph representation	46
3.1	Overview of the Dataset	78
3.2	Aircraft Axis System	79
3.3	Twist angles at two span-wise locations	80
3.4	Typical Mission Profile	80
3.5	Tree overview of the dataset	84
3.6	Design load margins	88
3.7	Design load margin versus wing stations	89
4.1	Boxplot of the model parameters	121
4.2	Correlation matrix	122
4.3	Independent parameters versus correlated parameters through the use of a Gaussian copula.	122
5.1	Calculated shear loads for 21 maneuver cases along the wingspan using sparse GP approximation	137

5.2	Calculated shear loads for 21 maneuver cases using sparse GP approximation continued	138
5.3	Calculated moment loads for 21 maneuver cases along the wingspan using sparse GP approximation	139
5.4	Calculated moment loads for 21 maneuver cases using sparse GP approximation along the wingspan continued	140
5.5	Calculated torque loads for 21 maneuver cases using sparse GP approximation along the wingspan	141
5.6	Calculated torque loads for 21 maneuver cases using sparse GP approximation along the wingspan continued	142
5.7	Calculated shear loads for 25 gust cases using sparse GP approximation along the wingspan continued	143
5.8	Calculated shear loads for 25 gust cases using sparse GP approximation along the wingspan continued	144
5.9	Calculated moment loads for 25 gust cases using sparse GP approximation along the wingspan	145
5.10	Calculated moment loads for 25 gust cases using sparse GP approximation along the wingspan continued	146
5.11	Calculated torque loads for 25 gust cases using sparse GP approximation along the wingspan	147
5.12	Calculated torque loads for 25 gust cases using sparse GP approximation along the wingspan continued	148
6.1	Performance of acquisition functions	161
6.2	Sobol indexes	162
6.3	Sobol index total effects	162
6.4	Robust optimisation of the wing aerostructural response using probability of improvement acquisition function	163
6.5	Points that violated the design load constraints	164
6.6	θ_1 versus $\Delta(L/D)$	165

6.7	θ_2 versus $\Delta(L/D)$	165
7.1	Causal information stored in a graph data structure	173
A.1	The proposed Engineering Bayesian framework	185
A.2	Directed multi-graph used to model a Bayesian hierarchical system . .	187
A.3	Problem solving Environment	188
A.4	UML activity diagram describing the main use case	191
A.5	UML class diagram showing the code structure	192
A.6	UML class diagram showing the Probabilistic Regression class	192
A.7	UML class diagram for the Bayesian Optimisation class	192
A.8	UML class diagram for the Uncertainty Quantification class	193
A.9	UML class diagram for the Sensitivity Analysis class	193
A.10	Directory structure for the UQOPT code	194

List of Tables

3.1	Locations of the wingstations contained in the dataset	79
3.2	Types of loads contained in the dataset	80
3.3	Aerodynamic performance improvement values contained in the dataset	81
3.4	Loads contained in the dataset	82
4.1	Estimate of number of load cases required	107
4.2	Input parameters of the surrogate model	116
A.1	Continuous probability distributions	181

List of Algorithms

2.1	Bayesian Optimisation Algorithm	27
3.1	Monte Carlo Hamiltonian Algorithm	68
4.1	Gaussian Process Regression Algorithm	120
6.1	Bayesian Optimisation Algorithm	152
6.2	Expected Improvement	154
6.3	Probability Of Improvement	155
6.4	Negative Lower Confidence Bound	156
6.5	Sobol indices	158
6.6	Constrained Bayesian Optimisation Algorithm	159
A.1	Modelling with a GP-LVM	184

Nomenclature

Abbreviations

AHPO Automated Hyperparameter Optimization

AutoML Automated Machine Learning

BO Bayesian Optimisation

CDF Cumulative Distribution Function

CE Concurrent Engineering

CFD Computational Fluid Dynamics

CV Cross Validation

DfSS Design for Six Sigma

DOE Design of Experiment

EDA Exploratory Data Analysis

EHI Expected Hypervolume Improvement

EHVIG Expected Hypervolume Improvement Gradient

EI Expected Improvement

ELPD Expected log predictive density for new data points

ELPPD Expected Log Pointwise Predictive Density

ELPPD Expected log point-wise predictive density for a new dataset

FEM Finite Element Method

FoMs Figures of Merit

FSL Few-shot learning

GP Gaussian Process

GPLVM Gaussian Process Latent Variable Model

HMC Hamiltonian Monte Carlo

IPDTs Integrated Product Development Teams

IQs Interesting Quantities

LHS Latin Hypercube Square

LOO Leave-One-Out

LPD Log Predictive Density

LPPD Log Pointwise Predictive Density

LSL Low-shot learning

LVM Latent Variable Model

MAP Maximum A posteriori Principle/Probability

MBSE Model-based systems Engineering

MCMC Markov Chain Monte-Carlo

MCS Markov Chain Simulation

MDA Model-driven architecture

MSE Mean Square Error

MTL Multi-task learning

NUTS No-U-Turn Sampler

OEM Original Equipment Manufacturer

OMG Object Management Group

PCA Principal Component Analysis

PDF Probability Distribution Function

PES Predictive Entropy Search

PP Probabilistic Programming

SA Sensitivity Analysis

SBCE Set-Based Concurrent Engineering

SBD Set-Based Design

SMT Shear, moment and torque

TPDS Toyota Product Development System

TPMs Technical Performance Measures

UCB Upper Confidence Bound

UML Unified Modelling Language

UQ Uncertainty Quantification

UQ&M Uncertainty Quantification

UQOPT Uncertainty quantification and robust optimisation code

V&V Validation and Verification

VI Variational Inference

Engineering notation

σ Compressible stress

M Moment

S Shear force

T Torque

Mathematical notation

$\alpha(\cdot)$ Acquisition Function

$\mathbf{X}$ Set of all input vectors $\mathbf{x}$

$\mathbf{x}$ Vector representing the inputs in a D dimensional space

$\mathbf{x}_n$ The n th input vector in X

$\mathbf{y}$ Vector of outputs

ϵ Standard deviation of Gaussian noise

$\mathbb{E}$ Statistical expectation

$\mathbb{R}$ Real numbers

$\mathbb{R}^D$ D -dimensional vector of real numbers

$\mathbb{V}$ Statistical Variance

$\mathcal{D}$ Data/dataset containing the vector of inputs $\mathbf{x}$, and the vector of outputs, $\mathbf{y}$

$\mathcal{N}$ Normal distribution

$\mathcal{V}$ Statistical Variance

$\mathcal{X}$ Set of all load cases under consideration

μ Population mean

Σ	Covariance matrix
σ	Population standard deviation
Θ	Set of all hyper-parameters
θ	Vector of hyper-parameters
$f(\cdot)$	denote a function or a compositional relation or a particular evaluation at $\mathbf{x}$ as $f(\mathbf{x})$
$k(\cdot, \cdot)$	Kernel function
$m(\cdot)$	Mean function
$P(\cdot)$	Probability of
x_d	for the d th dimension of a general input $\mathbf{x}$
x_{nd}	A specific dimension will be denoted using the subscript d , i.e. x_{nd} for the d th dimension of the n th input
ATS	Authoritative truth source
EAS	Estimated air speed
GPRN	Gaussian process regression network
KL	Kullback-Leibler divergence

Superscripts

$'$	used to generate more variable names for similar things avoiding other subscripts
$^\top$	Transpose of a matrix
d	Dimension of a set

Chapter 1

Introduction

1.1 Background and Motivation

Modern engineering design and analysis relies on optimisation as a concept and as a method. The word optimisation has its origins in the Latin word *optimus*, which means “best” or “very good”¹. Etymologically, it is connected to a family of words which include *optio*, *ōnis* (choice, option) and *opus*, *ĕris* (work), and it is the superlative of *bonus* (good), clearly conveying the underlying idea of “the best amongst the choices”². As a concept, the term optimisation is often used to mean “improvement”. Mathematically, however, the term optimisation has a precise meaning: the process of finding the *best* possible solution by changing the control variables, usually subject to constraints. Interesting enough, and authentic to its etymological roots, the process of optimisation can be applied in all domains where decisions are a constituent part of the problem. Given this broad range of applicability, it is natural that there are different types of optimisation classified according to its problem formulation, its objective and constraint functions characteristics [2].

The adoption of optimisation in the design and analysis of engineering systems and product life cycles has the benefit of accelerating the design cycle and obtaining better performance and results. To appreciate the benefits of optimisation in engineering design

¹P. G. W. Glare, *Oxford latin dictionary*. 1982

²Alfred Ernout and Antoine Meillet, *Dictionnaire étymologique de la langue latine: histoire des mots*. 1951

and analysis, one may consider the conventional engineering design and analysis process. In this conventional way, the first step is to gather requirements and specifications, which can come from a variety of sources. Based on these requirements, a number of design proposals are evaluated using simplified models and engineering intuition. This phase is referred as conceptual design phase. The next phase is the preliminary design phase. The goal of the preliminary design phase is to evaluate the design and its subsystems in a more refined way. Usually, after the preliminary design phase the product is “frozen”, that is, changes are not welcomed since they equate to additional costs and rework. It is important to notice, that all these different design phases require lots of design interactions. The advantage of design and analysis optimisation is that for a given well-posed optimisation problem, it is possible to perform the iterative design cycle in a mathematical way, seeking an optimal set of controlled variables giving a certain set of requirements [2].

Optimisation is also the core of data-driven machine learning methods. The discipline of machine learning has changed the industrial and scientific landscape and is considered a pillar of the so-called fourth Industrial Revolution [3]. The data-driven aspect of machine learning is intended not to supplant, but to complement the theoretical, experimental and numerical modes of scientific inquiry and engineering. Many of these methods are characterised by its ability to handle high-dimensionality and by its improvement with higher volumes of data, showing a great potential to solve optimisation problem in aeronautical design and high-value manufacturing. The aerospace industry is particularly heavily dependent on high-dimensional data during each phase of the product life cycle. This fact makes aerospace as a promising field for the development and application of machine learning and data-driven technologies.

The current trend of data-driven models for aircraft design and maintenance is triggering a digital transformation, in which it becomes feasible to create a digital representation of the entire product life cycle that, in principle, allows several improvements to the design and manufacturing, while fostering cost reduction. Historically, optimisation methods have enabled the design of aircraft and manufacturing through

physics-based simulations that resulted in substantial cost savings to the original equipment manufacturers (OEM) and in the Fourth Industrial Revolution they are going to play an even bigger role in the product life cycle.

Machine learning potentially can positively impact and support all the life cycle phases of the development and commercialisation of an aeronautical product [4]. The product development can certainly benefit from the high volume of quality data yielded by complex and expensive computer simulations. Machine learning can also support the production phase by improving the automation and safety of the manufacturing process through the harnessing of data to improve the process control. Thus, expanding the overall manufacturing capabilities. The widespread adoption of machine learning and its regulatory acceptance is going to revolutionise how systems are designed and provide new capabilities in human-machine interaction such as advanced design interfaces, interactive visualisations, and improved natural language interactions [4].

There are several challenges that need to be addressed before a fully fledged integration of data science and machine learning can be realised. In the aerospace industry, customers are varied and heavily dependent on expensive certification procedures required by the regulatory authorities. Essentially, due to the safety-critical aspect of aerospace engineering, data-driven models must be certifiable, verifiable and generalise beyond the training data. They also must be interpretable and explainable by humans [5]. For this reason, the aerospace industry is going to require a level of robustness and safety that is typically not required in other industries [6]. Despite these challenges, particularly related to the future regulatory framework, the benefits of the application of machine learning have been already demonstrated in a number of applications [4].

Industrial applications of optimisation are always subject to uncertainties that cannot be fully captured in physics simulations. It is essential when investigating a problem to understand the role that the uncertainties involved play in the input parameters and in the output results. The source of uncertainties are normally classified into two categories, namely, aleatoric and epistemic uncertainties [7]. The aleatoric

uncertainties are the uncertainties which cannot be reduced because one simply does not have control over them. The epistemic uncertainties are uncertainties which could be reduced by more accurate measurements and better modelling of the physics. The distinction is not sharp because it depends on the problem setting and on subjective judgement. However, for many analysis or designs, these uncertainties can either be included in the design as a factor of safety or as design margin. Thus, if the goal is to optimise a design or a high-value manufacturing process, it is of great benefit to include uncertainties and design margins as part of the problem formulation. This fact is particularly important to high-value manufacturing problems, where due to the strict tolerance margins required for aerospace manufacturing, the applications are often formulated as a constrained optimisation problem involving tightly coupled disciplines with the need to quantify the uncertainties involved.

There are a multitude of frameworks for quantifying uncertainties, with perhaps the most widely known of these being the theory of probability. In this thesis, the Bayesian paradigm is adopted. Its foundational text is *Probability Theory: The Logic of Science* by Jaynes [8]. An illuminating review of this important text is given by Robert et al. [9]. Classically, probabilities are interpreted as statistical frequencies of the outcome of experiments. This representation in terms of frequencies presents conceptual difficulties in problems where the associated events are not repeatable. In such cases, it is desirable to have a mathematical apparatus that allows a “degree of belief” interpretation.

On the other hand, in the Bayesian point of view both types of uncertainty are represented in a way that does not depend on the problem setting. Normally, the uncertainties are mathematically represented by random variables and associated with a particular probability distribution. They can be taken into account in a problem by assumption or by inference. By assumption means that the probability distributions are assigned to the parameters relying on experience and expert judgement. On the other hand, by inference usually means that the probability distributions are reconstructed using measurements of the interesting quantities (IQs).

1.2 Industrial Problem Statement

The industrial problem consists of optimising the wing “jig shape”³, in the presence of structural uncertainties, to balance the expected improvement on aerodynamic performance against the risk of exceeding the design loads margins constraints. As it is known from flight physics, the effect of the wing jig twist has opposing impacts on the confidence of achieving the aeroperformance and on the design margin constraints [10].

This problem originated in the aeronautical industry, where every aircraft design programme is unique in terms of programme schedule. For such cases, the timing of the design freeze of the tooling is very important, because it can have a huge impact in the programme lead times. In manufacturing and supply chain management, the term *lead time* is defined as the latency between the initiation and completion of a process [11]. The tooling required for wing manufacturing is very expensive and must be specifically designed for each aircraft product requirements that are dependent on the selected design margins. An example of the tooling required for building the wing at the right angle in the fuselage is shown in figure 1.1. In particular, there are two different scenarios related to the setting of the design margins and the fixing of long lead time tooling design for the wing “jig shape”. In the first scenario, the tooling is frozen prior to the setting of the design load margins. This scenario is often the case for long lead time items. In the second scenario, the tooling is frozen after the setting of the design load margins.

This thesis presupposes the second scenario where the tooling is still to be fixed and therefore there is an opportunity to account for uncertainties through a robust late optimisation during the final setting of the tooling, as in figure 1.2. In this scenario, the design has matured since the design margins have been set resulting in a change in the uncertainties, usually a reduction. In this case, the wing structural uncertainties due to the “incomplete” analysis and the incorporation of all the detailed design considerations and constraints still exist. However, there is the advantage that the application of design margins de-couples the multidisciplinary feedback aspects of the structural

³Unload and as-built shape.



Figure 1.1: The process of wing assembly

impacts and sizing. These multidisciplinary feedback aspects are replaced by design loads margins constraints. Thus, the problem is simplified and amenable to analysis.

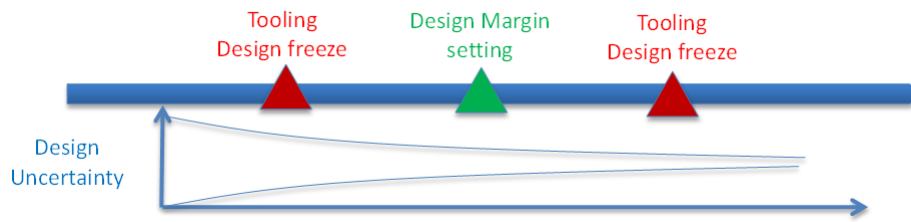


Figure 1.2: The two possible scenarios for setting the tooling

As a general rule, a knowledge of structural design methods and an understanding of load paths are essential to the progress of an aircraft programme. The load paths are the result of external applied loads, which must be determined with precision and accuracy. The detailed structural sizing requires an estimation of load paths. Generally a loading system may be simplified to a combination of up to four load types: axial load, shear force, bending moment, and torsion. These forces produce corresponding internal force systems, as illustrated by figure 1.3.

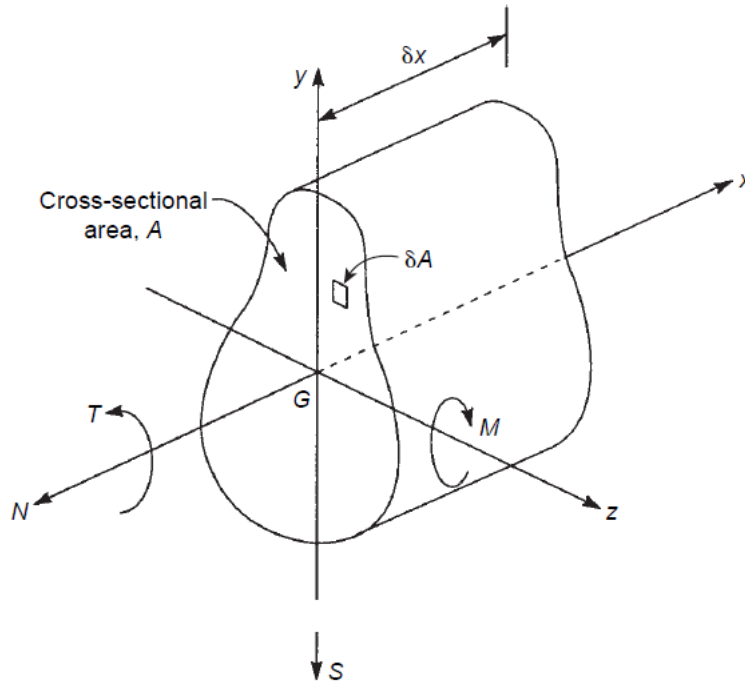


Figure 1.3: Shear, Moment and Torsion unravelled by the Principle of Virtual Work [1]

In figure 1.3, the shear force, S , acts on the member section to produce a distribution of vertical shear stress that depends upon the geometry of the cross-section. The bending moment, M , acts on the member section to produce a distribution of direct stress, σ , through the depth of the member cross-section and the torque, T .

The prediction of loads is a major programme driver and as its uncertainty decreases, it is still possible to take it into account and obtain an aerodynamic improvement. It is estimated that ten percent of the engineering hours spent on a new airframe design project are chargeable directly to the obtaining of design loads [12]. Therefore, the prediction of loads is extremely important and relevant in all programme phases.

The calculation of flight loads is computationally expensive and time-consuming, requiring different models and strategies, along with programme management tasks. Traditionally, the sizing process starts with hand calculations and first-principled approaches detailed in structural handbooks and progress to computational methods capable of very detailed analysis. Such analysis are performed using commercial finite-element method (FEM), for structural analysis, and computational fluid dynamics method (CFD), for aerodynamic load analysis. This engineering cycle featuring

FEM-CFD analysis usually generates the necessity of extensive re-work in the aircraft definition. As the programme moves forward re-working costs tend to become prohibitive. Therefore, recently there has been a context for methods that are able to make decisions under uncertainty earlier in the programme. This robust decision capability coupled with surrogate modelling brings many advantages, since it provides a reduction in the associated uncertainties and allows one to avoid the computational and time-consuming tasks related to more computational expensive methods such as FEM-CFD.

The very powerful analysis methods of FEM and CFD are often seen as expensive methods, because of the amount of computational resources they require. For many tasks, such as uncertainty quantification, where it is necessary to repeat the same calculation many times, the traditional methods of FEM and CFD are often prohibitive. Many of these tools are commercial packages and therefore not amenable to intrusive uncertainty quantification approaches, which require direct access to the source-code of the modelling package. One possible solution is to employ surrogate models that can leverage the availability of machine learning methods and use it for uncertainty quantification and robust optimisation.

Surrogate methods allow one to capture input-output relations from a variety of sources and build it in a “black-box” model. This model can be abstractly expressed as simple as:

$$\mathbf{y} = \hat{f}(\mathbf{x}) \tag{1.1}$$

where $\mathbf{x}$ is the input, $\mathbf{y}$ is the output and the ordered pair $\{\mathbf{x}, \mathbf{y}\} \in \mathbb{R}^d$, d being the number of dimensions. Although surrogates are intended to “mimic” the input-output pair behaviour of a physical or computational experiment, it is rarely possible to obtain a complete replacement of the engineering process of interest. Thus, there will be always an error associated with the prediction performed by the surrogate. This is referred as predictive uncertainty in the specialised literature[9]. Mathematically, one

can write for a given surrogate:

$$f(\mathbf{x}) = \hat{f}(\mathbf{x}) + \epsilon \quad (1.2)$$

where ϵ is the error (or residual) between the surrogate and the physical or computational experiment at $\mathbf{x}$. As later chapters show, Gaussian Processes (GP) have mathematical properties that makes them particularly useful for the applications of uncertainty quantification and robust optimisation.

As it was mentioned in the previous section, the recent advances in machine learning can be leveraged to tackle engineering problems that require advanced uncertainty quantification and robust optimisation methods. Thus, this thesis adopts the data-driven engineering method approach to solve the robust optimisation challenge provided by Airbus UK. A quick overview of the dataset provided and the features involved is shown in figure 1.4. In section 3.6 of chapter 3, the complete problem formulation along with all its related mathematical statements is presented.

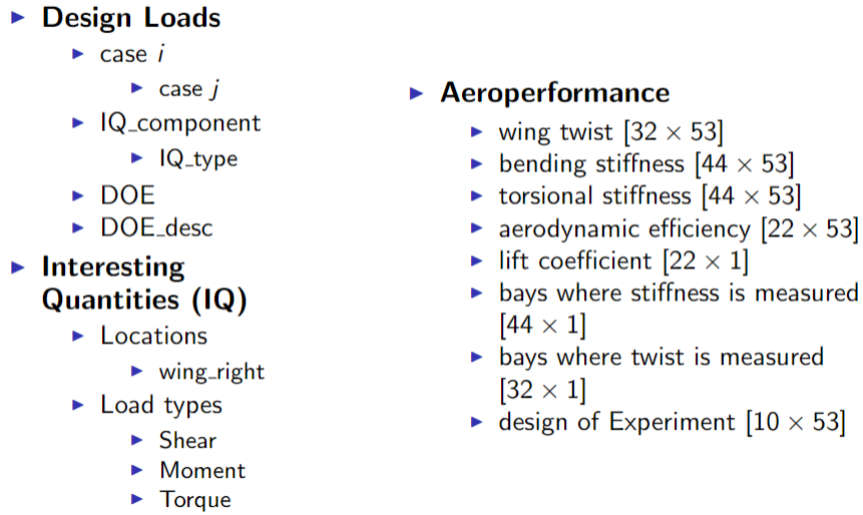


Figure 1.4: Overview of the Dataset

1.3 Research Question and Thesis Contributions

This dissertation is motivated by the need to apply uncertainty quantification and robust optimisation to an industrially relevant problem in the aerospace industry. Thus,

this work is intended to answer the following research question:

How to perform the robust optimisation of wing twist (jig shape) to balance aerodynamic performance and risk of exceeding the design loads margins in the presence of structural uncertainties?

The contributions of this dissertation are mostly related to the application of state-of-the-art probabilistic machine learning tools, i.e, robust emulators and optimisation under the presence of uncertainties to an industrially relevant problem of wing optimisation. These contributions are summarised as follows:

1. Assessment of the impact and comparison of several robust emulators and Bayesian Optimisation techniques on the robust optimisation a wing subject to structural uncertainties and design load constraints. The problem is formulated as having the wing jig twist as design variables and the bending and the torsional stiffnesses as uncertainty parameters, which varies according to manoeuvre and gust loads given as a function of the aircraft lift coefficient.
2. Investigation of variational approaches for Gaussian Processes. In particular, fully independent training conditional (FITC) as alternatives to the standard Markovian methods. This contribution is motivated by the necessity of scaling uncertainty quantification and robust optimisation methods for higher volumes of data, as prescribed by the fourth Industrial Revolution and digital transformation paradigm.
3. Investigation of the synergies and inter-dependencies of a number of design parameters, such as the wing sectional twist, bending and torsional stiffnesses on the overall aircraft lift-to-drag ratio and on the propagation of structural design uncertainties effects on shear, moment and torque loads across the wingspan of different load cases.
4. A Bayesian decision-making framework and toolbox directly applicable to aerospace engineering, in particular, to novel aircraft designs where often the technology employed is not mature enough. This Bayesian frameworks replaces values, variables

and parameters by probability distributions, allowing early uncertainty associated with early stages of a project or uncertainty related to novel technologies to be assessed in the conceptual stage along with potential trade-offs assessments to be made.

1.4 Aims and Objectives

The inherent uncertainty in the aircraft design process is usually accounted in the design margins. This practice helps to avoid costly design iterations. A dataset containing representative data of this process was provided by Airbus UK. Briefly, this dataset comprises lift-to-drag improvement (sometimes called aeroperformance in the industrial jargon) and design load data for several aircraft flight conditions. A detailed description of this dataset is given in section 3.6 of the chapter 3. Based on this dataset, the overall aim of this research consists of optimising the wing jig shape in the presence of uncertainties in the wing structural stiffness, to balance the aerodynamic performance against the risks of exceeding the design loads margins constraints.

The above-mentioned aim is achieved through four steps. The first step is to survey the scientific literature in aerospace engineering and computer science, in order to understand which probabilistic paradigm and set of methods are best applicable to the robust optimisation problem. The second step is to formulate in a mathematical way the robust optimisation problem based on the parameters contained in the dataset. The third step is to use this dataset to train and validate computational surrogates that accurately model the relations between the input and the output data contained in the dataset. These surrogates must also be *robust*, in the sense that they are able to output results with its uncertainties quantified. Thus, taking into account the inherent uncertainties in the design process. The fourth and last step is to perform the robust optimisation and understand the sensitivities of the lift-to-drag expected improvement to the input parameters, representing different scenarios in regards to the structural uncertainties and the design load margins constraints. Therefore, the formal objectives of this work are stated as follows:

1. To survey probabilistic regression, uncertainty quantification and robust optimisation methods.
2. To formulate, to solve and to validate the robust optimisation problem and - through discussions with Airbus - align the research work with the industrial needs and constraints.
3. To understand the problem sensitivity to the input parameters within the robust optimisation in relation to the structural uncertainties and design load constraints.
4. To develop a way to visualise the analysis with the intent of enabling engineers to make a design decisions in a risk/confidence⁴ framework.

1.5 Thesis Outline

This thesis is organised in seven chapters. Chapter 2 aim is twofold. First, to conduct a detailed literature survey identifying the state- of-the-art of uncertainty quantification and Bayesian (robust optimisation) methods. Second, to present the theoretical directions that guided the research work documented in this dissertation. This survey examines the most up-to-date contributions to the field and identifies the gaps in the current knowledge. In the last section of this chapter, the place where this research fits in the knowledge is restated in the context of aeronautical applications of uncertainty quantification and robust optimisation.

Chapter 3 introduces the fundamentals of probabilistic modelling under the Bayesian paradigm. The central concepts that underlie all the mathematical methods presented in this thesis are thoroughly explained in this chapter, avoiding tedious repetition in latter chapters. Special attention is given to the general presentation of the concept of variational inference in the light of the recent research progress. Based on this knowledge, the robust optimisation problem introduced in this first chapter is formulated in its mathematical terms.

⁴This is achieved by using probabilistic methods

Chapter 4 investigates the efficiencies the trade-offs and the limitations of different robust emulators that were identified in the literature review of the previous chapter. The word “robust” means, in our usage, both reliable and subject to uncertainties. A subset of surrogate models called Gaussian processes is investigated. These robust emulators are bench-marked using numerical experiments on canonical examples. The performance benefits and kernel design trends for the former are investigated and compared for single-output Gaussian processes. Additionally, chapter 4 also provides an overall and practical discussion of the cross-validation procedures and the validation methodology employed with the robust emulator evaluations performed in this work.

Chapter 5 introduces sparse Gaussian processes and explains the concept of variational inference (VI). A single scheme of VI approximation is introduced, i.e, fully Independent Training Conditional (FITC), with the goal of generating sparse Gaussian processes. These methods are used to build robust sparse emulators capable of evaluate the shear, moment and torque loads across the wingspan.

In the Chapter 6, based on the previous chapters, the problem that motivated this thesis - the robust aerostructural optimisation of a wing considering structural uncertainties and design load margins constraints is reformulated, solved and visualised. Firstly, this chapter presents the concepts, terminology and notation associated with Bayesian Optimisation, as well as the general Bayesian optimisation algorithm. The performance of a Bayesian algorithm is dependent not only in the nature of the problem being investigated, but in the acquisition function used as well. Therefore, in this chapter we examine the Bayesian Optimisation procedure with following acquisition functions: a) Entropy Search, b) Expected Improvement, c) Negative Lower Confidence Bound and d) Probability of Improvement. Lastly, questions related to sensitivity analysis are comprehensively discussed and ways to improve robust optimisation are pointed out.

Finally, Chapter 7 summarises the conclusions draw throughout this work and highlights directions for future developments.

1.6 Appendices

The appendices contains the material destined to supplement the main text and some theoretical and practical aspects related to this research. Appendix A.1 complements the text by giving some mathematical definitions that are used through out this work. The Gaussian Process Latent Variable Model is presented as a proof of concept for its use in aerospace engineering in the appendix A.2 for the visualisation of the high-dimensional data. Appendix A.3 explains how the methods used in this thesis may be used as the backbone of a probabilistic (Bayesian) engineering framework that is suitable for the decision-making and early conceptual design process of a novel aircraft. This framework is mathematically introduced and its current limitations are highlighted and directions for further improvements are given. Appendix A.4 documents the use of software engineering the development of the UQOPT code, featuring the methods presented in this thesis.

Chapter 2

Literature Review on Uncertainty Quantification and Robust Optimisation

The most relevant contributions to the state-of-the-art of uncertainty quantification and robust optimisation and its applications in aerospace are reviewed in this chapter. The emphasis is given on a concise and self-contained overview of different types of methods used for uncertainty quantification and robust optimisation compatible with the Bayesian paradigm. This chapter assumes familiarity with basic statistical concepts such as expected value $\mathbb{E}$, variance σ , probability distribution functions (PDFs) and cumulative distribution functions (CDFs). Detailed explanations of these concepts are presented in the appendix A.1, if needed. The sections are structured in a way compatible with the engineering Bayesian framework described in section A.3 and in the rational underlying the code structure described in section A.4.

This chapter is organised as follows. In section 2.1, the different approaches to the robust optimisation problem are surveyed and the general approach to robust optimisation in aerospace and of a wing under structural and material uncertainties is reviewed in the light of published scientific literature. In section 2.2, a machine learning method (Gaussian process) suited for uncertainty quantification is reviewed

under the title “Gaussian process for machine learning”. These are the main techniques used to create emulators (surrogate models) that enable probabilistic modelling, uncertainty quantification and Bayesian optimisation to be carried out. Section 2.3 presents a Bayesian optimisation technique that is applicable to a broad variety of engineering problems. This presentation focuses on the basic problem formulation and its mathematical solutions. Under section 2.4, methods that may be labelled when taken together as uncertainty quantification & management (UQ&M) are reviewed. These methods include dimension reduction, Monte Carlo methods, copula theory, sensitivity analysis and uncertainty propagation. Discussion about the advantages and disadvantages of each method along with its application context are dealt with on later chapters. The current gaps in the knowledge that are required for new fields such as digital-twins, model-based systems engineering, set-based concurrent engineering, model-based machine learning, automatic machine and probabilistic programming are discussed in section 2.5. Finally, this chapter ends with a summary of the research choices that guided this research and are discussed in the next chapters.

2.1 Introduction to Robust Optimisation

Uncertainty is ubiquitous in engineering design and analysis. Not only uncertainty is present on any measurements, but also in some types of assumptions that engineering models rely on. Manufacturing processes and the aging of any system introduce deviations from the specifications and the operating conditions always vary from the nominal ones. Moreover, some parameters are inherently unstable and many physical phenomena, such as turbulence, are physical instabilities [13]. Hence, in general, any system will be affected by a wide range of uncertainties. Under this perspective, one can adopt uncertainty metrics that are often referred as interesting quantities (IQ) to quantify the system's response to uncertain design variables and parameters. In such cases, performing an optimisation with deterministic inputs often results in sub-optimal performance. A more suitable approach that often leads to optimal results is to include random parameters in the design and in the control variables of the optimisation problem.

Engineering optimisation often distinguishes between a *robust* design and a *reliable* one. A system or design is robust when the parameters of engineering interest are not sensitive to the inherent variability of the parameters. It is reliable when it is less prone to violating a constraint when accounting for variability [2]. In this work, these two concepts are dealt with under the umbrella term of robust optimisation.

Robust optimisation is defined as a field of optimisation theory that seeks to find an optimal solution given the assumptions, in which a measure of robustness is sought against uncertainty in its mathematical formulation. In practical engineering, the robust optimisation approach is favoured when solving problems where robustness and reliability are deemed more significant than nominal result values or metrics. The advent of state-of-the-art machine learning algorithms and the increase in computational power has sparked an interest in leveraging machine learning techniques for the purpose of robust optimisation. Thus, taking robustness into account is a strong a design trend and a rich field of research in both engineering optimisation and machine learning.

In a robust optimisation, the robustness of the objective function reduces the

sensitivity to the uncertainties in the performance of a given system, while the robustness of the constraint function is determined by the feasibility condition that guarantees that the solution with its uncertainties lies in the feasible region [14]. Previous studies [14–20] have used aerospace applications to evaluate robust optimisation methods. Generally, robust optimisation methods find an optimum by constructing a robust counterpart to the original objective function. The robust counterpart function can be formulated based on different philosophies and therefore may result in different categories of robust optima.

2.2 Gaussian Process for Machine Learning

Machine learning has been used with success in a variety of problem of several domains [21–27]. Technological domains of interest where machine learning plays an important role are image classification, object identification and tracking, motion planning, anomaly detection and control, and natural language processing. As machine learning methods have improved in their capability and scope, ML has become a way with extremely good potential for the use in the aerospace industry, both in analysis and in design [4]. This potential is measured in terms of execution time, speed, robustness, deployment time and the amount of “human engineering” required [4].

Machine learning algorithms can often be categorised in three types: unsupervised, supervised and reinforcement algorithms [28]. Unsupervised learning algorithms work with a fixed dataset that contains many features and learn useful properties of the dataset structure. On the other hand, supervised learning algorithms also work with a fixed dataset containing features, but each outcome is associated with a label or a target. Reinforcement learning algorithms interact with an environment in a way that there is a feedback loop between the learning system and its outcome. Based on this classification, probabilistic regression fits the category of a supervised learning problem.

Gaussian processes (GP) are a family of Bayesian models that characterise distributions of functions [29]. Gaussian processes are attractive regression models because of their Bayesian formulation, which allows the uncertainty quantification in the pre-

dictions (or outputs). The uncertainty to be quantified may be the result of both intrinsic noise and/or errors in the parameter estimation procedure used in the problem. Gaussian process regression is non-parametric and hence can model essentially arbitrary functions of the input points. Gaussian processes are defined by two parameters: the mean function and the covariance function. The covariance function is a mathematical kernel [29]. A positive-definite kernel is a generalisation of a positive-definite function or a positive-definite matrix, that arise in diverse parts of mathematics [30]. Kernel methods operate in a high-dimensional, implicit without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space. This operation is often computationally cheaper than the explicit computation of the coordinates and it is often referred to as the kernel trick [30]. By selecting an appropriate kernel, the GP regression model may take advantage of structure in the data for better accuracy and faster computation. In the next subsections important topics as how the different formulations work out the hyper-parameters are intentionally omitted in the sake of brevity.

2.2.1 Single-Output Gaussian Process

There are several ways to interpret a Gaussian process [29]. In this work, a Gaussian process is understood as a distribution over functions. In this perspective, the Gaussian process is entirely described by its mean function, μ , and covariance function k . A GP prior consists of the mean and the covariance function. Thus, the GP prior defines that for any collection of n inputs $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T$, the corresponding function values

$$\mathbf{f} = (f(\mathbf{x}_1), \dots, f(\mathbf{x}_n))^T \in \mathbb{R}^n \quad (2.1)$$

follow a multivariate Normal distribution

$$\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{k}) \quad (2.2)$$

Usually, the mean is taken as zero. Thus, a zero-mean Gaussian process prior on latent function $f(\mathbf{x}) \in \mathbb{R}$ and its covariance function $k(\mathbf{x}, \mathbf{x}')$ of a stochastic process $f(\mathbf{x})$ are defined as

$$f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}')) \quad (2.3)$$

The covariance function is also referred as kernel. The kernel matrix encoding the function covariances is denoted by $\mathbf{k} = (k(\mathbf{x}_i, \mathbf{x}_j))_{i,j=1}^n \in \mathbb{R}^{n \times n}$. A key property of GPs is that output predictions $f(\mathbf{x})$ and $f(\mathbf{x}')$ correlate according to the similarity of the inputs $\mathbf{x}$ and $\mathbf{x}'$ as defined by the kernel $k(\mathbf{x}, \mathbf{x}') \in \mathbb{R}$. Thus, the modelling capacity is determined by the chosen kernel.

Gaussian processes can be developed to accommodate several applications, since it is a mathematical concept that can be used for classification and regression supervised problems. GP also possess several extensions that broadens its capability. Among many interesting extensions to the standard Gaussian process as defined by the fundamental reference (*Gaussian process for machine learning* [29]), there are four of particular importance to this research: Detecting and quantifying periodicity of a function, optimisation of GPs training using an exploration-exploitation trade-off, multi-fidelity capability and GPU acceleration of GP training.

According to Durande [31], GP can be used to tackle the problem of detecting and quantifying the periodicity component of a function that is being queried under noise, forming up a limited number of input/output pairs. In their work, the authors introduced a novel decomposition of the covariance function as the sum of periodic and aperiodic kernels. This decomposition allows for the creation of sub-models which capture the periodic nature of the signal and its complement. A somewhat similar application in active learning and sensing was done by Ling [32], in which he used a Gaussian Process framework endowed with a general class of Lipschitz continuous reward functions and Bayesian optimisation criteria, allowing the optimisation of GP training using an exploration-exploitation trade-off [33].

In many engineering applications, it is necessary to evaluate an expensive black box function f . However, in many cases, it is possible to utilise cheap approximations

of f via computer simulation or through surrogates. This process is refereed in the literature as multi-fidelity. Kandasamy [34] proposes an approximation to eliminate low function value regions cheaply and use the expensive evaluations of f in a small but promising region and speedily identify the optimum. Gaussian process regression can also be utilised in exploration and exploitation. Lastly, in order to increase the speed of computations, Dai [35] developed a formulation for Gaussian Process Models that allows parallelisation and GPU acceleration by exploiting the modular computational structure in a Gaussian process to develop the parallelisation scheme.

2.2.2 Multi-output Gaussian Processes

Multi-output regression problems have extensively arisen in modern engineering community. Gaussian processes are usually parameterised in terms of their covariance functions which makes it difficult to deal with multiple outputs, since ensuring that the covariance matrix is positive-definite in multi-output regressions is problematic. However, there are several ways to construct a multi-output Gaussian process. The simplest and most intuitive way is to assume two independent Gaussian processes. The first one can be written as

$$f_1(x) \sim \mathcal{GP}(0, k_1(x, x')) \quad (2.4)$$

$$D_1 = \{(x_{i,1}, y_1(x_{i,2} + \sigma_1^2 \mathbf{I})) | i = 1, \dots, N_1\} \quad (2.5)$$

$$y_1 \sim N(0, k_1 + \sigma_1^2 \mathbf{I}) \quad (2.6)$$

For the second Gaussian process,

$$f_2(x) \sim \mathcal{GP}(0, k_2(x, x')) \quad (2.7)$$

$$D_2 = \{(x_{i,2}, y_2(x_{i,2} + \sigma_2^2 \mathbf{I})) | i = 1, \dots, N_2\} \quad (2.8)$$

$$y_2 \sim N(0, k_2 + \sigma_2^2 \mathbf{I}) \quad (2.9)$$

One can then combine the two Gaussian process in the following form

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \sim N \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix} + \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 I \end{bmatrix} \right) \quad (2.10)$$

$$\mathbf{y} = \mathbf{0} + \mathbf{k}_{f,f} + \Sigma \quad (2.11)$$

Another way to formulate the covariance function for multi-output purposes is to assume that the covariance function is the result between of the multiplication of the covariances of different outputs. Vargas-Guzmán [36] used this idea, called coregionalization, to demonstrate its use in the more general case of non-orthogonality, which produces additional terms including cross-covariance in nested structures. Wilson [37] introduced an interesting regression framework called Gaussian process regression network (GPRN), that combines the Bayesian neural networks with Gaussian processes, allowing input dependent signal and noise correlations between multiple output variables, input dependent length-scales and amplitudes, and heavy-tailed predictive distributions. Wilson [37] also demonstrated substantially improved performance over eight popular multiple output (multi-task) Gaussian process models.

2.2.3 Multi-task Gaussian Processes

Multi-task learning (MTL) is a sub-discipline of machine learning where many learning tasks are executed simultaneously, with the benefit of exploiting the known commonalities between tasks. In the context of machine learning regression using Gaussian processes, the training model learns a covariance function on input-dependence features and a covariance matrix over tasks [38]. Multi-task learning can be used for learning a low-dimensional representation which is shared across a set of multiple related tasks. Argyriou [39] showed that this problem is equivalent to a convex optimisation problem. Peltonen et al. [40] investigated the problem of sub-task learning, which is a multi-task learning problem where the goal is to learn a good classifier for a task by exploiting supplementary data and with few training samples. In their work,

they showed that the problem can be solved by careful mixture modelling and their formulation can be used with variational approximation which makes it more suitable for high-dimensional data.

Multi-task learning is inherently a multi-objective problem because different tasks may conflict, necessitating a trade-off. Bonilla [41] describes two ways of achieving this using Gaussian process predictors. In the first method, the data from all tasks is combined into one dataset, making use of the task-specific features. In the second method, specific predictors are trained for each reference task, and then their predictions are combined using a gating network.

A multi-task Gaussian process model can be constructed as follows [38]. Let $\mathbf{x}$ be a set of n inputs $\mathbf{x}_1, \dots, \mathbf{x}_n$. Let $\mathbf{y}$ be the output vector of responses for M tasks, such as:

$$\mathbf{y} = (y_{11}, \dots, y_{n1}, \dots, y_{12}, \dots, y_{n2} \dots, y_{1M}, \dots, y_{nM})^\top \quad (2.12)$$

where y_{il} is the response for the l^{th} task on the i^{th} input $\mathbf{x}_i$. Given a set of observations $\mathbf{y}_o \in \mathbf{y}$, the goal for this model is to predict some of the unobserved response-values $\mathbf{y}_u$ at some input locations for certain tasks. Following Bonilla [38], the problem is approached by placing a Gaussian process prior over the latent functions f_l , directly inducing correlations between the tasks. Assuming that the Gaussian processes have zero mean

$$\langle f_l(\mathbf{x}) f_k(\mathbf{x}') \rangle = \mathbf{k}_{lk}^f k^x(\mathbf{x}, \mathbf{x}') \quad y_{il} \sim \mathcal{N}(f_l(\mathbf{x}_i), \sigma_l^2), \quad (2.13)$$

where $\mathbf{k}^f$ is a positive semi-definite matrix that models the similarities between tasks. k^x is a covariance function over inputs, and σ_l^2 is the noise variance for the l^{th} task.

In many engineering and scientific applications, there is the need to optimise a multi-task model. However, it's often the case that the computations for this type of optimisation becomes prohibitive. Moss [42] proposed a novel multi-task version of the acquisition functions known as entropy search that is scalable and efficient, allowing multi-task Bayesian optimisation to be deployed in problems with rich parameter and fidelity spaces. In the work by Sener [43], the problem of multi-task learning was

restated as a multi-objective optimisation problem, with the goal of finding a Pareto optimal solution. The authors introduced an upper bound for the multi-objective loss and demonstrated that it can be optimised efficiently under realistic assumptions. Often it is interesting to consider whether it is possible to transfer the knowledge gained from previous optimisation to new tasks in order to find optimal hyper-parameter settings more efficiently. An approach to this problem was made by Swersky [44] in the context of multi-task Bayesian optimisation.

2.2.4 Deep Gaussian Processes

Deep Gaussian processes were introduced by Damianou and Lawrence [45] and they consist of a deep belief network that is based on Gaussian process mappings. Thus, a single layered deep Gaussian process is equivalent to a standard Gaussian process. Deep Gaussian processes provide a flexible approach to probabilistic modelling of data using either supervised or unsupervised learning. In a deep Gaussian process architecture featuring several layers the generative process takes the form [45]:

$$\begin{aligned} y_{nd} &= f_d^Y(\mathbf{x}_n) + \epsilon_{nd}^Y, \quad d = 1, \dots, D, \quad \mathbf{x}_n \in \mathbb{R}^Q \\ x_{nq} &= f_q^X(\mathbf{z}_n) + \epsilon_{nq}^X, \quad q = 1, \dots, Q, \quad \mathbf{z}_n \in \mathbb{R}^{Q_z} \end{aligned} \quad (2.14)$$

and the intermediate node is involved in two Gaussian processes, f^Y and f^X , playing the role of an input and an output respectively:

$$f^Y \sim \mathcal{GP}(\mathbf{0}, k^Y(\mathbf{X}, \mathbf{X})) \quad (2.15)$$

$$f^X \sim \mathcal{GP}(\mathbf{0}, k^X(\mathbf{Z}, \mathbf{Z})) \quad (2.16)$$

Naturally, this structure may be extended vertically or horizontally, as deep networks do. For tractable inference approximations to the marginal likelihood of the model must be made. The original approach to approximate inference in these models used variational compression to allow for approximate variational marginalisation of the hidden variables leading to a lower bound on the marginal likelihood of the model

Damianou and Lawrence [45], 2013. In Hensman et al. [46] the authors extended this idea with a nested variational compression. The resulting lower bound on the likelihood can be easily parallelised or adapted for stochastic variational inference. Dai [47] developed a scalable deep non-parametric generative model by augmenting deep Gaussian processes with a recognition model. Inference is performed in a novel scalable variational framework where the variational posterior distributions are reparametrised through a multilayer perceptron. The key aspect of this reformulation is that it prevents the proliferation of variational parameters which otherwise grow linearly in proportion to the sample size. Deep Gaussian processes boost performance by modelling networks of GP nodes [48, 49] or by mapping inputs through multiple Gaussian process' layers [45, 50]. While more flexible and powerful than standard GPs, deep Gaussian processes result in degenerate models if the individual GP layers are not invertible, which limits their potential [51].

2.3 Robust Optimisation in the Bayesian Paradigm

Bayesian optimisation (BO) is very useful in optimisation problems involving unknown black-box functions that are expensive to evaluate. These problems arise in different scientific and engineering fields such as aerospace, robotics, drug discovery, machine learning and artificial intelligence [52]. The nature of these problems often requires that the optimisation is performed under a limited budget of time and computational power. In such problems, the goal generally is to optimise an expensive, black-box objective function f , such that $f : \mathcal{X} \rightarrow \mathbb{R}$. Formally, one can write:

$$\arg \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \quad (2.17)$$

where the domain is $\mathcal{X} \subseteq \mathbb{R}^d$ and d the dimensions.

Bayesian Optimisation consists of two blocks of functionality. The first is a Bayesian probabilistic model for modelling the objective function. Usually, this is a Gaussian process (refer to section 2.2). The second is an acquisition function, normally denoted

by α for deciding where to sample after an iteration. A common approach is to initially evaluate the domain, $\mathcal{X}$, using a space-filling experimental design, such as Latin Hypercube Square (LHS). This evaluation will be fed into the Bayesian statistical (or probabilistic) model, which provides a Bayesian posterior probability distribution that infers potential values for $f(\mathbf{x})$ at the point $\mathbf{x}$.

The so-called Bayesian update happens at each iteration, where the posterior distribution is updated at each new point observed. The acquisition function, in turn, computes based on the present posterior distribution over f , the value that would be generated by the query of the objective function at this new point $\mathbf{x}$. The flowchart 2.1 gives an overview of a typical optimisation procedure using Bayesian Optimisation and the algorithm 2.1 represents this procedure in an algorithmic way.

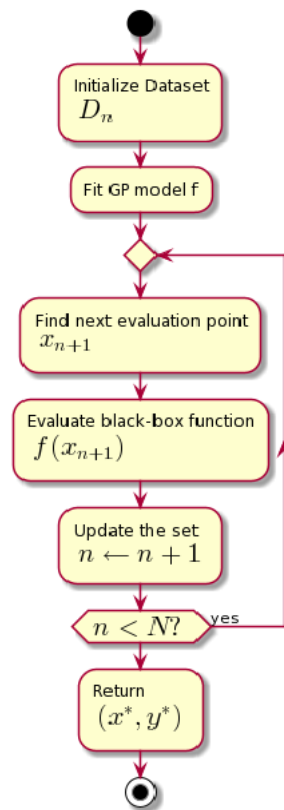


Figure 2.1: Bayesian optimisation represented as a UML flowchart

Algorithm 2.1: Bayesian Optimisation Algorithm

```

1 for  $n = 1, 2, \dots$  do
2   Select new  $x_{n+1}$  by optimizing acquisition function  $\alpha$ 
      
$$\mathbf{x}_{n+1} = \arg \max_{\mathbf{x}} \alpha(\mathbf{x}; \mathcal{D}_n)$$

3   Query objective function to obtain  $y_{n+1}$ 
4   Augment data  $\mathcal{D}_{n+1} = \{\mathcal{D}_n, (\mathbf{x}_{n+1}, y_{n+1})\}$ 
5   Update statistical model

```

2.3.1 Statistical Model - Gaussian Process

As previously said, the statistical model employed by the Bayesian optimiser is usually a Gaussian process. This model translates the information gained by the previous observations on the domain of the function by computing the posterior over the objective function conditioned over the data. Every time the function is evaluated (or queried), the posterior is re-computed or updated. Let $\mathcal{D}_n \leftarrow \{(\mathbf{x}_1, y_1) \dots (\mathbf{x}_n, y_n)\}$ be the initial set of noisy observation or data samples of the objective function f . The vector notation here is used for indicating that a new point $\mathbf{x}$ is high-dimensional. The posterior of the function f on the observed data $\mathcal{D}_n$ is also Gaussian, that is,

$$f(\mathbf{x}) \mid \mathcal{D}_n \sim \mathcal{N}(\mu_n(\mathbf{x}), \sigma_n^2(\mathbf{x})) \quad (2.18)$$

where the mean $\mu_n(\mathbf{x})$ and covariance $\sigma_n^2(\mathbf{x})$ are given as follows:

$$\mu_n(\mathbf{x}) = \mu(\mathbf{x}) + k^T (k(\mathbf{x}_i, \mathbf{x}_j) + \eta^2 I_n)^{-1} \mathbf{Y} \quad (2.19)$$

$$\sigma_n^2(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}_i) - k^T (k(\mathbf{x}_i, \mathbf{x}_j) + \eta^2 I_n)^{-1} k \quad (2.20)$$

The vector of observations is $\mathbf{y}$. The matrix k is $k_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$ $i, j \in \{1, \dots, n\}$. There are many possible choices for the covariance kernel, each kernel is designed to encode a property such as smoothness [29]. Common choices are the square exponential kernel or Matérn kernel [29].

2.3.2 Acquisition Function

As mentioned, the acquisition function is the component of a Bayesian Optimiser responsible for utilising the information from previously observed data to navigate through the domain. It is often denoted by α . Acquisition functions are used to measure the chances of finding a maximiser of the objective function, f , in the entire domain, $\mathcal{X}$. Thus, at each iteration of the Bayesian optimiser the function f is queried at that maximiser candidate. Common choices of acquisition function are the Expected Improvement (EI) [53] and the Upper Confidence Bound (UCB) [54], these are defined respectively as follows. The expected improvement may be written as a closed-form expression:

$$\begin{aligned} \text{EI}(\mathbf{x}) &= \sigma(\mathbf{x})(z(\mathbf{x})\Phi(z(\mathbf{x})) + \phi(z(\mathbf{x}))) \\ \text{with } z(\mathbf{x}) &\equiv \frac{t - \mu(\mathbf{x})}{\sigma(\mathbf{x})}, \end{aligned} \tag{2.21}$$

where $\mu(\mathbf{x})$ is the predictive mean at $\mathbf{x}$, $\sigma^2(\mathbf{x})$ is the predictive variance at $\mathbf{x}$, $\Phi(\cdot)$ is the standard normal cumulative distribution function (CDF), and $\phi(\cdot)$ is the standard normal probability distribution function (PDF). This function is differentiable and fast to compute, and can therefore be maximised with a standard gradient-based optimiser. The upper confidence Bound is defined as follows:

$$a_n(\mathbf{x}) \leftarrow \mu_n(\mathbf{x}) + \beta^{1/2} \sigma_n(\mathbf{x}) \tag{2.22}$$

In UCB, the hyper-parameter that controls the trade-off between exploitation and exploration of the domain, $\mathcal{X}$ is denoted by β .

The acquisition function computes samples from the domain $\mathcal{X}$ of the objective function, f thereby providing information within an uncertainty about how close to optimal the optimiser is. This maximisation is often performed using methods such as L-BFGS [55].

2.3.3 Extensions to Bayesian Optimisation

Previous content presented the standard Bayesian optimisation strategy. However, for a number of important applications the standard approach is not sufficient. An important extension is the capability of handling multiple objectives and constraint functions, when these functions are assumed to be smooth, non-linear and expensive to evaluate, as it is normally the case in complex industrial design optimisations. Many multi-objective problems are handled with an acquisition function named expected hypervolume improvement (EHI).

Yang et al. (2019) [56] proposed expected hypervolume improvement gradient (EHVIG) as a new criterion. EHVIG enables gradient ascent methods to be used in multi-objective Bayesian global optimisation that reduces the computational burden. Feliot et al. (2017) [57] addressed the problem of multi-objective Bayesian optimisation by using an extended domination rule to handle both the objectives and the constraints in a unified way. Many problems could also benefit of preference-order constraints, where the analyst is able to select which objective is preferential among a set of choices. Abdolshah et al. (2019) [58] addressed this problem and presented a multi-objective Bayesian optimisation algorithm that allows the user to express preference-order constraints on the objectives of the type “objective A is more important than objective B”.

The applications of Bayesian optimisation are vast and the amount of flight physics data captured by the aeronautical industry is immense. However, while BO is capable of performing very well in low dimensional problems (less than 20 parameters), scaling it to higher dimensional spaces is not a trivial task. A very comprehensive review of possible directions in this field is provided by Malu et al.(2021) [59]. Other excellent introductions and surveys on Bayesian Optimisation in general are Frazier (2018) [52], Shahriari et al. (2016) [60], Brochu et al. (2009) [61] and Frazier and Wang (2016) [62].

2.4 Uncertainty Quantification and Management

Uncertainty Quantification (UQ) is the rational process of determining quantitatively the uncertainties associated with predictions of a given model or real-world system. UQ is constructed from concepts and techniques originating from mathematics (mainly functional analysis and measure theory [63]), engineering, computational mechanics, probability theory and statistics. In the context of UQ, the uncertain parameters that cannot be assigned with deterministic values are called interesting quantities (IQ).

A typical UQ problem involves one or many mathematical models of a given process, which are subject to uncertainty over the parameters. These uncertainties are dealt with probabilistically via random variables (see section A.1.3). UQ can be defined as:

Uncertainty quantification (UQ) studies all sources of error and uncertainty, including the following: systematic and stochastic measurement error; ignorance; limitations of theoretical models; limitations of numerical representations of those models; limitations of the accuracy and reliability of computations, approximations, and algorithms; and human error [64].

The execution of UQ for a given problem often encompasses the use of different mathematical techniques and tasks such as parameter screening, construction and validation of surrogate models, quantitative assessment of model output uncertainties and sensitivities, model calibration and assessment of their posterior uncertainties. Some of these tasks are dependent on expert judgement and on the circumstances. Therefore, it is common to denote this field by the more encompassing name of uncertainty quantification and management (UQ & M).

The common sources of uncertainty associated with simulation models are uncertainties in the model parameters (design variables, initial and boundary conditions), uncertainties in the experimental data used to train the models or uncertainties in the mathematical form of the models. In order to take into account systematically all these uncertainties. Usually, a UQ & M study consists of the following steps: a) Uncertain sources identification, characterising the probability distributions or its ranges, b)

Propagate these uncertainties through the model, and c) Analyse the impact of these uncertain sources on the variability of the IQs. To illustrate the type of problems that UQ deals with. Assuming a function $y = f(\mathbf{x})$, the following classification is proposed:

- **The forward propagation or push-forward problem.** Suppose that the uncertainty about the inputs of f can be summarised in a probability distribution μ on X . Given this, determine the induced probability distribution $f \star \mu$ on the output space Y .
- **The reliability of certification problem:** Suppose that some set $Y_{fail} \subseteq Y$ is identified as a “failure set”. Given appropriate information about the input x and forward process f determine the failure probability.
- **The prediction problem:** Duality to the reliability problem, given a maximum acceptable probability of error $\epsilon > 0$, find a set $Y_\epsilon \subseteq Y$, such that

$$P[f(\mathbf{x}) \in Y_\epsilon] \geq 1 - \epsilon \quad (2.23)$$

- **The inverse problem:** An inverse problem, such as state estimation (often for a quantity that is changing in time) or parameter identification (usually for a quantity that is not changing, or is non-physical model parameter). Given some observations of the output, Y , which may be corrupted or unreliable in some way, attempt to determine the corresponding inputs X such that $f(X) = Y$.
- **The model reduction or model calibration problem:** Construct another function f_h (perhaps a numerical model with certain numerical parameters to be calibrated, or one involving far fewer input or output variables) such that $f_h \approx f$ in an appropriate sense. Quantifying the accuracy of the approximation may itself be a certification or prediction problem
- **The coupled or system UQ problem:** Sometimes a UQ problem consists of several of these problems coupled together: For example, one might have to solve an inverse problem to produce or improve some model parameters, and then

use those parameters to propagate some other uncertainties forwards, and hence produce a prediction that can be used for decision support in some certification problem.

UQ has been used with success in several aeronautical disciplines such as in multi-disciplinary design optimisation [65–70], expert systems [71, 72], in computational fluid dynamics [72–75], in multifidelity [76–79], in multi-level [80], in multiphysics simulations [81], aeroelasticity [82–84], hypersonics [85–87], in structural analysis and design [88–93], in model-based systems engineering [94, 95], in astrodynamics [96], in conceptual and decision-based design [97–102], in optimal control [103], in manufacturing [104]. Roy [105] presents a framework for verification, validation in scientific computing UQ has also been used in parametric modelling [106], in model-calibration [107], and System identification [108]. Additionally, UQ has been used in technology portfolio selection with satisfying results [109]. Numerical methods such as decomposition was used by Amaral [110] and Owhadi [111]. UQ software solutions suitable for industrial usage include Sequoia [112] and OpenTURNS [113].

2.4.1 Dimension Reduction

Principal component analysis (PCA) is a mainstay of modern data analysis [114]. PCA is used to tackle the problem of multi-collinearity – when inputs are correlated with other inputs. Principal component analysis (PCA), also known as proper orthogonal decomposition (POD), is currently one of the prevailing Reduced Order Modelling (ROM) techniques for aerodynamic data analysis. For example, the results found by Omiya and Rinoie [115] indicated that PCA would be a useful method to comprehend the relationships among design parameters for the aircraft conceptual designers. Interesting applications of PCA include crashworthiness optimisation [116], aerodynamic data reconstruction [117], multidisciplinary design optimisation of air propulsion components [118] and evaluation of flight parameters during approach and landing phases [119].

An important variation of the PCA method is the probabilistic principal component

analysis (PPCA) [117, 120, 121]. Tipping [120] demonstrated the advantages conveyed by the probabilistic approach to PCA. Specifically, he showed how the principal axes of a set of observed data vectors may be determined through maximum-likelihood estimation of parameters in a latent variable model considering the properties of the associated likelihood function.

2.4.2 Monte Carlo Methods

Monte Carlo methods (MCM) are a class of computational algorithms centred on the idea of repeated random sampling in order to obtain numerical results. The fundamental concept is the use of randomness to solve problems that in principle are formulated as deterministic. MCM are generally used in three types of problems: sampling from a probability distribution, numerical integration and optimisation problems.

In the aeronautical industry, MCM are used extensively in reliability estimation and, more recently, in uncertainty quantification [122]. The MCM has many advantages over conventional methods for the estimation of the uncertainties in the outputs of a complex system. For example, Monte Carlo simulation has the ability to take account of partial correlated measurement input uncertainties. For correlated input measurements, the probability distribution of the result could be biased or skewed, which are properties that cannot be revealed using conventional methods. The Monte Carlo method is relatively simple to implement. However, the increased demands of uncertainty based design, analysis, and optimisation in aerospace motivate the application of advanced Monte Carlo methods with wide adaptability and high accuracy.

A typical aerospace problem is the uncertainty quantification of defects in composite structures. Butler [123] approached this problem using a multilevel Monte Carlo methodology to quantify the structural performance of a composite wing skin panel with three types of random manufacturing defects (ply-angle perturbations and two classes of localised fibre waviness), achieving computational gains. Sobol [124] has developed global sensitivity indices (see item 2.4.4) for Monte Carlo (or quasi-Monte Carlo) estimates for nonlinear mathematical models.

Hu [125] reviews the use of Monte Carlo methods in uncertainty-based design optimization for aerospace vehicles. Their paper presents a comprehensive review of typical improved Monte Carlo methods and summarizes their characteristics to aid the uncertainty based multidisciplinary design optimization. A comprehensive reference of Monte Carlo and quasi-Monte Carlo sampling is the book by Lemiux [126]. Corbetta [127] proposed a fast Monte Carlo method to solve differential equations utilized in model-based prognostics prognostic and used it in cracked structures under fatigue loading. Peherstorfer [128] applied the multi-fidelity Monte Carlo method to a coupled aero-structural analysis of a wing and a flutter problem with a high aspect ratio wing.

2.4.3 Copula Theory

Copulae are tools for modelling dependence of several random variables. The term copula was first used in the work of Sklar [129] and is derived from the latin word *copulare*, to connect or to join. The main purpose of copulae is to describe the interrelation of several random variables. Copulae enable us to model marginal distributions and the dependence structure separately. Formally, a copula is defined as:

Definition 1 (Copula). *A d -dimensional copula is a multivariate distribution function $\mathcal{C}$ with standard uniform marginal distributions.*

Theorem 1 makes copulae excellent tools for modelling flexibility since any multivariate distribution input can serve as a copula.

Theorem 1 (Sklar, 1959). *Let H be a joint distribution function with margins $F_1, \dots, F_d$. Then there exists a copula $C : [0, 1]^d \rightarrow [0, 1]$ such that*

$$H(x_1, \dots, x_d) = C(F_1(x_1), \dots, F_d(x_d)) \quad (2.24)$$

Copulae applications include parameter estimation, model selection and model evaluation. In the aeronautical and aerospace literature, copulae have been used in

the probabilistic fatigue evaluation of wind turbines [130]. Copula theory can also be used in conjunction with the inversion method to mitigate both of these concerns while improving the accuracy of probabilistic assessments and enabling decision makers to better represent the uncertainty associated with the problem at hand [131].

The high degree of complexity and uncertainty associated with aerospace engineering applications have driven systems analysts toward the use of probabilistic and statistical analysis tools in order to understand and design for that uncertainty [132]. Common themes in aerospace applications of probabilistic theory include the assumed independence of random variables and the arbitrary or unjustified use of certain probability distributions to represent random variables [132]. These incorrect assumptions are often unintentionally used in systems analysis and can lead to inaccuracies in uncertainty quantification, poor decision making, and flawed designs. In several applications, empirical evidence showed that multi-normal distributions are not entirely satisfactory to represent the parameters in a problem [133]. For example, empirical margins distributions can be skewed and heavy-tailed, which are not possible to represent as a multi-normal distribution.

2.4.4 Sensitivity Analysis

Sensitivity analysis is the systematic study of how model inputs (and/or parameters, initial and boundary conditions) affect key model outputs. A sensitivity study is usually complicated by a number of factors, such as the dimensionality of the input space and the complexity of computational model. Challenges in sensitivity analysis include dealing with these factors while addressing the needs of the application. It is quite common that uncertainty quantification and sensitivity analyses run in tandem. A very good reference for beginners in sensitivity analysis is *Global Sensitivity Analysis: The Primer* [134].

There are a number of ways to compute sensitivity indexes. These indices are used for estimating the influence of individual variables or groups of variables on the model output. The canonical way is to compute global sensitivity indices using Monte Carlo

methods, some of which can be used for complex mathematical models in a efficient way. However, some mathematical models demand faster approaches. In this case, usually Sobol Indices are used [135]. The Sobol indices method [135, 136] assumes a model given by a function $f(x_1, \dots, x_n)$ defined in the unit hypercube H^n with Lebesgue measure $dx = dx_1, \dots, dx_n$. Sobol [136] showed that if the function is square integrable, then global sensitivity indices provide adequate estimates for the influence of individual factors x_i or groups of such factors. Kucherenko [137] proposes a derivative-based global sensitivity measure and shows its link with Sobol sensitivity indices.

The variance-based method of global sensitivity indices based on Sobol's sensitivity indices became very popular among practitioners due to its easiness of interpretation. However, as Kucherenko [138] points out, for complex practical problems computation of Sobol indices generally requires a large number of evaluations to achieve convergence. Therefore, Kucherenko [139] proposed four different direct formulas for computing Sobol's main effect sensitivity indices, that are evaluated using Monte Carlo (MC) and Quasi Monte Carlo (QMC) techniques, and compared it with an another approach called the "double loop reordering" (DLR) approach. The latter showed superior performance [138]. There are many other methods comparable to Sobol indexes [137, 139–141].

Saltelli [142] deals with computations of sensitivity indices in sensitivity analysis. Given a mathematical or computational model $y = f(x_1, x_2, \dots, x_k)$, where the input factors x_i 's are uncorrelated with one another, one can see y as the realization of a stochastic process obtained by sampling each of the x_i from its marginal distribution. In particular, Bergonovo [140] introduced a global sensitivity indicator which captures the influence of input uncertainty on the entire output distribution with moment independence. This is useful because Bergonovo indexes can also be defined in the presence of correlations among the parameters. A method for the estimation of main effects in global sensitivity analysis that applies to non-linear and non-additive models was developed by Tarantola [141].

Sensitivity analysis has been used in aeronautical and aerospace applications with

considerable success. The majority of the works employs several probabilistic sampling procedures, allowing the sensitivity analysis/uncertainty quantification calculations to take into account both aleatory and epistemic uncertainties. For example, Strand [143] uses sensitivity analysis for direct simulation Monte Carlo (DSMC) simulations of high-temperature air chemistry. A similar approach is employed by Burt [144], in their work the authors used a novel uncertainty quantification technique combining sampling sensitivity analysis with Latin hypercube sampling with surrogate models in order to investigate the shock and shock wave/boundary-layer interaction phenomena of a representative hypersonic double-cone flow. A validation assessment of a hypersonic double-cone flow simulation using uncertainty quantification, sensitivity analysis, and validation metric was performed by Kieweg [145], showing good agreement between the continuum CFD simulations and experiments. Decarlo [146] proposed a method for computing sensitivity indices for coupled time-depend analysis and observed that model errors and parameter correlations control the sensitivity estimates until coupling effects become dominant. Tryoen [147] used a stochastic spectral framework to perform a sensitivity analysis when investigating the impact of uncertain input data on stagnation-point measurements, in order to rebuild free-stream conditions for the trajectory of a re-entry vehicle.

2.4.5 Uncertainty Propagation

Polynomial chaos expansion has been used extensively in the aeronautic literature for uncertainty propagation. In computational fluid mechanics [148–151], in structural analysis [152], in the uncertainty quantification of airfoil icing [153], nonlinear estimation of hypersonic flight [154], in the on linear propagation of orbit uncertainty [155], optimal control [156], trajectory optimization [157], robust optimization of multi-physics problems [158], multi-fidelity aerodynamic design optimization through surrogate models [159, 160], reliability-based design [161] and in Gaussian processes to investigate uncertainty associated with off-design conditions [162].

The polynomial chaos expansion of a function $\mathbf{y} = f(\mathbf{u}(\mathbf{x}))$, where $\mathbf{x}$ is the design

variables vector, $\mathbf{u}$ is the uncertainty vector represented by random variables and $\mathbf{y}$ is the output/response/quantity of interest vector. Mathematically, the n th-order polynomial chaos expansion of this function, $\mathbf{y} = f(\mathbf{u}(\mathbf{x}))$, is

$$\begin{aligned}
 \mathbf{Y}(\mathbf{x}, \mathbf{u}) &= a_0 H_0 \\
 &+ \sum_{i_1=1}^{\infty} a_1 H_1(u_{i_1}(x)) \\
 &+ \sum_{i_1=1}^{\infty} \sum_{i_2=1}^{i_1} a_{i_1 i_2} H_2(u_{i_1}(x), (u_{i_2}(x))) \\
 &+ \sum_{i_1=1}^{\infty} \sum_{i_2=1}^{i_1} \sum_{i_3=1}^{i_2} a_{i_1 i_2 i_3} H_3(u_{i_1}(x), u_{i_2}(x) u_{i_3}(x)) \\
 &+ \dots
 \end{aligned} \tag{2.25}$$

Dodson and Parks [148] in conjunction with computational fluid dynamics to quantify the effects of uncertainty in the computational aerodynamic design process of problem of maximising the lift-to-drag ratio of a two-dimensional airfoil while minimising its sensitivity to uncertainty in the leading-edge thickness. When comparing with a non-robust airfoil, their results also indicate that the inherent geometric uncertainty could degrade the on-design as well as the off-design performance of the airfoil, and concluded that the global optimum for some design problems is unreachable without accounting for uncertainty. Sparse polynomial chaos expansions have been used by Abraham et al. [163] to quantify the uncertainties in the industrial turbo-machinery design in the presence of many geometrical and operational uncertainties. They showed that the sparse polynomial chaos method achieves faster convergence in comparison with the collocation method.

Uncertainty quantification is classified in intrusive and non-intrusive [63]. An example of non-intrusive uncertainty quantification using polynomial chaos is the work done by Hosder and Walters [164] where the authors examine uncertainty quantification in computational fluid dynamics (CFD) with non-intrusive polynomial chaos (NIPC) methods which require no modification to the existing deterministic models. Sparse

polynomial chaos expansions has also been used in the investigation of post-maneuvre collision probability estimation [165]. Ghisu et al. [166] developed a nonintrusive polynomial chaos formulation in which the standard orthogonal polynomials from the Askey scheme are replaced by a set of orthonormal polynomials calculated relative to the specific probability density function. Oladyshkin [167] developed a technique that generalises chaos expansion techniques towards arbitrary distributions with arbitrary probability measures, which can be either discrete, continuous, or discretised continuous and can be specified either analytically (as probability density/cumulative distribution functions).

2.5 Digital Transformation in the Aerospace Industry

The recent progress in data science and engineering has been possible due to the unprecedented advances in high performance computation, improvements in sensing and data storage technology and the development of novel algorithms. This has led to the creation of several open-source ecosystems and the availability of open-source code and benchmark problems. Under these circumstances, many projects are currently being undertaken to foster a digital transformation of the aerospace industry.

The rapid transformation and digitalisation of the aerospace industry is driven by the possibilities of capitalising on machine learning and big data technology. Aerospace engineering is by nature a data-intensive field that relies on powerful computations to solve a myriad of problems, in particular, optimisation problems. In addition to that, all stages in aerospace manufacturing can benefit of data-intensive approaches for the purpose of uncertainty quantification which fits well traditional quality engineering paradigms such as six sigma [168] and design for six sigma (DfSS) [169].

In the next sections, some concepts and technologies that are essential to digital engineering such as digital-twins and model-based machine learning are surveyed in the context of their use in data-driven engineering. Incidentally, these ideas are the motivators for the engineering Bayesian framework presented in section A.3.

2.5.1 Digital-Twins

The aim of a digital-twin technology or product is to create a virtual copy of the product of interest from the very first moment that the product idea materialises. The virtual copy reflects the real one by using all of the available data [170]. This idea originated in the context of product life cycle management (PLM) and was first introduced by Grieves in 2002 [171]. However, the idea has been used since the 1960s in the Apollo 13 project and others.

The digital-twin concept is a digital technology that promises to improve the understanding, learning and reasoning from the real-time product data. In the recent years, thanks to the rapid change towards the digital transformation [172], the feasibility of a digital thread is certain. Digital thread is a data-driven architecture that links together information generated across the product life cycle and is envisioned to be the primary or authoritative data and communication platform for a company's products at any instance of time. Thus, the digital thread is the means by which digital-twinning can be achieved.

Modern aerospace engineering design depends on computational models, however, the insights gathered from such models are only reliable as far as the model is an accurate representation of the underlying physical system. Two identical systems that share the same design parameters may have different operational histories that will cause differences, for instance, in material properties [173]. The use of a single static model to account for such differences may consist in a limitation when the history of a particular system is the main interest. The digital-twin paradigm aims to overcome this limitation by providing an authoritative digital model tailored to each unique system [174].

The realisation of a digital-twin is dependent on high-fidelity physical models that control the life of a vehicle. The digital-twin concept may integrate these high-fidelity simulations with on-board health management and monitoring system. Based on this, it is possible to track in much more detail the history of the vehicle including its maintenance services and configuration changes [175]. The data can be used for

purposes of fleet monitoring, enabling significant gains in safety and reliability. This knowledge can then be used in future design tasks for a complete understanding of the physical processes related to degradation of the materials, structures, and systems that control the vehicle life cycle. This will allow to move away from “factors-of-safety” heuristics in some circumstances, offering benefits in performance and cost reduction. Moreover, this technology may be the basis for models and methodologies forming the backbone of future certification and life cycle sustainability practices. While a fully-matured digital-twin is yet to be seen, there are already engineering applications that leverage reduced-order models such as Kapteyn et al. [176], collaborative design [177], structural health management [175], digital thread with an authoritative truth source (ATS) [174], aircraft failure detection and correction [178], rocket design [179] and fleet management [180].

2.5.2 Model-Based Systems Engineering

System Engineering is, according to Eisner [181], an iterative process of top-down synthesis, development, and operation of a real-world system that satisfies, in a near optimal manner, the full range of requirements for the system. By the same token, model-based systems engineering (MBSE) is defined by Blanchard [182] as the formalised application of graphical modelling to support system engineering activities beginning in the conceptual design and covering the entire life cycle. MBSE must be contrasted with the traditional system engineering approach where system engineering tasks as requirements capturing and analysis, design, validation and verification are conducted in a textual-based way. The use of MBSE supports and enhances the system engineering tasks in a project with the following benefits [182]:

- Improved communication among development stakeholders,
- Increased ability to manage complexity,
- Improved analysis of the impact of changes and
- Enhanced knowledge capture and reuse of information

Douglass [183] incorporated the use of the agile philosophy developed in the context of software engineering to system engineering to create the agile model-based systems engineering philosophy. The primary goal of such philosophy is to develop requirements that can drive the integrated product development teams (IPDTs) while enabling follow-on systems development by the use of verifiable models.

Krupa [184] adapted AMBSE to aircraft design, resulting in an incremental and spiral life cycle (figure 2.2), that is well suited for a digital transformation or digital engineering development strategy along with the digital-twin concept (see 2.5.1). The project initiates by the capturing of stakeholder requirements and by decomposing the expected functionality through a systems engineering functional analysis. Requirements analysis is performed after the functional analysis process. Then, it is possible to define the logical architecture, which is done by establishing a hierarchy within the system and by allocating requirements to its corresponding functions. Based on the logical architecture, initial architectures are proposed. These are analysed and new candidate architectures are synthesised in order to fit the desired functionality and its requirements. The candidate architectures are evaluated with figures of merit (FoMs) or against technical performance measures (TPMs), and the best one is selected. The proposed architecture is validated accordingly, and the overall system is verified. The cycle proceeds in a spiral way, meaning that each subsystem and its components, if necessary, are developed in the same incremental way. The process is iterative, and at each cycle requirements, functionality and system elements are continually updated to reflect the newly available information.

2.5.3 Set-Based Concurrent Engineering

Set-based concurrent engineering (SBCE) is described by Sobek [185] as an engineering design philosophy where engineers are “reasoning, developing, and communicating about sets of solutions in parallel independently”. This definition, albeit informal, links concurrent engineering (CE) to set-based design (SBD), which some authors distinguish [186].

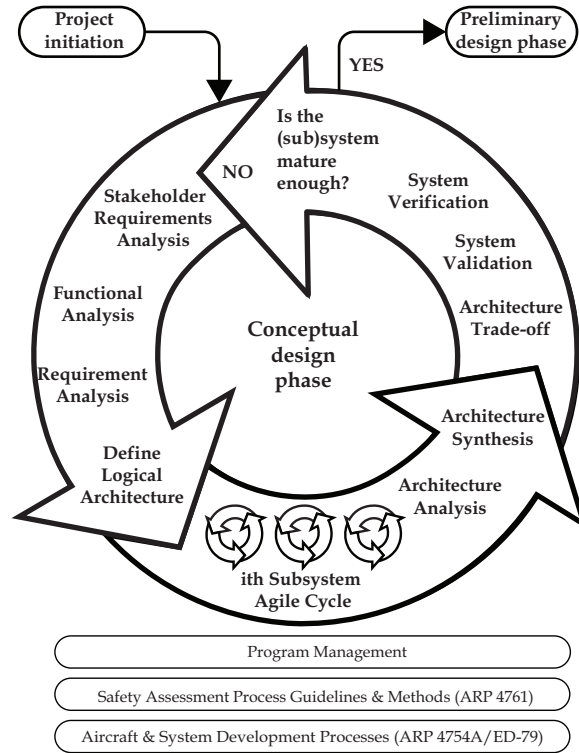


Figure 2.2: Agile System Engineering process for aircraft conceptual design

Concurrent Engineering (CE) is a systematic approach to the integrated, concurrent design of products and their related processes, including manufacturing and support. This approach is intended to cause the developers from the very outset to consider all elements of the product life cycle, from conception to disposal, including quality, cost, schedule, and user requirements [187]. Its essence is the simultaneous (or concurrent) execution of design processes with the design of downstream processes. The intent of concurrent engineering has been to increase the influence of downstream groups on upstream design decisions and to improve the quality of critical early design decisions.

Set-based concurrent engineering (SBCE) originated from the Toyota Product Development System (TPDS). SBCE takes advantage of parallel evaluation of multiple configurations in the early stages of design by using sets of possible solutions [188]. The SBCE process consists in the mapping of the design space and feasible regions, concurrently designing different alternative systems or candidate architectures. The concept of sets allows these candidate architectures or systems to be developed in a fully functional way across all discipline, identifying possible overlaps. Then as detail level design considerations increase the sets are narrowed down, with the consensus of

all IPDTs.

In set-based concurrent engineering, it is straightforward to generate configurations for systems that can be described by discrete design parameters such as subsystems or components. The initial step is to tabulate the identified design system parameters along with their possible options. These parameters are recombined for each new alternative configuration and evaluated according to a predefined set of performance metrics. The configurations that do not satisfy the design requirements are discarded and the remaining ones are stored for future use. This process continues until a desirable configuration emerges.

The conventional engineering design method is iterative and traditionally follows the classical V-model [182]. In the scope of set-based design, the conventional design approach may be referred collectively as a ‘point-based’ method’, where a small number of configurations and alternatives are evaluated in the conceptual design phase. After the conceptual design phase, the subsystems are developed in the preliminary design. The design is frozen and detail design work continues. Often this design process leads to the paradoxical situation of needing to establish requirements early. However, these requirements are very likely to change at each cycle of iteration. The adoption of set-based concepts in a engineering design or analysis in an aeronautical project can fundamentally change the manner in which engineers interact and the way that novel aircraft products are designed. However, the flexibility brought by SBCE adds uncertainty and organisation pressure in the process, since the consideration of multiple options may yield a delay in the design pipeline unless the narrowing is done aggressively [189].

2.5.4 Model-Based Machine Learning

In the last decade, machine learning technology has had a tremendous progress and thousands algorithms have been created. However, this rapid progress creates issues in knowledge management, since it is very difficult to keep in pace with the state-of-art technology. The practitioner’s choices are often limited by his familiarity

with a given technology or algorithm. Model-based machine learning (MBML) is a potential solution to this issue, that brings the promise of allowing a fast transition to novel algorithms, by changing the focus from the algorithm itself to the assumptions underlying or chosen for a given problem. The goal of MBML is often stated as: “to provide a single development framework which supports the creation of a wide range of bespoke models” [190]. MBML can be accomplished through the convergence of three main ideas: the Bayesian paradigm (see appendix A.3), the use of factor graphs (see Figure 2.3), and the application of rapid approximate inference algorithms. The core idea of MBML is that all the assumptions about the problem domain are made explicit in the form of a model and that by decoupling the model from the inference it is possible to apply the same inference method to a wide variety of models. Bishop [190] cites examples of software frameworks that implement the model-based machine learning philosophy such as Church [191] and Stan [192].

There are many potential benefits in the adoption of a model-based philosophy to machine learning. Since MBML separates the model from the inference process, it allows the seamless development of several models using the same or different inference algorithms to learn the model parameters. Thus, it is possible to quickly compare several alternative models and select the model that better fits the observed data. The Bayesian paradigm has been detailed in chapter 2, the next sections detail the other two cornerstones of MBML: Factor graphs and probabilistic programming.

A probabilistic graphical model (PGM) or structured probabilistic model or simple graphical model is a probabilistic model for which a graph expresses the conditional dependence structure between random variables. A particular type of PGM is the factor graph which consists of circular nodes representing random variables, square nodes for the conditional probability distributions (factors), and vertices for conditional dependencies between nodes (figure 2.3).

The purpose of a factor graph is to model the joint distribution of a set of random variables, acting as a general framework for probabilistic modelling in MBML. In factor graphs, the latent parameters are taken as random variables. Their probability

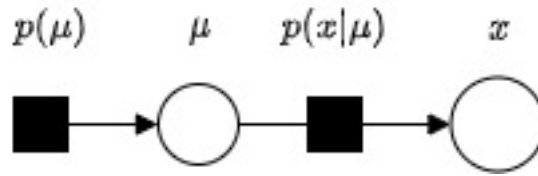


Figure 2.3: A factor graph representation

distributions are learned using Bayesian inference algorithms represented in the graph. In this way, the process of inference or learning is simply the product of factors over a subset of variables in the graph. This allows for easy implementation of local message-passing algorithms. The joint probability $P(\mu, \mathbf{x})$ over the whole model in figure 2.3 is factorised as:

$$P(\mu, \mathbf{x}) = P(\mu) * P(\mathbf{x}|\mu) \quad (2.26)$$

Where μ is the model parameter and $\mathbf{x}$ are the set of observed variables.

2.5.5 Automatic Machine Learning

The necessity of readily deployable machine learning models has given rise to a discipline called automated machine learning (AutoML) [193]. In a typical use case of a machine learning application or system, not equipped with any sort of AutoML, the engineer or practitioner must perform a series of tasks to set up, configure, check and test the model. However, each of these tasks may require close human intervention. For example, the dataset used for training the models may be not structured in a way that is ready to be used by a machine learning algorithm. In order to make the data amenable for machine learning, an experienced user may have to perform the time-consuming and difficult tasks of data pre-processing, feature engineering, feature extraction, and feature selection methods. After these steps, the engineer must then perform algorithm selection and hyper-parameter optimisation to maximise the predictive performance of their model. Each of these steps may be challenging, resulting in significant hurdles to the use of machine learning.

According to Hutter [193], the goal of AutoML is to automate all aspects of the entire machine learning and analysis pipeline. The entire machine learning pipeline

for most applications includes automating data collection and experiment design; automating data cleanup and missing data imputation; automating feature selection and transformation; automating model discovery, criticism, and explanation; automating the allocation of computational resources; automating hyper-parameter optimization; automating inference; and automating model monitoring and anomaly detection. In particular, the hyper-parameters of a machine learning system are a rich field for AutoML since there are information-theoretic concepts that make the optimisation of hyper-parameters amenable to efficient optimisation. Thus, by leveraging these concepts AutoML is capable of automatically set these hyper-parameters to optimise performance. Automated hyper-parameter optimisation (AHPO) has several important uses. Hyper-parameter optimization (HPO) can be used to adapt general-purpose pipelines to specific application domains [193]. The problem of HPO was identified early in machine learning history, since different hyper-parameter configurations lead to different performance in different datasets. HPO can be used to adapt general-purpose pipelines to specific application domains [193].

A more detailed explanation of AutoML is beyond the scope of this section. However, it is worth mentioning that many advancements related to the realisation of an effective AutoML system requires the use of surrogate models (Gaussian Processes and Bayesian networks) and Bayesian optimisation methods. Therefore, one can say that the progress of AutoML will directly impact the simulation and optimisation of novel aeronautical systems.

2.5.6 Probabilistic Programming

Probabilistic programming (PP) is a programming paradigm which aims to create systems that are capable of making decisions in face of uncertainty. In this paradigm the inference for probabilistic models are performed automatically. As such, it is proposed to better integrate probabilistic modelling with programming. PP is dedicated to the development of programming languages that allow the denotation of inference problems and their solution [194], while automating the tedious and troublesome derivation and

computation of gradients for optimisation.

The automation of inference achieved with probabilistic programming can be broadly applied to the entire tool-chain of Bayesian or probabilistic machine learning, allowing a rapid exploration of deep learning and big data regression, offering a great potential for the realisation of digital engineering of aircraft [194]. There are currently a very large number of PP languages such as: Turing [195], Pyro [196], Birch [197], and STAN [198] to mention a few.

2.6 Summary of Research Choices

This chapter presented an overview of the state-of-the-art of uncertainty quantification and robust optimisation methods. In chapter 1, section 1.2, the robust optimisation challenge provided by Airbus UK was presented. Although, many advances were obtained in the recent years in the fields of data science and machine learning, the methods still need to be assimilated by aeronautical researchers and engineers and aligned with current industrial practices. In particular, a better general understanding of how to build probabilistic models is beneficial and potentially has many applications in aeronautics. This, in turn, allows one to develop industrial probabilistic models that use surrogates models to speed-up time-consuming computations. Hence, a myriad of difficult problems can be approached, extending the current engineering capability. In addition, attention needs to be devoted to the important field of probabilistic regression. As the digital transformation trend continues, scalability issues of data and methods will arise. Hence, it is necessary to extend the current knowledge of Gaussian processes in the aeronautical realm [199] by exploring different formulations of Gaussian process in regards to its performance and applicability. Gaussian processes are particularly useful for aeronautical applications as opposed to alternative machine learning formulations, because continuous systems as physical models tend to possess a Gaussian distribution, as data becomes increasingly available. This known behaviour of natural phenomena can be used as an advantage and is mathematically guaranteed by the central limit theorem (see appendix A.1, section A.1.4, for a formal definition).

The important problem of scalability of Gaussian processes must be investigated. In particular, variational inference (VI) approaches that are likely to provide a cheaper and scalable alternative to time-consuming and resource expensively Monte Carlo approximations. Naturally, considerations of this sort lead to investigations in applied high-performance computing (HPC) and Big Data. However, this is outside of the immediate scope of this thesis. A promising alternative to the standard Gaussian process in relation to its scalability is the sparse formulation for Gaussian process. Thus, variational approaches of sparse Gaussian formulations must be investigated and compared with the standard formulation, in order to direct future research goals and industrial applications.

Many researchers have approached robust optimisation in the past by adopting frequentist approaches that seek to construct a surrogate that outputs the mean and the variance of a physical process or experiment, such as polynomial chaos [152] and spectral tensor-train decomposition [200]. However, this approach is not suitable for all classes of robust optimisation problems. The challenge provided by Airbus, for example, assumes that the simulation is a black-box and that the only way to understand its behaviour is to train a surrogate model under this assumption. A more appropriate paradigm for this application is the Bayesian approach, where the previous knowledge is stated by means of a prior function and the state of knowledge is described by a “degree of belief” instead. This, in turn, allows the creation of multi-level probabilistic models that are able to compute the solution in terms of a posterior distribution, which is parameterised by its mean and variance. In case, one can say that the uncertainty propagation was built-in in the problem and no external method is needed. Therefore, the Bayesian approach is selected for performing robust optimisation in this research.

Lastly, the application of uncertainty quantification and robust optimisation is framed under the digital transformation ideal to propose a way by which these methods can be integrated in aeronautical design and analysis. Thus, this thesis aims to extend the current applied aerospace knowledge on uncertainty quantification, robust optimisation and digital transformation by addressing the contributions discussed in

chapter 1, section 1.3.

Chapter 3

Probabilistic Modelling in the Bayesian Paradigm

This chapter introduces the fundamental modelling technique used in this work. This technique is probabilistic modelling in the Bayesian paradigm. Section 3.1 explains in an intuitive way the fundamental characteristics of the Bayesian paradigm and contrasts it with the more-usual frequentist approach. Section 3.2 further develops this theme and details the mathematical framework that allows one to think about “degrees of belief” when doing probabilistic modelling, as opposed to the frequentist approach. The section 3.3 details the inference engines used for Bayesian inference, explaining the difference between them. Section 3.4 expands the previous section discussion to the engineering context and sets its use in the context of engineering validation and verification (V&V). The important topic of how one checks a model either by comparing with data or other models is discussed in section 3.5. Based on the previous sections, in section 3.6, the robust optimisation problem is formulated in mathematical terms and a procedure to convert a deterministic problem into a probabilistic one is proposed. Finally, section 3.7 of this chapter, summarises the insights and conclusions of this chapter.

3.1 Introduction to Bayesian Paradigm

The basic insight of this work can be traced to Aristotle, in whose physics the ideal of a scientific study of nature is first formulated. The insight is that the human mind is capable of transcending the limitations of sense by reasoning from the clearly perceived appearances of things to their hidden, but intelligible underlying causes. In order to do that, one must reason by analogy (*kat' analogian*) and settle for a type of proportionate understanding¹. Bayesian analysis provides a way to do just that, to discover through inference relevant information and process it logically. Albeit, the modelling proposed in our work is designed to achieve a practical result, i.e, a quantitative prediction, and not an epistemic modelling, it has commonalities with it.

The main motivation is to use the models for several distinct purposes such as making predictions, designing inquiry and extracting information from data. For this purpose, it is assumed the existence of causal relations underlying the phenomenon or system of interest, and based on it, one can develop mathematical models with prediction capabilities. Models are abstractions of reality that often start as very high-level representations addressing a particular set of needs that progressively evolve as knowledge is gained.

Many models have closed-form solutions that enable a fast analytic evaluation. With this perspective, a model can be redefined as the map between the input parameters and the output interesting quantities (IQs), despite the fact that the full problem might need to be computed in order to retrieve the IQs. This map does not always need to be known analytically, but in general, one needs to be able to drive/measure the input, evaluate the model and measure the output, which are the minimum requirement for experiments. These models, as well as their inputs, may very well depend on the parameters $\mathbf{p}$, which affect their outputs. In order to have any predictive capability, models need to be evaluated, associating a particular input to a particular output. In its simplest form, the model takes the form of a mathematical mapping, $\mathbf{y} = f(\mathbf{x}(\mathbf{p}))$, that describes the relation between input(s) and output(s). Once the model, its parameters

¹Aristotle, Physics, Book I, chap. 7, 191a8

and IQs are defined, several kinds of analyses can be performed, not necessarily in any specific order. This particularity allows us to talk about a Bayesian optimisation framework that is implemented under the form of a numerical toolbox.

Several problems of engineering interest require an understanding of the uncertainties involved. This understanding answers questions related to the role that uncertainties play in the results of the problem under investigation. The source of uncertainties are normally classified into two categories, namely, aleatoric and epistemic uncertainties [29]. The aleatoric uncertainties are the uncertainties which cannot be reduced because we do not have control on them. The epistemic uncertainties are uncertainties which could be reduced by more accurate measurements and better modelling of the physics. The distinction is not sharp because it depends on the problem setting and on subjective judgement.

In the Bayesian point of view both types of uncertainty are represented in a way that does not depend on the problem setting. Normally, the uncertainties are mathematically represented by random variables and associated with a particular probability distribution. They can be taken into account in three ways. By assumption, where probability distributions are assigned to the parameters relying on experience and wisdom. By measurement, in which extensive measurements of the parameters are carried out and probability distributions are fitted to these experiments, and, lastly, by inference, the probability distributions are reconstructed using measurements of the IQs.

The probabilistic modelling consists of two basic frames. The first one is the reality itself or environment in which one expects to deploy the model. The second one is the self-contained logical frame of the probabilistic modelling, namely, the model. When the model one possess is an accurate description of the deployment environment, it is relatively safe to say that the model perform well in the real world as well. The reason is that Bayesian learning is arguable optimal in the model. Bayesian model allows Bayesian inference, by using a learning mechanics that passes back and forth between these two frames. This approach provides a data-based quantitative ranking

of hypothesis, given the assumption, that guarantees the best possible answer on the model perspective based on the quality of the information fed into it.

In order to employ Bayesian methods, the problem formulation must have passed beyond the initial, often exploratory phase. The sample space, the hypothesis space, the prior probability distributions, the sampling distribution and, more importantly, the model must all have been chosen and defined. However, it is also possible under the Bayesian paradigm to realise inferences for problems where not all components are known or have been defined. In this case, information-theoretic concepts such as maximum entropy can be used. One reason that statistical models usually correspond to many different detailed process models is because they rely upon distributions like the normal, binomial, Poisson, and others. These concepts are justified on the basis that many natural phenomena are governed by maximum entropy distributions. Therefore, one can utilise for a particular problem, additional theoretical concepts to fill in the unknown data, complementing the lack of information, to infer useful results.

3.2 Bayesian Probabilistic Modelling

The aim of this section is to present an up-to-date overview of the principles used for probability modelling from a data-centric perspective in a Bayesian way and to fix the notation used in later chapters. The fundamental and tacit assumption of statistical analysis, that is kept in Bayesian probabilistic modelling, is the assumption of **exchangeability**. This assumption affirms that the n values y_i are exchangeable, i.e, the values y_i may represent uncertainty as a joint probability distribution $p(y_1, \dots, y_n)$ which is invariant to permutations of the indexes.

3.2.1 Bayesian Data Analysis

The process of Bayesian analysis consists generally speaking in the solution of three different, but interrelated fundamental problems. The first problem is how to set up the full probability model, that is, a joint probability distribution for all

the interesting quantities in the problem. This model must be consistent with the underlying physical assumptions and to the data collection process. The second problem is how to condition on the observed data and how to compute the posterior through a process called Bayesian inference. The third and last problem in Bayesian Data Analysis is how to evaluate the fit of the model and the posterior distribution. Based on it, one can study its sensitivity under different scenarios and draw conclusions. If the model does not satisfy any of the criteria and the requirements, one can alter or expand the model. In summary, Bayesian Data Analysis consists of three steps: a) the setting of a joint probability distribution (**modelling problem**), b) the computation of the posterior (**inference problem**), c) the evaluation of the model fitness (**validation problem**).

3.2.2 Probabilistic Modelling

The probabilistic modelling process is iterative [201] and allow us to model uncertainty using random variables and probability distributions. In engineering, mathematical models are employed to make predictions based on assumptions about the problem of interest. Hence, Newtonian mechanics maybe used to predict the motion of an object, giving precise results. Normally, one does not need to represent the uncertainty associated with such results. However, there are problems and scenarios that can benefit from a probabilistic representation, since probability distributions are a natural way to represent uncertainty.

Given the state of knowledge required to fully characterise a probabilistic prediction is called likelihood. The epistemic limit of a probabilistic model may be estimated from the uncertainty associated with its likelihood. In the example used in the previous paragraph, by analogy, the limit of the model would be the limit of the theoretical framework of Newtonian mechanics when compared with relativistic mechanics in a problem involving fast particles.

In Bayesian probabilistic modelling, the prior distribution represents the uncertainty associated with the latent variables or parameters (noted by θ) in the model before

the observation of data. Related to the prior distribution is the posterior distribution. The posterior distribution represents the probability distribution of the parameters taken into account, what as “learned” or “inferred” from the observed data. The process of computing the posterior distribution using the data and the likelihood to update the prior distribution is called Bayesian inference. Hence, one can say that the basic building blocks of Bayesian data analysis are the problem assumptions, the prior, the likelihood that are used to create a joint probability distribution over the parameters and data that the model is based on. Mathematically, the conditioning (i.e, the training with the dataset) on the known value of the data $\mathbf{y}$, yields the posterior distribution.

Bayesian inference, that is, to draw Bayesian statistical conclusions about a parameter θ , or unobserved data $\hat{\mathbf{y}}$, are made in terms of probability statements. These probability statements are conditional (i.e, they depend upon) on the observed value of $\mathbf{y}$, and in our notation are written simply as $p(\theta \mid \mathbf{y})$ or $p(\hat{\mathbf{y}} \mid \mathbf{y})$. In order to make probability statements about θ given $\mathbf{y}$, we must begin with a model providing a joint probability distribution for θ and $\mathbf{y}$. The joint probability distribution function can be written as a product of two distributions that are often referred to as the prior distribution $p(\theta)$ and the sampling distribution $p(\mathbf{y} \mid \theta)$, respectively:

$$\begin{array}{lcl} \text{joint probability distribution} & = & \text{prior distribution} \times \text{sampling distribution} \\ \text{for } \theta \text{ and } \mathbf{y} & & p(\theta) \qquad \qquad p(\mathbf{y} \mid \theta) \end{array} \quad (3.1)$$

In the equation 3.1 the prior distribution reflects the variability one wants to assign to the input and parameters, while the sampling distribution reflects the uncertainty associated with the model itself. In mathematical terms,

$$p(\theta, \mathbf{y}) = p(\theta)p(\mathbf{y} \mid \theta) \quad (3.2)$$

The posterior distribution is calculated using the Bayes rule:

$$p(\mathbf{x} \mid \mathbf{y}, \theta) = \frac{p(\mathbf{y} \mid \mathbf{x}, \theta)p(\mathbf{x} \mid \theta)}{p(\mathbf{y} \mid \theta)} \quad (3.3)$$

In the equation above, $\mathbf{x}$ denotes the latent variables or parameters, the joint probability distribution of the model θ is the term $p(\mathbf{x}, \mathbf{y} \mid \theta)$, the likelihood is the term $p(\mathbf{y} \mid \mathbf{x}, \theta)$ and the prior is $p(\mathbf{x} \mid \theta)$. Hence, the posterior models our knowledge along with its uncertainty about the correct parameters for the model θ . The previous expression can be computed as an integral, which often is a hindrance for complex multi-level Bayesian models, since such integrals are mathematically intractable. The next section presents some inference engines that were developed with the goal of mitigating this issue.

$$\frac{p(\mathbf{y} \mid \mathbf{x}, \theta)p(\mathbf{x} \mid \theta)}{p(\mathbf{y} \mid \theta)} = \frac{p(\mathbf{x} \mid \mathbf{w}, \theta)p(\mathbf{x} \mid \theta)}{\int p(\mathbf{y} \mid \mathbf{x}, \theta)p(\mathbf{x} \mid \theta)d\mathbf{x}} \quad (3.4)$$

3.2.3 Marginal Likelihoods

The term $p(\mathbf{y} \mid \theta)$ in equation 3.3 is called marginal likelihood and has properties that allows one to capture the model complexity. This is interesting because when comparing different possible models for the same problem, it makes possible to treat the correct model as an unobserved variable that can be inferred from data using the Bayes' theorem. Hence, the same mechanism used for model inference is used for parameter inference, giving additional flexibility to Bayesian data analysis. In Bayesian probabilistic modelling, there is no clear theoretical distinction between the parameters and the model, since both are modelled by random variables that given data can be themselves inferred. It is this fact that gives the Bayesian paradigm an advantage when building hierarchical models, because a single model and its parameters may consists of several models and their parameters, while the computations are realised with the same process or mechanism.

The first problem in Bayesian probabilistic modelling is to compute the posterior distribution, which is done by the Bayes' theorem as explained in the previous section. A second problem is how to make predictions, considering uncertainty. This can be done using the basic rules of probability, which leads to the following mathematical

result, known as the posterior predictive distribution:

$$p(y^* | y) = \int p(y^* | \mathbf{x}, \theta) p(\mathbf{x} | \mathbf{y}, \theta) p(\theta | \mathbf{y}) d\mathbf{x} d\theta \quad (3.5)$$

Following [29], one can postulate a loss function for each pair of output y^* and a action $\mathcal{A}$. Then, minimise the expectancy of the posterior predictive distribution:

$$\mathcal{A}_{opt} = \min \int \mathcal{L}(y^*, \mathcal{A}) p(y^* | \mathbf{y}) dy^* \quad (3.6)$$

The most significant term in equation 3.6 is $p(y^* | \mathbf{y})$, the predicted distribution, which is obtained from the marginal likelihood, yielding

$$p(y^*, \mathbf{y}) = \int p(y^* | \mathbf{x}, \theta) p(\mathbf{y} | \mathbf{x}, \theta) p(\mathbf{x} | \theta) p(\theta) d\mathbf{x} d\theta \quad (3.7)$$

This result may be normalised by dividing the expression by $p(\mathbf{y})$, which in turn corresponds to the prior

$$p(y^* | \mathbf{y}) = \frac{p(y^*, \mathbf{y})}{p(\mathbf{y})} \quad (3.8)$$

Hence, one can conclude that Bayesian inference does not depend on an arbitrary choice and is invariant. Moreover, the effects can be placed in either the prior or the likelihood without affecting the outcome. What is important thus are the effects placed on the data distribution. The distinction between prior and likelihood is important because it is convenient, since conventional Independence makes more intuitive to assign structure to the data. The following discussion on hierarchical models intend to make this point clearer.

3.2.4 Bayesian Hierarchical Models

A Bayesian hierarchical model is a probabilistic model that is written in several hierarchical levels (sub-model) and capable of computing the posterior distribution using the Bayesian method. As an example, one can assume there is a random variable y , that was a Gaussian distribution with a parameter θ , which consists in a mean ϕ

and a variance σ . Assume also that the parameter θ has a Gaussian distribution with mean μ and variance ψ . In addition to that, assume that μ follows a standard Gaussian distribution $\mathcal{N}(0, 1)$. The parameter μ is called a hyper-parameter and the standard Gaussian distribution, $\mathcal{N}(0, 1)$, is called a hyper-prior. That is:

$$y \mid \phi \sim \mathcal{N}(\psi, \sigma) \quad (3.9)$$

$$\phi \mid \mu \sim \mathcal{N}(\mu, \psi) \quad (3.10)$$

$$\mu \sim \mathcal{N}(0, 1) \quad (3.11)$$

This simple example can be generalised as follows. In hierarchical models, ϕ is not known and the prior distribution is $p(\theta)$. The appropriate Bayesian posterior distribution is of the vector (ϕ, θ) . The joint prior distribution is

$$p(\phi, \theta) = p(\phi)p(\theta \mid \phi), \quad (3.12)$$

and the joint posterior distribution is

$$\begin{aligned} p(\phi, \theta \mid y) &\propto p(\phi, \theta)p(y \mid \phi, \theta) \\ &= p(\phi, \theta)p(y \mid \theta) \end{aligned} \quad (3.13)$$

The data distribution, $p(y \mid \phi, \theta)$, depends only on θ ; the hyper-parameters ϕ affect y only through θ .

3.2.5 Surrogates, Emulators and Reduced-Order Models

Engineering approximation methods and its use in optimisation originated in the 1970s [202]. These models are known under a myriad of different names such as response surfaces, kriging, splines, among others and are often tailored to suit a specific application. The general name is “surrogate” model. In this thesis and in the published literature the names surrogate model, emulators and reduced-order models are often

used interchangeably. However, for the sake of organisation and “*economy*” of thought², it is convenient to adopt a system of classification.

Eldred et al. [203] introduces a classification of surrogates that consists in three different classes: data-fit models, reduced-order models, and hierarchical models. Data-fit models are mathematical approximations of non-physics-type which involves interpolation or regression of a set of data. These surrogates are characterised by the number of data points they use and parameters related to their mathematical formulation. Examples of data-fit models are polynomial response surfaces, kriging, neural networks, radial basis functions, splines and Gaussian processes.

Another different type of surrogate model is the hierarchy type of model, which is also known as multi-fidelity, variable fidelity or variable complexity model. A hierarchy model (not to be confused with the Bayesian hierarchy model in the previous subsection) is a physics-based model of lower-fidelity, that is used as a surrogate in place of the high-fidelity model that is often the model used in computer simulations. The last type in this criteria is the reduced-order model which can be considered as part of the hierarchy model as well. Reduced-order model are based on numerical simplifications of the physics model and often employ numerical mathematical techniques that reduce the computational cost of evaluation. Examples of reduced-order modelling techniques are proper orthogonal decomposition [204] in CFD.

Bayesian probabilistic modelling allows the modeller to investigate the relations among two or more observable parameters, that is to say, *how does the outcome quantity, $\mathbf{y}$, vary as a function of a vector of quantities $\mathbf{x}$?* In Bayesian probabilistic terms, this question translates to *what is the **conditional distribution** of $\mathbf{y}$ given $\mathbf{x}$ taken as $p(\mathbf{y} \mid \theta, \mathbf{x})$ under a model in which the n observation $(\mathbf{x}, \mathbf{y})_i$ are exchangeable?*

To investigate the probabilistic relation between a pair of input-output, it is necessary to employ regression models that must satisfy at least three goals [205]: 1) understanding the behaviour of $\mathbf{y}$, given $\mathbf{x}$, 2) predicting $\mathbf{y}$, given $\mathbf{x}$, for future observations and 3) predicting how $\mathbf{y}$ would change if $\mathbf{x}$ were changed in a specified

²Mach, Ernst. “The economical nature of physical inquiry.” Popular scientific lectures (1882): 186-213.

way. The complete Bayesian regression model consists of a distribution for $\mathbf{x}$, $p(\mathbf{x} | \psi)$ indexed by a vector of parameters ψ . Thus, it has a joint likelihood, $p(\mathbf{x}, \mathbf{y} | \psi, \theta)$ and prior distribution, $p(\psi, \theta)$. In a Bayesian regression, the distribution of $\mathbf{x}$ does not necessarily give information about the conditional distribution of $\mathbf{y}$ given $\mathbf{x}$, i.e, it does not assume that prior independence of the parameters θ determines $p(\mathbf{y} | \mathbf{x}, \theta)$. Also, it does not assume that the parameters ψ determines $p(\mathbf{x} | \psi)$ either. Thus, from a Bayesian perspective, the fundamental characteristic that defines a “regression model” is that it ignores the information yielded by $\mathbf{x}$ about (ψ, θ) . This fact may be explained as follows. Let the parameters ψ and θ be independent in their prior distribution. Mathematically, one can write:

$$p(\psi, \theta) = p(\psi)p(\theta) \quad (3.14)$$

The factors of the posterior distribution are

$$p(\psi, \theta | \mathbf{x}, \mathbf{y}) = p(\psi | \mathbf{x})p(\theta | \mathbf{x}, \mathbf{y}) \quad (3.15)$$

the second factor can be analysed by a standard regression model without loss of information:

$$p(\theta | \mathbf{x}, \mathbf{y}) \propto p(\theta)p(\mathbf{y} | \mathbf{x}, \theta) \quad (3.16)$$

When the parameters $\mathbf{x}$ are chosen, for instance, by expert judgement or design of experiments their probability distributions $p(\mathbf{x})$ are known, and there are no parameters ψ . Thus, a practical advantage of using this regression model is that it makes easier to specify a realistic conditional distribution of one variable among others than a joint distribution on all the parameters.

3.3 Bayesian Inference Engines

This section is intended to give an overview of the possible mechanisms underlying the Bayesian computation of the posterior distribution. This knowledge is important

because for many decades Bayesian methods were hindered because of computational issues, even if conceptually speaking they are vastly superior to mainstream statistics based on the frequentist approach. Bayesian inference engines may be broadly classified into two classes: Non-Markovian and Markovian methods. Non-Markovian methods are grid computing [206], quadratic methods [207] and variational inference [208, 209]. Markovian methods are Markov Chain Monte Carlo [210], Gibbs sampling [211], random walk Metropolis [212], Metropolis-Hastings [213], Hamiltonian Monte Carlo [214] and sequential Monte Carlo [215].

The standard theoretical approach is the Markov chain simulation (MCS) and nowadays its usefulness in engineering lies in its conceptual value. For high-dimensional problems where the *curse of dimensionality* plays a significant role, MCS is not efficient [205]. A better alternative in terms of efficiency is the Hamiltonian Monte Carlo algorithm (HMC). However, in the era of big data, HMC does not scale well with higher dimensions. This has led the research and implementation of variational inference (VI) methods.

Non-Markovian methods may provide under very specific circumstance a fast and accurate enough approximation to the posterior. The most direct approach to approximate the posterior is called grid computing, which consists of a brute-force approach to compute the prior and the likelihood point-wise. A second method is the quadratic approximation (also known as the *Laplace method*), which consists of approximating the posterior, $p(x)$, with a normal distribution, $q(x)$. The Laplace method has two steps. The first one is to compute mode of the posterior distribution, i.e., $\mathbb{E}[q(x)]$. The second step is to compute the Hessian matrix, that is, the standard deviation of $q(x)$. The first step is usually numerically straightforward, while the second step requires more caution. This is the case because the computation of the standard deviation of $q(x)$ requires the evaluation of the curvature at the mode/mean of $q(x)$. Mathematically, this operation requires the inversion of the square-root of the Hessian matrix³. The last non-Markovian method to be covered in this section is the

³A Hessian matrix is the matrix of the second derivative of a function. The inverse corresponds to the the covariance matrix.

variational inference, which is presented next.

3.3.1 Variational Inference

Variational inference is the most powerful non-Markovian method. The majority of Bayesian data analysis is currently done using Markovian methods (see the next section), however these methods can be too slow for some classes of problems and perhaps are not sufficiently scalable. In this case, one alternative is to employ variation methods.

The fundamental idea underlying variational methods is to approximate the posterior distribution using less expensive and simpler probability distributions. This approach is, in principle, similar to the Laplace method mentioned above, but in the case of variational methods, it is done in a more elaborate way. In VI, the posterior approximate distribution is computed by solving an optimisation problem using a mathematical measure of closeness as objective function. There are many possible measures of closeness [208, 216]. Many variational inference formulations favour information-theoretic measures, such as Kullback-Leibler (KL) divergence A.1.5. The KL divergence can be computed as:

$$D_{KL}(q(\theta) \| p(\theta | y)) = \int q(\theta) \log \frac{q(\theta)}{p(\theta | y)} d(\theta) \quad (3.17)$$

The less complex distribution used to approximate the posterior, $\theta | y$, alluded above is denoted by q . This distribution is called variational distribution. By optimising its parameters (called variational parameters), one can find the parameters that make q as close as possible to the posterior distribution. The expression 3.17 shows that it is not possible to use the posterior, since it is an unknown distribution. Thus, it is necessary to execute some algebra steps to unravel a mathematical expression that can be computed and coded. The first step is to expand the term of the conditional

distribution with its definition

$$D_{KL}(q(\theta) \| p(\theta | y)) = \int q(\theta) \log \frac{q(\theta)}{\frac{p(\theta, y)}{p(y)}} d(\theta) \quad (3.18)$$

then rearranging the terms, one gets

$$= \int q(\theta) \log \frac{q(\theta)}{p(\theta, y)} p(y) d(\theta) \quad (3.19)$$

the logarithmic properties allows one to write

$$= \int q(\theta) \left(\log \frac{q(\theta)}{p(\theta, y)} + \log p(y) \right) d(\theta) \quad (3.20)$$

Rearranging the above equation, one gets

$$= \int q(\theta) \log \frac{q(\theta)}{p(\theta, y)} d(\theta) + \int q(\theta) \log p(y) d(\theta) \quad (3.21)$$

The integral of $q(\theta)$ is 1 and we can move $\log p(y)$ out of the integral, then we get:

$$= \int q(\theta) \log \frac{q(\theta)}{p(\theta, y)} d(\theta) + \log p(y) \quad (3.22)$$

And using the properties of the logarithm:

$$D_{KL}(q(\theta) \| p(\theta | y)) = \underbrace{- \int q(\theta) \log \frac{p(\theta, y)}{q(\theta)} d(\theta)}_{\text{evidence lower bound (ELBO)}} + \log p(y) \quad (3.23)$$

Hence, the marginal likelihood is always equal or larger than the evidence lower bound (ELBO), because $D_{KL} \geq 0$, then $\log p(y) \geq \text{ELBO}$. Thus, minimising the Kullback-Leibler divergence is equivalent to maximising the ELBO. This maximisation of the evidence lower bound translates mathematically to the approximation of $q(\theta)$ as close as possible to the posterior, $p(\theta | y)$. Note also that the term $\log p(y)$ is a constant. The variational approximation happens when $q(\cdot)$ is selected. Theoretically, $q(\cdot)$ can be any distribution. This fact is leveraged by assuming that the high-dimensional

posterior can be written by independent one-dimensional probabilistic distributions. This is known as the mean-field approximation. This is written as follows

$$q(\theta) = \prod_j q_j(\theta_j) \quad (3.24)$$

The q_j distributions normally are chosen to be from the exponential distribution class. The exponential family includes many of the distributions such as normal, exponential, beta, Dirichlet, gamma, Poisson, categorical, and Bernoulli (see appendix A.1.2 for a description of this distribution). Having restated the Bayesian inference as an optimisation problem, it is possible to solve it using the available optimisation methods in order to maximise the ELBO.

3.3.2 Hamiltonian Monte Carlo

Hamiltonian Monte Carlo (HMC) also called Hybrid Monte Carlo [217] is a Markovian method that combines the Markov Chain Monte Carlo method with powerful insights of molecular dynamics [214]. Neal [218] introduced HMC in Bayesian statistics and machine learning. For a comprehensive and detailed explanation of HMC, the reader is referred to [219] and Betancourt's paper [214].

In order to understand HMC, one must understand the basics of molecular dynamics and its fundamental insights related to Hamiltonian dynamics. Hamiltonian dynamics is based on few but powerful ideas. Assume that q and p are generalised coordinates [220] denoting a physical object's position and momentum, respectively. The action or Hamiltonian of the object is

$$H(q, p) = U(q) + K(p) \quad (3.25)$$

In the above, equation U , is the potential energy and K is the kinetic energy. The object's movement changes through time and is described by the follow system of

differential equations (Hamilton's equations)

$$\begin{aligned}\frac{dq_i}{dt} &= \frac{\partial H}{\partial p_i} \\ \frac{dp_i}{dt} &= -\frac{\partial H}{\partial q_i}\end{aligned}\tag{3.26}$$

By the conservation of energy and above equations, it is known that an increase in potential energy decreases the kinetic energy. Thus, the rate of change of the Hamiltonian is

$$\frac{dH}{dt} = \sum_{i=1}^d \left[\frac{dq_i}{dt} \frac{\partial H}{\partial q_i} + \frac{dp_i}{dt} \frac{\partial H}{\partial p_i} \right] = \sum_{i=1}^d \left[\frac{\partial H}{\partial p_i} \frac{\partial H}{\partial q_i} + \frac{\partial H}{\partial q_i} \frac{\partial H}{\partial p_i} \right] = 0.\tag{3.27}$$

The d in the summation symbol represents the dimension of the space.

Numerical Solution to Hamiltonian Dynamics

It is not possible to solve a problem involving Hamiltonian Dynamics analytically, except for a number of simple problems. Hence, it is necessary to employ numerical techniques that discretise the domain, allowing the numerical solution of the differential equations given some step size ϵ . The most common numerical method is the leapfrog method [221], which tends to perform better than the Euler's method [222] and is relatively easy to implement. The leapfrog method first starts with a $\frac{\epsilon}{2}$ step p . This initial step is then used to update q and p , according to the following update equations

$$p_i(t + \epsilon/2) = p_i(t) - (\epsilon/2) \frac{\partial U}{\partial q_i}(q(t))\tag{3.28}$$

$$q_i(t + \epsilon) = q_i(t) + \epsilon \frac{p_i(t + \epsilon/2)}{m_i}\tag{3.29}$$

$$p_i(t + \epsilon) = p_i(t + \epsilon/2) - (\epsilon/2) \frac{\partial U}{\partial q_i}(q(t + \epsilon))\tag{3.30}$$

The leapfrog method of integration has two interesting properties: time reversibility and the symplectic property. The time reversibility property simply means that a n -step forward integration followed by n -step backward integration yield the same starting

position. The symplectic property means that the method takes advantage of phase space conserving properties. These properties guarantee that this method diverges less quickly than the Euler's method.

Markov Chain Monte Carlo using Hamiltonian dynamics

The fundamental idea of the Hamiltonian Monte Carlo is to utilise the mathematical modelling of Hamiltonian dynamics to generate proposals of points as required by the Markov Chain Monte Carlo method. This is done by the use of momentum variables that are introduced as part of auxiliary probability distributions. This auxiliary probability distribution accepts a marginal distribution as part of its target distribution. Let $p(q) \propto \pi(q)L(q \mid D)$ be the target distribution, that is, the posterior of given prior π and dataset $\mathcal{D}$. Here, q denotes the variables of interest. Then the auxiliary probability distribution is defined as

$$\begin{aligned} P(q, p) &= P(q) \exp(-K(p)) \\ &= \frac{1}{Z} \exp(-U(q)/T) \exp(-K(p)) \end{aligned} \quad (3.31)$$

$$U(q) = -\log[\pi(q)L(q \mid D)] \quad (3.32)$$

$$K(p) = \sum_{i=1}^d \frac{p_i^2}{2m_i} \quad (3.33)$$

It is assumed that the momentum variables are independent Gaussian distributions. The goal is to sample from $p(q, p)$, since p is independent of q and is numerically tractable, it is possible to update it.

The update of the variable q is done with Metropolis-Hastings equipped with a Hamiltonian mechanism of proposal. The initial propose for (q^*, p^*) is done using the leapfrog method. Since the proposal of the Metropolis-Hasting is symmetric, its ratio is the ratio between the probability distribution of the new and the old points.

Mathematically, this has the form

$$A(q^*, p^*) = \frac{P(q^*, p^*) Q(q, p | q^*, p^*)}{P(q, p) Q(q^*, p^* | q, p)} = \frac{P(q^*, p^*)}{P(q, p)} = \exp(-H(q^*, p^*) + H(q, p)) \quad (3.34)$$

Hence, jointly samples of both p and q are realised. The method described can be presented in form of the algorithm 3.1. A very powerful Bayesian inference is STAN [192], which is equipped with a No-U-Turn sampler [223], which eliminates some of the steps in computing the target distribution, dramatically improving its efficiency.

Algorithm 3.1: Monte Carlo Hamiltonian Algorithm

```

1 Randomly select a location  $\theta_0$  from a initial probability distribution;
2 for  $t = 1, \dots, T$  do
3   Generate a random initial momentum from a probability distribution:
                                      $m \sim \mathcal{N}(\mu, \Sigma)$                                      (3.35)
4   Use the leapfrog integrator to solve for the path of a sledge moving over
   NLP space for a time period  $T$ ;
5   Record the momentum of the sledge  $m'$  and its position in parameter space
    $\theta_{t+1}$ , after time  $T$  has elapsed;
6   Compute
                                      $r = \frac{p(x | \theta_{t+1}) \cdot p(\theta_{t+1})}{p(x | \theta_t) \cdot p(\theta_t)} \times \frac{q(m')}{q(m)}$                                      (3.36)
7   Generate  $u \sim \mathcal{U}(0, 1)$ ;
8   if  $r > u$ ;
9     move to  $\theta_{t+1}$ ;
10  else;
11    remain at  $\theta_t$ 

```

3.4 Validation and Verification

In engineering, there is a distinction between validation and verification that is important to have in mind when working with data-driven engineering methods such as Bayesian probabilistic modelling. Verification is the process of checking that the solution or product achieves its goal without any issues, i.e, it verifies whether the developed product fullfills the requirements that one have. Thus, one may say that verification is the process that answers the question: *is the solution or the product*

correct? On the other hand, validation is the process of checking whether the solution or product satisfies the high level requirements. It answers the question: *is the solution or product under development the right one?*

As previously mentioned, the construction of a probabilistic model is a iterative process in which different models may be created and compared. Often, this process is automatically performed by the computer, however, it is enlightening to understand how the predictive accuracy may be computed and which steps are the next for improvements, as well as correcting for the bias in evaluating the models prediction of the data used to fit it. In this section, the conventional measures of predictive accuracy such as mean squared error, probabilistic prediction, log-likelihood, predictive accuracy for single data points, expected log predictive density for new data points (ELPD) and expected log point-wise predictive density for a new dataset (ELPPD).

3.4.1 Measures of Predictive Error

Point Prediction

In point prediction a single value is considered as a prediction of the unknown future outcome or observation. The measures of predictive accuracy for point prediction are called scoring functions. The most common metric of predictive accuracy used in point prediction is the mean squared error, that is defined as:

$$\frac{1}{n} \sum_{i=1}^n (y_i - E(y_i | \theta))^2 \quad (3.37)$$

Sometimes it is useful to have a weighted version of the mean squared error when a single variable is more important than the others. The weighted version of the mean squared error is

$$\frac{1}{n} \sum_{i=1}^n (y_i - E(y_i | \theta))^2 / \text{var}(y_i | \theta) \quad (3.38)$$

The mean squared error is very useful despite its simplicity because it is easy to compute and to interpret. However, it is not very appropriate for models that deviate from the normal or Gaussian distribution. for such cases, it is more useful to recur to

probabilistic prediction metrics.

Probabilistic Prediction - Log-likelihood

The aim of probabilistic prediction is to describe the inferences on $\hat{\mathbf{y}}$ taking the full uncertainties over $\hat{\mathbf{y}}$ into account. The most common scoring rules for probabilistic prediction are zero-one scores, quadratic and logarithmic. Scoring rules for prediction must have the local and proper properties. The logarithmic scoring rule is local and proper scoring rule because of its affine transformation and thus is commonly used for evaluating probabilistic predictions. Being proper in the mathematical sense translates as being a “accurate” metric, while the property of locality means that the predictions deemed “incorrect” or inaccurate may incur more penalties than the others.

3.4.2 Log-Likelihood

The logarithmic scoring rule is known as log-likelihood or log predictive distribution (also called log predictive density) is given by $\log p(y | \theta)$. In models consisting of Gaussian distribution with constant variance⁴, the log-likelihood is proportional to the mean squared error. The log-likelihood is important in statistical modelling because it is connected to the Kullback-Leibler information-theoretic metric (see A.1.5). In large sample sizes, the model with the lowest Kullback-Leibler information have the highest posterior probability. Hence, it makes sense to use the expected log-likelihood as a metric of the model fit.

Predictive Accuracy for Single Data Points

The out-of-sample predictive performance [205] is the ideal metric for measuring the model’s fit in a true data-generating process⁵. The out-of-sample predictive fit for a new data point $\tilde{y}_i$ using log-likelihood based score is defined as,

$$\log p_{\text{post}}(\tilde{y}_i) = \log E_{\text{post}}(p(\tilde{y}_i | \theta)) = \log \int p(\tilde{y}_i | \theta) p_{\text{post}}(\theta) d\theta \quad (3.39)$$

⁴In statistics, models featuring non-constant residual variance are called heterosdastic models.

⁵This is referred as external validation, refer to section 3.5.

In the notation above, θ , denotes the true model; y is the observed data; The distribution of a a single realisation, y , of the dataset is represented by $f(y)$. The future data or an alternative dataset is represented by the term $\tilde{y}$. The term $p_{\text{post}}(\tilde{y}_i)$ denotes the predictive density for $\tilde{y}_i$ induced by the posterior distribution $p_{\text{post}}(\theta)$. The terms p_{post} and E_{post} are used to denote any probability or expectation that averages over the posterior distribution of θ .

Averaging Over the Distribution of Future Data

As mentioned, the future data represented by the term $\tilde{y}_i$ in equation 3.39 is not known. Hence, the expected out-of-sample log predictive density (ELPD) (or expected out-of-sample log-likelihood) is used. The expected out-of-sample log predictive density for a new data point is defined as follows:

$$\begin{aligned} \text{ELPD} &= \text{Expected log predictive density for a new data point} \\ &= E_f(\log p_{\text{post}}(\tilde{y}_i)) = \int (\log p_{\text{post}}(\tilde{y}_i)) f(\tilde{y}_i) d\tilde{y} \end{aligned} \quad (3.40)$$

When evaluating the model, it is possible to compute the term, p_{post} , however, in general, it is not possible to know the data distribution f . Thus, in order to preserve the comparability property within the given dataset, one defines a metric of predictive accuracy for the n data points taken one at a time:

$$\begin{aligned} \text{ELPPD} &= \text{Expected log pointwise predictive density for a new dataset} \\ &= \sum_{i=1}^n E_f(\log p_{\text{post}}(\tilde{y}_i)) \end{aligned} \quad (3.41)$$

In this case, there must be a division of the data y into individual data points y_i . Usually, for computational experiments generated by design of experiments (DOE), each doe point correspond to a discrete ordered pair $(\mathbf{x}, \mathbf{y})$, which makes this data division for ELPPD quite natural. It is interesting to note that there is no difference between joint or point-wise prediction given a point estimate for models with independent data

parameters. Thus,

$$p(\tilde{y} \mid \hat{\theta}) = \prod_{i=1}^n p(\tilde{y}_i \mid \hat{\theta}) \quad (3.42)$$

Nevertheless, one advantage of using a point-wise metric instead of the joint posterior predictive distribution, $p_{\text{post}}(\tilde{y})$ is the link between the point-wise calculation to cross-validation, allowing general approaches to the calculation of the out-of-sample fit using available data.

Evaluating Predictive Accuracy for a Fitted Model

The parameter θ is unknown in practice, thus one cannot compute log predictive density, $\log p(y \mid \theta)$. Hence, it is more useful to work with the posterior distribution, $p_{\text{post}}(\theta) = p(\theta \mid y)$, and compute the predictive accuracy of the trained model to the data by

$$\begin{aligned} \text{LPPD} &= \log \text{point-wise predictive density} \\ &= \log \prod_{i=1}^n p_{\text{post}}(y_i) = \sum_{i=1}^n \log \int p(y_i \mid \theta) p_{\text{post}}(\theta) d\theta \end{aligned} \quad (3.43)$$

The computation of this predictive density is done by evaluating the expectation using realisations from $p_{\text{post}}(\theta)$, which corresponds to querying the model to compute the posterior simulations, as follow:

$$\begin{aligned} \text{computed LPPD} &= \text{computed log pointwise predictive density} \\ &= \sum_{i=1}^n \log \left(\frac{1}{S} \sum_{s=1}^S p(y_i \mid \theta^s) \right) \end{aligned} \quad (3.44)$$

where $\theta^s, s = 1, \dots, S$; It is typically assumed that the number of simulation queries S is large enough to fully capture the posterior distribution. Thus, in the literature, the theoretical value of the LPPD (3.43) and the computation 3.44 are used interchangeably as the log point-wise predictive density or LPPD of the data. The LPPD of observed data y is an overestimate of the ELPPD for future data [205]. Hence the plan is to start with equation 3.44 and then apply some sort of bias correction to get a reasonable

estimate of equation 3.43.

3.5 Model Checking

Previous sections 3.2, 3.3 and 3.4 addressed the first two steps of a Bayesian analysis: the creation of a suitable probability model and the computation of the posterior distribution of all outputs of interest. It follows that there is a necessity of checking the accuracy of the assumptions to the best of the modeller’s knowledge, which usually includes checking it against the dataset. This step is often referred as model checking⁶. This section introduces the concepts necessary to perform a comprehensive model checking generally enough to be used with the majority of probability models. After the theoretical exposition, a summary of the model checking procedure to quickly aid the modeller is given in the end of this section.

Model checking is essential to probabilistic modelling and it forms one of the pillars of probabilistic modelling [205]. The Bayesian inference complete process, from prior to posterior is executed based on the assumptions of the probabilistic model. If the assumptions are weak, this prior-to-posterior inference may yield misleading results. An acceptable Bayesian analysis must include some check of accuracy or adequacy of the fit of the model to the data, increasing the plausibility of the model for its purposes. In technical terms, often this is discussed as a problem of sensitivity to the prior distribution, however, this care must be extended to the likelihood model too. In common technical usage, the term “probabilistic model” consists of the sampling distribution, the prior distribution the explanatory variables or parameters and the hierarchical structure when available.

In probabilistic modelling, it is possible to build based on the same dataset more than one model that can provide reasonable adequate fit to the data. This begs the question: given more than one model, which one is the best for the current purposes? Other models may differ substantially from the present model in the prior specification,

⁶Some design methodologies such as agile model-based systems engineering [183, 184] encompasses this step as an essential part of the design cycle.

the sampling distribution, or in what information is included as the model's parameters. It is possible that the present model provides an adequate fit to the data, but that posterior inferences differ under plausible alternative models. One possible way of answering this question in an engineering context is to perform a sensitivity analysis (SA). The basic question of a sensitivity analysis is: how much do posterior inferences change when other reasonable models are used? In Chapter 6, section 6.4, a sensitivity analysis is presented in the context of the robust optimisation problem investigated in this thesis.

In engineering it is important to have measures on accuracy so different models can be compared and the suitability of each model assessed. However, it is equally important to be capable of checking the model using simpler techniques as well. This step is important, because in any applied problem, there is information that is not formally included at the first iterations, either in the prior distribution or the likelihood. Checks are important because they may suggest if additional information is needed, suggesting the creation of a more accurate model for the parameters or even the data collection process [205].

The first way that a model can be checked is by comparing its results with the experiments. In the context of probabilistic modelling this is referred to as external validation. In the aeronautical context, this would be equivalent to make future predictions about shear, moment, and torque loads, for example, and comparing with real data captured by sensors and instrumentation in test flight. In this case, the model will be deemed “correct”, if the posterior means are correct on average and 50% intervals contain the true values. The results in this case as referred as posterior predictive checking, which use global summaries to check the joint posterior predictive distribution $p(\tilde{\mathbf{y}} | \mathbf{y})$, in the case of a high-dimensional model. The posterior predictive checking is a self-consistency check in which the fit of a model to data is checked by sampling values from the joint predictive distribution of the probabilistic model and comparing these samples to the observed data. If there are any systematic difference between the posterior and the collected data, the model fails. Hence, it is useful to examine the

results by inspecting graphical comparisons of the data to summaries from the posterior predictive simulation together with a measure of its statistical significance [205].

3.5.1 Graphical Posterior Predictive Check

The easiest way to check graphically the predictive posterior is, as expected, to plot together the predicted results with the actual data, in the context of probabilistic modelling this is refereed as direct data display. However, in many cases, where hyper-dimensionality and multilevel modelling plays an important role it is not feasible to make direct data displays to check the model. In such cases, the principles of quantitative visualisation ([224]) and exploratory data analysis (EDA) hold. A key principle of EDA is to exploit regular structure to display data more effectively [225]. In multilevel modelling, the model checking task allows one to compare batches of parameters to their reference distribution, because of its hierarchical structure. In this case, the replications correspond to new draws of a batch of parameters.

Another graphical way to check the fit of a probability model is to use residual plots [205]. Bayesian residuals are linear and nonlinear regression models, which are the core tools of applied statistics, are characterised by a function $g(x, \theta) = E(y \mid x, \theta)$, where x is a vector of predictors. Then, given the unknown parameters θ and the predictors x_i for a data point y_i , the predicted value is $g(x_i, \theta)$ and the residual is $y_i - g(x_i, \theta)$. This is sometimes called a “realized” residual in contrast to the classical or estimated residual, $y_i - g(x_i, \theta')$, which is based on a point estimate θ' of the parameters. A Bayesian residual graph plots a single realisation of the residuals based on a single random draw of θ . Classical residual plots can be thought of as approximations to the Bayesian version, ignoring posterior uncertainty in θ .

3.5.2 Marginal Posterior Predictive Checks

Marginal predictive checks are another approach to compute the probability distribution for each marginal prediction $p(\tilde{y}_i \mid y)$ and compare these distributions to the data in order to find any discrepancies. for each marginal posterior predictive

distribution, the tail-area probability for an ordered data can be computed in terms of a p -value as follows

$$p_i = \Pr(T(y_i^{\text{rep}}) \leq T(y_i) \mid y) \quad (3.45)$$

for ordered data we can compute a “mid” p -value,

$$p_i = \Pr(y_i^{\text{rep}} < y_i \mid y) + \frac{1}{2} \Pr(y_i^{\text{rep}} = y_i \mid y) \quad (3.46)$$

Another related approach is to use cross-validation predictive distributions to replace predictive distributions. Thus, for each data point comparing to the inference given all the other data:

$$p_i = \Pr(y_i^{\text{rep}} \leq y_i \mid y_{-i}) \quad (3.47)$$

where y_{-i} contains all other data except y_i . Cross-validation requires additional computation and is explained in the next section that deals with model validation. If the marginal posterior p -values concentrate near 0 and 1, the data are overdispersed compared to the model and if the p -values concentrate near 0.5 the data are underdispersed compared to the model. A summary intended to aid model checking tasks is contained in box that follows.

Model Checking

- Check the model by external validation
- Perform Posterior predictive checking
- Perform Marginal predictive checks
- Perform Sensitivity Analysis
- Graphical posterior predictive checks
 - Direct data display
 - Displaying summary statistics or inferences
 - Residual plots and binned residual plots
- Check using Measures of predictive accuracy
 - Predictive point estimation
 - Mean squared error
 - Log predictive density or log-likelihood
 - Out-of-sample predictive fit for a new data point y_i^*
 - Expected log predictive density for a new data point
 - Expected log predictive density for a new dataset
 - Expected log predictive density, given θ^*
 - Log pointwise predictive density (lppd)
 - Computed log pointwise predictive density (computed lppd)

3.6 Formulation of the Robust Optimisation Problem

This section describes the process of the robust optimisation problem formulation. This problem was initially considered one of the challenges of the robust optimisation problem, because of the way the dataset was structured. A simple eight step heuristic procedure is designed to guide the formulation of robust optimisation problems, where the dataset structure may seem elusive at first. By following this eight step procedure, any problem can be formulated with great ease. This is important for industrial alignment, since most industries give preference to straightforward procedures that can

be part of the internal industrial documentation flow. The heuristic proposed in this section can also be used to convert deterministic problems into probabilistic ones and is central to the Model-based Bayesian framework proposed in appendix A.3.

The dataset provided by Airbus is intended to provide a realistic use case to evaluate the performance of robust, multidisciplinary optimisation techniques. For confidentiality reasons the data supplied in this dataset has been normalised according to a variety of metrics. Although the trends in the data are representative of reality, the values themselves are not. The data is packaged in Matlab structures within a .mat file. There are two structures: aero, for the aero performance data, and loads, for the loads response data. In the loads response data, the last Design of experiment (DOE) point (DOE 201) is the baseline, unmodified aircraft, while in the aero data, the first DOE point (DOE 1) is the baseline, unmodified aircraft. In addition, in the aero data, DOE 2 and DOE 3 are cases with zero stiffness changes and maximum twist-on and twist-off respectively. As a brief overview, the dataset is showed again in the figure 3.1.

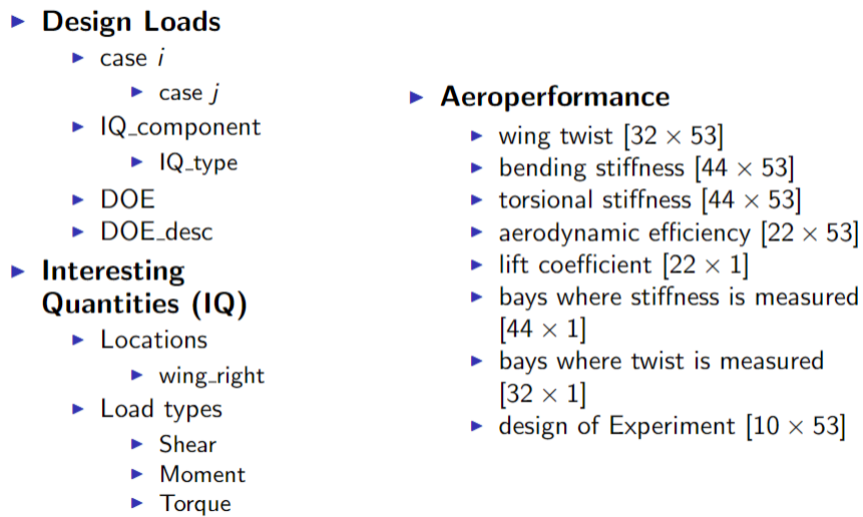


Figure 3.1: Overview of the Dataset

The next set of tables and figures gives more detailed information on the dataset used in this thesis. The first thing to notice is the location of the wingstations where the loads are calculated. For the purposes of this thesis, the loads over the winglet and the horizontal tail are not required. This is shown in table 3.1.

Locations	Description
wing right	spanwise stations along the right wing
winglet right	spanwise stations along the right winglet
horizontal tail right	spanwise stations along the right horizontal tail

Table 3.1: Locations of the wingstations contained in the dataset

Figure 3.2 shows the axis system to which the interesting quantities (IQ) refer.

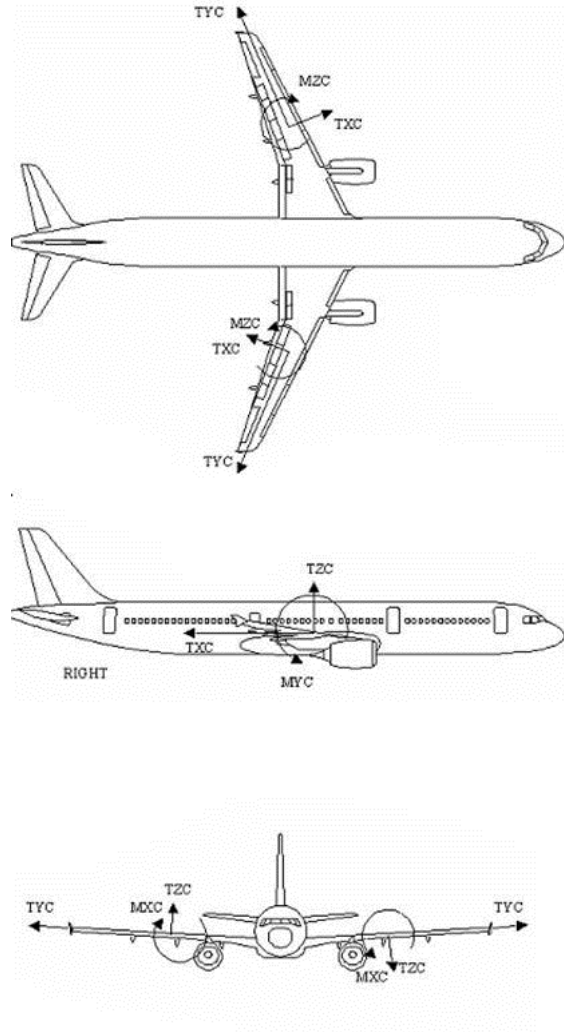


Figure 3.2: Aircraft Axis System

The loads investigated are the shear, the moment and the torque, as shown in table 3.2. The local lift coefficients along the wingspan are not necessary for the purpose of this investigation. Figure 3.3 shows the two wing twist jig shape locations that are referred as θ_1 and θ_2 along this work.

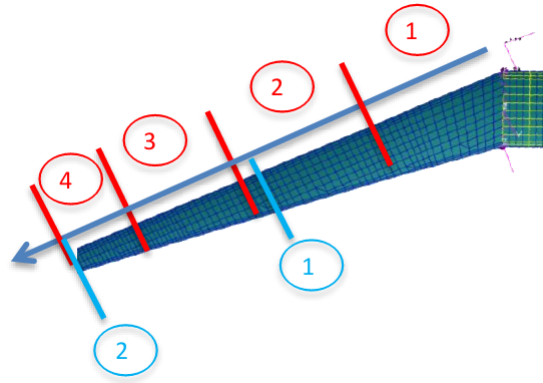


Figure 3.3: Twist angles at two span-wise locations

Loads Types	Description
shear	vertical shear force, positive upwards (TZC)
moment	bending moment along the aircraft longitudinal (fuselage) axis
torque	torque along the lateral (spanwise) axis, positive pitch up (MYC)
cl	coefficient of lift value, positive upwards (TZC)

Table 3.2: Types of loads contained in the dataset

Table 3.3 describes the aerodynamic dataset. The calculations were done by Airbus for 53 design of experiment points (DOE). The twist was calculated for 32 stations along the wing at 23 lift coefficients that corresponds to different points in the flight envelope, see figure 3.4. The structural stiffnesses are divided in two groups: bending and torsional stiffnesses. These were calculated at 44 locations. The parameter .LD corresponds to $\Delta L/D$, the difference in L/D between different DOE points.

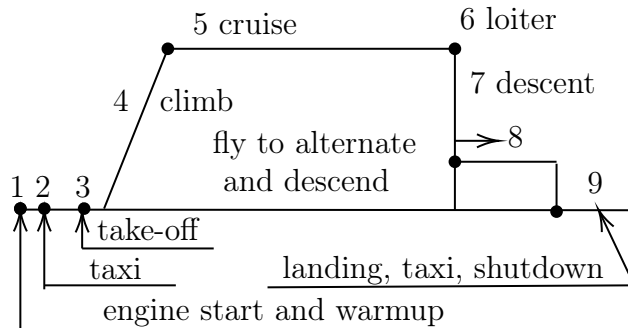


Figure 3.4: Typical Mission Profile

Aero	Description
.twist	A 32×53 array describing the change in jig twist applied at different stations along the wing (described by <code>aero.Bays_Twist</code>), where the rows are different stations, and the columns are different DOE points.
.EI	A 44×53 array describing the percent change in wing bending stiffness applied at different stations along the wing (described by <code>aero.Bays_EIGJ</code>), where the rows are different stations, and the columns are different DOE points.
.GJ	A 44×53 array describing the percent change in wing torsional stiffness applied at different stations along the wing (described by <code>aero.Bays_EIGJ</code>), where the rows are different stations, and the columns are different DOE points.
.LD	A 23×53 array describing the difference in L/D between the different DOE points and the baseline aircraft across different C_L values. The rows correspond to different C_L values across the polar, while the columns correspond to the different DOE points.
.Cl	A 23×1 vector describing the C_L values at which the L/D results in <code>aero.LD</code> are calculated. $C_L = 1$ corresponds to the design cruise point C_L .
.Bays_EIGJ	A 44×1 vector describing the normalized spanwise locations where the stiffness changes are applied on the wing.
.Bays_Twist	A 32×1 vector describing the normalized spanwise locations where the twist changes are applied on the wing.
.DOE	A 10×53 vector where the rows represent the different design variables as in <code>loads.DOE</code> . The twist values are extracted from <code>aero.twist</code> at $\eta = 0.5830$ and $\eta = 1$, while the stiffness values are extracted from <code>aero.EI</code> and <code>aero.GJ</code>

Table 3.3: Aerodynamic performance improvement values contained in the dataset

Table 3.4 shows how the loads dataset is structured. The data was calculated for 200 design of experiment points, in addition to the initial baseline (the first DOE). The loads are divided in two groups: gust and maneuver loads. They correspond to push over, pull up, roll, discrete gust, continuous turbulence load conditions for different operating conditions, including Mach, altitude, aircraft mass and thrust.

The loads data was generated using Airbus' certification standard simulation tools for a range of typical gust and manoeuvre cases. The loads cases provided occur across a variety of aircraft mass cases as well as different Mach and altitude points. They include discrete gust, continuous turbulence, as well as a number of different manoeuvres. The provided cases do not represent the actual design load cases but are sufficiently representative of the cases used to size the baseline aircraft for the purposes

Loads Types	Description
.ManData or .GustData	manoeuvre/gust cases data
.CASE{i}	data for Case Type i (push over, pull up, roll, discrete gust, continuous turbulence)
.CASEID{j}	data for subcase j (different operating conditions, including Mach, altitude, aircraft mass, thrust)
.{IQ_COMPONENT}	The component on the aircraft this set of IQ values applies to. Possible values are: wing_right, winglet, right, wing_aero_right, wing_aero_left, horizontal_tail_right.
.IQ_TYPE	The type of force the IQ describes (Example: shear, moment, torque, long, lat, vert). This is an $n \times 201$ matrix, where the rows correspond to different stations on the IQ_COMPONENT and the columns correspond to different DOE points.
.DOE	The structure summarising the input parameters for each DOE point (the design variations). The first two rows represent inboard and outboard twist, and the last eight rows represent percent changes in EI (inboard to outboard) and GJ (inboard to outboard)
.DOE_desc	A cell array corresponding to the labels for the different design variables in the DOE array

Table 3.4: Loads contained in the dataset

of the academic study. The precise cases which define the aircraft envelope may change depending on the wing stiffness and twist, for the purpose of this dataset they are considered constant as the changes would be minor. These loads cases represent those which produce the envelope shear, moment and torque values at different locations on the wing. These are not necessarily the highest stresses in the structure; however, this information is sufficient for the robust optimisation use case presented. The DOE simulations included 23 gust cases with 4 different gust types, 17 mass cases, 6 Mach numbers, 2 thrust settings, and 8 altitudes. This includes both continuous turbulence as well as discrete gusts. The DOE simulations included 21 different steady manoeuvre cases across the flight envelope (range of Mach / altitudes). The figure 3.5 in the next page shows a tree overview of the dataset used.

The number of DOEs for the load data and for the aeroperformance data are different (201 for the load dataset and 53 for the aeroperformance dataset), however, both datasets share the same parameters, i.e, the wing twist angles, θ_1 , θ_2 , at two

locations, at the wing root and at the wing tip; the bending stiffnesses, EI_1 , EI_2 , EI_3 , EI_4 , at four locations at the wingspan; the torsional stiffnesses, GJ_1 , GJ_2 , GJ_3 and GJ_4 , at four locations at the wingspan. The aeroperformance contain $\Delta(L/D)$ values for the overall aircraft as the output values, while the loads dataset contain values for the shear, moment and torque values across the wingspan.

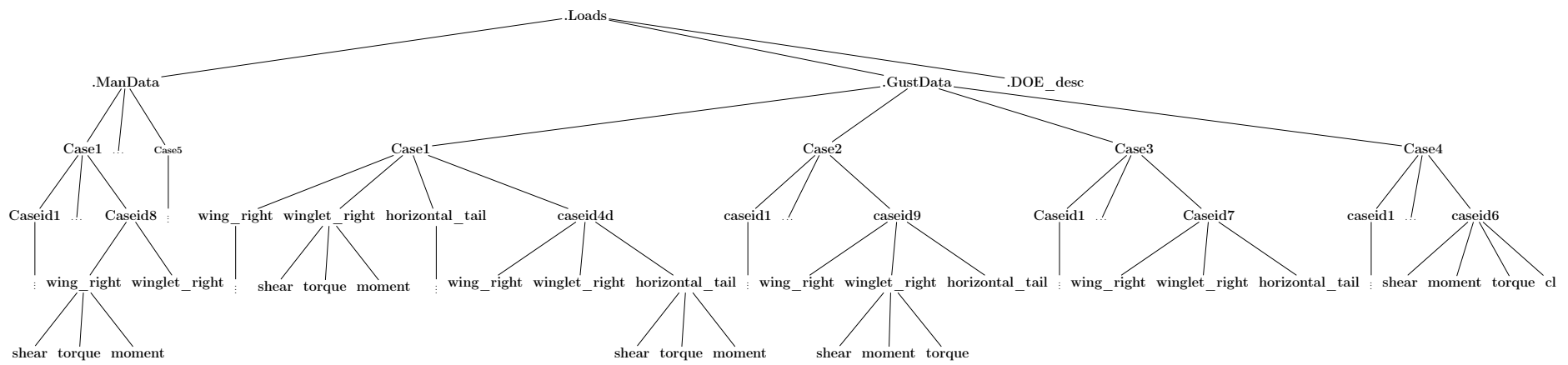


Figure 3.5: Tree overview of the dataset

As mentioned the structure of the dataset may be initially difficult to grasp, thus the robust optimisation problem formulation is not straightforward as expected. However, by following a simple eight step procedure the models for aerodynamic improvement and loads estimation may be formulated. This section does not attempt to describe the mathematics underlying such models, since they are presented in chapters 4, 5 and chapter 6.

Step 1 - Identify the Model Variables

The first step in the heuristics proposed is to identify the model variables or parameters. These are divided for convenience into optimisation parameters or control variables, uncertainty parameters and operational parameters. Operational parameters account for the changes in the mission or operational envelope, such as the different load cases correspondent to different flight conditions.

- (a) Identify the optimisation parameters: wing “Jig Shape” twist θ_1, θ_2 .
- (b) Identify the uncertainty parameters: $EI_1, EI_2, EI_3, EI_4, GJ_1, GJ_2, GJ_3$ and GJ_4 .
- (c) Identify the operational parameters: the design load case, λ .

Step 2 - Construct the Required Surrogate Models/Emulators

In order to formulate the necessary surrogates, it is convenient to identify the different disciplines involved in the problem. The different dataset shares DOE parameters which makes it possible to “couple” them. However, each model is associated with different DOEs. Although, these models share the same parameters, as required by the problem formulation, the outputs naturally are different.

- (a) Identify the disciplines: aeroperformance and design loads.
- (b) Identify the operational parameters: for a given C_L (e.g, $C_L = 1$), we have a certain aerodynamic efficiency $\Delta(L/D)$.

(c) Identify the number of training DOEs for each discipline:

Aeroperformance model DOEs: 1 – 50

Design loads model DOEs: 1 – 150

(d) Identify the model input variables:

Aeroperformance model input variables: $\theta_1, \theta_2, EI_1, EI_2, EI_3, EI_4, GJ_1, GJ_2, GJ_3$ and GJ_4 .

Design loads model input variables: $\theta_1, \theta_2, EI_1, EI_2, EI_3, EI_4, GJ_1, GJ_2, GJ_3$ and GJ_4 .

(e) Identify the model output variables:

Aeroperformance model: $\Delta(L/D)$

Design loads model: Shear force, moment and torque

Step 3 - Write the Functional Formulation of the Models

Based on the previous steps, it is possible to write in functional notation, the inputs and the outputs for the two models involved: the aeroperformance model that outputs the expected aerodynamic improvement $\Delta \frac{L}{D}$ and the shear, moment and torque loads across the wingspan. Since the SMT model is the union of the shear (S), the moment (M) and the torque (T) models, it is noted in bold to highlight it is vectorial output.

(a) Aeroperformance model:

$$\Delta(L/D) = M_{aero}(\theta_1, \theta_2, EI_1, EI_2, EI_3, EI_4, GJ_1, GJ_2, GJ_3, GJ_4) \quad (3.48)$$

(b) Design loads model:

$$\mathbf{SMT} = M_{loads}(\theta_1, \theta_2, EI_1, EI_2, EI_3, EI_4, GJ_1, GJ_2, GJ_3, GJ_4) \quad (3.49)$$

Step 4 - Assign Probability Distributions (uncertainties) to the Models

It is assumed that all the parameters are described by a normal or Gaussian distribution. This is because according to the central limit theorem any natural phenomena uncertain variable tends in the limit to the Gaussian distribution. As it is usual, the mean is chosen to be centred or null.

$$\theta_1 \sim \mathcal{N}(0, \sigma) \quad \mathbf{EI}_1 \sim \mathcal{N}(0, \sigma) \quad \mathbf{GJ}_1 \sim \mathcal{N}(0, \sigma) \quad (3.50)$$

$$\theta_2 \sim \mathcal{N}(0, \sigma) \quad \mathbf{EI}_2 \sim \mathcal{N}(0, \sigma) \quad \mathbf{GJ}_1 \sim \mathcal{N}(0, \sigma) \quad (3.51)$$

$$\mathbf{EI}_3 \sim \mathcal{N}(0, \sigma) \quad \mathbf{GJ}_1 \sim \mathcal{N}(0, \sigma) \quad (3.52)$$

$$\mathbf{EI}_4 \sim \mathcal{N}(0, \sigma) \quad \mathbf{GJ}_1 \sim \mathcal{N}(0, \sigma) \quad (3.53)$$

Step 5 - Define the Objective and the Constraint Functions

According to the Airbus delivery note, we have the following constraint for the objective function:

$$0.01 < \Delta(L/D) < 0.04 \quad (3.54)$$

Therefore, our objective function f is:

$$f : (\Delta L/D) \quad (3.55)$$

Step 6 - Define the Constraint Functions

The constraint functions are the SMT loads.

$$g : \mathbf{SMT} \quad (3.56)$$

SMT stands for Shear, Moment and Torque forces.

Step 7 - Apply/check the Design Margins to the Design Loads

It is interesting from an engineering point of the view that the design load variables are set in advance. The design load margin can be represented as a linear variation with wing span from a value of “ $x\%$ ” at the root to “ $y\%$ ” at the tip as figure 3.6 shows. These margins percentage on the design loads are applied to the “baseline” aircraft model and for initial studies the loads margin can be applied to the shear, moment and torque loads data

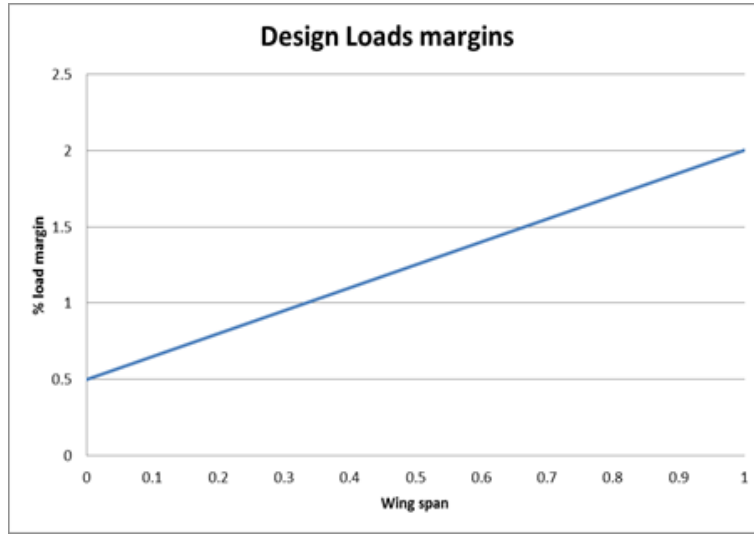


Figure 3.6: Design load margins

$$\beta = \frac{tip\% - root\%}{wing_stations} \quad (3.57)$$

Thus, for example, $root\% = 0.625\%$ and $tip\% = 2.5\%$.

$$\beta = \frac{2.5\% - 0.625\%}{62} = 0.030242 \quad (3.58)$$

In this thesis, the design load margin was converted into an algebraic parameter β (see equation 3.62) that is multiplied by the lower and upper bounds of the shear, moment and torque constraints, that are evaluated by the surrogate models. The derivation of this parameters is as follows.

Let r be the margin at the root in % , t the margin at the tip, s is the number of wingstations (62) and j a dummy index. Starting off from the fact that the SMT loads

are calculated at 62 locations and observing figure 3.6, the diagram in figure 3.7 is made. The subscript indicates the type of load and the wingstation where it is applied. Assuming that the margin linearly increases from the root to the tip, one arrives to equation 3.60.

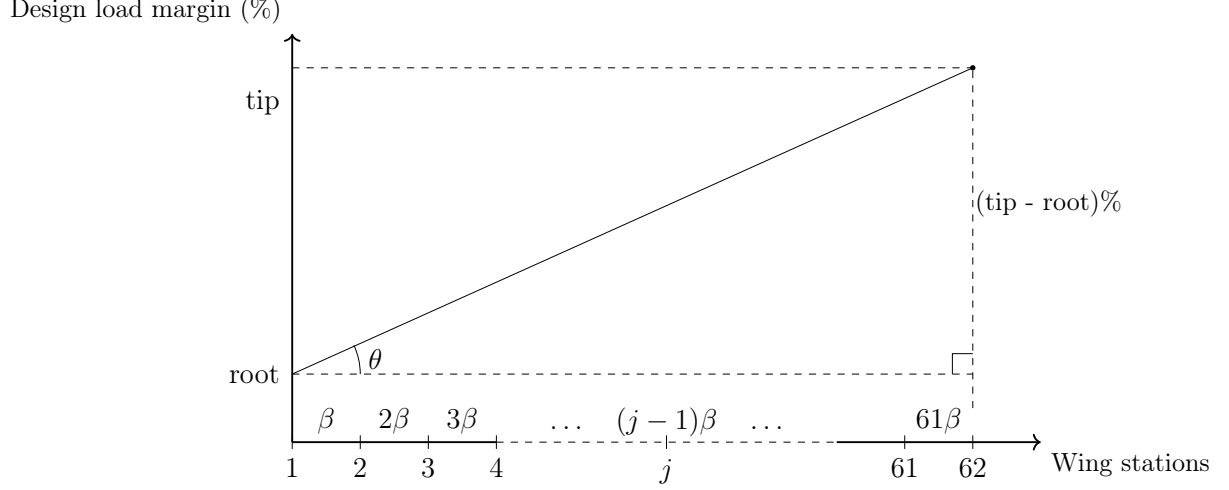


Figure 3.7: Design load margin versus wing stations

$$\begin{aligned}
 \beta_{\text{shear}_1} &= r_{\text{shear}} \\
 \beta_{\text{shear}_2} &= r_{\text{shear}} + \beta \\
 \beta_{\text{shear}_3} &= r_{\text{shear}} + 2 \cdot \beta \\
 \vdots &= \vdots
 \end{aligned} \tag{3.59}$$

$$\begin{aligned}
 \beta_{\text{shear}_{(j-1)}} &= r_{\text{shear}} + (j-1) \cdot \beta \\
 \vdots &= \vdots
 \end{aligned}$$

$$\beta_{\text{shear}_{62}} = r_{\text{shear}} + (62-1) \cdot \beta$$

$$\beta_{\text{shear}_{(j-1)}} = \beta_{\text{moment}_{(j-1)}} = \beta_{\text{torque}_{(j-1)}} = r + (j-1) \cdot \beta \tag{3.60}$$

A simple trigonometry yields the value of the beta parameter in terms of root and tip margin in percentage, as follows

$$\beta = \sin(\theta) = \frac{(t - r)}{s} \quad (3.61)$$

Since it is necessary to use this beta parameter in each wingstation, the above expressions can be conveniently written as

$$\begin{aligned} \left| \beta_{\text{shear}_{(j-1)}} \right| &= \left| \beta_{\text{moment}_{(j-1)}} \right| = \left| \beta_{\text{torque}_{(j-1)}} \right| = \left| r + (j - 1) \cdot \frac{(t - r)}{s} \right| \\ &= \left| \frac{r \cdot s + (j - 1) \cdot (t - r)}{s} \right| \\ &= \left| \frac{r \cdot s + j \cdot t + j \cdot r - t + r}{s} \right| \\ &= \left| \frac{r(s + j + 1) + t \cdot (j - 1)}{s} \right| \end{aligned} \quad (3.62)$$

Thus, its application in the lower and upper bound of the constraints used in the

SMT load models are

$$\begin{aligned}
 (1 - \beta_{\text{shear}_1}) \cdot S_1 &< S_1 < (1 + \beta_{\text{shear}_1}) \cdot S_1 \\
 &\vdots \\
 (1 - \beta_{\text{shear}_i}) \cdot S_i &< S_i < (1 + \beta_{\text{shear}_i}) \cdot S_i
 \end{aligned} \tag{3.63}$$

$$\begin{aligned}
 &\vdots \\
 (1 - \beta_{\text{shear}_{62}}) \cdot S_{62} &< S_{62} < (1 + \beta_{\text{shear}_{62}}) \cdot S_{62}
 \end{aligned}$$

$$\begin{aligned}
 (1 - \beta_{\text{moment}_1}) \cdot M_1 &< M_1 < (1 + \beta_{\text{moment}_1}) \cdot M_1 \\
 &\vdots \\
 (1 - \beta_{\text{moment}_i}) \cdot M_i &< M_i < (1 + \beta_{\text{moment}_i}) \cdot M_i
 \end{aligned} \tag{3.64}$$

$$\begin{aligned}
 &\vdots \\
 (1 - \beta_{\text{moment}_{62}}) \cdot M_{62} &< M_{62} < (1 + \beta_{\text{moment}_{62}}) \cdot M_{62}
 \end{aligned}$$

$$\begin{aligned}
 (1 - \beta_{\text{torque}_1}) \cdot T_1 &< T_1 < (1 + \beta_{\text{torque}_1}) \cdot T_1 \\
 &\vdots \\
 (1 - \beta_{\text{torque}_i}) \cdot T_i &< T_i < (1 + \beta_{\text{torque}_i}) \cdot T_i
 \end{aligned} \tag{3.65}$$

$$\begin{aligned}
 &\vdots \\
 (1 - \beta_{\text{torque}_{62}}) \cdot T_{62} &< T_{62} < (1 + \beta_{\text{torque}_{62}}) \cdot T_{62}
 \end{aligned}$$

Step 8 - Write the Complete Optimisation Formulation

The aeroperformance dataset provided by Airbus contains a 23×53 array describing the difference in L/D , that is $\Delta(L/D)$, between different DOE points and the baseline aircraft across different C_L values. It is important to notice that the $\Delta(L/D)$ and the C_L values are for the entire aircraft and not for the wingsections, see table 3.3. In this work, the optimisation was performed for the design cruise point C_L which equals to 1. Therefore, the objective function can be chosen as equation 3.67. For a specific C_L ,

there is a certain aerodynamic improvement, $\Delta(L/D)$, and a set of shear, moment and torque loads, these are given by two numerical models, respectively:

For the models at a given C_L as:

$$\begin{aligned}\Delta(L/D) &= M_{\text{aero}}(\bar{\theta}_1, \bar{\theta}_2, \overline{EI}_1, \overline{EI}_2, \overline{EI}_3, \overline{EI}_4, \overline{GJ}_1, \overline{GJ}_2, \overline{GJ}_3, \overline{GJ}_4, C_L) \\ \mathbf{SMT} &= M_{\text{loads}}(\bar{\theta}_1, \bar{\theta}_2, \overline{EI}_1, \overline{EI}_2, \overline{EI}_3, \overline{EI}_4, \overline{GJ}_1, \overline{GJ}_2, \overline{GJ}_3, \overline{GJ}_4, \lambda)\end{aligned}\tag{3.66}$$

Find

$$\mathbf{x}^* = \operatorname{argmax} f : \Delta \frac{L}{D}(C_L)\tag{3.67}$$

With respect to: $\bar{\theta}_1$ and $\bar{\theta}_2$

Subject to:

$$\begin{aligned}g_1 : [1 - \beta_{ij}^{\min}]^{\top} \cdot \mathbf{S} &< \mathbf{S} < [1 + \beta_{ij}^{\max}]^{\top} \cdot \mathbf{S} \\ g_2 : [1 - \beta_{ij}^{\min}]^{\top} \cdot \mathbf{M} &< \mathbf{M} < [1 + \beta_{ij}^{\max}]^{\top} \cdot \mathbf{M} \\ g_3 : [1 - \beta_{ij}^{\min}]^{\top} \cdot \mathbf{T} &< \mathbf{T} < [1 + \beta_{ij}^{\max}]^{\top} \cdot \mathbf{T}\end{aligned}\tag{3.68}$$

The steps that comprise the heuristic procedure are summarised in the following box.

Steps to Problem formulation

Step 1. Identify the Model Variables

Step 2. Construct the Required Surrogate Models/Emulators

Step 3. Write the Functional Formulation of the Models

Step 4. Assign Probability Distributions (uncertainties) to the Models

Step 5. Define the Objective and the Constraint Functions

Step 6. Define the Constraint Functions

Step 7. Apply/check the Design Margins to the Design Loads

Step 8. Write the Complete Optimisation Formulation

3.7 Summary and Conclusions

This chapter introduced the fundamental Bayesian concepts that underpins the probabilistic modelling under the Bayesian paradigm of an industrially relevant problem. Intuitively, Bayesian probabilistic modelling allows one to reason about a problem in terms of “degree of belief”, which is often contrasted to the frequentist approach, that takes each outcome as the realisation of a chance experiment. Thus, as it was shown in this chapter the Bayesian approach offers a more intuitive approach that can be applied recursively through the use of hierarchical Bayesian modelling.

A Bayesian probabilistic model is built iteratively by model comparison and checking different models using a variety of metrics presented in this chapter. After the model is synthesised, the next step is to compute the posterior. The posterior distribution is the probability distribution that corresponds to the degree of belief of the outcome yield by the probabilistic model. Often, it is important to generate predictions for new input points. In this case, the probability distribution is called predictive probability distribution. The computation of the posterior requires an understanding of Bayesian inference engines. In this chapter, the most important concepts and useful Bayesian inference engines Hamiltonian Monte Carlo and variational inference are briefly explained.

Finally, based on the concepts presented in the chapter, the robust optimisation problem is formulated. As discussed in the chapter, this is by no means at first a trivial problem, because the dataset structure tends to generate some level of confusion during the formulation. Nevertheless, the eight step heuristics presented allows one to convert a deterministic problem into a probabilistic one and is simple enough to be understood and adopted by the industry. Latter chapters discuss how the robust optimisation problem formulated in this section is investigated and solved numerically.

Chapter 4

Flight Loads Evaluation Using Gaussian Process Emulators

The aim of this chapter is to solve the probabilistic regression problem referred to in the mathematical problem formulation (section 3.6). The first section 4.1 presents the basic concepts related to the use of regression in engineering and reviews relevant literature. Section 4.2 introduces in detail the standard version of the main tool used in this work to build surrogate models - the Gaussian process. Later chapters present the sparse formulation which offers a scalable alternative to the standard Gaussian process formulation. Although, the main approach to the robust optimisation problem is the data-driven one, section 4.3 details how different loads and load cases are simulated in aeronautical engineering and the overall role it plays in engineering design. In section 4.4, the detailed construction of the robust emulators used to model the flight loads are validated and the results are discussed in the light of the Gaussian process theory. Finally, section 4.5, the main points of the chapter are recapitulated and the conclusions presented.

4.1 Introduction

Many machine learning problems that can be reduced to a regression require learning or training of models. This learning can be stated as a problem of learning the mapping itself of the space of the inputs, $\mathcal{X}$, to the space of outputs, $\mathcal{Y}$. In the Bayesian framework, this mapping is modelled by random variables, that represent the current degree of knowledge. The degree of knowledge is represented as a probability distribution. In this section the aim is to restate the concepts of the previous chapter in the context of Gaussian processes, which form the cornerstone of this work. Gaussian processes (GPs) represent a class of functional probability distributions that model the degree of belief on the prior and posterior over mathematical mappings [29]. In Bayesian data analysis, the procedure is always to start by defining a prior distribution that represents the prior belief and by Bayesian update the information contained in the observed data is taken into account using the Bayes' rule.

In this introductory section, the goal is to review the fundamental mathematical apparatus and concepts that are required for the manipulation of Gaussian processes priors and posteriors. The literature of Gaussian processes and machine learning in general is very vast and progressing very fast. Therefore, this introduction does not have the intent of being comprehensive. However, the main concepts to understand the latter sections are presented and, in this sense, this section is self-contained. Rasmussen and Williams [29] wrote a book called *Gaussian process for machine learning* that is the standard reference in Gaussian processes. The theoretical connections between Gaussian processes and stochastic processes are the basis for the work done in this thesis. However, to understand it completely it is necessary to use measure theory. Therefore, this section also has the goal of connecting Gaussian processes to stochastic processes without the mathematical machinery of measure theory, thus making it more accessible to engineers.

The starting point to understand Gaussian processes in the context of Bayesian inference is the mechanism of placing prior distributions on the mathematical mapping. As it is known, functions can be written as a sum of basis functions. Basis functions

are formed up by the sum of M different functions, ϕ , multiplied by a coefficient a_m . Mathematically,

$$f(x) = \sum_{m=1}^M a_m \phi_m(x) \quad (4.1)$$

The technical term “learning” in machine learning means that a method or algorithm is able to select a combination of weights or coefficients a_m , in such way that the behaviour of the original function or phenomenon of interest is completely modelled. In this case, one can say that the resulting function fits the data. The notation to be used is described now. The input space is denoted by $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^n$, where n is the number of dimensions. The outputs $\mathbf{y} \in \mathcal{Y} \subset \mathbb{R}^d$, where d is the number of output dimensions. These dimensions n and d not necessarily needs to be the same. The collection of the evaluations of the basis functions for the input $\mathbf{x}$ in a vector $\phi(\mathbf{x})$ and the inputs $\mathcal{X}$ are elements of the matrix $\Phi(\mathcal{X})$. The predictions at the points $\mathbf{x}^*$, resulting in the functional vector of evaluations $f(\mathcal{X}^*)$. The collection of N input vectors $\{\mathbf{x}_n\}_{n=1}^N$ are elements of a matrix $\mathbf{x} \in \mathbb{R}^{N \times d}$. The vector of observations or outcome are denoted by $f(\mathbf{y})$. The vector notation is used for y because computationally each probability distribution is a set of a number of samples that taken together form the probability distribution.

4.1.1 The Bayesian Prior

Assuming that functions can be represented in terms of basis functions, one can define a distribution over functions. This is done by placing a prior over the coefficients of equation 4.1. Thus, it is possible to understand the weights over the coefficients, $\{a_i\}$, or the basis function itself as a random variable. In this case, by the Bayes’s rule it is possible to compute its posterior. Assuming an independent Gaussian distribution for the coefficients of the basis functions, one can write mathematically,

$$p(\mathbf{a}) = \mathcal{N}(\mathbf{a}; 0, \sigma_a^2 I) \quad (4.2)$$

The properties of the prior distribution must match the properties of the prior

beliefs over the functions of interest. One can adjust theoretically, the properties of the functions by “calibrating” the basis functions of $p(\mathbf{w})$. The required manipulations involving random variables representing probability distributions over the basis functions coefficients require measure theory, as mentioned. This is because the random variables, in this case, are in the space of functions. Nevertheless, all the mathematical manipulations have the goal of giving a very rigorous guarantee that manipulating probability distributions on the basis function weights of coefficients is the same as manipulating these distributions over functions. Thus, the selection of a larger variance σ_a for the coefficients yields higher probability on functions with large values.

This approach has limits as expected. For example, the use of finite number of basis functions limits the types of functions that can be modelled. Thus, the choice of n finite basis functions, which can be specified arbitrarily, restrict the functions that can be used as the prior. Functions that cannot be prior also cannot be part of the posterior. Therefore, in such case, the mapping is not possible. Hence, it is important that there are mathematical guarantees which ensure that the prior is flexible enough to contain functions that are capable of representing the functions of engineering interest.

4.1.2 The Bayesian Posterior

As mentioned in the previous chapters, the Bayesian posterior is the result of Bayesian inference. Based on the discussion related to the prior, it follows that the posterior also contains weights that are adjusted after the method observes the data. A simple assumption that is very useful in engineering is to assume that the required function is affected by Gaussian noise. In such case, the likelihood for the function and its coefficients has the following form

$$(y_n \mid f(x_n)) = \mathcal{N}(y_n; f(x_n), \sigma^2) \quad (4.3)$$

$$p(y_n \mid \mathbf{w}, x_n) = \mathcal{N}\left(y_n; \sum_m w_m \phi_m(x_n), \sigma^2\right) \quad (4.4)$$

The Bayesian inference is executed by using the Bayes’ rule, which yields the

posterior, as follows

$$p(\mathbf{a} \mid \mathbf{y}) = \frac{\prod_n p(y_n \mid \mathbf{a}, x_n) p(\mathbf{a})}{p(\mathbf{y})} \quad (4.5)$$

To give a complete Bayesian solution to the Bayesian learning problem, it is necessary to find expressions for the properties of the prior and the likelihood as well. Thus, all the parameters that describe a particular model in this discussion may be included in the parameter θ , which is called hyper-parameter. These parameters are the variance of the functions σ_a , the noise σ and the number of basis functions n . The hyper-parameters themselves can be inferred using the Bayes' rule along with the marginal likelihood:

$$p(\theta \mid \mathbf{y}) = \frac{p(\mathbf{y} \mid \theta) p(\theta)}{p(\mathbf{y})} \quad (4.6)$$

In order to make predictions using the equation 4.6 above, it is necessary to integrate over both the hyper-parameter θ and the coefficients a . In practice, it is usual to adopt a single θ to carry on the necessary computations. According to Rasmussen and Williams [29], this is justified in the case of few hyper-parameters, where the θ posterior is dominated by the marginal likelihood $p(\mathbf{y} \mid \theta)$. In this case, it is possible to approximate the posterior $p(\theta \mid \mathbf{y})$ using a delta function, which also simplifies the prediction capability of the model, since the problem is reduced to a single integral over the Gaussian distribution, $\mathbf{w}$. Mathematically, this translates as follows

$$p(\theta \mid \mathbf{y}) \sim \delta(\theta - \hat{\theta}) \quad (4.7)$$

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} p(\mathbf{y} \mid \theta) \quad (4.8)$$

Equations 4.7 and 4.8 yields

$$\int p(y^* \mid \mathbf{a}) p(\mathbf{a} \mid \mathbf{y}, \theta) p(\theta \mid \mathbf{y}) d\theta d\mathbf{a} \quad (4.9)$$

that is,

$$\approx \int p(y^* \mid \mathbf{a}) p(\mathbf{a} \mid \mathbf{y}, \hat{\theta}) d\mathbf{a} \quad (4.10)$$

Functional Bayesian Inference

As discussed, the main insight for measure theory is that *a distribution over function basis coefficients imply a distribution over functions*. Thus, one can perform the Bayesian inference on the coefficients themselves and this translates to an inference over the functions. The main goal of Bayesian inference is the posterior for new prediction points. It is also possible to compute the marginal likelihood by considering a joint distribution of function evaluations. Starting with the relationship between $f(\mathbf{x})$ and $\mathbf{a}$, one can obtain the prior and the joint distribution with the points that form the likelihood. Mathematically, this is

$$f(\mathbf{x}) = \begin{bmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix} = \begin{bmatrix} f(\mathbf{a}^\top \phi(x_1)) \\ \vdots \\ \mathbf{a}^\top \phi(x_n) \end{bmatrix} = \Phi(\mathbf{x})\mathbf{a} \quad (4.11)$$

The joint distribution of the function points $\mathbf{x}$ is computed by an integration procedure:

$$p(f(\mathbf{x})) = \int p(f(\mathbf{x}) | \mathbf{a})p(\mathbf{a})d\mathbf{a} = \mathcal{N}(f(\mathbf{x}), \boldsymbol{\mu}, k_{ff}) \quad (4.12)$$

The mean and covariance of the probability distribution in equation 4.12 are given by:

$$\boldsymbol{\mu} = \Phi(\mathbf{x})\mathbb{E}_{\mathbf{a}}[\mathbf{a}] = 0 \quad (4.13)$$

$$k_{\text{ff}} = \text{cov}_{\mathbf{a}}[\Phi(X)\mathbf{a}] = \Phi(X)\mathbb{E}_{\mathbf{a}}[\mathbf{a}\mathbf{a}^\top]\Phi(X)^\top \quad (4.14)$$

$$= \sigma_a^2 \Phi(X)\Phi(X)^\top \quad (4.15)$$

The matrix created out of the inner product basis pairs is designated by $\Phi(X)\Phi(X)^\top$ and the vectors of evaluation are $\phi(\mathbf{x})$. The covariance function, k , must always be positive-definite and thus it is a kernel function [226]. The kernel function [29] is

written as

$$k(\mathbf{x}_i, \mathbf{x}_j) = [\Phi(X)\Phi(X)^\top]_{ij} = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j) = \sum_{b=1}^B \phi_b(\mathbf{x}_i) \phi_b(\mathbf{x}_j) \quad (4.16)$$

Posterior Predictive Distribution

The posterior predictive distribution can be computed for a new point by considering the the joint prior for function values of the inputs observed, since the prior used is mathematically equivalent to the prior that refers to the coefficients of the basis function. Thus, one have

$$p(f(\mathbf{x}), f(\mathbf{x}^*) | \mathbf{y}) = \frac{p(\mathbf{y} | f(\mathbf{x}))p(f(\mathbf{x}), f(\mathbf{x}^*))}{p(\mathbf{y})} = p(f(\mathbf{x}) | \mathbf{y})p(f(\mathbf{x}^*) | f(\mathbf{x})) \quad (4.17)$$

4.2 Gaussian Processes

Having introduced the basic insights that allow Bayesian inference to be applied to functional representations, in this section, the focus is on Gaussian process (GP). GP is a collection of random variables in which the joint distribution of any finite subset of its variables is a Gaussian distribution. Essentially, GPs are Gaussian distribution with infinite dimensions. In this sense, GPs can be seen as a generalisation of Gaussian distributions with infinite dimensions. Gaussian processes have interesting analytical properties derived from the properties of the Gaussian distribution, such as having closed-form solutions to the common operations of summation, conditioning and marginalisation [29]. Let X be a dataset where each feature has a mean μ and covariance Σ , one can write the marginals as following

$$X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}, \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix} \quad (4.18)$$

$$p(X_1) = \mathcal{N}(X_1 \mid \mu_1, \Sigma_{11}) \quad (4.19)$$

$$p(X_2) = \mathcal{N}(X_2 \mid \mu_2, \Sigma_{22}) \quad (4.20)$$

the conditionals are defined as

$$p(X_1 \mid X_2) = \mathcal{N}(X_1 \mid \mu_{1|2}, \Sigma_{1|2}) \quad (4.21)$$

$$\mu_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2) \quad (4.22)$$

$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \quad (4.23)$$

An advantage that Gaussian processes have over deep neural networks is the fact that a GP is non-parametric by definition. GPs work by marginalising the entire function space in order to obtain its predictions. This also has the benefit that one does not need to make decisions about the Gaussian process inner working regarding the model complexity. The marginalisation process also has an analogous effect to the Occam's razor, thus preventing overfitting. The Gaussian property of Maximum Entropy also applies to the Gaussian process given a dataset mean and variance. Therefore, the GP can model distributions in the most informative way. Gaussian processes also have an interesting property in regards with its relation with Perceptrons. Perceptrons are often deemed to be universal approximators to any mathematical function, given a sufficient amount of data [190]. In regards to accuracy performance, the Central Limit Theorem guarantees that any dataset's mean follows a Gaussian distribution as the number of samples from the distribution increases.

Gaussian processes may also be defined as a probabilistic mapping from an input space to an output space. Formally, one can say that a function f maps an input space $\mathbf{x}$ to an output space $\mathbf{y}$, along with an uncertain or Gaussian noise term ϵ .

$$\mathbf{y} = f(\mathbf{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \beta^{-1}\mathbf{I}) \quad (4.24)$$

The fact that GPs assume a distribution over the functions f can be contrasted with

the Maximum A Posteriori (MAP) is used to estimate the function parameters in a non-probabilistic approach, where f has a parametric form instead. In Gaussian processes, the function values are called latent variables. The GP predictions marginalise over the entire functions space, where the distribution over the functions is parameterised by the mean and covariance matrix. The covariance is computed as a kernel function k . The relevant mathematical equations are as follows

$$\mathbf{f} \mid \mathbf{x} \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x})) \quad (4.25)$$

$$\mu(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \quad (4.26)$$

$$k(\mathbf{x}, \mathbf{x}) = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x})) (f(\mathbf{x}) - \mu(\mathbf{x}))] \quad (4.27)$$

The distribution $\mathbf{y}$ conditioned on the input $\mathbf{x}$ is computed as

$$p(y \mid f) = N(f, \beta^{-1}I) \quad (4.28)$$

$$p(y \mid x) = \int p(y \mid f)p(f \mid x)df \quad (4.29)$$

The integral equation above, a marginalisation corresponds to the likelihood of the dataset. The procedure of marginalisation is used to fit the hyper-parameters involved in the Gaussian procedure formulation by maximising the training set likelihood. The goal of Gaussian process in engineering is not only to replicate the data contained in the training dataset, but to produce new predictions for new data inputs, while considering a joint distribution over the function input values of x . These new input values and its predictions are denoted by $*$.

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N} \left(\mathbf{0}, \begin{bmatrix} k(\mathbf{x}, \mathbf{x}) & k(\mathbf{x}, \mathbf{x}_*) \\ k(\mathbf{x}_*, \mathbf{x}) & k(\mathbf{x}_*, \mathbf{x}_*) \end{bmatrix} \right) \quad (4.30)$$

The output, that is, the posterior distribution of $\mathbf{f}^{**}$ is obtained by conditioning $\mathbf{x}^*$, $\mathbf{x}$

and $\mathbf{y}$ as follows

$$p(\mathbf{f}_* | \mathbf{x}_*, \mathbf{x}, \mathbf{y}) = \frac{1}{p(\mathbf{y})} \int p(\mathbf{y} | \mathbf{f}) p(\mathbf{f}, \mathbf{f}_*) d\mathbf{f} \quad (4.31)$$

$$p(\mathbf{f}_* | \mathbf{x}_*, \mathbf{x}, \mathbf{y}) \sim \mathcal{N}(k(\mathbf{x}_*, \mathbf{x}) k(\mathbf{x}, \mathbf{x})^{-1} \mathbf{f} \quad (4.32)$$

$$k(\mathbf{x}_*, \mathbf{x}_*) - k(\mathbf{x}_*, \mathbf{x}) k(\mathbf{x}, \mathbf{x})^{-1} k(\mathbf{x}, \mathbf{x}_*) \quad (4.33)$$

The term that represents the mean of the predictive distribution represents the weighted average of the training set output. The kernel matrices assign higher weights to data points close to the test samples in feature space. The variance has two components. The first one equates to the prior variance from the testing data. The second component is the variance induced by the training data. The subtraction of these two terms yields the initial test data variance reduced by the training data's information.

4.2.1 Kernel Function

A Gaussian process has theoretically infinite dimensions. This fact makes infeasible to define a covariance matrix for a Gaussian process, as it is commonly done for conventional Gaussian distributions. For this reason, GPs rely on mathematical objects called kernel functions to define a special covariance matrix that allows a compact representation of the kernel matrix.

Kernel functions are defined mathematically as dot products between data points in a high-dimensional manifold without mapping these data points to a manifold explicitly. The Gaussian process takes a kernel function and enforces as a property of positive semi-definiteness, transforming the Gaussian process kernel function into a positive semi-definite matrix. In this way, Gaussian processes are able to effectively measure data similarity while providing useful metrics for its fitness efficiency.

There are many kernel functions with different properties and formulations in the kernel literature. It is relatively easy to define new kernel functions that suits a particular task. The standard reference [29] in the Gaussian literature contains a detailed presentation on different kernels. In this section, the squared exponential

kernel and the automatic relevance detection (ARD) kernel functions are covered with the purpose of presenting more details on how Gaussian processes work.

The squared exponential kernel [29] is defined by two parameters: the length scale l , and the variance scale σ . In practical terms, the length scale determines the minimum distance between a pair of (high-dimensional) data points needed in order to be correlated. The variance scale parameter models the variance induced from the data points. The exponential form of this kernel allows some computational advantages in terms of speed [29]. Mathematically, this kernel is defined as follows, for a pair of inputs $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp \left(-\frac{1}{2l^2} \sum_{j=1}^n (x_j - x'_j)^2 \right) \quad (4.34)$$

The automatic relevance detection (ARD) kernel has setting parameters for each dimension of the n -dimensional input space. This kernel resembles a weighted kernel. Mathematically, the automatic relevance detection kernel is defined for a pair $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$ as

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp \left(-\frac{1}{2} \sum_{j=1}^n w_j (x_j - x'_j)^2 \right) \quad (4.35)$$

These weight parameters may be defined manually and are used to scale down specific features or increase the relevance of the ones deemed important in a given task or application. However, in practice, this approach is not recommended, since usually it is not known in advance what are the really important features in the training dataset. Fortunately, there are two principled approaches to select the function parameters. These are the *maximum a posteriori* (MAP) and the Bayesian approach.

The MAP approach works on the log probability of the $\log p(\mathbf{Y} / \mathbf{X})$ conditional term associated to a dataset. By computing the log probability derivative with respect to the kernel parameters, the optimal parameters are obtained. For bigger datasets, it is necessary to employ gradient descent to calculate the parameters. In mathematical

terms,

$$\log p(\mathbf{f} \mid \mathbf{X}) = -\frac{1}{2}\mathbf{f}^\top K^{-1}\mathbf{f} - \frac{1}{2}\log |K| - \frac{n}{2}\log 2\pi \quad (4.36)$$

$$\log p(\mathbf{y} \mid \mathbf{X}) = -\frac{1}{2}\mathbf{y}^\top (K + \sigma_n^2 I)^{-1}\mathbf{y} - \frac{1}{2}\log |K + \sigma_n^2 I| - \frac{n}{2}\log 2\pi \quad (4.37)$$

A common limitation of MAP is that this approach may lead to suboptimal conditioning, depending on the dataset size and the kernel function employed. This issue may be mitigated by the Bayesian approach. The Bayesian approach consists of defining a prior distribution over the kernel parameters to compute the posterior parameter distribution conditioned on the training dataset. The Bayesian approach is often used in conjunction with sampling methods to determine the parameter distributions. However, for many problems, these sampling methods have limitations and expensive computational costs.

4.2.2 Limitation of Standard GPs

The “standard” Gaussian process formulation without any additional feature has a few limitations. In general, these limitations are enumerated as: 1) its computational cost, 2) storage cost and 3) Hierarchical feature extraction. The computational cost of a Gaussian process originates in the process of inverting the kernel matrix that is necessary to infer the predictive distribution. Specifically, the computational cost of intervening a kernel matrix of size $n \times n$, where n is the number of points that the training dataset contains takes $O(n^3)$ computation time. The predictive distribution’s mean and variance of a new point takes $O(n)$ and $O(n^2)$, respectively.

An important technical consideration to make is related to the computational storage cost of Gaussian Processes. For a Gaussian process the storage cost is $O(n^2)$. Depending on the nature of the problem to be modelled, the required size of the entire training dataset’s storage will be prohibitive, limiting the scalability of the method. This is certainly the case for the current trend of digital-twinning [172], where training set is supposed to increase as a function of the system’s operating time.

The standard formulation of a Gaussian Process lacks in the kernel functions the capability of hierarchical feature extraction to calculate the similitude of data point

pairs. This feature is necessary when one is interested in taking into account the dependence in the input parameters, for instance. In such cases, deep learning seems to a promising machine learning method, since DL models contain feature extraction features embedded in its mathematical formulation. However, this feature must still work in the Bayesian paradigm, in order to preserve GP's advantages.

4.2.3 Gaussian Process Requirements

It is important to define some requirements for Gaussian process engineering approximations. The Gaussian process and its alternative formulations must retain the advantages of the Bayesian paradigm, which was discussed in chapter 3, along with the non parametric GPs. The advantages of Bayesian methods may be kept by separating the contributions of the prior and the approximation utilised by the Gaussian process. In this case, one can use marginal likelihoods for model comparison to develop new models. Although this operation is embedded in the computational implementation of the Gaussian process. GP are interesting to engineering applications, if they can satisfy the following requirements.

- It is necessary that they are capable of providing a distance to the exact solution.
- They must be capable under infinite computational resources to recover the best as possible the phenomenon to which it is intended to be modelled.
- Provide error bars that shows the quantified uncertainty.

4.3 Flight Physics

In this section, the basic concepts of flight physics are reviewed with the goal of contextualise the robust optimisation problem in aeronautical engineering. Flight physics is an umbrella term used in the industry to denote the primary disciplines and technologies concerned with aeronautics, i.e, aerodynamics, propulsion, performance, stability and control and its applications to aircraft flight. For this reason, the equations

of flight motion are presented without derivation and based on it the topic of flight loads is introduced. On it, manoeuvre and gust loads are discussed.

The tasks of design and development of an aircraft require multidisciplinary analyses. The design process needs a knowledge of the loads and aeroelastic response since these are the main drivers of the sizing process during all development phases. The external and internal loads have a strong impact upon structural design, aerodynamic characteristics, flight control system and control surface design, weight and performance. In this regard, the development of methods that provide fast, but accurate, prediction of loads will enable a quicker determination of the critical loads cases and also permit more design configurations to be considered.

Of particular interest is the identification of those flight loads that lead to the most extreme stress levels for a specific design. This aim is achieved by applying a large number of loads cases due to dynamic gusts and manoeuvres to detailed aeroelastic models for a range of different Interesting Quantities (IQs) e.g. bending moments, shear forces, etc. These calculations are very complex and requires a tool-set of coupled finite element method (FEM) and computational fluid dynamics (CFD). This coupled simulation however has its limits. The most obvious is the large number of conditions for which the aircraft must studied. An estimation of the number of conditions for a typical large civil aircraft is shown in table 4.1. One possible solution to the large

Parameters	Number of Points
Flight points (altitude and speed)	50
Mass cases (loaded weight and weight distribution)	100
Control surface configurations	10
Manoeuvres and gusts	50
Control laws	4
Total number of cases required	10^8

Table 4.1: Estimate of number of load cases required [227]

number of simulations required may be the use of simpler physical models. However, for large aircraft manufacturers this is not an option, since these companies need to exploit the design space as much as they can in order to stay competitive in the markets. The flight loads calculation is so critical to the design and the safety of

an aircraft that even a conventional design demands approximately 100 thousand hours of simulations [227]. It is important to note that despite all the sophisticated computational engineering tools employed in a large civil aircraft programme expert judgement is still a big part of the critical load identification process. Furthermore, these analyses have to be repeated every time that there is an update in the aircraft structure. Within the modern civil airframe industry, each of these loads calculation cycles takes a considerable time. Previous work investigated the use of several surrogate models and optimisation methods for fast and efficient prediction of the worst case gust loads for each IQ [227]. It has been shown that savings in computational time can be made without sacrificing accuracy.

4.3.1 Equations of Motion During Flight

The equations of motion for an aircraft may be written in the general vector form as follows:

$$\frac{d\mathbf{x}(t)}{dt} \triangleq \dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), \mathbf{w}(t), t] \quad (4.38)$$

where $\mathbf{f}$ is the n -dimensional vector function that corresponds to the variables of motion $\mathbf{x}$, controls $\mathbf{u}$, disturbances $\mathbf{w}$, and time t on the state vector's rate of change $d\mathbf{x}/dt$. Assuming flat-earth and symmetric-aircraft the following vectorial function of equation 4.38 represents the 12 equations of motion:

$$\dot{u} = \frac{X}{m} + g_x + r \cdot v - q \cdot w \quad (4.39)$$

$$\dot{v} = \frac{Y}{m} + g_y - r \cdot u + p \cdot w \quad (4.40)$$

$$\dot{w} = \frac{Z}{m} + g_z + q \cdot u - p \cdot v \quad (4.41)$$

$$\dot{\phi} = p + (q \sin \phi + r \cos \phi) \tan \theta \quad (4.42)$$

$$\dot{p} = \frac{(I_{zz} \cdot L + I_{xx} \cdot N - \{I_{xz}(I_{yy} - I_{xx} - I_{zz})p + [I_{xx}^2 + I_{zz}(I_{zz} - I_{yy})]r\}q)}{(I_{xx}I_{zz} - I_{xz}^2)} \quad (4.43)$$

$$\dot{q} = \frac{[M - (I_{xx} - I_{zz}p\dot{r} - I_{xz}(p^2 - r^2))]}{I_{yy}} \quad (4.44)$$

$$\dot{r} = \frac{(I_{xz} \cdot L + I_{xx} \cdot N + \{I_{xz}(I_{yy} - I_{xx} - I_{zz})r + [I_{xz}^2 + I_{zz}(I_{zz} - I_{yy})]r\}q)}{I_{xx}I_{zz} - I_{xz}^2} \quad (4.45)$$

$$\dot{x}_I = (\cos \theta \cos \psi)u + (-\cos \phi \sin \psi + \sin \phi \cos \psi)v + (\sin \phi \sin \psi + \cos \phi \sin \theta \cos \psi)w \quad (4.46)$$

$$\dot{y}_I = (\cos \theta \cos \psi)u + (-\cos \phi \sin \psi + \sin \psi \sin \theta \cos \psi)v + (-\sin \psi \cos \psi + \cos \psi \sin \theta \sin \psi)w \quad (4.47)$$

$$\dot{z}_I = (-\sin \theta)u + (\sin \phi \cos \theta)v + (\cos \phi \cos \theta)w \quad (4.48)$$

$$\dot{\phi} = p + (q \sin \phi + r \cos \phi) \tan \theta \quad (4.49)$$

$$\dot{\theta} = q \cdot \cos \phi - r \cdot \sin \phi \quad (4.50)$$

$$\dot{\psi} = (q \cdot \sin \phi + r \cdot \cos \phi) \sec \theta \quad (4.51)$$

In the above equations, the components of the gravity force g are related as follows

$$\begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} = \begin{bmatrix} -g \sin \theta \\ g \sin \phi \cos \theta \\ g \cos \phi \cos \theta \end{bmatrix} \quad (4.52)$$

The applied forces (X, Y, Z) and moments (L, M, N) are functions of airspeed and aerodynamic angles, they are mathematically described by the following equations

$$\begin{bmatrix} V_A \\ \beta_A \\ \alpha_A \end{bmatrix} = \begin{bmatrix} \sqrt{u_A^2 + v_A^2 + w_A^2} \\ \sin^{-1}(\frac{v_A}{V_A}) \\ \tan^{-1}(\frac{w_A}{u_A}) \end{bmatrix} \quad (4.53)$$

The velocity components for a body axis relative to the air must account for the wind field

$$\begin{bmatrix} u_a \\ v_a \\ w_a \end{bmatrix} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} - H_1^B w \quad (4.54)$$

$$X = C_X \bar{q} S + T_x \quad (4.55)$$

$$Y = C_Y \bar{q} S + T_y \quad (4.56)$$

$$Z = C_Z \bar{q} S + T_z \quad (4.57)$$

$$L = C_l \bar{q} S b + M_{T_x} \quad (4.58)$$

$$M = C_m \bar{q} S \bar{c} + M_{T_y} \quad (4.59)$$

$$N = C_n \bar{q} S b + M_{T_z} \quad (4.60)$$

Remembering that the dynamics pressure is given by

$$\bar{q} = \frac{\rho(z) V_A^2}{2} \quad (4.61)$$

The air density $\rho(z)$ is a function of altitude and the Mach number is given by

$$\mathcal{M} = \frac{V_A}{a(z)} \quad (4.62)$$

Where the speed of sound $a(z)$ is a function of altitude. The previous equations apply to all aircraft. The models of each type of aircraft characterise different mathematical expressions for inertial, aerodynamic, and thrust effects.

4.3.2 Flight Loads

Having introduced the importance of flight load evaluation in an aircraft development program and the flight dynamics model, in this section the goal is to further detail the concept of flight loads.

Flight loads analysis is an important engineering activity. It has a decisive role from the initial conceptual design until decommissioning. Usually, in the conceptual design the common practice is to use analytical methods or statistical ones to rapidly evaluate the potential loads that drives the design. These tools often depend on parametric studies [228] and although they provide fast evaluations they do not capture the underlying physics. Thus, in some cases, they are not accurate as they need to be. Often this is translated to additional costs and rework in case it is latter detected that the structure was under or oversized.

A correct load estimation in aircraft design results an appropriate structural design where its weight is minimised. There are few papers or books available that details methods for load estimation and most of the OEM produces and documents its own methods. Niu [229] offers a good presentation of fundamental load analysis methods and how loads interface with different departments in an aircraft design project. Also, Lomax [230] presents methods for load estimation that are based on regulations such as FAR part 25 and CS-25.

To understand how the loads affect the aircraft development programme on a high-level perspective, without entering the realm of structural design and aeroelasticity, one can take a basic flight physics approach. As it is known, an aircraft performing a turn is subject to a higher load than in a straight cruise flight. Other types of manoeuvres such as during an aborted landing or during an emergency may subject the aircraft to an even higher load

Higher loads may be also the result of turbulence that affect the aircraft in the form of horizontal or vertical gusts. The parameter used in flight physics to quantify the general load that the aircraft is subject is called *load factor* [228]. Its mathematical expression is simply

$$n = L/W \tag{4.63}$$

Even in free flight, the load factor depends on the aircraft configuration (for example, wing flap and landing gear position) and the distribution of useful load such as fuel, cargo, equipment and passengers.

Maneuver Loads

The effect of maneuver loads on an aircraft may be quantified by a diagram called the load factor diagram. This diagram consists of several segments that correspond to a specific phase in the aircraft flight envelope. Thus, one can say that the load factor forms up the loading enveloped for which an aeronautical structure must be designed.

The manoeuvre loads are the result of a change in the condition of the level flight, usually in the form of an increase in the wing incidence angle relative to the horizontal cruise flight vector, where the condition is normally

$$W = L = C_L \frac{1}{2} \rho V^2 \quad (4.64)$$

If the aircraft lift coefficient, C_L , increases by ΔC_L during a symmetric manoeuvre at a constant speed, the lift is given by

$$L' = (C_L + \Delta C_L) \frac{1}{2} \rho V^2 S \quad (4.65)$$

Thus, the load factor increases to

$$n = \frac{L'}{W} = 1 + \frac{\Delta C_L}{C_L} \quad (4.66)$$

Inspecting the equation 4.38 and its components, it is concluded that in a motion normal to the flight path, considering $\alpha_T = 0$, the aircraft is subject to a vertical force that has a magnitude of

$$\frac{W}{g} V \frac{d\gamma}{dt} = \frac{W}{g} a_n = L' - W \quad (4.67)$$

This motion yields a vertical acceleration that is normal to the flight path

$$\frac{a_n}{g} = \frac{L'}{W} - 1 = \frac{\Delta C_L}{C_L} = n - 1 \quad (4.68)$$

Thus the aircraft flies a curved path in which its airspeed tends to decrease against an increased flight path angle and increased drag. If this condition is too extreme, the

aircraft stalls. In this case, the load factor, n is

$$n_{\max} = \frac{L_{\max}}{W} = \frac{C_{L_{\max}} \frac{1}{2} \rho V^2 S}{W} = \frac{C_{L_{\max}} \rho_{\text{sl}} V_{\text{eq}}^2}{2W/S} \quad (4.69)$$

Gust Loads

The same flight physics basic analysis can be done for the gust loads. An aircraft in steady flight may fly through a thunderstorm and experience a strong turbulence. In a turbulence, the aircraft is normally subject to upward and downward currents of air. These currents are known as gusts. Gusts can cause sudden changes in the speed, in the lift generated by the wings and by the tail and change the angle of attack of the aircraft, giving rise to fluctuating loads on the aircraft. In addition to that, it is known that gusts affect the lifespan of the aircraft structure as this is limited by damage due to structural fatigue.

Gusts are a stochastic phenomena. However, during the initial steps of sizing gusts may be modelled with a very simple analytical model that assumes a sharp-edged vertical gust with speed U that increases the angle of attack by $\Delta\alpha \approx (U/V)$, without changing the airspeed. In this case, the lift coefficient is increased by

$$\Delta C_L = \frac{dC_L}{d\alpha} \Delta\alpha = \frac{dC_L}{d\alpha} \frac{U}{V} \quad (4.70)$$

In this case, the load factor [228] is

$$n = 1 + \frac{dC_L}{d\alpha} \frac{\rho UV}{2W/S} \quad (4.71)$$

A more realistic model for the gust is not sharp-edged, but one where the aircraft moves with the displacement of the air mass, relieving the gust load.

The correction factor K_g is used to quantify this effect and its order of magnitude is between 0.7 and 0.9, depending on the aircraft [228]. The speed in U and V in the equations 4.54 and 4.39 are written as an EAS, yielding in a simple flight dynamics

model the follow load factor

$$n = 1 + K_g \frac{dC_L}{d\alpha} \frac{\rho_{sl} U_e V_{eq}}{2W/S} \quad (4.72)$$

The component u_e is positive for an upward gust and negative for a downward gust. Equation 4.72 shows that the load factor increases, for a given speed u_e as the wing loading is lower and the EAS is higher. Thus, one can conclude that in order to reduce the loads on the structure and improving the level of flight comfort, it is undesirable to fly with a low wing loading at low altitude and high speed. Normally, the aircraft is more affected by gusts when it has a high-lift gradient $dC_L/d\alpha$.

4.4 Results and Discussion

As it was discussed in previous sections, the knowledge of flight physics, aeroelastic modelling and flight loads are an important phase of aircraft structural sizing. The forces applied to the aircraft during flight (equations 4.55 to 4.60). These forces when applied to the aircraft structure during flight results in internal stresses, that must be within an allowable value for safe and reliably operation. Thus, the structure must be designed considering such stresses. These stresses however are a function of the external forces acting on the aircraft, which are a function of a specific design load case and aircraft geometry.

Since the aircraft structure is flexible, such forces are also a function of the aircraft structure. This “paradox” is a problem for aeronautical engineers, because to know the sizing allowable for the structure to be designed, it is necessary to know the maximum stresses and other parameters. But, in order to known these parameters, the structure must be sized and defined. Since maximum performance is sought, recently the insight that it is possible to take this structural uncertainty into account through the use of random variables was adopted by the industry.

The application of random variables in aeroelastic computer simulations it is a very challenging task since it requires huge computational effort and parametrisation of the

aircraft structure. Thus, there is an opportunity to employ surrogate models that are able to replicate the results from physical simulations at a much cheaper computational cost.

4.4.1 Problem Description

To solve this problem, it is first necessary to formalise the engineering use case of load estimation. The external aircraft flight loads can be considered a function of a given aircraft design and a number of parameters defining a load case. A load case can be interpreted as a set of parameters defining a particular aerodynamic manoeuvre or a gust, which results in a force being applied to the airframe.

A computational model for flight loads can be abstracted as a black box which accepts two vectors, $\mathbf{x}$ and λ , describing a given aircraft design and load case, respectively, and returns a vector, $\mathbf{y}$, of corresponding forces and moments at locations of interest on the airframe, which it is referred as interesting quantities (IQs).

$$\mathbf{y} = h(\mathbf{x}, \lambda) \quad (4.73)$$

Both $\mathbf{x}$ and λ are constrained to take values from permissible sets, $\mathcal{X}$ and Λ . These sets contain all feasible combinations of individual design and load case parameters, respectively.

The bending and torsional stiffness parameters are variables that are “abstracted” from the complete aircraft definition and used as proxies for the entire aircraft structure at a given point of the project. Chapter 1 presents the two scenarios that occur in aircraft programmes. The method used here can be applied to both scenarios, but it is intended to be a representative case of the use in the scenario where there is an opportunity to perform a “late optimisation”. That is when the structure is already defined, but it is possible to set the fixing tooling in a way that will give a small increment in the expected aerodynamic performance. Thus, the objective function when evaluated outputs the expected aerodynamic improvement, $\Delta(L/D)$. Thus, the

stiffness parameters used represent a small percentage deviation from the baseline which is referred as design of experiment (DOE) number 0. In practice, it is not the external forces, $\mathbf{y}$, but internal forces that are responsible for constraining the design of the structure. The process by which external forces $\mathbf{y}$ are mapped to internal force $\mathbf{t}$ may also be considered as a black box $t = g(\mathbf{y})$. Thus, this problem is translated to a problem of training and evaluating the loads using Gaussian process, which were described in the previous sections of this chapter. For the models at a given C_L as:

$$\mathbf{SMT} = \mathcal{GP}_{\text{loads}}(\bar{\theta}_1, \bar{\theta}_2, \bar{EI}_1, \bar{EI}_2, \bar{EI}_3, \bar{EI}_4, \bar{GJ}_1, \bar{GJ}_2, \bar{GJ}_3, \bar{GJ}_4, \lambda) \quad (4.74)$$

$$\mathbf{SMT} = \mathbf{S} \cup \mathbf{M} \cup \mathbf{T} \quad (4.75)$$

$$\text{and } \mathcal{GP}_{\text{loads}} = \mathcal{GP}_{\text{shear}} \cup \mathcal{GP}_{\text{moment}} \cup \mathcal{GP}_{\text{torque}} \quad (4.76)$$

In this section, the random variable, i.e, a variable that is a probability distribution defined by a mean μ and σ is denoted by a bar ($\bar{\cdot}$) Equation 5.24 shows that the $\mathbf{SMT}$ model is in fact the union of different Gaussian processes model, which are themselves also a union of an specific Gaussian process that models the load applied to one of the 62 wingstations that parametrise the wing. The input parameters are described the in table 4.2.

Input variables	Description
$\bar{\theta}_1$	twist angle at wing twist jig-shape station 1
$\bar{\theta}_2$	twist angle at wing twist jig-shape station 2
$\bar{EI}_1$	bending stiffness at position 1
$\bar{EI}_2$	bending stiffness at position 2
$\bar{EI}_3$	bending stiffness at position 3
$\bar{EI}_4$	bending stiffness at position 4
$\bar{GJ}_1$	torsional stiffness at position 1
$\bar{GJ}_2$	torsional stiffness at position 2
$\bar{GJ}_3$	torsional stiffness at position 3
$\bar{GJ}_4$	torsional stiffness at position 4
λ	load case variable

Table 4.2: Input parameters of the surrogate model

4.4.2 Design of Experiments

The dataset provided by Airbus was generated using a optimised latin hypercube square (OLHS) design to sample the input space in a coupled FEM-CFD simulation using internal proprietary tools of Airbus. However, it is convenient to present this step as part of the analysis.

One may assume that $\mathcal{S} \triangleq \{\mathbf{x}_1, \dots, \mathbf{x}_s\}$ is a collection of s sample points generated via an OLHS. The function $f_i(\mathbf{x})$ may be evaluated for the set of sample points $\mathcal{S}$ that corresponds to each of the k_c load case functions. Since f_i returns a vector, $\mathbf{t}_i$. For a given $\mathbf{x}$, a matrix of responses $\mathbf{T}_i$ is obtained for each of the k_c load case functions, as follows:

$$\mathbf{T}_i \triangleq \begin{bmatrix} \mathbf{t}_{i,1} \\ \vdots \\ \mathbf{t}_{i,s} \end{bmatrix} \triangleq \begin{bmatrix} f_i(\mathbf{x}_1) \\ \vdots \\ f_i(\mathbf{x}_s) \end{bmatrix} \quad (4.77)$$

$\mathbf{T}_i$ has m columns, m being equal to the number of IQs under consideration, and s rows, corresponding to the number of points in the OLHS.

4.4.3 Dimensional Reduction

Before training the Gaussian process models and evaluate the loads, it is necessary to investigate how the different input parameters are to be handled, since one can also build multi-output Gaussian processes instead of an union of single-output Gaussian processes. In this work, a dimension reduction step was performed on the space of loads. This is done to avoid predicting all the forces in all the locations. The tool that accomplishes this is called principal component analysis [114] to transform the data into independent components. The advantage of using PCA is that principal component analysis preserves the correlations in the outputs. It has the advantage as well of reducing the number of models required. In addition to that, the new basis is by definition independent.

Principal Component Analysis

For the reasons previously described it is necessary to use the dimension reduction method known as principal component analysis. The PCA method is widely popular in data analysis and used in many different contexts [114, 231]. In this section, the treatment used is the one given by Loxham [199]. Principal component analysis is done using a singular value decomposition (SVD). SVD is a tool used in linear algebra to factor a real or complex matrix and it has connections with polar decomposition. In PCA, the principal components of the matrix that contain the data has the size $m \times s$. Along with it, one must denote $\mathbf{T}$ as the the eigenvectors. The covariance of the sample is $\mathbf{C}$. This term is defined as follows

$$\mathbf{C} = \mathbf{T}^\top \mathbf{T} / (s - 1) \quad (4.78)$$

$\mathbf{C}$ being a covariance matrix, it must be by definition symmetrical and positive semi-definite. From linear algebra, it is known that these matrices allow the operation of diagonalisation:

$$\mathbf{C} = \mathbf{V} \cdot \text{diag}(\lambda) \cdot \mathbf{V}^\top \quad (4.79)$$

Here $\mathbf{V}$ denotes a columnwise eigenvectors matrix of size $m \times m$. The diagonal operator, $\text{diag}(\lambda)$, is a diagonal $m \times m$ matrix of eigenvalues. Hence, the singular value decomposition $\mathbf{T}$ is as follows

$$\mathbf{T} = \mathbf{U} \mathbf{S} \mathbf{V}^\top \quad (4.80)$$

The term $\mathbf{U}$ is an unitary matrix (i.e. $\mathbf{U} \mathbf{U}^\top = \mathbf{U}^\top \mathbf{U} = \mathbf{I}$) The term $\mathbf{S}$ denotes the diagonal matrix of singular values. Inserting the equation 4.80 into 4.78 yields:

$$\mathbf{C} = \frac{\mathbf{V} \mathbf{S} \mathbf{U}^\top \cdot \mathbf{U} \mathbf{S} \mathbf{V}^\top}{(s - 1)} \quad (4.81)$$

which by equation 4.79 implies:

$$\mathbf{C} = \mathbf{V} \frac{\mathbf{S}^2}{s-1} \mathbf{V}^\top \quad (4.82)$$

yielding the following relationship between the singular values, S , and eigenvalues of the sample covariance matrix, λ :

$$\lambda = \text{diag}(\mathbf{S})/(s-1) \quad (4.83)$$

The dimension reduction happens when by the PCA method it is necessary to choose which eigenvectors shall be retained. Normally, the first l eigenvalues in descending order are the primary eigenvalues to be kept. The other eigenvectors are collected into a matrix P of size $m \times p$. The PCA operations correspond to a rotation or transformation of points. This is a connection with polar decomposition and one of the reasons for the success of PCA. This operation can be used to transform a collection of points in the original space onto the PCA basis, by a dot product operation. In mathematical terms,

$$\mathbf{Q}^* = \mathbf{T}^* \mathbf{P} \quad (4.84)$$

Principal component analysis is often seen in many application as a method that allows the identification of the directions of maximum variation in a dataset. As Loxham [199] points out PCA is sensitive to the scaling of the data. Thus, without scaling there is a tendency for the output of PCA to be biased. One common way to correct this issue is to scale the columns of the dataset first.

Assume that $\mathbf{t}_i$ is the i^{th} column of $\mathbf{T}$. One possible way to scale this data is to simply subtract the sample mean of the column and divide it by the standard deviation of the column.

4.4.4 Evaluation of Manoeuvre and Gust Loads using Gaussian Process Surrogate Models

As mentioned in the chapter 1, Airbus UK provided a dataset in which the values are normalised using a “password” of coefficients to protect industrial information. While this is understandable from an industrial perspective, it removes the opportunity for investigating the loads using engineering judgement. Thus, the approach taken here was the data-driven one, where the concern is on the results and how well does the underlying model replicated the dataset.

Since all the input parameters ($\theta_1, \theta_2, EI_1, EI_2, EI_3, EI_4, GJ_1, GJ_2, GJ_3, GJ_4$) are probability distributions, one can generate many draws as one wants. Thus, it is impractical to plot all the loads that may affect an aircraft. However, relying on just data analysis metrics is not sufficient for engineers. A common solution is therefore to describe the parameters by their mean and variance. However, it was found out that even when the mean and the variance are the same (see figure 4.1), very different results may be observed in the case there is correlation between the parameters or the parameters are independent, as it was show [227, 232]. A large component of the present work was the assimilation and coding of the tools described in this thesis. For load evaluation, a Gaussian process based on the theoretical considerations of section 4.1 and 4.2 can be condensed into the algorithm 4.1. This algorithm based

Algorithm 4.1: Gaussian Process Regression Algorithm

Input: $\mathbf{x}$ (inputs), $\mathbf{y}$ (output), $\boldsymbol{\mu}^*(f^*)$ (covariance function), σ_n^2 (noise level), $\mathbf{x}^*$

Output: $\boldsymbol{\mu}_*$ (mean), $\boldsymbol{\sigma}_*$ (variance) $\log p(\mathbf{y} | \mathbf{x})$ (log marginal likelihood)

- 1 $L \leftarrow \text{cholesky}(k + \sigma_n^2 I)$;
 - 2 $\boldsymbol{\alpha} \leftarrow L^\top \setminus (L \setminus \mathbf{y})$;
 - 3 $f^* \leftarrow \mathbf{k}_*^\top \boldsymbol{\alpha}$;
 - 4 $\mathbf{v} \leftarrow L \setminus \mathbf{k}^*$;
 - 5 $\boldsymbol{\mu}^*(f^*) \leftarrow k(\mathbf{x}^* \mathbf{x}^*) - \mathbf{v}^\top \mathbf{v}$;
 - 6 $\log p(\mathbf{y} | \mathbf{x}) \leftarrow -\frac{1}{2} \mathbf{y}^\top \boldsymbol{\alpha} - \sum_i \log L_{ii} - \frac{n}{2} \log 2\pi$
-

on conventional Gaussian process performed reasonably well in small sized datasets. Nevertheless, this initial modelling was useful to demonstrate the correlation between

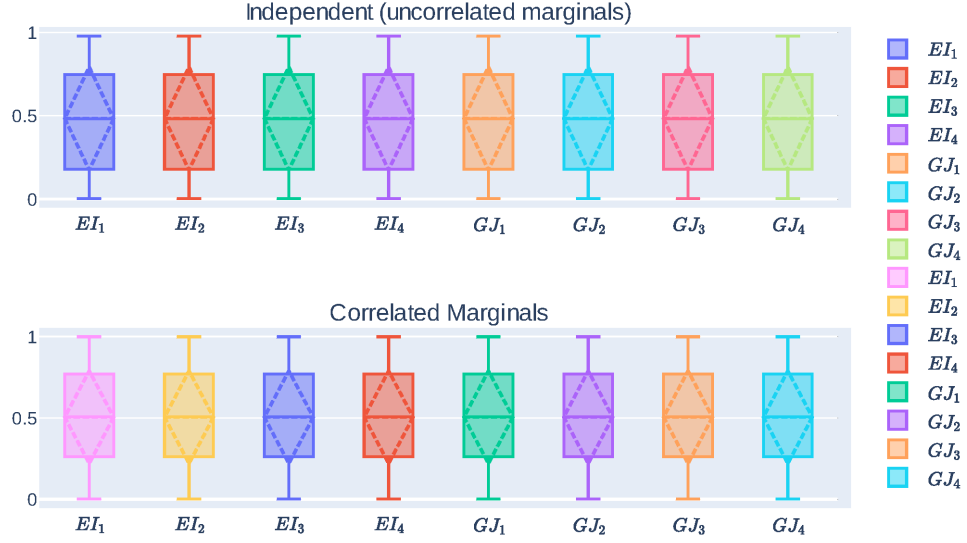


Figure 4.1: Boxplot of the model parameters

parameters have great influence on the design. One of the ways to assign a correlation between probabilistic variables is to use a Gaussian Copula. Let $\vec{y}$ be the flight loads black-box model and let X be the Gaussian Copula

$$\mathbf{Y} \sim f(\vec{x}, \vec{\lambda}) \quad (4.85)$$

$$X \sim \mathcal{C}(F_1, \dots, F_n) \quad (4.86)$$

Where $\mathbf{x}$ is the design variable vector; $\boldsymbol{\lambda}$ is the load case vector; $\mathbf{y}$ is the output vector loads; $F_1, \dots, F_n$ are the design uncertainties and $\mathcal{C}$ is the correlation structure between $F_1, \dots, F_n$. In addition to that, this correlation (often represented as figure 4.2) in a industrial setting has to be created out of expertise knowledge which has lots of subjectivity into it.

By inspecting figure 4.3, one can notice how different the loads are despite both variables having the same statistical metrics. The results are shown as force envelopes [233] which are plotted for the correlated variables and also for the independent variables.

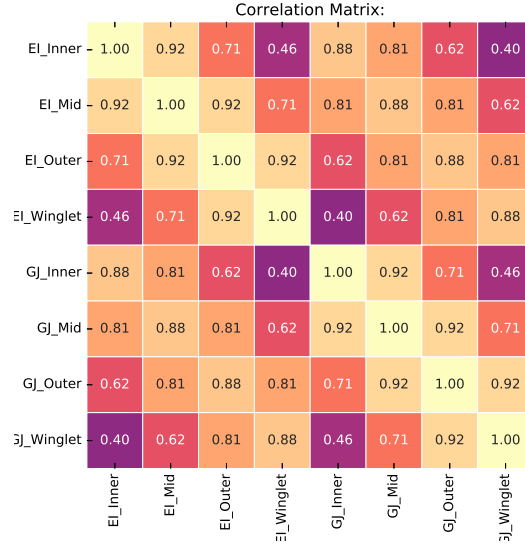


Figure 4.2: Correlation matrix

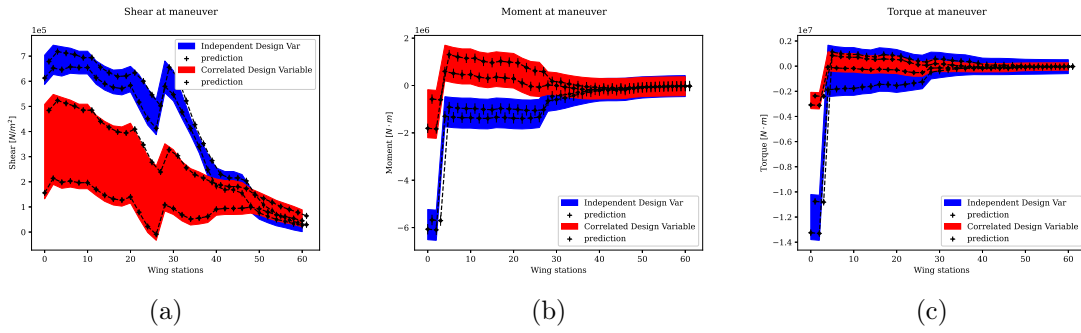


Figure 4.3: Independent parameters versus correlated parameters through the use of a Gaussian copula.

Each force envelope contain two lines, an lower and an upper one. The cross in each dot represents the variance estimated. The coloured area corresponds to the 95% confidence interval that is calculated taking the mean value of the lower and upper limit as the baseline. This result has important consequences for the use of probabilistic surrogate models in aeronautics. It is important to highlight that the correlation matrix used in this case may not be realistic, due to the difficulties in calculating or estimating the correlation between different parts of a wing structure. Nevertheless, it proves the point that the results might differ a lot, despite the same statistical metrics.

4.4.5 Analysis and Limitations

The conventional or standard Gaussian process formulation is a very powerful one to build probabilistic surrogate models at cheap computational cost. It has theoretical properties that allows one to evaluate how good its fit is and whether is capable in theory to model very complex physical phenomena. Another huge advantage of Gaussian process is its compatibility with Bayesian probabilistic modelling as it was show in sections 4.1 and 4.2. Moreover, it naturally provides the solution with errors bars that shows the quantified uncertainty.

The main limitations of the conventional Gaussian process is that its numerical capability is constrained by computational speed and memory. In the code developed during this research, a number of difficulties related to memory management had to be solved with partial success. However, it is important to notice that some of the memory issues detected may be connected to the number of python overheads which must be attributed to the programmer and not to the method. On the other hand, many authors [35, 234] were motivated by the standard Gaussian process limitations to develop new ways of improving its efficiency. This is useful for new technologies in aeronautics such as digital-twin. Digital-twin is supposed to rely on surrogate models to build huge models that are intended to replicate the physical performance of an entire aircraft or machine.

4.5 Summary and Conclusions

This chapter introduced Gaussian process for machine learning as a tool for probabilistic regression in the Bayesian framework. The primary concepts that led to the formulation of a GP were reviewed and used to demonstrate how the Bayesian posterior can be computed for a probabilistic model made out of Gaussian processes. Next, the main requirements for an engineering surrogate were given. This chapter also reviewed the concepts of flight physics and without entering into flight dynamics some of the ideas of why gust and manoeuvre loads are important in an aircraft programme were

discussed.

The number of load cases necessary to be evaluated in an aircraft programme motivates the use of GP for modelling load cases. The basic steps to build a probabilistic model capable of performing this load evaluation were covered. An important result was found, i.e, that for a given problem where all the parameter have same statistical metrics very different results may be obtained depending on the correlation matrix. In addition to that, it was discussed that it is not a trivial task to select or compute a correlation matrix. Thus, a designer or engineer must rely on expert judgement which is prone to bias.

Chapter 5

Flight Loads Evaluation Using Sparse Gaussian Process Emulators

The aim of this chapter is to investigate the use of sparse Gaussian process emulators in the modelling of the flight loads and compare it with standard Gaussian processes. To do so, in section 5.1, a brief introduction to the context and motivation for sparse formulation is given. Next, in section 5.2, all the mathematical machinery of this chapter are used to solve the probabilistic regression problem using sparse Gaussian processes and a comparison with alternative formulations is made. In section 5.3 the results obtained are discussed. Finally, in section 5.4, the main results and conclusions of this chapter are summarised.

5.1 Introduction

Gaussian processes (GPs) may be used as regression models that are relatively easy to implement. They are also flexible and capable of producing fully-fledged probability models that suit many applications. However, GPs have many limitations in terms of computational resources and memory limitations. For a conventional Gaussian process, the computational requirements in computing capability grow cubically, while the memory grows with the square of the number of training points, n . This fact limits the use of GPs to problems that do not require large number of training points. Many authors have attempted to circumvent these limitations with sparse approximations. The use of sparse approximations in Gaussian processes are implemented exactly in a subset of the latent variables. The remaining variables however are approximated, yielding a computationally less expensive format [29].

In this chapter, the goal is to remedy the computational issues identified in chapter 4. It starts by giving a somewhat unified view of sparse approximations that are useful to Gaussian process regression. The intent here is to provide a path for future aeronautical research by exploring somewhat “exotic” formulations of Gaussian process that require less computational resources. Although the aeronautical industry has access to powerful computational resources, there are number of new technologies and methods that require more scalable tools. In particular, the concept of digital-twins, which was covered in chapter 2.

The computational and storage limitation of the standard Gaussian process prevent widespread use in several applications. The main problem in Gaussian process lies in the inversion of the covariance matrix (see the algorithm 4.2, in chapter 4). Sparse methods address this problem by introducing sparsity, which reduce the matrix size that needs to be inverted. A comprehensive survey of sparse Gaussian processes is given by Liu et al. [235]. In the literature, there are a number ways to introduce sparsity in Gaussian Processing in order to address the issues and limitations pointed out in chapter 4. These are: the Nystrom, the inducing points approach, the variational inducing points and, lastly, the stochastic gradient descent based variational inducing

points.

5.2 Sparse Gaussian Process Approximations

Gaussian processes have many interesting properties, which makes them excellent tools as surrogate models, however for medium sized datasets, it is not unusual that they lead to computationally intractable integrals, despite the fact that the marginal likelihood, posterior, predictive and the prior are Gaussian distributions and hence tractable. These distributions in Gaussian processes may also become computationally intractable since the derivatives contain mathematical matrix operations that have grows cubically, $\mathcal{O}(n^3)$, where n is the number of observations in the GP.

To fix this computational issue, many authors developed low-rank approximations to the the complete Gaussian process avoiding the operations that have a computational cost of $\mathcal{O}(N^3)$. In the majority of cases, the basic insight is to approximate the term K_{ff} by a rank $M < N$ matrix. The resulting matrix often has the computational cost of $\mathcal{O}(NM^2)$. These methods may be understood as a mathematical learning that focus on a small number of (M) quantities that are considered very informative of the Gaussian process posterior. This shows that the GP can be approximated like this [29]. The following subsections present different ways that were proposed to approximate GP.

5.2.1 Inducing Point Approximations

One of the ways to approximate the Gaussian process is by inducing point approximations. In this approach, the n quantities that must be learned are function values. These values are called inducing outputs and they are located at specific input locations that are called inducing inputs. This approach contrasts with the coefficients of the basis functions explained in the previous chapter. The insight that makes this approximation possible is to reduce the number of input data points x to the GP from n to m , where $m < n$. The set of m points are called inducing points. Reducing the

number of input data points leads to a smaller covariance matrix K , with the potential to reduce the computational complexity from $O(n^3)$ to $O(nm^2)$. This reduction in computational complexity translates into a faster training and querying. Thus, this approximation method seeks to discover which n inputs and constrained outputs (by the GP prior) yield the most fit posterior to the true GP posterior.

5.2.2 Nystrom Approximation

The Nystrom approximation developed by Williams and Seeger [236] has the aim of reducing the inversion cost of the covariance matrix in a Gaussian process. This approximation consists of a procedure to generate a low rank approximation that is suitable to any kernel matrix. This approach works by selecting m data points with $m \leq n$ from the training set. This low rank approximation $\tilde{k}$ to the kernel matrix is defined as follows

$$\tilde{k} = k_{n,m} k_{m,m}^{-1} k_{m,n} \quad (5.1)$$

where $k_{n,m}$ represents the kernel matrix associated with the n and m data-points in the training dataset and the selected subset, respectively. Since the approximation needs the inversion of only an m_m matrix, the computation cost reduces from $O(n^3)$ to $O(m^3)$. This approximation however presupposes that the data comes from a low-rank manifold¹, which requires that the data dimension is $d < n$. In this case, the low rank approximation guarantees that there is not information loss. Although, the selected m data-points influence the approximation. In this formulation, there is still the possibility that some data-points will yield a low-fidelity approximation even when the data is from a low dimensional manifold. However, since most datasets have more datapoints than the number of features, this method is still applicable to most cases. The selection of data points is essential to the performance of the Nystrom approximation. In their work, Williams and Seeger used a random subset of m data points in their approach [236] and reported that data selection process limited the

¹A low-rank manifold is the set of $m \times n$ rank d , where m , n and $d < \min(m, n)$ are positive integers. This low-rank manifold may be informally understood as the space of covariance matrices that a given functional prediction relies on.

performance of the method. This issue of the subset selection was addressed by Snelson and Ghahramani [237] by the introduction of pseudo data, which means reformulating the subset as part of the model parameters.

5.2.3 Pseudo Points Approach

The pseudo points gets its name because it does not necessarily correspond to any data point in the dataset. Rather the pseudo data is assumed to be synthetic. The pseudo points are computed using the maximum likelihood, which requires an appropriate way to parameterise the Gaussian process. The work by Snelson and Ghahramani [237] introduced a distribution over the pseudo points and considered a joint distribution over latent representations of data from the training, testing and, pseudo points given by $\mathbf{f}$, $\mathbf{f}^*$ and $\mathbf{m}$. The marginalisation of the pseudo points to obtain the posterior distribution is as follows

$$p(\mathbf{y} \mid \mathbf{x}, \mathbf{m}) = \mathcal{N}(\mathbf{y} \mid k_{nm}k_{mm}^{-1}\mathbf{m}, \mathbf{\Lambda} + \sigma^2\mathbf{I}) \quad (5.2)$$

$$\mathbf{\Lambda} = \text{diag}(\boldsymbol{\lambda}) \quad (5.3)$$

$$\lambda_n = k_{nn} - k_{mn}^\top k_{mm}^{-1} k_{mn} \quad (5.4)$$

$$p(\mathbf{f}_*, \mathbf{f}) = \int p(\mathbf{f}_*, \mathbf{f}, \mathbf{m}) d\mathbf{m} = \int p(\mathbf{f}_*, \mathbf{f} \mid \mathbf{m}) p(\mathbf{m}) d\mathbf{m} \quad (5.5)$$

$$p(\mathbf{m}) = \mathcal{N}(\mathbf{0}, k_{\mathbf{m}, \mathbf{m}}) \quad (5.6)$$

The use of the maximum likelihood to determine the pseudo points distribution has one disadvantage - it has the risk of overfitting. Ideally it would be better to use a Bayesian approach to compute these pseudo points distribution conditioned in the training data set. This approach however is not feasible and it becomes intractable to analytical solutions of the pseudo points. The joint $p(\mathbf{f}^*, \mathbf{f})$ distribution of the pseudo points approach can be written as follows

$$p(\mathbf{f}^*, \mathbf{f}) = \int p(\mathbf{f}^*, \mid \mathbf{m}) p(\mathbf{f} \mid \mathbf{m}) p(\mathbf{m}) d\mathbf{m} \quad (5.7)$$

5.2.4 Variational Inducing Points

The assumption underlying this joint distribution limits the information a Gaussian process obtains from the information that the GP gets from the training set to be induced only through the pseudo set, which are referred as inducing inputs. The idea of treating the pseudo points as hyperparameters associating a prior over them was introduced by Snelson and Ghahramani [237]. However, when compared to the standard or default Gaussian process formulation, it resulted in an inaccurate posterior, because of the kernel approximation. This issue was approached by Titsias [209] using a variational approach. The variational approach establishes a lower optimisable bound to determine the inducing inputs and the kernel hyper-parameters, resulting in a tractable computation for the predictive distribution. The choice of the number of latent variables is made through the pseudo points method developed by Titsias and Lawrence [238] and explained in section 5.2.3, which works by cancelling out the intractable terms in the variational bound. The bound, $\mathcal{F}$, after this optimisation is:

$$\mathcal{F}_V = \log [\mathcal{N}(\mathbf{Y} + \mathbf{0}, \sigma^2 \mathbf{I} + Q_{n\pi})] - \frac{1}{2\sigma^2} \text{Tr}(\widehat{K}) \quad (5.8)$$

$$Q_{n\pi} = k_{nm} k_{mm}^{-1} k_{mn} \quad (5.9)$$

$$\widetilde{k} \neq k_{nn} = k_{nm} k_{mm}^{-1} k_{mn} \quad (5.10)$$

5.2.5 Stochastic Gradient Descent

This approach however lacked the factorisation term that is necessary for using an stochastic gradient descent algorithm, which scales well to large datasets while retaining the reduced model complexity of $O(m^3)$. Hensman et al. [239] improved on this by developing a new bound that could be optimised with stochastic gradient descent, as follows

$$\mathcal{F}_V = \sum_{i=1}^n \left\{ \log \mathcal{N}(\mathbf{y}_i \mid K_i^\top K_{mm}^{-1} \mathbf{v}, \beta^{-1}) - \frac{1}{2} \beta \widetilde{k}_{i,i} - \frac{1}{2} \text{tr}(\mathbf{S} \boldsymbol{\Lambda}_i) \right\} - \text{KL}(q(\mathbf{u}) \parallel p(\mathbf{u})) \quad (5.11)$$

$$q(\mathbf{u}) = \mathcal{N}(\mathbf{u} \mid \mathbf{v}, \mathbf{S}) \quad (5.12)$$

$$\mathbf{\Lambda} = \beta K_{mm}^{-1} k_i k_i^\top K_{mm}^{-1} \quad (5.13)$$

Where $\mathbf{u}$ is the set of feature space representations of the inducing points and k_i is the i^{th} column of k_{mn} . A good reference that surveys sparse Gaussian process is Liu et al. [235].

5.2.6 Likelihood Approximations

Another way to approximate a Gaussian process is by approximating its likelihood. Snelson and Ghahramani [240] presented a inducing point approximations based on approximations to the likelihood. In their approach the focus is on obtaining the posterior made out n inducing points. These points are chosen with the goal of speeding up the predictions. This is done by drawing from the posterior over the latent function at some of the marginal likelihoods, which are determined by the inputs.

$$p(f(\mathbf{x}) \mid \mathbf{y}, \mathbf{x}) = \frac{p(\mathbf{y} \mid f(\mathbf{x}), \mathbf{x})p(f(\mathbf{x}))}{p(\mathbf{y} \mid \mathbf{x})} \quad (5.14)$$

The marginalisation over $f(\mathbf{x})$ gives the likelihood $p(\mathbf{y})$, which in turn gives the likelihood and posterior, as follows:

$$\begin{aligned} p(\mathbf{y} \mid \mathbf{u}, \mathbf{x}) &= \int \prod_i p(y_i \mid f_i) p(\mathbf{f} \mid \mathbf{u}, \mathbf{x}, x) d\mathbf{f} \\ &= \mathcal{N}(\mathbf{y}; K_{fu} K_{uu}^{-1} \mathbf{u}, K_{ff} - K_{fu} K_{uu}^{-1} K_{uf} + \sigma^2 I) \end{aligned} \quad (5.15)$$

$$p(\mathbf{u} \mid \mathbf{y}, \mathbf{x}) = \mathcal{N}(\mathbf{u}; K_{uf} (K_{ff} + \sigma^2)^{-1} \mathbf{y}, K_{uu} - K_{uf} (K_{ff} + \sigma^2)^{-1} K_{fu}) \quad (5.16)$$

To obtain a first approximation in the likelihood, it is necessary to factor over each observation. This approximated likelihood may be denoted by $q(\mathbf{y} \mid \mathbf{u})$

$$q(\mathbf{y} \mid \mathbf{u}, \mathbf{x}) = \mathcal{N}(\mathbf{y}; K_{fu} k_{uu}^{-1} \mathbf{u}, \text{diag}[\Sigma] + \sigma^2 I). \quad (5.17)$$

By using the so-called Woodbury identity [237], the inversion of the $n \times n$ matrix is avoided. Thus, the computational cost is decreased.

5.3 Results and Discussion

This chapter has the goal of evaluating the entire spectrum of load cases using cheap, but powerful surrogate models of the Gaussian process family. The previous section introduced the topic of sparse approximation, which tend to improve the computational capability of the conventional Gaussian process. As it was mentioned in the previous chapter. The Gaussian process modelling using conventional (non-sparse) was made, many computational issues, specially related to memory were encountered. During the modelling using conventional (non-sparse) Gaussian process many computational issues, specially, related to memory use were encountered. Possibly, such issues would not affect a more powerful industrial machine. Nevertheless, the study and investigation of computationally cheaper posterior approximations that maintain accuracy are very desirable in engineering.

The standard Gaussian process showed in the previous chapter was extensively modified to accommodate the so-called fully independent training conditional approximation (FITC). This approximation is designed to match the variances of the conditional. However this is done with a certain flexibility between the training points, in a way that the training points preserve some independence that is designed to reduce the computational tractability [240]. The expression for the conditional in the FITC approximation is given by equations 5.19.

$$p_{FITC}(\mathbf{y} \mid \mathbf{u}) = \mathcal{N}(\mathbf{f}; k_{\text{fu}} k_{\text{uu}}^{-1} \mathbf{u}, \text{diag}[K_{\text{ff}} - q_{\text{ff}}] + \sigma^2 I) \quad (5.18)$$

$$q(\mathbf{f} \mid \mathbf{u}) = \mathcal{N}(\mathbf{f}; k_{\text{fu}} k_{\text{uu}}^{-1} \mathbf{u}, \text{diag}[k_{\text{ff}} - q_{\text{ff}}]) \quad (5.19)$$

The posterior over $\mathbf{u}$, the set of feature space representation, can be considered as an

exact inference in a modified Gaussian process equipped with a kernel $\tilde{k}$:

$$\tilde{k}(\mathbf{x}, \mathbf{x}') = \begin{cases} k(\mathbf{x}, \mathbf{x}'), & \text{if } \mathbf{x} = \mathbf{x}'. \\ k(\mathbf{x}, Z)k(Z, Z)^{-1}k(Z, \mathbf{x}) = \mathbf{k}_{f(\mathbf{x})u}K_{uu}^{-1}\mathbf{k}_{uf(\mathbf{x})}, & \text{otherwise} \end{cases} \quad (5.20)$$

In the fully independent training conditional approximation, one can choose the testing ($f^* |$) and the training conditional f to be exactly the same. Thus training and prediction are connected to the kernel exact inference. FITC is limited in the sense that its kernel mean has only n degrees of freedom. However, it has also the advantage that the inducing points could be chosen using gradient based optimisation on FITC's marginal likelihood.

$$p_{\text{FITC}}(\mathbf{y}) = \mathcal{N}(\mathbf{y}; 0, q_{\text{ff}} + \text{diag}[K_{\text{ff}} - q_{\text{ff}}] + \sigma^2 I) \quad (5.21)$$

In FITC, the exact solution is reconstructed in case of $Z = \mathbf{x}$. When the inducing points are optimised in a FITC approximation there is a tendency for this approximation to deviate from the Gaussian process model. A major drawback is that it is not possible with this approximation to assess its closeness to the exact solution. However, FITC is still capable of providing the quantification of uncertainties.

An important field of further research is the application of iterative matrix inversion methods to improve the speed and reduce the computational cost of inverting the covariance matrix, which is central to any Gaussian process. This class of numerical methods are designed to improve a numerical approximation to the inverse of a matrix by the method of successive iterations. In principle, as long as these methods are designed for positive-definite squared matrices, there should be gains in performance. This is a strict requirement that must be obeyed because the covariance matrix in Gaussian process is a positive define kernel. This type of kernel is key to GP because it allows the intuitive idea of “similarity of patterns” to be mathematically encapsulated in a positive define kernel. Such a positive definite kernel is able to reproduce a kernel

Hilbert space², which allows the use of the tools of functional analysis, which is the basis for Gaussian process. Thus, any method iterative matrix inversion applicable to squared positive-definite matrices may potentially improve the performance and perform better.

5.3.1 Analysis and Limitations

Having explained the sparse approximation used, its application in load evaluation may be analysed. It is important to note that the dataset that Airbus UK has provided has the cases numbered, but not identified. Therefore, it is not possible to apply any engineering judgement to evaluate whether the load distributions match reality or do not. The sole guide of correctness is the data-driven approach used during the building of the models that depends on the concepts explained in chapter 3. Thus, the following model of shear, moment and torque for 21 cases of manoeuvre and 25 cases of gust is completely evaluated for the baseline configuration.

$$\mathbf{SMT} = \mathcal{GP}_{\text{loads}}(\bar{\theta}_1, \bar{\theta}_2, \overline{EI}_1, \overline{EI}_2, \overline{EI}_3, \overline{EI}_4, \overline{GJ}_1, \overline{GJ}_2, \overline{GJ}_3, \overline{GJ}_4, \lambda) \quad (5.22)$$

$$\mathbf{SMT} = \mathbf{S} \quad \cup \quad \mathbf{M} \quad \cup \quad \mathbf{T} \quad (5.23)$$

$$\text{and } \mathcal{GP}_{\text{loads}} = \mathcal{GP}_{\text{shear}} \quad \cup \quad \mathcal{GP}_{\text{moment}} \quad \cup \quad \mathcal{GP}_{\text{torque}} \quad (5.24)$$

In order to understand the results plotted in the next figures, it is important to note that area contained between the two lines corresponds to the structural engineering concept of “force envelope distribution” [233]. The force envelope distribution is the visual representation of the maximum (upper design margin) and minimum force values (lower design margin) that were probabilistic inferred during the calculation. The minimum and the maximum force corresponds to the two points at each wing-station, respectively. The shear, the moment and the torque envelope are thus shown in the

²In functional analysis, a reproducing kernel Hilbert space (RKHS) [241] is a Hilbert space of functions in which point evaluation is a continuous linear functional. This means that two functions f and g in the RKHS are close in norm, that is, $\|f - g\|$ is small, then f and g are also point-wise close, that is to say $\|f(x) - g(x)\|$ is small for all x .

figures, along with the uncertainty quantified for the upper and the lower design margin, which are represented by the bars in each point. Having these two points, it is possible to calculate a mean value and from this mean value we can estimate the 95% confidence interval to which the gray area corresponds.

The first observation one can make for the group of load cases in figures 5.1 and 5.2 is that when taken together it seems that the surrogate models were able to capture a myriad of flight physics conditions. The magnitude of the shear stress for manoeuvre loads from 1 to 13 are positive. This is consistent with the sign convention showed in chapter 3. Some “loads envelopes” are quite small, while others have large area, indicating that on this load case there are many possibilities that can happen. Regarding the quantified uncertainty, the smaller the bars, the less uncertainty associated. This probably means that these are load cases associated with more “conventional” flight conditions. The shear on load case 15 of the manoeuvre has an inflection point and the uncertainty on each point is very large. Load cases 9, 14, 15, 18, 19 and 20 shows interesting patterns and negative values.

Figures 5.3 and 5.4 shows the moment at manoeuvre conditions for 21 cases. All the 21 cases have negative magnitude. Case 9 to 13 have very small confidence of interval and the uncertainty is small when compared to other cases such as case 14 and 15. Again, this seems to be compatible to what is known of flight physics for an aircraft. The torque load cases in 5.5 and 5.6 follow a pattern similar to the moment, which is physically sensible.

It is interesting to note that all gust load cases in figures 5.7, 5.8, 5.9, 5.10, 5.11 and 5.12 have very interesting envelopes that are very large when compared with the manoeuvre load cases. It is also possible to conclude that the gust conditions experienced by the tip of the wing are less varied than in other segments of the wing.

5.4 Summary and Conclusions

In this chapter, the concept of sparse Gaussian processes was presented. Different schemes of approximation were surveyed. It was highlighted that for conventional

Gaussian processes, the computational requirements grow cubically, while the memory grows with the square of the number of training points, n . Thus, sparse approximations seek to solve this issue. They can be grouped into inducing point approximations, the Nystrom approximation, pseudo data approach, variational inducing points, stochastic gradient descent and likelihood approximations. All these approximations attempt to reduce the computational cost by approximating the posterior computation in a less expensive way.

Although the identification of the load cases were not disclosed by Airbus, it was possible to verify that the surrogate models are capable of capturing a variety of conditions and phenomena. Moreover, despite the drawbacks of the FITC sparse formulation adopted, it can be concluded that they are superior in performance to the conventional Gaussian process. Thus, in future endeavours of aircraft engineering, specially for novel configurations, such approximations might offer a great extend in capability. Moreover, novel technologies such as digital-twin and the introduction of machine learning methods in aircraft certification are going to offer more opportunities to the employment of special Gaussian processes that are capable of more predictive power without sacrificing the performance.

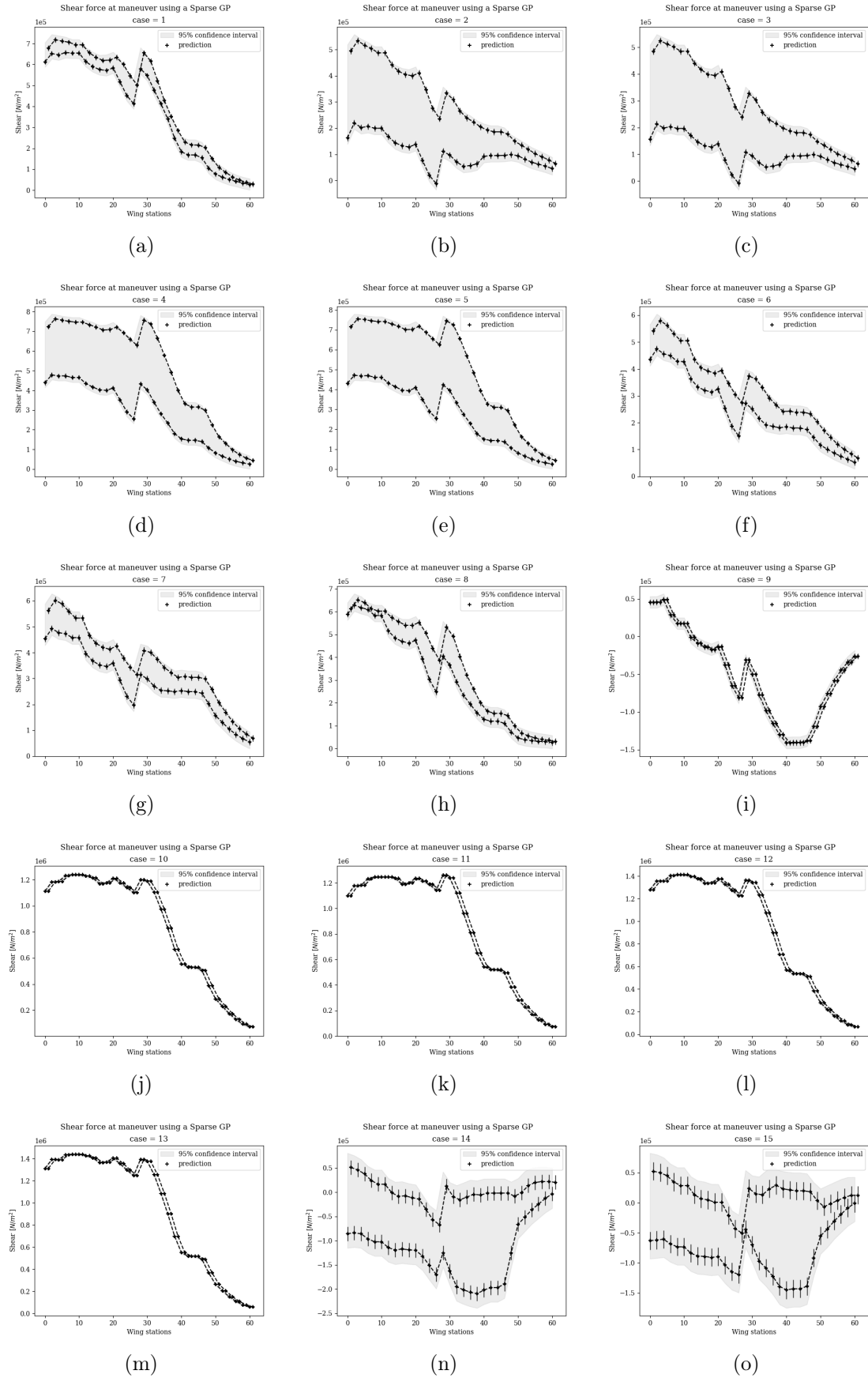


Figure 5.1: Calculated shear loads for 21 maneuver cases along the wingspan using sparse GP approximation

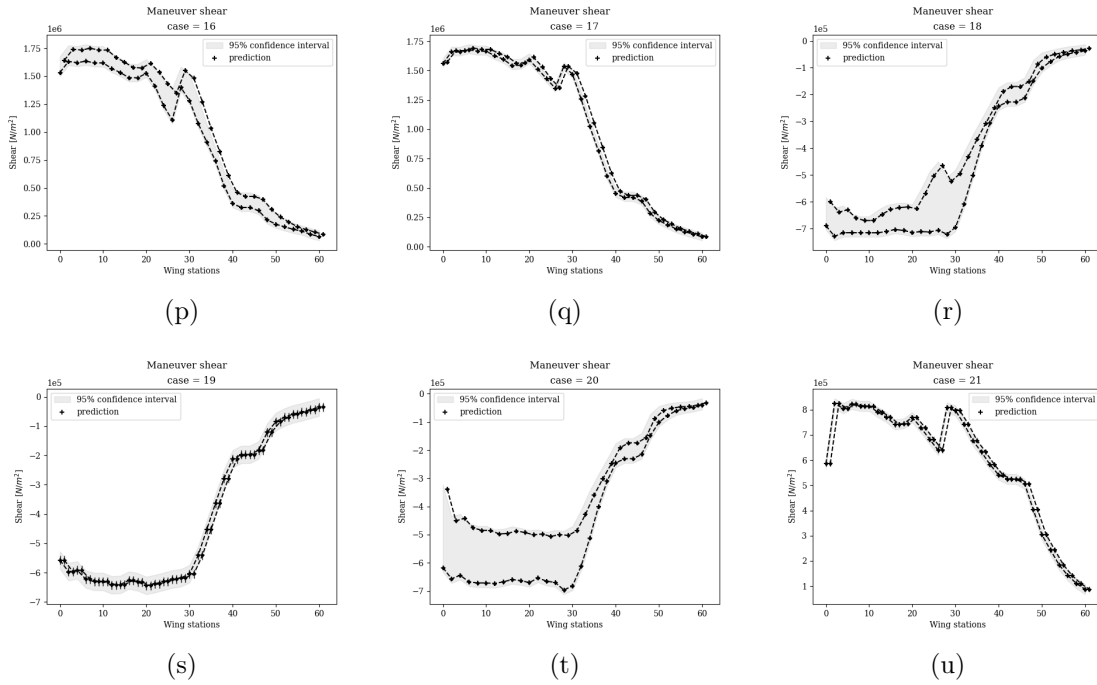


Figure 5.2: Calculated shear loads for 21 maneuver cases using sparse GP approximation continued

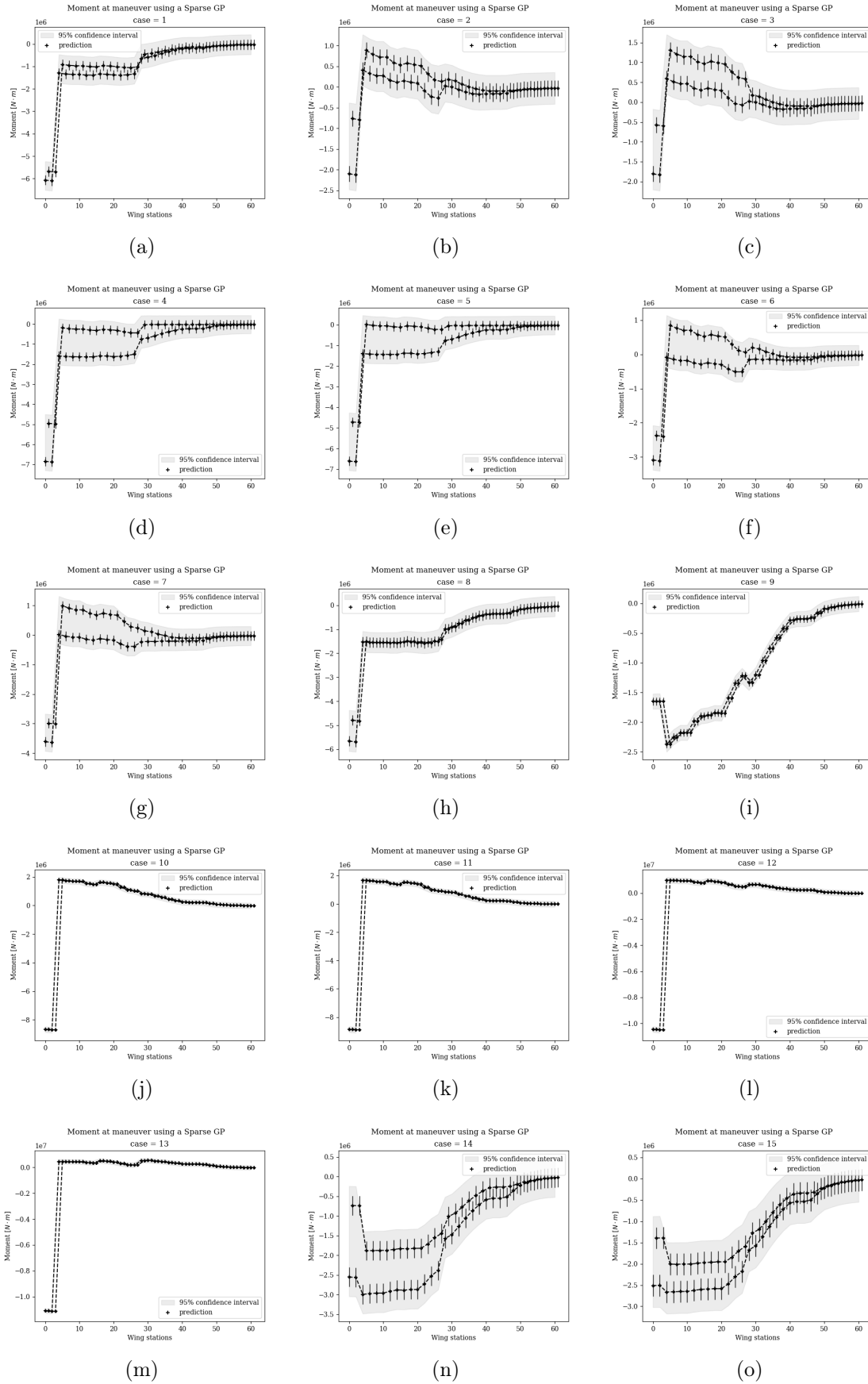


Figure 5.3: Calculated moment loads for 21 maneuver cases along the wingspan using sparse GP approximation

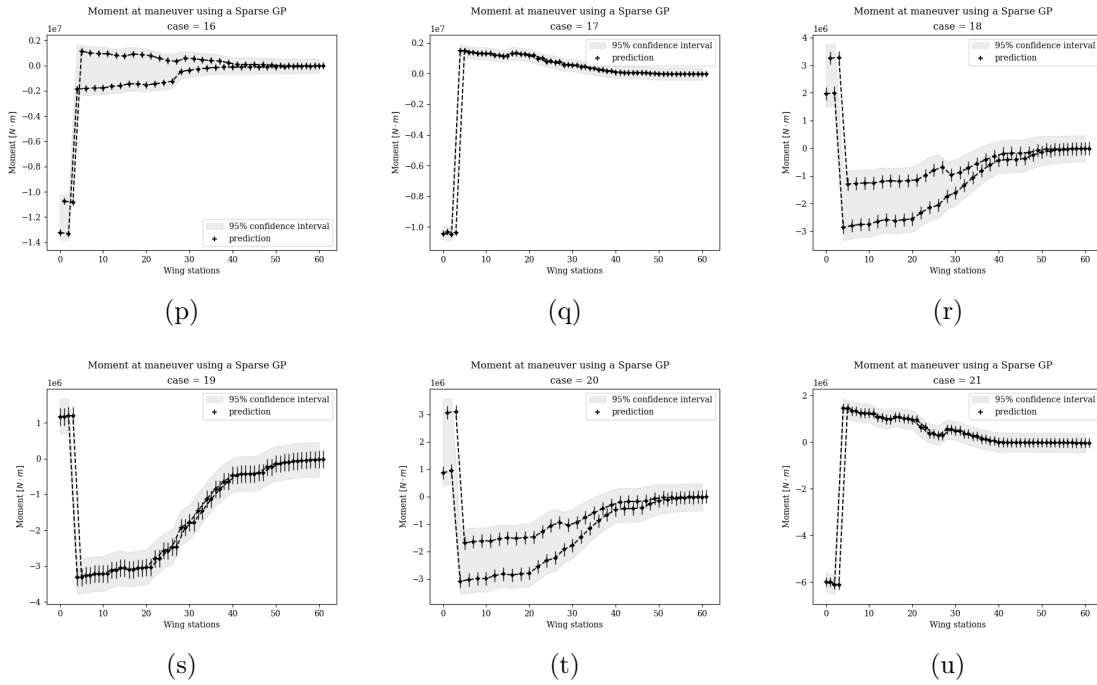


Figure 5.4: Calculated moment loads for 21 maneuver cases using sparse GP approximation along the wingspan continued

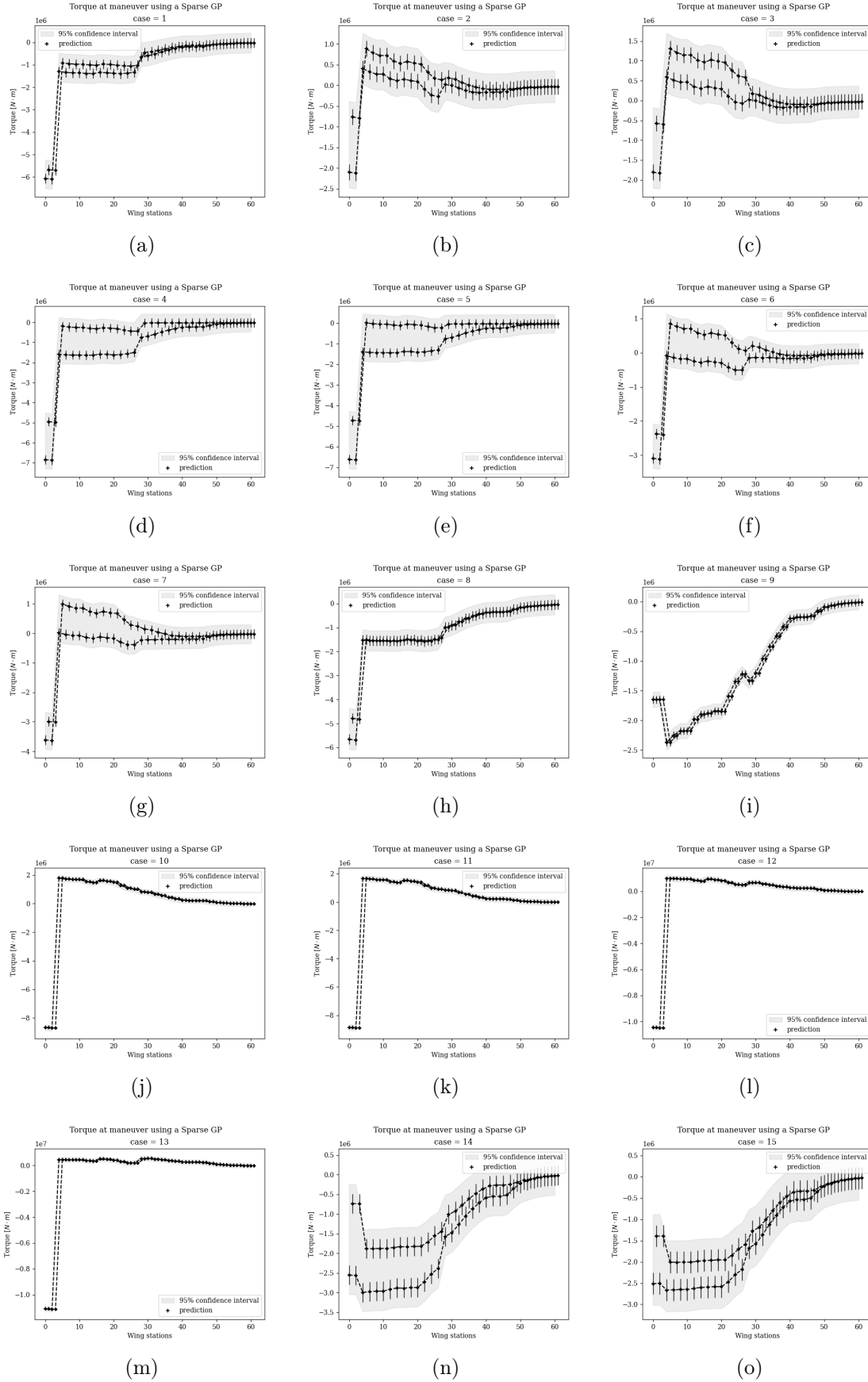


Figure 5.5: Calculated torque loads for 21 maneuver cases using sparse GP approximation along the wingspan

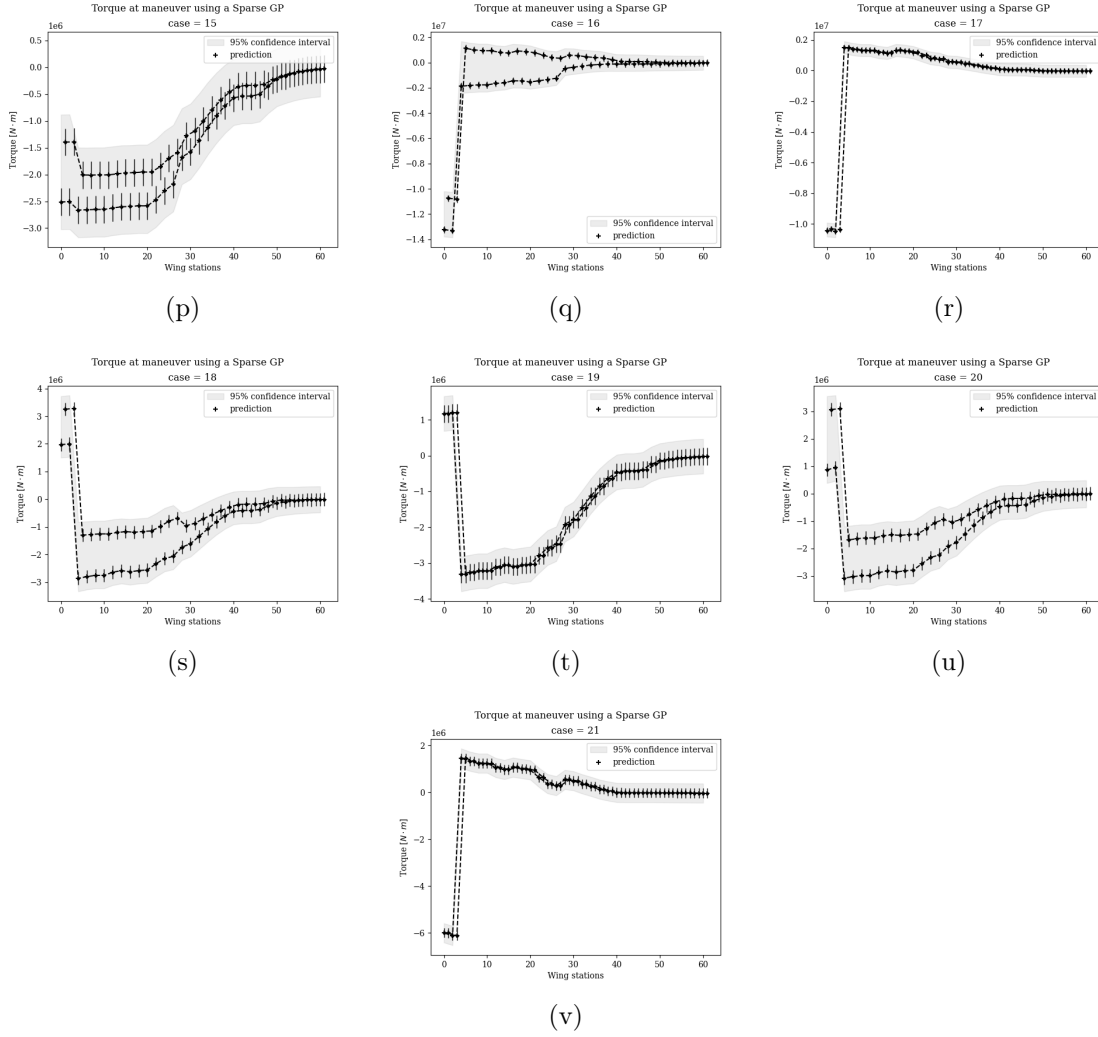


Figure 5.6: Calculated torque loads for 21 maneuver cases using sparse GP approximation along the wingspan continued

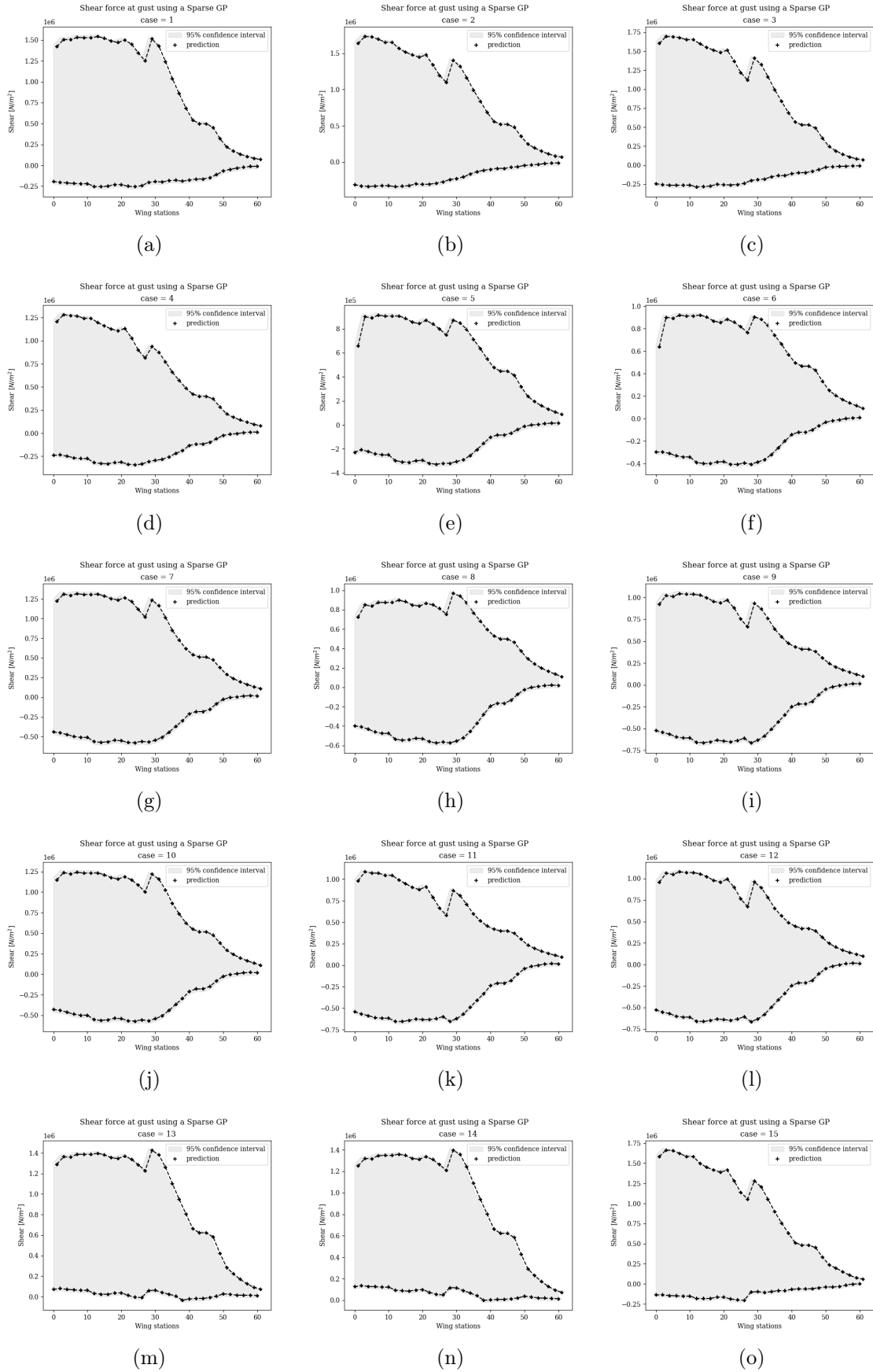


Figure 5.7: Calculated shear loads for 25 gust cases using sparse GP approximation along the wingspan continued

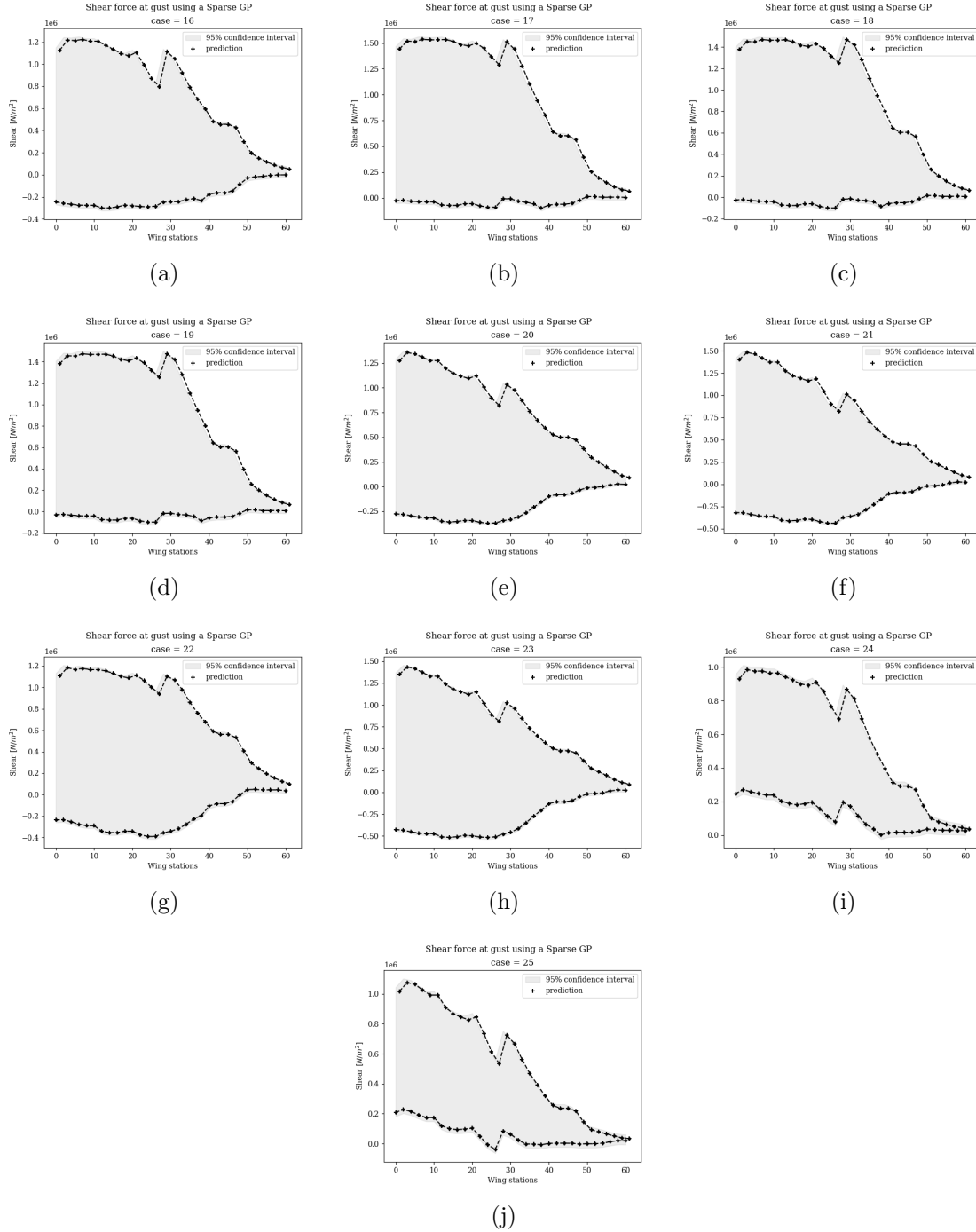


Figure 5.8: Calculated shear loads for 25 gust cases using sparse GP approximation along the wingspan continued

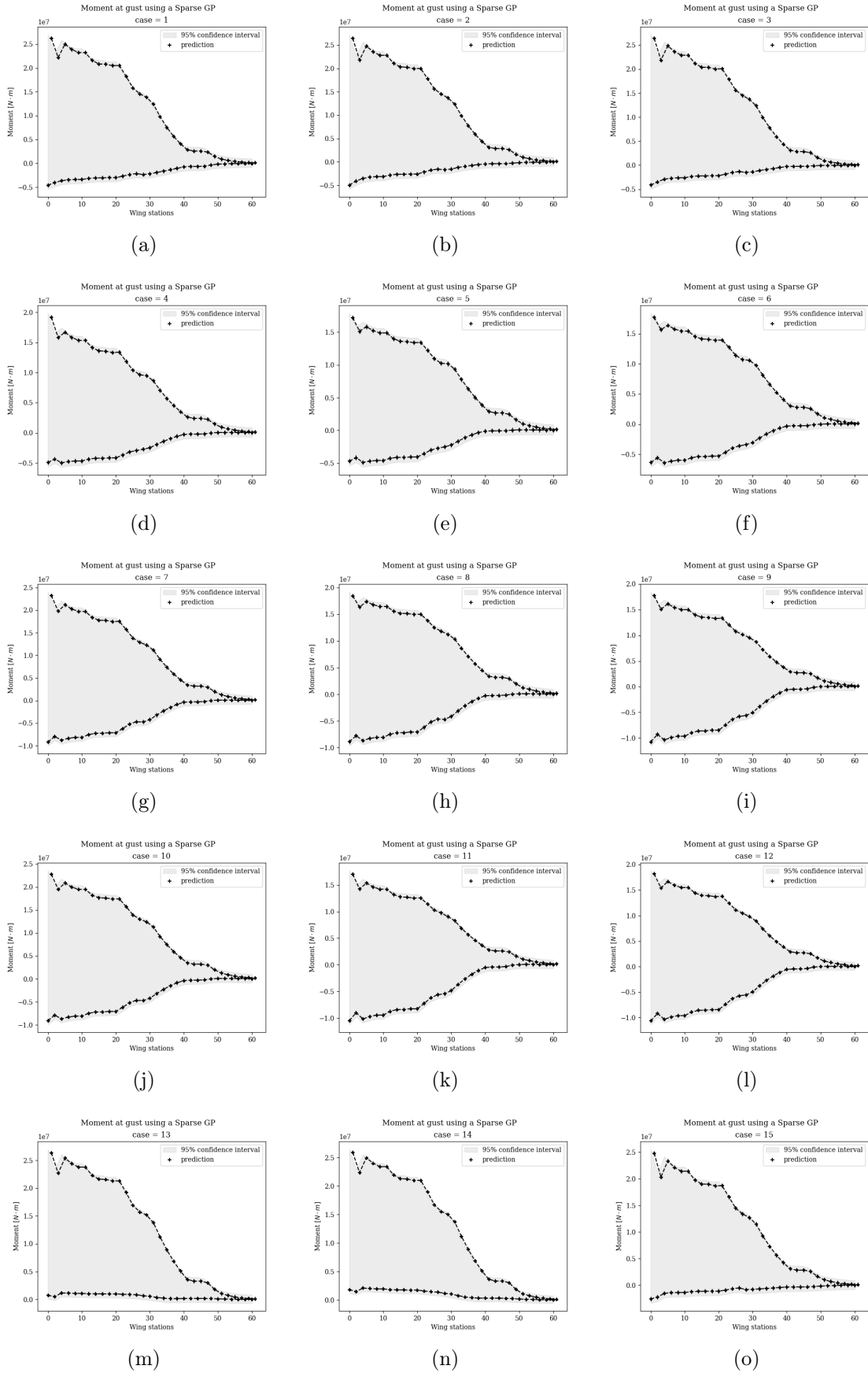


Figure 5.9: Calculated moment loads for 25 gust cases using sparse GP approximation along the wingspan

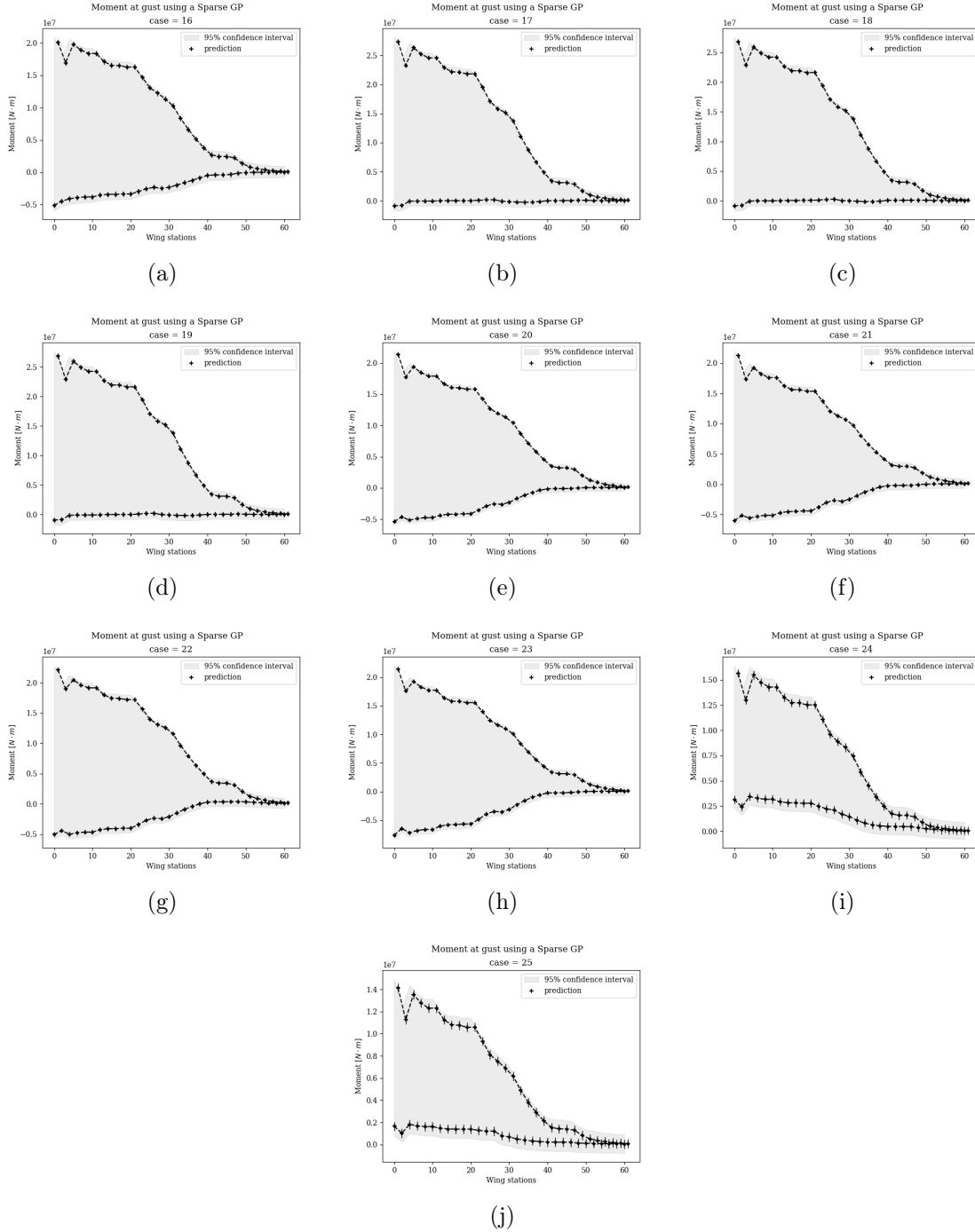


Figure 5.10: Calculated moment loads for 25 gust cases using sparse GP approximation along the wingspan continued

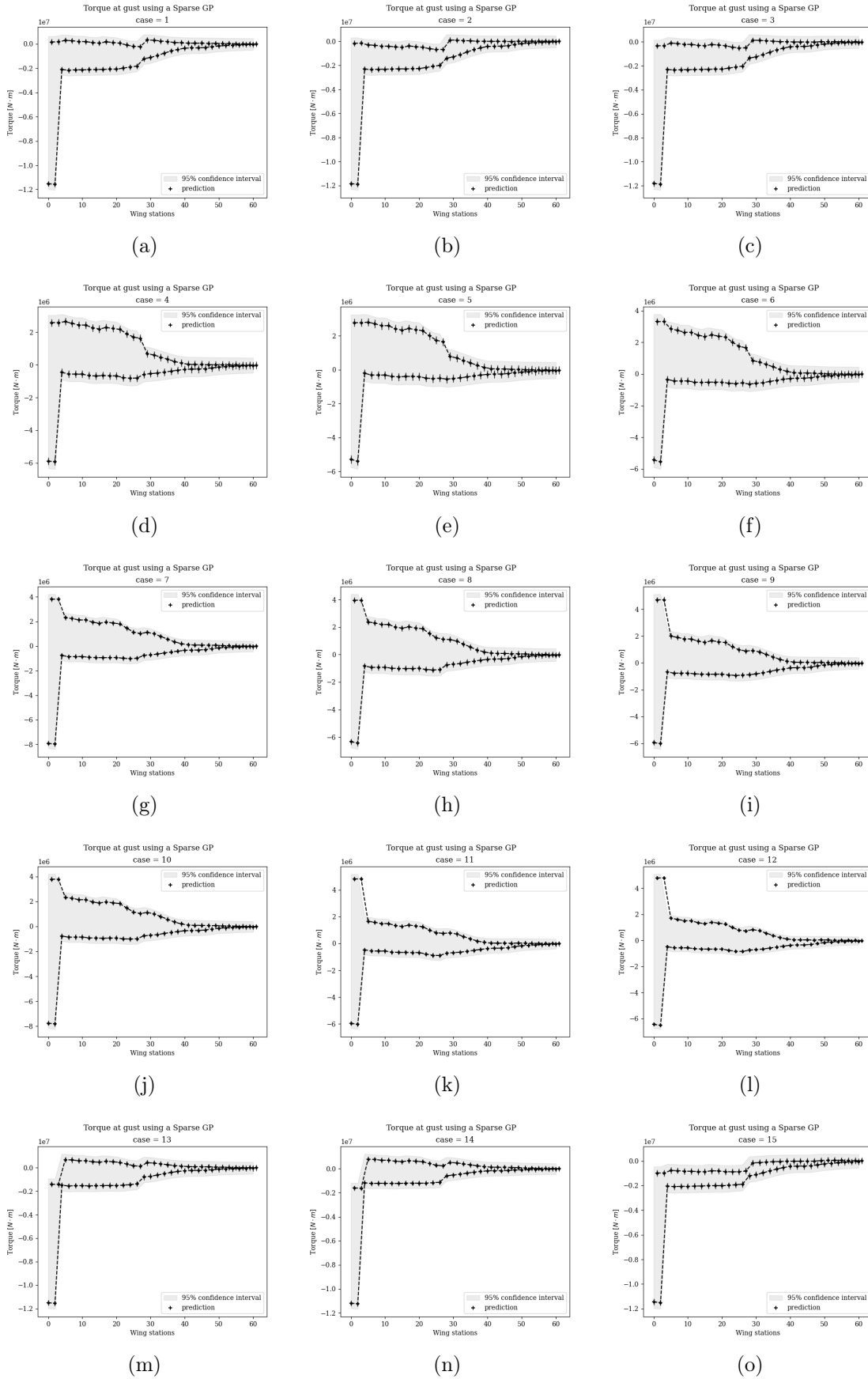


Figure 5.11: Calculated torque loads for 25 gust cases using sparse GP approximation along the wingspan

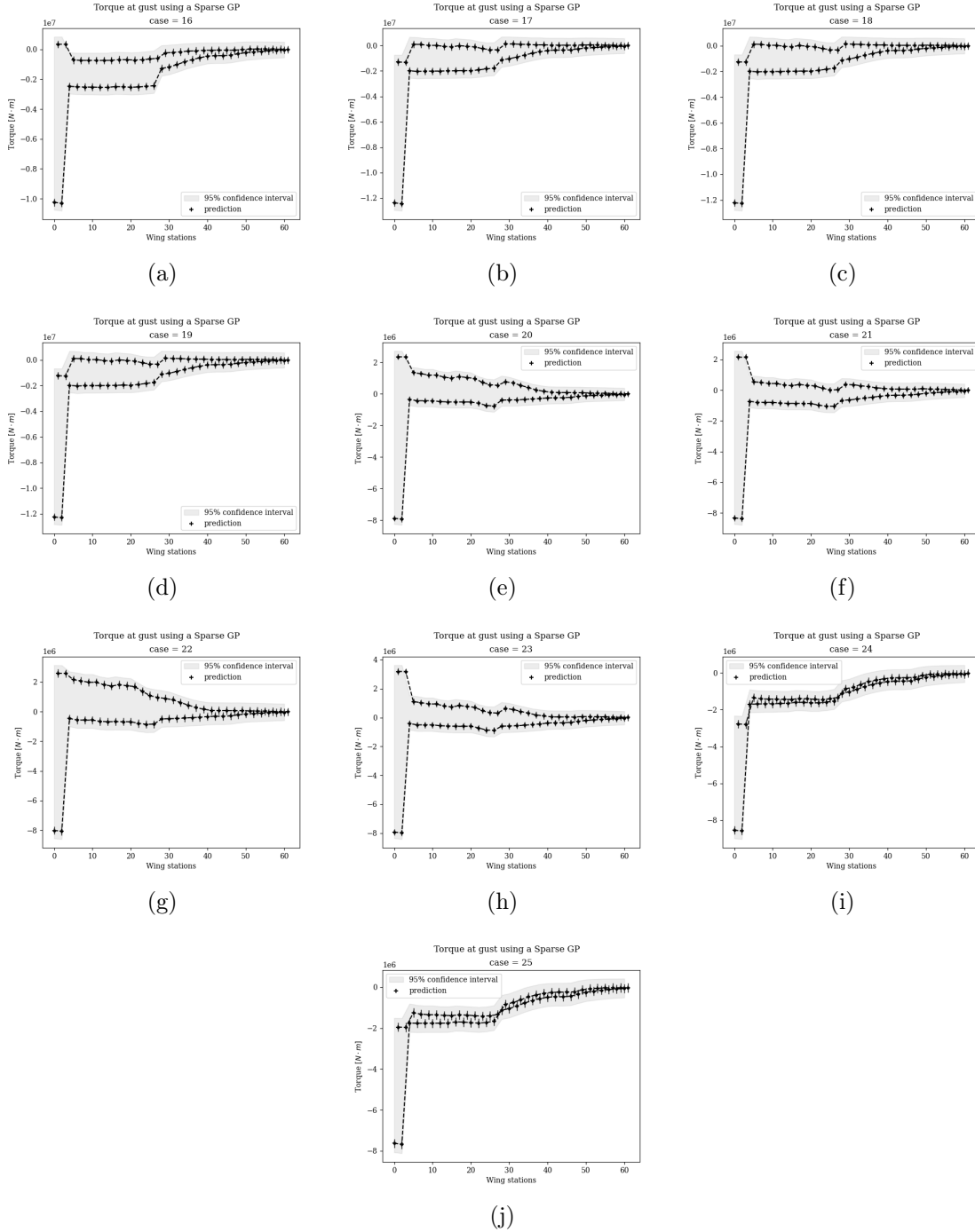


Figure 5.12: Calculated torque loads for 25 gust cases using sparse GP approximation along the wingspan continued

Chapter 6

Robust Aerostructural Optimisation for Improved Lift-to-Drag under Design Load Margin Constraints

The final chapter of this thesis subsumes the previous chapters and a complete optimisation solution involving the structural uncertainties and the design load constraints of the problem that motivated this thesis is presented. Section 6.1 discusses in more detail the context and challenges of the problem. Section 6.2 introduces the basic theory of Bayesian optimisation. Next, section 6.3 presents the complete formulation and algorithm of Bayesian optimisation with multiple constraints. Section 6.4 presents methods of sensitivity analysis that can be used to investigate how the outputs vary in terms of the inputs. Based on the previous theoretical analysis and experiments, the results of the robust optimisation for aeroperformance under constraint loads problem are presented and discussed in section 6.5. Finally, in section 6.6, the main points are restated along with the final conclusions.

6.1 Introduction

In this chapter, the goal is to use all the knowledge contained in the previous chapters to develop a powerful optimiser that can be used to perform the “optimisation under uncertainties” in an aircraft programme, where the sizing of the aircraft is in an advanced state, but there is still an opportunity to seek some gains in performance by choosing an adequate tooling.

In accordance to previous chapters, the family of optimisers selected for this task is the Bayesian one. This is not only to keep all the tools compatible with the Bayesian paradigm, but also because Bayesian optimisation has unique attributes. Bayesian Optimisation (BO) is a method of optimising expensive objective functions. These objective functions are often the result of systems or problems involving distributed design choices [60]. This versatility allows BO to be applied in a variety of products and different fields. The end-products of the optimisation often comprises several configuration parameters that require fine tuning, such as recommendation systems, medical analysis tools, speech recognizers, real-time physics engines and self-driving cars [60].

The problems that Bayesian optimisation is able to tackle may involve unusual search spaces containing categorical, conditional, combinatorial or even causal inputs. Additionally, it is assumed that the function f is a black-box function that has no simple analytical form, but can be evaluated at any arbitrary query point $\mathbf{x}$ in the domain. This evaluation yields stochastic (noise-corrupted or uncertain) outputs $\mathbf{y} \in R$ that when taken as an expectancy results in $E[y \mid f(x)] = f(x)$. Thus, Bayesian optimisation offers great promises for both probabilistic modelling and also engineering design.

6.2 Bayesian Optimisation

Bayesian Optimisation can be defined mathematically as the following problem. The goal is to find a global maximiser (or minimiser) of an unknown objective function

f :

$$\mathbf{x}^* = \arg \max_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) \quad (6.1)$$

where $\mathcal{X}$ is the design space of interest. In global optimisation, $\mathcal{X}$ is often taken as a compact subset of $\mathbb{R}^n$, but it does not need to be restricted to that.

To perform a Bayesian Optimisation (BO) one needs to build a model that is often referred to as a surrogate model or emulator. This model will be queried (or updated) at each Bayesian iteration driving in this way the optimiser decisions. By observing the Bayesian optimisation formulation in 6.1 one can notice that in the context of both knowledge representation [242] and model-based machine learning [190], BO can be considered a sequential model-based approach to problem solving [243] (see section 2.5.4 of chapter 2 for details).

Bayesian Optimisation in its simplest setting has three components: a surrogate model, an objective function and an acquisition function. The optimiser queries the objective function (often a surrogate model too) and builds its own surrogate model for the objective function, f . An usual choice for this surrogate is Gaussian Process regression (see section 4.2). Through the use of an acquisition function, the optimiser is able to decide where to sample next. The purpose of the acquisition function, α , is to evaluate the utility (or loss function) of candidate points for the next query of f . In this way, the next query, $\mathbf{x}_{n+1}$ is chosen by maximising α_n , where the index n represents the optimiser iteration.

The choice of a Gaussian Process as the surrogate model implies that we can prescribe a prior belief (see section 4.1) over the objective function parameters. Then, via Bayesian posterior updating, at each iteration the optimiser will refine its model as more data are observed. The Bayesian posterior represents our updated beliefs, after the given data has been observed, on the likely objective function being optimised. This probabilistic model allows the acquisition functions $\alpha_n : \mathcal{X} \mapsto \mathbb{R}$ to be sequentially evaluated while leveraging the uncertainty in the posterior to guide the exploration. In mathematical terms, given a dataset $\mathcal{D}$ and a likelihood model $p(\mathcal{D} \mid \mathbf{w})$, the posterior

distribution $p(\mathbf{w} \mid \mathcal{D})$ is inferred using the Bayes' rule:

$$p(\mathbf{w} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathbf{w})p(\mathbf{w})}{p(\mathcal{D})} \quad (6.2)$$

This posterior represents our updated beliefs about $\mathbf{w}$ after observing data $\mathcal{D}$. The denominator $p(\mathcal{D})$ is the marginal likelihood, or evidence, and is usually computationally intractable. The simplest Bayesian algorithm featuring its essential ingredients - an acquisition and an objective function can be written the following algorithm 6.1:

Algorithm 6.1: Bayesian Optimisation Algorithm

Output: optimum $\mathbf{x}_i$

- 1 Place a Bayesian Gaussian Process prior on f
 - 2 Observe f at $n_0 > 0$ points
 - 3 Set $n = n_0$ and $\mathcal{D}_n = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$
 - 4 **for** $n = 1, 2, \dots, N$ **do**
 - 5 Compute the posterior $f \mid \mathcal{D}_n$ using:

$$p(f \mid \mathcal{D}_n) = \frac{p(\mathcal{D}_n \mid f)p(\mathcal{D}_n)}{p(\mathcal{D}_n)} \quad (\text{Bayes Theorem}) \quad (6.3)$$
 - 6 Compute the acquisition function $\alpha \mathbf{x}$ based on the posterior.
 - 7 Compute

$$\mathbf{x}_{n+1} = \arg \max_{\mathbf{x} \in \mathcal{X}} \alpha_n(\mathbf{x}) \quad (6.4)$$
 - 8 Observe

$$y_{n+1} = f(\mathbf{x}_{n+1}) + \epsilon_{n+1} \quad (\text{where } \epsilon \sim \mathcal{N}(0, \eta^2)) \quad (6.5)$$
 - 9
$$\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{(\mathbf{x}_{n+1}, y_{n+1})\} \quad (6.6)$$
 - Increment n
 - 10 Return the point with the largest y_i
-

The goal of the next section is to explain the acquisition functions used in this research, without the complex mathematical machinery found in the specialised literature.

6.3 Acquisition Functions

As it was mentioned, Bayesian optimisation has fundamentally three components, acquisitions functions being on them. In this work, any acquisition function is denoted

by the Greek letter α . Acquisition functions have the purpose of informing and deciding where the optimiser is going to sample next. Many acquisition functions are first-principled meaning that they are derived from first principles such as information-theoretic results such as entropy. The choice of an acquisition function is still somewhat based on experiences. Yet, there are few theoretical guarantees that tell one which acquisition function may perform better in a given problem. Thus, the selection of an acquisition functions is still based on trial and error, test and comparison. Several acquisition functions of engineering interests are presented in detail next, along with its algorithm. In this way, the reader is able to get a brief overview of the different acquisition functions without the need to access the Bayesian optimisation literature, which at the present moment, is a bit arcane.

6.3.1 Expected Improvement

The first acquisition function that was derived and it is the first one a beginner is likely to encounter is the expected improvement (EI) acquisition function. It was first derived by Mockus [244]. EI computes the expected improvement for a given input over the current best observed value. This acquisition function encourages both exploitation and exploration because it is large for inputs with a low predictive mean (exploitation) and/or a high predictive variance (exploration). If one's goal is to evaluate an objective function f at $\mathbf{x}$, the expected improvement acquisition function is defined by

$$\alpha_{EI} = \text{EI}(\mathbf{x}) = \mathbb{E}[(t - y)_+] = \int_{-\infty}^{\infty} (t - y)_+ p(y | \mathbf{x}) dy \quad (6.7)$$

Considering that the model that learns (in machine learning sense) is a Gaussian process, the predictive posterior distribution is a Gaussian. In this case, because of Gaussian properties, the expected improvement has an analytical form:

$$\text{EI}(\mathbf{x}) = \sigma(\mathbf{x})(z(\mathbf{x})\Phi(z(\mathbf{x})) + \phi(z(\mathbf{x}))) \quad (6.8)$$

In the above expression at equation 6.8, the term $z(\mathbf{x})$ equals $\frac{t-\mu(\mathbf{x})}{\sigma(\mathbf{x})}$. The predictive mean is given by $\mu(\mathbf{x})$ at $\mathbf{x}$, likewise the predictive variance is $\sigma^2(\mathbf{x})$ at $\mathbf{x}$. $\Phi(\cdot)$ is the standard normal cumulative distribution function (CDF), and $\phi(\cdot)$ is the standard normal probability distribution function(PDF). This function is differentiable and fast to compute, hence it may be maximised with a standard gradient-based optimiser. It is easier to understand the expected improvement acquisition function by its algorithm, such as shown in algorithm 6.2.

Algorithm 6.2: Expected Improvement

Data: Write here the required data

Result: X_t, Y_t

- 1 Generate initial set of k points $\mathbf{x}_k = (x^{(1)}, \dots, x^{(k)})$;
- 2 Evaluate initial points $y_k = (y^{(1)} = f(\mathbf{x}^{(1)}), \dots, y^{(k)} = f(\mathbf{x}^{(k)}))$;
- 3 Set X_k, Y_k to the subsequence of non-dominated solutions among X_k, Y_k ;
- 4 $t \leftarrow k$;

5 **while** $t < t_{max}$ **do**

6 Update the Gaussian model based on x_t, y_t ;

7 $t \leftarrow t + 1$;

8 Choose

$$\mathbf{x}^{(t)} \in \arg \min_{\mathbf{x} \in \mathcal{X}} \text{EI}(\mathbf{x}, \mathbf{Y}_t) \quad (6.9)$$

$$y^{(t)} = \mathbf{f}(\mathbf{x}^{(t)}) \quad (6.10)$$

$$X_t = X_{t-1} \cup \mathbf{x}^{(t)} \quad (6.11)$$

$$Y_t = Y_{t-1} \cup y^{(t)} \quad (6.12)$$

Set X_t, Y_t to the sequence of non-dominated solutions among X_t, Y_t ;

9 **return** X_t^{nd}, Y_t^{nd}

6.3.2 Probability of Improvement

The acquisition function known as probability of improvement seeks to compute the probability of improvement for a given input point over the currently best observed function value. Basically it puts a prior of over the results yielded by expected improvement acquisition function. Its inner workings are shown in algorithm 6.3.

$$\alpha_{PI}(\mathbf{x}) = \text{EI}(\mathbf{x}) \Pr(\mathcal{C}(\mathbf{x})) \quad (6.13)$$

$$= \text{EI}(\mathbf{x}) \prod_{k=1}^K \text{Pr}(\mathcal{C}_k(\mathbf{x})) \quad (6.14)$$

Algorithm 6.3: Probability Of Improvement

Data: D , Gaussian process prior, $\mathcal{GP}(0, k)$

Result: Write here the expected result

1 **for** $t = 1, \dots, T$ **do**

2 Choose

$$\mathbf{x}_t = \underset{\mathbf{x} \in D}{\text{argmax}} \text{EI}(\mathbf{x}) \prod_{k=1}^K \text{Pr}(\mathcal{C}_k(\mathbf{x})) \quad (6.15)$$

3 Sample

$$y_t = f(\mathbf{x}_t) + \epsilon_t \quad (6.16)$$

 Perform Bayesian update to obtain μ_t and σ_t ;

6.3.3 Negative Lower Confidence Bound

The last acquisition function presented in this section is the negative lower confidence bound. This acquisition function is equivalent to the upper confidence bound, but instead of maximising one would be minimising. Both acquisition functions work by computing a lower/upper confidence bound for a given input. By inspection of the equation 6.17, it is possible to notice that this acquisition function simply adds a squared coefficient multiplying the variance. The lower confidence bound algorithm is shown in 6.4.

$$\alpha_{LCB}(\mathbf{x}) = \mu_{t-1}(\mathbf{x}) + \sqrt{\beta_t \sigma_{t-1}(\mathbf{x})} \quad (6.17)$$

In the next section, a common and well understood technique of sensitivity analysis is presented. In this work, this technique is used to understand how the different parameters related to one another and affect the models output.

Algorithm 6.4: Negative Lower Confidence Bound

Data: D , Gaussian process prior, $\mathcal{GP}(0, k)$

Result: Write here the expected result

1 **for** $t = 1, \dots, T$ **do**

2 Choose

$$\mathbf{x}_t = \underset{\mathbf{x} \in D}{\operatorname{argmax}} -\alpha(\mathbf{x}) = \mu_{t-1}(\mathbf{x}) + \sqrt{\beta_t} \sigma_{t-1}(\mathbf{x}) \quad (6.18)$$

 Sample

$$y_t = f(\mathbf{x}_t) + \epsilon_t \quad (6.19)$$

 Perform Bayesian update to obtain μ_t and σ_t ;

6.4 Sensitivity Analysis

In this section, the goal is to present Sobol's indices as a tool of global sensitivity analysis. Sobol's indices is a variance-based method that is already accepted by the industry and considered robust and general. Sobol's indices also work well with surrogate models [136]. In this discussion, the system of interest (a surrogate) is analysed from a probabilistic perspective. In this way, the model input is treated as a random vector $\mathbf{x}$ that is characterized by a joint probability density function (PDF) $f_{\mathbf{x}}$ with support $I_{\mathbf{x}}$. Hence, this stochastic model may be represented as

$$\mathbf{y} = \hat{\mathcal{M}}(\mathbf{x}) \quad (6.20)$$

This model has a joint probability distribution function $f_{\mathbf{y}}$ that is not known before the propagation of uncertainties. For simplicity it is assumed that the interesting quantity (IQ) is a scalar value and that the input parameters are composed by independent and identically distributed (iid) uniform parameters, x_i , in the interval $[0, 1]$, the Hoeffding-Sobol decomposition [245] is given by

$$y = \mathcal{M}_0 + \sum_{i=1}^n \mathcal{M}_i(x_i) + \sum_{1 \leq i < j \leq n} \mathcal{M}_{ij}(x_i, x_j) + \dots + \mathcal{M}_{1\dots n}(x_1, \dots, x_n) \quad (6.21)$$

where

$$\mathcal{M}_0 = \mathbb{E}[y] \quad (6.22)$$

$$\mathcal{M}_i(x_i) = \mathbb{E}[y \mid X_i] - \mathcal{M}_0 \quad (6.23)$$

$$\mathcal{M}_{ij}(x_i, x_j) = \mathbb{E}[y \mid x_i, x_j] - \mathcal{M}_0 - \mathcal{M}_i - \mathcal{M}_j, \quad (6.24)$$

This term $\mathcal{M}_0$ is the mean value. The terms of increasing order are conditional expectations that are defined in a recursive way characterising an unique orthogonal decomposition of the model response [124]. Thus, it is possible to decompose the total variance of the response as follows

$$\text{Var}(y) = \sum_{\mathbf{u}} \text{Var}(\mathcal{M}_{\mathbf{u}}(\mathbf{x}_{\mathbf{u}})) \quad (6.25)$$

where $\text{Var}(\mathcal{M}_{\mathbf{u}}(\mathbf{x}_{\mathbf{u}}))$ expresses the conditional variance for the sub-vector $\mathbf{x}_{\mathbf{u}}$, containing the variables which indices are indicated by the subset $\mathbf{u}$. The total variance itself is given by

$$S_{\mathbf{u}} = \frac{\text{Var}(\mathcal{M}_{\mathbf{u}}(X_{\mathbf{u}}))}{\text{Var}(Y)} \quad (6.26)$$

As a result of this equation it is possible to verify that, for $\mathbf{u} \subset \{1, \dots, n\}, \mathbf{u} \neq \emptyset$,

$$\sum_{\mathbf{u}} S_{\mathbf{u}} = \sum_{i=1}^n S_i + \sum_{1 \leq i < j \leq n} S_{ij} + \dots + S_{1\dots n} = 1 \quad (6.27)$$

that is, by construction the sum of all the Sobol' indices must be equal to the unit.

The terms

$$S_i = \frac{\text{Var}(\mathcal{M}_i(X_i))}{\text{Var}(Y)}, \quad i = 1, \dots, n \quad (6.28)$$

are called the first order Sobol' indices for the single variable X_i and denote the individual effect of X_i for the total model variate. Similarly,

$$S_{ij} = \frac{\text{Var}(\mathcal{M}_{ij}(X_{ij}))}{\text{Var}(Y)}, \quad 1 \leq i < j \leq n \quad (6.29)$$

are the second-order indices that contemplate the effect of the interaction between the variables X_i and X_j . To measure the full contribution of the i -th random variable X_i for the total variance either by its single effect or by its interaction with others, one

can use the total Sobol' indices, which are defined by

$$S_i^T = \sum_{\mathbf{u} \subset \{1, \dots, n\}} S_{\mathbf{u}} \quad i = 1, \dots, n \quad (6.30)$$

The algorithm 6.5 illustrates how Sobol indices may be implemented in a numerical code.

Algorithm 6.5: Sobol indices

- 1 Create two sample matrices A and B.
- 2 Total number of model evaluations is $2M(1 + p)$. Estimate the Sobol indices.
The first-order Sobol indices are approximated by

$$S_i \approx \frac{\frac{1}{M} \sum_{j=1}^M \left[f(A)_j f\left(B_A^{(i)}\right)_j - f(A)_j f(B)_j \right]}{\frac{1}{2M} \sum_{j=1}^{2M} f(C)_j f(C)_j - \mathbb{E}^2[f(C)]} \quad (6.31)$$

and the total Sobol indices are approximated by

$$S_{Ti} \approx \frac{\frac{1}{2M} \sum_{j=1}^M \left[f(A)_j - f\left(A_B^{(i)}\right)_j \right]^2}{\frac{1}{2M} \sum_{j=1}^{2M} f(C)_j f(C)_j - \mathbb{E}^2[f(C)]} \quad (6.32)$$

6.5 Results and Discussions

This section presents the robust optimisation problem based on the previous investigations. The algorithm 6.6 proposed is a modification of the algorithm proposed by Gelbart [246]. The main modification is a loop (Σ) that allows different load cases to be considered and evaluated. The algorithm used for constrained Bayesian optimisation is shown in figure 6.6. An arbitrary acquisition function is denoted by simply α , without any subscript. The constrained acquisition function is denoted by α_C . Mathematically, the full form of this acquisition function after the integration of the Gaussian process hyper-parameter is

$$a_C(\mathbf{x}) = \int \alpha(\mathbf{x} \mid \theta) p(\theta \mid \mathbf{D}) p(\mathcal{C}(\mathbf{x}) \mid \mathbf{x}, \mathbf{D}', \omega) p(\omega \mid \mathbf{D}') d\theta d\omega \quad (6.33)$$

In the case where the acquisition function α_C is not defined for some reason and one probabilistic constraint is violated, α can't be computed. Then, only the term on the right is calculated, thus

$$\alpha_C = \sum_l^L \prod_{k=1}^K \Pr(g_k(\mathbf{x})) \geq 0 \quad (6.34)$$

If the constraint in equation 6.34 is violated in all wing stations and load cases, the objective function is ignored. The optimiser will then try to satisfy the probabilistic constraint until all the constraints are satisfied.

Algorithm 6.6: Constrained Bayesian Optimisation Algorithm

Output: optimum $\mathbf{x}_i$

- 1 Place a Bayesian Gaussian Process prior on f
 - 2 Observe f at $n_0 > 0$ points
 - 3 Set $n = n_0$ and $\mathcal{D}_n = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$
 - 4 **for** $n = 1, 2, \dots, N$ **do**
 - 5 Compute the posterior $f \mid \mathcal{D}_n$ using:

$$p(f|\mathcal{D}_n) = \frac{p(\mathcal{D}_n \mid f)p(\mathcal{D}_n)}{p(\mathcal{D}_n)} \quad (\text{Bayes Theorem}) \quad (6.35)$$
 - 6 Compute the acquisition function $\alpha(\mathbf{x})$ based on the posterior.
 Let

$$\alpha_C(\mathbf{x}) = \alpha(\mathbf{x}) \Pr(\mathcal{C}(\mathbf{x})) = \alpha(\mathbf{x}) \sum_l^L \prod_{k=1}^k \Pr(\mathcal{C}_k(\mathbf{x})) \quad (6.36)$$
 - 7 Compute

$$\mathbf{x}_{n+1} = \arg \max_{\mathbf{x} \in \mathcal{X}} \alpha_n(\mathbf{x}) \quad (6.37)$$
 - 8 Observe

$$y_{n+1} = f(\mathbf{x}_{n+1}) + \epsilon_{n+1} \quad \text{with } \epsilon \sim \mathcal{N}(0, \eta^2) \quad (6.38)$$

$$\mathcal{D}_{n+1} = \mathcal{D}_n \cup \{(\mathbf{x}_{n+1}, y_{n+1})\} \quad (6.39)$$
 - 9 Increment n
 - 10 Return the point with the largest y_i
-

6.5.1 Single-Objective Robust Optimisation for a Single C_L

Finally, having presented the necessary tools, it is possible to present and solve the complete robust optimisation problem. As stated in chapter 3, the robust optimisation

problem that motivated this research can be stated as

For the models at a given C_L as:

$$\Delta(L/D) = \mathcal{GP}_{\text{aero}}(\bar{\theta}_1, \bar{\theta}_2, \overline{EI}_1, \overline{EI}_2, \overline{EI}_3, \overline{EI}_4, \overline{GJ}_1, \overline{GJ}_2, \overline{GJ}_3, \overline{GJ}_4, C_L) \quad (6.40)$$

$$\mathbf{SMT} = \mathcal{GP}_{\text{loads}}(\bar{\theta}_1, \bar{\theta}_2, \overline{EI}_1, \overline{EI}_2, \overline{EI}_3, \overline{EI}_4, \overline{GJ}_1, \overline{GJ}_2, \overline{GJ}_3, \overline{GJ}_4, \lambda) \quad (6.41)$$

Find

$$\mathbf{x}^* = \operatorname{argmax} f : \Delta \frac{L}{D}(C_L) \quad (6.42)$$

With respect to: $\bar{\theta}_1$ and $\bar{\theta}_2$

Subject to:

$$g_1 : [1 - \beta_{ij}^{\min}]^\top \cdot \mathbf{S} < \mathbf{S} < [1 + \beta_{ij}^{\max}]^\top \cdot \mathbf{S} \quad (6.43)$$

$$g_2 : [1 - \beta_{ij}^{\min}]^\top \cdot \mathbf{M} < \mathbf{M} < [1 + \beta_{ij}^{\max}]^\top \cdot \mathbf{M} \quad (6.44)$$

$$g_3 : [1 - \beta_{ij}^{\min}]^\top \cdot \mathbf{T} < \mathbf{T} < [1 + \beta_{ij}^{\max}]^\top \cdot \mathbf{T} \quad (6.45)$$

It is important to note that the objective function is denoted by f , that is

$$f : \Delta \frac{L}{D}(C_L) \quad (6.46)$$

The aerodynamic model is the following

$$\Delta(L/D) = \mathcal{GP}_{\text{aero}}(\bar{\theta}_1, \bar{\theta}_2, \overline{EI}_1, \overline{EI}_2, \overline{EI}_3, \overline{EI}_4, \overline{GJ}_1, \overline{GJ}_2, \overline{GJ}_3, \overline{GJ}_4, C_L) \quad (6.47)$$

6.5.2 Acquisition Functions

The optimisation was performed using the algorithm 6.7 for a number of acquisition functions which include expected improvement, probability of improvement and negative lower confidence bound for 1000 iterations. The results are shown in figure 6.1. The lower confidence bound (LCB) acquisition function performed very poorly and was not capable of identifying any feasible region. This behaviour of the LCB acquisition function is explained by the fact that LCB, as other acquisition functions, contain

terms for explicit exploitation $\mu(\mathbf{x})$ and exploration $\sigma(\mathbf{x})$, as equation 6.17 states. With LCB, the exploitation vs. exploration trade-off is straightforward and easy to tune via the parameter β_t . This means that when β_t is small, the Bayesian optimiser(BO) will favour solutions that are expected to have high β_t . On the contrary, when is large, BO rewards the exploration of currently uncharted areas in the search space. For the robust optimisation problem dealt in this work it was not possible to tune this parameter in a sensible way. Thus, this acquisition function performed poorly. The expected improvement and the probability of improvement acquisition functions, however, performed the best and were able to reach the expected aerodynamic improvement of $\Delta(L/D) = 0.03$.

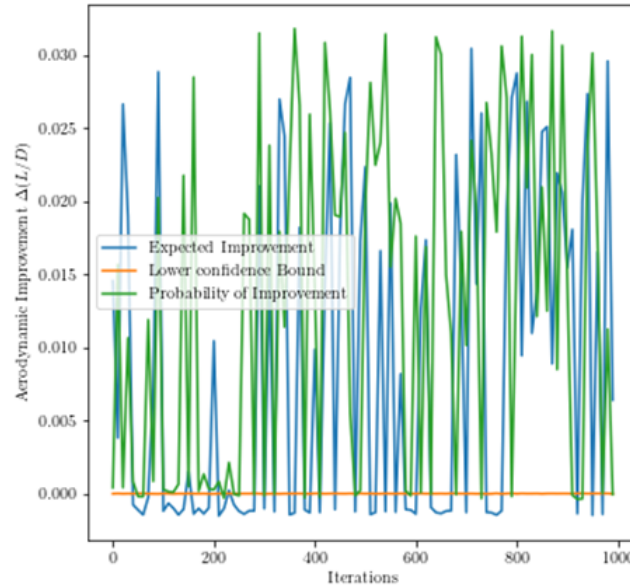


Figure 6.1: Performance of acquisition functions

6.5.3 Sobol Indexes for Sensitivity Analysis

To investigate the synergy of the parameters as stated in the introduction, Sobol indexes are employed to a point close to the optimal. Sobol indexes were computed by the algorithm 6.5. The results are shown in figures 6.2 and 6.3. Figure 6.2 shows that the most relevant parameters are the wing jig twist. This was already expected, since the bending and torsional stiffness are engineering parameters set out arbitrary by the engineer. The measure of the full contribution of random variable θ_1 is given by

figure 6.3. As it is possible to observe both Sobol indexes are somewhat similar. The

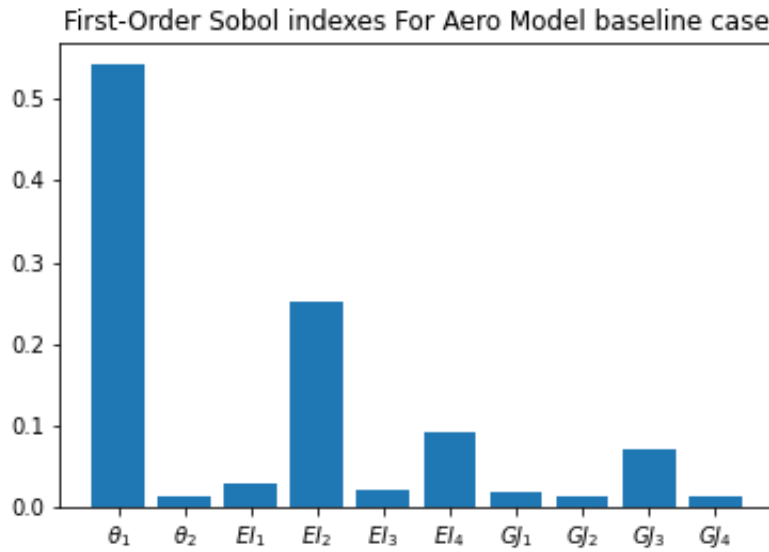


Figure 6.2: Sobol indexes

most relevant parameter is θ_1 which corresponds to the wing twist angle at the root.

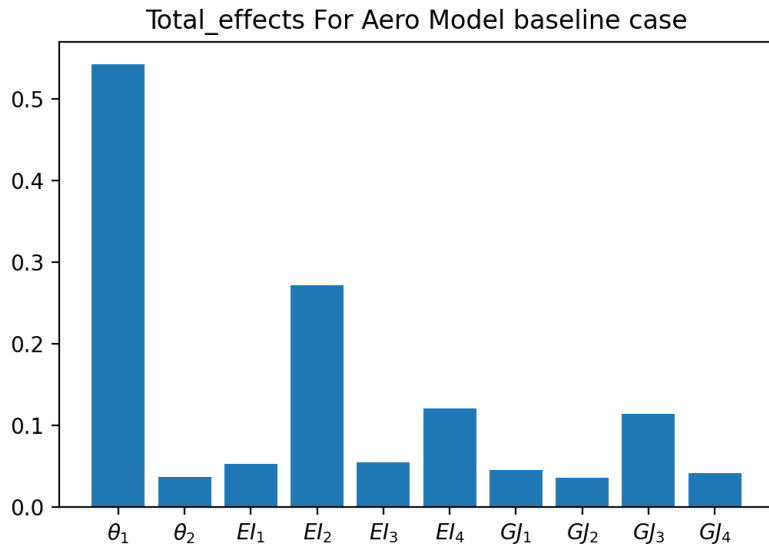


Figure 6.3: Sobol index total effects

6.5.4 Robust Optimisation

The robust optimisation final plot is shown in figure 6.4, where the colours are intended to give a sense of depth in the 3D plot. The acquisition function chosen for

this optimisation procedure was probability of improvement. The criteria adopted is based on an inspection of figure 6.1, which shows the behaviour of different acquisition function during 1000 iterations. Due to its poor performance, the lower confidence bound was immediately rejected. The choice between probability of improvement and expected improvement was based on the maximum value obtained during the optimisation and in the number of “peaks”, which suggests that for this particular problem the probability of improvement is capable of achieving higher values, more frequently and with less iterations than expected improvement.

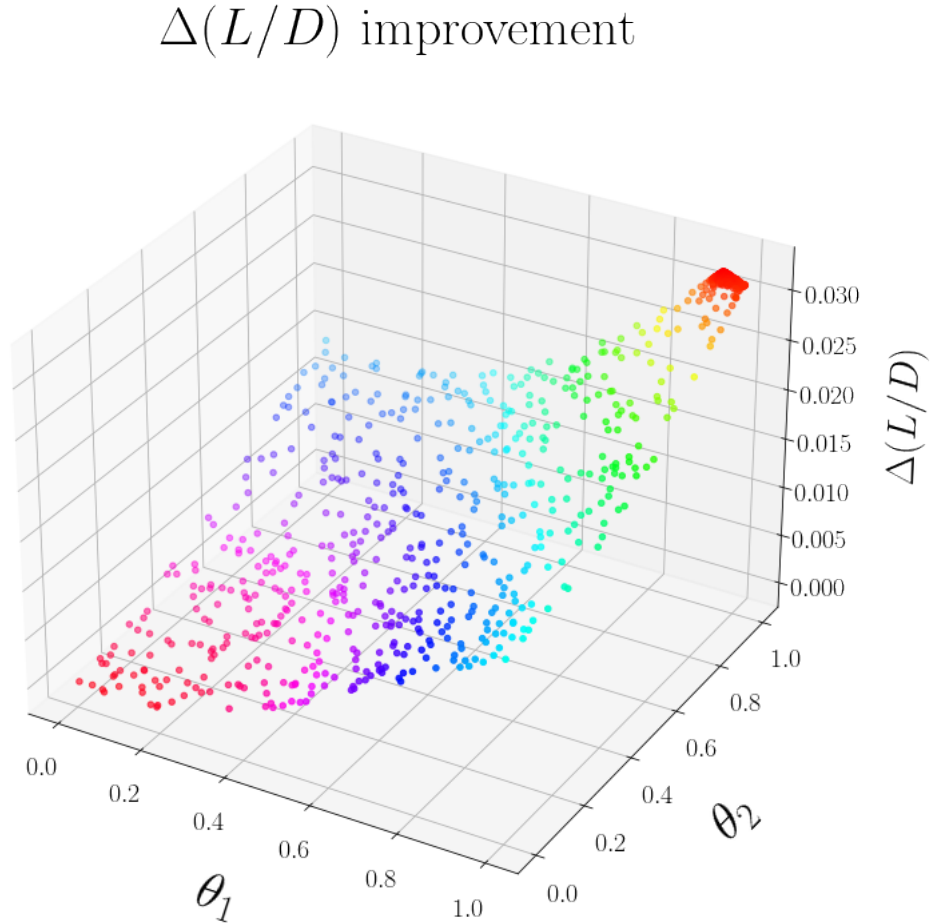


Figure 6.4: Robust optimisation of the wing aerostructural response using probability of improvement acquisition function

It is possible to conclude that it is possible to achieve a aerodynamic improvement

of $\Delta(L/D) = 0.03$ in the lift coefficient for a baseline configuration taking into account manoeuvre and gust loads with design load margin constraints of 2% on the root and 1% on the tip. As one can observe, the more aerodynamic improvement expected the less risk in achieving it in the configuration studied. Figure 6.5 demonstrates the iterations

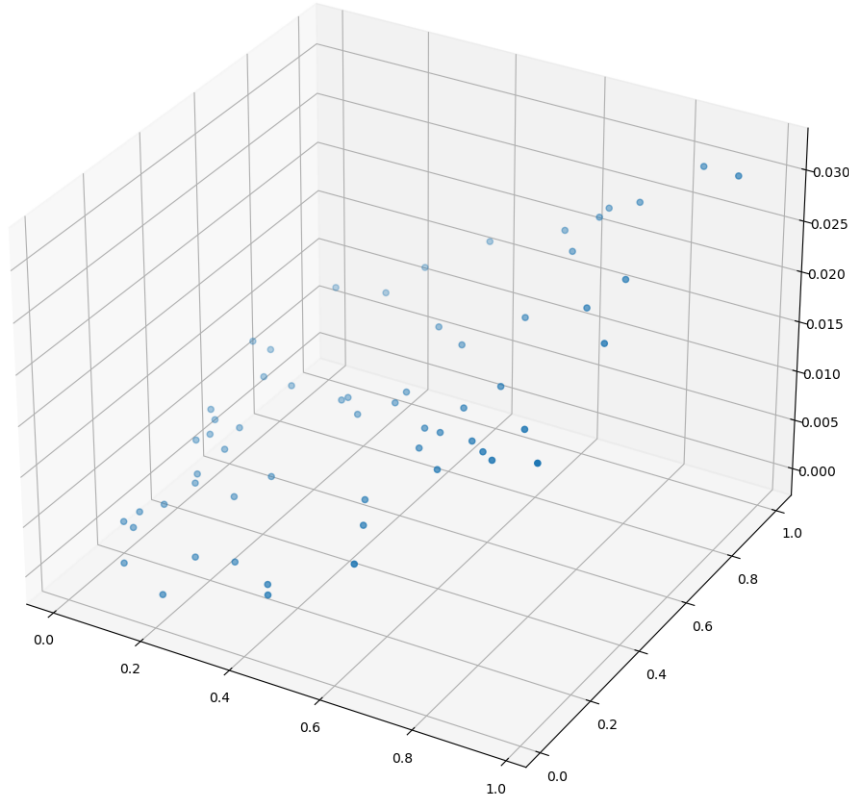


Figure 6.5: Points that violated the design load constraints

where the design load constraints were violated for the baseline case investigated. The code is able to print out in which wing stations the load was excessive. Normally, the violation happens at just one wing-station. In this cases, the optimiser is able to calibrate and compensate for points nearby that satisfy the constraints. Interesting enough, the optimiser can be thought as working over a surface that is the result of the latin hypercube experimental design that is used to feed the optimiser with information about the design space, before the first iteration of the optimiser. Each point correspond to a quantity that is a tuple $\Delta L/D, \theta_1, \theta_2$ with its uncertainties quantified. Thus, it is necessary to observe figures 6.6 and 6.7 to inspect the uncertainties.

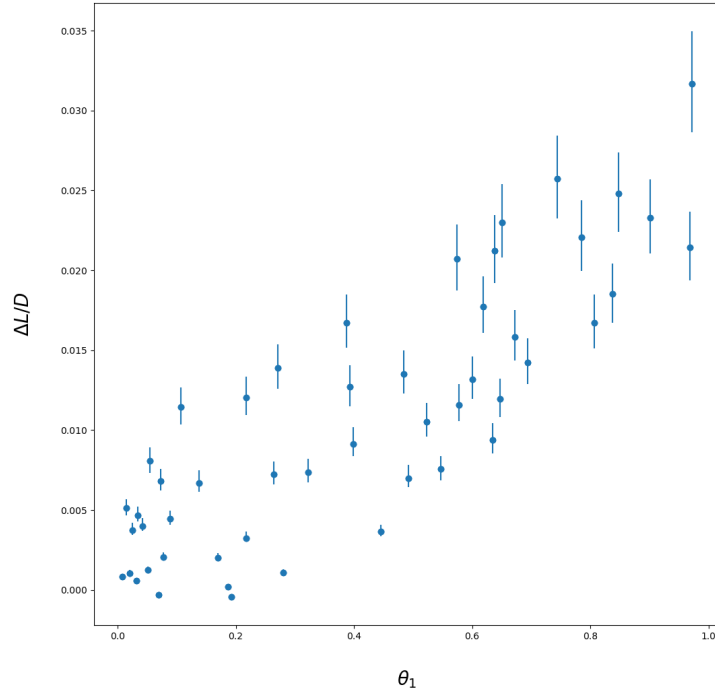


Figure 6.6: θ_1 versus $\Delta(L/D)$

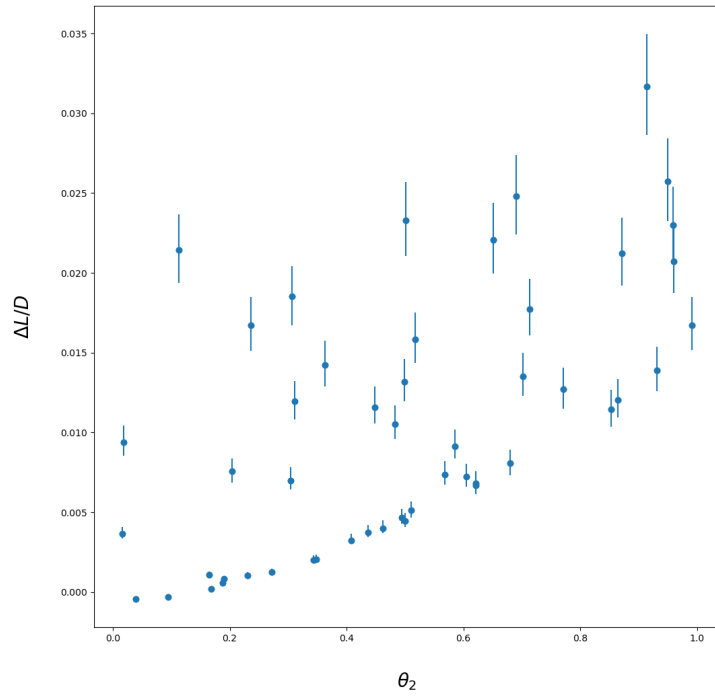


Figure 6.7: θ_2 versus $\Delta(L/D)$

6.5.5 Advantages and Limitations of the Developed Framework

As it was stated in earlier chapters, robust optimisation is a branch of optimisation theory that attempts to find an optimum robust solution, rather than a nominal one. In current practice, one may consider to use a robust optimisation approach when solving problems where robustness and reliability are more important than nominal values. This usually happens when it is not possible or practical to have an analytical mathematical expression or model that gives robustness and reliability metrics with confidence. In such cases, it is not recommended to perform a robust optimisation with a deterministic setting.

There are engineering problems where the uncertainty is presented in the assumptions of the engineering model. This is common in problems involving quasi-random operating conditions, manufacturing processes and materials aging which introduce deviations from the specifications and the operating conditions. For such problems, it is more suitable to approach these problems by including random parameters representing the known uncertainty in the design and in the control variables of the optimisation problem.

An advantage of the adoption of Bayesian methods versus the current practice (deterministic) is the capability to include uncertainties that are presented in the engineering problem as “degrees of belief”. Usually, Bayesian robust optimisation methods require the construction of a robust counterpart to an optimisation problem formulated in a deterministic way. Thus, one can say that both methods are complementary to each other and are designed to solve slightly different problems. Another advantage of Bayesian methods is the fact that they are capable of given the nominal parameters as well, for situations, where the statistical mean can be taken as such.

One of the limitations of the developed Bayesian robust optimisation framework developed is the number of objective functions it can work it. **Currently, it can work with two objective functions only. Another limitation is the number of design variables which is also two.** The reason for these limitations in the number of design variables lies in the Gaussian process that the optimiser uses which

does not converge when more design variables are used. It is important to highlight that these are limitations related to coding and not, in principle, to the method. The kernel of Gaussian processes tend to be extremely reliable and work well in the current implementation with at least 30 dimensions. The biggest limitation lies in the memory because each model stored in the computer takes approximately 62 *Mb*.

6.6 Summary and Conclusions

In this chapter, the investigation on the robust optimisation problem was concluded. The topic of Bayesian optimisation was presented. The results obtained in the previous chapters were used to develop a solution that is capable of optimising the baseline configuration and achieving an improvement of $\Delta(L/D) = 0.03$ in the lift coefficient for a baseline configuration taking into account manoeuvre and gust loads with design load margin constraints of 2% on the root and 1% on the tip.

Chapter 7

Conclusions and Recommendations

7.1 Conclusions

In this thesis, different formulations of Gaussian processes and their applications in uncertainty quantification and robust optimisation were investigated with the goal of answering the problem set out in chapter 1:

How to perform the robust optimisation of wing twist (jig shape) to balance aerodynamic performance and risk of exceeding the design loads margins in the presence of structural uncertainties? (Chapter 6)

In order to properly address the robust optimisation problem with all the necessary rigour, the research was presented in five chapters, which answers five questions:

- What is the state-of-art of uncertainty quantification and robust optimisation methods? (Chapter 2)
- How can one formulate the robust optimisation problem using Bayesian probabilistic modelling (Chapter 3)
- How can one rapidly evaluate flight loads, allowing uncertainty quantification and robust optimisation? (Chapter 4)
- How can one rapidly and **scalably** evaluate flight loads, allowing uncertainty quantification and robust optimisation? (Chapter 5)

Chapter 2 presented an overview of the state-of-the-art of uncertainty quantification and robust optimisation methods. In particular, the chapter focuses on methods that are compatible with the Bayesian paradigm, such as Gaussian process, multi-task Gaussian process and deep Gaussian processes. The Bayesian approach, where the previous knowledge is stated by means of a prior function and the state of knowledge is described by a “degree of belief” instead. This, in turn, allows the creation of multi-level probabilistic models that are able to compute the solution in terms of a posterior distribution, which is parameterised by its mean and variance. Another field that requires particular attention is digital transformation. Gaussian processes are

particularly useful for aeronautical applications as opposed to alternative machine learning formulations, because continuous systems as physical models tend to possess a Gaussian distribution, as data becomes increasingly available. It was concluded that a better general understanding of how to build probabilistic models is beneficial and potentially has many applications in aeronautics. This, in turn, allows one to develop industrial probabilistic models that use surrogate models to speed-up time-consuming computations. Hence, a myriad of difficult problems can be approached, extending the current engineering capability. Thus, the first objective set out in the introduction of survey the probabilistic regression, uncertainty quantification and robust optimisation methods was concluded.

In chapter 3, it was shown that the Bayesian approach offers a more intuitive approach that can be applied recursively through the use of hierarchical Bayesian modelling, when contrasted to the frequentist (non-Bayesian) approach. Based on the concepts presented in the chapter, the robust optimisation problem was formulated. As discussed in the chapter, this is by no means a trivial problem, because the dataset structure tends to generate some level of confusion during the formulation. Nevertheless, the seven step heuristics presented allows one to convert a deterministic problem into a probabilistic one and is simple enough to be understood and adopted by the industry. Thus, the second objective set out in the introduction was concluded.

In chapter 4 the use of Gaussian processes as surrogate models for the robust optimisation problem set out in the introduction for a variety of manoeuvre and gust loads was documented. This chapter also documents the necessity of employing such models in an aircraft programme. In particular, its application in flight physics activities. The models developed are capable of modelling the flight loads with a high degree of accuracy for the interesting quantities with quantified uncertainties. It was also shown that correlated parameters versus uncorrelated ones yield different results despite the same statistics metrics.

In chapter 5, the concept and use of sparse Gaussian process was presented. The common approximations were surveyed. It was highlighted that for conventional

Gaussian processes, the computational requirements grow cubically, while the memory grows with the square of the number of training points, n . Hence, the importance of sparse approximations which seek to solve this issue. Although the identification of the load cases were not disclosed by Airbus, it was possible to verify that the surrogate models are capable of capturing a variety of conditions and phenomena. Moreover, despite the drawbacks of the FITC sparse formulation adopted, it can be concluded that they are superior in performance to the conventional Gaussian process. Thus, in future endeavours of aircraft engineering, specially for novel configurations, such approximations might offer a great extend in capability. Moreover, novel technologies such as digital-twin and the introduction of machine learning methods in aircraft certification are going to offer more opportunities to the employment of special Gaussian processes that are capable of more predictive power without sacrificing the performance.

Finally, in chapter 6 the results of previous chapter were used to completely solve the robust optimisation problem. It was possible to understand the synergies and inter-dependencies of a number of design parameters, such as the wing sectional twist, bending- and torsional stiffnesses on the overall aircraft lift-to-drag ratio using Sobol indexes. Moreover, the results obtained in the previous chapters were used to develop a solution that is capable of optimising the baseline configuration and achieving an improvement of $\Delta(L/D) = 0.03$, in the lift coefficient for a baseline configuration taking into account manoeuvre and gust loads with design load margin constraints of 2% on the root and 1% on the tip

7.2 Recommendations for Future Work

The aim of this section is to provide future research directions based on the issues addressed throughout the thesis and point out future possible directions in the context of the so-called Digital Transformation.

It is recognised that despite the advances in Gaussian Process, in particular, in sparse approximations and scalable methods, the GP models are in computational disadvantage when compared to neural networks. This disadvantage results from the

necessity in all Gaussian processes of inverting a matrix, which is usually accomplished by Cholesky decomposition. Therefore, it is necessary to investigate the consequences of more efficient matrix inversion algorithms in the inference procedure used by GPs and its implications for its use in engineering. An important alternative to Gaussian process as emulators are neural networks. However, the applications in aerospace engineering are limited due to the fact that most neural networks operate on the black-box assumption. This issue may be mitigated using a physically informed convolutional neural network [247, 248]. This type of convolutional network is powerful enough to model complex systems with the advantage that they are not completely disassociated from the physical system which it intends to model. Another advantage that is worth mentioning is its flexibility, which is enough to model the dependence between input parameters and output parameters.

One very interesting application of Gaussian process that has research merit for aeronautics is the Gaussian process latent variable model (GPLVM). GPLVMs are useful for high-dimensional data problems in which the goal is to learn the low dimensional underlying patterns from observations. GPLVMs combine characteristics of a Gaussian process with a latent variable model (LVM). A LVM is a statistical model that relates a set of variable to a set of latent variables by assuming that the responses (or output) is controlled by them. Thus, these latent variables can be considered as functional noise-free version of the observed variables. GPLVM models the functional variables by a GP trained with low dimensional latent variables. In this way, the inference of the latent variables is possible from data. A quick overview of the mathematics behind a GPVLM is contained in the appendix ?? and a comprehensive review of GPLVMs is provided by Li and Chen [249].

Future research may also focus on investigating few-shot learning (FSL). Few-shot (or low-shot) learning (LSL) is a machine learning method designed to work with training datasets that contain limited information. This does not only reduces the data collection effort, but also reduces computational costs. Normally, ML algorithms improve their predictive performance with higher volumes of data. FS methods may

be useful in aeronautical design and high-value manufacturing, especially when applied to novel experimental configurations where the data is sparse.

A very interesting idea and potential field for future research is causal Bayesian optimisation, i.e, the seeking global optimal solutions of a problem where a causal model is available. In causal optimisation problems, a sequence of interventions may be performed according to the causal relations stored in a causal graph. This method is very rich in applications, such as operational research, drug discovery, biology, but it is particularly important in aeronautics, where usually the causal relations are widely known given the maturity of the field and the knowledge of physical equations that govern the flight. Causal Bayesian optimisation take advantage of such structure to improve the optimisation process in terms of accuracy and computational cost and is based on the do-calculus developed by Pearl [250]. The do-calculus is a effective way to perform interventions in probabilistic model given that causal information is available. Thus, Causal Bayesian optimisation balances the two classical trade-offs of conventional optimisation, the exploration and exploitation, with observation-intervention, which is operationalise through do-calculus. figure 7.1 shows a potential causal graph for the problem treated in this thesis.

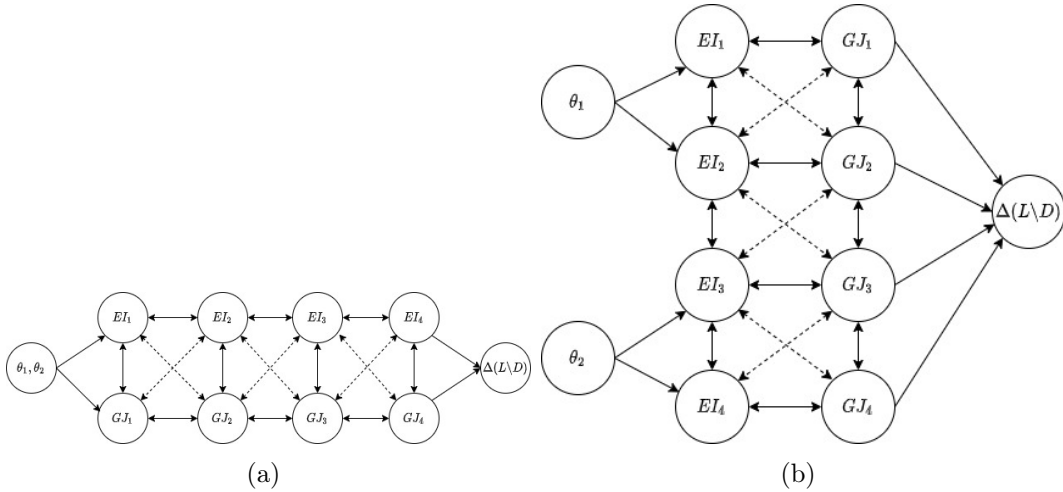


Figure 7.1: Causal information stored in a graph data structure

In figure 7.1, a causal Bayesian optimiser would try to reach the global optimum by firstly performing a sequence function queries, where all variables are manipulated.

This manipulation happens in the sense of a probabilistic intervention in the sense set by Pearl [251], which effectively does not break the statistical/probabilistic dependency, resulting from the causal dependency, that exists among θ_1, θ_2 and EI_1, EI_2, EI_3, EI_4 and GJ_1, GJ_2, GJ_3, GJ_4 . In causal Bayesian optimisation depending on the causal relationships between variables, the intervention on a subgroup of the graph may lead to better results and faster convergence to the optimum solution.

This work has focused on one sensitivity analysis method - Sobol indexes. However, further research should investigate other methods such as and Fourier amplitude sensitivity testing, delta moment independent measures, derivative-based global sensitivity measure and high dimensional model representation. These methods should be compared especially in terms of their scalability and their capability to support digital-twin technologies.

A very rich field for future research is the Model-Based Bayesian Engineering framework presented in the appendix. It subsumes the previous chapters into a model-based scheme designed to interconnect system-level parameters with subsystem. This Bayesian framework even with its limitations is a potential candidate that is needed to move to robust and scalable digital twin representation. The employment of directed multi-graphs equipped with a compositional relation provides an alternative formulation to the probabilistic graph factor model suggested for the synthesis of a digital-twin.

As an abstraction layer of the real yet-to-be-materialised system, it consists of a set of mathematical expressions and operations written in a graph-theoretical way that allows compositional relations. This may be used to model the subsystems as a set of coupled dynamical systems within a system, that evolves over time-steps or iterations. However, currently, the adoption of directed multi-graphs embedded in a tree data structure to model the system-subsystem interaction raises many questions and opens several paths for further investigation. The underlying formal background of this framework corresponds to a hierarchical (or multi-level) joint probability distribution that allows one to draw from well-established methods and techniques in Bayesian analysis. Ideally, this includes the capability of performing sampling, design of experiments,

uncertainty propagation, dimension reduction, sensitivity analysis and high-dimensional visualisation. Another aspect of such framework that should be investigated is its capability of working with surrogates or emulators based on different libraries. Ideally, this should not be an issue since the framework operates based on input-output relation. However, different size of data may result in difficulties during the coupling, which may be approached using data fusion methods. To further improve the framework, different algorithms may be proposed that can exploit the intrinsic structure and pattern in each system and its subsystems, for instance, in aerospace systems. This structure may benefit from a myriad of theorems, results, and techniques in the field of network theory and optimisation. Moreover, the surrogate models or emulators employed in the framework to guarantee the capability of uncertainty quantification and management can be expanded to include different techniques, for instance, state machines. In effect, such addition would connect the proposed Bayesian engineering framework with cyber-physical systems, a step towards an executable model within the and digital twin technology.

A carefully designed engineering framework that operates under the Bayesian paradigm potentially is a key-enabler for the synthesis and simplification of a variety of technologies being developed. Normally, it is accepted that highly complex systems tend to correlate with higher maintenance and development costs, which may be even prohibitive from different perspectives, such as managerial and engineering ones. Thus, one can safely affirm that progress in engineering and science may also happen in the opposite direction – by simplification. Therefore, the investigation of frameworks that subsume and incorporate several concepts and methods in its mathematical formulation is highly desirable. Moreover, this is the key to avoid the huge risks associated with gigantic computational frameworks being currently developed. Such computational frameworks compensate the lack of underlying mathematical assumptions with software and tools coupling, hugely increasing the risks associated with managerial and software integration challenges.

Another research direction that this work has identified is the possibility of de-

veloping executable models. An executable model, by definition, is a model that can be run, tested, debugged and measured for performance. Thus, an executable model is a key-enabler technology for model-driven architecture (MDA). The Bayesian framework built on the thesis content is a step towards MDA, since it has the potential of connecting system-level mathematical expressions with subsystem-level. However, this is preliminary work, and a number of questions remain without a clear answer. One of these questions is how can we associate each system and subsystem with a graphical language such as UML and SysML. The answer must be in the form of an metamodel description defining the logical rules that effectively extend the UML/SysML metamodel as defined by the Object Management Group (OMG) and a compiler that reads the model and produces an executable verifiable model based on it.

The cyber-physical system above mentioned as a potential extension of the ideas contained in this thesis and the proposed framework would have the additional feature of combining multi-domain and multi-physics capabilities in a relatively lighter framework than the approach of gigantic frameworks currently being undertaken. This is due to the fact that it would such framework must operate on the premise of mathematical primitives, in which all the complexity is built upon. This approach contrasts with the tendency of building bigger software systems in large-scale projects involving several stakeholders that are prone to too many project and management issues. Further expected advancements in digital-twin technology converges with the executable modelling and graphical description language path delineated. Since the proposed technology is idealised to be used in collaborative tasks, issues and problems in project management and its application in engineering design must be taken as well. In addition to that, there are a number of information technology (IT) issues that cannot be overlooked. For instance, such framework may be part of an client-server application where a master model containing all the different models and version control is located in the main server and each stakeholder corresponds to a client. This is also may be used with the idea of AMBSE as reviewed in the chapter 2. On it different designs may be developed, verified and tested and frequent computational checks and data

may be gathered to monitor the project progress along with the uncertainty associated with each process, allowing better in-run allocation of resources and time. As a consequence, complex tasks such as technology portfolio allocation and optimisation may be performed under a distributed risk basis.

Regarding future research directions in the application and widespread adoption of machine learning methods in aeronautics, such as the ones used in this thesis, the following recommendations for future research are made. These recommendations for future research cover datasets, improvement of accuracy and robustness of ML applications to aerospace, safety assurance in design/analysis, runtime assurance, ML/AI Interpretability, and the use of ML in Safety-Critical Applications. The need is urgent as AI/ML technologies are rapidly being deployed in other applications, driving down the cost of implementations and increasing the confidence in these technologies. Traditional aerospace industry stakeholders are eager to reap the potential benefits of ML, such as design-oriented MBSE and its companion Digital-Twins, while new applications such as hybrid/electric aircraft, unmanned systems and urban mobility are driving the desire to transition these technologies from other domains into aerospace.

The main work of this thesis adopts a data-driven approach to robust optimisation of a relevant industrial problem in the aeronautical industry. In the paradigm of the fourth industrial revolution, the data set is expected to be a signification part of the requirements specification and in some cases even being the requirements itself. Very often, too many efforts are necessary in the process of data cleansing, a fact that not only is time-consuming but prone to many errors. Thus, a very important direction of research concerns the methods and ways to process datasets among stakeholders in collaborative projects, where the dataset must be consistent with the requirements. These requirements must refer data types, expected ranges, format and structure. This is a step necessary to any model-based engineering project supported by future digital twin solutions and must not be overlook. Thus, dataset management and curation in aeronautical projects is a crucial area for research. Much can be gained by incorporating principles of software engineering, DevOps and continuous improvement and integration

approach to improve the data handling.

Appendix A

Miscellaneous Material

A.1 Mathematical Background

A.1.1 Means and variances of conditional distributions

The mean μ and the variance σ of a random variable u can be expressed as the conditional mean and variance given a related quantity v [205]. The mean of the random variable u may be obtained by the average of the conditional mean over the marginal distribution of v , that is

$$\mathbb{E}(u) = \mathbb{E}(\mathbb{E}(u | v)) \quad (\text{A.1})$$

where the inner expectation averages over u , conditional on v , and the outer expectation averages over v . The equation A.1 can be derived by writing the expectation of the joint distribution of u and v and refactoring:

$$\mathbb{E}(u) = \iint up(u, v)dudv = \iint up(u | v)dvp(v)dv = \int \mathbb{E}(u | v)p(v)dv \quad (\text{A.2})$$

The resulting variance has two terms, the mean of the conditional variance and the variance of the conditional mean, respectively:

$$\text{var}(u) = \mathbb{E}(\text{var}(u | v)) + \text{var}(\mathbb{E}(u | v)) \quad (\text{A.3})$$

This result can be derived by expanding the terms on the right side of equation A.3:

$$\begin{aligned} \mathbb{E}(\text{var}(u | v)) + \text{var}(\mathbb{E}(u | v)) &= \mathbb{E}(\mathbb{E}(u^2 | v) - (\mathbb{E}(u | v))^2) + \mathbb{E}((\mathbb{E}(u | v))^2) - (\mathbb{E}(\mathbb{E}(u | v)))^2 \\ &= \mathbb{E}(u^2) - \mathbb{E}((\mathbb{E}(u | v))^2) + \mathbb{E}((\mathbb{E}(u | v))^2) - (\mathbb{E}(u))^2 \\ &= \mathbb{E}(u^2) - (\mathbb{E}(u))^2 \\ &= \text{var}(u) \end{aligned} \quad (\text{A.4})$$

The expressions A.2 and A.3 also hold if u is a vector, in the case where $\mathbb{E}(u)$ is a

vector and $\text{var}(u)$ is a matrix.

A.1.2 Continuous distributions

The table A.1 introduce the notation, the probability density functions, the parameters, the means, modes and standard deviations for some analytical probability distributions. These are useful because realistic distributions for complex multivariate models containing hierarchical and mixture models may be constructed using these probability functions as building blocks.

Distribution	Notation	Parameters
Uniform	$\theta \sim U(\alpha, \beta)$ $p(\theta) = U(\theta \mid \alpha, \beta)$	boundaries α, β with $\beta > \alpha$
Normal	$\theta \sim \mathcal{N}(\mu, \sigma^2)$ $p(\theta) = \mathcal{N}(\theta \mid \mu, \sigma^2)$	location μ , scale $\sigma > 0$
Lognormal	$\theta \sim \text{lognormal}(\mu, \sigma^2)$ $p(\theta) = \text{lognormal}(\theta, \mu, \sigma^2)$	location μ , log-scale $\sigma > 0$
Multivariate normal	$\theta \sim \mathcal{N}(\mu, \Sigma)$ $p(\theta) = \mathcal{N}(\theta \mid \mu, \Sigma)$ (implicit dimension d)	symmetric, positive definite, $d \times d$ variance matrix Σ

Table A.1: Continuous probability distributions

A.1.3 Random variables

A random variable is a mathematical quantity that can take any arbitrary value from a given set of values. Normally, in the Bayesian paradigm, a random variable is used to model a “degree of belief”. In this way, one can say that a particular random variables is distributed according to a probability distribution function. For instance, for a continuous random variable $\mathbf{x}$ to be normal or Gaussian it must have a probability distribution function of the form

$$f_{\mathbf{x}}(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2} \quad (\text{A.5})$$

where μ and σ are two scalar parameters characterising the Gaussian probability distribution function, with σ assumed non-negative.

A.1.4 Central Limit Theorem

The central limit theorem is a very important result in probability theory and in Bayesian methods. This theorem states that when independent random variables are summed up their sum tends toward the normal distribution, even in the case where these independent variables are not normally distributed themselves. This implies that statistical methods that work for normal distributions can be applicable to many problems involving other types of distributions. Formally, it may be stated as follows:

If $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_n$ are random samples drawn from a population with mean μ and variance σ^2 , and if $\bar{\mathbf{x}}_n$ is the sample mean of the first n samples, then the limit of the resulting distribution, Z , is

$$\lim_{n \rightarrow \infty} \left(\frac{\bar{\mathbf{x}}_n - \mu}{\sigma_{\bar{\mathbf{x}}}} \right) \quad (\text{A.6})$$

where $\sigma_{\bar{\mathbf{x}}} = \frac{\sigma}{\sqrt{n}}$ is a standard normal distribution.

A.1.5 Kullback-Leibler (KL) divergence

The Kullback-Leibler (KL) divergence is a mathematical way of measuring the matching between two probability distributions information theoretic terms Gibbs and Su [252]. In Bayesian statistics, this is very useful because it allows us to measure how close to the true distribution an approximation distribution really is. Mathematically, the KL divergence is given by

$$D_{\text{KL}(p||q)} = \sum_{i=1}^N p(x_i) \log \left(\frac{p(x_i)}{q(x_i)} \right) \quad (\text{A.7})$$

In the equation A.1.5, $q(x)$ is the approximated distribution and $p(x)$ is the true distribution. Thus, one can say that the KL divergence measures how close a given arbitrary distribution is to the true distribution. If they match, the result will be null, i.e, $D_{\text{KL}}(p||q) = 0$.

A.2 Gaussian Process Latent Variable Model

Gaussian process latent variable models (GPLVM) may be understood as a variant of Gaussian processes that can be trained in a supervised manner [253, 254]. GPLVMs assume that the feature space is a latent space with an unknown data distribution that is learned during the training phase. They can also be seen as a probabilistic variant of principal component analysis (PCA) as shown by Lawrence[253, 254]. However, if the function space is defined as non-linear space by a kernel function, the GPLVM model can be interpreted as a probabilistic non-linear PCA. In this case, it is assumed a standard Gaussian prior over the input space, that maximizes the log probability of the dataset $p(\mathbf{y} \mid \mathbf{X}, \theta)$ with respect to the input data likelihood $\mathbf{X}$.

The non-linearities associated with the kernel function makes the Latin space distribution intractable that is they can't be computed analytically. In this case, however, Lawrence[253, 254] demonstrated that this latent space distribution can be estimated using the expectation-maximisation algorithm, which returns the mode of the data distribution. Gaussian Process latent variable models are also useful for reconstructing input, which is useful it should be used in denoising tasks. GPLVMs assume access to the full kernel matrix this means starrng the entire training data set and therefore presuppose the inversion of a full-kernel $n \times n$ matrix. However, Lawrence[255] demonstrated that many sparse Gaussian approaches are compatible with Gaussian process latent variable models.

The adoption of a sparse approach to Gaussian process latent variable models yields a Maximum A Posteriori estimate of the latent space, which is prone to overfitting of the training dataset. This overfitting issue is approached by proposing a Bayesian approach [238]. This Bayesian approach consists at using a variational approach instead of a Maximum A Posteriori solution to the latent space. This however introduces intractabilities. The application of the pseudo points method by developed by Titsias

and Lawrence [209] for GPLVMs, yields the following optimisation bound:

$$\mathcal{F}(q) \geq \log \left(\int e^{\langle \log \mathcal{N}(\mathbf{y}_d | \alpha_d, \beta^{-1} I) \rangle_{q(\mathbf{x})} p(\mathbf{m}_d) d\mathbf{m}_d} \right) \quad (\text{A.8})$$

$$- \frac{\beta}{2} \text{Tr} \left(\langle K_{nn} \rangle_{q(\mathbf{x})} \right) + \frac{\beta}{2} \text{Tr} \left(K_{mm}^{-1} \langle K_{mn} K_{nm} \rangle_{q(\mathbf{x})} \right) \quad (\text{A.9})$$

Here, q is the variational distribution over pseudo points $\mathbf{m}$ and $\alpha = K_{nm} K_{mm}^{-1} \mathbf{m}$. The subscript d is used to denote each dimension of the features. Algorithm A.1 shows how GP-LVM works.

Algorithm A.1: Modelling with a GP-LVM

Data: A size for the active set d , a number of iterations T

- 1 Initialise $\mathbf{X}$ through PCA;
 - 2 **for** T iterations **do**
 - 3 Select a new active set using the IVM algorithm;
 - 4 Optimise with respect to the parameter of $\mathbf{K}$ using scaled conjugate gradients;
 - 5 Select a new active set;
 - 6 **for** Each point not in active set, j , **do**
 - 7 Optimise with respect to $\mathbf{x}_j$ using scaled conjugate gradients;
-

A.3 Model-based Bayesian Framework and Graph-based Optimisation

A.3.1 Introduction

In this section, the discussed probabilistic methods are generalise into a Model-Based framework that allows Uncertainty Quantification and Robust Optimisation of a system and its subsystems. Much of this work relates to digital twins and model-based systems engineering. In this section a combination of Bayesian methods and a Model-based framework that can be directly applied to engineering design is proposed. The aim is to demonstrate how system methods may be used coupled to model-based systems engineering and a design methodology to solve a complex multi-domain uncertainty engineering problem.

A.3.2 Bayesian Engineering framework

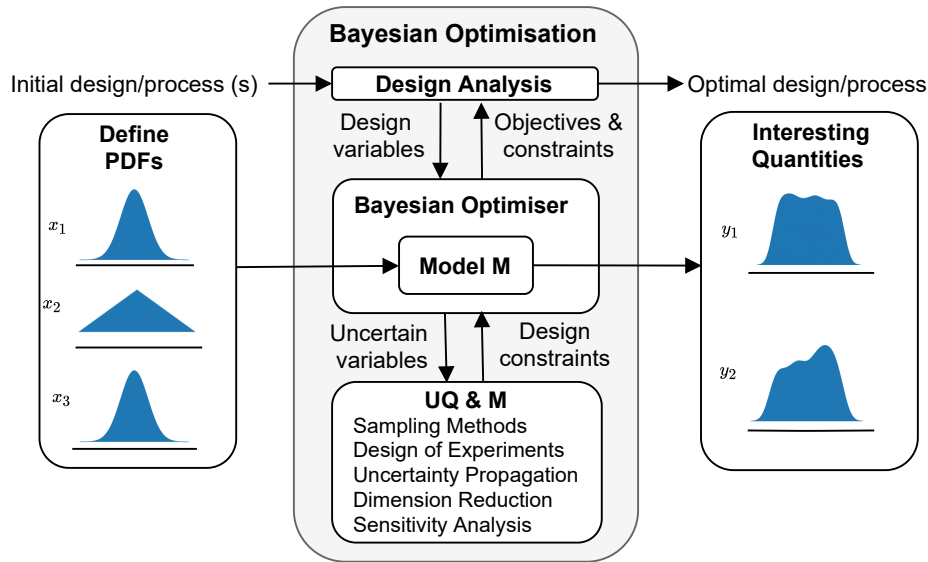


Figure A.1: The proposed Engineering Bayesian framework

The aim is to demonstrate how previous chapters' methods may be used coupled to model-based systems engineering and a design methodology to solve a complex multi-domain uncertainty engineering problem. To achieve this aim, it is interesting to start with some assumptions.

A.3.3 Assumptions

The entire framework as it is now may be stated as five *ad hoc* propositions:

1. The hierarchical system structure (or product structure) can be modelled as a tree data structure.
2. A subsystem can be represented mathematically as a directed multi-graph.
3. A hierarchical (or multi-level) Bayesian model may be modelled as a directed multigraph (see [256]) for demonstration.
4. A directed multigraph representing mathematically a subsystem can be embedded on the node of a tree data structure.
5. The directed multigraph is equipped with a compositional (functional) relationship.

For decomposable models, the multigraph can be used directly to factor the joint distribution of the contingency table in terms of the generators of the model. In particular, let M be the multigraph of a decomposable generating class, and let T be any maximum spanning tree with set $V(T)$ of vertices and set $B(T)$ of branches. Then, the joint distribution of a hierarchical model is

$$p(\theta_1, \dots, \theta_J) = \int \prod_{j=1}^J [N(\theta_j | \mu, \tau^2)] p(\mu, \tau) d(\mu, \tau) \quad (\text{A.10})$$

where $p(\theta_1, \dots, \theta_J)$ represents the probability associated with level θ_1 of the first factor, level θ_2 of the second factor and level θ_J of the J th factor.

A preliminary sketch of how a Bayesian framework operating with multigraphs can be constructed and applied is shown in figure A.2. The idea is to find a way to realise sizing and optimisation on system-level and on subsystem-level as well.

A.3.4 Notation

To differentiate between system and subsystem level description, the capital superscript is used in a given variable to denote a system-level variable. To denote a subsystem, a lower case letter is used instead. Random variables are denoted by $\tilde{\cdot}$ and emulators or surrogates are denoted by $\hat{\cdot}$.

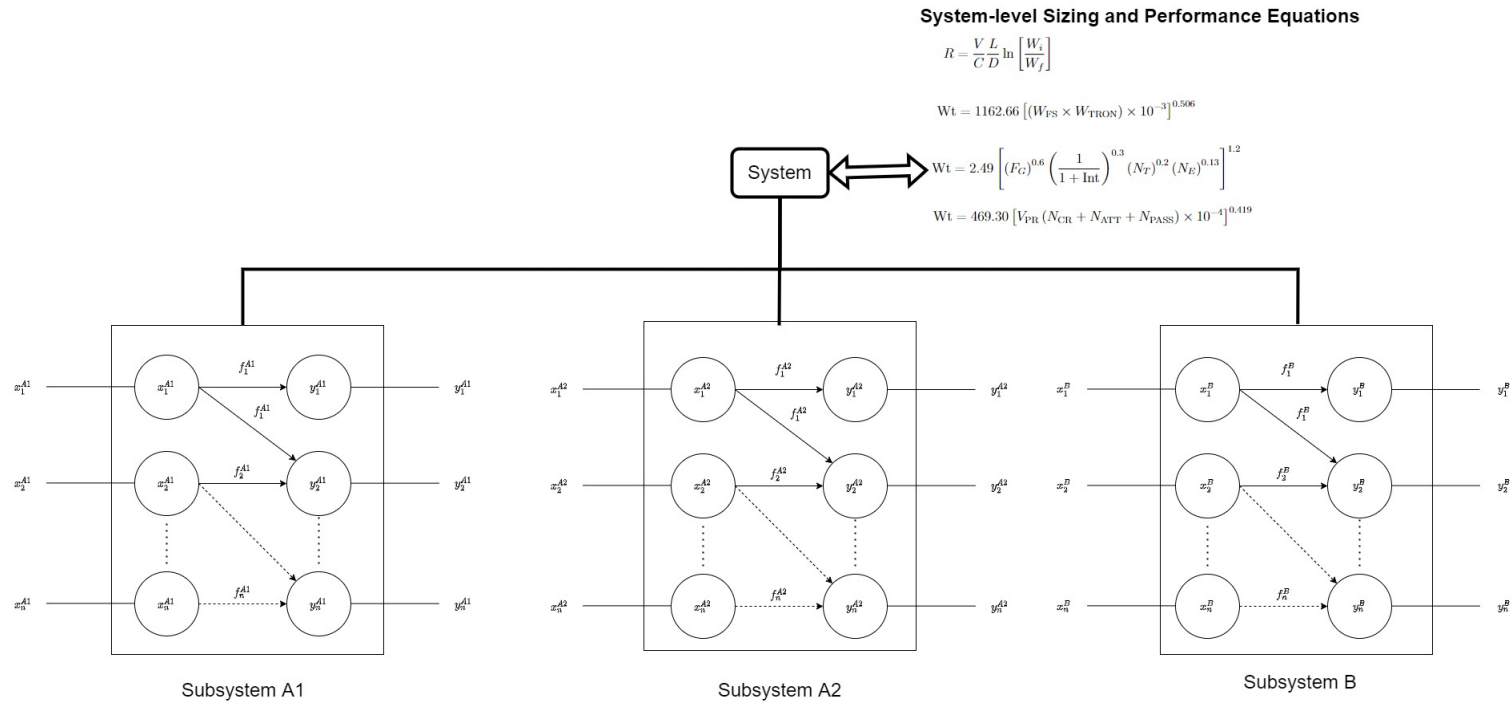


Figure A.2: Directed multi-graph used to model a Bayesian hierarchical system

A.3.5 Problem Solving Environment

An attempt to illustrate the possibility of an IDE that operates under the Bayesian paradigm is shown in figure A.3.

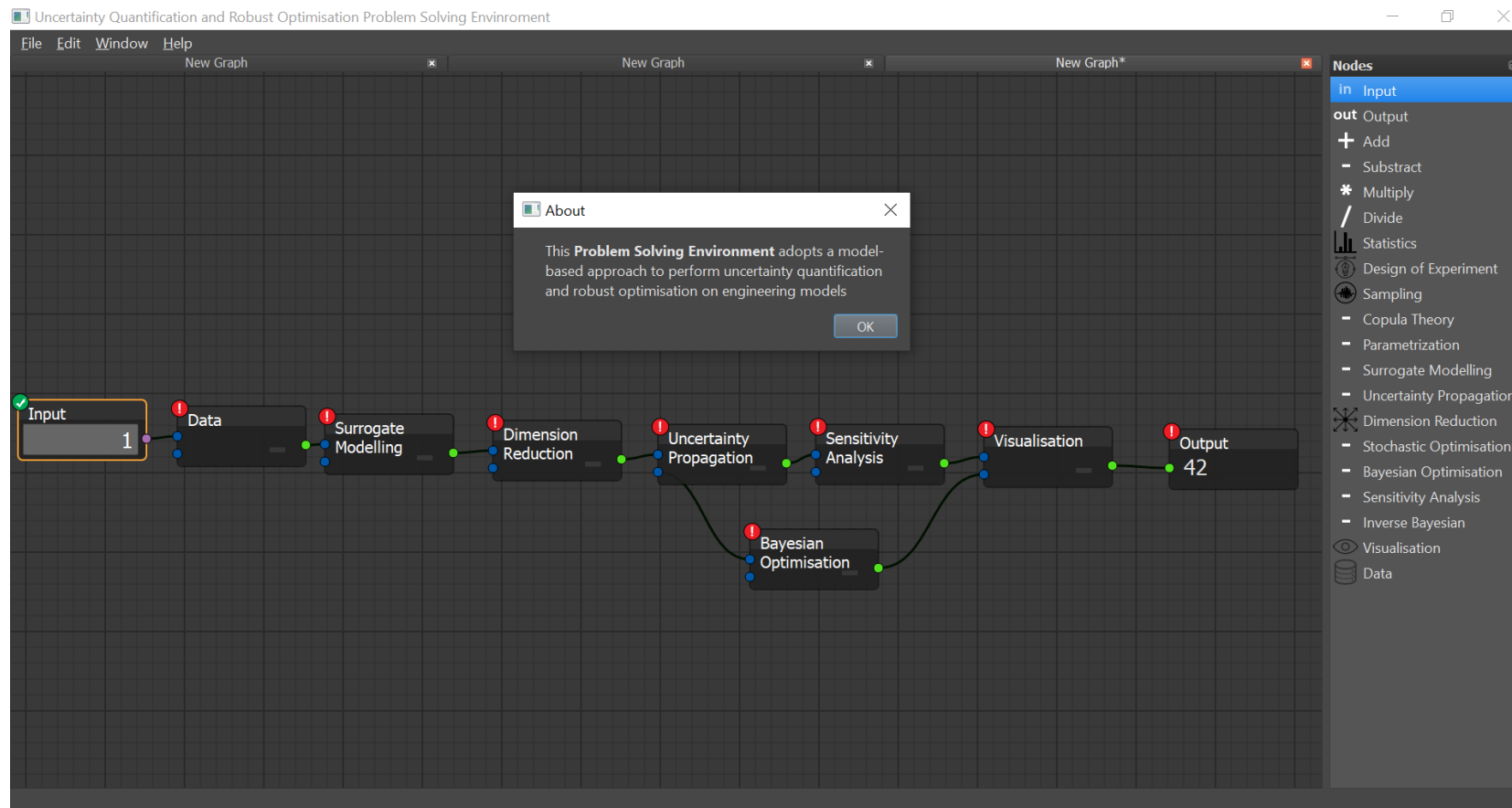


Figure A.3: Problem solving Environment

A.3.6 Limitations

The present framework is still on a draft level. Despite its simplicity there are a number of notions that require a deep mathematical exploration. For instance, despite the use of multigraph, the weights in the arrows are functions. Therefore, they represent compositional relationships that put this framework outside the realm of graph theory. It is necessary instead to use Category theory to understand and develop its mathematical relations.

A.3.7 Further Work

This Bayesian framework is still a proof of concept that requires extensive development. However, it seems very promising because of its foundational ideas

A.4 Software Engineering of the UQOPT Code

The aim of this section is to give a brief overview of some of the activities related to software engineering that helps the development of in-house research codes. Thus, the focus is in the object-oriented analysis and the design process adopted based on the literature survey presented in chapter 2.

Many different notations for object-oriented designs were proposed in the 1980s and 1990s. One of the most successful notation is the Unified Modelling Language (UML), which integrates many of the proposed notations into a single one. UML can describe different models produced during an object-oriented analysis and design. This feature made UML a *de facto* standard for object-oriented modelling. In UML, there are nine standard diagrams to represent a particular “view” of the system or software to be designed. They are divided in two major groups: static and dynamic view. The static view which comprises the use case, class, object, component, deployment diagrams. The dynamic view comprises the sequence, collaboration, state chart and the activity diagrams.

The process of object-oriented modelling is complex and a detailed description of the process is beyond the scope. The reader is referred to Fowler [257]. In this section, however, it is presented the activity diagram representing a dynamic view of the system and the class diagrams representing the static view. Activity diagrams show the dependencies and are often used during the requirements gathering of the process. They help in identifying use cases of a system and operations involved in the realisation of a use case and can be attached to any model element to model its dynamic behaviour. An example of activity diagram is show in figure A.4.

The activity diagram in figure A.4 shows how different operations are sequenced. Based on it, one may infer an “analysis class”. An analysis class abstracts one or more classes and are used to focus on handling functional requirements and defining responsibilities. The structure of the code based on an analysis class is shown in figure A.5. Thus, by referring to these diagrams, one is able to decompose each class in figure A.5 into more specific classes that reflect the tools and techniques surveyed in the

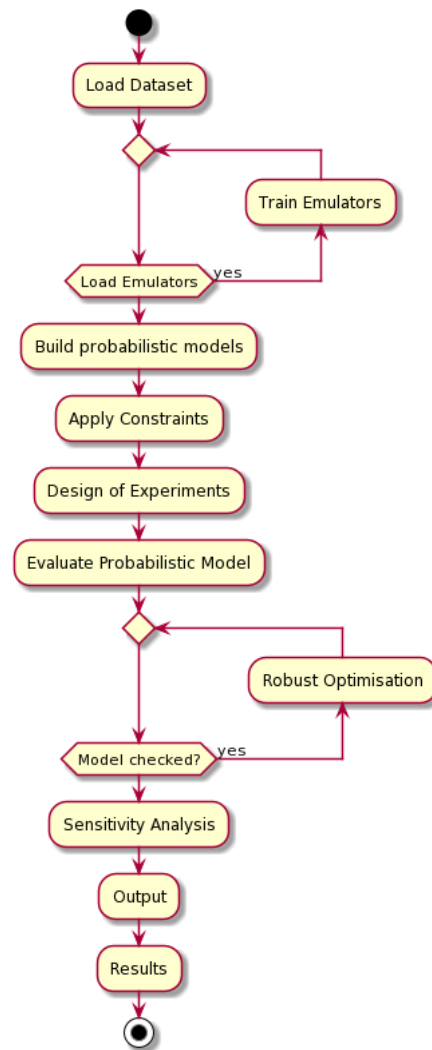


Figure A.4: UML activity diagram describing the main use case

literature review of chapter 2. The result (simplified) is shown in figures A.6, A.7, A.8 and A.9.

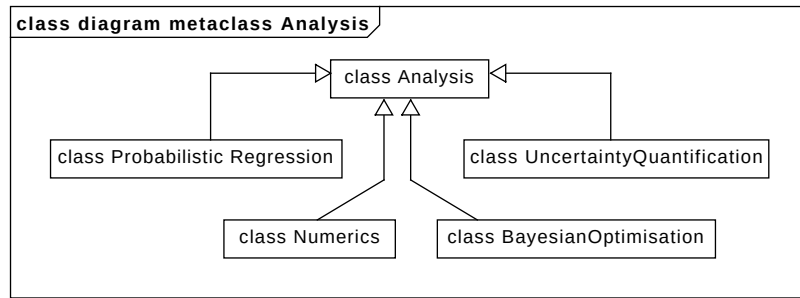


Figure A.5: UML class diagram showing the code structure

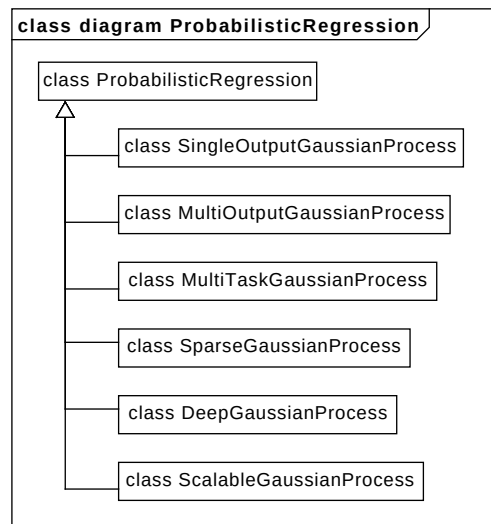


Figure A.6: UML class diagram showing the Probabilistic Regression class

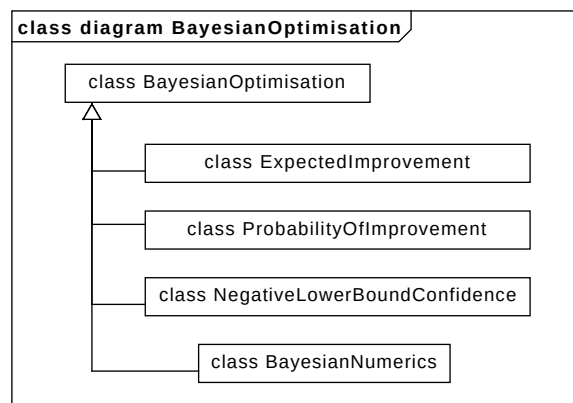


Figure A.7: UML class diagram for the Bayesian Optimisation class

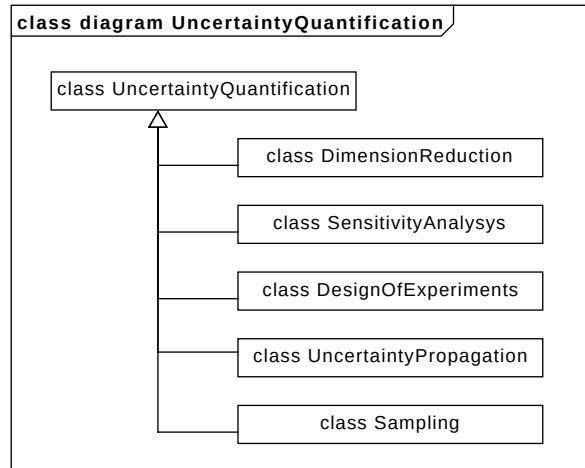


Figure A.8: UML class diagram for the Uncertainty Quantification class

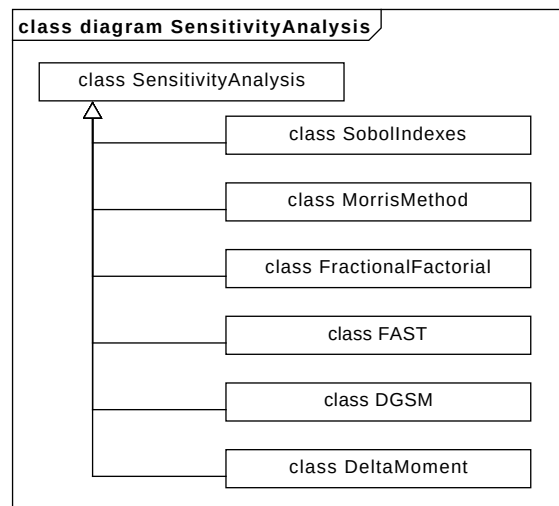


Figure A.9: UML class diagram for the Sensitivity Analysis class

Finally, the directory structure for the code is shown in figure A.10, as an example of how to organise a research code that is capable of performing data-driven analysis in an Jupyter Notebook or Jupyter Lab environment.

```

uqopt-dev
├── docs ← A default Sphinx project
├── datasets
│   └── raw ← The original, immutable data dump
├── models ← Code to train models and then use trained models to make
│   predictions
├── notebooks ← Jupyter notebooks
├── references ← Data dictionaries, manuals, and all other explanatory
│   materials
├── reports ← Generated analysis as HTML, PDF, LaTeX, etc
├── tests ← files for test-driven environment
├── uqopt ← source code for use in this project
│   ├── analysis ← general analysis classes
│   ├── benchmarks ← benchmark problems
│   ├── bo ← Bayesian optimisation classes
│   ├── pr ← probabilistic regression classes
│   ├── plots ← code snippets for plotting
│   ├── uqm ← uncertainty quantification and management classes
│   ├── sa ← sensitivity analysis classes
│   └── utils ← utility and numerical classes

```

Figure A.10: Directory structure for the UQOPT code

Bibliography

- [1] Thomas Henry Gordon Megson. *Aircraft structures for engineering students*. Butterworth-Heinemann, 2016.
- [2] Joaquim RRA Martins and Andrew Ning. *Engineering design optimization*. Cambridge University Press, 2021.
- [3] Yong Yin, Kathryn E Stecke, and Dongni Li. The evolution of production systems from industry 2.0 through industry 4.0. *International Journal of Production Research*, 56(1-2):848–861, 2018.
- [4] Steven L Brunton, J Nathan Kutz, Krithika Manohar, Aleksandr Y Aravkin, Kristi Morgansen, Jennifer Klemisch, Nicholas Goebel, James Buttrick, Jeffrey Poskin, Adriana W Blom-Schieber, et al. Data-driven aerospace engineering: reframing the industry with machine learning. *AIAA Journal*, 59(8):2820–2847, 2021.
- [5] Eric Jenn, Alexandre Albore, Franck Mamalet, Grégory Flandin, Christophe Gabreau, Hervé Delseny, Adrien Gauffriau, Hugues Bonnin, Lucian Alecu, Jérémy Pirard, Baptiste Lefevre, Jean-Marc Gabriel, Cyril Cappi, Laurent Gardès, Sylvaine Picard, Gilles Dulon, Brice Beltran, Jean-Christophe Bianic, Mathieu Damour, Kevin Delmas, and Claire Pagetti. Identifying Challenges to the Certification of Machine Learning for Safety Critical Systems. page 10.
- [6] Christoph Torens, Umut Durak, and Johann C. Dauer. Guidelines and Regulatory Framework for Machine Learning in Aviation. In *AIAA SCITECH 2022 Forum*,

- San Diego, CA & Virtual, January 2022. American Institute of Aeronautics and Astronautics.
- [7] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 2021.
- [8] Harold Jeffreys. *The theory of probability*. OUP Oxford, 1998.
- [9] Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.
- [10] Egbert Torenbeek and H Wittenberg. *Flight physics: essentials of aeronautical disciplines and technology, with historical notes*. Springer Science & Business Media, 2009.
- [11] Martin Christopher. *Logistics & supply chain management*. Pearson Uk, 2016.
- [12] Chunyun Niu and Michael C Niu. *Airframe structural design: practical design information and data on aircraft structures*. Conmilit Press, 1988.
- [13] Tuncer Cebeci. Stability and transition: theory and application. efficient numerical methods with computer programs. *Heidelberg, Springer-Verlag Berlin Heidelberg, 2004.*, pages 1–262, 2004.
- [14] Kwon-Hee Lee and Gyung-Jin Park. Robust Optimization in Discrete Design Space for Constrained Problems. *AIAA Journal*, 40(4):774–780, April 2002.
- [15] Komahan Boopathy, Markus P. Rumpfkeil, and Raymond M. Kolonay. Robust Optimization of a Wing Under Structural and Material Uncertainties. In *17th AIAA Non-Deterministic Approaches Conference*, Kissimmee, Florida, January 2015. American Institute of Aeronautics and Astronautics.
- [16] Andres S. Padron, Juan J. Alonso, and Michael S. Eldred. Multi-fidelity Methods in Aerodynamic Robust Optimization. In *18th AIAA Non-Deterministic*

-
- Approaches Conference*, San Diego, California, USA, January 2016. American Institute of Aeronautics and Astronautics.
- [17] Riccardo Mura, Tiziano Ghisu, and Shahrokh Shahpar. Least Squares Approximation-based Polynomial Chaos Expansion for Uncertainty Quantification and Robust Optimization in Aeronautics. In *AIAA AVIATION 2020 FORUM*, VIRTUAL EVENT, June 2020. American Institute of Aeronautics and Astronautics.
- [18] Gorkem Demir, Recep M Gorguluarslan, and Selin Aradag. A Multi-Objective Robust Aircraft Wing Shape Optimization Approach. page 20.
- [19] Chen Liang and Sankaran Mahadevan. Bayesian Sensitivity Analysis and Uncertainty Integration for Robust Optimization. page 15.
- [20] Kevin M. Ryan, Mark J. Lewis, and Kenneth H. Yu. Comparison of Robust Optimization Methods Applied to Hypersonic Vehicle Design. *Journal of Aircraft*, 52(5):1510–1523, September 2015.
- [21] Michael I Jordan and Tom M Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.
- [22] Liran Einav and Jonathan Levin. Economics in the age of big data. *Science*, 346(6210), 2014.
- [23] W Xindong, J Vipin Kumar, R Quinlan, J Ghosh, Q Yang, H Motoda, GJ McLachlan, N Angus, Bing Liu, S Philip, et al. Top 10 algorithms in data mining. knowledge and information systems. 2008.
- [24] Clifford Lynch. How do your data grow? *Nature*, 455(7209):28–29, 2008.
- [25] Jenny A Harding, Muhammad Shahbaz, and A Kusiak. Data mining in manufacturing: a review. 2006.
- [26] Vivien Marx. The big challenges of big data. *Nature*, 498(7453):255–260, 2013.

-
- [27] Muin J Khoury and John PA Ioannidis. Big data meets public health. *Science*, 346(6213):1054–1055, 2014.
- [28] Yoshua Bengio, Ian Goodfellow, and Aaron Courville. *Deep learning*, volume 1. MIT press Massachusetts, USA:, 2017.
- [29] Carl Edward Rasmussen and Christopher KI Williams. Gaussian processes for machine learning. 2006.
- [30] Colin Campbell. An introduction to kernel methods. *Studies in Fuzziness and Soft Computing*, 66:155–192, 2001.
- [31] Nicolas Durrande, James Hensman, Magnus Rattray, and Neil D Lawrence. Detecting periodicities with gaussian processes. *PeerJ Computer Science*, 2:e50, 2016.
- [32] Chun Kai Ling, Kian Hsiang Low, and Patrick Jaillet. Gaussian process planning with lipschitz continuous reward functions: Towards unifying Bayesian optimization, active learning, and beyond. *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, pages 1860–1866, 2016.
- [33] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of Mathematical Psychology*, 85:1–16, 2018.
- [34] Kirthevasan Kandasamy, Gautam Dasarathy, Junier Oliva, Jeff Schneider, and Barnabás Póczos. Gaussian process optimisation with multi-fidelity evaluations. In *Proceedings of the 30th/International Conference on Advances in Neural Information Processing Systems (NIPS’30)*, 2016.
- [35] Zhenwen Dai, Andreas Damianou, James Hensman, and Neil Lawrence. Gaussian process models with parallelization and gpu acceleration. *arXiv preprint arXiv:1410.4984*, 2014.

-
- [36] J. A. Vargas-Guzmán, A. W. Warrick, and D. E. Myers. Coregionalization by linear combination of nonorthogonal components. *Mathematical Geology*, 34(4):405–419, 2002.
- [37] Andrew Gordon Wilson, David A Knowles, and Zoubin Ghahramani. Gaussian process regression networks. *arXiv preprint arXiv:1110.4411*, 2011.
- [38] Edwin V. Bonilla, Kian Ming A. Chai, and Christopher K.I. Williams. Multi-task Gaussian Process prediction. *Advances in Neural Information Processing Systems 20 - Proceedings of the 2007 Conference*, 2009.
- [39] Andreass Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. *Advances in Neural Information Processing Systems*, pages 41–48, 2007.
- [40] Jaakko Peltonen, Yusuf Yaslan, and Samuel Kaski. Variational Bayes Learning from Relevant Tasks Only. *Learning from Multiple Sources Workshop, 13 December 2008, Whistler Canada*, (May), 2008.
- [41] Edwin V. Bonilla, Felix V. Agakov, and Christopher K.I. Williams. Kernel multi-task learning using task-specific features. *Journal of Machine Learning Research*, 2:43–50, 2007.
- [42] Henry B. Moss, David S. Leslie, and Paul Rayson. MUMBO: Multi-task Max-value Bayesian Optimization. pages 1–23, 2020.
- [43] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in Neural Information Processing Systems*, 2018-December(NeurIPS):527–538, 2018.
- [44] Kevin Swersky, Jasper Snoek, and Ryan P. Adams. Multi-task Bayesian optimization. *Advances in Neural Information Processing Systems*, pages 1–9, 2013.

-
- [45] Andreas Damianou and Neil D Lawrence. Deep gaussian processes. In *Artificial intelligence and statistics*, pages 207–215. PMLR, 2013.
- [46] James Hensman and Neil D. Lawrence. Nested Variational Compression in Deep Gaussian Processes. pages 1–21, 2014.
- [47] Zhenwen Dai, Andreas Damianou, Javier González, and Neil Lawrence. Variational auto-encoded deep Gaussian processes. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, pages 1–11, 2016.
- [48] David Duvenaud, Hannes Nickisch, and Carl Edward Rasmussen. Additive gaussian processes. *arXiv preprint arXiv:1112.4394*, 2011.
- [49] Shengyang Sun, Guodong Zhang, Chaoqi Wang, Wenyuan Zeng, Jiaman Li, and Roger Grosse. Differentiable compositional kernel learning for gaussian processes. In *International Conference on Machine Learning*, pages 4828–4837. PMLR, 2018.
- [50] Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. *arXiv preprint arXiv:1705.08933*, 2017.
- [51] David Duvenaud, Oren Rippel, Ryan Adams, and Zoubin Ghahramani. Avoiding pathologies in very deep networks. In *Artificial Intelligence and Statistics*, pages 202–210. PMLR, 2014.
- [52] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.
- [53] Jonas Mockus. Application of bayesian approach to numerical methods of global and stochastic optimization. *Journal of Global Optimization*, 4(4):347–365, 1994.
- [54] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

-
- [55] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.
- [56] Yibo Yang and Paris Perdikaris. Conditional deep surrogate models for stochastic, high-dimensional, and multi-fidelity systems. *Computational Mechanics*, 64(2):417–434, 2019.
- [57] Paul Feliot, Julien Bect, and Emmanuel Vazquez. A Bayesian approach to constrained single- and multi-objective optimization. *Journal of Global Optimization*, 67(1-2):97–133, 2017.
- [58] Majid Abdolshah, Alistair Shilton, Santu Rana, Sunil Gupta, and Svetha Venkatesh. Multi-objective Bayesian optimisation with preferences over objectives. 2019.
- [59] Mohit Malu, Gautam Dasarathy, and Andreas Spanias. Bayesian optimization in high-dimensional spaces: A brief survey. In *2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA)*, pages 1–8. IEEE, 2021.
- [60] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.
- [61] Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.
- [62] Peter I Frazier and Jialei Wang. Bayesian optimization for materials design. In *Information science for materials discovery and design*, pages 45–75. Springer, 2016.
- [63] Ralph C Smith. *Uncertainty quantification: theory, implementation, and applications*, volume 12. Siam, 2013.

-
- [64] Dave M Higdon, Mark C Anderson, Salman Habib, Richard Klein, Mark Berliner, Curt Covey, Omar Ghattas, Carlo Graziani, Mark Seager, Joseph Sefcik, et al. Uncertainty quantification and error analysis. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2010.
- [65] Harish Agarwal, John E Renaud, Evan L Preston, and Dhanesh Padmanabhan. Uncertainty quantification using evidence theory in multidisciplinary design optimization. *Reliability Engineering & System Safety*, 85(1-3):281–294, 2004.
- [66] Roger G Ghanem, Charanraj Thimiseti, Iman Yadegaran, Vahid Kasharvazadeh, Sami Masri, John Red-Horse, Robert Moser, Todd A Oliver, Pol Spanos, and Osama J Aldraihem. A probabilistic approach to the nasa langley multidisciplinary uncertainty quantification challenge problem. In *16th AIAA Non-Deterministic Approaches Conference*, page 1348, 2014.
- [67] SF Ghoreishi and DL Allaire. Adaptive uncertainty propagation for coupled multidisciplinary systems. *AIAA journal*, 55(11):3940–3950, 2017.
- [68] Seyede Fatemeh Ghoreishi and Douglas L Allaire. Compositional uncertainty analysis via importance weighted gibbs sampling for coupled multidisciplinary systems. In *18th AIAA non-deterministic approaches conference*, page 1443, 2016.
- [69] Xingzhi Hu, Xiaoqian Chen, Valerio Lattarulo, and Geoffrey T Parks. Multidisciplinary optimization under high-dimensional uncertainty for small satellite system design. *AIAA Journal*, 54(5):1732–1741, 2016.
- [70] Zhen Jiang, Wei Chen, and Brian J German. Multidisciplinary statistical sensitivity analysis considering both aleatory and epistemic uncertainties. *AIAA journal*, 54(4):1326–1338, 2016.
- [71] Farooq Akram and Dimitri N Mavris. Surrogate modeling based uncertainty quantification in subject matter expert systems. In *55th AIAA/ASMe/ASCE/AHS/SC Structures, Structural Dynamics, and Materials Conference*, page 0218, 2014.

-
- [72] Bogdan Profir, Murat Hakki Eres, James Scanlan, Michael Moss, and Ron Bates. Uncertainty quantification via elicitation of expert judgements. In *16th AIAA Aviation Technology, Integration, and Operations Conference*, page 3459, 2016.
- [73] Eric Dow and Qiqi Wang. Uncertainty quantification of structural uncertainties in rans simulations of complex flows. In *20th AIAA Computational Fluid Dynamics Conference*, page 3865, 2011.
- [74] J Hollom and N Qin. Uncertainty analysis and robust shape optimisation for laminar flow aerofoils. *The Aeronautical Journal*, 125(1284):365–388, 2021.
- [75] Vishal Srivastava and Karthik Duraisamy. Aerodynamic design of aircraft engine nozzles with consideration of model form uncertainties. In *2018 AIAA Non-Deterministic Approaches Conference*, page 2175, 2018.
- [76] Anirban Chaudhuri, John Jasa, Joaquim RRA Martins, and Karen E Willcox. Multifidelity optimization under uncertainty for a tailless aircraft. In *2018 AIAA Non-Deterministic Approaches Conference*, page 1658, 2018.
- [77] Anirban Chaudhuri and Karen E Willcox. Multifidelity uncertainty propagation in coupled multidisciplinary systems. In *18th AIAA Non-Deterministic Approaches Conference*, page 1442, 2016.
- [78] Gianluca Geraci, Michael S Eldred, Alex Gorodetsky, and John Jakeman. Recent advancements in multilevel-multifidelity techniques for forward uq in the darpa sequoia project. In *AIAA Scitech 2019 Forum*, page 0722, 2019.
- [79] Leo Wai-Tsun Ng and Karen E Willcox. A multifidelity approach to aircraft conceptual design under uncertainty. In *10th AIAA Multidisciplinary Design Optimization Conference*, page 0802, 2014.
- [80] Shankar Sankararaman, Sankaran Mahadevan, Kyle McLemore, Samuel Bradford, Chen Liang, and Lee Peterson. Test resource allocation for uncertainty quantification of multi-level and coupled systems. In *52nd*

- AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 19th AIAA/ASME/AHS Adaptive Structures Conference 13t*, page 1904, 2011.
- [81] Sam Friedman, Benson Isaac, Seyede Fatemeh Ghoreishi, and Douglas L Allaire. Efficient decoupling of multiphysics systems for uncertainty propagation. In *2018 AIAA Non-Deterministic Approaches Conference*, page 1661, 2018.
- [82] Philip Beran, Bret Stanford, and Christopher Schrock. Uncertainty quantification in aeroelasticity. *Annual review of fluid mechanics*, 49:361–386, 2017.
- [83] Patrick H Reisenthel, John F Love, Daniel J Lesieutre, and Robert E Childs. Cumulative global metamodels with uncertainty-a tool for aerospace integration. *Aeronautical Journal*, 110(1108):375–384, 2006.
- [84] Matthew E Riley and Ramana V Grandhi. Quantification of modeling-induced uncertainties in simulation-based analyses. *AIAA journal*, 52(1):195–202, 2014.
- [85] Jonathan M Burt, Mark A Hagenmaier, Dean R Eklund, Heidi L Wilkin, and Eswar Josyula. Uncertainty quantification for a scramjet inlet flow. In *50th AIAA/ASME/SAE/ASEE Joint Propulsion Conference*, page 3844, 2014.
- [86] Xun Huan, Cosmin Safta, Khachik Sargsyan, Gianluca Geraci, Michael S Eldred, Zachary P Vane, Guilhem Lacaze, Joseph C Oefelein, and Habib N Najm. Global sensitivity analysis and estimation of model error, toward uncertainty quantification in scramjet computations. *AIAA Journal*, 56(3):1170–1184, 2018.
- [87] Xun Huan, Cosmin Safta, Zachary P Vane, Guilhem Lacaze, Joseph C Oefelein, and Habib N Najm. Uncertainty propagation using conditional random fields in large-eddy simulations of scramjet computations. In *AIAA Scitech 2019 Forum*, page 0724, 2019.
- [88] Vadiraj Hombal, Kevin Wolfe, You Ling, and Sankaran Mahadevan. Uncertainty quantification in non-planar crack growth analysis. In *53rd*

- AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA*, page 1541, 2012.
- [89] Luc Huyse and Ben H Thacker. A framework to estimate uncertain random variables. In *AIAA Paper 2004-1828, 45 th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*. Citeseer, 2004.
- [90] M Rezaei, SA Fazlzadeh, A Mazidi, MI Friswell, and HH Khodaparast. Fuzzy uncertainty analysis and reliability assessment of aeroelastic aircraft wings. *The Aeronautical Journal*, 124(1275):786–811, 2020.
- [91] Matthew Riley and Ramana Grandhi. A novel surrogate modeling technique for parametric uncertainty quantification in simulation-based design. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA*, page 1761, 2012.
- [92] Matthew E Riley, Ramana V Grandhi, and Raymond Kolonay. Quantification of modeling uncertainty in aeroelastic analyses. *Journal of Aircraft*, 48(3):866–873, 2011.
- [93] C Scarth and S Adhikari. Modeling spatially varying uncertainty in composite structures using lamination parameters. *AIAA Journal*, 55(11):3951–3965, 2017.
- [94] Edward M Kraft. Integrating model-based systems engineering and uncertainty quantification to design optimum aircraft test campaigns. In *2018 Applied Aerodynamics Conference*, page 2853, 2018.
- [95] Bryan Mesmer. Challenges in modeling of stakeholders in systems engineering: From end users to designers, individuals to groups. 2018.
- [96] Mauro Massari, Pierluigi Di Lizia, and Mirco Rasotto. Nonlinear uncertainty propagation in astrodynamics using differential algebra and graphics processing units. *Journal of Aerospace Information Systems*, 14(9):493–503, 2017.

-
- [97] Arturo Molina-Cristóbal, Xin Chen, Marin D Guenov, Atif Riaz, and Albert SJ van Heerden. Interactive uncertainty allocation and tradeoff for early stage design of complex systems. *AIAA Journal*, 58(3):1356–1368, 2020.
- [98] Vijitashwa Pandey. Uncertainty modeling using mixture distributions for decision-based design. In *19th AIAA Non-Deterministic Approaches Conference*, page 1093, 2017.
- [99] H-U Park, J Chung, and D Neufeld. Uncertainty based aircraft derivative design for requirement changes. *The Aeronautical Journal*, 120(1224):375–389, 2016.
- [100] Angel Urbina, Sankaran Mahadevan, and Thomas Paez. Resource allocation using quantification of margins and uncertainty. In *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference 18th AIAA/ASME/AHS Adaptive Structures Conference 12th*, page 2510, 2010.
- [101] Angel Urbina, Laura Swiler, Shankar Sankararaman, and Sankaran Mahadevan. Decision making under epistemic uncertainty for a complex mechanical system. In *52nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 19th AIAA/ASME/AHS Adaptive Structures Conference 13th*, page 1907, 2011.
- [102] Kendra L Van Buren and François M Hemez. Robust decision making applied to the nasa multidisciplinary uncertainty quantification challenge problem. *Journal of Aerospace Information Systems*, 12(1):35–48, 2015.
- [103] Yoshinori Matsuno, Takeshi Tsuchiya, and Naoki Matayoshi. Near-optimal control for aircraft conflict resolution in the presence of uncertainty. *Journal of Guidance, Control, and Dynamics*, 39(2):326–338, 2016.
- [104] Paromita Nath, Zhen Hu, and Sankaran Mahadevan. Modeling and uncertainty quantification of material properties in additive manufacturing. In *2018 AIAA Non-Deterministic Approaches Conference*, page 0923, 2018.

- [105] Christopher J Roy and William L Oberkampf. A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing. *Computer methods in applied mechanics and engineering*, 200(25-28):2131–2144, 2011.
- [106] Lucas A Bonomo, André MN Spillere, and Julio A Cordioli. Parametric uncertainty analysis for impedance eduction based on prony’s method. *AIAA Journal*, 58(8):3625–3638, 2020.
- [107] Sifeng Bi, Saurabh Prabhu, Scott Cogan, and Sez Atamturktur. Uncertainty quantification metrics with varying statistical information in model calibration and validation. *AIAA journal*, 55(10):3570–3583, 2017.
- [108] Qiang Yang, Guokai Han, Weihua Xie, Songhe Meng, and Hua Jin. Uncertainty characterization methods for sparsely sampled quantity: A tradeoff analysis considering propagation. *AIAA Journal*, 58(7):3129–3138, 2020.
- [109] Katherine N Gatian and Dimitri N Mavris. Enabling technology portfolio selection through quantitative uncertainty analysis. In *15th AIAA Aviation Technology, Integration, and Operations Conference*, page 2436, 2015.
- [110] Sergio Amaral, Douglas Allaire, and Karen Willcox. A decomposition approach to uncertainty analysis of multidisciplinary systems. In *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, page 5563, 2012.
- [111] Houman Owhadi. The game theoretic approach to uncertainty quantification, reduced order modeling and numerical analysis. In *19th AIAA Non-Deterministic Approaches Conference*, page 1092, 2017.
- [112] Juan J Alonso, Michael S Eldred, Paul Constantine, Karthikeyan Duraisamy, Charbel Farhat, Gianluca Iaccarino, and John Jakeman. Scalable environment for quantification of uncertainty and optimization in industrial applications (sequoia). In *19th AIAA Non-Deterministic Approaches Conference*, page 1327, 2017.

-
- [113] Michaël Baudin, Anne Dutfoy, Bertrand Iooss, and Anne-Laure Popelin. Open turns: An industrial software for uncertainty quantification in simulation. *arXiv preprint arXiv:1501.05242*, 2015.
- [114] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.
- [115] Ryohei Omiya and Kenichi Rinoie. Study on classification of aircraft conceptual design parameters by principal component analysis. In *17th AIAA Aviation Technology, Integration, and Operations Conference*, page 3070, 2017.
- [116] Lionel Richard and B Hazet. Use of principal component analysis for crash-worthiness optimization. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, page 5559, 2002.
- [117] Kyunghoon Lee, Sriram Rallabhandi, and Dimitri Mavris. Aerodynamic model reconstruction via probabilistic principal component analysis. In *46th AIAA Aerospace Sciences Meeting and Exhibit*, page 899, 2015.
- [118] Zhang Lizhang, Zeyong Yin, Dong Mi, Yanyun Zhao, and Zhengming Qian. Multidisciplinary design optimization of air propulsion components using principal component analysis method. In *21st AIAA International Space Planes and Hypersonics Technologies Conference*, page 2375, 2017.
- [119] Sameer K Jasra, Gianluca Valentino, Alan Muscat, David Zammit-Mangion, and Robert Camilleri. Evaluation of flight parameters during approach and landing phases by applying principal component analysis. In *AIAA Scitech 2020 Forum*, page 0674, 2020.
- [120] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.
- [121] Chongsun Park, Morgan C Wang, and Eun Bi Mo. Probabilistic penalized

- principal component analysis. *Communications for Statistical Applications and Methods*, 24(2):143–154, 2017.
- [122] Gamze Bayrak and Erdem Acar. Reliability estimation using markov chain monte carlo-based tail modeling. *AIAA Journal*, 56(3):1211–1224, 2018.
- [123] Richard Butler, Timothy J Dodwell, Raphael T Haftka, Nam Ho Kim, Tatiana Kim, Stephanie Kynaston, and Rob Scheichl. Uncertainty quantification of composite structures with defects using multilevel monte carlo simulations. In *17th AIAA non-deterministic approaches conference*, page 1598, 2015.
- [124] Ilya M Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and computers in simulation*, 55(1-3):271–280, 2001.
- [125] Xingzhi Hu, Xiaoqian Chen, Geoffrey T Parks, and Wen Yao. Review of improved monte carlo methods in uncertainty-based design optimization for aerospace vehicles. *Progress in Aerospace Sciences*, 86:20–27, 2016.
- [126] Christiane Lemieux. *Monte carlo and quasi-monte carlo sampling*. Springer Science & Business Media, 2009.
- [127] Matteo Corbetta and Chetan S Kulkarni. A fast monte carlo method for model-based prognostics based on stochastic calculus. In *AIAA Scitech 2019 Forum*, page 2232, 2019.
- [128] Benjamin Peherstorfer, Philip S Beran, and Karen E Willcox. Multifidelity monte carlo estimation for large-scale uncertainty propagation. In *2018 AIAA Non-Deterministic Approaches Conference*, page 1660, 2018.
- [129] Abe Sklar. Random variables, joint distribution functions, and copulas. *Kybernetika*, 9(6):449–460, 1973.
- [130] Xuan Li and Wei Zhang. Probabilistic fatigue evaluation of floating wind turbine

- using combination of surrogate model and copula model. In *AIAA Scitech 2019 Forum*, page 0247, 2019.
- [131] Turab Zaidi, Hernando Jimenez, and Dimitri Mavris. Copulas theory for probabilistic assessment: Overview with application to airplane performance analysis. *Journal of Aircraft*, 52(6):1802–1820, 2015.
- [132] Turab Zaidi, Hernando Jimenez, and Dimitri Mavris. Copulas Theory for Probabilistic Assessment: Overview with Application to Airplane Performance Analysis. *Journal of Aircraft*, 52(6):1802–1820, November 2015. Number: 6.
- [133] Piotr Jaworski, Fabrizio Durante, Wolfgang Karl Hardle, and Tomasz Rychlik. *Copula theory and its applications*, volume 198. Springer, 2010.
- [134] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global sensitivity analysis: the primer*. John Wiley & Sons, 2008.
- [135] I. M. Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Computers in Simulation*, 55(1-3):271–280, 2001.
- [136] I. M. Sobol’ and S. Kucherenko. Derivative based global sensitivity measures and their link with global sensitivity indices. *Mathematics and Computers in Simulation*, 79(10):3009–3017, 2009.
- [137] Sergei Kucherenko and Shugfang Song. Derivative-based global sensitivity measures and their link with sobol’sensitivity indices. In *Monte Carlo and Quasi-Monte Carlo Methods*, pages 455–469. Springer, 2016.
- [138] Sergei Kucherenko and Shufang Song. Different numerical estimators for main effect global sensitivity indices. *Reliability Engineering & System Safety*, 165:222–238, 2017.

-
- [139] S. Kucherenko and S. Song. Different numerical estimators for main effect global sensitivity indices. *Reliability Engineering and System Safety*, 165(April):222–238, 2017.
- [140] E. Borgonovo. A new uncertainty importance measure. *Reliability Engineering and System Safety*, 92(6):771–784, 2007.
- [141] S. Tarantola, D. Gatelli, and T. A. Mara. Random balance designs for the estimation of first order global sensitivity indices. *Reliability Engineering and System Safety*, 91(6):717–727, 2006.
- [142] Andrea Saltelli. Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145(2):280–297, 2002.
- [143] James Strand and David Goldstein. Sensitivity analysis for dsmc simulations of high-temperature air chemistry. In *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 535, 2011.
- [144] Jonathan Burt, Evgeny Titov, and Eswar Josyula. Sensitivity analysis and uncertainty quantification for a set of hypersonic shock interaction flows. In *43rd AIAA Thermophysics Conference*, page 3100, 2012.
- [145] Sarah L Kieweg, Jaideep Ray, V Gregory Weirs, Brian Carnes, Derek Dinzl, Brian Freno, Micah Howard, Eric Phipps, William Rider, and Thomas Smith. Validation assessment of hypersonic double-cone flow simulations using uncertainty quantification, sensitivity analysis, and validation metrics. In *AIAA SciTech 2019 Forum*, page 2278, 2019.
- [146] Erin C DeCarlo, Sankaran Mahadevan, Benjamin P Smarslok, and Daniel M Sparkman. Efficient global sensitivity analysis for time-dependent, multidisciplinary models. In *19th AIAA Non-Deterministic Approaches Conference*, page 0594, 2017.
- [147] Julie Tryoen, Pietro Congedo, Remi Abgrall, Nadège Villedieu, and Thierry Magin. Sensitivity analysis and characterization of the uncertain input data for

- the expert vehicle. In *51st AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, page 956, 2013.
- [148] Michael Dodson and Geoffrey T Parks. Robust aerodynamic design optimization using polynomial chaos. *Journal of Aircraft*, 46(2):635–646, 2009.
- [149] Robert Walters. Stochastic fluid mechanics via polynomial chaos. In *41st Aerospace Sciences Meeting and Exhibit*, page 413, 2006.
- [150] Wataru Yamazaki. Stochastic drag analysis via polynomial chaos uncertainty quantification. *Transactions of the Japan society for aeronautical and space sciences*, 58(2):89–99, 2015.
- [151] Avinash Prabhakar, James Fisher, and Raktim Bhattacharya. Polynomial chaos-based analysis of probabilistic uncertainty in hypersonic flight dynamics. *Journal of guidance, control, and dynamics*, 33(1):222–234, 2010.
- [152] Seung-Kyum Choi, Ramana Grandhi, Robert Canfield, and Chris Pettit. Polynomial chaos expansion with latin hypercube sampling for predicting response variability. In *44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1749, 2003.
- [153] Anthony M DeGennaro, Clarence W Rowley, and Luigi Martinelli. Uncertainty quantification for airfoil icing using polynomial chaos expansions. *Journal of Aircraft*, 52(5):1404–1411, 2015.
- [154] Parikshit Dutta and Raktim Bhattacharya. Nonlinear estimation of hypersonic flight using polynomial chaos. In *AIAA Guidance, Navigation, and Control Conference*, page 7590, 2010.
- [155] Brandon A Jones, Alireza Doostan, and George H Born. Nonlinear propagation of orbit uncertainty using non-intrusive polynomial chaos. *Journal of Guidance, Control, and Dynamics*, 36(2):430–444, 2013.

- [156] Lisa J Whittle and Marco Sagliano. Stochastic optimal trajectory generation via multivariate polynomial chaos. In *2018 AIAA Guidance, Navigation, and Control Conference*, page 0849, 2018.
- [157] Xiang Li, Prasanth B Nair, ZhiGang Zhang, Lin Gao, and Chen Gao. Aircraft robust trajectory optimization using nonintrusive polynomial chaos. *Journal of Aircraft*, 51(5):1592–1603, 2014.
- [158] Riccardo Mura, Tiziano Ghisu, and Shahrokh Shahpar. Least squares approximation-based polynomial chaos expansion for uncertainty quantification and robust optimization in aeronautics. In *AIAA AVIATION 2020 FORUM*, page 3163, 2020.
- [159] Thomas K West and Clyde Gumbert. Multifidelity, multidisciplinary uncertainty quantification with non-intrusive polynomial chaos. In *58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1936, 2017.
- [160] Jethro Nagawkar, Leifur T Leifsson, and Xiaosong Du. Applications of polynomial chaos-based cokriging to aerodynamic design optimization benchmark problems. In *AIAA Scitech 2020 Forum*, page 0542, 2020.
- [161] Yicheng Zhou and Zhenzhou Lu. Active polynomial chaos expansion for reliability-based design optimization. *AIAA Journal*, 57(12):5431–5446, 2019.
- [162] John A Schaefer, Andrew W Cary, Mortaza Mani, and Thomas Grandine. Uncertainty quantification across design space using spatially accurate polynomial chaos. In *AIAA Scitech 2020 Forum*, page 1416, 2020.
- [163] Simon Abraham, Panagiotis Tsirikoglou, Chris Lacor, Ghader Ghorbaniasl, Dirk Wunsch, Charles Hirsch, and Francesco Contino. Uncertainty quantification in industrial turbo-machinery design using sparse polynomial chaos expansions. In *2018 Multidisciplinary Analysis and Optimization Conference*, page 3103, 2018.

-
- [164] Serhat Hosder and Robert Walters. Non-intrusive polynomial chaos methods for uncertainty quantification in fluid dynamics. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, page 129, 2010.
- [165] Brandon A Jones, Nathan Parrish, and Alireza Doostan. Postmaneuver collision probability estimation using sparse polynomial chaos expansions. *Journal of Guidance, Control, and Dynamics*, 38(8):1425–1437, 2015.
- [166] Tiziano Ghisu, Geoffrey T Parks, Jerome P Jarrett, and P John Clarkson. Adaptive polynomial chaos for gas turbine compression systems performance analysis. *AIAA journal*, 48(6):1156–1170, 2010.
- [167] Sergey Oladyshkin and Wolfgang Nowak. Data-driven uncertainty quantification using the arbitrary polynomial chaos expansion. *Reliability Engineering & System Safety*, 106:179–190, 2012.
- [168] Matthew John Franchetti. *Lean six sigma for engineers and managers: with applied case studies*. CRC Press, 2015.
- [169] Basem El-Haik and Raid Al-Aomar. *Simulation-based lean six-sigma and design for six-sigma*. John Wiley & Sons, 2006.
- [170] Hakan Aydemir, Ugur Zengin, and Umut Durak. The Digital Twin Paradigm for Aircraft Review and Outlook. In *AIAA Scitech 2020 Forum*, Orlando, FL, January 2020. American Institute of Aeronautics and Astronautics.
- [171] Michael Grieves. Digital twin: manufacturing excellence through virtual factory replication. *White paper*, 1:1–7, 2014.
- [172] Christof Ebert and Carlos Henrique C Duarte. Digital transformation. *IEEE Softw.*, 35(4):16–21, 2018.
- [173] Edward Glaessgen and David Stargel. The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles. In *53rd AIAA/ASME/ASCE/AHS/ASC*

- Structures, Structural Dynamics and Materials Conference* & 20th *AIAA/ASME/AHS Adaptive Structures Conference* & 14th *AIAA*, Honolulu, Hawaii, April 2012. American Institute of Aeronautics and Astronautics.
- [174] Edward M. Kraft. Developing a Digital Thread / Digital Twin Aerodynamic Performance Authoritative Truth Source. In *2018 Aviation Technology, Integration, and Operations Conference*, Atlanta, Georgia, June 2018. American Institute of Aeronautics and Astronautics.
- [175] Banavara R. Seshadri and Thiagarajan Krishnamurthy. Structural Health Management of Damaged Aircraft Structures Using Digital Twin Concept. In *25th AIAA/AHS Adaptive Structures Conference*, Grapevine, Texas, January 2017. American Institute of Aeronautics and Astronautics.
- [176] Michael G. Kapteyn, David J. Knezevic, and Karen Willcox. Toward predictive digital twins via component-based reduced-order models and interpretable machine learning. In *AIAA Scitech 2020 Forum*, Orlando, FL, January 2020. American Institute of Aeronautics and Astronautics.
- [177] Stanislaus Reitenbach, Maximilian Vieweg, Richard Becker, Carsten Hollmann, Florian Wolters, Jens Schmeink, Tom Otten, and Martin Siggel. Collaborative Aircraft Engine Preliminary Design using a Virtual Engine Platform, Part A: Architecture and Methodology. In *AIAA Scitech 2020 Forum*, Orlando, FL, January 2020. American Institute of Aeronautics and Astronautics.
- [178] Omar Hazbon, Luis Gutierrez, Cees Bil, Marcello Napolitano, and Mario Fravolini. Digital Twin Concept for Aircraft System Failure Detection and Correction. In *AIAA Aviation 2019 Forum*, Dallas, Texas, June 2019. American Institute of Aeronautics and Astronautics.
- [179] David Jimenez Mena, Sylvain Pluchart, Stephane Mouvand, and Olivier Broca. Rocket Engine Digital Twin – Modeling and Simulation Benefits. In *AIAA*

- Propulsion and Energy 2019 Forum*, Indianapolis, IN, August 2019. American Institute of Aeronautics and Astronautics.
- [180] Kenneth Reifsnider and Prasun Majumdar. Multiphysics Stimulated Simulation Digital Twin Methods for Fleet Management. In *54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Boston, Massachusetts, April 2013. American Institute of Aeronautics and Astronautics.
- [181] Howard Eisner. *Essentials of project and systems engineering management*. John Wiley & Sons, 2008.
- [182] Benjamin S Blanchard, Wolter J Fabrycky, and Walter J Fabrycky. *Systems engineering and analysis*, volume 4. Prentice hall Englewood Cliffs, NJ, 1990.
- [183] Bruce Powel Douglass. *Agile systems engineering*. Morgan Kaufmann, 2015.
- [184] Gustavo P Krupa. Application of agile model-based systems engineering in aircraft conceptual design. *The Aeronautical Journal*, 123(1268):1561–1601, 2019.
- [185] Durward K Sobek II, Allen C Ward, and Jeffrey K Liker. Toyota’s principles of set-based concurrent engineering. *MIT Sloan Management Review*, 40(2):67, 1999.
- [186] Donald Reinertsen and Tom Bellinson. *The principles of product development flow: second generation lean product development*, 2014.
- [187] Robert I Winner, James P Pennell, Harold E Bertrand, and Marko M Slusarczyk. The role of concurrent engineering in weapons system acquisition. Technical report, Institute for Defense Analyses Alexandria VA, 1988.
- [188] David J Singer, Norbert Doerry, and Michael E Buckley. What is set-based design? *Naval Engineers Journal*, 121(4):31–43, 2009.

-
- [189] Alex Georgiades, Sanjiv Sharma, Timoleon Kipouros, and Mark Savill. Adopt: An augmented set-based design framework with optimisation. *Design Science*, 5, 2019.
- [190] Christopher M Bishop. Model-based machine learning. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984):20120222, 2013.
- [191] Noah Goodman, Vikash Mansinghka, Daniel M Roy, Keith Bonawitz, and Joshua B Tenenbaum. Church: a language for generative models. *arXiv preprint arXiv:1206.3255*, 2012.
- [192] Stan Development Team et al. Stan modeling language users guide and reference manual. *Technical report*, 2016.
- [193] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.
- [194] Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood. An introduction to probabilistic programming. *arXiv preprint arXiv:1809.10756*, 2018.
- [195] Hong Ge, Kai Xu, and Zoubin Ghahramani. Turing: A language for flexible probabilistic inference. In *International conference on artificial intelligence and statistics*, pages 1682–1690. PMLR, 2018.
- [196] Eli Bingham, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman. Pyro: Deep universal probabilistic programming. *The Journal of Machine Learning Research*, 20(1):973–978, 2019.
- [197] Lawrence M Murray and Thomas B Schön. Automated learning with a probabilistic programming language: Birch. *Annual Reviews in Control*, 46:29–43, 2018.

- [198] Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76(1):1–32, 2017.
- [199] Joseph Loxham. *Uncertainty Quantification and Management in Multidisciplinary Design Optimisation*. PhD thesis, Cranfield University, 2019.
- [200] Daniele Bigoni, Allan P Engsig-Karup, and Youssef M Marzouk. Spectral tensor-train decomposition. *SIAM Journal on Scientific Computing*, 38(4):A2405–A2439, 2016.
- [201] Noga Alon and Joel H Spencer. *The probabilistic method*. John Wiley & Sons, 2004.
- [202] Lucien A Schmit and Claude Fleury. Structural synthesis by combining approximation concepts and dual methods. *AIAA Journal*, 18(10):1252–1260, 1980.
- [203] Michael Eldred, Anthony Giunta, and S Collis. Second-order corrections for surrogate-based optimization with model hierarchies. In *10th AIAA/ISSMO multidisciplinary analysis and optimization conference*, page 4457, 2004.
- [204] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.
- [205] Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. CRC press, 2013.
- [206] Richard McElreath. *Statistical rethinking: A Bayesian course with examples in R and Stan*. Chapman and Hall/CRC, 2020.
- [207] Osvaldo Martin. *Bayesian Analysis with Python: Introduction to statistical modeling and probabilistic programming using PyMC3 and ArviZ*. Packt Publishing Ltd, 2018.

-
- [208] Yarin Gal and Mark van der Wilk. Variational inference in sparse gaussian process regression and latent variable models-a gentle tutorial. *arXiv preprint arXiv:1402.1412*, 2014.
- [209] Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.
- [210] Christian P Robert, George Casella, and George Casella. *Monte Carlo statistical methods*, volume 2. Springer, 1999.
- [211] Philip Resnik and Eric Hardisty. Gibbs sampling for the uninitiated. Technical report, Maryland Univ College Park Inst for Advanced Computer Studies, 2010.
- [212] Todd L Graves. Automatic step size selection in random walk metropolis algorithms. *arXiv preprint arXiv:1103.5986*, 2011.
- [213] Patrick Lam. Mcmc methods: Gibbs sampling and the metropolis-hastings algorithm. *Bayesian Statistics in Political Science Research*, 2008.
- [214] Michael Betancourt. A conceptual introduction to hamiltonian monte carlo. *arXiv preprint arXiv:1701.02434*, 2017.
- [215] Olivier Cappé, Simon J Godsill, and Eric Moulines. An overview of existing methods and recent advances in sequential monte carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.
- [216] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*, May 2014. arXiv: 1312.6114.
- [217] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.
- [218] Radford M Neal. Bayesian training of backpropagation networks by the hybrid monte carlo method. Technical report, Citeseer, 1992.

- [219] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.
- [220] Lev Davidovič Landau and Evgenii Mikhailovich Lifshitz. *Mechanics and electrodynamics*. Elsevier, 2013.
- [221] Matthieu Leclair and Gurvan Madec. A conservative leapfrog time stepping method. *Ocean Modelling*, 30(2-3):88–94, 2009.
- [222] John H Mathews. *Numerical methods for mathematics, science and engineering*, volume 10. Prentice-Hall International, 1992.
- [223] Matthew D Hoffman, Andrew Gelman, et al. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.
- [224] Edward Tufte. *The visual display of quantitative information*, 2001.
- [225] Ronald K Pearson. *Exploratory data analysis using R*. CRC Press, 2018.
- [226] Mauricio A Alvarez, Lorenzo Rosasco, and Neil D Lawrence. Kernels for vector-valued functions: A review. *arXiv preprint arXiv:1106.6251*, 2011.
- [227] Irene Tartaruga, Pia Sartor, Jonathan E Cooper, Simon Coggon, and Yves Lemmens. Efficient prediction and uncertainty propagation of correlated loads. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1847, 2015.
- [228] Jan Roskam. *Airplane Design: Preliminary sizing of airplanes*. DARcorporation, 1985.
- [229] Chunyun Niu and Michael Chun-Yung Niu. *Composite airframe structures: practical design information and data*. Adaso Adastra Engineering Center, 1992.
- [230] Ted L Lomax. *Structural loads analysis for commercial transport aircraft: theory and practice*. American Institute of Aeronautics and Astronautics, 1996.

-
- [231] A Bishop, P Messina, et al. Scientific grand challenges in national security: The role of computing at the extreme scale. In *ASCR Scientific Grand Challenges Workshop Series, Tech. Rep*, 2009.
- [232] Gustavo Krupa, Joe Loxham, Timoleon Kipouros, Simon Coggon, Sharma Sanjiv, and A. M. Savill. Efficient prediction and uncertainty propagation of correlated loads. In *Aerodynamics Tools And Methods in Aircraft Design*, 2019.
- [233] Elmer Franklin Bruhn. *Analysis and design of flight vehicle structures*. Tri-state offset company, 1965.
- [234] Mark van der Wilk. Sparse Gaussian Process Approximations and Applications. (November), 2019.
- [235] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When gaussian process meets big data: A review of scalable gps. *IEEE transactions on neural networks and learning systems*, 31(11):4405–4423, 2020.
- [236] Christopher Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Proceedings of the 14th annual conference on neural information processing systems*, number CONF, pages 682–688, 2001.
- [237] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. *Advances in neural information processing systems*, 18:1257, 2006.
- [238] Michalis Titsias and Neil D Lawrence. Bayesian gaussian process latent variable model. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 844–851. JMLR Workshop and Conference Proceedings, 2010.
- [239] James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*, 2013.
- [240] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. *Advances in neural information processing systems*, 18, 2005.

-
- [241] Vern I Paulsen and Mrinal Raghupathi. *An introduction to the theory of reproducing kernel Hilbert spaces*, volume 152. Cambridge university press, 2016.
- [242] H Levesque and G Lakemeyer. Handbook of knowledge representation edited by f. van harmelen, v. Lifschitz and B. Porter, pages 869–886, 2008.
- [243] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization*, pages 507–523. Springer, 2011.
- [244] Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. The application of bayesian methods for seeking the extremum. *Towards global optimization*, 2(117-129):2, 1978.
- [245] Gaëlle Chastaing, Fabrice Gamboa, and Clémentine Prieur. Generalized hoeffding-sobol decomposition for dependent variables-application to sensitivity analysis. *Electronic Journal of Statistics*, 6:2420–2448, 2012.
- [246] Michael A Gelbart, Jasper Snoek, and Ryan P Adams. Bayesian optimization with unknown constraints. *arXiv preprint arXiv:1403.5607*, 2014.
- [247] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.
- [248] Stefano Markidis. The old and the new: Can physics-informed deep-learning replace traditional linear solvers? *Frontiers in big Data*, page 92, 2021.
- [249] Ping Li and Songcan Chen. A review on gaussian process latent variable models. *CAAI Transactions on Intelligence Technology*, 1(4):366–376, 2016.
- [250] Judea Pearl et al. Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146, 2009.
- [251] Judea Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3(September):96–146, 2009.

- [252] Alison L Gibbs and Francis Edward Su. On choosing and bounding probability metrics. *International statistical review*, 70(3):419–435, 2002.
- [253] Neil D Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Nips*, volume 2, page 5. Citeseer, 2003.
- [254] Neil Lawrence and Aapo Hyvärinen. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of machine learning research*, 6(11), 2005.
- [255] Neil D Lawrence. Learning for larger datasets with the gaussian process latent variable model. In *Artificial intelligence and statistics*, pages 243–250. PMLR, 2007.
- [256] Harry J Khamis. Application of the multigraph representation of hierarchical loglinear models. 1996.
- [257] Martin Fowler. *UML distilled: a brief guide to the standard object modeling language*. Addison-Wesley Professional, 2004.