UNIVERSITY OF
LIVERPOOL

Uncertainty Modelling for Scarce and Imprecise Data in Engineering Applications

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Jonathan Cyrus Sadeghi

January 2020

Abstract

In this thesis, models for uncertainty quantification in the case of scarce and imprecise data are described, and the computational efficiency of simulations with these models is improved. Specifically, probability boxes are used to describe imprecision in cumulative distribution functions. This may be the case when imprecise data is used to train a model, or the prior knowledge regarding a property of the system being studied is very weak. Performing simulations with probability boxes is often computationally expensive, because an optimisation program must be solved to obtain each sample in a Monte Carlo simulation. When the system model is known analytically, it is possible to significantly reduce the cost of the analysis. However, the system model is often a *black box* which can only be queried for a particular point value of the input. Each evaluation or query of the system model is often computationally expensive in itself. Currently, few efficient methods exist to perform computations with probability boxes, and the techniques which exist do not provide rigorous bounds on the obtained probability of failure.

Interval Predictor Models are a technique to create an approximate representation of a function, where the uncertainty in the true function is described as an interval, with statistical guarantees on the coverage of the true function. This thesis proposes the use of Interval Predictor Models to create an approximate surrogate model for the true black box system model and hence obtain rigorous bounds on the probability of failure of a system. Techniques are described to create Interval Predictor Models which are tailored to model the performance of a system for reliability analysis. This thesis also describes analytical techniques which can be used for probabilistic safety analysis, in the case that the system model is not a black box. This is advantageous as it enables engineers to perform calculations without spending time programming complex Monte Carlo simulations. A technique is presented to efficiently create Interval Predictor Models for datasets of arbitrary complexity and size, which may contain imprecise data, and we call these models *Interval Neural Networks.* Case studies or numerical examples are presented to demonstrate the performance of the proposed techniques, including some common benchmarks and a finite element model. The interval predictor models used in this thesis were implemented in the open source uncertainty quantification software OpenCossan, and are now freely

available.[*]

[*] https://github.com/cossan-working-group/OpenCossan/tree/development/+opencossan/+metamodels/@IntervalPredictorModel

Acknowledgements

I am grateful for the academic support of my supervisors Dr. Edoardo Patelli and Dr. Marco de Angelis. Edoardo Patelli has introduced me to the field of uncertainty quantification, whilst always giving me the opportunity to grow as a researcher. In particular, he has allowed me to contribute to the software development of OpenCossan, which has provided me with an invaluable educational experience. I am very grateful to him for having taken me on board. Marco de Angelis has provided excellent guidance, even before he joined the supervisory team for my PhD. His explanations of concepts in imprecise probability have had a profound impact on my understanding of the field, and these conversations have shaped my opinion of how reliable engineering can be achieved. In addition, he has frequently provided practical assistance to assist me with the OpenCossan software.

I gratefully acknowledge the financial support of Wood Plc (formerly Amec Foster Wheeler) and the Engineering and Physical Sciences Research Council (EPSRC Grant reference: EP/L015390/1), which enabled me to undertake my studies. In addition, the Next Generation Nuclear centre for doctoral training provided many opportunities to develop my knowledge of the nuclear industry. The Institute for Risk and Uncertainty at the University of Liverpool was also an important part of doctoral studies, where I benefited from frequent seminars and events. The welcoming atmosphere of the Institute provided an enjoyable environment in which to conduct my studies. During my time there, I met many collaborators and friends, who are too numerous to mention on this page and hence I apologise to anyone who I have not mentioned.

The contribution of Prof. Nawal Prinja (Wood Plc) to the work in Chapter 5 is gratefully acknowledged. Specifically, Prof. Prinja was responsible for the description of the structural experiments and devising the structural model to which he applied the First Order Reliability Method. Conversations with Prof. Prinja also inspired many of the developments in Chapter 6. His decades of experience in the nuclear industry have been a great inspiration.

The contribution of Dr. Matthias Faes to Chapter 7, is gratefully acknowledged. Specifically, Dr. Faes contributed the benchmarks to which the Interval Predictor Model methodology was compared in the case study, in addition to a large amount of the text for the original paper, on which this chapter is based (not including novel contributions related to Interval Predictor Models by the present author).

I owe a debt of gratitude to Phoebe, my partner and closest friend, who has been beside me to continually raise my spirits during my PhD. Finally, I wish to thank my family, and in particular my parents, for their invaluable support over the years. They have always pushed me forward at difficult times throughout my studies.

Declaration

I hereby confirm that the results presented in this dissertation are from my own work and that I have not presented anyone else's work and that full and appropriate acknowledgements have been given where references have been made to the work of others.

This dissertation contains 202 pages, 59 figures, 23 tables and 38362 words.

Jonathan Sadeghi

Contents

List of Figures

List of Tables

List of Published Work

The following literature was published containing the work presented in this thesis.

Conference papers

[1] Edoardo Patelli, Matteo Broggi, Silvia Tolo, and Jonathan Sadeghi. Cossan software: A multidisciplinary and collaborative software for uncertainty quantification. In *2nd ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering*, number UNCECOMP 2017, 2017. doi: 10.7712/120217.5364.16982.

[2] Edoardo Patelli, Silvia Tolo, Hindolo George-Williams, Jonathan Sadeghi, Roberto Rocchetta, Marco de Angelis, and Matteo Broggi. Opencossan 2.0: an efficient computational toolbox for risk, reliability and resilience analysis. In *Proceedings of the joint ASCE ICVRAM ISUMA UNCERTAINTIES conference*, 2018.

[3] Jonathan Sadeghi, Marco de Angelis, and Edoardo Patelli. Robust propagation of probability boxes by interval predictor models. In *Proceedings of the joint ASCE ICVRAM ISUMA UNCERTAINTIES conference*, 2018.

[4] Jonathan Sadeghi, Marco de Angelis, and Edoardo Patelli. Analytic imprecise-probabilistic structural reliability analysis. In *ANS Best Estimate Plus Uncertainty International Conference (BEPU 2018)*, 2018.

[5] Jonathan C Sadeghi, Marco de Angelis, and Edoardo Patelli. Efficient computational structural reliability analysis of concrete containments. In *2nd International Conference on Nuclear Power Plants: Structures, Risk & Decommissioning*, 2018.

[6] Jonathan C Sadeghi, Marco de Angelis, and Edoardo Patelli. Efficient training of neural networks with interval uncertainty. In *Reliable Engineering Computing*, 2018.

Journal papers

[1] Matthias Faes, Jonathan Sadeghi, Matteo Broggi, Marco de Angelis, Edoardo Patelli, Michael Beer, and David Moens. On the robust estimation of small failure probabilities for strong non-linear models. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems Part B: Mechanical Engineering*, 2019. doi: 10.1115/1.4044044.

[2] Nawal K. Prinja, Azeezat Ogunbadejo, Jonathan Sadeghi, and Edoardo Patelli. Structural reliability of pre-stressed concrete containments. *Nuclear Engineering and Design*, December 2016. doi: 10.1016/j.nucengdes.2016.11.036.

[3] Jonathan Sadeghi, Marco de Angelis, and Edoardo Patelli. Analytic probabilistic safety analysis under severe uncertainty. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, 2019. doi: 10.1061/AJRUA6.0001028.

[4] Jonathan Sadeghi, Marco de Angelis, and Edoardo Patelli. Efficient training of interval neural networks for imprecise training data. *Neural Networks*, 2019. doi: 10.1016/j.neunet.2019.07.005.

[5] Jonathan Sadeghi, Marco de Angelis, and Edoardo Patelli. Robust propagation of probability boxes by interval predictor models. *Structural Safety*, 2019. doi: 10.1016/j.strusafe.2019.101889.

Open source software

[1] Jonathan Sadeghi. Open Cossan IPM Toolbox. `https://github.com/cossan-working-group/OpenCossan/tree/development/+opencossan/+metamodels/@IntervalPredictorModel`, 2018.

[2] Jonathan Sadeghi. PyIPM. doi: 10.5281/zenodo.3349670, `https://github.com/JCSadeghi/PyIPM`, 2018.

[3] Jonathan Sadeghi. Efficient Training of Interval Neural Networks for Imprecise Training Data Experiments. `https://github.com/JCSadeghi/Efficient-Training-of-Interval-Neural-Networks-for-Imprecise-Training-Data`, 2019.

Nomenclature

List of Abbreviations

ℓ_p norm pth root of the sum (or integral) of the pth-powers of the absolute values of the vector components

ADAM Adaptive moment estimation

AK-MCS Adaptive Kriging Monte Carlo simulation

BARC Bhabha Atomic Research Centre

BARCOM Bhabha Atomic Research Centre containment test model

CDF Cumulative distribution function

cov Covariance

DGM Data generating mechanism

DOF Degrees of freedom

DS-Structure Dempster-Shafer structure

eCDF Empirical cumulative distribution function

FAST Fourier amplitude sensitivity testing

FAST algorithm Fast algorithm for the scenario technique

FORM First order reliability method

GPU Graphics processing unit

IPM	Interval predictor model
MAP	Maximum a posteriori
MCMC	Markov chain Monte Carlo
ML	Maximum likelihood
MLP	Maximum likelihood perceptron
MPa	Mega Pascals
MSE	Mean squared error
NASTRAN	NASA STRucture ANalysis (software)
nMSE	Normalised mean squared error
PCCV	Pre-stressed concrete containment vessel
PDF	Probability distribution function
PSA	Probabilistic safety analysis
QUAD4 element	4 nodes quadrilateral element
RBDO	Reliability based design optimisation
ReLU	Rectified linear unit
SNL	Sandia National Laboratories
SNS	Standard normal space
VC dimension	Vapnik-Chervonenkis dimension

List of Symbols

$\bar{P}_f$	Upper bound failure probability
β	Reliability index

$\delta(x)$	The Dirac delta function
$\mathbb{I}$	Indicator function
$\mathbb{R}$	Set of real numbers
$\mathcal{L}$	A loss function
$\mathcal{N}$	Normal distribution
$\mathcal{X}$	A dataset
$\mathbb{E}$	Expectation operator
Var	Variance operator
$\underline{P}_f$	Lower bound failure probability
CoV	Coefficient of Variation
$F(x)$	Cumulative distribution function
$g'(\boldsymbol{x})$	Derivative of $g(\boldsymbol{x})$
$g(\boldsymbol{x})$	Performance function
$P(\cdot)$	Probability operator
$p(x)$	Probability density function
P_d	Design pressure
P_f	Probability of failure
$S_n(x)$	Empirical cumulative distribution function

Chapter 1

Introduction

1.1 Motivation

1.1.1 Uncertainty in engineering simulation

Since the advent of the computer, engineering simulation has become an invaluable tool for the design and analysis of systems. Simulation allows engineers to bridge the gap between theoretical models of systems and empirical evidence, whilst making predictions about yet to be constructed systems [136]. Some important applications of this include structural engineering [168], aeronautical structures [168], and petroleum reservoir engineering [154].

The engineer's theoretical model of the system's physics can be defined by a mathematical function or a more complex simulation, and this theoretical model will depend upon associated parameters which determine the specific properties of the system under consideration. The physical model to be used may be motivated by the engineer's expert judgement, or prescribed by a relevant design standard document. Provided the model's parameters are known, the model can be used to make predictions about the system. In some cases, e.g. well known material properties, the parameters to be used will also be prescribed by the design standard document. If this is not the case, the engineer must identify these parameters from data or expert judgement. Hence, in many realistic situations these parameters will not be known exactly, and therefore will be associated with some uncertainty. If the system is not well understood, then the theoretical model of the system's physics may itself be uncertain. However, in many cases this situation can be dealt with by adding more uncertain parameters to the model, thereby increasing the model's degrees of freedom. This uncertainty will be reflected in the predictions made by the model, hence the ability of the system to meet some specified objective, e.g. safe operation, now becomes uncertain. In essence, this motivates the well known structural reliability analysis problem, where we wish to calculate the probability that the system under uncertainty doesn't meet a specified objective, which is referred to as the failure probability of the system [115].

1.1.2 Reliability engineering

Researchers in the discipline of reliability engineering have proposed many techniques to solve the reliability analysis problem. Most generally, Monte Carlo simulation can solve any reliability analysis problem with arbitrary accuracy, given sufficient samples of the uncertain system parameters and evaluations of the system model [115]. If the failure probability of the system is small, which is typical in most realistic engineering problems, then the number of model evaluations required increases significantly, and creates a bottleneck to the calculation. Therefore, in practice, more efficient methods are required to solve the reliability analysis problem for expensive computational models with many inputs. These include approximate methods, e.g. the First Order Reliability Method (FORM) and advanced simulation techniques, e.g. line sampling, which require fewer samples of the system model [168, 50].

Alternatively, since the cost of the analysis depends strongly upon the cost of evaluating the system model, one may attempt to replace the expensive system model with a cheaper surrogate, known as a metamodel. This metamodel is usually obtained by using machine learning technologies to learn a function which is a sufficiently accurate representation of the true model. Well known metamodels applied in reliability engineering include neural networks, response surfaces (polynomial regression), polynomial chaos, and Kriging (Gaussian process emulators) [168, 157, 76]. If the metamodel is inaccurate then this can introduce additional uncertainty into the calculation, and typically this must be traded off against time required to create the metamodel. In any case, the uncertainty in the metamodel should be quantified and its influence on the failure probability of the system stated. As such, the problem is challenging and does not yet have an entirely satisfactory solution, although significant progress has been made in recent years.

1.1.3 Models of uncertainty

Many techniques exist for modelling uncertainty, and therefore the chosen uncertainty model is also, to an extent, an engineering judgement. This judgement is usually based on the type of uncertainty being modelled, and usually two types of uncertainty are considered; epistemic uncertainty and aleatory uncertainty [68, 50]. Broadly speaking, epistemic uncertainty represents uncertainty which originates from a lack of knowledge, and aleatory uncertainty represents uncertainty which originates from natural variability, i.e. stochasticity.* Again, some guidelines are available in design standards or regulations, for example the United States Nuclear Regulatory Commission often suggests using Bayesian probability theory [175, 176].

Bayesian probability theory is a logically consistent method of reasoning under uncertainty, though it has been shown to lack empirical justification in some circumstances, e.g. Balch et al. [11]. Efficient computational methods exist to identify many probabilistic models for uncertain variables in the Bayesian paradigm, in addition to convenient analytic techniques [88]. In recent decades, several extensions to the traditional probabilistic models for uncertainty have been proposed, e.g. Dempster-Shafter Theory, probability boxes, and random sets [50, 60] (often referred to as specific manifestations of *imprecise probabilities*). These methods enable reasoning with imprecise data and a severe lack of prior information.

Imprecise data consists of data where each measurement is not specified by a real number (sometimes referred to as crisp measurements), but instead the data falls within certain bounds which can be characterised. Scarce data refers to the case where insufficient data is available to accurately identify an unknown model parameter. Therefore our knowledge of the parameter places undue weight on our prior belief about the parameter. In such cases an engineer may wish to check the sensitivity of their model's predictions to the chosen prior, and it is therefore essential that the engineer can accurately represent uncertainty in their prior belief about a parameter [16, 18]. Crucially, imprecise probabilities offer a method of reasoning with uncertainty which is more flexible and hence requires fewer assumptions than traditional probabilistic methods.

*Note that in some cases this distinction is unclear. For example when a very simple model is used for the behaviour of a coin, the outcome of the coin flip may appear to be random. However, one could imagine a situation where the kinematics of the coin can be simulated exactly using Newton's laws of mechanics, and the only uncertainty in the outcome of the coin flip is caused by lack of knowledge in the coin's initial position and velocity. Chaotic systems (e.g. the Lorenz attractor), where the future evolution of the system depends strongly on the initial conditions, may appear to be random, for example when the model considered for the system is insufficiently detailed and the initial conditions are not known with sufficient accuracy.

Despite these advantages, working with imprecise probabilities in practice introduces some new difficulties. Since the uncertainty model is more complex, the computational techniques required to perform computations are also more complex. Typically, this computation involves some optimisation in addition to the already expensive Monte Carlo simulation [50]. Hence, in recent years, researchers have described techniques to solve the reliability analysis problem with imprecise probability models which apply similar approximations to those used in the straightforward probabilistic case.

In this introduction we will focus our discussion of uncertain model parameters on probability boxes, but we will see later in the thesis that different models of imprecise probability have much in common, and are sometimes interchangeable.

Since the predictions of metamodels used in reliability analysis (as described in Section 1.1.2) are unlikely to be perfect, their uncertainty must be modelled. Machine learning practitioners typically achieve this by using conventional probability theory. Unfortunately, this sometimes requires strong assumptions to be made about the form of the function being represented. Therefore, Interval Predictor Models (IPMs) have recently been proposed by Campi et al. [34] as a method of quantifying uncertainty in machine learning, which relax some common assumptions required by probabilistic methods. An interval predictor model is any model which predicts an interval, as opposed to a point value. Sophisticated techniques relying on the advances of scenario optimisation can be used to guarantee bounds on the probability that the prediction interval contains the desired 'true' value of the modelled function [29].

1.2 Problem definition and objectives

For the purpose of this introduction, probability boxes can be considered to be a representation of a set of contiguous cumulative probability distribution functions (in fact, this is a specific class of probability boxes and a more general definition is given in Chapter 2). Although probability boxes have existed for some time [60], efficient methods for performing calculations with probability boxes on expensive computational models have only started to appear in the literature in the past decade. Therefore unsurprisingly, their use in industry with expensive computational models is not yet widespread, particularly relative to traditional probabilistic models. However, this lack of adaptation is also true of calculations with probability boxes for more simple systems, which do not require significant computational resources for analysis. The following possible explanations are proposed for this:

- Most of the computational methods available are still too slow for practical usage on large systems.
- The efficient methods which are available rely upon approximations and are therefore not trusted by engineers in safety critical domains.
- The methods available are not implemented in readily available software, and developing bespoke software for one-off usage may be economically impractical and too time consuming.

Note that for a particular calculation only a subset of these issues may apply. Increasing the usage of these techniques is a fundamental aim of the imprecise probability community [182]. Ensuring that these techniques exist and are used is important due to the obvious economic and societal advantages of enabling engineering that is one the one hand safe, and on the other hand not excessively conservative in its design. Overly conservative designs are undesirable because, by definition, they result in excessive construction and manufacture costs, which may prevent the benefits of new technology being realised by society.

This thesis aims to mitigate these issues via the following objectives:

1. Propose computational methods, which provide statistically rigorous and theoretically guaranteed bounds on the propagation of probability distributions and, by generalisation, probability boxes in specific calculations, by applying interval predictor model metamodels.
2. Decrease the computational cost of interval predictor models for imprecise data, whilst enabling more complex Interval Predictor Models to be constructed.
3. Provide analytical methods for some common probabilistic safety analysis calculations with probability boxes, which do not require computer simulation.
4. Demonstrate the applications of the developed techniques on calculations relevant to the nuclear industry.

1.3 Structure of this thesis

In order to address the objectives discussed in the previous section the following nine chapters are presented.

Chapter 2 reviews uncertainty models, i.e. models describing the uncertainty in a variable, or set of variables. We review techniques to construct these models from data, and expert opinion, and describe techniques for converting between common models. This chapter describes probability boxes in detail, and shows how traditional probability distributions emerge as a particular case of a probability box.

Chapter 3 reviews machine learning techniques for creating regression models, which are used as metamodels in engineering. Regression models differ from the uncertainty models described in Chapter 2, as they model the behaviour of a uncertain variable which depends on the behaviour of another variable. In this chapter, the theory behind interval predictor models is described in detail, and compared to traditional techniques in statistical learning.

Chapter 4 describes the well known reliability analysis problem in engineering, where one wishes to calculate the probability that the performance of a system under the influence of uncertainty meets a particular design condition. State-of-the-art techniques for efficiently solving this problem with random variables and probability box variables are reviewed.

In Chapter 5, we describe the analysis of an industrial test case from the nuclear sector, where a concrete containment for a nuclear reactor was pressurised to failure. The containment is modelled using an analytic equation derived from structural engineering principles. The properties of the containment are represented by random variables where the distribution parameters are not precisely known, i.e. probability boxes. We apply sensitivity analysis using Monte Carlo simulation to study the effect of changing these distribution parameters.

Chapter 6 provides equations to solve the problem described in Chapter 4 analytically for some particularly important and commonly occurring system configurations for probabilistic safety analysis in the nuclear industry. A generalisation of the equations is presented for the case where the system's parameters are represented by probability boxes. Then these developments are applied to analytically calculate the reliability of the containment from Chapter 5 and an additional containment test case, i.e. without using Monte Carlo simulation.

Chapter 7 utilises the interval predictor models introduced in Chapter 3 as a metamodel for solving the reliability analysis problem in Chapter 4. The results are compared to other state of the art techniques involving metamodels.

In Chapter 8, the results from Chapter 7 are extended to solve the reliability analysis problem for the case where the system's parameters are described by probability box variables, by using an interval predictor model metamodel.

Chapter	Content	Objective
1	Introduction	
2	Review of Literature	
3	Review of Literature	
4	Review of Literature	
5	Novel Contribution	4
6	Novel Contribution	3
7	Novel Contribution	1
8	Novel Contribution	1
9	Novel Contribution	2
10	Conclusion	

Table 1.1: Content of thesis chapters.

Chapter 9 demonstrates novel methods of training interval predictor models with complex structures on huge data sets. This is achieved by using neural networks to represent the bounds of the interval predictor model, and applying a new loss function to ensure robust and efficient training.

Finally, Chapter 10 summarises the presented work, and makes recommendations for future research.

The content of the chapters is summarised in Table 1.1. The structure of the thesis is illustrated by the flowchart in Figure 1.1.

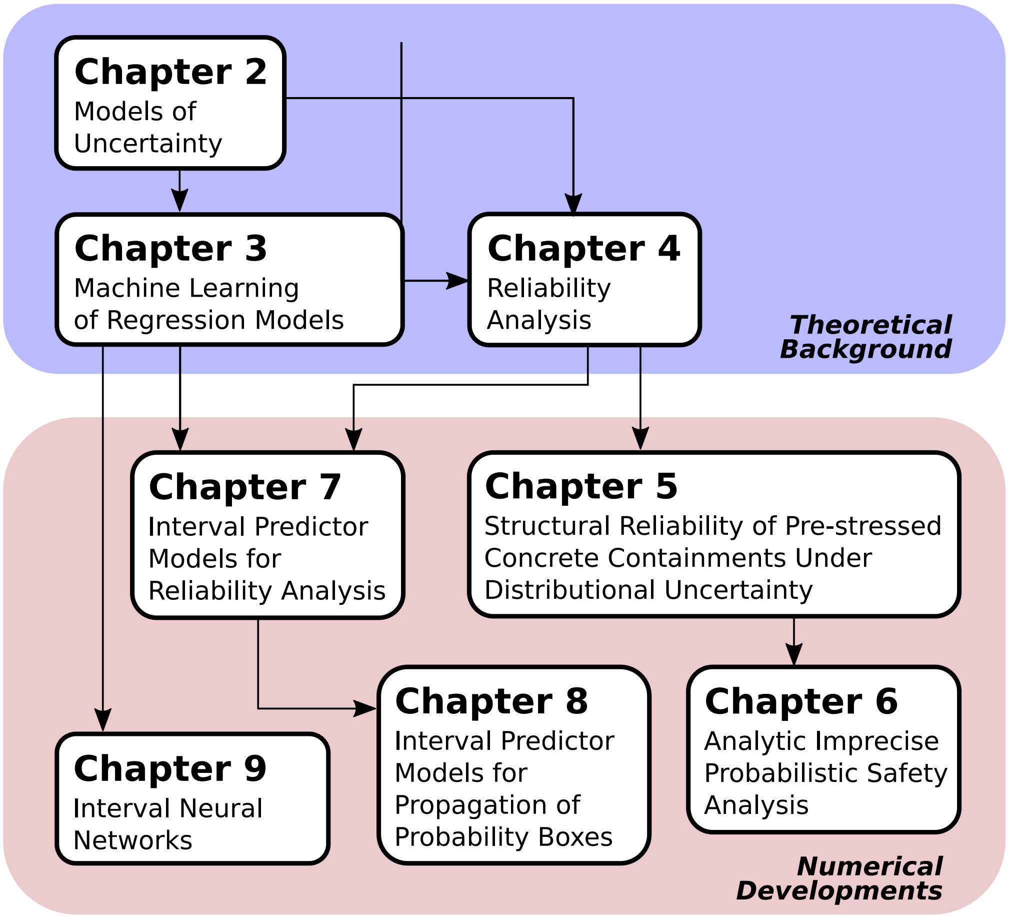
Chapter 2
Models of Uncertainty
Chapter 3
Machine Learning of Regression Models
Chapter 4
Reliability Analysis
Theoretical Background
Chapter 7
Interval Predictor Models for Reliability Analysis
Chapter 5
Structural Reliability of Pre-stressed Concrete Containments Under Distributional Uncertainty
Chapter 8
Interval Predictor Models for Propagation of Probability Boxes
Chapter 6
Analytic Imprecise Probabilistic Safety Analysis
Chapter 9
Interval Neural Networks
Numerical Developments

Figure 1.1: Organisation of thesis chapters.

Chapter 2

Models of Uncertainty

2.1 Overview of uncertainty models

The purpose of uncertainty models is to allow analysts to move beyond computation with point values of variables. By using an uncertainty model one can describe the variability or lack of knowledge in a parameter, and consider how this affects the result of a calculation. In this chapter, we describe the process of quantifying the uncertainty in a set of variables independently of the values of other variables, which is known as *generative modelling* (the alternative case, *regression modelling* is described in the subsequent chapter).

As a concrete example, consider predicting the weather at a particular location, on a particular day. A model to predict the weather on Tuesday independent of other information would be considered a generative model. If the model predicted the weather on Tuesday, given the weather conditions on Monday, which were already known, this would be termed a regression model.

So far we have avoided discussing any type of uncertainty model in particular, since many different models of uncertainty exist. The particular model chosen for an application may be selected based on a number of factors, including its ability to be created from the type of data available, ease of computation, and the desired properties of predictions made by the model.

Models which can be created from imprecise data, i.e. interval data, often result in less informative predictions than models which cannot. However, uninformative predictions are not necessarily undesirable, if the predictions are a true representation of the state of knowledge of the analyst or engineer. Hence the chosen uncertainty model should truthfully represent the level of uncertainty of the analyst, given the available information.

It should also be noted that, contrary to common opinion, particular classes of uncertainty model do not necessarily restrict the type of uncertainty which can be represented. In some cases, a probabilistic model may allow epistemic uncertainty to be communicated [88], and in principle one could use an interval model to represent aleatory uncertainty — though usually probabilistic models are more useful representations of aleatory uncertainty.

The ease of calculation should also be considered when the class of uncertainty model to be used is chosen. If the speed of calculation is very important, because a decision based on the predictions of the model must be made with severe time limitations, then it could actually be *more unsafe* to choose an uncertainty model which is *more accurate* but reduces the speed with which predictions can be made. For example, this could be the case in an online safety system in a power plant [172]. However, in most situations the speed with which predictions must be made is more flexible, and therefore it is usually preferable to choose an uncertainty model which is accurate, and then use approximations or efficient computational methods to compute the desired prediction.

With this in mind, in this section several uncertainty models will be presented, including probabilistic models, non-probabilistic models, and imprecise probabilistic models. The type of information which can be used to create the models will be discussed, and their amenabilities to calculation will be compared.

2.1.1 Probabilistic models of uncertainty

Overview

Probabilistic models of uncertainty require the definition of a probability space, which is given by $(\Omega, \mathcal{F}, P)$, where Ω represents the space of all possible outcomes (the sample space), $\mathcal{F}$ is a σ algebra representing the set of events, where each event is a set containing outcomes, and $P \in [0, 1]$ represents probabilities which are assigned to the events in $\mathcal{F}$ [88]. Note that the probability assigned to the space of all outcomes is 1, $P(\Omega) = 1$, and the probability of the empty set of outcomes is zero, $P(\emptyset) = 0$. By definition, for an outcome $\omega \in \Omega$ and its complementary outcome ω^C, their intersection is the empty set, $\omega \cap \omega^C = \emptyset$, and their union is equal to the entire sample space, $\omega \cup \omega^C = \Omega$. It follows that $P(\omega) + P(\omega^C) = 1$.

The aim of defining the probability space is usually to obtain the random variable $X : \Omega \to E$, which is a function from the set of outcomes Ω to the measurable space E.

The probability that X takes a value falling inside in the set S is given by

$$\Pr(X \in S) = P(\{\omega \in \Omega | X(\omega) \in S\}). \tag{2.1}$$

When the measurable space is the real line, $E = \mathbb{R}$, the cumulative distribution function (CDF) can be obtained by calculating the probability of the event that that X is less than a particular value x, $\{X < x\}$. Therefore the cumulative distribution function is given by

$$F_X(x) = P(X \leq x). \tag{2.2}$$

The well known probability distribution function (PDF) of the variable, $p_X(x)$, is the gradient of the cumulative distribution function, $p_X(x) = \frac{dF_X(x)}{dx}$. The probability density represents the relative likelihood of a random variable taking a particular value. Note that, as follows from our definition, the cumulative distribution must increase monotonically with x, and as such the probability density function is always positive. In addition, the integral of the probability density function over the whole real line will always be equal to 1, since $\lim_{x\to\infty} F_X(x) = 1$ and $\lim_{x\to-\infty} F_X(x) = 0$ [149].

Note that important properties of the random variable are summarised by the mean of the random variable,

$$\mu_X = \mathbb{E}[X] = \int_{\mathbb{R}} x p_X(x) dx, \tag{2.3}$$

where $\mathbb{E}$ is the expectation operator, and the variance

$$\mathrm{Var}(X) = \mathbb{E}[(X - \mu_X)^2] = \int_{\mathbb{R}} (x - \mu_X)^2 p_X(x) dx, \tag{2.4}$$

which is sometimes quoted in terms of the standard deviation, $\sigma_X = \sqrt{\mathrm{Var}(x)}$ [149]. The standard deviation is sometimes expressed as the coefficient of variation of a varible (CoV), which is defined as $\mathrm{CoV} = \frac{\mu_X}{\sigma_X}$.

In practical calculations, one is not usually concerned with the probability space $(\Omega, \mathcal{F}, P)$, because defining either the probability distribution function or cumulative distribution function by assigning probability density to the possible values for X is usually sufficient for calculations to proceed. The measure theoretic framework for probability theory which we have summarised here was introduced by Kolmogorov [102]. An equivalent theory was derived by extending Boolean logic to assign quantitative values of truthfulness to statements, termed plausibilities, in the seminal work of Jaynes [88].

The probability of an event may be considered in terms of probabilities of sub-events, for example if $A = A_1 \cap A_2$, where $\cap$ represents the logical and operation, then $P(A) = P(A_1 \cap A_2)$. This can be evaluated as $P(A_1 \cap A_2) = P(A_1|A_2)P(A_2)$, by definition. $P(A_1|A_2)$ is the probability that A_1 is true, given that A_2 is true. The dependence between A_1 and A_2 is encoded in $P(A_1|A_2)$; $P(A_1|A_2) = P(A_1)$ if A_1 and A_2 are independent events, so that $P(A) = P(A_1)P(A_2)$. If the dependence is not known then bounds for $P(A)$ can be established, and this is discussed in Chapter 6. Similar bounds are available for the logical or operation, $\cup$.

One may model dependencies between variables by considering the joint distribution over more than one variable, e.g. $p(x_1, x_2) = P(X_1 = x_1 \cap X_2 = x_2)$, where X_1 and X_2 are two random variables. Traditionally, a generative model is defined as a joint probability distribution. One may summarise the properties of a joint distribution using the co-variance between two variables

$$\operatorname{cov}(X_1, X_2) = \mathbb{E}[(X_1 - \mu_{X_1})(X_2 - \mu_{X_2})] = \int_{\mathbb{R}} \int_{\mathbb{R}} (x_1 - \mu_{X_1})(x_2 - \mu_{X_2}) p(x_1, x_2) dx_1 dx_2. \tag{2.5}$$

Taking the co-variance of a variable with itself yields the variable's variance, $\operatorname{cov}(X_1, X_1) = \operatorname{Var}(X_1)$ [149].

Computation with Probabilistic Models

The expectation (mean value) of a general function $g(x)$ with respect to an uncertain variable, which is modelled with a probability distribution $p(x)$, can be evaluated as

$$\mathbb{E}[g(X)] = \int_{\mathbb{R}^N} g(x)p(x)dx, \tag{2.6}$$

where we now allow x to have multiple components.

In the most general case, one can draw realisations, known as samples, from the random variable by selecting values for the variable in the ratio of their assigned probability densities. Given M samples drawn from $p(x)$, $\{x_{(1)}, \dots, x_{(M)}\}$, $\mathbb{E}[g(X)]$ can be approximated by the Monte Carlo estimator

$$\mathbb{E}[g(X)] \approx I = \frac{1}{M} \sum_{i=1}^{M} g(x_{(i)}). \tag{2.7}$$

The error of the Monte Carlo estimator is given by

$$\frac{\sigma_g}{\sqrt{M}} = \frac{\sqrt{\sum_{i=1}^{M}(g(x_{(i)}) - I)^2}}{\sqrt{M(M-1)}}, \tag{2.8}$$

where σ_g is the unbiased sample estimator of the variance of g [150]. Clearly the error in the estimator decreases with $\frac{1}{\sqrt{M}}$, so collecting more samples will slowly decrease the error in the estimator [140]. However in many circumstances collecting more samples of M is not feasible, and σ_g may be large.

Therefore, in practice, more efficient methods are used to compute the expectation. In Chapter 4 efficient methods for solving a specific form of this integral from engineering reliability analysis will be explained. However, it is useful to note there are some more general efficient approaches for computing an approximation of the expectation in Eqn. 2.6.

For example, provided that $g(x)$ can be differentiated, one can attempt to approximate the expectation in Eqn. 2.6 with a Taylor expansion

$$\mathbb{E}[g(X)] = \mathbb{E}[g(\mu_X + (X - \mu_X))] \approx \mathbb{E}[g(\mu_X) + g'(\mu_X)(X - \mu_X) + \frac{1}{2}g''(\mu_X)(X - \mu_X)^2 \ldots], \tag{2.9}$$

where μ_X is the mean of random variable X. When $g(x)$ is linear, the the expectation in Eqn. 2.6 can therefore be calculated trivially, since $X - \mu_X = 0$ and $g''(x) = 0$. If $g(x)$ is non-linear then using the approximation $\mathbb{E}[g(X)] \approx \mathbb{E}[g(\mu_X)]$ is correct only to first order [179]. At the expense of generality, the expectation is evaluated using only two evaluations of $g(x)$, which is a clear advantage when $g(x)$ is expensive to evaluate, or the computation of $\mathbb{E}[g(X)]$ is required quickly. This idea is applied to engineering in the Kalman Filter and Extended Kalman Filter [179].

Uhlmann [174] proposed a more accurate approximate method for approximating the expectation in Eqn. 2.6 which requires few evaluations of $g(x)$, and does not require $g(x)$ to be differentiated. The Unscented Transformation, sometimes referred to as deterministic sampling, requires the computation of so called sigma points which are specially chosen represent the covariance of the input distributions. It is then only required to compute $g(x)$ for these sigma points, after which a weighted average is used to obtain the approximation of the expectation in Eqn. 2.6. Wan and Van Der Merwe [179] demonstrate that the Unscented Transformation is accurate to at least second order, but potentially to third or forth order in some cases. The number of sigma points required, and hence the cost of the computation, depends linearly on the dimensionality of x. Although this approach is more generally applicable and accurate than the Taylor Series method, the Unscented Transformation is limited to use cases where x has reasonably low dimensionality, because otherwise too many evaluations of $g(x)$ are required. The method has been applied to engineering in the Unscented filter [179] and uncertainty quantification in the nuclear industry by Zhang et al. [188] and Perret et al. [139].

In order to avoid these approximations, techniques can be used to obtain more accurate approximations of $g(x)$ in specific cases (metamodels), or alternatively efficient sampling strategies can be used. This is discussed in Chapter 4 for application to reliability analysis.

2.1.2 Set-based models of uncertainty

The two most common set-based models of uncertainty are convex sets and interval models. The popularity of these models can be attributed to the computational simplifications they enable, relative to non-convex set models of uncertainty. Set-based models are a method of characterising uncertainty, without describing the level of belief for each distinct value, as in probabilistic models. Alternatively, they can be seen as an initial way of prescribing the support for an as-yet undetermined probabilistic model.[*]

Interval models of uncertainty

An interval model of uncertainty is a set of numbers where any number falling between the upper and lower bounds of the interval is included in the set. In the interval notation, an interval uncertainty model is given by $X = [\underline{x}, \overline{x}] = \{x \in \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}\}$ [121]. Note that interval models can also be defined for the case where the endpoints are excluded from the intervals, and in this case regular brackets are used instead of square brackets. Interval models can be re-parameterised in terms of their centre and interval radius, $c = \frac{\overline{x}+\underline{x}}{2}$ and $r = \frac{\overline{x}-\underline{x}}{2}$. Interval models usually don't express dependency between multiple variables.

Note that when using an interval model of uncertainty, the relative likelihoods of different values of the variable are unknown and therefore it is impossible to draw precise samples from the model in the same sense that one draws samples from a probability distribution. Put simply, an interval model represents complete lack of knowledge, apart from the bounds. For this reason, they are often preferred for modelling severe epistemic

[*] The support of a distribution is the set of values for which the probability measure is non-zero.

uncertainty, sometimes known as incertitude, rather than aleatory uncertainty, where the ability to model the variability of a quantity is the main focus [104].

Basic computation with interval models is typically computationally inexpensive since one can rely upon interval arithmetic for elementary operations. Complex functions and computer codes can be ‘converted’ to accept interval inputs by applying the so-called natural extension, where arithmetic operations are converted to interval arithmetic and real valued inputs are replaced with interval values [121]. However, applying the natural interval extension can result in gradual widening of the prediction interval during propagation through the code, due to repeated appearance of the same variables [121].

In many cases, one can create a Taylor expansion for $g(x)$ up to terms of a specific order in x, and then use an interval bound for the rounding error to rigorously and accurately bound $g(x)$ [81, 113, 124]. Intermediate linear Taylor models may be combined together to yield relatively tight bounds on the output with reduced computational expense, since computing the maximum of a multivariate linear function over a set of intervals is trivial. Engineers could choose to neglect the modelling of the remainder in cases where a rigorous bounding of the range of $g(x)$ over X is not required, or the Taylor model is sufficiently accurate. Alternatively, if $g(x)$ is represented a Bernstein polynomial, bounds on the range of $g(x)$ over X can be obtained analytically [45].

For complex functions or computer codes it may not be possible to rewrite the code in terms of interval arithmetic, as the code may be ‘black-box’ and therefore inaccessible. In these cases it may be necessary to rely upon numerical optimisation to compute approximate bounds on the range of $g(x)$. For example, if one wishes to make predictions about the output of a model $g(x)$ for $x \in [\underline{x}, \overline{x}]$ then one must evaluate

$$[\min_{x \in [\underline{x}, \overline{x}]} g(x), \max_{x \in [\underline{x}, \overline{x}]} g(x)]. \tag{2.10}$$

For general $g(x)$, a non-linear optimiser with the ability to respect bounds must be used to solve Eqn. 2.10, for example Bayesian optimisation or a genetic algorithm. A brute force search could also be used to solve Eqn. 2.10. If $g(x)$ is convex, then one may apply convex optimisation routines [22]. Due to the curse of dimensionality this optimisation becomes more difficult when there are many interval variables to be propagated, since the complexity for many convex optimisation algorithms increases with the dimensionality of the problem [26].

The method of Cauchy deviates offers an efficient alternative to computing finite difference gradients of $g(x)$ in order to apply a Taylor expansion for black-box functions [103]. The Cauchy deviates method evaluates $g(x)$ for a set of transformed samples drawn from Cauchy distributions and then uses the property that the linear combination of Cauchy distributed random variables is also Cauchy distributed with a known distribution parameter to bound $g(x)$.

Convex set models of uncertainty

A convex set is defined as a set where, for any two points in the set, all points along the connecting line between the two points are also included in the set. Convex sets are useful as they are in many ways similar to interval models, but allow dependencies to be modelled between variables. Ben-Haim and Elishakoff [15] provide examples of how convex set models may be used in engineering practice. There is a deep connection between interval models and convex set models. An interval model with multiple variables would be represented as the specific case of a hyper-rectangular convex set. In addition, affine transformations of hyper-rectangular convex sets result in a class of models known as zonotopes [163].

The smallest convex set containing a particular set of data points is termed the convex hull of the dataset. Computing the convex hull of a dataset has complexity $\mathcal{O}(n \log n + n^{\lfloor \frac{d}{2} \rfloor})$ [41], where $\lfloor \cdot \rfloor$ is the floor operator which rounds a real number down to the nearest integer, and therefore in practice one often learns a simplified representation of the convex set with desirable computational properties.

ℓ_p ellipsoids (for $p > 1$) are a particularly useful case of a convex set, because they can be easily manipulated in calculations [180]. An ℓ_p ellipsoid is given by the set $X = \{x \mid \|x - c\|_{p,w} \leq r\}$, where the weighted ℓ_p norm is $\|x\|_{p,w} = (\sum_{i=1}^{N} |w_i x_i|^p)^{\frac{1}{p}}$, x_i are variables in a set of dimensionality N, $c \in \mathbb{R}^N$ represents the centre point of the set, and w_i are weights controlling the relative uncertainty in each variable [85]. p can be adjusted to control the correlation between the variables. The case ℓ_∞ corresponds to the hyper-rectangle (interval) model, where there is no correlation between variables. When p is decreased the variables become more correlated.

Computation with convex set models is performed in the same way as with interval models, except the bounds on x are replaced with convex constraints:

$$[\min_x \{g(x) : x \in X\}, \max_x \{g(x) : x \in X\}], \tag{2.11}$$

where X is a convex set. If $g(x)$ is linear with known gradient, and X is a hyper-sphere or hyper-rectangle, then this optimisation program in Eqn. 2.11 admits an analytic solution [116]. If the convex set is small then the analytic solution may be a reasonable solution when used with a Taylor expansion for $g(x)$, as with interval uncertainty models.

2.1.3 Imprecise probabilistic models of uncertainty

Probability Boxes

Probability boxes generalise probability distributions and intervals; they model a set of cumulative distribution functions. Probability boxes are used to communicate epistemic uncertainty in the precise form of a probability distribution [60]. In the particular limiting cases of no epistemic uncertainty and no aleatory uncertainty, traditional CDFs and intervals can be recovered from the probability box, respectively.

Broadly speaking, probability boxes can be split into two types. Distribution-free probability boxes consist of an envelope defined by two CDFs. Any CDF contained within the envelope is permitted, i.e. the probability box contains all cumulative distribution functions $F(x)$ which satisfy the envelope condition $\underline{F}(x) \leq F(x) \leq \overline{F}(x) \forall x$.

Distributional probability boxes consist of a conventional probability distribution where at least one parameter of the distribution is given as an interval rather than a crisp value, i.e. the probability box is given by the probability distribution $p_\theta(x)$ with parameters $\theta \in \Theta$, where Θ is a hyper-rectangular convex set. It is possible to perform a conversion from distributional probability boxes to distribution-free probability boxes, by finding a distribution-free probability box which encloses the distributional probability box. This conversion results in the loss of information about the distribution types enclosed, and hence the conversion can not be easily reversed. The envelope of a distributional probability box, which will be a distribution-free probability box, can be obtained by evaluating

$$[\underline{F}(x), \overline{F}(x)] = [\min_\theta F_\theta(x), \max_\theta F_\theta(x)], \tag{2.12}$$

where $F_\theta(x)$ is the CDF corresponding to the probability distribution $p_\theta(x)$.

Computation with Probability Boxes

Probability boxes are a specific case of a random set [50, 3], and therefore when they are propagated through a calculation their propagation can be decomposed into two distinct parts; the propagation of a set of epistemic uncertain variables which fall within an uncertain hyper-rectangle $\theta \in \Theta$, and the propagation of an aleatory set of variables α, which are associated with a probability distribution. The result of propagation through the model is a probability box, rather than a single CDF. The propagation is non-trivial since the epistemic variables are intervals and have no probability distribution, which means that conventional Monte Carlo simulation cannot be applied. Two methods are commonly used to propagate probability boxes: Double Loop Monte Carlo (sometimes referred to as search or optimisation of the epistemic space), and integration of the aleatory variables [133].

For a distributional probability box the upper and lower expectation are defined by

$$\overline{\mathbb{E}}[g(X)] = \max_{\theta \in \Theta} \int_{\mathbb{R}^N} g(x) p_\theta(x) dx \tag{2.13}$$

and

$$\underline{\mathbb{E}}[g(X)] = \min_{\theta \in \Theta} \int_{\mathbb{R}^N} g(x) p_\theta(x) dx. \tag{2.14}$$

If $g(x)$ is linear or easily approximated by a Taylor series then similar approximation techniques to those discussed for probability distributions earlier in the chapter can be applied.

In the general case, to accurately approximate the upper expectation in Eqn. 2.13 we can use the Monte Carlo estimator from Eqn. 2.7 inside an optimisation routine:

$$\overline{\mathbb{E}}[g(X)] = \max_{\theta \in \Theta} \frac{1}{M} \sum_{i=1}^{M} g(x_{(i)}), \tag{2.15}$$

where $x_{(1)}, \ldots, x_{(M)}$ are drawn from $p_\theta(x)$. This is known as Double Loop Monte Carlo simulation, and can be applied only in the case of distributional probability boxes. This estimator can be shown to have a positive bias [173]. As is the case with computation with interval models, it is necessary to use a non-linear optimisation routine to evaluate the outer loop in the general case.

The Double Loop Monte Carlo method can be modified to propagate distribution-free probability boxes by drawing the samples required for Eqn. 2.15 from a monotonically increasing staircase function, which is parameterised by θ and satisfies the envelope condition for the probability box under consideration. However, this approach will be computationally inefficient in general since a high dimensionality θ would be required to approximate the probability box with sufficient accuracy, and therefore another approach is required.

For a distribution-free probability box note that one can sample intervals from the

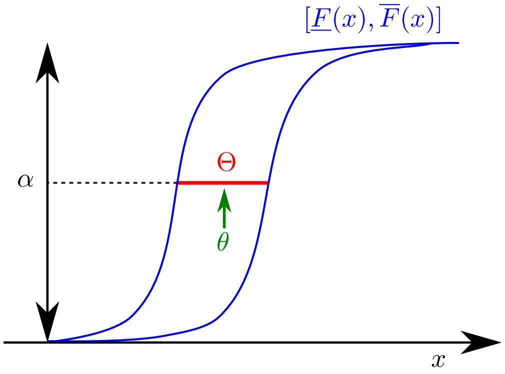

Figure 2.1: Obtaining samples from a probability box.

probability box by generating random numbers between 0 and 1 and applying inverse transform sampling to $\overline{F}$ and $\underline{F}$, i.e. generate sampled intervals $[\underline{x}, \overline{x}] = [\underline{F}^{-1}(\alpha), \overline{F}^{-1}(\alpha)]$ by sampling $\alpha \sim \mathcal{U}(0,1)$, where $\underline{F}^{-1}$ and $\overline{F}^{-1}$ are the inverse CDFs of the probability box envelopes and $\mathcal{U}$ is the uniform distribution. The process of generating a single sample is known as taking the α cut, and the sampled intervals constitute focal elements [50]. This process is shown in Figure 2.1. One can then propagate these intervals through $g(x)$ using the techniques described for intervals in the previous section, e.g. using the natural interval extension of $g(x)$, or using black-box optimisation. Therefore one can calculate the upper expectation as

$$\overline{\mathbb{E}}[g(X)] = \frac{1}{M} \sum_{i=1}^{M} \max_{x \in [\underline{x}_{(i)}, \overline{x}_{(i)}]} g(x), \tag{2.16}$$

where $[\underline{x}_{(i)}, \overline{x}_{(i)}]$ are the intervals generated from inverse transform sampling. In this thesis we dedicate most of our attention to the case of Double Loop Monte Carlo simulation for distributional probability boxes.

Dempster-Shafer structures

A Dempster-Shafer Structure is another form of imprecise probability model which has been widely applied in engineering [147]. In this thesis we do not use Dempster-Shafer structures, however we briefly outline the model and the relationship to probability boxes here for context. A Dempster-Shafer (DS) structure represents the assignment of probability mass to intervals rather than point values, as is the case with probability density functions.

Consider the case where probability mass has been assigned to intervals on the real line,

$\{([\underline{x}_{(1)}, \overline{x}_{(1)}], p_{(1)}), \ldots, ([\underline{x}_{(n)}, \overline{x}_{(n)}], p_{(n)})\}$, where $\sum_{i=1}^{n} p_{(i)} = 1$. We can define two measures called belief and plausibility which bound the probability mass contained in a particular interval. The belief measure is defined by

$$\mathrm{bel}(A) = \sum_{\substack{i \\ [\underline{x}_{(i)}, \overline{x}_{(i)}] \subseteq A}} p_{(i)}, \tag{2.17}$$

and the plausibility measure is defined by

$$\mathrm{pls}(A) = \sum_{\substack{i \\ [\underline{x}_{(i)}, \overline{x}_{(i)}] \cap A \neq \emptyset}} p_{(i)}, \tag{2.18}$$

where $\mathrm{bel}(A) \leq P(A) \leq \mathrm{pls}(A)$ and $\mathrm{pls}(A) = 1 - \mathrm{bel}(\overline{A})$, and $\subseteq$ represents a subset. For a particular DS structure, Ferson et al. [60] define an associated probability box using

$$\overline{F}(x) = \sum_{\underline{x}_{(n)} \leq x} p_{(i)} \tag{2.19}$$

and

$$\underline{F}(x) = \sum_{\overline{x}_{(n)} < x} p_{(i)}. \tag{2.20}$$

2.2 Creating models in practice

2.2.1 Choosing a model

As discussed previously, it is essential to exercise engineering judgement when choosing an uncertainty model, both in terms of speed of computations which can be performed with the model and the appropriateness of the model's representation of uncertainty. Engineers should also note that it is essential to exercise their judgement even after the theoretical uncertainty framework is chosen, since the set of hypotheses included in the uncertainty model has a strong affect on the conclusions drawn from analysis.

For example, consider the following problem proposed by Zadeh [185]: two doctors examine a patient, but differ in their diagnoses. Doctor A believes the patient has a 99% chance of meningitis and 1% chance of concussion. Doctor B believes the patient has a 99% chance of tumor and 1% chance of concussion. Since the doctors' diagnoses strongly conflict with each other, a naïve application of Bayesian probability concludes that the patient most likely has concussion. Zadeh proposes that this problem can be solved with fuzzy logic. However, Maskell [114] shows that Bayesian probabilities can, in fact, be used to solve the problem, by allowing the model to consider that the doctors may have made a mistake in their estimations of probabilities.

Another interesting example is demonstrated by Balch [10] and Balch et al. [11]; it is shown that using probability distributions to represent epistemic uncertainty in satellite conjunction analysis does not provide a useful description of the likelihood of collision between satellites, since the likelihood of collision appears to decrease when data with more incertitude is collected.

2.2.2 Training models from data

Here we provide a non-exhaustive review of methods to calibrate probabilistic and non-probabilistic generative uncertainty models, in order to set the context for the remainder of the thesis.

Creating parametric Bayesian probabilistic models

Consider the probability distribution $p_\theta(x) = p(x^{(i)}|\theta)$ with vector of parameters θ, which we wish to identify based on a set of n training samples, $\mathcal{X}_{\text{train}} = \{x^{(1)}, \ldots, x^{(n)}\}$, drawn from the random variable specified by $p_\theta(x)$. A distribution over the parameters θ, given the data $\mathcal{X}_{\text{train}}$ can be obtained by applying Bayes' law:

$$P(\theta|\mathcal{X}_{\text{train}}) = \frac{P(\mathcal{X}_{\text{train}}|\theta)p(\theta)}{P(\mathcal{X}_{\text{train}})}, \tag{2.21}$$

where $p(\theta)$ represents a prior distribution on θ, $P(\mathcal{X}_{\text{train}}) = \int P(\mathcal{X}_{\text{train}}|\theta)d\theta$ acts as a normalising constant, and the data likelihood can be written as $P(\mathcal{X}_{\text{train}}|\theta) = \prod_i p(x^{(i)}|\theta)$ by assuming independence of training samples. This approach, known as Bayesian Hierarchical Modelling [74], has desirable properties. For example, the epistemic uncertainty on θ will decrease as more data becomes available which will be observed as a 'concentration' of the posterior distribution for θ around one point.

Although simple analytical distributions are often used for the likelihood $P(\mathcal{X}_{\text{train}}|\theta)$ (e.g. a Gaussian distribution with mean θ_1 and scale θ_2). One can also extend the framework to consider more complex likelihood functions. For example, often the likelihood is $P(\mathcal{X}_{\text{train}}|\theta) = \int P(\mathcal{X}_{\text{train}}|y)\delta(f(\theta) - y)dy$, where $f(\theta)$ is an arbitrarily expensive and complex function, for which we may not know the gradient, and $p(x^{(i)}|\theta)$ is a simple probability density, for example a normal distribution, and $\delta(x)$ is the Dirac delta function. This setting is referred to as an 'inverse problem' [169].

The probability distributions over θ represent epistemic uncertainty in θ, whilst the data likelihood, $p(x^{(i)}|\theta)$, represents the natural stochasticity (aleatory uncertainty) of the data generating mechanism. Note that the prior distribution, $p(\theta)$, should be chosen to represent our prior knowledge of the parameter θ, and in the case of no knowledge, should be set to an appropriate uninformative distribution. The distribution used for the uninformative prior should be chosen based on physical considerations regarding the parameter of interest, but is often somewhat arbitrarily assumed to be uniform [88].

The prior distribution is not the only place where prior knowledge enters into the probabilistic model; the model specification, i.e. the data likelihood, represents another form of prior knowledge which must be carefully considered with this approach [88]. It is particularly important to decide which parameters in the likelihood function should be modelled as uncertain, e.g. if the likelihood is assumed to be a Gaussian density, will a value be assumed for the standard deviation of the distribution, or will this be an element of θ, and hence an uncertain parameter?

We can derive point estimates for θ from the Bayesian approach [67]. The maximum a posteriori estimator for θ is obtained by evaluating $\theta_{\text{MAP}} = \max_\theta P(\theta|\mathcal{X}_{\text{train}})$, where $P(\theta|\mathcal{X}_{\text{train}}) \propto P(\mathcal{X}_{\text{train}}|\theta)P(\theta)$. The maximum likelihood estimator for θ is obtained by evaluating $\theta_{\text{ML}} = \max_\theta P(\mathcal{X}_{\text{train}}|\theta)$. Note that the maximum likelihood estimator is equivalent to the maximum a posteriori estimator when a uniform prior distribution is used. These estimators can be evaluated using any optimisation method. Stochastic Gradient Descent, a widely used optimisation method, will be discussed in Chapter 3 since it is most often applied to regression models.

We do not necessarily have to disregard uncertainty in θ when using the maximum a posteriori approach, since the covariance of the distribution can be estimated by inverting the Hessian (matrix of second derivatives with respect to parameters θ) of $P(\theta|\mathcal{X}_{\text{train}})$. This is known as the Laplace approximation. This estimate is exact in some well known cases, e.g. the case of a Gaussian likelihood and prior, where the optimisation loss (the logarithm of the posterior) becomes the mean squared error [169].

In many cases the full posterior for θ can be calculated analytically, for example where a conjugate prior is used so that the posterior distribution has the same functional form as the prior distribution. If this is not the case, and one wishes to compute the full posterior distribution, then a Markov Chain Monte Carlo (MCMC) method is often used to obtain samples from the posterior distribution, or other approximate numerical techniques are used.

An MCMC algorithm constructs a Markov chain with the desired distribution as its equilibrium distribution, so that if the Markov chain is simulated for a sufficient time then the samples drawn are from the posterior distribution [74]. MCMC methods typically do not require the gradient of the posterior to be known, and are hence applicable to a wide class of problems. Unfortunately, MCMC simulation can be computationally infeasible when θ has high dimensionality, or when the training dataset is large. Recently, efficient sampling based algorithms have been proposed to combat this problem [80].

As an alternative to MCMC based methods, Variational Inference can be used to find the closest match between an approximating parametric 'proposal' distribution and the true posterior distribution. This method typically requires the gradient of the likelihood function to be known, but scales very well to high dimensionality problems [21].

Approximate Bayesian Computation is an efficient computational method which can be used to sample from an approximation of the posterior distribution in the case that the likelihood is too expensive to compute [48]. Sadeghi et al. [154] demonstrate a similar method, where the true likelihood probability density function is replaced by an interval with associated probability, and show that bounds on the likelihood function can still be obtained in this case.

Frequentist confidence intervals

In this thesis, traditional frequentist statistics are not used, except for in the validation of some Interval Predictor Models, but for the interested reader we briefly describe here how a frequentist confidence interval can be obtained for θ.

In frequentist statistics, one aims to identify a region of parameter space which would contain the true value of the parameter with a specified frequency if the experiment was repeated, i.e. we aim to find the confidence interval $\Theta = [\underline{\theta}, \overline{\theta}]$, where $P(\theta \in \Theta) = 1 - \alpha$, and α is an arbitrarily small probability.

Non-parametric prediction intervals

The maximum and minimum of a dataset (i.e. $[\min_i x_{(i)}, \max_i x_{(i)}]$ when x is one dimensional) can be used to produce a prediction interval with coverage probability $\frac{n-1}{n+1}$, i.e. the probability that $x_{(n+1)}$ will fall inside the prediction interval [183]. A tighter prediction interval, with a lower coverage probability of $\frac{n+1-2j}{n+1}$ can be obtained by using the j-th smallest and largest values in the dataset.

Creating parametric imprecise probability models

The application of Bayes' law in Eqn. 2.21 assumes that the data $\mathcal{X}_{\text{train}}$ consists of real, 'crisp', values. However, we can also apply Bayes' law to imprecise, interval data. For example, consider the set of training data $\mathcal{X}_{\text{train}} = \{[\underline{x}^{(1)}, \overline{x}^{(1)}], \ldots, [\underline{x}^{(n)}, \overline{x}^{(n)}]\}$. If an analytic equation is available for the posterior parameters then in many cases it is possible to obtain bounds on the posterior parameters given interval data. For example, if a Gaussian density is used for the likelihood and prior, then one may obtain bounds on the posterior normal distribution parameters analytically [120].

The standard Bayesian paradigm can also be made robust by considering a set of prior distributions. This is known as Robust Bayes [17]. Again, bounds on the posterior parameters are available analytically in many cases, e.g. the Imprecise Dirichlet model.

Probability boxes can also be obtained by creating so-called confidence structures, which are encoded as probability boxes. Confidence boxes encode confidence intervals at all confidence levels. The binomial confidence bounds, which bound the success probability of a binomial random variable, are a particularly useful example which can be found by inverting the CDF of a binomial random variable [62].

Creating non-parametric imprecise probability models

Several methods exist to obtain non-parametric CDFs from data. The CDF can be estimated from n training samples, $\mathcal{X}_{\text{train}} = \{x^{(1)}, \ldots, x^{(n)}\}$, using the empirical cumulative distribution function (eCDF), which is given by

$$S_n(x) = \frac{1}{n} \sum_{i}^{n} \mathbb{I}_{x \geq x^{(i)}}(x), \tag{2.22}$$

where $\mathbb{I}$ is the indicator function, which is equal to 1 if the subscript statement is satisfied, and is otherwise equal to zero [101]. The eCDF is effectively the random variable which is formed by assigning probability density equally at the point value of each sample, and hence when plotted the eCDF looks like a staircase function. The eCDF can be generalised to the case of imprecise sampled data, by considering a separate eCDF for the upper and lower bounds of the samples. These upper and lower bounds represent the envelope of a probability box, and hence an empirical probability box is obtained [60].

So-called concentration inequalities can be used to obtain bounds on the CDF of a random variable with a certain confidence. A probability box can be obtained for the random variable by choosing a cutoff confidence, such that the CDFs at that confidence will form the envelope of the probability box [60].

The Kolmogorov-Smirnov statistic can be used to measure the confidence that the true CDF of a random variable differs by more than a certain probability from the eCDF obtained from sampled data (i.e. the vertical distance between the CDFs is compared) [126]. The Kolmogorov-Smirnov statistic is given by

$$D = \sup_{x} |S_n(x) - F(x)|, \tag{2.23}$$

where $F(x)$ is the true CDF and the values for D can be obtained from Kolmogoroff [101]. Now a set of CDFs can be found with associated confidence. The Kolmogorov-Smirnov statistic can be applied to eCDF bounds obtained by considering imprecise data [60].

Chebyshev's inequality bounds the probability density of a random variable which can fall more than a certain number of standard deviations from the mean [149]. Therefore knowledge of the mean and standard deviation of a random variable imposes bounds on its CDF. Chebyshev's inequality is given by

$$P((X - \mu) \geq k\sigma) \leq \frac{1}{k^2}, \tag{2.24}$$

where $k > 1$ is a real number, μ is the mean of a random variable, and σ is the standard deviation of the random variable.

2.2.3 Creating uncertainty models without data

When insufficient data is available to create a satisfactory uncertainty model using the techniques described in the previous section, one may resort to creating a model based on the opinions of experts. This process is known as expert elicitation [167]. In this section we briefly outline how various uncertainty models can be obtained from expert opinion, in order to further justify and provide context for the uncertainty models used in this thesis. Expert elicitation is not the main focus of this thesis, and therefore this section may be skipped without consequence.

Probabilistic elicitation

In the Bayesian Hierarchical Modelling paradigm, discussed in the previous section, the posterior distribution concentrates as data is received and gradually the prior has less influence on the posterior. The prior distribution represents the state of knowledge about a parameter before data is available, and if limited data is available then more care should be taken to choose an appropriate prior, i.e. the opinions of experts should be considered and assessed quantitatively.

When eliciting multiple expert opinions one must attempt to aggregate the opinions of experts, regardless of the model chosen. Usually the opinions of experts are fused using quantitative rules [167], and feedback may be given to the experts in order to allow their opinions to be changed. Oakley and O'Hagan [127] propose the SHELF framework which gives specific rules for how the opinions of experts should be elicited, and proposes that the opinions should be aggregated by a rational unbiased observer during the elicitation process.

Non-probabilistic elicitation

Imprecise probability models also have a role to play in expert elicitation. For example, if the model exhibits severe dependency on a probabilistic prior which can only be elicited approximately, one may wish to conduct a sensitivity analysis to the prior by considering a probability box prior, as discussed in Section 2.2.2.

Other non-probabilistic models may be considered for a parameter, for example interval bounds on a parameter may be available from physical considerations. Alternatively, experts may prefer to specify their estimates as intervals, or may feel more comfortable specifying bounding CDFs (i.e. probability boxes). Ferson et al. [60] discuss several methods for aggregating probability boxes which can be chosen based on the desired properties of the analysis.

2.2.4 Validating a trained model

Once an uncertainty model has been obtained it is essential that the model is validated, to ensure that it suitably represents the analysts uncertainty. Even if one uses a Bayesian framework and trusts the priors and probability calculus, it is still possible that the model has been misspecified. For this reason, one should qualitatively inspect the results of the analysis, and quantitatively check that a probabilistic model is correctly calibrated using numerical techniques.

Validating probabilistic models

Usually one partitions the data available for creating the model into the data set used for training the model, $\mathcal{X}_{\text{train}}$, and the data set used for testing, $\mathcal{X}_{\text{test}}$ (containing N_{test} data points).

If a probabilistic model is correctly calibrated then we expect the stated probabilities to represent the real frequencies with which events occur, for example if a set of events are predicted to occur with 0.9 probability then we expect that they occur 90% of the time in reality. This can be verified by plotting the test data relative to the trained distribution. One method of achieving this is ‘binning’ the data into a histogram, and visually comparing the histogram to the plotted distribution for the trained model [74].

One may use the test set, $\mathcal{X}_{\text{test}}$, to compute various statistics of the model. For example, classical statistical tests can be used, such as the χ^2 summary statistic for the sum of squared errors which represents goodness of fit [74]. The test set can be used to compute the negative logarithmic predictive density for the model, i.e. $-\log P(\mathcal{X}_{\text{test}}|\text{Model}) = -\log \mathbb{E}_{P(\theta|\mathcal{X}_{\text{train}})} P(\mathcal{X}_{\text{test}}|\theta)$, which can be used as a figure of merit for comparing models.

If one wishes to compare two probabilistic models Model_1 and Model_2 then one may compare the evidence for the models by computing the Bayes Factor:

$$\frac{P(\mathcal{X}_{\text{train}}|\text{Model}_1)}{P(\mathcal{X}_{\text{train}}|\text{Model}_2)} = \frac{P(\text{Model}_1|\mathcal{X}_{\text{train}})P(\text{Model}_2)}{P(\text{Model}_2|\mathcal{X}_{\text{train}})P(\text{Model}_1)}, \tag{2.25}$$

where the model evidence (or marginal likelihood) $P(\mathcal{X}_{\text{train}}|\text{Model})$ is computed by evaluating the expectation of the data likelihood for the model over the posterior obtained in training ($P(\mathcal{X}_{\text{train}}|\text{Model}) = \mathbb{E}_{P(\theta|\mathcal{X}_{\text{train}})} P(\mathcal{X}_{\text{train}}|\theta) = \int P(\mathcal{X}_{\text{train}}|\theta)P(\theta|\mathcal{X}_{\text{train}})d\theta$). If the Bayes factor is greater than one then Model_1 is preferred, otherwise one should choose Model_2 [106]. When $P(\text{Model}_2) = P(\text{Model}_1)$ the prior belief in each model is equal and the Bayes factor becomes equal to the likelihood ratio. The likelihood ratio can also be computed for the test data set. One can also generate new data by sampling from the distribution $\mathbb{E}_{P(\theta|\mathcal{X}_{\text{train}})}p(x|\theta)$ and comparing this to the training and test data. This is known as a posterior predictive check [74].

Validating non-probabilistic models

If a convex-set or interval based model is compared to crisp data then one can check that all elements of the test set $\mathcal{X}_{\text{test}}$ fall within the model. Ferson et al. [61] proposes that interval models are validated against interval data using the metric for comparison of two sets A and B:

$$\Delta(A, B) = \inf_{a \in A, b \in B} |a - b|, \tag{2.26}$$

where A would represent the trained convex model, and B would represent an element of $\mathcal{X}_{\text{test}}$. The mean of $\Delta(A, B)$ over every element $B \in \mathcal{X}_{\text{test}}$ could be used to validate against the entire test set, i.e. $\frac{1}{N_{\text{test}}} \sum_{B \in \mathcal{X}_{\text{test}}} \Delta(A, B)$.

Following this, Ferson et al. [61] proposes a generalisation of the Wasserstein distance to measure the distance between an eCDF and a probability box, as a probability box validation metric. The proposed metric, termed mean absolute difference of deviates, is given by

$$\mathbb{E}_x \Delta([\underline{F}(x), \overline{F}(x)], [\underline{S_n}(x), \overline{S_n}(x)]), \tag{2.27}$$

where $[\underline{F}(x), \overline{F}(x)]$ are the bounds of the probability box to be validated, and $[\underline{S_n}(x), \overline{S_n}(x)]$ are the bounds of the empirical probability box created from the training data, which becomes a single CDF in the case of crisp data ($[S_n(x), S_n(x)]$). The metric reduces to zero when there is overlap of the probability boxes at every x. We compare models by computing the metric for each model, and then choosing the model with the lowest value for the metric.

2.3 Chapter summary

This chapter presents a review of uncertainty models which can be used to describe unknown parameters in a computational model, in addition to describing how the models can be created in practice, from data or otherwise, and how computation can be performed with these models. In particular, we discussed probabilistic models, which are used for conventional uncertainty quantification, and non-probabilistic models, which are used in cases where only limited or imprecise data may be available, and prior knowledge may be difficult to obtain in the form of a probabilistic prior. In this chapter, all considered uncertainty models did not depend upon the behaviour of other variables — their uncertainty was constant or homoscedastic. In many cases, it may be desirable to model how the uncertainty in a variable changes with respect to another variable. This is typical, for example, when we wish to consider how the output of a computer code changes with respect to its inputs. Therefore, the following chapter describes so called regression models, where modelling this dependency is possible.

Chapter 3

Machine Learning of Regression Models

Regression models differ from generative uncertainty models in that they model the effect of one variable on another variable. In the language of probability theory this involves modelling a conditional probability distribution, rather than a joint probability distribution. In some fields regression models are known as discriminative models, but this is usually when the dependent variable in the probability distribution is discrete, and the problem to be solved involves classification [125]. This thesis is concentrated only on the case of continuous variables. In this chapter, we review different classes of regression model and describe how they can be trained and validated from data. We describe in which circumstances each type of model should be used.

3.1 Parametric regression models

Typically when learning a regression model, one wishes to obtain a estimate of the relationship between the two variables, in addition to a measure of uncertainty in this relationship. Typically this is achieved by specifying a function relating the two variables, and then using Machine Learning techniques to calibrate parameters of that function based on sampled data. The function can be specified based on expert knowledge, or alternatively a very general function is chosen.

The most simple regression models usually consist of the inner product of a parameter vector and a vector of ‘features’, such that the model output’s dependency on the parameters to be calibrated is linear. The features, usually known as the basis, are a function of the input variables, which may be non-linear. Models of this form are amenable to computation, and in general a wide variety of functions can be expressed in this way. In this formulation the output of the model is given by

$$f(x, \boldsymbol{p}) = \sum_i p_i \phi_i(x), \tag{3.1}$$

where x and $f(x, \boldsymbol{p})$ represent the input and output to the regression model, respectively, p_i are the parameters of the model to be calibrated which are components of the vector $\boldsymbol{p}$, and $\phi_i(x)$ is the basis [67]. Let x be a vector in $\mathbb{R}^N$, with components x_i.

The basis should be chosen based on prior knowledge and engineering judgement. Basis functions are either global or local. A local basis consists of radial functions, which only depend upon the distance from a certain point, i.e. $\phi(x) = \phi(|x|)$. The Gaussian basis is a common radial basis function, where $\phi_i(x) = \exp\left(-\epsilon_i(x - c_i)^2\right)$. ϵ_i and c_i are parameters which can be set based on knowledge of the physical process being modelled or learnt from data at the same time as p_i, though this is more difficult because $f(x)$ has non-linear dependency on these parameters. The Gaussian basis function tends to zero far away from c_i. On the other hand, a global basis consists of functions which in general are non-zero at all points in the input space. For example, the global polynomial basis of the form $\phi_i(x) = \prod_i x_i^{l_i}$, with indices l_i chosen based on engineering judgement, is nonzero everywhere except for $x = 0$.

Polynomial Chaos Expansions offer a principled way to choose basis functions. Polynomial Chaos Expansions are a class of regression model with a basis consisting of polynomials which are orthogonal to each other, i.e. their inner product is zero with respect to a probability distribution over their inputs ($\int_{\mathbb{R}^N} \phi_i(x)\phi_j(x)p(x)dx = 0 \forall i \neq j$) [157].

In order to learn more complex functions, a more complex function representation is needed, with the ability to model arbitrary non-linearities in the model parameters to be learned. Neural networks are a widely used regression model which fulfil this purpose [67]. Neural networks consist of neurons which apply an inner product between the input and a parameter vector, and then apply an arbitrary non-linearity, before feeding into other neurons, until finally the result is outputted. These computational neurons are organised into layers, which is equivalent to multiplying the input by a parameter matrix (known as a weight matrix), rather than a vector. The way in which layers are connected can lead to desirable properties, for example spatial invariance of particular layers over the

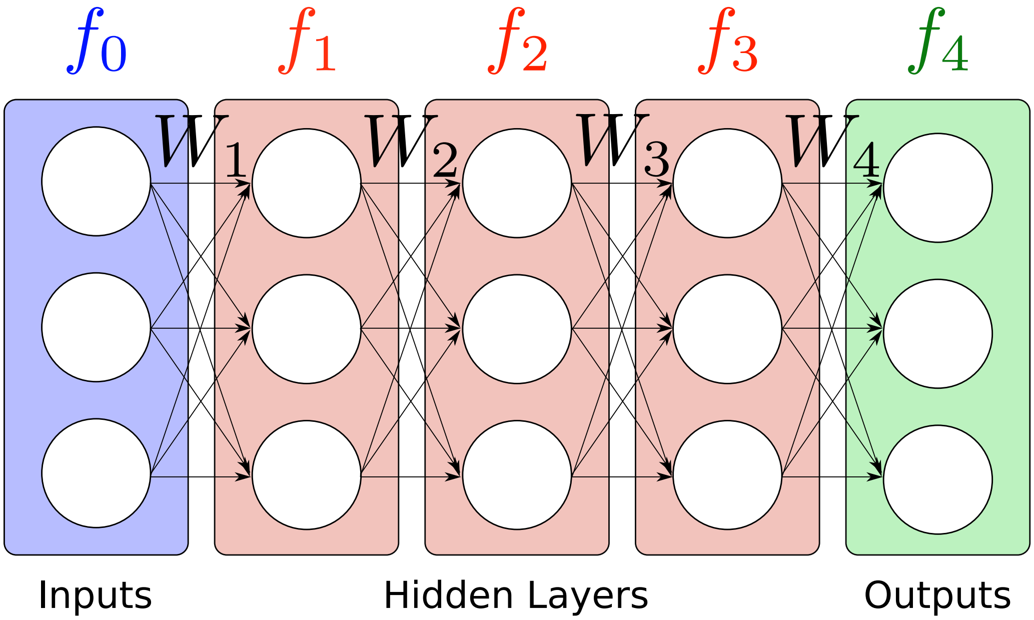

Figure 3.1: A diagram of a feed-forward neural network with three hidden layers, each with a width of three neurons. The activation function, which is applied to the weighted sum of the inputs to each neuron, is not shown.

input when the input is an image, i.e. a matrix. This is equivalent to repeating parameters in the weight matrix. The most simple way to connect the layers is to allow each layer to be completely connected to the subsequent layer, which is known as the feedforward architecture. Layer i of a feed-forward neural network is given by

$$f_i(x, W) = \text{act}\left(W_i f_{i-1}(x)\right), \tag{3.2}$$

where $f_i(x, W)$ is a vector (which is the input vector x when $i = 0$, i.e. $f_0(x) = 0$), act is a non-linear activation function, and W_i is the ith weight matrix. The activation function is typically the hyperbolic tangent function, the soft-max function or the rectified linear function. A diagram of a feed-forward neural network is shown in Figure 3.1. Sundararajan [168] demonstrates how neural networks can be trained to replicate the opinions of expert engineers on the probability of failure of particular pipe welds in a power plant.

3.1.1 Bayesian parameter learning

Defining a data likelihood

It is common to define a probability distribution based on the output of a regression model, and then use the probability calculus introduced in Chapter 2 to learn distributions over the parameters of the regression model. For simple models it is usually assumed that the output $f(x)$ of the model is some meaningful parameter of the distribution, e.g. the mean of a normal distribution:

$$p(y|x, \boldsymbol{p}) = \mathcal{N}(f(x, \boldsymbol{p}), \, \sigma^2), \tag{3.3}$$

where σ is the scale parameter of the normal distribution, which should be learned form data. This results in a model where the level of uncertainty in $p(y|x)$ does not depend on the input to the model. This is known as a homoscedastic model of uncertainty.

Sometimes it is desirable to explicitly allow the uncertainty in the predictions to depend on x. This is known as heteroscedastic uncertainty [68]. In this case it can be useful to define a model where other parameters of the distribution depend on x, i.e. we define

$$p(y|x, \boldsymbol{p}) = \mathcal{N}(f_1(x, \boldsymbol{p}_1), \, f_2(x, \boldsymbol{p}_2)^2), \tag{3.4}$$

where $f_1(x, \boldsymbol{p}_1)$ and $f_2(x, \boldsymbol{p}_2)$ are different functions with different parameter sets, $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$.

Any valid probability distribution can be used in a similar way, for example the Dirac delta function can be used to define

$$p(y|x, W, u) = \delta(y - f_i(x, u, W)), \tag{3.5}$$

where $f_i(x)$ is the output of a neural network, where in this case the input layer is a function of the true input and a random vector of noise, i.e. $f_0(x, u, W) = \text{concatenate}(x, u)$ where $u \sim \mathcal{U}(0, 1)$. This is a very popular formulation in Machine Learning for Computer Vision [77] [55], because $p(y|x, W, u)$ can now be used to learn a very general probability density in a computationally tractable way, since $p(y|x, W) = \int \delta(y - f_i(x, u, W))\mathcal{U}(0, 1)du$ can be evaluated easily using a Monte Carlo estimator during inference of the posterior distribution.

Performing the Bayesian computation

Using a set of n training samples, $\mathcal{X}_{\text{train}} = \{\{x^{(1)}, y^{(1)}\}, ..., \{x^{(n)}, y^{(n)}\}\}$, one can learn a distribution over $\boldsymbol{p}$ in the same way as in Chapter 2 by using

$$P(\boldsymbol{p}|\mathcal{X}_{\text{train}}) = \frac{P(\mathcal{X}_{\text{train}}|\boldsymbol{p})p(\boldsymbol{p})}{P(\mathcal{X}_{\text{train}})} = \frac{\prod_i P(x^{(i)}, y^{(i)}|\boldsymbol{p})p(\boldsymbol{p})}{P(\mathcal{X}_{\text{train}})} = \frac{\prod_i p(y^{(i)}|x^{(i)}, \boldsymbol{p})p(x^{(i)})p(\boldsymbol{p})}{P(\mathcal{X}_{\text{train}})}, \tag{3.6}$$

where the data likelihood can be written as $P(\mathcal{X}_{\text{train}}|\boldsymbol{p}) = \prod_i p(y^{(i)}, x^{(i)}|\boldsymbol{p})$ by assuming independence of training samples, $p(\boldsymbol{p})$ represents a prior distribution on $\boldsymbol{p}$, $p(x^{(i)}, \boldsymbol{p}) = p(x^{(i)})p(\boldsymbol{p})$ by assuming independence of $\boldsymbol{p}$ and the sampled inputs, and $P(\mathcal{X}_{\text{train}}) = \int \prod_i p(y^{(i)}|x^{(i)}, \boldsymbol{p})p(x^{(i)}, \boldsymbol{p})d\boldsymbol{p}$ acts as a normalising constant. As in Chapter 2, the posterior distribution on $\boldsymbol{p}$ will tend to concentrate around one point as more data is received.

The maximum likelihood and maximum a posteriori estimators described in Chapter 2 are equally applicable here. These estimators are evaluated by minimising so-called loss functions (objective functions). The maximum a posteriori estimator for $\boldsymbol{p}$ is obtained by evaluating $\boldsymbol{p}_{\text{MAP}} = \max_{\boldsymbol{p}} P(\boldsymbol{p}|\mathcal{X}_{\text{train}})$, where $P(\boldsymbol{p}|\mathcal{X}_{\text{train}}) \propto P(\mathcal{X}_{\text{train}}|\boldsymbol{p})P(\boldsymbol{p}) = \mathcal{L}_{\text{MAP}}(\boldsymbol{p})$. The maximum likelihood estimator for $\boldsymbol{p}$ is obtained by evaluating $\boldsymbol{p}_{\text{ML}} = \max_{\boldsymbol{p}} \mathcal{L}_{\text{ML}}(\boldsymbol{p}) = \max_{\boldsymbol{p}} P(\mathcal{X}_{\text{train}}|\boldsymbol{p})$. The maximum likelihood estimator is equivalent to the maximum a posteriori estimator when a uniform prior distribution is used. Using a normal distribution for the data likelihood leads to the well known mean squared error or ℓ_2 norm loss function when the maximum likelihood estimator is used. Using a polynomial basis with the mean squared error loss function leads to ordinary least squares regression. If a normal distribution prior is used then this leads to an ℓ_2 weight regularisation (squared penalty) in the maximum a posteriori loss function. In a similar way, most sensible loss functions which aim to estimate point values for parameters have a Bayesian interpretation.

Computational methods

In practice, the most common way to create regression models is to evaluate the estimators $\boldsymbol{p}_{\text{ML}}$ or $\boldsymbol{p}_{\text{MAP}}$ with Stochastic Gradient Ascent, which maximises the logarithm of the relevant probability distribution (or equivalently by using Stochastic Gradient Descent to minimise the negative of the log posterior). This is computationally tractable even for high dimensional $\boldsymbol{p}$, since usually the gradient of $\log P(\boldsymbol{p}|\mathcal{X}_{\text{train}})$ is known analytically. Gradient Descent methods are a class of optimisation methods which adjust the value for a parameter at each step of the algorithm by subtracting a small learning rate constant, η, multiplied by the gradient of the loss, $\mathcal{L}$, with respect to the trainable parameter, i.e.

$$p_i \leftarrow p_i + \eta \frac{\partial \mathcal{L}}{\partial p_i}. \tag{3.7}$$

This is repeated for a set number of iterations until the algorithm has converged. Stochastic Gradient Descent approximates the product of likelihoods in the loss function by evaluating the likelihood for one different sampled data point at each iteration. This is effective since the expectation of the loss used in Stochastic Gradient Descent will still be equal to the true value of the loss function. In this case, the learning rate constant must be reduced to ensure convergence, which means many iterations of the algorithm are required to ensure a good estimate for the parameters is obtained. Mini-batches, where the likelihood is evaluated for a small set of data points at each iteration, can be used to achieve good convergence at higher learning rates, whilst decreasing the required computational time, since a GPU can be used [151]. Various improvements to Stochastic Gradient Descent aim to ensure that the optimiser reaches a true minimum of the loss function, a particularly common improvement being the ADAM optimiser [100].

Using the maximum likelihood and maximum a posteriori estimators can allow some estimate of the uncertainty in the model to be made, but this uncertainty is an underestimate of the true model uncertainty. For very simple regression models, MCMC can be used to obtain the full posterior distribution on $\boldsymbol{p}$, however this is usually intractable for models with large parameter sets. As an alternative, variational inference can be used to minimise the difference between the $P(\boldsymbol{p}|\mathcal{X}_{\text{train}})$ and an approximating posterior distribution, as described in Chapter 2. Note that $P(\boldsymbol{p}|\mathcal{X}_{\text{train}})$ must be differentiable in $\boldsymbol{p}$ for this to be possible. Using Bayes' law to infer posterior distributions over the weights of a neural network is referred to as training a Bayesian neural network, and this is almost always achieved by using variational inference [21].

The technique of dropout sampling has been shown to improve the performance of Stochastic Gradient Descent solvers, by improving the performance on validation tests [162]. Dropout sampling involves randomly setting a fraction, p_{dropout} of the weights to zero during each training iteration. For particular choices of activation function, when an l_2 penalty on the weights is used in the loss function, it can be shown that dropout sampling is equivalent to variational inference on a Bayesian neural network, where a Bernoulli distribution is used as the approximating posterior distribution [70]. In order for the approximating posterior distribution to be an accurate representation of the true posterior distribution, it is necessary to adjust the dropout probability, p_{dropout}. Rather than repeatedly performing training with different dropout probabilities, it is more efficient to make the dropout probability a parameter which can be optimised during training, by making the loss differentiable in terms of the dropout probability. This can be achieved with concrete dropout, where a continuous approximation of the Bernoulli distribution is used [71].

3.1.2 Validation

As was the case with generative uncertainty models in Chapter 2, it is necessary to validate Regression Models. For probabilistic regression models this involves many of the same techniques which are applied when validating generative models. However, validating conditional probability densities presents additional challenges; although the model's predicted probabilities may be correctly calibrated on average, the model may be overly certain in some areas of the input domain and too uncertain in other areas. For example, using a regression model with homoscedastic uncertainty on a dataset where the uncertainty is heteroscedastic may predict the correct mean squared error on average [69], but the model evidence will be lower than for a more appropriate model.

To briefly recap the content from Section 2.2.4, before training one should split the data into training and test data sets, and then begin the validation by applying sanity checks. For example, a posterior predictive check could be used, where data is sampled from the trained model and compared to the training data. Alternatively, one could produce a plot of the normalised residuals, where the difference between the model output and the training and test data divided by predicted standard deviation ($\frac{y-y^{(i)}}{\sqrt{\text{Var}_{p(y|x^{(i)})}(y)}}$) is plotted against the model output. Then more formal methods can be used, for example the Bayes factor can be computed as in Eqn. 2.25, to compare several models [67]. This is similar to comparing the negative logarithmic predictive density of different models on the test sets, which is equal to the Mahalanobis distance for Gaussian predicted probability densities. It is essential to compare the value of the loss (the negative logarithmic predictive density) between the test and training data sets. If the value of the loss is much higher on the test data set it is likely that the model is over-fitting the data, and will not generalise well to new data. One may also wish to compute the expected variance of the Model's predictions ($\int \text{Var}_{p(y|x)}(y)p(x)dx$), as it is likely that this can be compared to the expected uncertainty of a subject matter expect in order to appraise the performance of the model.

If the uncertainty is too high it is likely that the model is under-fitting the data so the model complexity should be increased.

3.1.3 The bias-variance tradeoff

Bayesian techniques rely upon priors to control the complexity of a regression model. Well chosen priors prevent learning too much information from a sample of data, and hence prevent overfitting by restricting the effective learning capacity of the model. The issue of underfitting is usually addressed by giving the model as much complexity as is possible and necessary to reduce the bias of the model, in order to ensure that the model is able to represent the desired function in principle. Then, overfitting is prevented by using a well chosen prior to reduce the variance of the fitted model. Non-Bayesian machine learning techniques often arbitrarily introduce mechanisms to constrain model complexity such as weight penalties; these techniques are unnecessary in the Bayesian paradigm due to the effect of prior distributions, which are in some cases equivalent to weight penalties. Bootstrapping (averaging over maximum likelihood models trained on re-sampled selections of the training data) is often used to reduce the variance of the trained model. Friedman et al. [67] describes how bootstrapping can also be seen as a method to compute maximum likelihood estimates of difficult to compute quantities like the standard error in an estimator, and an alternative implementation of maximum a posteriori estimation for the case of an uninformative prior. For certain likelihood functions and priors the bootstrap distribution can be seen as an approximate Bayesian posterior distribution.

The Vapnik-Chervonenkis (VC) dimension, which represents the complexity of a classification model, can be used to derive a bound between the test error and training error of a classification model (i.e. where y is a binary outcome) [177]. Similar bounds exist for regression models. This means that the test error can be established without partitioning the data. We do not use the VC dimension in this thesis because it is difficult to calculate in practice, but we return to the idea of calculating a bound on the test error of a model without partitioning the data in Section 3.3.2.

3.2 Non-parametric models

Bayesian non-parametric models are models for regression which learn a prior distribution over functions at training time. This is achieved by learning so-called ‘Kernel hyper-parameters' from the training data set, which specify a Gaussian Process Prior. Inference is performed at test time by applying Bayes' law to this prior with the available data [54]. The process of performing computation inference at test time is referred to as lazy learning (as opposed to eager learning presented in the previous section, where the inference happens during training, and only a single pass through the network is required to make predictions). Gaussian Process models can be tuned to have desirable properties for many applications, for example one can assume that the training data is noise-free and hence the relevant function can be learned from very few samples. Gaussian Processes are particularly useful for global optimisation of non-linear functions, because the predictive uncertainty can be used to decide where the next sample should be chosen [65]. Gaussian processes are not used in this thesis (except for as a comparison in some of the numerical examples), but we will briefly describe how they relate to the Bayesian parametric models discussed in this Chapter.

It can be shown that a single layer Bayesian neural network, with an infinite number of neurons in the layer, is equivalent to a Gaussian Process, and can therefore be used as a more convenient alternative, since the inference is performed at training time [123]. Variational approximations can be made for Gaussian Processes to make the inference computationally tractable for large data sets and deep architectures [82]. Neural Processes learn a distribution over functions in a similar way to Gaussian processes, but with significantly reduced computational expense since only a forward pass through the neural network is required at test time [73]. Neural Processes are useful for meta-learning (learning to learn). For example, they were used to learn how to predict 2D views of 3D spaces, given limited training data [58].

3.3 Learning bounds on a model

Instead of learning a probability distribution to describe the effect of one variable on another, one may instead attempt to learn a function which maps the input variables to an interval representing the possible range of the output. Such models are known as interval predictor models. Sometimes the predicted intervals have an associated confidence level (or a bound on the confidence level), and as such they can be considered as bounds on the quantiles of a random variable [49]. Typically the obtained intervals represent an outer approximation, i.e. the intervals are overly wide and hence conservative in an engineering sense. An interval predictor model can be seen as prescribing the support of a Random Predictor Model, which is defined as a function which maps input variables to an output random variable. A Gaussian Process Model is a specific case of a Random Predictor Model.

In this section, we describe how interval predictor models can be trained in practice. We then describe how the theory of scenario optimisation can be used to provide guaranteed bounds on the reliability of the trained interval predictor models, for the purpose of validation.

3.3.1 Training interval predictor models

Let us consider a black box model (sometimes referred to as the Data Generating Mechanism or DGM) which acts on a vector of input variables $x \in \mathbb{R}^{n_x}$ to produce an output $y \in \mathbb{R}$. We wish to obtain the two functions $\overline{y}(x)$ and $\underline{y}(x)$ which enclose a fraction, ϵ, of samples from the DGM, i.e. samples of $y(x)$ where x is sampled from some unknown probability density. The functions $\overline{y}(x)$ and $\underline{y}(x)$ are bounds on a prediction interval, and as such we wish them to be as tight as possible. This can be written as a so-called chance constrained optimisation program:

$$\arg\min_{\boldsymbol{p}} \{\mathbb{E}_x(\overline{y}_{\boldsymbol{p}}(x) - \underline{y}_{\boldsymbol{p}}(x)) : P\{\overline{y}_{\boldsymbol{p}}(x) > y(x) > \underline{y}_{\boldsymbol{p}}(x)\} \leq \epsilon\}, \tag{3.8}$$

where $\boldsymbol{p}$ is a vector of function parameters to be identified, and ϵ is a parameter which constrains how often the constraints may be violated. Chance constrained optimisation programs can be solved by using a so-called scenario program, where the chance constraint is replaced with multiple sampled constraints based on data, i.e.

$$\arg\min_{\boldsymbol{p}} \{\mathbb{E}_x(\overline{y}_{\boldsymbol{p}}(x) - \underline{y}_{\boldsymbol{p}}(x)) : \overline{y}_{\boldsymbol{p}}(x^{(i)}) > y^{(i)} > \underline{y}_{\boldsymbol{p}}(x^{(i)}), i = 1, ..., N\}, \tag{3.9}$$

where $\mathcal{X}_{\text{train}} = \{\{x^{(1)}, y^{(1)}\}, ..., \{x^{(n)}, y^{(n)}\}\}$ are sampled from the DGM. Most of the literature on scenario optimisation Theory aims to obtain bounds on ϵ. Finding bounds on ϵ using scenario optimisation is easier in practice than other similar methods in statistical learning theory, since no knowledge of the Vapnik-Chervonenkis dimension (a measure of the capacity of the model, which is difficult to determine exactly) is required.

A key advantage over other machine learning techniques is that interval training data (i.e. where the training data inputs are given in the form $x^{(i)} \in [\underline{x}^{(i)}, \overline{x}^{(i)}]$ due to epistemic uncertainty or some other reason) fits into the scenario optimisation framework coherently

[108]. This can be seen as equivalent to defending against the attack model of adversarial examples considered by Madry et al. [112], where the network is trained to produce the same outputs for small perturbations of the input data. The framework also permits robustness against uncertainty in training outputs, i.e. $y^{(i)} \in [\underline{y}^{(i)}, \overline{y}^{(i)}]$, where $y^{(i)}$ is a single training example output.

Convex interval predictor models

If the objective and constraints for the scenario program are convex then the program can be easily solved, and bounds can be put on ϵ. We will approximate the DGM with an interval predictor model (IPM) which returns an interval for each vector $x \in X$, the set of inputs, given by

$$I_y(x, P) = \big\{y = G(x, \boldsymbol{p}), \boldsymbol{p} \in P\big\}, \tag{3.10}$$

where G is an arbitrary function and $\boldsymbol{p}$ is a parameter vector. By making an approximation for G and considering a linear parameter dependency Eqn. (3.10) becomes

$$I_y(x, P) = \big\{y = \boldsymbol{p}^T \phi(x), \boldsymbol{p} \in P\big\}, \tag{3.11}$$

where $\phi(x)$ is a basis (polynomial and radial bases are commonly used), and $\boldsymbol{p}$ is a member of a convex parameter set. The convex parameter set is usually assumed to be either ellipsoidal or hyper-rectangular [34]. Crespo et al. [46] demonstrates that hyper-rectangular parameters sets result in an IPM with bounds with a convenient analytical form. The hyper-rectangular parameter uncertainty set can be defined as

$$P = \big\{\boldsymbol{p} : \underline{\boldsymbol{p}} \leq \boldsymbol{p} \leq \overline{\boldsymbol{p}}\big\}, \tag{3.12}$$

where $\underline{\boldsymbol{p}}$ and $\overline{\boldsymbol{p}}$ are parameter vectors specifying the defining vertices of the hyper rectangular uncertainty set. The IPM with linear parameter dependency on the hyper-rectangular uncertain set of parameters is defined by the interval

$$I_y(x, P) = [\underline{y}(x, \overline{\boldsymbol{p}}, \underline{\boldsymbol{p}}), \overline{y}(x, \overline{\boldsymbol{p}}, \underline{\boldsymbol{p}})], \tag{3.13}$$

where $\underline{y}$ and $\overline{y}$ are the lower and upper bounds of the IPM, respectively. Explicitly, the lower bound is given by

$$\underline{y}(x, \overline{\boldsymbol{p}}, \underline{\boldsymbol{p}}) = \overline{\boldsymbol{p}}^T \left(\frac{\phi(x) - |\phi(x)|}{2} \right) + \underline{\boldsymbol{p}}^T \left(\frac{\phi(x) + |\phi(x)|}{2} \right), \tag{3.14}$$

and the upper bound is given by

$$\overline{y}(x, \overline{\boldsymbol{p}}, \underline{\boldsymbol{p}}) = \overline{\boldsymbol{p}}^T \left(\frac{\phi(x) + |\phi(x)|}{2} \right) + \underline{\boldsymbol{p}}^T \left(\frac{\phi(x) - |\phi(x)|}{2} \right). \tag{3.15}$$

To identify the hyper-rectangular uncertainty set one trains the IPM by minimising the value of

$$\delta_y(x, \overline{\boldsymbol{p}}, \underline{\boldsymbol{p}}) = (\overline{\boldsymbol{p}} - \underline{\boldsymbol{p}})^T |\phi(x)|, \tag{3.16}$$

subject to the constraint that the training data points fall inside the bounds on the IPM, by solving the linear and convex optimisation problem

$$\left\{ \hat{\underline{\boldsymbol{p}}}, \hat{\overline{\boldsymbol{p}}} \right\} = \underset{\boldsymbol{u}, \boldsymbol{v}}{\arg\min} \left\{ \mathbb{E}_x[\delta_y(x, \boldsymbol{v}, \boldsymbol{u})] : \underline{y}(x^{(i)}, \boldsymbol{v}, \boldsymbol{u}) \leq y^{(i)} \leq \overline{y}(x^{(i)}, \boldsymbol{v}, \boldsymbol{u}), \boldsymbol{u} \leq \boldsymbol{v} \right\}. \tag{3.17}$$

The constraints ensure that all data points to be fitted lie within the bounds and that the upper bound is greater than the lower bound. This combination of objective function and constraints is linear and convex [46]. In this thesis all interval predictor models have polynomial bases, i.e. $\phi(x) = \left[1, x^{i_2}, x^{i_3}, ...\right]$ with $x = [x_a, x_b, ...]$ and $i_j = [i_{j,a}, i_{j,b}, ...]$ with $i_j \neq i_k$ for $j \neq k$.

For illustrative purposes an example degree 2 IPM is shown without training data points in Figure 3.2. The hyper rectangular uncertainty set corresponding to the IPM in Figure 3.2 is plotted in Figure 3.3. The discontinuity observed in the upper and lower bounds is a consequence of the chosen basis, and can be avoided by choosing a basis where $\phi(x) = |\phi(x)|$.

Non-convex interval predictor models

In some circumstances engineers may wish to represent more complex functions with IPMs, and hence the functions used to represent the bounds of the IPM may have more trainable parameters. The interior point method used to solve linear optimisation programs, such as those used for convex IPMs, has complexity $d^2 n_{\text{cons}}$, where d represents the number of

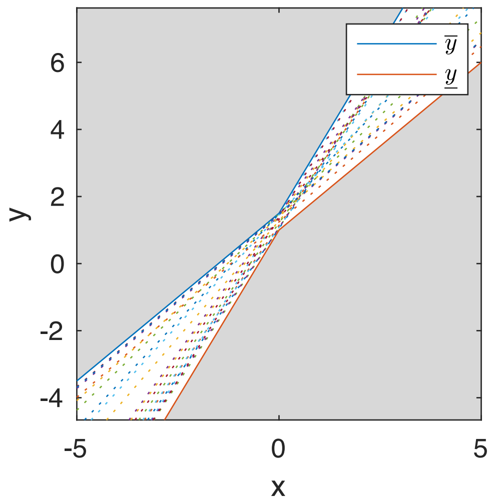

Figure 3.2: A degree 1 IPM in the 'data space' with $\overline{\boldsymbol{p}} = [1.5, 2]$ and $\underline{\boldsymbol{p}} = [1, 1]$. Sampled polynomials within the bounds of the IPM are shown as dashed lines. The grey region is outside the IPM and cannot be sampled from.

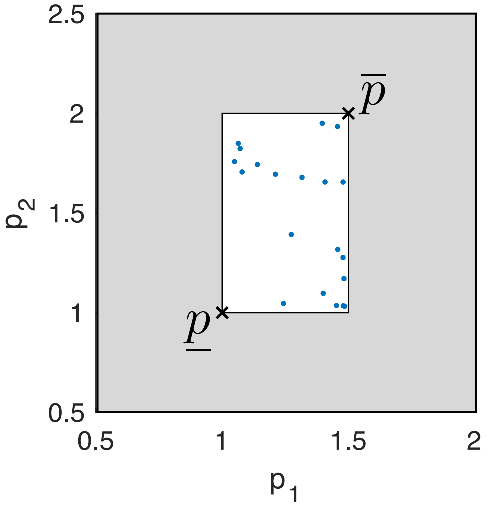

Figure 3.3: The IPM's hyper rectangular uncertainty set plotted in 'parameter space'. The uniformly sampled parameter vectors of the polynomials shown in Figure 3.2 are displayed as points in the uncertain set.

optimisation variables and n_{cons} represents the number of constraints in the optimisation (which in this case scales linearly with the number of training data points), and hence the method does not scale well to IPMs with large numbers of trainable parameters [22].

Neural networks enable uncertainty models to be created with vast numbers of parameters in a feasible computational time. Neural networks with interval outputs were proposed by Ishibuchi et al. [87], and further described by Huang et al. [83]. In these papers the learning takes place by identifying the weights W, which solve the following program:

$$\underset{\overline{W},\underline{W}}{\arg\min}\,[\mathbb{E}_x(\overline{y}(x)-\underline{y}(x)) : \overline{y}(x^{(i)}) > y^{(i)} > \underline{y}(x^{(i)}) \ \forall\ i], \tag{3.18}$$

where $\overline{y}(x)$ and $\underline{y}(x)$ are obtained from two independent neural networks, such that $\overline{y}(x)$ and $\underline{y}(x)$ are the output layers of networks, such as those defined by Eqn. 3.2. In practice this problem is solved by using a mean squared error loss function with a simple penalty function to model the constraints. In general, penalty methods require careful choice of hyper-parameters to guarantee convergence. These neural networks act in a similar way to interval predictor models, however the interval neural networks do not attempt to use the training data set to bound ϵ. Freitag et al. [66] define similar networks with fuzzy parameters to operate on fuzzy data. The fuzzy neural networks are trained by minimising a least square loss function (a set inclusion constraint is not used), which can also be applied to time series data sets. These approaches are very different from the approach of Patiño-Escarcina et al. [137], where a traditional neural network loss function is intervalised using interval arithmetic.

Campi et al. [35] extended the scenario approach to non-convex optimisation programs, and hence applied the approach to a single layer neural network, with a constant width interval prediction, which was trained using the interior-point algorithm in Matlab. In other words the following program is solved:

$$\underset{W,h}{\arg\min}\,[h : |y^{(i)} - \hat{y}(x^{(i)})| < h \ \forall\ i], \tag{3.19}$$

where h is a real number, and $\hat{y}$ represents the central line of the prediction obtained from the same network specified by Eqn. 3.2. The bounds on the prediction interval are therefore given by $\overline{y}(x) = \hat{y}(x) + h$ and $\underline{y}(x) = \hat{y}(x) - h$. The constant width interval neural network expresses homoscedastic uncertainty. The solution to the optimisation program in Eqn. 3.19 can also be obtained by finding the neural network weights which minimise the so-called maximum-error loss:

$$\mathcal{L}_{\text{max-error}} = \max_i |y^{(i)} - \hat{y}(x^{(i)})|, \tag{3.20}$$

where h is the minimum value of the loss. It is trivial to show this is true, since the set inclusion constraint in Eqn. 3.19 requires that h is larger than the absolute error for each data point in the training set [37].

3.3.2 Validating models with the scenario approach

We will first present an overview of the scenario optimisation theory for the validation of models in the convex case, before describing more general techniques which apply in the non-convex case.

Convex case

Intuition tells us that the solution of the scenario program will be most accurate when the dimensionality of the design variable is low and we take as many samples of the constraints as possible (in fact, an infinite number of sampled constraints would allow us to reliably estimate $P\{\overline{y}_{\boldsymbol{p}}(x) > y(x) > \underline{y}_{\boldsymbol{p}}(x)\}$, and hence solve the program exactly). However, in practice obtaining these samples is often an expensive process. Fortunately, the theory of scenario optimisation provides robust bounds on the robustness of the obtained solution. The bounds generally take the following form:

$$P^n(V(\hat{z}_n) > \epsilon) \leq \beta. \tag{3.21}$$

This equation states that the probability of observing a bad set of data (i.e. a bad set of constraints) in future, such that our solution violates a proportion greater than ϵ of the constraints (i.e. $V(\hat{z}_n) > \epsilon$ where $V(\hat{z}_n) = \frac{1}{n}\sum_i^n V^{(i)}$ and $V^{(i)} = 1$ only if $\overline{y}_{\boldsymbol{p}}(x) > y(x) > \underline{y}_{\boldsymbol{p}}(x)$), is no greater than β. The scenario approach gives a simple analytic form for the connection between ϵ and β in the case that the optimisation program is convex:

$$\beta = \frac{1}{\epsilon}\frac{d}{n+1}, \tag{3.22}$$

where n is the number of constraint samples in the training data set used to solve the scenario program, and d is the dimensionality of the design variable, z. For a fixed d and n we obtain a confidence-reliability plot as shown in Figure 3.4. The plot demonstrates that by decreasing ϵ slightly, $1-\beta$ can be made to be insignificantly small. Other tighter bounds exist in the more recent scenario optimisation literature, e.g. [30, 32, 1], for example

$$\beta = \sum_{i=0}^{d-1} \binom{n}{i} \epsilon^i (1-\epsilon)^{n-i}. \tag{3.23}$$

Crucially the assessment of $V(\hat{z}_n)$ is possible a priori, although other techniques exist [12]. Care et al. [39] analyse the reliability of solutions of the maximum error loss functions (Eqn. 3.20) in the scenario framework when $\hat{y}(x)$ is convex in x and the function weights.

In the convex case, the a priori assessment is made possible by the fact that the number of support constraints (the number of constraints which if removed result in a more optimal solution) for a convex program is always bounded by the dimensionality of the design variable. Campi and Garatti [33] explore this connection for convex programs in further detail, by analysing the number of support constraints after a solution is obtained. In fact, the bound in Eqn. 3.23 if often overly conservative, because in many cases the number of support constraints is less than the dimensionality of the design variable, and hence a more accurate bound on the reliability of the IPM can be obtained. The improved bound is given by letting ϵ be a function of the number of support constraints s_n^* such that $\epsilon(s_n^*) = 1 - t(s_n^*)$. Then for $0 < \beta < 1$ and $0 < s_n^* < d$ the equation

$$\frac{\beta}{n+1} \sum_{m=k}^{n} \binom{m}{k} t^{m-k} - \binom{n}{k} t^{n-k} = 0 \tag{3.24}$$

has one solution, $t(k)$ in the interval $[0, 1]$.

This idea has a deep connection with the concept of regularisation in machine learning [31]. Garatti and Campi [72] demonstrates how the number of support constraints of a scenario program can be used to iteratively increase the number of sampled constraints, which requires fewer sampled constraints in total than Eqn. 3.23 for equivalent ϵ and β. For a non-convex program, the number of support constraints is not necessarily less than the dimensionality of the design variable, and therefore a new approach is required, which we describe in the following section.

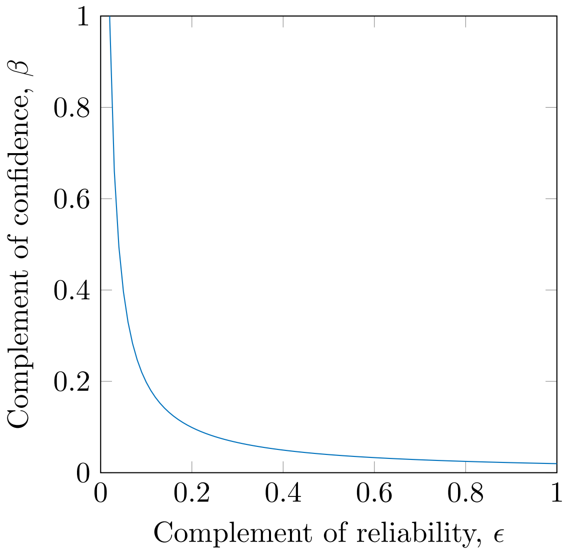

Figure 3.4: Plot of Eqn. 3.22 for $n = 100$ and $d = 2$.

Non-convex case

Campi et al. [36] provide the following bound for the non-convex case:

$$P^n(V(\hat{z}_n) > \epsilon(s)) < \beta, \tag{3.25}$$

where

$$\epsilon(s) = \begin{cases} 1, & \text{for } s = n, \\ 1 - \sqrt[n-s]{\frac{\beta}{n\binom{n}{s}}}, & \text{otherwise,} \end{cases} \tag{3.26}$$

and s is the cardinality of the support set (in other words, the number of support constraints). The behaviour of this bound is similar to the convex case since in general increasing n should increase the size of the support set.

Finding the cardinality of the support set is in general a computationally expensive task, since the scenario program must be solved n times. Campi et al. [35] present a time-efficient algorithm which only requires that the scenario problem is solved s times.

A posteriori frequentist analysis

When data is available in abundance, as is typically the case in most machine learning tasks where a neural network is currently used, $V(\hat{z}_n)$ can be evaluated more easily by using a test set to collect samples from $V(\hat{z}_n)$. Estimating $V(\hat{z}_n)$ is similar to estimating a probability of failure in the well known reliability theory. Therefore one can construct a Monte Carlo estimator of $V(\hat{z}_n)$, or use more advanced techniques from reliability analysis if it is possible to interact with the data generating mechanism. For example, if the number of test data points is large we can use the normal approximation Monte Carlo estimator of $V(\hat{z}_n)$ with $V(\hat{z}_n) \approx \frac{N_v}{N_t}$ and standard deviation $\sqrt{\frac{\frac{N_v}{N_t}(1-\frac{N_v}{N_t})}{N_t}}$, on a test set of size N_t, where N_v data points fall outside the interval bounds of the neural network.

A particularly robust method of estimating the probability of a binary outcome involves using the binomial confidence bounds. In this case specifically, one can bound $V(\hat{z}_n)$ with the desired confidence using the binomial confidence bounds:

$$\sum_{i=0}^{N_t-N_v} \binom{N_t}{i} (1-\underline{v})^i \underline{v}^{N_t-i} = \frac{\beta}{2} \tag{3.27}$$

and

$$\sum_{i=N_t-N_v}^{N_t} \binom{N_t}{i} (1-\overline{v})^i \overline{v}^{N_t-i} = \frac{\beta}{2}, \tag{3.28}$$

where $P(V(\hat{z}_n) < \overline{v} \cap V(\hat{z}_n) > \underline{v}) = \beta$. Estimating $V(\hat{z}_n)$ using a test set also offers the advantage that when the neural network is used for predictions on a different data set, $V(\hat{z}_n)$ can be evaluated easily. If the value of $V(\hat{z}_n)$ obtained on the test set is higher than that on the training dataset, one can apply regularisation in order to implicitly reduce the size of the support set and increase $V(\hat{z}_n)$ on the test set (e.g. dropout regularisation, or ℓ_2 regularisation on the weights).

This methodology is ideal for models with a complex training scheme, where determining the support set would be prohibitively expensive. Note that the probabilistic assessment of the reliability of the model takes place separately from the training of the regression model, such that it is still robust, even if there is a problem with the regression model training. This is an important advantage over Variational Inference methods which are often used with neural networks.

3.3.3 Software for interval predictor models

Patelli et al. [134] describe the first open source software implementation of interval predictor models in the generalised uncertainty quantification software OpenCossan, which is written in Matlab. The OpenCossan software allows convex IPMs to be trained, with hyper-rectangular uncertainty sets. The OpenCossan software is modular and allows the IPMs to be automatically trained as approximations of expensive engineering models, and then used in other engineering calculations, e.g. design optimisation. A partial Python port of the OpenCossan IPM code was released as open source software by Sadeghi [153].

The introduced software has been applied in [23], to study fatigue damage estimation of offshore wind turbines jacket substructure.

3.4 Chapter summary

This chapter presents a review of regression models with predictive uncertainty which can be used to describe the relationship between variables in engineering models, in addition to describing how the models can be created in practice from data. We reviewed probabilistic models and non-probabilistic models. Probabilistic regression models use probability distributions to express information about the variability and uncertainty in the modelled output; they are currently the most widely used regression models. Non-probabilistic models are useful in cases where only limited or imprecise data may be available, and prior knowledge of regression model parameters may be difficult to obtain. A particular advantage of Convex IPMs are the a priori bounds on the model bound violation, which can be used to validate the model at training time without test data.

Chapter 4

Reliability Analysis

4.1 Reliability analysis with random variables

4.1.1 Problem definition

Reliability theory

The aim of structural reliability analysis is to compute the probability that the performance of a system is less than some specified threshold; this probability is known as the failure probability of the system. Firstly, the performance of the system, $g(\boldsymbol{x})$, is defined as a function of the vector of system variables, $\boldsymbol{x} = (x_1, x_2, \ldots, x_i, \ldots)$. The performance function is negative when the system fails, and otherwise positive. Then the failure probability can be found by solving the integral

$$P_f = P(g(\boldsymbol{x}) < 0) = \int \mathbb{I}_f(\boldsymbol{x}) f_X(\boldsymbol{x}) d\boldsymbol{x}, \tag{4.1}$$

where the indicator function, $\mathbb{I}_f(\boldsymbol{x})$, is defined as

$$\mathbb{I}_f(\boldsymbol{x}) = \left\{ \begin{array}{ll} 1, & \text{for } g(\boldsymbol{x}) < 0 \\ 0, & \text{for } g(\boldsymbol{x}) \geq 0 \end{array} \right\}, \tag{4.2}$$

and the probability density function of the system random variables is $f_X(\boldsymbol{x})$ [115]. It is common for the performance function to be defined in the load resistance form, e.g.

$$g(\boldsymbol{x}) = \sum_{i \in R} x_i - \sum_{i \in L} x_i, \tag{4.3}$$

where R are indices corresponding to resistance factors and L are indices corresponding to load factors, so that the system fails when the sum of loads is greater than the sum of resistances [143]. When the resistance and load are balanced, $g(\boldsymbol{x}) = 0$ and the system is on the interface of the safety and failure regions. The limit state surface is specified by the $\boldsymbol{x}$ for which $g(\boldsymbol{x}) = 0$.

In fault tree analysis the failure event of a system is written in terms of failure events for smaller subsystems or components, using Boolean algebra. Probability arithmetic can be used with the fault tree to combine failure probabilities for individual subsystems to obtain the failure probability for the whole system. This requires knowledge of the dependencies between the probability of failure events for the considered sub-systems [115].

Reliability based design optimisation

In reliability based design optimisation (RBDO), a cost function, e.g. the weight or construction cost of the system, is minimised subject to the constraint that the failure probability of the system does not fall below a certain value. The reliability based design optimisation problem can be stated as the optimisation program

$$\arg\min \{\text{cost}(\boldsymbol{d}) : P_f(\boldsymbol{d}) < P_{\text{target}}\}, \tag{4.4}$$

where $\boldsymbol{d}$ is the vector of design variables, $\text{cost}(\boldsymbol{d})$ is the cost function of the design, $P_f(\boldsymbol{d})$ is the failure probability of the design, and P_{target} is the target failure probability. Usually the vector of design variables, $\boldsymbol{d}$, will be parameters of the random variables, $f_X(\boldsymbol{x})$, associated with the resistance, such that Eqn. 4.4 usually finds a balance between a cost effective design and a design where the resistance of the system is sufficiently greater than the load [115].

In engineering practice the model of the structure is often computationally expensive to evaluate, and therefore it may be more convenient to find a sub-optimal solution to Eqn. 4.4, by designing the structure based on engineering judgement. Approximate rules of thumb, such as partial safety factors, allow the reliability of the system to be constrained approximately using analytical equations [115]. Then the full reliability analysis can be performed with the proposed design to ensure that the reliability of the system satisfies the constraints in Eqn. 4.4. Hence a safe and efficient design can be obtained with reduced computational effort.

Sensitivity analysis

Sensitivity analysis allows the effect of each uncertain variable on the variability of the model response to be quantified. This can be achieved either by local methods, which describe variability of the model response at the expected value of the system variables, or global methods, which describe the total variability of the model response.

Local sensitivity analysis is often achieved by evaluating the first derivative of the model response with respect to the system variables. Similarly, Birnbaum [20] defines component importance as the partial derivative of the system reliability with respect to the reliability of the component.

Global sensitivity analysis is usually performed by evaluating the Sobol indices,

$$S_i = \frac{\mathrm{Var}_{x_i}(\mathbb{E}_{x_{\sim i}}(g(\boldsymbol{x})))}{Var(g(\boldsymbol{x}))}, \tag{4.5}$$

which describe the contribution of the variance of x_i to the total variance of the model response $g(\boldsymbol{x})$, when only x_i is varied. $x_{\sim i}$ represents all random variables other than x_i. The Total Effect indices, given by

$$T_i = 1 - \frac{\mathrm{Var}_{x_{\sim i}}(\mathbb{E}_{x_i}(g(\boldsymbol{x})))}{Var(g(\boldsymbol{x}))}, \tag{4.6}$$

include the effect of interactions caused by varying x_i whilst varying other variables [155].

The most simple way to evaluate the Sobol indices and total effect indices is by using a Monte Carlo estimator for the expectation and variance terms in Eqns. 4.5 and 4.6. In some cases, for example when the model has too many parameters or the model is very computationally expensive, it is necessary to use a more complex method to compute the Sobol and total sensitivity indices. For example, the upper bound of the total sensitivity index can be efficiently calculated by integrating the local sensitivity analysis over the whole space of the inputs [131], and the Sobol indices can be efficiently calculated by use of the Fourier Amplitude Sensitivity Testing (FAST) method [170].

4.1.2 Methods to compute the failure probability

In some circumstances the failure probability can be computed analytically, for example when the system variables $\boldsymbol{x}$ are normally distributed and the performance function is linear [115]. However, often a closed form solution of Eqn. 4.1 is not available and hence alternative methods must be used.

Monte Carlo simulation

In general, the failure probability can be computed by Monte Carlo simulation, as discussed for general functions in Chapter 2. The Monte Carlo estimator for the failure probability is

$$\hat{P}_f = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_f(\boldsymbol{x}^{(i)}), \tag{4.7}$$

where N samples, $\boldsymbol{x}^{(i)}$, are drawn from the the probability density function of the system random variables $f_X(\boldsymbol{x})$. The coefficient of variation of the failure probability estimator is

$$\mathrm{CoV}[\hat{P}_f] = \sqrt{\frac{1-P_f}{NP_f}}. \tag{4.8}$$

Therefore, obtaining order of magnitude estimates of P_f with Monte Carlo simulation requires at least $\frac{1}{P_f}$ samples, and for an accurate estimate even more samples are required. If P_f is small and $g(\boldsymbol{x})$ is expensive to evaluate then the number of samples required is unreasonably large, and more efficient strategies are required to evaluate P_f.

Efficient sampling strategies

Several sampling strategies have been proposed to choose a set of samples which can be used to reduce the variance of the Monte Carlo estimator in Eqn. 4.7 without expending additional computational effort. Low-discrepancy sampling strategies aim to choose a set of samples which cover the sampling domain with the desired density. This is often not the case with a small random set of samples, which may fall disproportionately in one area of the sampling domain before the law of large numbers takes effect. Stratified Sampling strategies, such as Latin Hypercube Sampling divide the probability density of the system variables into an n-dimensional grid, where each grid element contains equal probability density. Then a sample may be chosen at random in each grid element, resulting in a set of samples which covers the sampling domain well. For linear functions it can be shown that this sampling strategy has a lower variance than the traditional Monte Carlo estimator [86]. The main disadvantage of efficient sampling strategies is that the reduction in the coefficient of variation is small when compared to other techniques, e.g. Line Sampling.

First Order Reliability Method

The First Order Reliability Method (FORM) allows the probability of failure of a system to be computed without Monte Carlo simulation. Assuming the system variables are distributed normally and independently, the probability of failure can be obtained analytically if the performance function, $g(\boldsymbol{x})$, is linear. If the performance function is not linear, a Taylor expansion can be used to find a linear approximation of the limit state function as shown in Figure 4.1. If the system random variables are not normally distributed then a transformation must first be applied to the random variables and the limit state function, so that FORM can be applied [115].

The performance function, $g(\boldsymbol{x})$, is written as the Taylor series expansion

$$g(\boldsymbol{x}) = g(\boldsymbol{x}^*) + (\boldsymbol{x} - \boldsymbol{x}^*)\nabla g(\boldsymbol{x}^*) + \ldots = (\boldsymbol{x} - \boldsymbol{x}^*)\nabla g(\boldsymbol{x}^*) + \ldots, \tag{4.9}$$

about the point $\boldsymbol{x}^*$, which is usually chosen to be the point on the limit state surface with the highest probability density. This point is known as the design point, and can be obtained by solving the optimisation program

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}}\{|\boldsymbol{x}|^2 : g(\boldsymbol{x}) = 0\}. \tag{4.10}$$

Alternatively, using the assumption of a linear performance function, the design point can be determined using the gradient of the performance function. The reliability index is defined as $\beta = \sqrt{|\boldsymbol{x}^*|^2}$, and in the case of normally distributed random variables and a linear limit state function $P_f = \phi(-\beta)$. This can be shown by observing that when $\boldsymbol{x}^*$ has a standard normal distribution and $g(\boldsymbol{x})$ is linear (as in Eqn. 4.9), the system performance will have a normal distribution with mean

$$\mathbb{E}_{\boldsymbol{x}}(g(\boldsymbol{x})) = (\mathbb{E}_{\boldsymbol{x}}(\boldsymbol{x}) - \boldsymbol{x}^*)\nabla g(\boldsymbol{x}^*) = -\boldsymbol{x}^*\nabla g(\boldsymbol{x}^*) \tag{4.11}$$

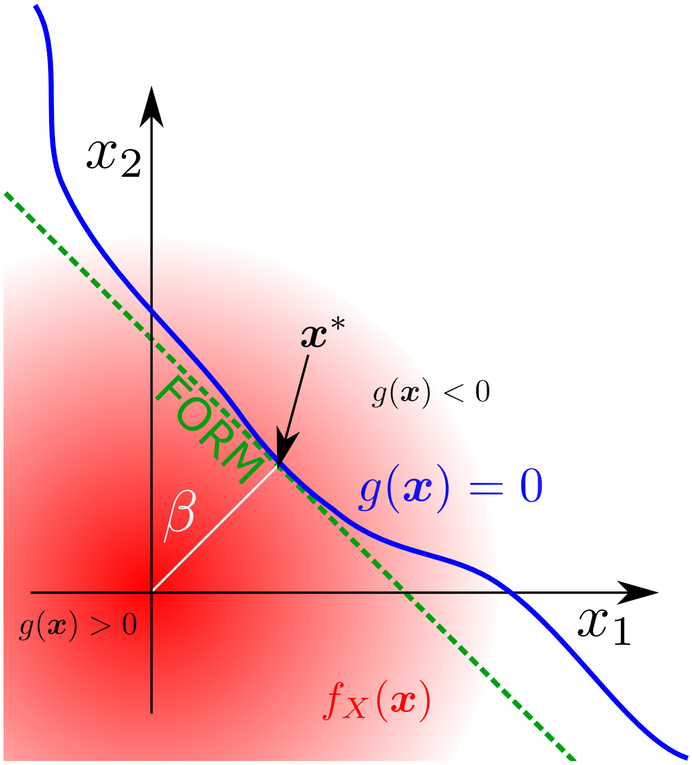

Figure 4.1: A diagram of the First Order Reliability Method for two system variables, shown with random variables in the standard normal space.

and variance

$$\mathrm{Var}_{\boldsymbol{x}}(g(\boldsymbol{x})) = |\nabla g(\boldsymbol{x}^*)|^2. \tag{4.12}$$

Therefore, since $\boldsymbol{x}^* = \beta \frac{\nabla g(\boldsymbol{x}^*)}{\sqrt{|\nabla g(\boldsymbol{x}^*)|^2}}$, $P_f = \phi\left(\frac{-\mathbb{E}_{\boldsymbol{x}}(g(\boldsymbol{x}))}{\sqrt{\mathrm{Var}_{\boldsymbol{x}}(g(\boldsymbol{x}))}}\right)$ leads to the desired expression.

A similar method relying on a more accurate approximation of the limit state function is the Second Order Reliability Method.

The main advantage of FORM is the small number of samples required to estimate the failure probability. The main disadvantage of FORM is that for non-linear limit state surfaces the method is likely to be extremely inaccurate, due to the degradation of the Taylor series approximation for limit state surfaces with high curvature. Non-linear limit state surfaces are often induced by the transformation of the system's random variables to the standard normal space.

Line sampling

The fundamental idea behind Line Sampling is to refine estimates obtained from the First-order reliability method (FORM), which may be incorrect due to the non-linearity of the limit state function. Conceptually, this is achieved by averaging the result of different FORM simulations [51]. Firstly, the approximate direction of the failure region from the origin in standard normal space must be determined. This is known as the importance direction. It is usually obtained by finding the design point, by approximate means if necessary. Following this, samples are randomly generated in the standard normal space and lines are drawn parallel to the importance direction in order to compute the distance to the limit state function, which enables the probability of failure to be estimated for each sample.

For each sample of $\boldsymbol{x}$, the probability of failure in the line parallel to the important direction is defined as:

$$P_f(\boldsymbol{x}) = \int_{-\infty}^{\infty} \mathbb{I}(\boldsymbol{x} + \beta\boldsymbol{\alpha}) d\beta, \tag{4.13}$$

where $\boldsymbol{\alpha}$ is the importance direction, and ϕ is the probability density function of a Gaussian distribution (and β is a real number). In practice, the roots of a nonlinear function must be found to estimate the partial probabilities of failure along each line. This is either done by interpolation of a few samples along the line, or by using the Newton-Raphson method. The global probability of failure is the mean of the probability of failure on the lines:

$$P_f = \frac{1}{N_L} \sum_{i}^{N_L} P_f^{(i)} \tag{4.14}$$

where N_L is the total number of lines used in the analysis, and $P_f^{(i)}$ are the partial probabilities of failure estimated along all the lines.

For problems in which the dependence of the performance function is only moderately non-linear with respect to the parameters modelled as random variables, setting the importance direction as the gradient vector of the performance function in the underlying standard normal space leads to highly efficient Line Sampling. De Angelis [50] describes enhancements which can be made to Line Sampling to increase the efficiency. For example, the solution of the Newton-Raphson search used on the previous line can be used to inform the search on the next line, if the lines are sorted by proximity. In addition, the importance direction can be updated during simulation based on the completed subset of lines.

The Line Sampling methodology is more expensive than FORM, but far less expensive than Monte Carlo simulation. It is likely to perform poorly for highly non-linear limit state surfaces, but in general offers a good balance between accuracy and computational expense.

Importance sampling

In Importance Sampling, samples are drawn from a distribution with a higher density in the failure region and then re-weighted to obtain a Monte Carlo estimator with reduced variance. The re-weighted estimator is written as

$$P_f = \int h(\boldsymbol{x}) \frac{\mathbb{I}_f(\boldsymbol{x}) f_X(\boldsymbol{x})}{h(\boldsymbol{x})} d\boldsymbol{x} = \frac{1}{N} \sum_{i=1}^{N} \frac{\mathbb{I}_f(\boldsymbol{x}_i) f_X(\boldsymbol{x}_i)}{h(\boldsymbol{x}_i)}, \tag{4.15}$$

where $\boldsymbol{x}_i$ are drawn from the proposal density $h(\boldsymbol{x})$. The optimal proposal density, which results in the greatest reduction of the variance of the estimator is

$$h(\boldsymbol{x})_{\text{optimal}} = \frac{\mathbb{I}_f(\boldsymbol{x}) f_X(\boldsymbol{x})}{P_f}, \tag{4.16}$$

which is not useful in practice because of the dependence on the quantity to be estimated, P_f. However, the optimal proposal density can be used to motivate the choice of the proposal density in practice. An appropriate $h(\boldsymbol{x})$ can be chosen by finding the design point with an approximate method and centring the proposal density on the design point, since Eqn. 4.16 indicates that the failure region has a higher proposal density. A complete discussion of the technique is given in Schuëller and Stix [159] and Melchers and Beck [115].

Importance Sampling is useful as it offers an unbiased estimator which can estimate the failure probability with few samples. The main difficultly is determining the proposal distribution $h(\boldsymbol{x})$. This is usually achieved by engineering judgement and knowledge of the design point.

Subset simulation

Subset simulation aims to calculate P_f by decomposing the space of the random variables into several intermediate failure events with decreasing failure probability. The conditional probabilities for the intermediate failure regions can then be used to calculate P_f which is given by

$$P_f = P(F_m) = P(F_m) \prod_{i=1}^{m-1} P(F_{i+1}|F_i) \tag{4.17}$$

where F_i represents intermediate failure event i. By making the conditional probability of samples falling in the intermediate failure regions large, the coefficient of variation of each individual failure event can be minimised, hence minimising the coefficient of variation of P_f. Markov chains are used to generate conditional samples between intermediate failure regions in order to calculate $P(F_{i+1}|F_i)$. A complete description of the method is given in Au and Beck [8].

The main advantage of Subset Simulation is that it can estimate failure probabilities for non-linear limit state surfaces in a black box manner with relatively low computational expense. However, Breitung [25] shows that subset simulation is not accurate for some limit state surfaces, for example limit state surfaces with multiple importance directions.

Metamodels

If inexpensive samples of $g(\boldsymbol{x})$ or $\mathbb{I}_f(\boldsymbol{x})$ are available then the estimator $\hat{P}_f$ can be evaluated trivially. Therefore, the problem of estimating P_f can be effectively reduced to a machine learning problem. In the case of modelling $g(\boldsymbol{x})$, the problem is one of regression. In the case of modelling $\mathbb{I}_f(\boldsymbol{x})$, the problem is classification of the failure region. A machine learning model which fulfils this purpose is known as a metamodel or surrogate model.

In the literature many machine learning techniques have been applied to the reliability analysis problem: linear regression (known as the response surface methodology) [27], support vector machines [148], polynomial chaos expansions [19], neural networks [160] and Gaussian process emulators (sometimes known as Kriging) [94]. Neural networks and Gaussian processes have the advantage of being able to quantify their uncertainty accurately, so the required number of training samples can be assessed. Polynomial chaos expansions allow the sensitivity indices of $g(\boldsymbol{x})$ to be computed analytically from the trained metamodel [164].

In general, the most useful metamodels produce the most accurate estimates of P_f, whilst requiring the smallest number of training samples. An ‘experimental design’ specifies where the samples of $g(\boldsymbol{x})$ will be made for training. Usually a uniform design is chosen, but other sampling strategies can be used [165]. Active learning can be used to sequentially choose the samples required to train the metamodel. These samples are usually chosen based on where the uncertainty of the metamodel is largest. In adaptive Kriging Monte Carlo simulation (AK-MCS) the samples are chosen at points with large uncertainty, close to the limit state surface. This strategy achieves state of the art efficiency [56]. This strategy is known as active learning, and the function which is used to choose the subsequent sample is known as the probability of misclassification acquisition function.

The main advantage of metamodels is that the estimator for the failure probability based on the metamodel can be made arbitrarily accurate. The main disadvantage is that the metamodel introduces uncertainties, so the problem is effectively shifted to trying to create an accurate metamodel with a small number of samples.

4.2 Convex set models for reliability

4.2.1 Problem definition

In contrast to probabilistic models of uncertainty, where the statistics of $g(\boldsymbol{x})$ — and hence the failure probability — can be determined, in convex uncertainty models the focus is on the best and worst possible values for $g(\boldsymbol{x})$. This is good, because underestimation of the worst case due to inaccurate sampling in the tails of probability distributions can be avoided. In addition, the analysis can include extreme lack of knowledge of the possible loads the system will be subjected to. However, sometimes the analysis may be overly conservative if the worst case is extremely unlikely to occur. Convex set models of uncertainty also offer a framework to analyse a possible set of future designs, before the design for a system has been finalised.

For system parameters $\boldsymbol{x} \in X$, the structural response is given by the interval $[\min_{\boldsymbol{x}\in X} g(\boldsymbol{x}), \max_{\boldsymbol{x}\in X} g(\boldsymbol{x})]$, which can be determined by the methods discussed in Section 2.1.2 [57]. For a system with response $g_{\text{load}}(x)$ which must not exceed $g_{\text{threshold}}$, Ben-Haim [14] proposes a non-probabilistic figure of merit,

$$R_{\text{non-probabilistic}} = 1 - \frac{\overline{g}_{\text{load}}(\boldsymbol{x})}{g_{\text{threshold}}}, \tag{4.18}$$

where $\overline{g}_{\text{load}}(x) = \max_{\boldsymbol{x}\in X} g_{\text{load}}(\boldsymbol{x})$. $R_{\text{non-probabilistic}}$ describes how close the system is to failure, but it cannot be interpreted as a probability or rate of failure.

4.3 Reliability analysis with probability boxes

4.3.1 Problem definition

Probability boxes offer a hybrid of the convex set and probabilistic approaches for reliability analysis. In structural reliability analysis with probability boxes, the objective is to compute the failure probability in the same sense as with traditional random variables in Section 4.1.1. However, it is now impossible to compute an exact value for the failure probability; only bounds on the failure probability are available [4]. For distributional probability boxes, $f_{X_{\boldsymbol{\theta}}}(\boldsymbol{x})$ for $\boldsymbol{\theta} \in \Theta$, the bounds on the failure probability can be found by solving the integrals

$$\underline{P}_f = \underline{\mathrm{P}}[g(\boldsymbol{x}) < 0] = \min_{\boldsymbol{\theta} \in \Theta} \int_{\mathbb{R}^N} \mathbb{I}_f(\boldsymbol{x}) f_{X_{\boldsymbol{\theta}}}(\boldsymbol{x}) d\boldsymbol{x}, \tag{4.19}$$

and

$$\overline{P}_f = \overline{\mathrm{P}}[g(\boldsymbol{x}) < 0] = \max_{\boldsymbol{\theta} \in \Theta} \int_{\mathbb{R}^N} \mathbb{I}_f(\boldsymbol{x}) f_{X_{\boldsymbol{\theta}}}(\boldsymbol{x}) d\boldsymbol{x}. \tag{4.20}$$

For distribution-free probability boxes given by $[\underline{F}_i(x_i), \overline{F}_i(x_i)]$, each system variable can be written as a function of separate probabilistic and set based variables as $x_i' = \underline{F}_i^{-1}(\alpha_i) + (\overline{F}_i^{-1}(\alpha_i) - \underline{F}_i^{-1}(\alpha_i))\theta_i$, where the aleatory variable $\alpha = (\alpha_1, \alpha_2, \ldots)$ is a uniformly distributed random vector with the same dimensionality as $\boldsymbol{x}$, and $\boldsymbol{\theta} \in \Theta$ is the unit hypercube with the same dimensionality as $\boldsymbol{x}$ [50]. This enables the performance function $g(\boldsymbol{x})$ to be rewritten in terms of $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$, i.e. $g(\boldsymbol{\alpha}, \boldsymbol{\theta})$. Bounds on the failure probability can then be obtained by evaluating

$$\underline{P}_f = \underline{\mathrm{P}}[g(\boldsymbol{x}) < 0] = \mathrm{P}[\underline{g}(\boldsymbol{\alpha}) < 0] \tag{4.21}$$

and

$$\overline{P}_f = \overline{\mathrm{P}}[g(\boldsymbol{x}) < 0] = \mathrm{P}[\overline{g}(\boldsymbol{\alpha}) < 0], \tag{4.22}$$

where the upper and lower performance function are obtained from

$$\underline{g}(\boldsymbol{\alpha}) = \min_{\boldsymbol{\theta} \in \Theta} g(\boldsymbol{\alpha}, \boldsymbol{\theta}) \tag{4.23}$$

and

$$\overline{g}(\boldsymbol{\alpha}) = \max_{\boldsymbol{\theta} \in \Theta} g(\boldsymbol{\alpha}, \boldsymbol{\theta}) \tag{4.24}$$

By finding the envelope of a distributional probability box, the algorithm for computing the failure probability for the distribution-free case can be applied. As expected, Fetz and Oberguggenberger [64] demonstrate that this results in overly conservative bounds on the failure probability, since clearly information is lost by taking the envelope of the distributional probability box. Therefore, only Eqn. 4.19 and Eqn. 4.20 should be applied when computing failure probabilities with distributional probability boxes.

4.3.2 Methods to compute the failure probability

In this section we concentrate on methods to compute the failure probability for distributional probability boxes.

Monte Carlo estimators

A Monte Carlo estimator can be applied for the integrals in the failure probability computation with distribution-free probability boxes (Eqns. 4.19 and 4.20) to yield the bounds

$$[\underline{P}_f, \overline{P}_f] = [\min_{\boldsymbol{\theta}\in\Theta} \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_f(\boldsymbol{x}_{\boldsymbol{\theta}}^{(i)}), \max_{\boldsymbol{\theta}\in\Theta} \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_f(\boldsymbol{x}_{\boldsymbol{\theta}}^{(i)})], \tag{4.25}$$

where the samples $\boldsymbol{x}_{\boldsymbol{\theta}}^{(i)}$ are drawn from $f_{X_{\boldsymbol{\theta}}}(\boldsymbol{x})$. This is the double loop Monte Carlo approach; an inner loop is used to compute a Monte Carlo estimator which is optimised over in the outer loop [133]. The outer loop optimisation can be evaluated using brute force grid sampling of $\boldsymbol{\theta}$, which is known as naïve double loop Monte Carlo simulation. It is usually more efficient to use an efficient global optimisation algorithm to evaluate the optimisation loop, such as Bayesian Optimisation, or a genetic algorithm [50].

Evaluating the failure probability using double loop Monte Carlo simulation is computationally expensive, since now each inner loop Monte Carlo estimator must be computed multiple times. This is particularly the case when the problem dimensionality is large or the failure probability is small. Troffaes [173] shows that the bias of the estimator is negative, and the magnitude of the bias decreases as more samples are made.

Imprecise First Order Reliability Method

A generalisation of FORM for systems with components which are described by probability boxes was introduced by Qiu et al. [144]. The system's performance function must be written in the load resistance form (Eqn. 4.3), and the system must have one strength and one load component. Therefore, the system variables consist of the resistance variable with mean $\mu_R \in [\underline{\mu_R}, \overline{\mu_R}]$ and standard deviation $\sigma_R \in [\underline{\sigma_R}, \overline{\sigma_R}]$, and load variable with mean $\mu_L \in [\underline{\mu_L}, \overline{\mu_L}]$ and standard deviation $\sigma_L \in [\underline{\sigma_L}, \overline{\sigma_L}]$. Then the failure probability lies in the interval $[\underline{P}_f, \overline{P}_f] = [\phi(-\overline{\beta}), \phi(-\underline{\beta})]$, where

$$\overline{\beta} = \frac{\overline{\mu}_R - \underline{\mu_L}}{\underline{\sigma_L}^2 + \underline{\sigma_R}^2}, \tag{4.26}$$

and

$$\underline{\beta} = \frac{\underline{\mu_R} - \overline{\mu}_L}{\overline{\sigma}_L^2 + \overline{\sigma}_R^2}. \tag{4.27}$$

In more complex cases, one may need to solve an optimisation program to find the reliability index [90]. For example, one could imagine a system which fails if the sum of many different products of probability box distributed variables falls below a threshold.

Line sampling

De Angelis [50] describes two ways in which Line Sampling can be used to increase the efficiency of probability box propagation. Line Sampling can be applied as an alternative to the Monte Carlo estimator used to approximate the integral in the double loop approach (Eqns. 4.19 and 4.20). Alternatively, Line Sampling can be applied to the aleatory variables $\boldsymbol{\alpha}$ for the upper and lower performance functions in Eqns. 4.21 and 4.22. When Line Sampling is applied in the aleatory space, the importance direction updating strategy proposed by De Angelis [50] significantly increases the accuracy of the computation. Judged by number of samples required for computation, Line Sampling is close to the state of the art. However, Line Sampling is ineffective on highly non-linear limit state surfaces.

Importance sampling

The Importance Sampling estimator in Eqn. 4.15 can be applied to greatly reduce the number of samples required when computing the failure probability for a system subject to probability box random variables [64]. The bounds on the failure probability are given by

$$\underline{P}_f = \min_{\boldsymbol{\theta} \in \Theta} \int \mathbb{I}_f(\boldsymbol{x}) \frac{f_{X_{\boldsymbol{\theta}}}(\boldsymbol{x})}{h(\boldsymbol{x})} h(\boldsymbol{x}) d\boldsymbol{x} = \min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}_f(\boldsymbol{x}^{(i)}) \frac{f_{X_{\boldsymbol{\theta}}}(\boldsymbol{x}^{(i)})}{h(\boldsymbol{x}^{(i)})} \tag{4.28}$$

and

$$\overline{P}_f = \max_{\boldsymbol{\theta} \in \Theta} \int \mathbb{I}_f(\boldsymbol{x}) \frac{f_{X_{\boldsymbol{\theta}}}(\boldsymbol{x})}{h(\boldsymbol{x})} h(\boldsymbol{x}) d\boldsymbol{x} = \max_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}_f(\boldsymbol{x}^{(i)}) \frac{f_{X_{\boldsymbol{\theta}}}(\boldsymbol{x}^{(i)})}{h(\boldsymbol{x}^{(i)})}, \tag{4.29}$$

where the samples $\boldsymbol{x}^{(i)}$ are drawn from the proposal distribution $h(\boldsymbol{x})$. The proposal distribution can be iteratively updated to provide more accurate results. Decadt et al. [52] demonstrates that the bias in both cases is negative and decreases in magnitude as more samples are collected. The Importance Sampling estimator requires a similar number of samples to Line Sampling.

Multi level metamodels

Multilevel Metamodelling requires the creation of two Gaussian Process emulators [158]. The first metamodel is created for the performance function in the space of the system variables $\boldsymbol{x}$, using Adaptive Kriging Monte Carlo simulation. The second metamodel is used to perform Bayesian Optimisation on the obtained failure probability from the first metamodel, in the epistemic space. This greatly reduces the amount of repeated similar evaluations of the system model, and hence the Multi-level metamodelling technique is close to state of the art when judged by number of required samples.

4.4 Chapter summary

In this chapter we reviewed the application of the uncertainty models introduced in Chapter 2 to the field of reliability engineering. Specifically, this chapter demonstrated how probabilistic, set-based and imprecise probability models can be used to calculate the reliability of systems under uncertainty. The optimal design of a system under uncertainty can also be computed, and the local or global sensitivity of the response of a system to changes in the system variables can be determined. For systems where the probability of failure is small, computing the failure probability using a Monte Carlo estimator can be computationally expensive. For this reason, it is necessary to apply advanced techniques in order to calculate the probability of failure in a feasible computational time. For imprecise probability models, computation of the failure probability of the system is even more expensive, and hence efficient computational techniques are also required. This motivates the novel contributions introduced in the following chapters.

Chapter 5

Structural Reliability of Pre-stressed Concrete Containments under Distributional Uncertainty

5.1 Introduction

A pre-stressed concrete containment is an important safety related structure as it acts as one of the final barriers to radioactive release. These structures are normally designed in accordance with the allowable stress codes to sustain the specified loading conditions. However, the compliance with the industry standard allowable stress codes does not give any reliable indication of the probability of failure (P_f) if the containment is over-pressurised under postulated beyond design basis events.

In recent years, two international round robin exercises have been conducted which have provided valuable test data related to failure under over-pressurisation. The first exercise involved the numerical analysis of the 1/4 scale steel-lined Pre-stressed Concrete Containment Vessel (PCCV) with design pressure (P_d) of 0.39 MPa which was tested at Sandia National Laboratories (SNL) in USA and has been analysed by Prinja and Shepherd [141]. The second exercise involved the unlined Bhabha Atomic Research Centre (BARC) Containment test model (BARCOM) with P_d of 0.1413 MPa that was tested by the BARC in Tarapur, India and has been analysed by Kamatam and Prinja et al. [142]. These studies are essentially deterministic studies that have helped validate the analysis methodology and modelling techniques that can be used to predict pre-stressed concrete containment capacity and failure modes. Such deterministic analytical and experimental studies have helped to establish the mode of failure, but do not give any indication of the failure probability. Furthermore, the conventional allowable stress codes used to design such containments also do not provide the probability of failure.

In this chapter, methods to calculate the probability of failure for such a containment are described, given the failure mode identified in the previous studies. In addition, the effect of uncertain structural variability distribution parameters on the analysis is considered. As a numerical example, this analysis is performed for the SNL containment.

5.2 Structural model

Both SNL and BARCOM tests have shown that the collapse of the containment structure subjected to internal pressure is not expected to occur soon after the design pressure is exceeded. There is no cliff edge, but a gradual progressive damage of the containment structure under over-pressurisation indicating the safety margin of the structure against collapse. The experiments and the attendant numerical analyses have established the ultimate structural collapse mode of the containments under internal pressure loading which indicates that the failure takes place in the general field of the containment wall around mid-height and away from any major structural discontinuities like the penetrations. This is because robust design procedures have been used that provide adequate compensation and local strengthening to avoid structural failure at discontinuities. In the case of the SNL model shown in Figure 5.1, the failure location at applied pressure (P) of $3.65P_d$ was accurately predicted by the computational model at mid-height of the cylinder in the general area away from the buttress and main penetrations. The BARCOM model is also predicted to fail at mid-height of the cylinder wall as indicated in the deformed shape shown in Figure 5.2. Based on these experimental studies and the attendant numerical analyses, a failure function is presented that assumes first yielding in the hoop direction at mid-height of the cylinder wall.

Failure of a containment structure is dictated by the strain levels experienced by the tendons, rebars and the liner following the tensile cracking of the concrete. The first membrane yield is expected to occur in the hoop direction in the cylinder wall. If the failure

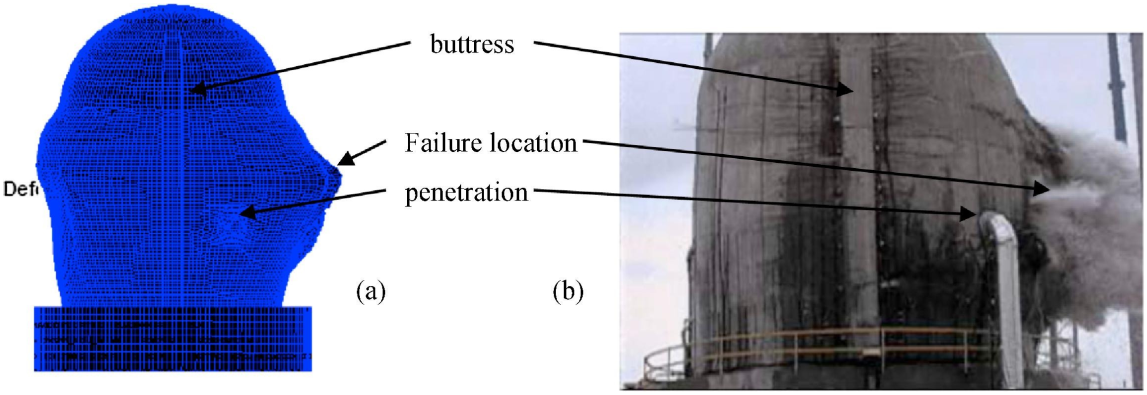

Figure 5.1: Predicted failure mode of the SNL model (a) Finite Element Analysis results vs (b) test at $P = 3.65P_d$.

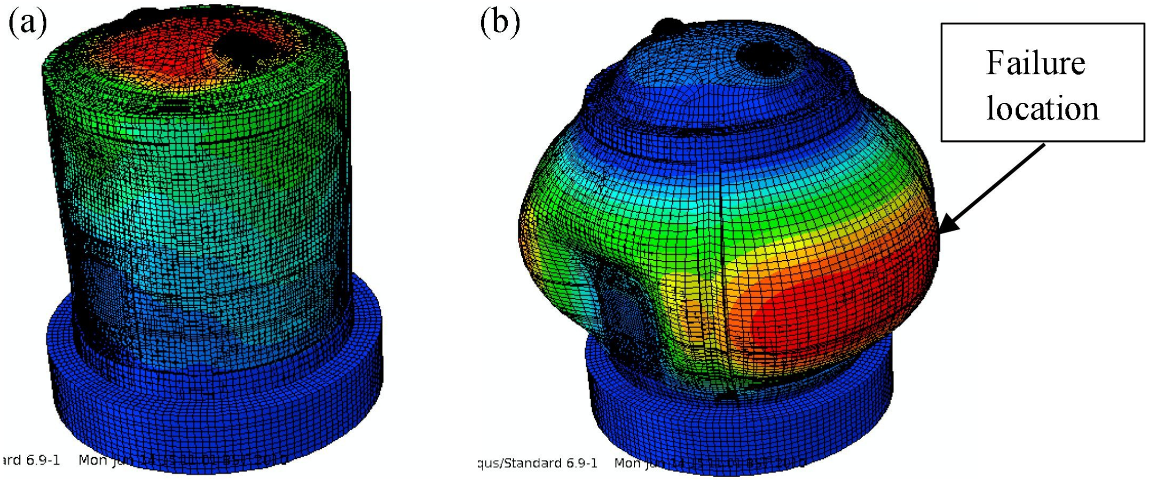

Figure 5.2: Predicted response of the BARC model (a) under prestress only and (b) at $P = 2.89P_d$.

state is defined as the tensile cracking of the concrete and yielding of the tendons, rebars and the liner, then the internal pressure at a specific deformed shape is given by:

$$P = \frac{1}{R}(A_sF_s + A_cF_c + A_lF_l + A_tF_t) \tag{5.1}$$

where A_s, A_c, A_l and A_t are the cross-sectional areas of the rebar steel, concrete, liner plate and tendons respectively given as area per unit height of the cylinder wall. F_s, F_l and F_t are the yield stress of rebar steel, liner plate and tendons respectively and F_c is the tensile strength of the concrete. R is the mid radius of the cylinder wall.

The failure function g can be written as:

$$g = PR - (A_sF_s + A_cF_c + A_lF_l + A_tF_t). \tag{5.2}$$

Structural parameters

We assign normal distributions to R, P_d, F_s, F_l, F_t, F_c, A_s, A_c, A_l and A_t. The mean values for F_s, F_l, F_t and F_c were set to known values [143]. The mean values for R and P_d are set to the measured point values from the design. The mean values for the cross-sectional area properties were obtained from the geometric data, summarised in Table 5.1, as follows. The steel rebar area per unit height is given by

$$A_s = \frac{n_s \pi r_s^2}{h_s}, \tag{5.3}$$

where n_s is the number of steel rebars through the thickness of the wall, r_s is radius of steel rebar and h_s is the vertical spacing. The liner area unit height $A_l =$ thickness of the plate$\times 1$. The tendon area per unit height is

$$A_t = \frac{n_t \pi r_t^2}{h_t}, \tag{5.4}$$

where n_t is the number of tendons through wall thickness, r_t is tendon radius and h_t is tendon vertical spacing. The concrete area per unit height is

$$A_c = ((r_o - r_i) - (A_l + A_s + A_t)) \tag{5.5}$$

Geometric Property	Value
Outside radius of the wall, r_o (mm)	5700
Inner radius of the wall, r_i (mm)	5375
Wall thickness (mm)	325
Wall height (mm)	10750
No. of tendons through wall, n_t	3
Tendon vertical spacing, h_t (mm)	119.4
Tendon radius, r_t (mm)	6.85
No. of rebars through wall, n_s	2
Rebar vertical spacing, h_s (mm)	113
Rebar radius, r_s (mm)	11.1
Liner plate thickness (mm)	1.6

Table 5.1: Summary of geometric data point values for the SNL containment model.

where r_o and r_i are outer and inner radii of the wall, and the mid radius of the wall, $R = \frac{r_o+r_i}{2}$.

In structural reliability analysis for concrete containment capacities, tensile strength of two different types of materials need to be considered: concrete and steel. Concrete behaves like a brittle material whereas steel components like the tendons, rebars and liner plate will exhibit plastic behaviour when loaded beyond their yield stress. In case of concrete, variability in strength can be traced to two fundamentally different sources: variability in the properties of the concrete mixture and ingredients and variability in the way the strength is tested and measured. Similarly, variability in yield strength of a given steel varies due to variation in chemistry, heat treatment and mechanical processing. Typically, it is the compressive strength concrete which is specified and measured. The tensile strength of concrete is taken to be about 10% of its compressive strength at room temperature. At higher temperatures, the strength tends to decrease and any loss in the tensile strength is proportional to the corresponding loss in the compressive strength.

Variability in geometric dimensions of engineered components depends on the manufacturing process and the specified tolerances. Usually, tolerances in manufacturing processes are tight and tend to follow normal distribution. In this example, all material, geometric and loading parameters are assumed to be distributed normally, with a coefficient of variation (CoV) of 0.2. In practice, the CoV in yield strength of steel components could be less than 0.1 and the CoV of geometric dimensions could be even lower. The variables used in the performance function for the SNL containment are summarised in Table 5.2.

Variable	Mean Value	Coefficient of Variation
Concrete tensile strength (MPa), F_c	4.4	0.2
Liner yield (MPa), F_l	382	0.2
Rebar yield (MPa), F_s	465	0.2
Tendon yield (MPa), F_t	1740	0.2
Design Pressure (MPa), P_d	0.39	0.2
Radius (mm), R	5537.5	0.2
Concrete area per unit height (mm), A_c	312.85	0.2
Liner area per unit height (mm), A_l	1.6	0.2
Rebar area per unit height (mm), A_s	6.85	0.2
Tendon area per unit height (mm), A_t	3.7	0.2

Table 5.2: Uncertainty model parameters used for SNL containment. Inputs are independently normally distributed.

5.3 Analysis

The coefficient of variation (CoV) of the defined structural parameters was not known precisely. Therefore, sensitivity analysis was used to determine the variation in the probability of failure with respect to these parameters. The effect of varying the parameters whose variance had the greatest contribution to the variance of the output, i.e. the greatest sensitivity, was considered in greater detail.

The failure probability, P_f for the SNL containment was calculated using the first order reliability method on the performance function g, where the design point was obtained from the iterative algorithm proposed in Rackwitz [146]. Then the calculated failure probabilities were validated using the importance sampling, line sampling and subset simulation methods from OpenCossan, described in Section 4.1.2, as it was found that the failure probability was too small to be evaluated in a short time using standard Monte Carlo simulation. The failure probabilities when the applied pressure $P = P_d$ are shown in Table 5.3. The failure probabilities when the applied pressure was $5.4P_d$ are shown in Table 5.4.

Although the advanced FORM result at the design pressure has slight disagreement with the Monte Carlo value of P_f, it is correct to an order of magnitude and therefore serves as a useful estimator for P_f. In addition, the percentage error of the FORM is reduced at higher values of P_f (for example, at $P = 5.4P_d$ the probability of failure computed by FORM is 0.51 and the value computed by Monte Carlo is 0.49), and therefore for most of the fragility curve the FORM gives a reasonably accurate approximation.

Method	P_f	Standard Deviation of P_f
Advanced FORM	2.7×10^{-8}	Not applicable
Line Sampling	2.6×10^{-8}	7×10^{-9}
Subset Simulation	2.7×10^{-8}	2.7×10^{-9}
Importance Sampling	6.7×10^{-8}	1.8×10^{-9}

Table 5.3: Probability of failure at applied pressure P_d computed by Advanced FORM, Subset Simulation and Importance Sampling. 10^6 samples were used in the Importance Sampling simulation. A maximum of 40 failure thresholds were used for the Subset Simulation, with an intermediate failure probability threshold of 0.5, and 5000 initial samples were used. Line Sampling was performed using 100 lines, with 6 model evaluations on each line.

We would like to know which uncertainties make important contributions to our calculated measure of uncertainty, which in this case is the uncertainty in P_f. The uncertainty in P_f is caused by uncertainty in coefficients of variation of input parameters to the advanced FORM analysis. The Sobol indices for the sensitivity of P_f, calculated by the advanced FORM method, with respect to the coefficients of variation of each parameter in Table 5.2 were calculated using OpenCossan [130]. A uniform distribution between 0 and 1 was applied for the coefficients of variation of the input parameters to the advanced FORM, i.e. any value for the coefficients of variation was equally likely. This is a useful assumption, as it allows us to study the effect of an arbitrary variation in this parameter. The mean values for the parameters were taken from Table 5.2.

The sensitivity analysis was then repeated with an applied pressure of $5.4P_d$, chosen for the strength to design load ratio calculated in the previous section in order to make $P_f = 0.5$, and an applied pressure of $5P_d$, chosen to increase the pressure beyond the design load whilst maintaining $P_f < 0.5$. At increased pressures the variance in the Sobol and total sensitivity indices computed by Monte Carlo simulation was impracticably high and so it was necessary to compute the Sobol indices using the Fourier Amplitude Sensitivity Testing (FAST) method [170] and the upper bound of the Total Sensitivity indices using Patelli's method of integrating the local sensitivities [131], both of which have been implemented in OpenCossan. This allowed the calculation to be completed in a shorter time as fewer samples were required.

The calculated Sobol indices and total sensitivity indices for applied pressure equal to P_d are shown in a bar plot in Figure 5.3. It is clear that the biggest contributors to uncertainty in the output are the coefficients of variation of A_t and F_t. The bar plots show

Method	P_f	Standard Deviation of P_f
Advanced FORM	0.507	Not applicable
Monte Carlo	0.489	0.005

Table 5.4: Probability of failure at applied pressure $5.4P_d$ computed by Advanced FORM and Monte Carlo Simulation. 10^4 samples were used in the Monte Carlo simulation.

error bars to represent the uncertainty in the Monte Carlo estimators for the indices.

Figure 5.4 and Figure 5.5 show the effect of varying A_t and F_t separately, whilst keeping the other variables fixed at their values from Table 5.2. There is a sharp increase in failure probability when the coefficient of variation is larger than 0.2. Further analysis shows that the location for this knee in the graph depends upon the value of the other parameters, i.e. if the other coefficients of variation are set as 0.3 then the location of the knee changes to 0.3. Figure 5.6 shows the effect of varying both of these parameters simultaneously.

The calculated Sobol indices and total sensitivity indices upper bounds for applied pressure equal to $5.4P_d$ are shown in a bar plot in Figure 5.7. It is clear that the biggest contributors to uncertainty in the output are the coefficients of variation of R and P, followed by F_t and A_t. Figure 5.8 shows the effect of varying P and R simultaneously when the applied pressure is equal to $5.4P_d$, whilst keeping the other variables fixed at their values from Table 5.2.

The calculated Sobol indices and total sensitivity indices upper bounds for are shown in a bar plot in Figure 5.9. Again, it is clear that the biggest contributors to uncertainty in the output are the coefficients of variation of R and P, followed by F_t and A_t. Figure 5.10 shows the effect of varying P and R simultaneously when $5P_d$, whilst keeping the other variables fixed at their values from Table 5.2.

5.4 Discussion

The results show a large variability of the failure probability at the design pressure for changing coefficients of variation of A_t and F_t, and this could possibly be explained by the large mean value of these variables. At increased pressures, it is clear that the coefficients of variation of P and R play a greater role in the variability of P_f.

It is interesting to note that the variability of P_f is greatly decreased when the applied pressure is $5.4P_d$, implying that the choice of CoV is unimportant when $P_f = 0.5$. Intuitively, it is clear that if a distribution is centred on the edge of the failure region (i.e. on the limit

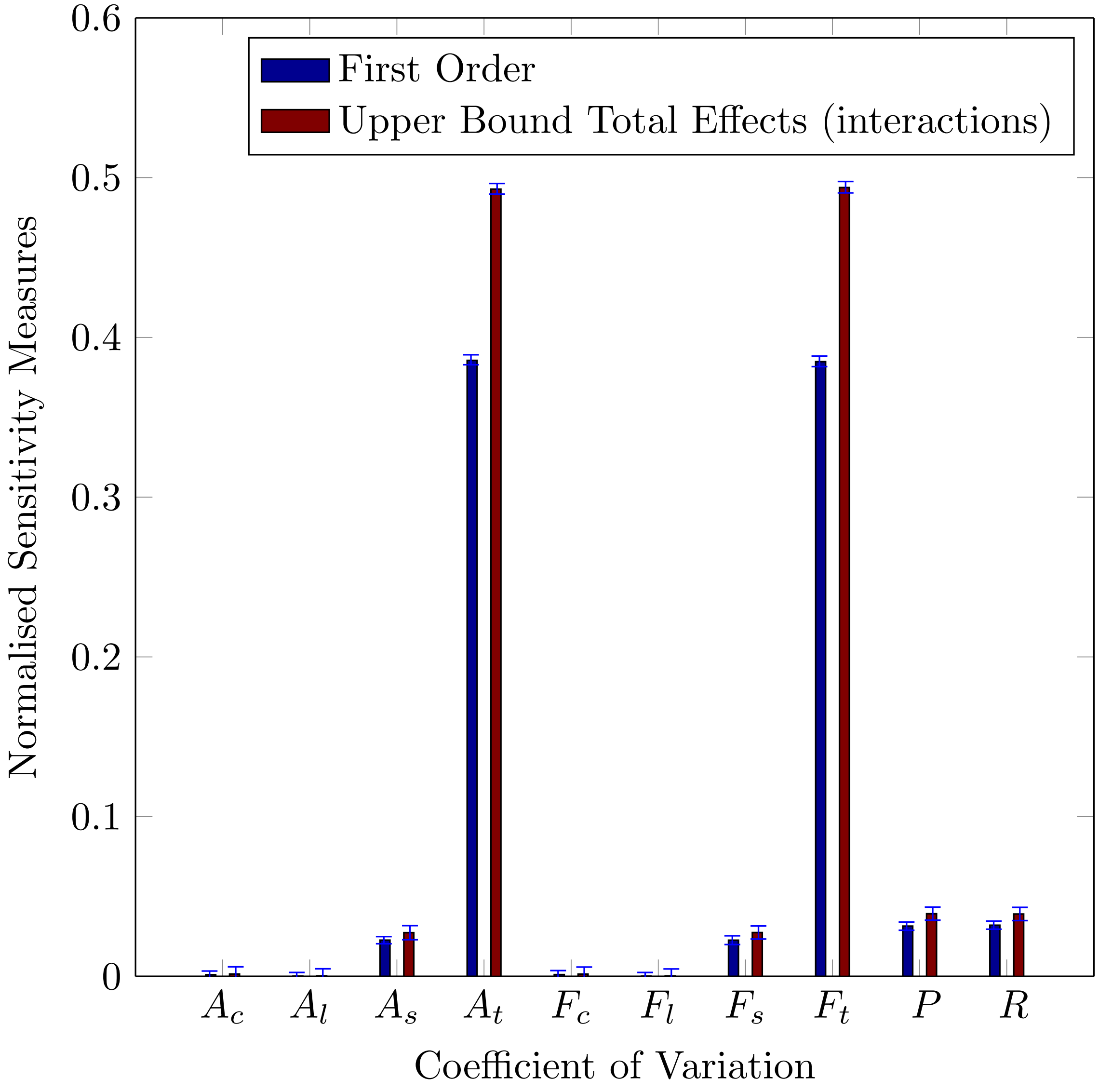

Figure 5.3: Plot of Sobol indices and total sensitivity indices for uncertain coefficient of variation for all input parameters to advanced FORM when the applied pressure is equal to the design pressure, P_d. The error bars represent one standard deviation.

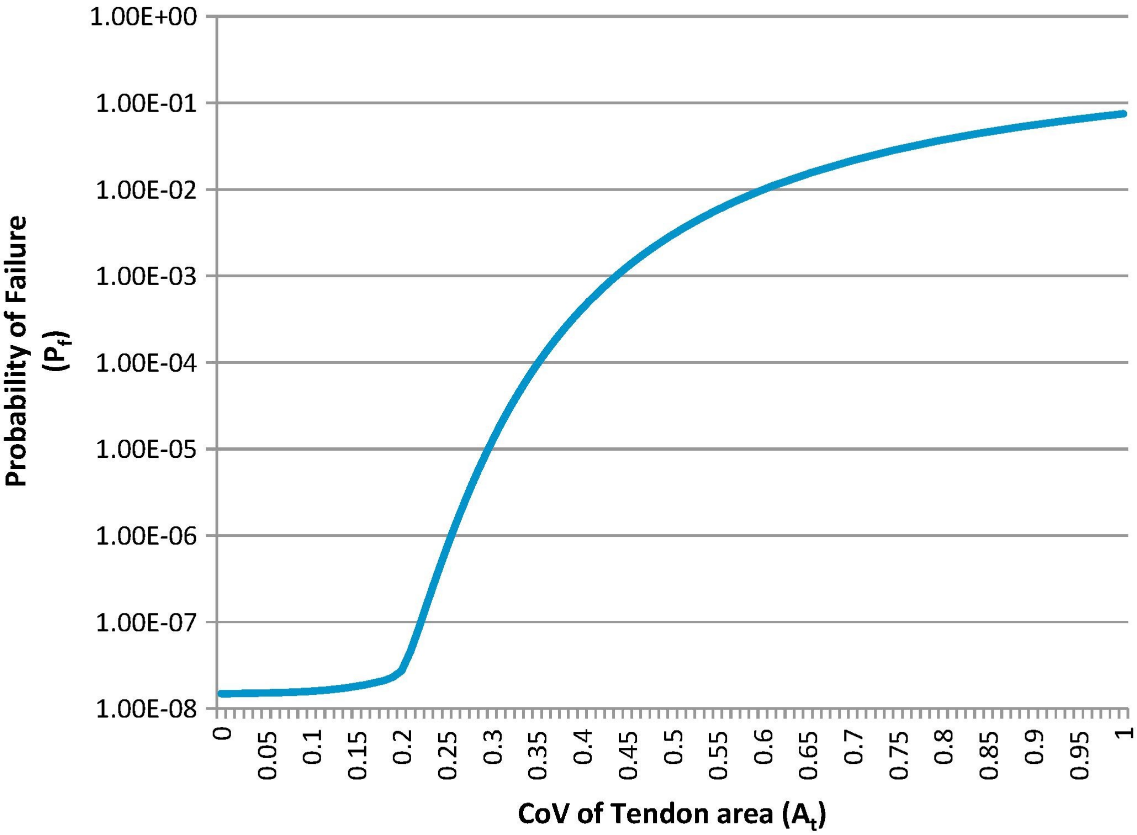

Figure 5.4: Plot of failure probability at applied pressure equal to the design pressure for varying coefficient of variation of tendon area, A_t, while keeping other variables fixed.

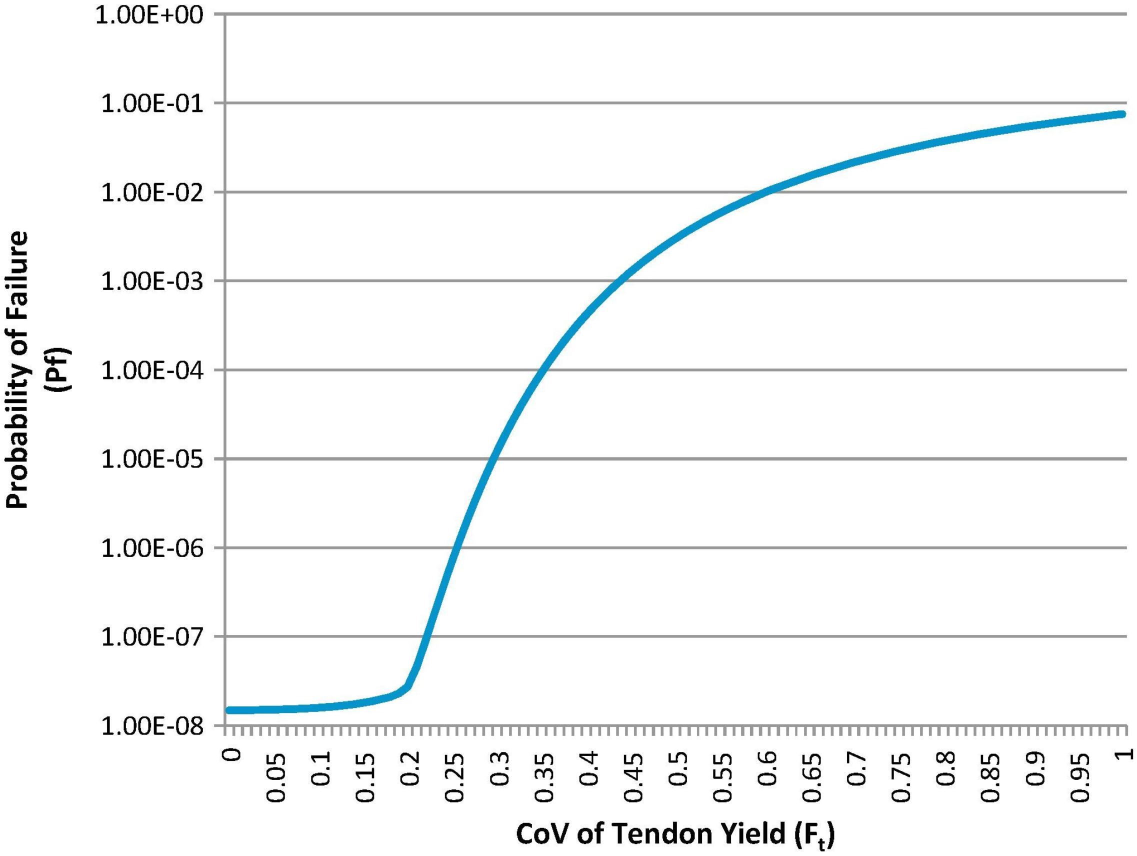

Figure 5.5: Plot of failure probability at applied pressure equal to the design pressure for varying coefficient of variation of tendon yield, F_t, while keeping other variables fixed.

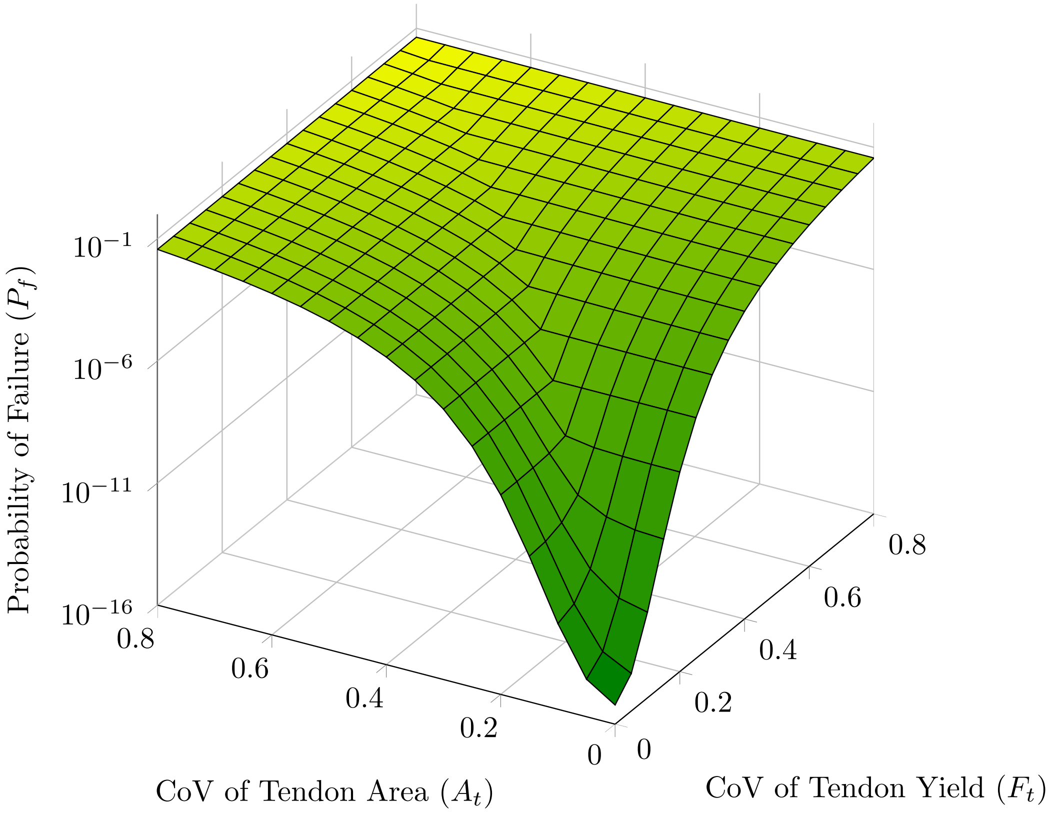

Figure 5.6: Plot of failure probability at applied pressure equal to the design pressure for varying coefficient of variation of tendon yield, F_t, and tendon area, A_t, while keeping other variables fixed

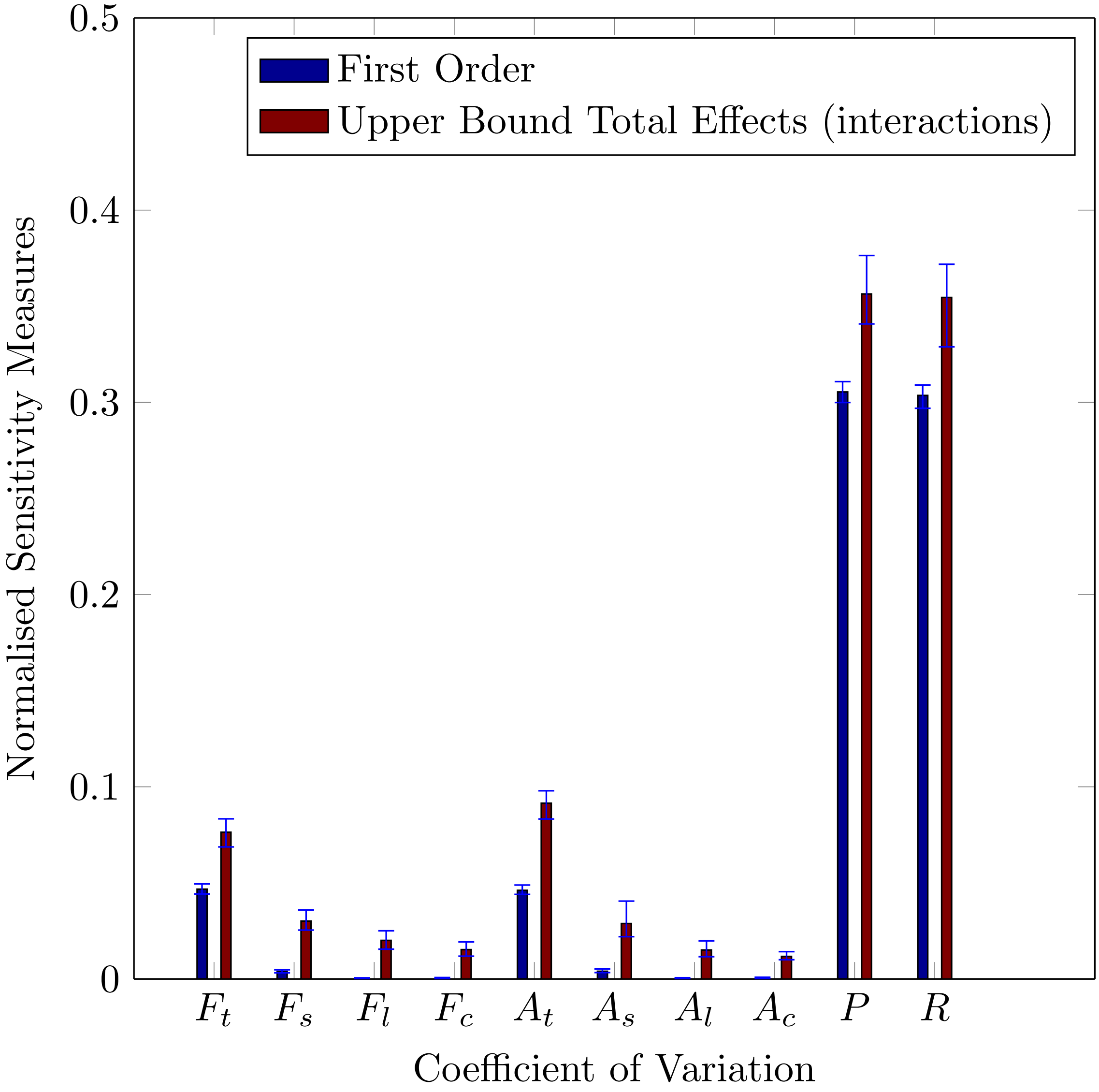

Figure 5.7: Plot of Sobol indices and total sensitivity indices (upper bound) for uncertain coefficient of variation for input parameters to advanced FORM at applied pressure equal to $5.4P_d$. In this figure the error bars represent the 5% - 95% confidence interval.

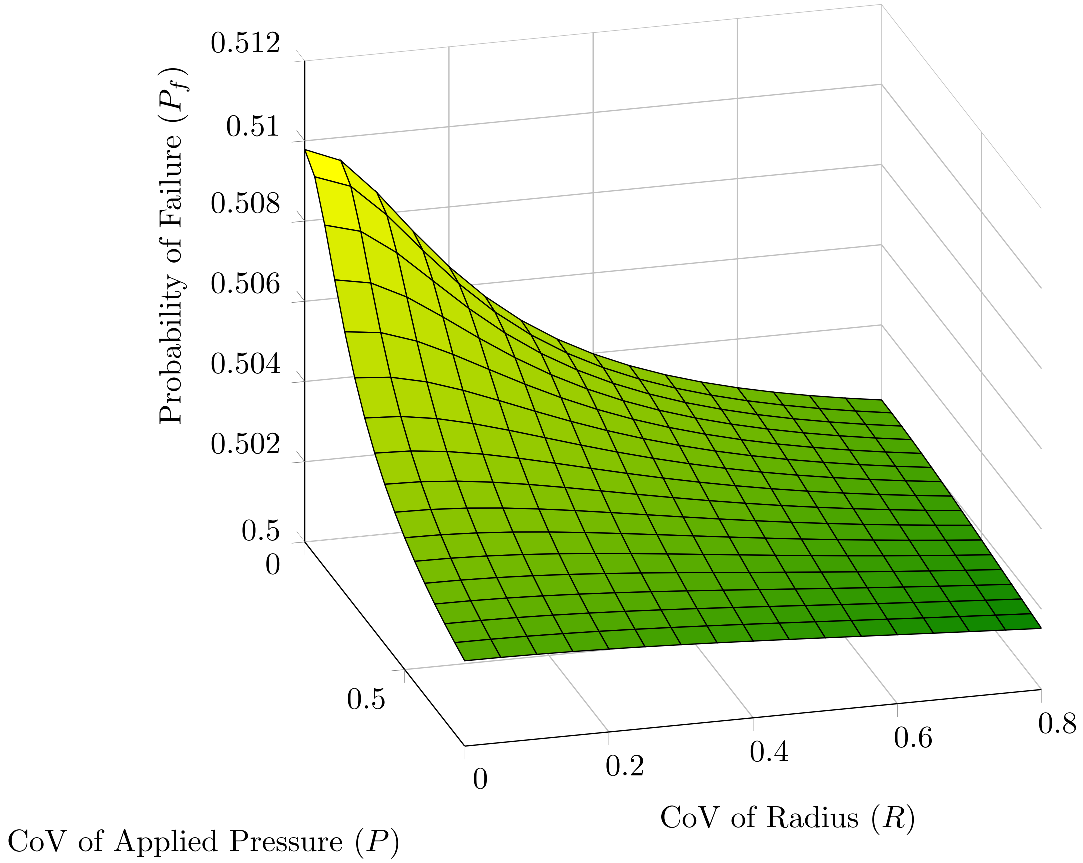

Figure 5.8: Plot of failure probability at applied pressure equal to $5.4P_d$ for varying coefficient of variation of applied pressure, P, and radius, R, while keeping other variables fixed.

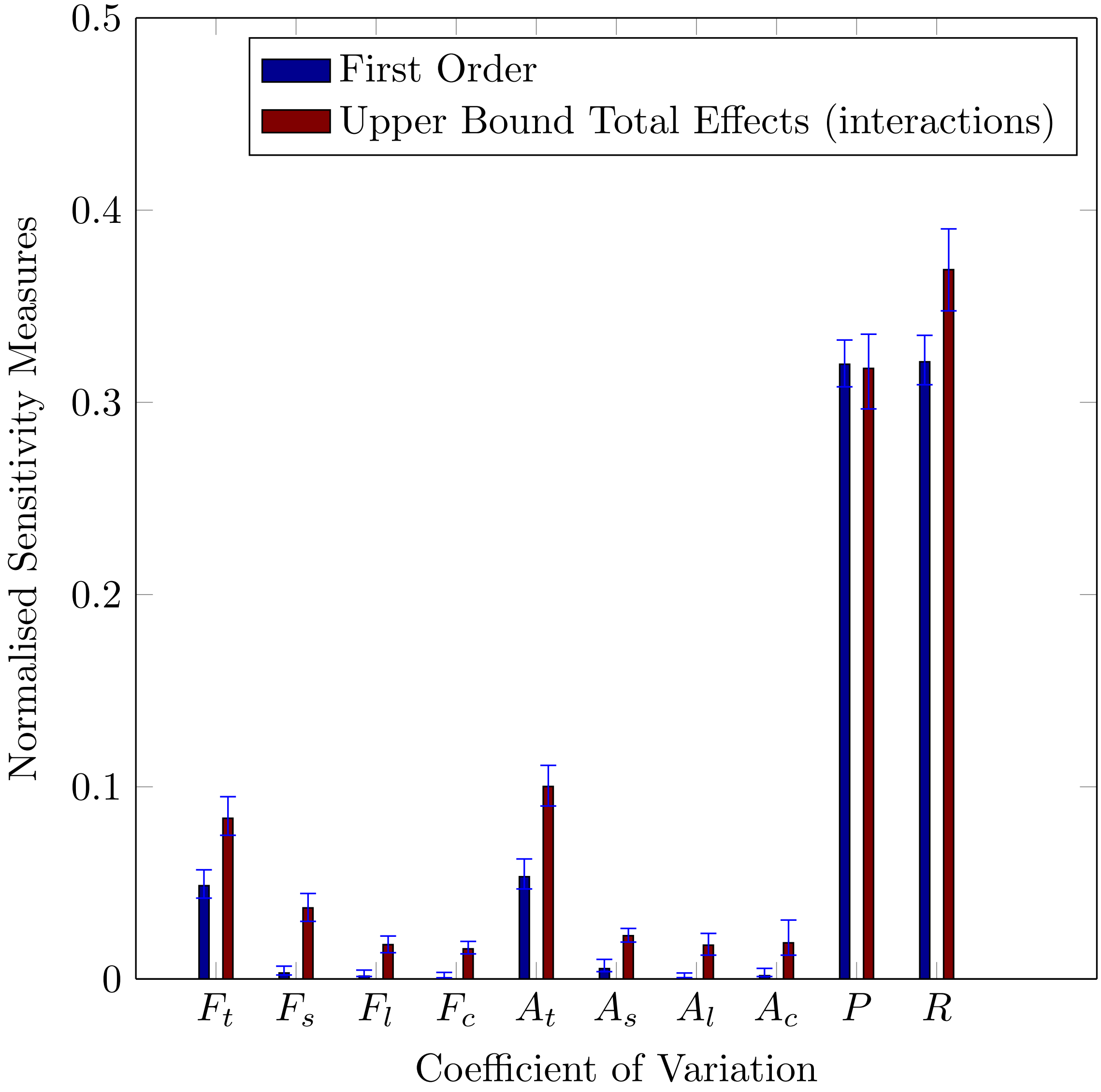

Figure 5.9: Plot of Sobol indices and total sensitivity indices (upper bound) for uncertain coefficient of variation for input parameters to advanced FORM at applied pressure equal to $5P_d$. In this figure the error bars represent the 5% - 95% confidence interval.

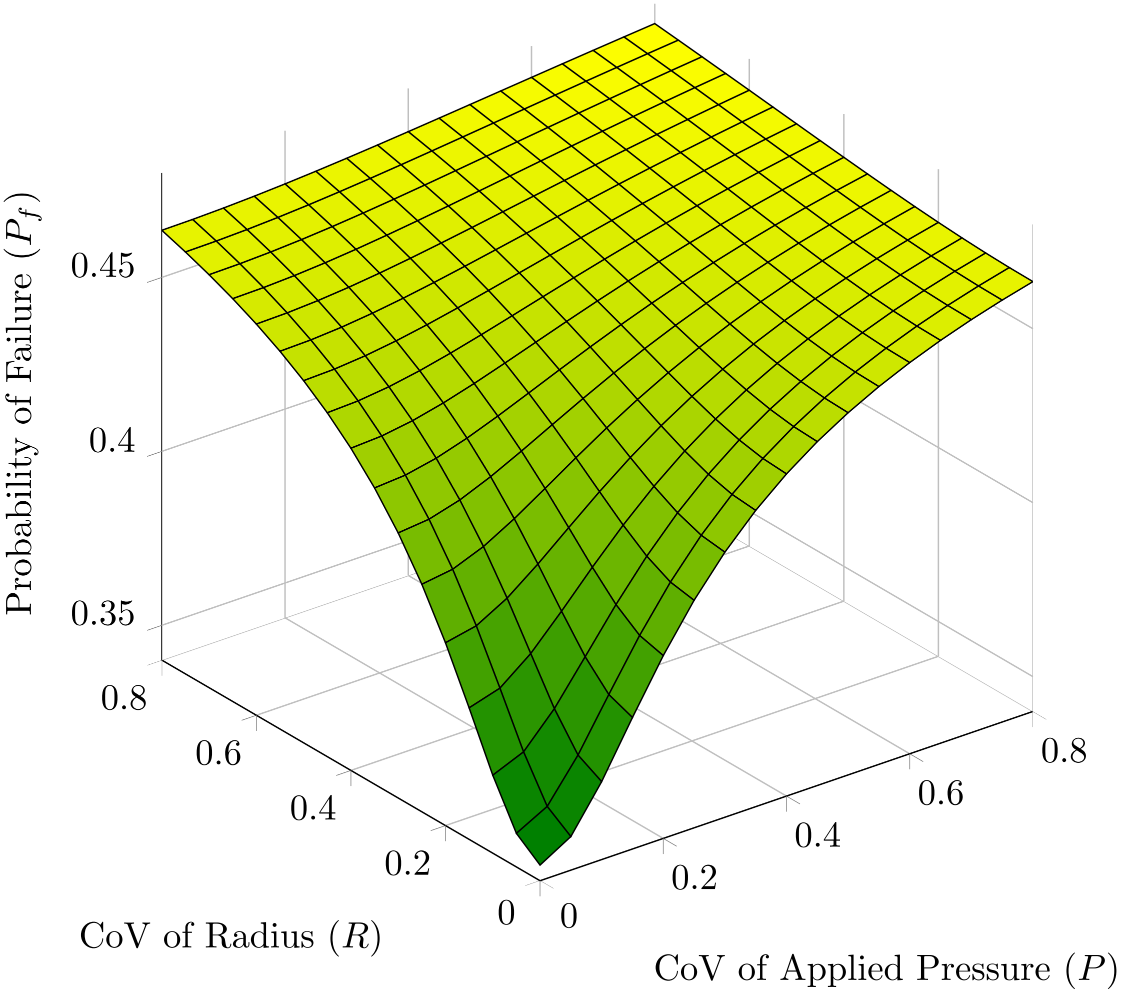

Figure 5.10: Plot of failure probability at applied pressure equal to $5P_d$ for varying coefficient of variation of applied pressure, P, and radius, R, while keeping other variables fixed.

state function) then changing the CoV of the input variables should not significantly move the probability density from the safe region into the failure region.

There is significant variability of P_f when the applied pressure is $5P_d$, however the failure probability appears to plateau when the coefficients of variation of P and R are above approximately 0.5. This implies that if there is no data to determine of the CoV, then a larger CoV would be a conservative choice for this pressure. In this context a conservative choice is one which gives an overestimate of P_f. An overestimate is preferable to an underestimate, because implying a structure is safer than it actually is could have severe consequences. However, we also wish for our estimates to be as close as possible to the true value of P_f, as large overestimates can cause unnecessary over engineering which is undesirable as this can lead to increased costs. The size of coefficient for which this plateau takes place is dependent on mean applied pressure, and this should be considered when attempting to find a conservative value of the coefficients of variation. Moreover, for applied pressures above the strength, Figure 5.10 shows that choosing a lower value of the coefficients of variation would be conservative in this case.

Our analysis justifies the choice of coefficients of variation chosen in this work, as the values given in Spencer et al. [161] and Sundararajan [168] are less than those chosen here, and hence the assumptions for these parameters in this chapter can be considered conservative for applied pressures below the strength.

5.5 Chapter summary

In this chapter, the structural reliability of concrete containments under distributional uncertainty was analysed. Previous studies established the ultimate structural collapse mode of the containments under internal pressure loading, indicating that the failure takes place in the general field of the containment wall, around mid-height and away from any major structural discontinuities. Firstly, the first order reliability method (FORM) was applied to predict probability of failure of the containment for this mode of failure. Then sensitivity analysis was applied to determine the dependence of the variability of the probability of failure on the coefficients of variation of structural variables.

It has been shown that there is a strong dependence of the probability of failure of a concrete containment computed by advanced FORM on the coefficients of variation of the rebar yield and rebar area at the design pressure. The coefficients of variation of the pressure and radius are also important parameters, especially in the centre of the fragility curve when the applied pressure is increased. The variability of the probability of failure is decreased at this applied pressure; however it is still important to apply conservatism in scenarios where we lack knowledge of the true value of these parameters.

In order to accurately describe the epistemic uncertainty in distribution parameters, particularly in the tails of the fragility curve, it may be necessary to construct a probability box by defining the CoV as an interval [60]. This approach can be understood as the engineer testing many different values for the CoV of each variable, and choosing the most and least conservative values to give an interval for P_f (in practice the engineer would use a sophisticated optimisation algorithm coupled with conventional reliability analysis to perform the calculation). In the following chapters, efficient methods for performing this computation are described.

Chapter 6

Analytic Imprecise Probabilistic Safety Analysis

6.1 Introduction

Probabilistic safety analysis (PSA) was first introduced in the 1970s as a means of establishing the probability of a certain amount of radiation release to the environment from a nuclear structure. It is perceived to address many of the weaknesses of deterministic analysis [118]. For example, deterministic analysis relies heavily on engineering conservatism which could be difficult to quantify in practice. In addition, it is not always clear what the most conservative value for a particular parameter is when performing a black box analysis.

In recent years, techniques from the area of imprecise probability have been increasingly applied to probabilistic safety analysis studies in academic literature [93] [13], because they offer a natural framework to model epistemic uncertainty. Epistemic uncertainty is particularly important in the nuclear industry where there is often a lack of sufficient data to completely model relevant phenomena. However, the proposed models usually require sophisticated simulation techniques, as discussed in Chapter 4. In Le Duy et al. [110] recommendations are made for how available data can be used to define probability boxes for risk assessment. In the United States, the nuclear regulator [28] refers to Kennedy et al. [98] who provide many analytic relationships to establish the fragility of a containment with a conventional probabilistic treatment. The effect of epistemic uncertainty in probabilistic safety analysis with conventional probability was considered in [143, 166].

In (conventional) structural probabilistic safety analysis, often the relations used are simple analytic expressions which, in contrast to the methods based on imprecise probability, allow the failure probability of the system to be computed with no Monte Carlo simulation at all. This offers two significant advantages. Firstly, the computational time required to complete the calculations is greatly reduced, which allows projects to be completed on shorter timescales and less money to be spent on high performance computing. Secondly, the time of engineers is saved as they are not required to spend large amounts of time programming Monte Carlo simulations, which reduces expenditure for their employer, and consequently benefits the industry as a whole.

In this chapter, we will propose imprecise probabilistic analogues to many of the probabilistic formulae proposed in Kennedy et al. [98] which have become standard expressions used in probabilistic safety analysis. In this way, we hope to unite the conventional literature which is applied to probabilistic safety analysis in industry with relatively recent developments in imprecise probability. The analysis will make extensive use of the probability boxes introduced in probability bounds theory. We will demonstrate how to establish the fragility curve of a system when components are connected in parallel or series, and when the failures of the components may have unknown dependencies. We will demonstrate how to establish a probability box fragility curve when the product of random variables must be considered. Then, we will also demonstrate how this can be used to calculate the failure probability when there is additional imprecision in the load distribution. We will also consider the implications of the imprecise first order reliability method (FORM), and show how we can analytically obtain results from a simplified calculation when the exact reliability index is difficult to obtain. All of the above are particularly useful when combined with an event tree to e.g. yield the expected radiation release to the environment or to calculate the reliability of complex plant.

The merit of this approach is that the entire fragility curve can be constructed by one analyst using conventional spreadsheet packages, without the requirement to use complicated simulation techniques which would require large amounts of time spent programming by the analyst. Therefore the benefits of traditional probabilistic safety analysis approaches are retained whilst also obtaining the advantages of using probability bounds theory.

6.2 Probabilistic safety analysis

Probabilistic Safety Analysis is broken down into three levels. Level 1 probabilistic safety analysis studies the reactor and determines accident sequences which are likely to result in a release from the reactor pressure vessel. Level 2 considers the containment structure, and how likely this is to fail in an accident. This is done by creating a fragility curve for the containment, which quantifies the failure probability at a particular load. Level 3 probabilistic safety analysis combines the information produced by level 1 and level 2 probabilistic safety analysis to provide the probability of radiation release to the environment [43].

In probabilistic safety analysis level 2 the main goal is to establish the fragility curve of a (nuclear) structure [138]. In seismic hazard analysis the fragility curve expresses the failure probability of the structure as a function of the peak ground acceleration. This can then be used to conduct safety analysis once the conditions inside the reactor (the 'source term') and the external conditions are known [168].

The fragility of a system is its probability of failure conditioned on a particular load. Therefore, in the context of this section, bounds on failure probabilities may be taken as bounds on fragilities. For a system, S, of components, c_i, connected in series (i.e. the system will fail if one component fails) the fragility of the system, $f(s|a)$, at a damage measure a (i.e. the peak ground acceleration) is given by

$$f(s|a) = 1 - \prod_{c_i \subset S} [1 - f(c_i|a)], \tag{6.1}$$

when the fragilities of the individual components are independently distributed [98].

If the dependence is not known then Boole's inequality can be used to calculate an upper bound on the probability that at least one event from a set of events occurs, i.e. the probability that a series system fails, when the dependence between different events is unknown. The Fréchet inequalities are similar upper and lower bounds that apply to the probability of the union and intersection of events when no information is available about the dependence of events [152].

Boole's inequality is equivalent to the upper bound given by the Fréchet inequalities for the union of n events:

$$\max(P(A_1), ..., P(A_n)) \leq P\left(\bigcup_{i=1}^{n} A_i\right) \leq \min(1, P(A_1) + ... + P(A_n)). \tag{6.2}$$

The other Fréchet inequality (which applies for components connected in parallel) being

$$\max(0, P(A_1) + ... + P(A_n) - (n-1)) \leq P\left(\bigcap_{i=1}^{n} A_i\right) \leq \min(P(A_1), ..., P(A_n)). \quad (6.3)$$

Note that both Boole's inequality and the Fréchet inequalities are conservative bounds which should be used when the dependence between failure events is unknown.

If the fragilities of the components are independently distributed and the components are connected in parallel (i.e. the system has redundancy and fails if every component fails) then the system's fragility is given by

$$f(s|a) = \prod_{c_i \subset S} f(c_i|a). \quad (6.4)$$

The rare event approximation states that the value of $f(s|a)$ given by Eqn. 6.1 is usually approximately equal to the value given by Boole's inequality for the small probabilities relevant to this type of analysis [42], i.e. $1 - \prod_{c_i \subset S}[1 - f(c_i|a)] \leq \sum_{c_i \subset S}[f(c_i|a)]$ and $\sum_{c_i \subset S}[f(c_i|a)] \approx 1 - \prod_{c_i \subset S}[1 - f(c_i|a)]$ for small $f(c_i|a)$. This is useful because for systems with many components, applying Boole's inequality requires evaluating fewer terms than Eqn. 6.1. Similarly, combining fragilities using Eqn. 6.3 provides an upper bound to the value of $f(s|a)$ given by Eqn. 6.4 [98]. These formulae can also be applied to connected systems which form super systems, in which case the unknown dependence versions on the equations should be used [98].

In Probabilistic Safety Analysis $f(c_i|a)$ is usually modelled as a log normally distributed random variable, because the physical quantities being modelled must be greater than zero, i.e.

$$f(c_i|a) = \phi\left(\frac{\log\left(\frac{a}{\beta_i}\right)}{\sigma_i}\right), \quad (6.5)$$

where β_i represents the median failure value and σ_i is the logarithmic standard deviation of component c_i, and ϕ is the cumulative distribution function (CDF) of a standard normal variable. Typically in Probabilistic Safety Analysis aleatory uncertainty can be distinguished from epistemic uncertainty by modelling the β for any particular component as a lognormally distributed random variable with parameters β_e and σ_e. Hence the outer distribution (i.e. Eqn. 6.5, with logarithmic standard deviation σ_a) will describe aleatory uncertainty, and epistemic uncertainty is modelled by the nested distribution (i.e. the inner distribution, the CDF over β, with parameters β_e and σ_e).

In order to allow this model to be used for computation, typically the mean distribution is obtained (more widely known as the 'composite' distribution), which is also log-normally distributed. This is an averaged distribution obtained by combining the aleatory uncertainty (i.e. σ_a from the outer distribution) and the epistemic uncertainty (our uncertainty in the distribution parameters, σ_e) [99]. For the composite distribution, the logarithmic standard deviation, σ_c, is the euclidean norm of the two lognormal logarithmic standard deviations, i.e. $\sigma_c = \sqrt{\sigma_a^2 + \sigma_e^2}$, and the median is simply the median of the inner (epistemic) distribution, $\beta_c = \beta_e$ (a detailed derivation is provided in Kaplan et al. [91]). This distribution is assumed to be conservative, since it approaches the asymptotic values in the tails of the distributions described by the extrema of the epistemic distribution [98]. However, in many cases assuming that the epistemic uncertainty is log-normally distributed may be an overly strong assumption.

Figure 6.1 shows an example of a composite distribution compared to the median fragility curve and the 5th and 95th percentiles of the epistemic uncertainty. As discussed, the mean curve approaches the extreme outer distributions' tails (obtained by taking β from the 5th and 95th percentiles of the nested epistemic distribution and $\sigma = \sigma_a$). Clearly, the median curve could not be used for this purpose as it does not adequately describe the range of our belief in the peak ground acceleration.

6.3 Probability bounds analysis

6.3.1 Fragility curve

Let us consider the fragility distribution for a general component given by Eqn. 6.1. Instead of considering β_i as a random variable and finding the composite distribution we will instead consider uncertainty in β_i and σ_i as intervals. This enables the random variables to be converted into probability boxes, which is useful for several reasons. Firstly, we do not need to assume a distribution for our epistemic uncertainty, which permits a robust analysis even with limited data. Secondly, instead of having to find the composite distribution we can simply find the envelope of our distributions. Note that uniform distributions are conceptually different from interval incertitude, since a uniform distribution specifies that

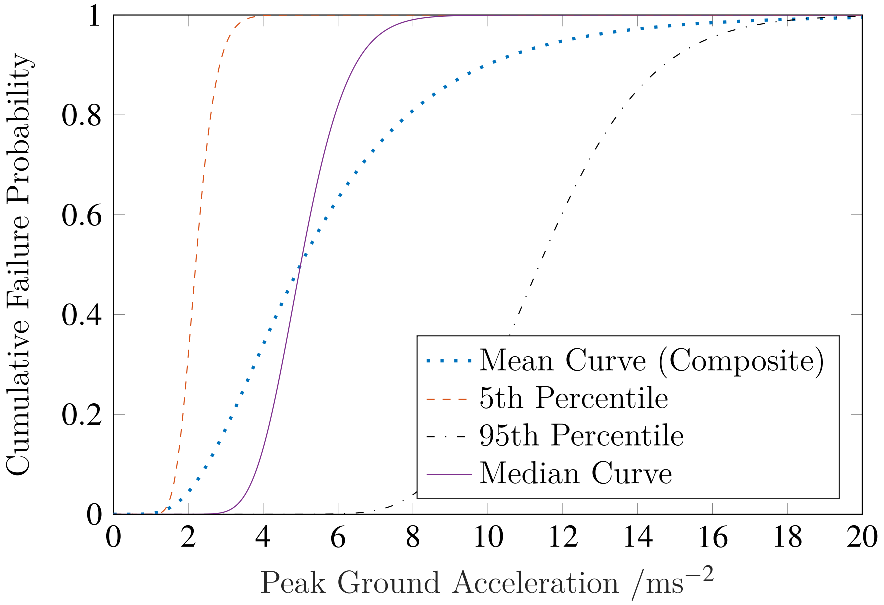

Figure 6.1: The composite curve compared to the median curve ($\beta = \beta_e$ and $\sigma = \sigma_a$), and the curves with 5th and 95th percentiles of β and $\sigma = \sigma_a$. In the example $\sigma_a = 0.2$, $\beta_e = 5\text{ms}^{-2}$ and $\sigma_e = 0.5$.

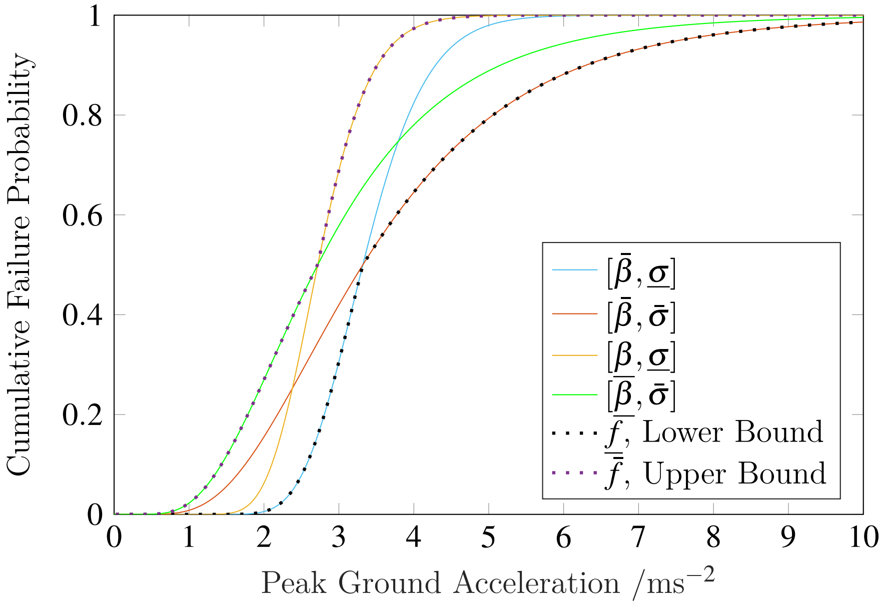

Figure 6.2: A comparison of extreme fragility curves enclosed within the fragility probability box. The parameters for the plotted probability box were $\underline{\mu} = \log \underline{\beta} = 1\text{ms}^{-2}$, $\bar{\mu} = \log \bar{\beta} = 1.2\text{ms}^{-2}$, $\underline{\sigma} = 0.2$ and $\bar{\sigma} = 0.5$.

each value in the support of the distribution is equally likely, whereas an interval describes lack of knowledge in a set-like manner, without implications for the likelihood of different elements within the set.

If $\beta_i \in [\underline{\beta_i}, \overline{\beta_i}]$ and $\sigma_i \in [\underline{\sigma_i}, \overline{\sigma_i}]$ then the distributional probability box can be converted to a distribution-free probability box where the upper bound of the fragility is given by

$$\overline{f(c_i|a)} = \phi\left(\frac{\log\left(\frac{a}{\underline{\beta_i}}\right) - \left|\log\left(\frac{a}{\underline{\beta_i}}\right)\right|}{2\overline{\sigma_i}} + \frac{\log\left(\frac{a}{\underline{\beta_i}}\right) + \left|\log\left(\frac{a}{\underline{\beta_i}}\right)\right|}{2\underline{\sigma_i}} \right), \tag{6.6}$$

and the lower bound of the fragility is given by

$$\underline{f(c_i|a)} = \phi\left(\frac{\log\left(\frac{a}{\overline{\beta_i}}\right) + \left|\log\left(\frac{a}{\overline{\beta_i}}\right)\right|}{2\overline{\sigma_i}} + \frac{\log\left(\frac{a}{\overline{\beta_i}}\right) - \left|\log\left(\frac{a}{\overline{\beta_i}}\right)\right|}{2\underline{\sigma_i}} \right), \tag{6.7}$$

where the $|\cdot|$ operator represents the absolute value of a quantity. These bounds are shown in Figure 6.2. This follows from noting that ϕ is a monotonic function of its arguments, so finding the maxima and minima of Eqn. 6.5 can be reduced to finding the maxima and minima of $\frac{\log\frac{a}{\beta_i}}{\sigma_i}$ when $\beta_i \in [\underline{\beta_i}, \bar{\beta}_i]$ and $\sigma_i \in [\underline{\sigma_i}, \bar{\sigma}_i]$. Then note that $\log\frac{a}{\bar{\beta}_i} < \log\frac{a}{\beta}_i < \log\frac{a}{\underline{\beta_i}}$. The upper bound is found by noting that if $0 < \log\frac{a}{\underline{\beta_i}}$ then $\frac{\log\frac{a}{\beta}_i}{\sigma_i} < \frac{\log\frac{a}{\underline{\beta_i}}}{\underline{\sigma_i}}$ and if $0 > \log\frac{a}{\underline{\beta_i}}$ then $\frac{\log\frac{a}{\beta}_i}{\sigma_i} < \frac{\log\frac{a}{\underline{\beta_i}}}{\bar{\sigma}_i}$. The lower bound is found by noting that if $0 < \log\frac{a}{\bar{\beta}_i}$ then $\frac{\log\frac{a}{\beta}_i}{\sigma_i} > \frac{\log\frac{a}{\bar{\beta}_i}}{\bar{\sigma}_i}$ and if $0 > \log\frac{a}{\bar{\beta}_i}$ then $\frac{\log\frac{a}{\beta}_i}{\sigma_i} > \frac{\log\frac{a}{\bar{\beta}_i}}{\underline{\sigma_i}}$. Finally, note that it is trivial to construct a function which takes a different value above and below zero, e.g. $\frac{f_1(x)-|f_1(x)|}{c_1} + \frac{f_1(x)+|f_1(x)|}{c_2}$ is equal to $\frac{2f_1(x)}{c_2}$ above zero and $\frac{2f_1(x)}{c_1}$ below zero.

In general, converting distributional probability boxes to distribution-free probability boxes results in loss of information [5]. However, in this case Eqn. 6.6 and Eqn. 6.7 are a result of taking the natural extension of Eqn. 6.5, and therefore the values obtained will be the tightest bounds possible, so in the specific case of Eqn. 6.6 and Eqn. 6.7 there is no consequence to making the conversion. The other results in this section provide the tightest possible bound in the case of unknown dependence, since we simply apply a Fréchet inequality. Note that the results in subsequent sections do not make use of the conversion used in this section, in order to avoid the potential information loss.

For systems containing components in series or parallel, when the component failures are known to be independent, the fragility can be calculated by using Eqn. 6.1 and Eqn. 6.4, respectively. Alternatively, if the failure dependence is unknown we can use the relevant Fréchet inequality, Eqns. 6.2 and 6.3, to yield the fragility. Alternatively, in the case of unknown failure dependence, the rare event approximation (described in Section 6.2) can be used to justify the application of Eqn. 6.1 and Eqn. 6.4 which will be accurate in the tails of the distributions (i.e. for rare events).

Therefore, using the natural interval extension of Eqn. 6.2 with Eqn. 6.6 and Eqn. 6.7 it can be shown that, for components in series, the probability of failure at a particular ground motion, a, with certainty falls in the interval given by

$$f(s|a) \in \left[\max_i \left[\underline{f(c_i|a)} \right], \min\left(1, \sum_{i=1}^{n} \left[\overline{f(c_i|a)} \right] \right) \right], \tag{6.8}$$

i.e.

$$\begin{aligned} f(s|a) \in & \left[\max_i \left[\phi \left(\frac{\log\left(\frac{a}{\overline{\beta_i}}\right) + \left|\log\left(\frac{a}{\overline{\beta_i}}\right)\right|}{2\overline{\sigma_i}} + \frac{\log\left(\frac{a}{\overline{\beta_i}}\right) - \left|\log\left(\frac{a}{\overline{\beta_i}}\right)\right|}{2\underline{\sigma_i}} \right) \right], \right. \\ & \left. \min\left(1, \sum_{i=1}^{n} \left[\phi \left(\frac{\log\left(\frac{a}{\underline{\beta_i}}\right) - \left|\log\left(\frac{a}{\underline{\beta_i}}\right)\right|}{2\overline{\sigma_i}} + \frac{\log\left(\frac{a}{\underline{\beta_i}}\right) + \left|\log\left(\frac{a}{\underline{\beta_i}}\right)\right|}{2\underline{\sigma_i}} \right) \right] \right) \right]. \end{aligned} \tag{6.9}$$

6.3.2 Product of log-normally distributed random variables

Often the fragility curve for a component must be established by considering the product of a number of random variables with lognormal distributions. If this is the case then the probability bounds analysis approach can be extended to allow us to find the relevant fragility curve. To demonstrate, consider a general random variable d which is given by the product of other random variables, i.e.

$$d = q \frac{a^r b^s}{c^t}, \tag{6.10}$$

where a, b and c are lognormal random variables and q, r, s and t are constants. It is clear that d will be lognormally distributed with median $\beta_d = q\frac{\beta_a^r \beta_b^s}{\beta_c^t}$, and logarithmic standard deviation $\sigma_d^2 = r^2\sigma_a^2 + s^2\sigma_b^2 + t^2\sigma_c^2$ [98].

In the case of interval imprecision in the distribution parameters of a, b and c we can obtain

$$\overline{\beta_d} = q \cdot \frac{\overline{\beta_a^r} \cdot \overline{\beta_b^s}}{\underline{\beta_c^t}}, \tag{6.11}$$

and

$$\underline{\beta_d} = q \cdot \frac{\underline{\beta_a^r} \cdot \underline{\beta_b^s}}{\overline{\beta_c^t}}, \tag{6.12}$$

by using the endpoint formulae for interval multiplication [121] with knowledge of the support of the distribution parameters. The logarithmic standard deviation can be obtained from

$$\overline{\sigma_d^2} = r^2\overline{\sigma_a^2} + s^2\overline{\sigma_b^2} + t^2\overline{\sigma_c^2}, \tag{6.13}$$

and

$$\underline{\sigma_d^2} = r^2\underline{\sigma_a^2} + s^2\underline{\sigma_b^2} + t^2\underline{\sigma_c^2}, \tag{6.14}$$

by taking the interval extension of the expression stated above for the case of no interval imprecision.

This is principally of use when computing the response factor, F, which can be expressed as the product of a number of response factors applying to different pieces of equipment and processes (for example damping effects or modelling effects), i.e. $F = \prod_i F_i$. The F_i are modelled as lognormal random variables and may have interval imprecision in the median [168].

6.3.3 Failure probability

Consider a system which fails when the load exceeds the strength. For a general load distribution the failure probability is given by

$$P_f = -\int_0^\infty \frac{dH(a)}{da} f(s|a)da, \tag{6.15}$$

where $H(a)$ is the seismic hazard curve (i.e. the probability that the 'load' exceeds a certain value in a particular unit of time, which usually takes the form of the complement of a CDF since it must be monotonically decreasing, and the probability cannot exceed

1) [6]. When $H(a)$ and $f(s|a)$ are both log normally distributed, it is simple to solve Eqn. 6.15 by transforming the integral [92]. However, in general this integral is not solvable analytically and it cannot be solved analytically when the fragility curve is replaced with the distribution-free probability boxes derived in the previous section.

Therefore, to derive bounds on the failure probability of systems subject to distributional probability box loads and fragilities, we will apply Fréchet bounds and interval arithmetic to well known results obtained by solving Eqn. 6.15 for common probability distributions.

For example, consider the case where the probability distribution function of the load, $\frac{dH(a)}{da}$, is log-normally distributed with parameters β_l and σ_l and the fragility, $f(s|a)$, is lognormally distributed with parameters β_i and σ_i. In this case, the failure probability can be evaluated as

$$P_f = \phi\left(-\frac{\log\beta_i - \log\beta_l}{\sqrt{\sigma_i^2 + \sigma_l^2}}\right). \tag{6.16}$$

A plot of distributions used in Eqn. 6.16 with example parameters is shown in Figure 6.3.

To calculate an upper bound on the failure probability for a series system we evaluate the maximum and minimum of Eqn. 6.16 with $\beta_l \in [\underline{\beta_l}, \bar{\beta}_l], \sigma_l \in [\underline{\sigma_l}, \bar{\sigma}_l], \beta_i \in [\underline{\beta_i}, \bar{\beta}_i], \sigma_i \in [\underline{\sigma_i}, \bar{\sigma}_i]$ and Eqn. 6.2. Analogously, for components in parallel a similar result can be obtained from Eqn. 6.3. For simple systems these bounds provide useful analytic quantification of the reliability of the system under epistemic uncertainty. However, for more complex systems the bounds are usually not analytically calculable and hence numerical integration may be necessary (e.g. [156], [135], [59]).

It is likely that there is uncertainty in β_l and σ_l. If this is the case then the analysis can be made robust using an uncertainty quantification approach for the load distribution which is analogous to the approach used for the fragility.

In some works, such as ASCE 43-05 [24], the hazard curve has been modelled as a power law, since this is a good approximation to the Cauchy-Pareto complementary cumulative distribution function [97]. Such an equation takes the form of

$$H(a) = k_1 a^{-K_H}, \tag{6.17}$$

where k_1 and K_H are positive fitted constants. K_H represents the slope of the mean seismic hazard curve when plotted on log-log scale. With a log-normal fragility in the parametrisation used in this chapter, the failure probability for a single component is given

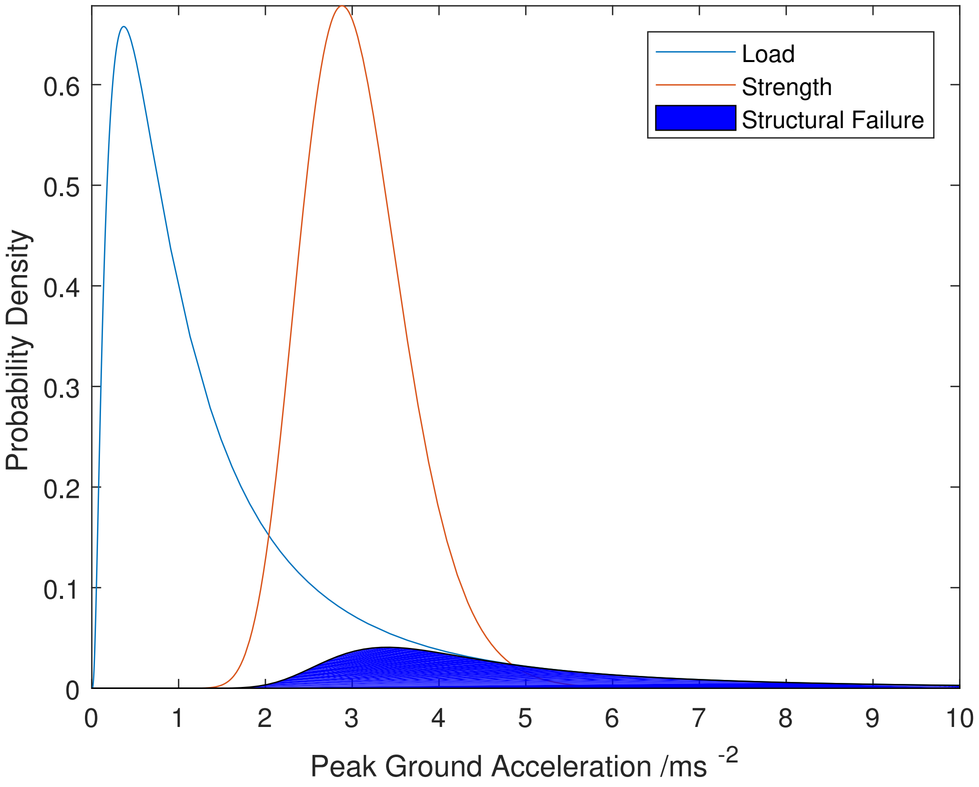

Figure 6.3: Demonstration of failure probability calculation with Eqn. 6.16. The lognormal probability density functions for the stress and strength are shown. The shaded area represents the integrand in Eqn. 6.15, which yields the failure probability $P_f = 0.14$. The example parameter values for the plotted distributions were $\beta_l = 1\text{ms}^{-2}$, $\sigma_l = 1$, $\beta_i = 3\text{ms}^{-2}$ and $\sigma_i = 0.2$.

by

$$P_f = H(\beta_i) \exp \frac{(K_H \sigma_i)^2}{2}. \tag{6.18}$$

A plot of distributions used in Eqn. 6.18 with example parameters is shown in Figure 6.4.

When there is interval imprecision in K_H, k_1, β_i and σ_i we can obtain bounds on the failure probability, and this result can be generalised trivially to the case of a parallel or series system using the formulae given in Section 6.2.

6.3.4 Summary of failure probability expressions

The following list of results can be derived from the previous section:

- Parallel System with unknown dependence; Lognormal load and Strength:

$$\underline{P_f} = \sum_{c_i \subset S} \min \left[\phi \left(-\frac{\log \overline{\beta_i} - \log \overline{\beta_l}}{\sqrt{\overline{\sigma_i^2} + \bar{\sigma}_l^2}} \right), \phi \left(-\frac{\log \overline{\beta_i} - \log \bar{\beta}_l}{\sqrt{\underline{\sigma_i^2} + \bar{\sigma}_l^2}} \right) \right] - (n-1) \tag{6.19a}$$

and

$$\overline{P_f} = \min_{c_i \subset S} \left[\max \left[\phi \left(-\frac{\log \underline{\beta_i} - \log \underline{\beta_l}}{\sqrt{\overline{\sigma}_i^2 + \underline{\sigma_l^2}}} \right), \phi \left(-\frac{\log \underline{\beta_i} - \log \underline{\beta_l}}{\sqrt{\underline{\sigma_i^2} + \underline{\sigma_l^2}}} \right) \right] \right] \tag{6.19b}$$

- Series system with unknown dependence; Lognormal load and Strength:

$$\overline{P_f} = \sum_{c_i \subset S} \max \left[\phi \left(-\frac{\log \underline{\beta_i} - \log \bar{\beta}_l}{\sqrt{\bar{\sigma}_i^2 + \bar{\sigma}_l^2}} \right), \phi \left(-\frac{\log \underline{\beta_i} - \log \bar{\beta}_l}{\sqrt{\underline{\sigma_i}^2 + \underline{\sigma_l}^2}} \right) \right] \tag{6.20a}$$

and

$$\underline{P_f} = \max_{c_i \subset S} \left[\min \left[\phi \left(-\frac{\log \bar{\beta}_i - \log \underline{\beta_l}}{\sqrt{\bar{\sigma}_i^2 + \bar{\sigma}_l^2}} \right), \phi \left(-\frac{\log \bar{\beta}_i - \log \underline{\beta_l}}{\sqrt{\underline{\sigma_i}^2 + \underline{\sigma_l}^2}} \right) \right] \right] \tag{6.20b}$$

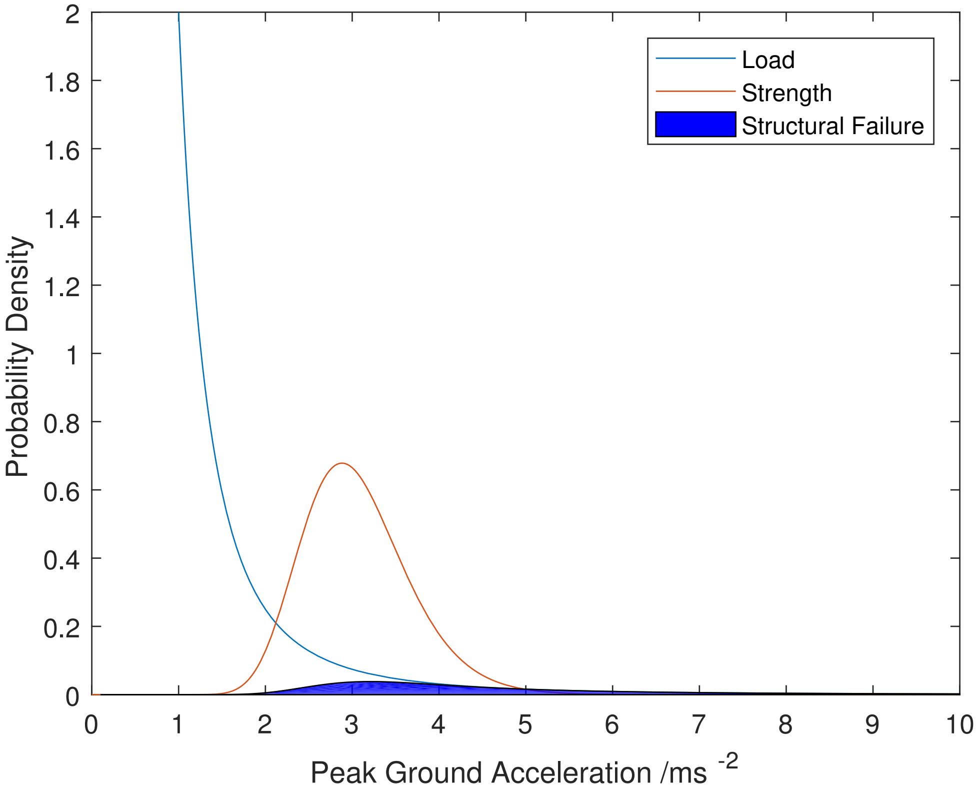

Figure 6.4: Demonstration of failure probability calculation with Eqn. 6.18. The lognormal probability density functions for the stress and strength are shown. The shaded area represents the integrand in Eqn. 6.15, which yields the failure probability $P_f = 0.12$. The example parameter values for the plotted distributions were $K_H = 2$, $k_1 = 1(\mathrm{ms}^{-2})^{K_H}$, $\beta_i = 3\mathrm{ms}^{-2}$ and $\sigma_i = 0.2$.

- Series system with independent components (upper bound also valid for dependent rare events); Log-normal load and strength:

$$\overline{P_f} = 1 - \prod_{c_i \subset S} \left[1 - \max \left[\phi \left(- \frac{\log \underline{\beta_i} - \log \bar{\beta}_l}{\sqrt{\bar{\sigma}_i^2 + \bar{\sigma}_l^2}} \right), \phi \left(- \frac{\log \underline{\beta_i} - \log \bar{\beta}_l}{\sqrt{\underline{\sigma_i}^2 + \underline{\sigma}_l^2}} \right) \right] \right] \quad (6.21a)$$

and

$$\underline{P_f} = 1 - \prod_{c_i \subset S} \left[1 - \min \left[\phi \left(- \frac{\log \bar{\beta}_i - \log \underline{\beta_l}}{\sqrt{\bar{\sigma}_i^2 + \bar{\sigma}_l^2}} \right), \phi \left(- \frac{\log \bar{\beta}_i - \log \underline{\beta_l}}{\sqrt{\underline{\sigma_i}^2 + \underline{\sigma}_l^2}} \right) \right] \right] \quad (6.21b)$$

- Parallel system (independent components - upper bound also valid for dependent rare events); Log-normal load and strength

$$\underline{P}_f = \prod_{c_i \subset S} \min \left[\phi \left(- \frac{\log \bar{\beta}_i - \log \bar{\beta}_l}{\sqrt{\bar{\sigma}_i^2 + \bar{\sigma}_l^2}} \right), \phi \left(- \frac{\log \bar{\beta}_i - \log \bar{\beta}_l}{\sqrt{\underline{\sigma_i}^2 + \bar{\sigma}_l^2}} \right) \right] \quad (6.22a)$$

and

$$\bar{P}_f = \prod_{c_i \subset S} \max \left[\phi \left(- \frac{\log \underline{\beta_i} - \log \underline{\beta}_l}{\sqrt{\bar{\sigma}_i^2 + \underline{\sigma}_l^2}} \right), \phi \left(- \frac{\log \underline{\beta_i} - \log \underline{\beta}_l}{\sqrt{\underline{\sigma_i}^2 + \underline{\sigma}_l^2}} \right) \right] \quad (6.22b)$$

- Single Component; Power Law Load, with $k_1 \in [\underline{k}_1, \bar{k}_1]$ and $K_H \in [\underline{K}_H, \bar{K}_H]$; Lognormal, with median $\beta \in [\underline{\beta}, \bar{\beta}]$ and logarithmic standard deviation $\sigma \in [\underline{\sigma}, \bar{\sigma}]$:

$$\bar{P}_f = \bar{k}_1 \max \left[\underline{\beta}^{-\bar{K}_H} \exp \frac{(\bar{K}_H \bar{\sigma})^2}{2}, \underline{\beta}^{-\underline{K}_H} \exp \frac{(\underline{K}_H \bar{\sigma})^2}{2} \right] \quad (6.23a)$$

and conservative lower bound

$$\underline{P}_f = \underline{k}_1 \bar{\beta}^{-\bar{K}_H} \exp \frac{(\underline{K}_H \underline{\sigma})^2}{2} \quad (6.23b)$$

If $\underline{K_H} > \frac{\log\bar{\beta}}{\underline{\sigma}^2}$ or $\bar{K}_H < \frac{\log\underline{\beta}}{\bar{\sigma}^2}$ a tighter lower bound is obtained from:

$$\underline{P}_f = \underline{k}_1 \min\left[\bar{\beta}^{-\underline{K}_H} \exp\frac{\left(\underline{K}_H\underline{\sigma}\right)^2}{2}, \bar{\beta}^{-\bar{K}_H} \exp\frac{\left(\bar{K}_H\underline{\sigma}\right)^2}{2}\right] \tag{6.23c}$$

- Parallel system with unknown dependence; Power Law Load, with $k_1 \in [\underline{k}_1, \bar{k}_1]$ and $K_H \in [\underline{K}_H, \bar{K}_H]$ Lognormal, with median $\beta_i \in [\underline{\beta}_i, \bar{\beta}_i]$ and logarithmic standard deviation $\sigma_i \in [\underline{\sigma}_i, \bar{\sigma}_i]$:

$$\bar{P}_f = \bar{k}_1 \min_{c_i \subset S}\left[\max\left[\underline{\beta}_i^{-\bar{K}_H} \exp\frac{(\bar{K}_H\bar{\sigma}_i)^2}{2}, \underline{\beta}_i^{-\underline{K}_H} \exp\frac{(\underline{K}_H\bar{\sigma}_i)^2}{2}\right]\right] \tag{6.24a}$$

and

$$\underline{P}_f = \underline{k}_1 \sum_{c_i \subset S}\left[\bar{\beta}_i^{-\bar{K}_H} \exp\frac{(\underline{K}_H\underline{\sigma}_i)^2}{2}\right] - (n-1) \tag{6.24b}$$

- Series system with unknown dependence; Power Law Load, with $k_1 \in [\underline{k}_1, \bar{k}_1]$ and $K_H \in [\underline{K}_H, \bar{K}_H]$ Log-normal, with median $\beta_i \in [\underline{\beta}_i, \bar{\beta}_i]$ and logarithmic standard deviation $\sigma_i \in [\underline{\sigma}_i, \bar{\sigma}_i]$:

$$\bar{P}_f = \bar{k}_1 \sum_{c_i \subset S} \max\left[\underline{\beta}_i^{-\bar{K}_H} \exp\frac{(\bar{K}_H\bar{\sigma}_i)^2}{2}, \underline{\beta}_i^{-\underline{K}_H} \exp\frac{(\underline{K}_H\bar{\sigma}_i)^2}{2}\right] \tag{6.25a}$$

and

$$\underline{P}_f = \underline{k}_1 \max_{c_i \subset S}\left[\bar{\beta}_i^{-\bar{K}_H} \exp\frac{(\underline{K}_H\underline{\sigma}_i)^2}{2}\right] \tag{6.25b}$$

- Parallel system with independent components (upper bound also valid for dependent rare events); Power Law Load, with $k_1 \in [\underline{k}_1, \bar{k}_1]$ and $K_H \in [\underline{K}_H, \bar{K}_H]$; Log-normal, with median $\beta_i \in [\underline{\beta}_i, \bar{\beta}_i]$ and logarithmic standard deviation $\sigma_i \in [\underline{\sigma}_i, \bar{\sigma}_i]$:

$$\bar{P}_f = \prod_{c_i \subset S} \underline{k}_1 \max\left[\underline{\beta}_i^{-\bar{K}_H} \exp\frac{(\bar{K}_H\bar{\sigma}_i)^2}{2}, \underline{\beta}_i^{-\underline{K}_H} \exp\frac{(\underline{K}_H\bar{\sigma}_i)^2}{2}\right] \tag{6.26a}$$

and

$$\underline{P}_f = \prod_{c_i \subset S} \bar{k}_1 \bar{\beta}_i^{-\bar{K}_H} \exp \frac{(\underline{K}_H \underline{\sigma_i})^2}{2} \tag{6.26b}$$

- Series system with independent components (upper bound also valid for dependent rare events); Power Law Load, with $k_1 \in [\underline{k}_1, \bar{k}_1]$ and $K_H \in [\underline{K}_H, \bar{K}_H]$; Log-normal, with median $\beta_i \in [\underline{\beta}_i, \bar{\beta}_i]$ and logarithmic standard deviation $\sigma_i \in [\underline{\sigma}_i, \bar{\sigma}_i]$:

$$\bar{P}_f = 1 - \prod_{c_i \subset S} \left[1 - \bar{k}_1 \max \left[\underline{\beta}_i^{-\bar{K}_H} \exp \frac{(\bar{K}_H \bar{\sigma}_i)^2}{2}, \underline{\beta}_i^{-\underline{K}_H} \exp \frac{(\underline{K}_H \bar{\sigma}_i)^2}{2} \right] \right] \tag{6.27a}$$

and

$$\underline{P}_f = 1 - \prod_{c_i \subset S} \left[1 - \underline{k}_1 \bar{\beta}_i^{-\bar{K}_H} \exp \frac{(\underline{K}_H \underline{\sigma_i})^2}{2} \right] \tag{6.27b}$$

The failure probability bounds for a parallel system with unknown dependencies and lognormally distributed load and strength, Eqn. 6.19, can be derived by applying the natural interval extension of the Fréchet inequality for the intersection, Eqn. 6.3, to the natural interval extension of the failure probability for a lognormal component, Eqn. 6.16.

Eqn. 6.20, the series system with unknown dependencies and lognormally distributed load and strength is derived in the same way, except this time the union Fréchet inequality (Eqn. 6.2) is applied.

Eqn. 6.21 and Eqn. 6.22 can be derived in the same way by applying Eqn. 6.1 and Eqn. 6.4, respectively.

The derivation of Eqn. 6.23 (single component with power law load and log normal fragility) is more complex, due to repeated variables (K_H) [121]. Firstly, note that $P_f = H(\beta_i) \exp \frac{(K_H \sigma_i)^2}{2} = k_1 \exp(-K_H \log \beta_i + \frac{1}{2} K_H^2 \sigma_i^2)$. Recall that $k_1 > 0$, $K_H > 0$, $\beta > 0$ and $\sigma > 0$. Note that P_f is monotonic in k_1, σ_i and β_i, so our task is simply to find $\max_{K_H} \bar{k}_1 \exp(-K_H \log \underline{\beta_i} + \frac{1}{2} K_H^2 \bar{\sigma}_i^2)$ and $\min_{K_H} \underline{k_l} \exp(-K_H \log \bar{\beta}_i + \frac{1}{2} K_H^2 \underline{\sigma}_i^2)$.

The function $k_1 \exp(-K_H \log \beta_i + \frac{1}{2} K_H^2 \sigma_i^2)$ is quadratic in K_H and has a global minima in K_H at $K_H = \frac{\log \beta}{\sigma^2}$. Clearly $\max_{K_H} \bar{k}_1 \exp(-K_H \log \underline{\beta_i} + \frac{1}{2} K_H^2 \bar{\sigma}_i^2)$ takes its maximum value at $\bar{K}_H$ or $\underline{K_H}$. Elementary interval analysis reveals that $\underline{k_l} \exp(-K_H \log \bar{\beta}_i + \frac{1}{2} K_H^2 \underline{\sigma}_i^2) > \underline{k_l} \exp(-\bar{K}_H \log \bar{\beta}_i + \frac{1}{2} \underline{K_H}^2 \underline{\sigma}_i^2)$. However in reality $\overline{K_H}$ and $\underline{K_H}$ cannot appear in the same expression, as they represent specific values of the same quantity. A tighter bound is obtained by checking if $\underline{K_H} < \frac{\log \bar{\beta}_i}{\underline{\sigma_i}^2} < \bar{K}_H$. If this inequality holds then the minimum occurs at $K_H = \frac{\log \bar{\beta}_i}{\underline{\sigma_i}^2}$. Otherwise we must check which of $\bar{K}_H$ and $\underline{K_H}$ minimises the failure probability. Then the remaining results can be obtained by applying the union or intersection Fréchet inequalities, or rare event approximation as appropriate.

6.3.5 Imprecise FORM

The bounds on the failure probability of a structural system in the load resistance form, subject to distributional probability box uncertainty, can be approximated using the well known FORM approximation described in Section 4.3.2. If the resistance variable consists of the sum of many component strengths then one may need to use optimisation to find the reliability index. Alternatively, we can attempt to analyse in which conditions the system is likely to fail using a simple analytical method.

Consider a load term which is the product of a constant and a random variable, i.e. $L = CL_d$, where C is a constant and L_d is a random variable representing the design load. The system will have a $P_f = 0.5$ when $\beta = 0$, which implies the strength to load ratio, $\gamma = \frac{\mu_S}{\mu_L}$, will be equal to 1. Clearly, this is only the case when $C = \gamma_d = \frac{\mu_S}{\mu_{L_d}}$, i.e. the applied load is scaled by the strength to design load ratio [143].

This can be trivially extended in the case of probability box variables to find an interval load for which $P_f = 0.5$, i.e. $L_{0.5} \in [\underline{L_{0.5}}, \bar{L}_{0.5}] = [\underline{\gamma}_d L_d, \bar{\gamma}_d L_d,]$ where

$$\bar{\gamma}_d = \frac{\bar{\mu}_S}{\underline{\mu}_{L_d}}, \tag{6.28}$$

and

$$\underline{\gamma}_d = \frac{\underline{\mu}_S}{\bar{\mu}_{L_d}}, \tag{6.29}$$

where μ_S and μ_L are the mean values of the strength and load and σ_L and σ_S are the standard deviations of the strength and load. Note that the standard deviation of the random variables is not involved in the calculation of this load.

6.4 Numerical examples

6.4.1 Reliability analysis of a simple concrete containment

To demonstrate the results described in the previous sections, we will consider a modified version of an example given in Modarres et al. [119] with interval imprecision in the coefficient of variation of the random variables. The random variables will be modelled with lognormal distributions since lognormal distributions are commonly used in the Probabilistic Safety Analysis literature to model physical quantities which must always be positive [168], [98]. However, our approach could be applied to similar problems with different distribution types, and many other distributions exist to ensure positivity of random variables. The problem description will be briefly replicated in this section for clarity.

A concrete containment is a structure designed to prevent radioactive release from nuclear power plants to the environment. It is therefore important that the reliability of this structure can be determined accurately, as failing to do so could have severe consequences for the environment and the general public. During the process of determining the reliability of a containment, engineers wish to determine the relationship between applied pressure and failure probability of the containment. A simplified performance function is used to perform reliability analysis without having to run simulations on a complex finite element model. This approach is advantageous as the computational time required is significantly reduced. The approach assumes that the system will fail if the load is larger than the strength.

The containment's strength is considered to be divided between 7 failure mode contributors, all of which may cause system failure. Therefore, this example can be treated as a system composed of 7 components (which are modelled as random variables), connected in series.

The probability of failure for the containment is given by

$$P_f = \int_{S_t < L_t} f(\boldsymbol{x}) d\boldsymbol{x}, \tag{6.30}$$

where $f(\boldsymbol{x})$ is the joint probability distribution function of the random variables, $\boldsymbol{x} = (x_1, x_2 ...)$ and S_t and L_t represent the strength and load terms respectively. The input parameter values assumed in this analysis were taken from the original example given by Modarres et al. [119], but modified to fit lognormal variables and include some imprecision as shown in Table 6.1. The pressure load inside the containment, for the specific accident being

Failure Mode	Logarithmic Median, μ, $\log \beta$ /MPa	Logarithmic Standard Deviation, σ /MPa
Liner tear around personnel airlock	-0.0943	$[0, 0.0017]$
Basemat shear	-0.0141	$[0, 0.0016]$
Cylinder hoop membrane	0.0853	$[0, 8.8641 \times 10^{-4}]$
Wall-basemat junction shear	0.1231	$[0, 0.0014]$
Cylinder meridional membrane	0.2159	$[0, 8.3320 \times 10^{-4}]$
Dome membrane	0.5911	$[0, 5.345 \times 10^{-4}]$
Personnel air lock door buckling	0.2159	$[0, 0.0013]$

Table 6.1: Input parameters for the modified concrete containment example from Modarres et al. [119].

considered, was taken to be lognormally distributed with mean 0.575 MPa and standard deviation of 0.117 MPa (such that the parameters for the fitted lognormal distribution were $\log \beta = -0.5737$ MPa and $\sigma = 0.2014$ MPa).

The fragility of the series system was bounded using Eqn. 6.9 and compared to the empirical CDFs obtained by randomly sampling the epistemic uncertainty. The results are shown in Figure 6.5.

The failure probability was calculated using Eqn. 6.20, since the dependence between failure modes was unknown. This resulted in a failure probability between 0.0086 and 0.0123, which contains the precise probability of failure ($P_f = 0.0122$) given in Modarres et al. [119]. This result was verified by use of double loop Monte Carlo simulation, which was performed using the same samples used to generate Figure 6.5 (100 epistemic samples and 10^6 aleatory samples). The analytic code took 0.027 seconds to run, whilst the double loop Monte Carlo simulation took 0.16 seconds to run on an 2.9 GHz Intel Core i5 processor in Matlab. In addition the result from double loop Monte Carlo simulation would require more samples, and hence even greater time, to increase accuracy in the tails of the p-box to an arbitrary amount already achieved by the analytic approach.

These results reveal a good agreement with the expensive simulation procedures in a fraction (one fifth) of the time. Note that although in this case the double loop Monte Carlo was quick to run, this may not be true in general (such as in high dimensional cases). In addition, the Monte Carlo simulation could be one nested component in a much larger computation. Even when this is not the case, it is unrealistic to expect practising engineers to resort to double loop Monte Carlo simulation for what should be a simple

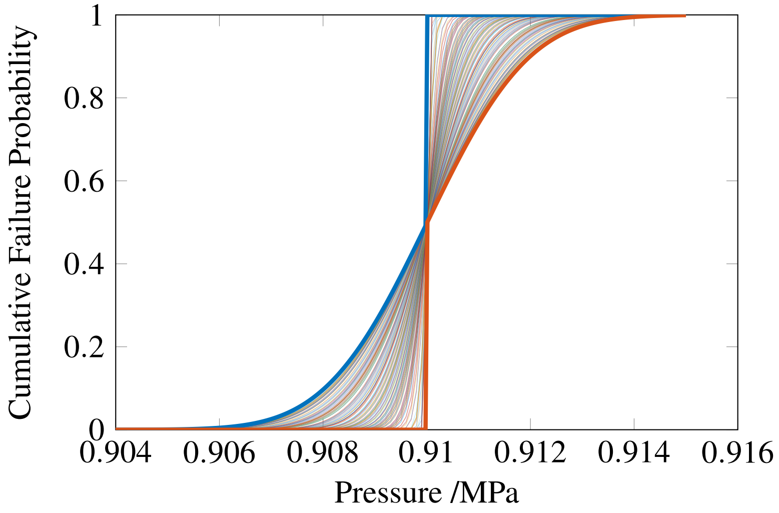

Figure 6.5: Probability box representing the fragility curve of the series system, computed analytically. For comparison, sampled fragility curves double loop Monte Carlo simulation are shown, which was computed by making 100 epistemic samples.

design calculation, even with the inclusion of epistemic uncertainty. In practical cases it would also be necessary to consider uncertainty in the Logarithmic Mean of the random variables which can be easily accounted for given the developments in Section 6.3.

6.4.2 Containment with additive component strengths

In many real systems the components' strengths may be added together, rather than combined in parallel or series. Such an example is given in Chapter 5. This poses a challenge for analytical methods, as in general normal distributions and log normal distributions cannot be summed easily (except in limited cases such as independently distributed normal random variables). Therefore, in order to consider such systems in the imprecise probabilistic safety analysis framework, we resort to using the imprecise FORM approximations given in Qiu et al. [144].

In Chapter 5, Probabilistic Safety Analysis of a concrete containment was presented as part of a round robin international test exercise. Two experimental test cases (Sandia National Laboratories and Bhabha Atomic Research Centre) were described and the probability of failure for each containment was calculated. The experiments were compared

Variable	Mean Value, $[\underline{\mu},\bar{\mu}]$	Coefficient of Variation
Concrete tensile strength, F_c	$[4.3, 4.5]$	0.2
Liner yield, F_l	$[370, 390]$	0.2
Rebar yield, F_s	$[450, 370]$	0.2
Tendon yield, F_t	$[1700, 1800]$	0.2
Design Pressure, P_d	0.39	0.2
Radius, R	5537.5	0.2
Concrete area, A_c	312.85	0.2
Liner area, A_l	1.6	0.2
Rebar area , A_s	6.85	0.2
Tendon area, A_t	3.7	0.2

Table 6.2: Input parameters for Sandia National Laboratories containment test case with additive component strengths.

to a cylindrical concrete containment model, where the area and strength of the concrete, rebar, tendons and liner are modelled as normally distributed random variables. In this example, the Sandia National Laboratories Containment will be analysed with intervalised epistemic uncertainty describing the mean value of the random variables representing yield values of structural materials. This could indicate lack of knowledge about the materials used, i.e. insufficient experiments. The modified properties of the Sandia National Laboratories containment are summarised in Table 6.2.

The performance function of the containment is obtained as a load-strength relationship, i.e.

$$g = (A_s F_s + A_t F_t + A_l F_l + A_c F_c) - PR. \tag{6.31}$$

We set the applied pressure to be equal to the design pressure, scaled by a constant.

Using the strength to design load ratio method from Eqn. 6.28 and Eqn. 6.29 with

$$\frac{\bar{\mu}_S}{\underline{\mu}_L} = \frac{\bar{\mu}_{A_s}\bar{\mu}_{F_s} + \bar{\mu}_{A_t}\bar{\mu}_{F_t} + \bar{\mu}_{A_c}\bar{\mu}_{F_c} + \bar{\mu}_{A_l}\bar{\mu}_{F_l}}{\underline{\mu}_{P_d}\underline{\mu}_R} \tag{6.32}$$

and

$$\frac{\underline{\mu}_S}{\bar{\mu}_L} = \frac{\underline{\mu}_{A_s}\underline{\mu}_{F_s} + \underline{\mu}_{A_t}\underline{\mu}_{F_t} + \underline{\mu}_{A_c}\underline{\mu}_{F_c} + \underline{\mu}_{A_l}\underline{\mu}_{F_l}}{\bar{\mu}_{P_d}\bar{\mu}_R} \tag{6.33}$$

we find that $P_f = 0.5$ when $P \in [5.2P_d, 5.24P_d]$. In order words, because of our epistemic uncertainty in the structural properties of the system we are unsure which pressure causes

$P_f = 0.5$. Clearly the epistemic uncertainty we have considered does not significantly change the pressure at which $P_f = 0.5$.

For a more complete understanding of the system (i.e. understanding which pressures cause large and small failure probabilities), advanced simulation methods would be necessary. This is because the strength to design-load ratio method only considers the mean values of the random variable in order to find the pressure at which the structure has $P_f = 0.5$, and does not consider the variability of the structural components. For example, one could resort to the method proposed by de Angelis et al. [51], where line sampling is applied to structures with epistemic uncertainties.

6.5 Chapter summary

In this chapter, we have demonstrated methods to analytically propagate probability boxes in commonly used Probabilistic Safety Analysis equations. These equations include series and parallel systems with unknown dependencies, lognormal fragility distributions and equations where lognormally distributed factors are multiplied. In addition, Power Law Load load distributions are considered. Crucially, we use intervals to model epistemic uncertainty in the parameters of these distributions. This enables the robust quantification of epistemic uncertainty when performing Probabilistic Safety Analysis, particularly in an industrial context. These distributions are sufficient for the analysis of many industrial problems, but in general the imprecise probability methods proposed could be generalised to other distributions as well.

These expressions are imprecise probabilistic analogues to many of the probabilistic formulae proposed in Kennedy et al. [98], which have become standard expressions used in Probabilistic Safety Analysis. We also demonstrated how similar techniques can be applied to simplified calculations involving more complex models.

Our proposed expressions enable engineers to complete essential design calculations whilst considering epistemic uncertainty, and avoid the impracticalities of double loop Monte Carlo simulation which we believe is a significant barrier to the modelling of epistemic uncertainty in many industrial probabilistic safety assessment workflows. However, the proposed methodology in this chapter cannot be applied for black box models, which do not have an analytic performance function. Therefore, the following chapters propose alternative methodologies which still reduce computational demands, whilst being compatible with more general simulations and models.

Chapter 7

Interval Predictor Models for Reliability Analysis

7.1 Introduction

As discussed in Chapter 4, metamodels can be used to reduce the computational expense of a Monte Carlo simulation to calculate the probability of failure of a system. However, since an approximate model is used to predict the model response, the surrogate approximation introduces a prediction uncertainty in the model response [178]. Consequently, this prediction uncertainty propagates to uncertainty concerning the computed probability of failure, that has to be effectively estimated and accounted for in such approximate analyses. Interval predictor models (validated by the scenario optimisation theory), as discussed in Chapter 3, are a type of metamodel which provide a robust quantification of their predictive uncertainty. This chapter therefore presents a systematic approach to consider such prediction uncertainty in the estimation of small failure probabilities in nonlinear models. Section 7.3.2 describes how IPMs used in the literature can be modified to create more accurate metamodels for performance functions. An analytic case study is performed in Section 7.4 to illustrate the proposed approach, where the performance of interval predictor models is compared to that of Kriging models (i.e. Gaussian Processes). Advanced Monte Carlo methods are used to present a benchmark for the proposed approaches.

The use of interval predictor models for reliability analysis is a novel contribution of the author in Patelli et al. [134]. Subsequently, they have also been applied in Crespo et al. [47].

7.2 Interval failure probability

When calculating the failure probability of a system, a metamodel is often applied to predict the performance function $g(\boldsymbol{x})$. The prediction of $g(\boldsymbol{x})$ by the metamodel will be referred to as $\hat{g}$. Both Kriging and Interval Predictor Model (IPM) surrogate models either give an estimate of the uncertainty on the prediction of the model response or provide the analyst with a set-valued response that prescribes this uncertainty. In both cases, the predicted response $\hat{g}$ is modelled as belonging to an interval $\hat{g}^I$. In the context of failure probability estimation, the resulting random model response can be regarded as belonging to a probability box $[\hat{g}]$ due to the superposition of the interval uncertainty from the surrogate model on the probabilistic description of the response $g(\boldsymbol{x})$ stemming from the random model parameters $\boldsymbol{x}$. As a consequence, $\hat{P}_f$ becomes interval valued. Specifically, the interval probability of failure, $\hat{P}_f^I$, can be computed as:

$$\hat{P}_f^I = \int_{\mathbb{R}} \mathbb{I}_f([\hat{g}]) f^I_{\hat{g}^I}([\hat{g}]) d[\hat{g}] \tag{7.1}$$

which can be solved following e.g. a nested optimisation approach [111].

However in this specific context, some considerations allow for simplification of this equation. In case of Kriging, the superimposed interval uncertainty on the predicted model response is strict in the sense that the upper and lower bounds do not cross. This is a direct result from the truncation of the random variable that is associated to each predicted response. During the training of the IPM, the upper prediction is constrained to be greater than the lower prediction and hence a similar observation can be made in this context, as demonstrated in Patelli et al. [134] and Crespo et al. [47]. Therefore, only the extreme bounds of the predicted response intervals need to be considered in the evaluation of the failure probability. As such, Eqn. 7.1 can be split up as in Zhang et al. [187], and a Monte Carlo estimator applied to yield

$$\underline{\hat{P}}_f = \int_{\mathbb{R}} \mathbb{I}_f(\underline{\hat{g}}) \underline{f}_{\underline{\hat{g}}}(\underline{\hat{g}}) d\underline{\hat{g}} \quad \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}_f(\underline{\hat{g}}_i) \tag{7.2a}$$

$$\hat{\overline{P}}_f = \int_{\mathbb{R}} \mathbb{I}_f(\hat{\overline{g}}) \overline{f}_{\hat{\overline{g}}}(\hat{\overline{g}}) d\hat{\overline{g}} \quad \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}_f(\hat{\overline{g}}_i) \tag{7.2b}$$

where, $\underline{f}_{\underline{\hat{g}}}(\underline{\hat{g}})$ and $\overline{f}_{\hat{\overline{g}}}(\hat{\overline{g}})$ are respectively the distribution function of the lower and upper

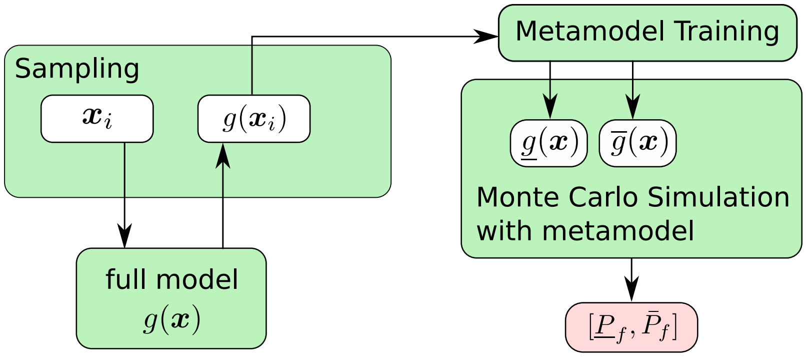
Metamodel Training
Sampling
$\boldsymbol{x}_i$
$g(\boldsymbol{x}_i)$
$\underline{g}(\boldsymbol{x})$
$\overline{g}(\boldsymbol{x})$
Monte Carlo Simulation with metamodel
full model $g(\boldsymbol{x})$
$[\underline{P}_f, \bar{P}_f]$

Figure 7.1: A diagram of Monte Carlo simulation with an uncertain surrogate model.

bounds on the prediction of the surrogate model, $\hat{\bar{g}}$ and $\underline{\hat{g}}$, and $\hat{\bar{g}}_i$ and $\underline{\hat{g}}_i$ are the N samples drawn from these distributions. This is shown in Figure 7.1.

In case dependent random model parameters are considered, the computation of the failure probability is usually performed in standard normal space (SNS). Due to the interval-valued uncertainty that is attributed to each realisation of the random model response, the limit state function obtained from the metamodel becomes interval valued after transformation to SNS. However, it can be shown that due to the monotonicity of the iso-probabilistic transformation to SNS (see Jiang et al. [89]), the minimum and maximum value of the limit state function correspond to the vertices of the interval-valued uncertainty on the model response realisations. Therefore, the above arguments also hold in this case.

This method as such allows an analyst to make a robust prediction of the probability of failure of a highly non-linear, computationally demanding computer model at greatly reduced cost. It furthermore allows the analyst to uniquely separate the uncertainty stemming from the modelled physics from the uncertainty that stems from applying a surrogate model instead of the full numerical simulation code.

7.3 Uncertain surrogate model predictions

This section describes how the uncertain surrogate model predictions required by the interval failure probability estimation described in Section 7.2 are obtained for Kriging and interval predictor models. Specific optimisations which can be used to create IPMs which more accurately model performance functions are explained.

7.3.1 Kriging

Kriging approximates the full model $g(\boldsymbol{x})$ as the sum of a functional regression model $F(\boldsymbol{\beta}, \boldsymbol{x})$, where F is usually a polynomial function and $\boldsymbol{\beta}$ represents the regression coefficients, and a stationary zero-mean Gaussian stochastic process $z(\boldsymbol{x})$ [105]. Formally, the Kriging surrogate model $\hat{m}_{Kr}()$ is expressed as:

$$\hat{g} = \hat{m}_{Kr}(\boldsymbol{x}) = F(\boldsymbol{\beta}, \boldsymbol{x}) + z(\boldsymbol{x}). \tag{7.3}$$

In Eqn. 7.3, the polynomial regression model is given as the linear superposition of a number of functions $f(\boldsymbol{x}) : \mathbb{R}^n \mapsto \mathbb{R}$:

$$F(\boldsymbol{\beta}, \boldsymbol{x}) = \boldsymbol{f}^T(\boldsymbol{x})\boldsymbol{\beta} \tag{7.4}$$

where $\boldsymbol{\beta}$ are the corresponding regression coefficients that have to be estimated. The auto-covariance of the stationary zero-mean Gaussian stochastic process $z(\boldsymbol{x})$ is given as:

$$\mathbb{E}[z(\boldsymbol{x}_i), z(\boldsymbol{x}_j)] = \sigma^2 R(\theta, x_i, x_j) \tag{7.5}$$

with σ the process variance and $R(\theta, x_i, x_j)$ the correlation model between two x_i, x_j in $\mathcal{X}$. The correlation model is characterised by a set of coefficients θ.

The degree of the polynomial regression model and the correlation function family are selected by the analyst, based on expert opinion. Then, the correlation coefficients, process variance and correlation parameter θ are determined using a Bayesian supervised learning procedure, as discussed in Chapter 3.

Since Kriging associates a Gaussian random variable to each predicted $\hat{g} = \hat{m}_{Kr}(\boldsymbol{x})$, an estimation of the variance $\zeta(\boldsymbol{x})$ in the prediction is given by the Kriging model. By considering the $k \cdot \sigma$-bounds of a Kriging Metamodel, the response of the Kriging predictor can as such be interpreted as an interval:

$$\hat{g}^I = [\hat{m}_{Kr}(\boldsymbol{x}) - k \cdot \zeta(\boldsymbol{x});\ \hat{m}_{Kr}(\boldsymbol{x}) + k \cdot \zeta(\boldsymbol{x})]. \tag{7.6}$$

This interval is by definition symmetric around the deterministic estimate of the Kriging model. By applying this method for the model response, an interval $\hat{g}^I$ containing the $k \cdot \sigma$ confidence interval of the model response is obtained next to the deterministic estimate $\hat{g}_l$ of the model response.

7.3.2 Interval predictor models

In contrast to most surrogate modelling approaches, interval predictor models (IPMs) provide the analyst with a set-valued mapp ing $m^I_{IPM} : \boldsymbol{x} \mapsto g^I \subset \mathcal{G}$ instead of only one crisp value. Specifically, the IPM maps the crisp valued vector of input parameters $\boldsymbol{x}$ to an interval g^I bounding the range of the actual crisp model prediction, as discussed in Chapter 3. IPMs for reliability analysis are created by obtaining an input training data set of samples from $f(\boldsymbol{x})$, $\boldsymbol{x}_i$, which are propagated through $g(\boldsymbol{x})$ to obtain output training data, g_i. When training an IPM as a metamodel to estimate small failure probabilities, it is important to evaluate the objective function (the expected distance between the bounds) either analytically or with high accuracy. This is because the standard deviation of the empirical estimate of the failure probability may well be larger than the failure probability in these cases. It is clear that obtaining more data will expand the IPM's prediction interval, and without observing an infinite amount of data the obtained bounds on the IPM prediction interval will never be completely robust. Therefore, the accuracy of the IPM is assessed using tools from the scenario optimisation theory described in Chapter 3.

Modified objective function

Since the purpose of structural reliability analysis is to obtain the failure probability, which is calculated by integration of an indicator function, the priority should be to model the performance function as accurately as possible where it is close to zero [2]. This is not achieved in regular interval predictor models because the objective function ($\mathbb{E}_{\boldsymbol{x}}[\delta_y(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{u})]$) minimises the expectation of the width of the IPM everywhere in the data containing region. A hyperbolic tangent objective function ($w(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{u})$) can be used to remedy the problem, which is given by

$$w(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{u}) = \sum_i \tanh\left(b\hat{\overline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u})\right) - \tanh\left(b\underline{\hat{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u})\right), \tag{7.7}$$

where b is a positive real scaling factor which can be increased or decreased to alter the convergence of the IPM. Increasing b will reduce the objective function to the indicator function — i.e. the IPM will behave more like a classifier. In fact, since the proposed objective function is an expectation, it will be equal to twice the difference between the empirically computed bounds on the probability of failure ($2(\underline{P}_f - \overline{P}_f)$). This loss function is particularly useful whenever a metamodel is created for the purposes of reliability analysis.

Decreasing b will result in a linear scaling between the objective and $g(\boldsymbol{x})$, i.e. similar to the unmodified objective function. This function is advantageous because when the IPM is close to zero there is a clear incentive to make the model as tight as possible here. Furthermore, the function is smooth and analytically differentiable with respect to the parameter vector, which permits easier optimisation, and therefore its gradient can be obtained as

$$\begin{aligned}\nabla_{\boldsymbol{u}} w(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{u}) = \sum_i b \frac{\phi(\boldsymbol{x}_i) - |\phi(\boldsymbol{x}_i)|}{2} \operatorname{sech}^2\left(b\hat{\overline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u})\right) - \\ \sum_i b \frac{\phi(\boldsymbol{x}_i) + |\phi(\boldsymbol{x}_i)|}{2} \operatorname{sech}^2\left(b\hat{\underline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u})\right), \quad (7.8)\end{aligned}$$

and

$$\begin{aligned}\nabla_{\boldsymbol{v}} w(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{u}) = \sum_i b \frac{\phi(\boldsymbol{x}_i) + |\phi(\boldsymbol{x}_i)|}{2} \operatorname{sech}^2\left(b\hat{\overline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u})\right) - \\ \sum_i b \frac{\phi(\boldsymbol{x}_i) - |\phi(\boldsymbol{x}_i)|}{2} \operatorname{sech}^2\left(b\hat{\underline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u})\right). \quad (7.9)\end{aligned}$$

tanh is a non-convex function, and therefore one may wish to define a convex approximation of the function for practical purposes. In this chapter the approximate loss function

$$w_{\text{approx}}(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{u}) = \sum_i w_i(\hat{\overline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u}) - \hat{\underline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u})), \quad (7.10)$$

is used, with $w_i = \frac{\tanh g_i}{g_i}$. In other words, the original IPM loss is re-weighted when the data is close to the limit state surface, whilst the loss remains a linear function of the IPM parameters. In fact, Eqn. 7.10 is an upper bound to the first order Taylor expansion of Eqn. 7.7 about g_i for $b = 1$ $(\operatorname{sech}^2(g_i)(\hat{\overline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u}) - \hat{\underline{g}}(\boldsymbol{x}_i, \boldsymbol{v}, \boldsymbol{u})))$. This approximation will only be reasonable when the data has low noise in g, otherwise higher order terms in the Taylor approximation will become important.

Adaptive training of interval predictor models

In order to reduce the number of support constraints in the IPM and hence improve its reliability, two strategies were adopted. Firstly we set $\bar{p}_i = \underline{p}_i$ for $i > 1$, in other words the parameter vector was the same for the upper and lower bound except for a constant, which almost halves the bound on the number of support constraints. This strategy works particularly well when modelling deterministic functions. Secondly, an iterative scheme is used to refine the basis chosen. Firstly, a polynomial basis with the maximum required degree is created and then the IPM is trained. The monomial term with the lowest p_i is removed. The IPM is now retrained with the new basis and the procedure is repeated until the IPM has a sufficiently small uncertainty.

Producing point predictions from interval predictor models

The IPM does not provide a crisp *best estimate* value of the model response. For comparison with the crisp value that is provided by the full model $g(\boldsymbol{x})$ and the Kriging predictor, the least squares estimate using the basis chosen for the IPM is used. This should be roughly similar to finding the mean of a staircase predictor model, as in [47].

7.4 Case study

In the study of the uncertainty concerning the estimation failure probability due to the application of surrogate modelling techniques, Adjiman's function is applied:

$$f_{adj}(x_1, x_2) = \cos(x_1) \cdot \sin(x_2) - \frac{x_1}{(x_2^2 + 1)}. \tag{7.11}$$

Based on this function, decreasing levels of failure probability are estimated by considering the threshold value for $y_{th} \in \{2, 2.5, 3, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 3.95, 4\}$. The performance function is therefore $g(\boldsymbol{x}) = y_{th} - f_{adj}(x_1, x_2)$. Both x_1 and x_2 are assumed to be marginally uniform distributed within the interval $[-4, 4]$ with zero covariance.

As a benchmark, advanced Monte Carlo methods such as Line Sampling and Subset simulation, as well as regular Monte Carlo simulation are applied, and their performance in terms of necessary number of function evaluations and variance of the predictor are compared. Then, different surrogate models for Adjiman's function are constructed using three techniques:

- an Interval Predictor Model, based on a 7^{th}-order polynomial basis, refined using a basis refinement algorithm until only 12 monomials are present,
- a Kriging model with 2^{nd}-order regression model $F(\beta, \boldsymbol{x})$ and an exponential correlation model $R(\boldsymbol{\theta}; \boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\theta|\boldsymbol{x}_i - \boldsymbol{x}_j|)$,

and these surrogate models are applied to perform a Monte Carlo integration of Eqn. 7.2b. Both modelling techniques are applied to the same training data sets containing either 100, 250, 500 or 1000 deterministic training samples. In order to make a fair comparison between the two proposed techniques, the modified objective function from Eqn. 7.10 is not used in this case study, and instead the IPM is trained in the standard way (i.e. minimising the expectation of the difference between the bounds with Eqn. 3.9). It should be noted that no computational gain is expected in the application of a surrogate model for the considered test function, because the chosen test function is a simple analytic function. Nonetheless, the experiment enables the accuracy in predicting small failure probabilities of the considered surrogate modelling techniques to be compared in a rigorous way.

Since the considered surrogate modelling approaches are conceptually very different, comparison of their accuracy based on some a priori (i.e. before computing P_f) metric is non-informative. The most obvious way would be to compute, for instance, the R^2-value and the Chebyshev norm (D_{ch}) of the difference between the analytical model and surrogate prediction using a set of validation data. However, since the Interval Predictor Model only provides a set valued response for each combination of parameter values, such metrics computed over, for instance, the midpoint of the predicted intervals are non-informative. Hence, such comparisons do not reveal much about the performance of the methods. All numerical computations, are performed using OpenCossan.

7.4.1 Advanced Monte Carlo sampling

As a first step in the analysis, the performance of Monte Carlo, Line Sampling with an adaptive algorithm to find the important direction (see De Angelis [50]) and Subset-∞ [9] is tested in terms of the estimation of the failure probability, the coefficient of variance of this estimation and the number of samples that were needed to obtain the estimate.[*] These simulation methods are applied directly using the analytical function, as introduced in Eqn. 7.11, to ensure that this analysis is not biased due to prediction errors of the surrogate models.

The Monte Carlo and Line Sampling methods were applied until a coefficient of variation (CoV) of the estimator of 5% was reached, albeit with a maximum of 10^7 samples. The sampling was performed in batches of $5 \cdot 10^2$ samples for Monte Carlo simulation and 200

[*] The Subset-∞ algorithm proceeds as described in Chapter 4, but with a more efficient method to generate samples between intermediate failure events.

lines for Line Sampling. Then, after each batch the CoV is estimated and the simulation is stopped if $CoV < 0.05$. For Subset-∞, the intermediate levels of P_f were set to 0.1 and the initial population size was heuristically set until a sufficiently small CoV was obtained. A CoV of approx. 8% for the prediction of P_f for $y_{th} = 2$ was obtained at 10^3 samples, as the CoV did not improve significantly when the population size was further increased. The same initial population size was kept constant for all other evaluations of the failure probability.

Figure 7.2 illustrates the topology of the limit state function of Adjiman's function in the standard normal space $\mathcal{U}$. Herein, u_1 and u_2 respectively correspond to $u_1 = T_u(x_1)$ and $u_2 = T_u(x_2)$, with $T_u : \mathcal{X} \mapsto \mathcal{U}$ a transformation operator mapping responses from physical to standard normal space. This plot is generated by performing $5 \cdot 10^4$ Monte-Carlo evaluations of the analytical function, with a threshold value of $y_{th} = 3.7$. The red dots in this figure indicate the samples laying in the failure domain $\mathcal{F}$ (i.e., $\mathcal{I} \leq 0$), whereas the samples in the safe domain $\mathcal{S}$ (i.e., $\mathcal{I} > 0$) are indicated in green. As it may be noted, a highly non-linear notched limit state function $g(\boldsymbol{u})$ is obtained, which poses a challenge for the applied advanced Monte Carlo methods.

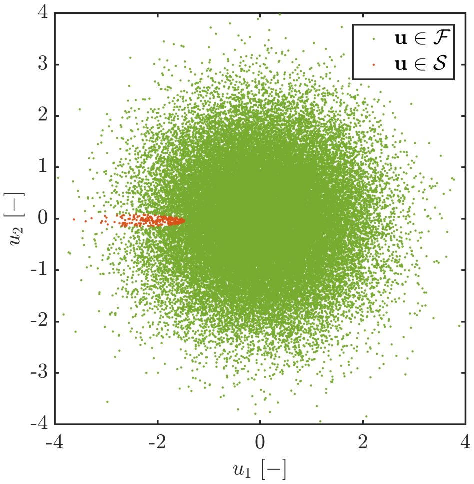

Figure 7.2: Failure domain $\mathcal{F}$ and safe domain $\mathcal{S}$ in standard normal space for Adjiman's function.

Figure 7.3 shows the estimated failure probability, as obtained using Monte Carlo, Advanced Line Sampling and Subset-∞, as a function of the threshold value. First, it can be seen that the estimate of the failure probability as a function of the threshold of y_{th} is approximately equal for Monte Carlo and the Subset methods, as long as the failure probability remains moderately large (i.e., $P_f > 10^{-3}$). However, the obtained results diverge significantly when smaller failure probabilities are computed. Advanced Line Sampling on the other hand provides in this case a better estimate for the smaller failure probabilities, which is explained by the independence of Line Sampling performance to the magnitude of the probability of failure [50].

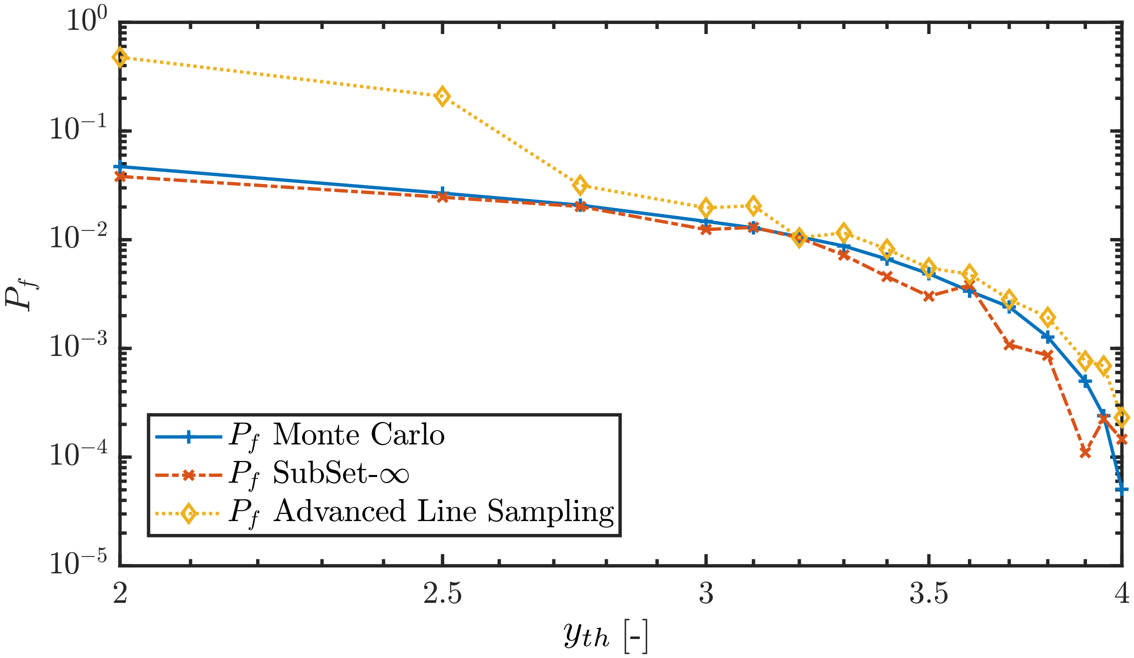

Figure 7.3: Estimated failure probability and the coefficient of variance for different threshold values y_{th} for Adjiman's function.

Figure 7.4 shows the CoV of the failure probabilities estimated by the three methods. It can be noted that the variance on the failure probability predictor that is obtained by Monte Carlo and Advanced Line Sampling is up to a factor 5 smaller as compared to Subset-∞. This is a direct result from the fact that in the case of Monte Carlo and Advanced Line Sampling, additional samples were generated until a specified CoV of 5% was reached, whereas the Subset method was heuristically tuned to minimise the CoV of the prediction. Moreover, in the case of Subset, the CoV measures up to 60% in the case of the smallest considered failure probabilities.

Figure 7.5 shows the computational efficiency in terms of necessary number of samples

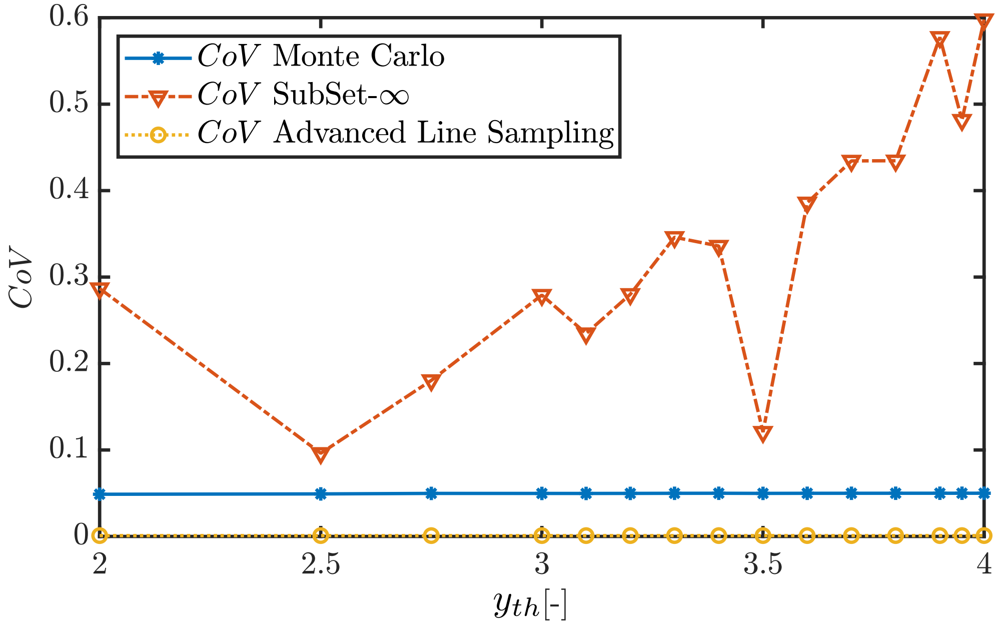

Figure 7.4: Estimated failure probability and the coefficient of variance for different threshold values y_{th} for Adjiman's function.

to perform the probability of failure estimate. From this figure, it is clear that Subset-∞ is more efficient than Advanced Line Sampling, which in its turn is more efficient than standard Monte Carlo simulation for the estimation of the failure probability. This is particularly true when small failure probabilities are considered. However, in that context it should be noted that the variance of the Monte Carlo estimator is an order of magnitude lower as compared to the variance of $\hat{P}_f$, as obtained by Subset, which limits the credibility of the estimate. The variance of P_f obtained via Advanced Line Sampling is approximately equal to that of Monte Carlo, albeit at a strongly reduced computational cost.

It should be noted that Subset-∞, the most efficient technique, still requires more than 2000 model evaluations, which is prohibitive when the estimation of the failure probability of a structure using computationally expensive computer models $g(\boldsymbol{x})$ is considered.

As such, it can be concluded that although highly performing advanced Monte Carlo methods exist to date, the estimation of small failure probabilities in highly non-linear models still can prove to be computationally very demanding. Therefore, even using these advanced Monte Carlo methods, the application of surrogate modelling techniques still

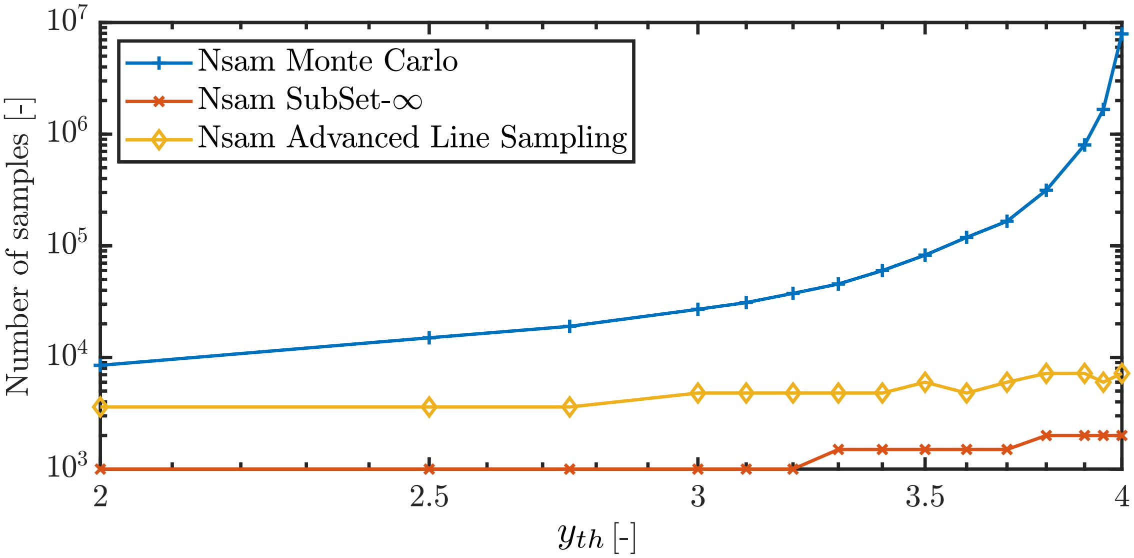

Figure 7.5: Number of necessary samples of the advanced Monte Carlo methods for different threshold values y_{th} for Adjiman's function.

proves to be of importance, as the training of such surrogate model typically requires less model evaluations as compared to a direct application of the advanced Monte Carlo methods for the estimation of a small probability of failure. As discussed in Section 7.2, this imposes uncertainty on the prediction of the failure probability as well.

7.4.2 Surrogate model based estimation

Using the constructed surrogate models, decreasing levels of failure probability are estimated by performing Monte Carlo sampling until the CoV of the predictor was less than 5%, analogously to the method that was applied in Section 7.4.1.

The results for each estimation of the failure probability, for each of the constructed surrogate models is illustrated in Figures 7.6 and 7.7. For the Kriging models, the $2 \cdot \sigma$ bounds are considered, which yield a 95.5% confidence interval for P_f. In order to make a fair comparison, for the IPM the uncertainty in the bounds on P_f is considered as being less than ϵ when $\beta = 1 - 96$. In other words, the bounds on P_f obtained from integrating over the bounds of the IPM must be expanded by ϵ.

Figure 7.6 illustrates the performance of the regular Kriging surrogate modelling approach. Specifically, the $\pm 2 \cdot \sigma$ bounds are illustrated together with the crisp (mean) estimate of the model for all considered training data sets. Also the prediction of the failure probability using the analytic model is illustrated. First, in case sufficient data are used for the training, the regular Kriging is capable of providing a relatively accurate crisp estimate of the failure probability, as long as $P_f > 5 \cdot 10^{-3}$. For smaller failure probabilities, Kriging fails in all cases. Second, it can be noted that the Kriging prediction is conservative in the sense that the $\pm 2 \cdot \sigma$ always encompasses the true failure probability. However, the lower bound prediction fails in all cases when $y_{th} > 3.7$. This is due to the difficulty of sampling small failure probabilities with standard Monte Carlo with a limited sample set. Finally, when more data are included in the training of the Kriging model, the $\pm 2 \cdot \sigma$ bounds on the prediction become tighter. This is a direct result of the conditioned random field that underlies these predictions. When more points are located throughout the model domain, the relative distance between training points decreases, and as such also the variance of the predicted random variable.

Figure 7.7 illustrates the performance of the Interval Predictor Model in predicting the upper bound of P_f. Specifically, the $\pm\epsilon$ bounds on the prediction of the upper limit of the failure probability $\bar{P}_f$ are illustrated for all data sets. Also the prediction of the failure probability using the analytic model is illustrated. Only the upper bound of the IPM is illustrated for visualisation purposes, since this is the most relevant from an engineering standpoint. First, it can be seen that the exact failure probability always lies inside the ϵ bounds of the upper bound prediction of the IPM. Hence, the IPM always gives a safe estimation of the failure probability. However, when the true P_f becomes smaller than 0.01 for the model trained with 1000 samples, the ϵ bounds inflate very quickly, making the estimate very conservative. This behaviour is more pronounced for smaller data sets, since the confidence in the interval is proportional to the size of the training data set. Finally, it can be noted that the upper bound prediction of the set $\{y_{th} : y_{th} < 3.5\}$, without taking ϵ into account is more accurate than the IPM that is trained with 1000 samples. This perhaps indicates over-fitting of the polynomial basis to the training data, which could be aggravated by the iterative pruning of the polynomial basis as explained in Section 7.3.2.

The results indicate that to an order of magnitude, the performance of the IPM is similar to that of Kriging. However, the IPM requires fewer assumptions to be made regarding the functional form of the function being approximated. For example, the Kriging assumption of a continuous function is not required.

It can be seen that by using a surrogate model, computational expenses for evaluating small failure probabilities can be decreased drastically. This statement is based on the

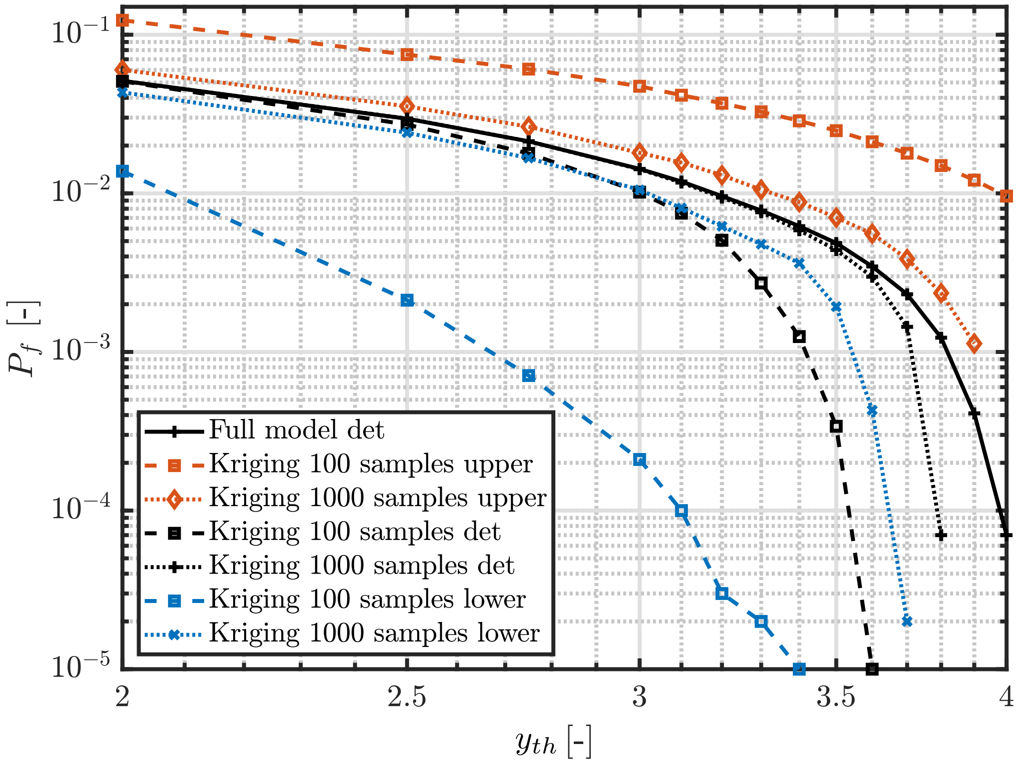

Figure 7.6: Performance of the Kriging surrogate models trained with different data sets in predicting the failure probability of Adjiman's function. For clarity, only the results of the models trained with 100 and 1000 are shown.

argument that the application of advanced Monte Carlo methods for the estimation of small failure probabilities in conjunction with non-linear limit-state functions might prove to be computationally demanding when a full numerical model is used for the prediction of P_f.

7.4.3 When is creating an IPM surrogate worthwhile?

Here, simple equations are provided to motivate the use of IPMs as surrogates for calculating bounds on P_f.

When an IPM metamodel is constructed for a deterministic performance function and the basis dimensionality, d, is chosen so that the function is modelled perfectly, such that there is no gap between the upper and lower bound of the IPM, then the entire uncertainty

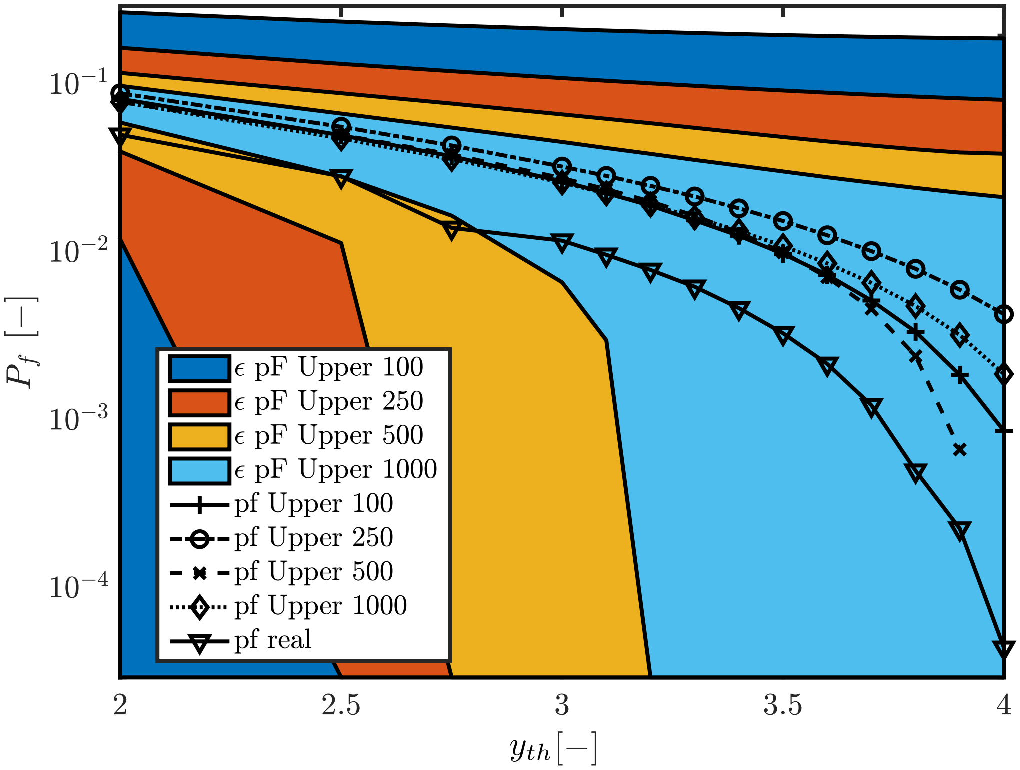

Figure 7.7: Performance of the IPM surrogate models trained with different data sets in predicting the failure probability of Adjiman's function.

in P_f is due to the uncertainty in the IPM bounds as a consequence of Eqn. 3.23.

When using a standard Monte Carlo simulation, without a metamodel, the uncertainty in the failure probability can be obtained by calculating $\overline{P}_f - \underline{P}_f$, where the bounds of the probability of failure are found by solving

$$\sum_{i=0}^{N-N_s} \binom{N}{i} (1-\underline{P}_f)^i \underline{P}_f^{N-i} = \frac{\beta}{2} \tag{7.12}$$

and

$$\sum_{i=N-N_s}^{N} \binom{N}{i} (1-\overline{P}_f)^i \overline{P}_f^{N-i} = \frac{\beta}{2} \tag{7.13}$$

which are the two-sided binomial confidence bounds on the success probability parameter for a particular confidence β [95], where N_s is the number of samples observed inside the failure region.

If the value of $\overline{P}_f - \underline{P}_f$ is larger than the IPM bound uncertainty, ϵ, obtained from Eqn. 3.23 then using an IPM metamodel is worthwhile. In the author's experience this is usually the case when d is small. For the IPM, ϵ does not depend on the probability of failure. If the performance function being analysed is less complex then the uncertainty in the calculated probability of failure will be lower, because the dimensionality of the basis, d, can be reduced. Typically, functions in high dimensional spaces must be modelled with a more complex basis.

7.5 Conclusions

In this chapter, the uncertainty in surrogate model predictions is studied in the context of failure probability estimation. This is achieved by analysing the performance of interval predictor models and Kriging on the robust estimation of small failure probabilities for non-linear models. Since intervals are used to model the uncertainty on the surrogate model estimation in addition to the propagated variability stemming from the random model parameters, the failure probability should be computed using a probability box formulation of the model response. It is shown that this problem reduces to computing two separate failure probabilities, using only a single run of model evaluations. Therefore, instead of focusing on the crisp estimate of the surrogate model to compute the probability of failure, it is suggested to take the corresponding uncertainty into account. For practical purposes, it is moreover even sufficient to consider the upper bound on the failure probability prediction. The analysis reveals that interval predictor models always provide a robust estimate of the probability of failure, yet when small training sets are considered, the bounds on the prediction may become large. This is a natural consequence of making fewer assumptions regarding the form of the function to be approximated by the metamodel.

In the following chapter, this approach is generalised to reliability analysis with probability boxes.

Chapter 8

Interval Predictor Models for Propagation of Probability Boxes

8.1 Introduction

As discussed in Chapter 4, probability boxes are useful in reliability analysis when limited data is available to model random variables. An example of such a situation in the nuclear industry is given by Prinja et al. [143], and in the aerospace industry by Patelli et al. [133]. However, the propagation of probability boxes through models is more difficult than for conventional random variables, and hence sophisticated optimisation techniques are required. However, these are often computationally expensive for black box models as a large number of evaluations of the model performance function are required to ensure convergence and guarantee a robust result. Although some metamodeling techniques have been proposed to reduce the computational demands of the analysis, which would allow the failure probability estimator to be evaluated with less computational expense, these techniques are usually dependent on the assumptions required to construct the metamodel.

In this chapter, a novel technique is proposed which uses Interval Predictor Models to propagate distributional probability boxes through black box models. This chapter generalises the contribution of Chapter 7, with a different sampling strategy and improved techniques for creating IPMs. This presents advantages over direct implementation of the double loop Monte Carlo algorithm. In Fetz [63], it was acknowledged that optimising a non-smooth function is a challenging aspect of double loop Monte Carlo simulation; by using an IPM, smooth bounds on the model response can be obtained even if the response of the model under analysis is non-continuous.

In Section 8.2.1, a novel technique is proposed to propagate distributional probability boxes through black box models, and calculate bounds on the probability of failure, P_f, of structures in the well known structural reliability analysis problem. This is achieved by using an IPM to model the performance function as a function of the aleatory variables, and then conducting Monte Carlo analysis on the identified upper and lower bounds. This is effectively a naïve double loop approach using a robust metamodel. In Section 8.2.2, a modification is proposed to the method in Section 8.2.1, where the IPM is trained on so-called *focal elements* in the aleatory space, which are obtained by brute force sampling over the epistemic variables to find the extrema of model response for each point in the aleatory space. In Section 8.2.3, a similar approach is demonstrated for the general double loop Monte Carlo propagation problem; an IPM is trained on samples of the model from a proposal distribution, and then the samples from the metamodel are re-weighted using importance sampling to find bounds on the probability of failure. Section 8.3 demonstrates the application of the developed techniques to the deflection of a cantilever beam with uncertain parameters, a non-linear oscillator, and the modal analysis of a small satellite.

8.2 Proposed approaches: obtaining bounds on the failure probability

All of the discussed approaches in this section make no assumptions about the functional form of the model, and as such the model can truly be treated as a black box — including stochastic system models with an unknown noise structure. In addition, the samples required to train the metamodels may be collected in parallel, since the model is not built sequentially or by optimising the performance function — the proposed approach relies exclusively on sampling the performance function. The proposed methods are flexible and can account for epistemic uncertainty contained inside the limit state function. For example, this could be the case if there were several feasible models but a probability could not be associated with each model. This is a feature which, to the best of the authors' knowledge, no existing methods of this type can deal with. Interval failure thresholds can be used with either algorithm without an increase in the number of evaluations of the full model, since the indicator function must be monotonic with respect to changes in the threshold used to define the limit state function.

8.2.1 Approach 1: metamodels for naïve double loop approach

In this approach, the Naïve double loop Monte Carlo approach is used, as described in Chapter 4. To recap, the Naïve Double Loop Monte Carlo algorithm is shown in Algorithm 1. In this example it is assumed that $g(\boldsymbol{x})$ has been transformed with a Copula, so that its aleatory inputs may be sampled from uniform distributions (the function $\boldsymbol{x}_i = T(\boldsymbol{\alpha}_j, \boldsymbol{\theta}_i)$ will be used to map between the epistemic and aleatory spaces and the true system variables).

Algorithm 1 The Naïve Double Loop Monte Carlo Algorithm (once Copula has been applied).

```
for $i = 1, .., N_e$ do
  Sample one realisation of epistemic parameters $\boldsymbol{\theta}_i$, from a uniform distribution
  for $j = 1, .., N_a$ do
    Sample one realisation of aleatory parameters $\boldsymbol{\alpha}_j$, from a uniform distribution
    $g_{ij} = g(T(\boldsymbol{\alpha}_j, \boldsymbol{\theta}_i))$
  end for
  $E_i = \frac{1}{N_a} \sum_{j=1}^{N_a} g_{ij}$
end for
$\underline{E} = \min_i (E_i)$ and $\overline{E} = \max_i (E_i)$
```

It is assumed that the system variables can be written as a function of separate epistemic uncertain variables falling in the unit hyper-rectangle, $\boldsymbol{\theta}$, and aleatory uncertain variables, $\boldsymbol{\alpha}$, which are uniformly distributed between 0 and 1. N samples are drawn of $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$, both from a uniform distribution between 0 and 1. Samples of the performance function can then be calculated by transforming the aleatory and epistemic variables into the actual variables of the problem. Then an IPM is trained with the aleatory variables as inputs and the performance function as the output, i.e. Eqn. 3.9 is solved whilst replacing $\boldsymbol{x}_i$ with $\boldsymbol{\alpha}_i$ and $y(\boldsymbol{x}_i)$ with $g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta}_i))$. This IPM is an uncertain model of the performance function as a function of the aleatory input variables; the epistemic uncertainties are now represented as the uncertainty in the IPM. The upper and lower bounds on the failure probability can then be calculated with minimal computational expense by performing Monte Carlo simulation on the polynomial upper and lower bounds of the performance function from the IPM, by sampling $\boldsymbol{\alpha}$ uniformly between 0 and 1 and then calculating $\overline{g}(\boldsymbol{\alpha})$ and $\underline{g}(\boldsymbol{\alpha})$ for the samples. A diagram of the algorithm is shown in Figure 8.1.

In contrast to the random set approach applied to the propagation of distribution-free probability boxes in the multi-level metamodel algorithm [158], the approach proposed in

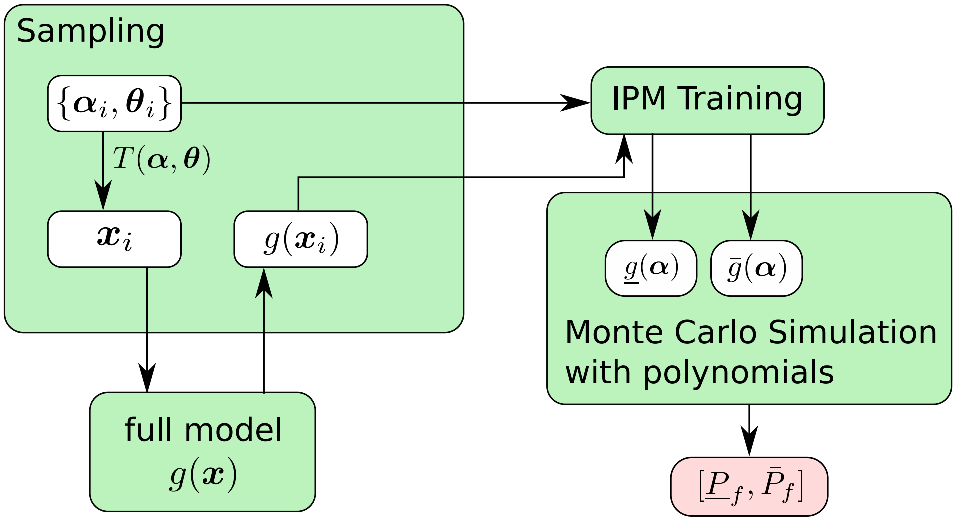
Sampling
$\{\boldsymbol{\alpha}_i, \boldsymbol{\theta}_i\}$
$T(\boldsymbol{\alpha}, \boldsymbol{\theta})$
$\boldsymbol{x}_i$
$g(\boldsymbol{x}_i)$
full model
$g(\boldsymbol{x})$
IPM Training
$\underline{g}(\boldsymbol{\alpha})$
$\bar{g}(\boldsymbol{\alpha})$
Monte Carlo Simulation
with polynomials
$[\underline{P}_f, \bar{P}_f]$

Figure 8.1: Approach 1: Diagram of algorithm to obtain bounds P_f by constructing metamodels for Naïve double loop approach), by modelling the performance function in the aleatory space.

this chapter does not require multiple levels of metamodeling, since one IPM is sufficient to obtain both the upper and lower bound of the performance function. Therefore the algorithm proposed in this chapter is effectively a single loop approach, as the optimisation takes place during the creation of the metamodel. If the approach is applied to distributional probability boxes then the bounds will not be tight, since random set theory based approaches overestimate the bounds on P_f when applied to problems with distributional probability boxes [5].

8.2.2 Approach 2: IPMs trained on propagated focal elements

A focal element $\boldsymbol{\alpha}_i$ is simply a sampled interval from a probability box. When N_a samples of i are made, for each $\boldsymbol{\alpha}_i$ the corresponding focal element becomes $[\min_{\boldsymbol{\theta}\in\Theta} T(\boldsymbol{\alpha}_i, \boldsymbol{\theta}), \max_{\boldsymbol{\theta}\in\Theta} T(\boldsymbol{\alpha}_i, \boldsymbol{\theta})]$. When this is propagated through $g(\boldsymbol{x})$ the associated output will be the interval $[\min_{\boldsymbol{\theta}\in\Theta} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta})), \max_{\boldsymbol{\theta}\in\Theta} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta}))])$. It is trivial to train an IPM with inputs $\boldsymbol{\alpha}_i$, and output $[\min_{\boldsymbol{\theta}\in\Theta} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta})), \max_{\boldsymbol{\theta}\in\Theta} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta}))]$. However, obtaining the focal elements for training is more expensive than propagating single values of $\boldsymbol{x}$, since the maximisation over $\boldsymbol{\theta}$ requires multiple evaluations of $g(\boldsymbol{x})$. If analytic gradients of the performance function are available then approximate focal elements can

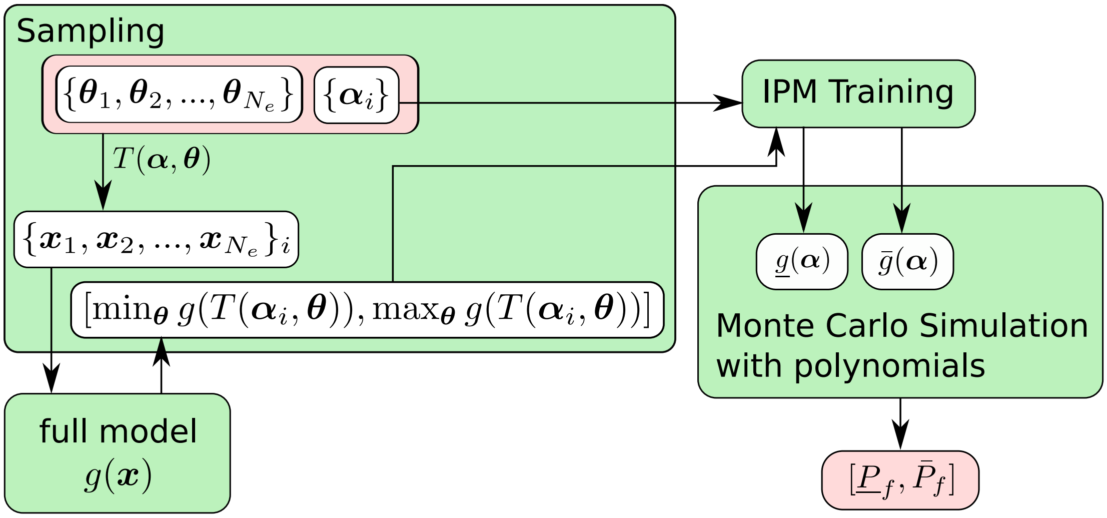
Sampling
$\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, ..., \boldsymbol{\theta}_{N_e}\}$
$\{\boldsymbol{\alpha}_i\}$
$T(\boldsymbol{\alpha}, \boldsymbol{\theta})$
$\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_{N_e}\}_i$
$[\min_{\boldsymbol{\theta}} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta})), \max_{\boldsymbol{\theta}} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta}))]$
full model
$g(\boldsymbol{x})$
IPM Training
$\underline{g}(\boldsymbol{\alpha})$
$\bar{g}(\boldsymbol{\alpha})$
Monte Carlo Simulation with polynomials
$[\underline{P}_f, \bar{P}_f]$

Figure 8.2: Approach 2: Diagram of algorithm to obtain bounds P_f by constructing metamodels for focal element propagation, by modelling the performance function in the aleatory space.

be obtained in a reduced computational time by using a Taylor series model. In this chapter we take a more general approach by obtaining focal elements for training by approximating $[\min_{\boldsymbol{\theta}\in\Theta} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta})), \max_{\boldsymbol{\theta}\in\Theta} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta}))]$ with brute force sampling, i.e. $[\min_{\boldsymbol{\theta}_i\in\{\boldsymbol{\theta}_1,...,\boldsymbol{\theta}_{N_e}\}} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta}_i)), \max_{\boldsymbol{\theta}_i\in\{\boldsymbol{\theta}_1,...,\boldsymbol{\theta}_{N_e}\}} g(T(\boldsymbol{\alpha}_i, \boldsymbol{\theta}_i))]$, where $\{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_{N_e}\}$ are sampled by imposing a uniform distribution on the unit hypercube. Latin Hybercube Sampling or Sobol Sequence Sampling could also be used to sample $\boldsymbol{\theta}_i$. Then, once the IPM has been trained, the estimation of P_f proceeds in the same way as approach 1 (independent sampling), by numerically integrating the IPM bounds (see Section 8.2.1). A diagram of the algorithm is shown in Figure 8.2.

8.2.3 Approach 3: metamodels for non-naïve approach

It is also possible to directly construct an IPM metamodel of $g(\boldsymbol{x})$, which can then be used to find bounds on P_f. The metamodel should be constructed by collecting samples of $g(\boldsymbol{x})$, by sampling $\boldsymbol{x}$ from a composite distribution which has standard deviation roughly equal to the spread of the probability box. Then $\overline{P}_f$ can be obtained from Monte Carlo simulation on $\underline{g}(\boldsymbol{x})$, and vice versa. The samples used for Monte Carlo simulation of $\overline{P}_f$ should be drawn from the proposal distribution and then re-weighted using importance sampling, as

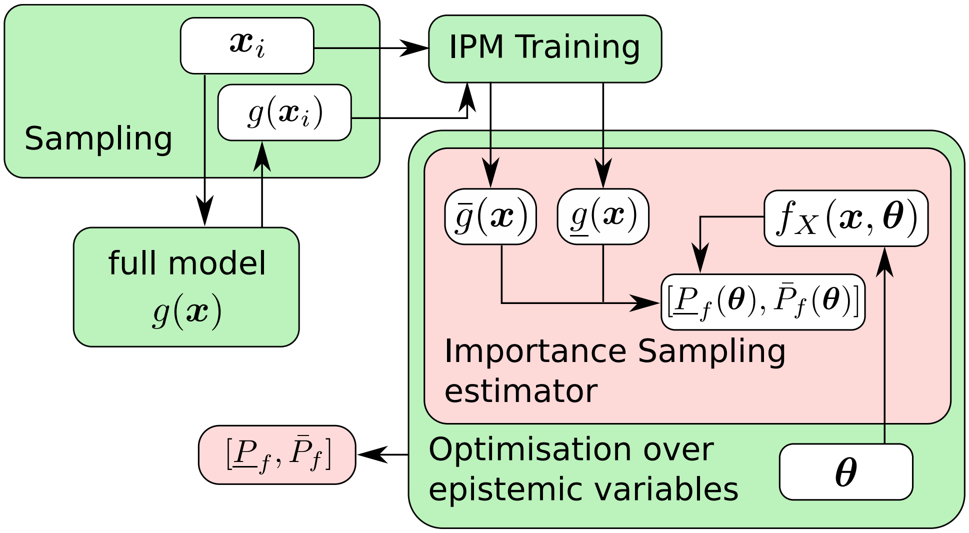
$\boldsymbol{x}_i$
IPM Training
$g(\boldsymbol{x}_i)$
Sampling
full model
$g(\boldsymbol{x})$
$\bar{g}(\boldsymbol{x})$
$\underline{g}(\boldsymbol{x})$
$f_X(\boldsymbol{x}, \boldsymbol{\theta})$
$[\underline{P}_f(\boldsymbol{\theta}), \bar{P}_f(\boldsymbol{\theta})]$
Importance Sampling estimator
$[\underline{P}_f, \bar{P}_f]$
Optimisation over epistemic variables
$\boldsymbol{\theta}$

Figure 8.3: Approach 3: Diagram of algorithm to obtain bounds P_f by constructing metamodels for non-naïve approach, by applying importance sampling to the metamodel. Using the IPM, the importance sampling estimator produces bounds on the failure probability for a particular $\boldsymbol{\theta}$, which can be optimised over $\boldsymbol{\theta}$ to yield the true bounds on the failure probability.

proposed in Fetz [63]. Therefore the upper bound on the failure probability can be obtained by evaluating

$$\overline{P}_f = \max_{\boldsymbol{\theta} \in \Theta} \int \overline{I}_f(\boldsymbol{x}) \frac{f_X(\boldsymbol{x}, \boldsymbol{\theta})}{h_X(\boldsymbol{x})} h_X(\boldsymbol{x}) d\boldsymbol{x}, \tag{8.1}$$

where $h_X(\boldsymbol{x})$ is the proposal distribution, which is used to generate training samples for the IPM from the full model, $\overline{I}_f(\boldsymbol{x})$ is the upper bound on the indicator function obtained from the IPM (returning 1 when $\underline{g}(\boldsymbol{x}) \leq 0$, and 0 otherwise), and $f_X(\boldsymbol{x}, \boldsymbol{\theta})$ is a particular distribution contained by the distributional probability box. A diagram of the algorithm is shown in Figure 8.3.

The IPM metamodel is a useful addition to vanilla importance sampling, because now the optimisation in Eqn. 8.1 is being performed on a continuous function, even if the performance function used is not smooth, or if a set of performance functions are being analysed. Troffaes [173] shows that importance sampling results in a consistent estimator when the failure probability is continuous in the epistemic uncertain parameters.

8.2.4 Confidence bounds on failure probability

In each of the proposed algorithms the interpretation of the confidence-reliability plot (described in Section 3.3.2) is different. To recap, when constructing an IPM, the confidence-reliability plot is interpreted as displaying the Neyman confidence bound on the probability that an unseen sample from the performance function falls inside the IPM bounds. The complement of the probability that an unseen sample from the performance function falls inside the IPM bounds is known as the IPM bound violation probability. To reiterate Section 3.3.2, the confidence-reliability plot is usually closely related to the number of training examples used to create the IPM and other properties of the IPM and data generating process.

When only aleatory uncertainty is present in the system inputs, i.e. random variables are used as opposed to probability boxes, the violation probability of the IPM is the maximum possible uncertainty in the obtained bounds on the failure probability. This is the interpretation of the confidence-reliability plot for re-weighting (*Approach 3*) — the confidence bound applies to the Monte Carlo estimator for the failure probability at the proposal distribution. For focal element propagation (*Approach 2*), if the number of aleatory samples is N_a, then obtaining the confidence reliability plot using N_a only measures confidence in the propagation of aleatory uncertainty, i.e. that the next sampled focal element from the performance function will fall outside the IPM bounds. This is because the IPM's training constraints become set inclusion relations for the focal elements (this reliability will be known as R_a).

By setting $d = 2$ and setting the number of samples equal to the number of epistemic samples ($N = N_e$), one can obtain a confidence-reliability plot relating to the brute force propagation of the focal elements in *Approach 2*. In this case the bound violation probability refers to the probability that the next sample in the brute force optimisation of the epistemic space falls outside the bounds. Alternatively, it is well known that the maximum and minimum of a sample can be used to produce a prediction interval with reliability $\frac{N_e-1}{N_e+1}$ [183], and this is the approach that will be used to determine the epistemic propagation reliability (R_e) for focal element propagation (*Approach 2*).

When the probability box is sampled in *Approach 1*, i.e. there are a mix of epistemic and aleatory variables sampled independently, the reliability of the IPM is not associated solely with epistemic or aleatory uncertainty propagation, but rather a hybrid of the two. However, it is still clear that the obtained bounds become more trustworthy when the reliability of the IPM improves.

All reliabilities (R) quoted in the numerical examples for this chapter are obtained by finding the reliability where the confidence is greater than 0.999 (i.e. $\beta = 0.001$).

8.3 Numerical examples

8.3.1 Cantilever beam

Problem description

For the simple example of a cantilever beam with a point load, F, at any point on the beam, the maximum deflection of the end of the beam is given by

$$\delta_{max} = \frac{Fa^2}{6EI}(3l - a) \tag{8.2}$$

where I is the moment of inertia of the beam, a is the distance of the point load from the fixed end of the beam, l is the length of the beam and E is the modulus of elasticity of the beam [79]. E, I and a were fixed, and l and F were given by distributional probability boxes with normal distributions and uncertain means (Case A). The chosen values of the parameters are shown in Table 8.1. In order to demonstrate the application of Approach 1 (independent sampling) to a problem where epistemic uncertainty is more influential, the analysis was repeated for a modified set of inputs with epistemic uncertainty in the standard deviation of the random variables (Case B). It was assumed that the beam 'fails' when the maximum deflection is greater than 35 mm.

The independent sampling method described in Section 8.2.1 for the naïve double loop approach was used to find the probability of failure of the system, by creating a polynomial IPM of maximum degree 1 ($\phi(\alpha_l, \alpha_F) = [1, \alpha_l, \alpha_F, \alpha_l\alpha_F]$) of the performance function. In total 1000 samples of the true model were made. The focal element propagation method in Section 8.2.2 was applied with the same basis.

This was compared to the re-weighting approach described in Section 8.2.3. A normal proposal distribution was used with $\mu_F = 30500$, $\sigma_F = 230$, $\mu_l = 5050$, $\sigma_l = 230$. Again, a polynomial IPM of maximum degree 1 ($\phi(l, F) = [1, l, F, lF]$) was trained with 1000 samples. The bounds on the probability of failure were obtained by using MATLAB's `fmincon` function on the failure probability re-weighted estimator.

The confidence-reliability analysis was then performed using Eqn. 3.23.

Variable	Distribution	Mean	Standard Deviation (Case A)	Standard Deviation (Case B)
E	Fixed	200000 N/mm^2	N/A	N/A
I	Fixed	78125000 mm^4	N/A	N/A
l	Normal	[5000, 5100] mm	200 mm	[200, 220] mm
a	Fixed	3000 mm	N/A	N/A
F	Normal	[30000, 31000] N	200 N	[200, 220] N

Table 8.1: Values of input variables for cantilever beam problem.

Results

The metamodel for the performance function with a polynomial IPM of maximum degree 1 is shown in Figure 8.4. The reference solution ($P_f = [0.40, 0.81]$) was obtained by naïve Monte Carlo simulation with a large number of samples. Figure 8.6 shows the confidence-reliability analysis (calculated with Eqn. 3.23) for the calculation of P_f using the IPM in Figure 8.4, corresponding to a reliability of approximately 0.98 with high confidence (0.999). This IPM has 6 support constraints and hence the bound on R is fairly tight. Inverting Eqn. 3.23, it was calculated that 19619 samples would be required to obtain $\beta = 0.001$ and $\epsilon = 0.001$. Figure 8.5 shows the bounds of the CDF which were computed by Monte Carlo analysis of the performance functions shown in Figure 8.4. The bounds on P_f can be tightened by increasing the number of training samples which allows the degree of the IPM to be increased without decreasing the bound on R. For example, by taking 2000 samples and using the loss function in Eqn. 7.10, $P_f = [0.388, 0.795]$ for a polynomial IPM with basis $\phi(\alpha_l, \alpha_F) = [1, \alpha_l, \alpha_F, \alpha_l\alpha_F, \alpha_l^2, \alpha_F^2]$, where $R > 0.987$ with high confidence (0.999). When using the loss function in Eqn. 7.7, no notable improvement was found in the bounds on P_f, since the performance function and IPM were relatively simple, and the epistemic uncertainty in P_f is large. Figure 8.9 shows the bounds of the CDF which were computed by Monte Carlo analysis of the obtained IPM, when using the input Case B. The reference solution computed with Double Loop Monte Carlo simulation was $P_f = [0.40, 0.81]$.

The re-weighting strategy with the direct IPM metamodel (shown in Figure 8.7) directly obtains the reference solution. Since the direct IPM is a model of a linear function it can be easily represented exactly by the IPM, and hence there are no support constraints. This allows us to bound the reliability of the IPM to at least 0.99 with high confidence ($\beta = 0.001$) by using the wait and judge approach. For similar reasons, the correct solution can be obtained with as few as 10 samples using the re-weighting estimator. Figure 8.8 shows the confidence-reliability analysis for the calculation of P_f using Figure 8.7.

A summary of the results for this example are given in Table 8.2.

Approach	N_{Samples}	Case A	Case B
Reference Solution, Double Loop Monte Carlo	$N_a = 10^3$, $N_e = 10^6$ $N_e \times N_a = 10^9$	$P_f = [0.4, 0.81]$	$P_f = [0.4, 0.81]$
Approach 1 (independent sampling, IPM degree 1)	1000	$P_f = [0.36, 0.81]$, $R \geq 0.98$	$P_f = [0.36, 0.81]$, $R \geq 0.98$
Approach 1 (independent sampling, IPM degree 2)	2000	$P_f = [0.39, 0.80]$, $R \geq 0.99$	$P_f = [0.39, 0.80]$, $R \geq 0.99$
Approach 1 (independent sampling, IPM degree 1)	300	$P_f = [0.39, 0.81]$, $R \geq 0.94$	$P_f = [0.38, 0.79]$, $R \geq 0.94$
Approach 2 (focal element propagation, IPM degree 1)	1000 ($N_a = 125$, $N_e = 8$)	$P_f = [0.375, 0.795]$ $R_{\text{a}} > 0.85$, $R_{\text{e}} \approx 0.78$	$P_f = [0.373, 0.785]$ $R_{\text{a}} > 0.85$, $R_{\text{e}} \approx 0.78$
Approach 3 (re-weighting, IPM degree 1)	1000	$P_f = [0.4, 0.81]$, $R \geq 0.99$	$P_f = [0.4, 0.81]$, $R \geq 0.99$
Approach 3 (re-weighting, IPM degree 1)	10	$P_f = [0.4, 0.81]$, $R \geq 0.021$	$P_f = [0.4, 0.81]$, $R \geq 0.021$

Table 8.2: Summary of results for cantilever beam reliability analysis in Section 8.3.1.

8.3.2 Dynamic response of a non-linear oscillator

Problem description

In order to demonstrate the application of the method on a non-linear performance function, the well known non-linear oscillator example is used [56] [84]. The performance function is defined by

$$g_{\text{oscillator}}(C_1, C_2, M, R, T_1, F_1) = 3R - \left| \frac{2F_1}{M\omega_0^2} \sin(\frac{\omega_0 T_1}{2}) \right|, \tag{8.3}$$

where the natural frequency of the oscillator, $\omega_0 = \sqrt{\frac{C_1+C_2}{M}}$, M is the mass, C_1 and C_2 are the spring constants of the primary and secondary springs, R is the displacement at which the secondary spring yields, t_1 is the duration of the loading, and F_1 is the amplitude of the applied force. As usual, the system fails when $g_{\text{oscillator}} \leq 0$, hence the failure probability to be estimated is $P_f = \mathbb{P}(g_{\text{oscillator}} \leq 0)$. The distributions and probability boxes assigned to

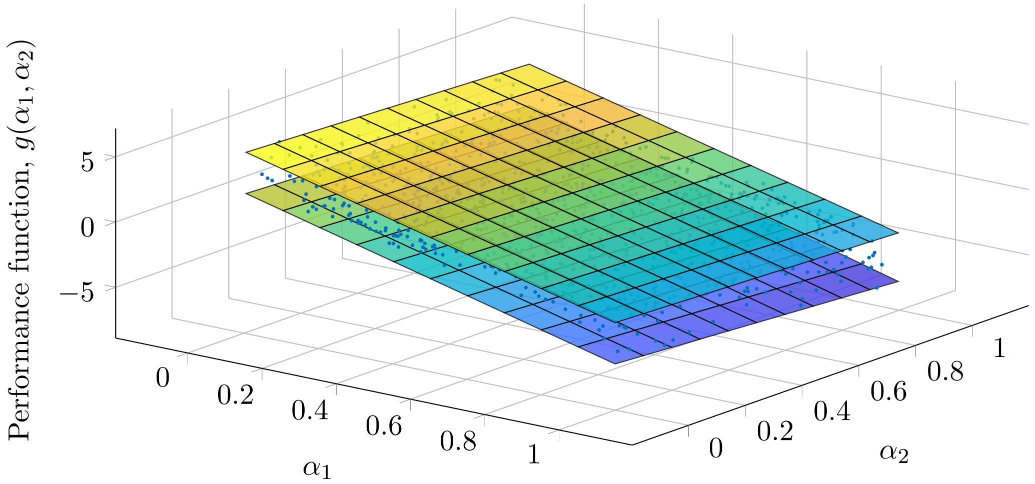

Figure 8.4: Degree 1 IPM of the performance function in the aleatory space for the cantilever beam, created by independently sampling the aleatory and epistemic variables (Approach 1).

the inputs are listed in Table 8.3. A diagram of the system is shown in Figure 8.10.

The methods described in Section 8.2 for the naïve double loop approach were used to find the probability of failure of the system, by creating a polynomial IPM of maximum degree 1 of the performance function. In total, 1000 samples of the true model were made.

This was compared to the approach described in Section 8.2.3, with a normal proposal distribution with mean at the centre of the probability box and standard deviation set to cover the support of the probability box ($\mu_{\text{proposal}} = \frac{\overline{\mu}+\underline{\mu}}{2}$ and $\sigma_{\text{proposal}} = \sqrt{(\frac{\overline{\mu}-\underline{\mu}}{2\times 3})^2 + \sigma^2}$). Again, a polynomial IPM of maximum degree 1 was trained with 1000 samples. The bounds on the probability of failure were obtained by using MATLAB's `fmincon` function on the failure probability re-weighted estimator. The performance function loss (Eqn. 7.7) was not used in this example, as it was not found to significantly improve the performance of the model.

Results

The results for the analysis of the oscillator are shown in Table 8.4, including number of support constraints is shown for each trained IPM, and a bound on the reliability computed as described in Section 3.24. The reference solution was computed with naïve double loop

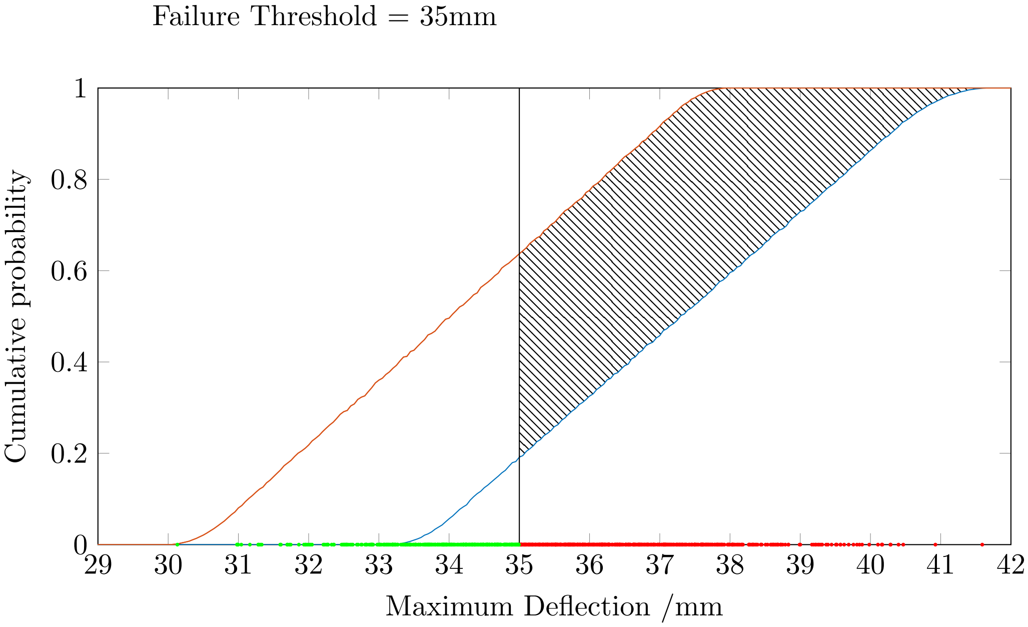

Figure 8.5: CDF bounds obtained by Monte Carlo analysis on the performance function modelled in Figure 8.4. The ‘flat’ bounds are a remnant of the low degree IPM chosen to represent the performance function of the cantilever beam. Training samples are shown on the abscissa axis.

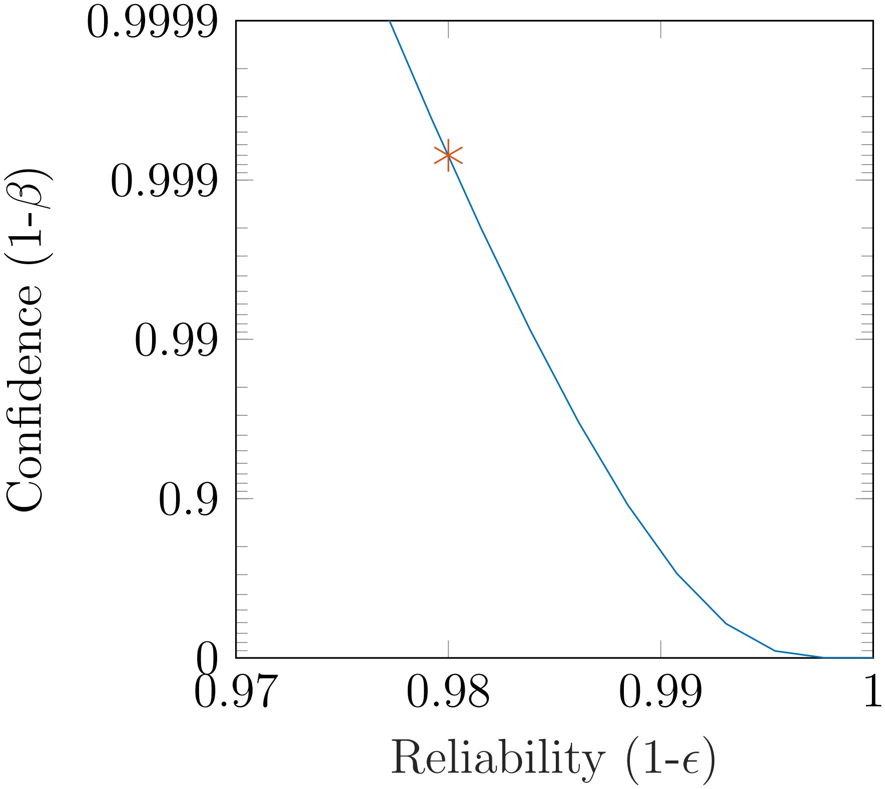

Figure 8.6: Confidence-reliability plot corresponding to the IPM used to model the performance function in Figure 8.4 for the cantilever beam and calculate P_f (as described in Section 3.3.2). This plot corresponds to a reliability of approximately 0.98 with confidence 0.999, which is shown on the plot as a star.

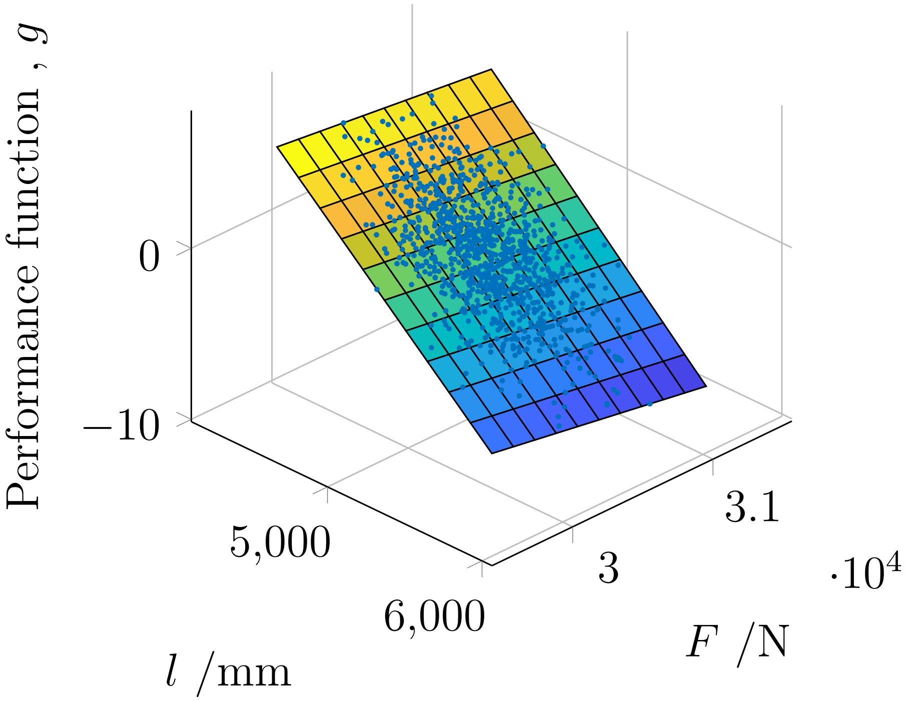

Figure 8.7: Direct degree 1 IPM of the performance function for the cantilever beam, created for the re-weighting approach (Approach 3).

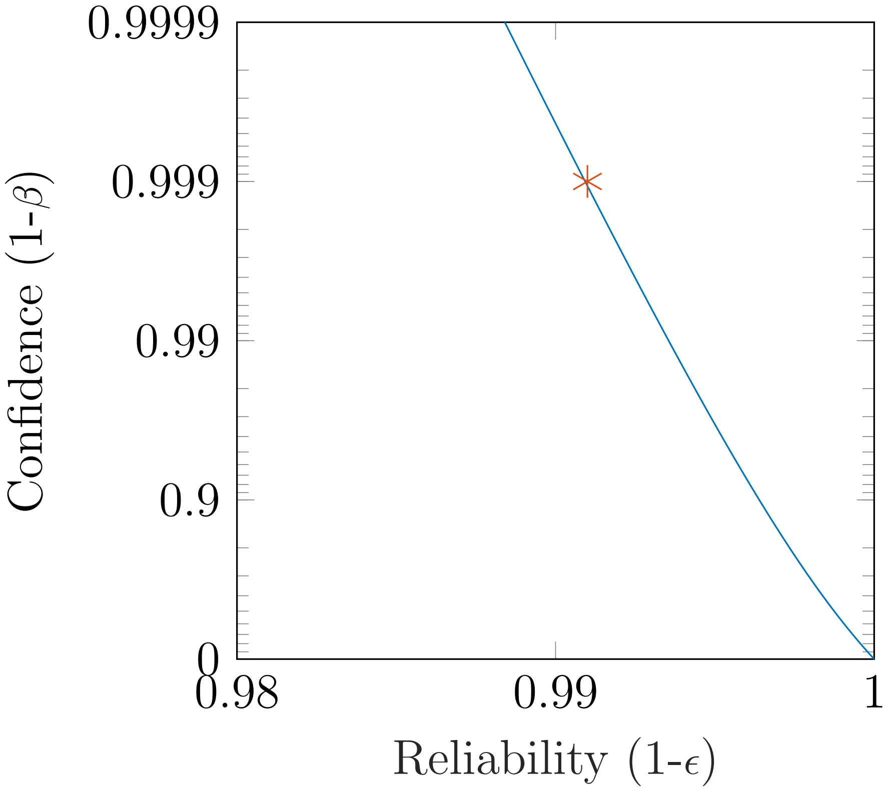

Figure 8.8: Confidence-reliability plot corresponding to the IPM used to model the performance function in Figure 8.7 for the cantilever beam and calculate P_f. This plot corresponds to a reliability of over 0.99 with confidence 0.999, which is shown on the plot as a star.

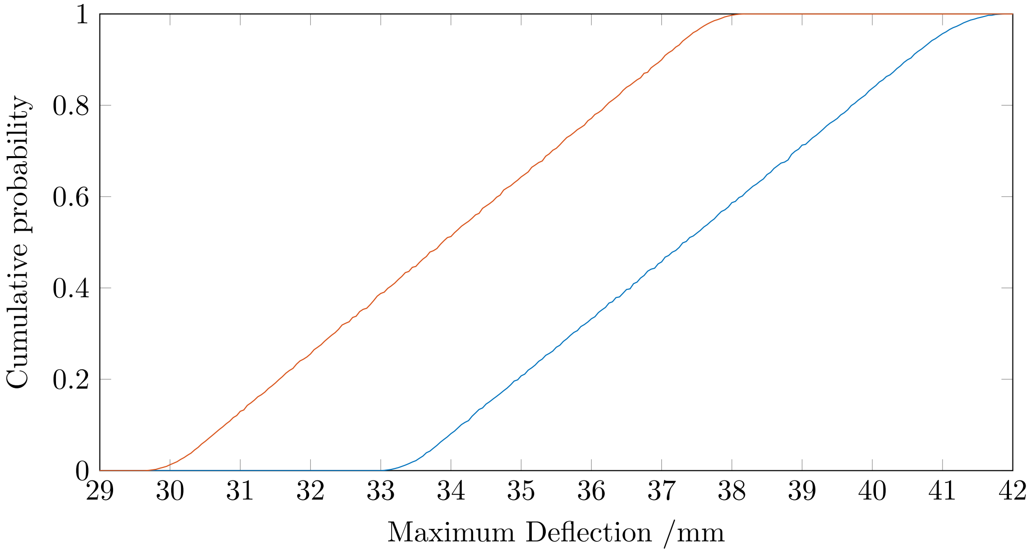

Figure 8.9: CDF bounds obtained by Monte Carlo analysis on the performance function modelled the IPM for the second cantilever beam input set. The 'flat' bounds are a remnant of the low degree IPM chosen to represent the performance function of the cantilever beam.

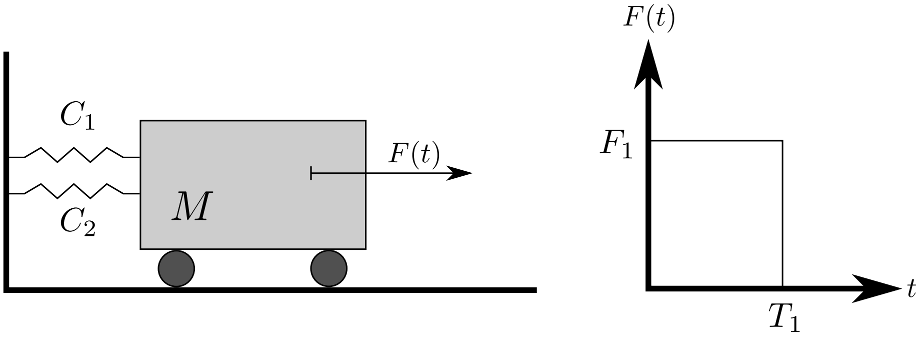

Figure 8.10: A non-linear oscillator.

Variable	Distribution	Mean	Standard Deviation
C_1	Normal	1	0.1
C_2	Normal	0.1	0.01
R	Normal	[0.45, 0.5]	0.05
M	Normal	1	0.05
t_1	Normal	[0.95, 1]	0.2
F_1	Normal	[0.95, 1]	0.2

Table 8.3: Values of input variables for non-linear oscillator.

Monte Carlo simulation, using 10000 inner loop and 10000 outer loop samples, resulting in a total of 10^{10} model queries.

Approach	N_{Samples}	$[\overline{P}_f, \underline{P}_f]$	Confidence	s_N^*
Reference solution (double loop Monte Carlo)	10^{10}	$[0.0132, 0.0712]$	-	-
Approach 1 (independent sampling, IPM degree 3)	1000	$[0.0138, 0.0741]$	$R \geq 0.80$, $R^* > 0.87$	90
Approach 1 (independent sampling, IPM degree 2)	1000	$[0.0073, 0.123]$	$R \geq 0.92$, $R^* > 0.94$	33
Approach 2 (focal elements, IPM degree 2)	1000	$[0.012, 0.11]$	$R_{\text{a}} > 0.62$, $R_{\text{e}} \approx 0.67$, $R_{\text{a}}^* > 0.76$	34
Approach 3 (re-weighting, IPM degree 2)	1000	$[0.0735, 0.0114]$	$R \geq 0.92$, $R^* > 0.94$	33

Table 8.4: Summary of results for non-linear oscillator (S_N^*: maximum support constraints,R: Confidence a priori; R^*: confidence wait and judge).

8.3.3 Small satellite

Problem description

In this section, the developed techniques are applied to a NASTRAN model of a satellite with 900 QUAD4 elements ($\sim$ 5300 DOF) [129]. The model is available in the OpenCossan software [132]. All DOF of the nodes at the bottom of the nozzle have been constrained by modelling the boundary conditions that the nozzle is attached to the supporting structure with bolts, and a vertical acceleration of $6g$ has been applied together with a horizontal acceleration of $1g$ ($g = 9.81\text{m/s}^2$). The structure consists of 4 components, namely nozzle, upper and lower panels, central cylinder and the vertical panels. The combined effect of the uncertainty in the young's modulus and density of each of these components on the second natural frequency is investigated, and epistemic uncertainties in these two quantities as shown in Table 8.5 are considered.

Variable	Distribution	Mean	Standard Deviation
Young's Modulus × 4	Normal	[65,75] GPa	1.05 GPa
Density × 4	Normal	[2500,2900] kg/m^3	270 kg/m^3

Table 8.5: Summary of the 8 random inputs for Satellite Model. Both of the random variables shown above are repeated for the 4 structural components of the model.

The method described in Section 8.2.1 for the naïve double loop approach was used to find bounds on the CDF for the second eigenvalue and also bounds on the expectation of the second eigenvalue, by taking 1000 samples from the full model. In order to achieve this, the method was modified to build an IPM for the response of the model rather than the performance function, which is required when calculating expectations rather than probabilities of failure. Approach 2 (Section 8.2.2) was used to train an IPM on focal elements propagated with brute force optimisation, making 4 samples in the brute force optimisation and 250 aleatory samples (samples of $\boldsymbol{\alpha}$). An IPM with a polynomial basis of maximum degree 1 was used.

To obtain a reference solution, the double loop Monte Carlo method was used with 50 inner loop Monte Carlo samples, and 100 outer loop Bayesian Optimisation evaluations made by MATLAB's `bayesopt` routine, for both the upper and lower bound (i.e. $2 \times 100 \times 50 = 10000$ samples in total).

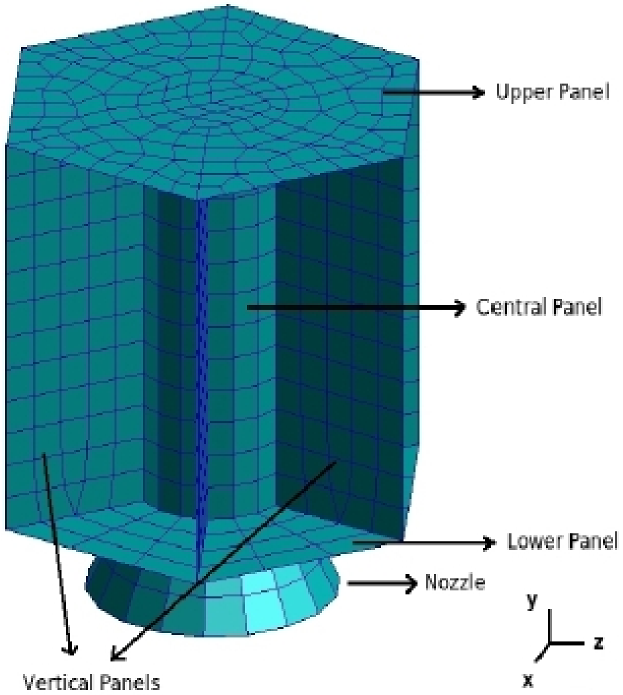

Figure 8.11: Small Satellite Model in NASTRAN. Full details of model available from Panayirci [129].

Results

For Approach 1 (independent sampling) and Approach 2 (focal element propagation), bounds on the CDF of the output are shown in Figure 8.12. Figure 8.13 shows the confidence-reliability analysis for the IPM, corresponding to a reliability of approximately 0.97 with high confidence. Increasing the maximum degree of the IPM to 2 tightens the prediction interval, however the reliability of the bounds is decreased. Similar results were obtained by using a radial basis, trained using with the same number of terms. Applying Approach 3 (re-weighting), with a normal proposal distribution with mean at the centre of the probability box and standard deviation set to cover the support of the probability box ($\mu_{\text{proposal}} = \frac{\overline{\mu}+\underline{\mu}}{2}$ and $\sigma_{\text{proposal}} = \sqrt{(\frac{\overline{\mu}-\underline{\mu}}{2\times 3})^2 + \sigma^2)}$), similar bounds were obtained on the expectation of the 2nd eigenvalue. A summary of the results for this example are given in Table 8.6. Some of the results appear overly conservative, indicating the IPM is a poor fit for the model response. However, Approach 2 (IPM for focal element propagation) has impressive agreement with the reference solution. As expected, the IPM with Maximum Degree 2 has a lower reliability, and hence underestimates the upper bound of the expectation.

Computing the reference solution resulted in bounds on the expectation of the 2nd eigenvalue of $\mathbb{E}_{\boldsymbol{x}} = [359, 481]$. For comparison, a double loop method with Latin Hyper Cube sampling for both loops (with 40 inner loop samples and 25 outer loop samples, resulting in 1000 total samples), was found to underestimate the interval width, $\mathbb{E}_{\boldsymbol{x}} = [367, 447]$. Therefore, the authors do not recommend the use of Latin Hypercube Sampling, since in probability bounds analysis it is desirable to find the outer approximation of the interval containing the results, in order to be conservative in an engineering sense.

8.4 Chapter summary

In this chapter, a computational method of bounding the reliability of the propagation of epistemic uncertainty was proposed. Novel loss functions were introduced to ensure tightness when IPMs are created from data representing performance functions. The approach proposed in this chapter is applicable to the double loop Monte Carlo algorithm as well as the naïve approach, where an uninformative distribution is sampled rather than using optimisation to propagate intervals. A key benefit is that the performance function is smoothed, which enables easier optimisation of the probability of failure. Both of the

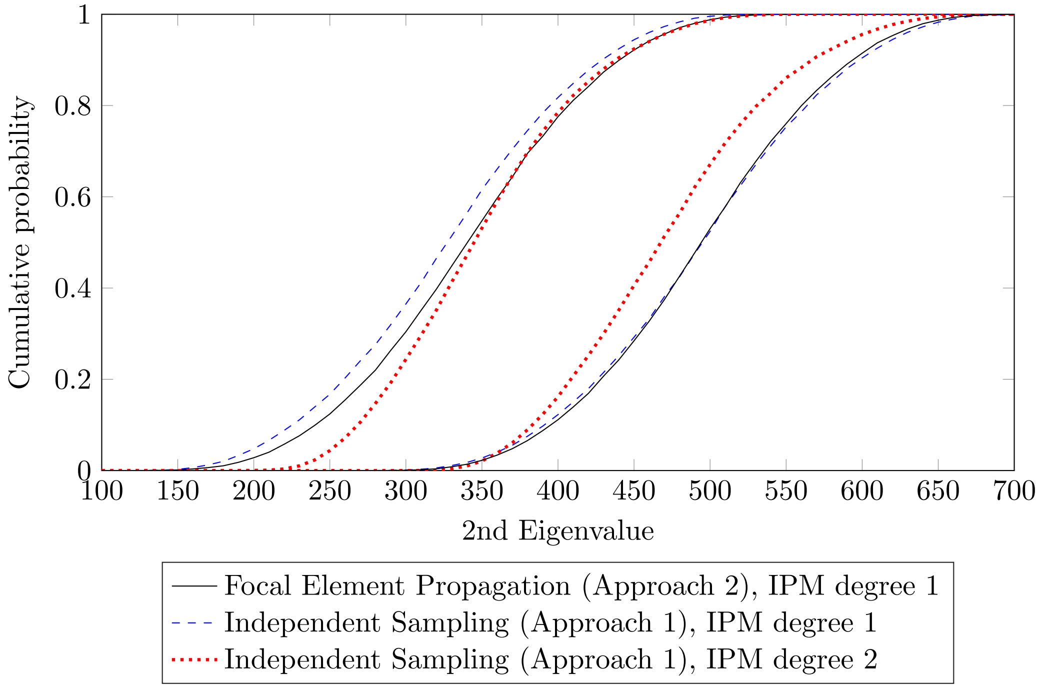

Figure 8.12: CDF bounds obtained by Monte Carlo analysis on an IPM for the 2nd Eigenvalue of a small satellite (modal analysis).

Approach	N_{samples}	[$\underline{\mathbb{E}}_{\boldsymbol{x}}$, $\overline{\mathbb{E}}_{\boldsymbol{x}}$]	Confidence
Reference (Double Loop Monte Carlo with Bayesian Optimisation)	10000	$[359, 481]$	N/A
Latin Hypercube Double Loop	1000	$[367, 447]$	N/A
Approach 1 (independent sampling, IPM degree 1)	1000	$[327, 495]$	$R \geq 0.97$
Approach 1 (independent sampling, IPM degree 2)	2000	$[349, 471]$	$R \geq 0.88$
Approach 2 (focal element, IPM degree 2)	1000 ($N_a = 250$, $N_e = 4$)	[352, 471]	$R_\text{a} > 0.55$, $R_\text{e} \approx 0.6$
Approach 2 (focal element, IPM degree 1)	1000 ($N_a = 200$, $N_e = 5$)	$[346, 489]$	$R_\text{a} > 0.84$, $R_\text{e} \approx 0.67$
Approach 2 (focal element, IPM degree 1)	1000 ($N_a = 40$, $N_e = 25$)	$[353, 485]$	$R_\text{a} > 0.33$, $R_\text{e} \approx 0.92$
Approach 2 (focal element, IPM degree 1)	1000 ($N_a = 125$, $N_e = 8$)	$[339, 487]$	$R_\text{a} > 0.746$, $R_\text{e} \approx 0.78$
Approach 3 (re-weighting, IPM degree 1)	1000	$[321, 477]$	$R \geq 0.966$

Table 8.6: Summary of results for small satellite model using different approaches.

proposed approaches do not make restrictive assumptions about the functional form of the model response, and are easily parallelisable.

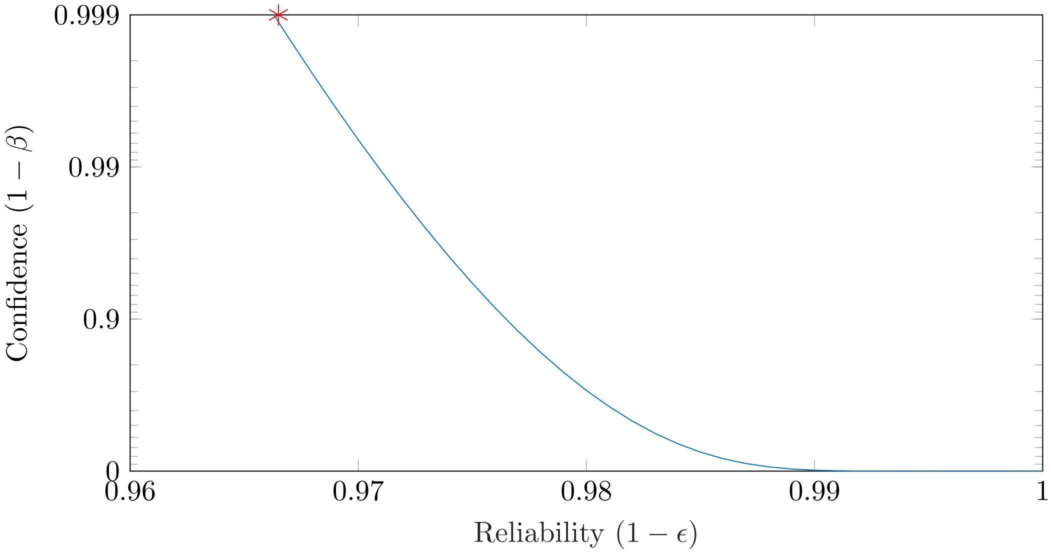

Figure 8.13: Confidence-reliability plot corresponding to the IPM of the 2nd Eigenvalue of small satellite modal analysis. This plot corresponds to a reliability of over 0.97 with confidence 0.999, which is shown on the plot as a star.

Chapter 9

Interval Neural Networks

9.1 Introduction

In recent years, deep learning using artificial neural networks has emerged as a generalised machine learning tool which has revolutionised supervised learning, reinforcement learning, as well as finding many applications in the field of engineering, most often as efficient surrogates for large models [171]. In all fields, but particularly in safety critical engineering applications, it is essential to quantify the uncertainty of the neural network, as discussed in Chapter 3. The simplest approaches attempt to quantify this uncertainty by analysing the mean squared error or explained variance (r^2) of the neural network on a test set. However, these approaches do not robustly bound the predictions of the neural network. Bayesian neural networks (assisted by variational inference), where the weights are modelled probabilistically as random variables, have emerged as the most popular tool for making a prediction of the uncertainty of the neural network [123]. However, many assumptions are necessary in order to apply this approach. For example, the weights are commonly assumed to have a Gaussian prior (or mixture of Gaussians). In addition, variational inference is more computationally expensive than typical back-propagation algorithms [21].

Probabilistic techniques are not the only method of modelling epistemic uncertainty. Chapter 3 described interval predictor models; a recently developed machine learning technique for supervised learning which makes interval predictions with guaranteed accuracy [34]. To recap, for every input example, $x^{(i)}$ an interval predictor model would predict bounds on the output, $\bar{y}(x^{(i)})$ and $\underline{y}(x^{(i)})$, instead of just $y(x^{(i)})$. The technique relies upon the solution of chance constrained convex optimisation programs by the scenario technique

[29], which can accommodate interval valued training data.

In this chapter, we propose a back-propagation algorithm for neural networks with interval predictions, and show how this can be efficiently used to create interval neural networks. The proposed constant width interval predictions can be seen as a robust homoscedastic bound on the uncertainty of the trained network, though we also propose a method for obtaining non-constant width predictions. The work can be seen as a generalisation of the non-convex interval predictor models discussed in Chapter 3.3, where the maximum error loss function is minimised. The contribution in this chapter allows significantly deeper networks to be trained with less computational expense, by proposing modifications to the maximum error loss function which are minimised for minibatches of training data. The proposed networks can make predictions with hetreoscedastic uncertainty for multiple outputs, and imprecise training data can be used. We present results for a test function example, and an engineering problem. Furthermore, we demonstrate the effect using different minibatch sizes.

9.2 Comparison with related work

Osband [128] and Kendall and Gal [96] discuss epistemic and aleatory uncertainty in deep neural networks from a Bayesian perspective. However, several other probabilistic approaches to quantifying the uncertainty in neural networks exist. For example, ensemble techniques can be applied to create multiple neural networks which improve the uncertainty quantification compared to mean squared error approaches [67, 109].

Interval neural networks offer a principled framework for dealing with imprecision in training data. This chapter describes a novel framework for training neural networks which output a specific type of convex set prediction: super-ellipsoids, which are mathematically parameterised as ellipsoids in a space equipped with an ℓ_p norm (this acts essentially as a transformation enabling continuous deformation between hyper-spheres and hyper-cubes). The hyper-ellipsoidal case, representing correlated uncertainty between outputs, and hyper-rectangular case, representing no correlation between outputs, are discussed in detail in Section 9.3.4. Bayesian techniques, and other probabilistic techniques for neural networks are not capable of handling the set inclusion constraints required for the neural network output, since the output is a point value or probability distribution rather than a set. The interval neural networks in this chapter have several advantages over those proposed in previous literature. The main advantage is that the training algorithms allow more complex network architectures to be trained. Specifically, the gradient descent algorithm can be used and no penalty or barrier functions are required in the loss function. In addition, ℓ_2 ball training data is considered while other papers (e.g. Lacerda and Crespo [108]) only consider ℓ_∞ training data.

Like other set-based and interval uncertainty models, the networks do not indicate the relative likelihood of different outputs within the prescribed output interval, because an interval communicates less information than a fuzzy number or probability distribution; it indicates complete uncertainty within the defined range. Although the interval output could be seen as a disadvantage because it is less expressive than a probability distribution, it enables simple guarantees to be made on the performance of the network (e.g. Section 3.3.2). Furthermore, the loss functions proposed in this chapter currently only apply to regression problems; no attempt has been made to generalise typical classification loss functions, e.g. the logistic loss function or cross entropy loss function.

9.3 Interval neural network training

9.3.1 Overview

Firstly, we recall from Chapter 3 that one can create a homoscedastic interval predictor model (Eqn. 3.19) by finding the neural network weights which minimise the maximum absolute error loss:

$$\mathcal{L}_{\text{max-error}} = \max_i |y^{(i)} - \hat{y}(x^{(i)})|, \tag{9.1}$$

where $\hat{y}(x)$ represents the central line of the prediction from a neural network (feedforward or otherwise, as described in Chapter 3), and the prediction width h is the minimum value of the loss.[*] It is trivial to show this is true, since the set inclusion constraint in Eqn. 3.19 requires that h is larger than the absolute error for each data point in the training set. For the avoidance of doubt, Eqn. 9.1 takes the maximum over each point in the training dataset, rather than each component of a multi-output neural network (though this is discussed in Section 9.3.4). In order to minimise the loss in Eqn. 9.1, stochastic gradient descent is used. To obtain an accurate estimate of h (the minimum value of the loss), the loss function

$$\mathcal{L}_{\text{actual}} = \mathcal{L}_{\text{max-error}} + (h - \mathcal{L}_{\text{max-error}})^2 \tag{9.2}$$

[*] i.e. $\hat{y}(x) = \frac{\overline{y}(x)+\underline{y}(x)}{2}$ and $h = \frac{\overline{y}(x)-\underline{y}(x)}{2}$.

is minimised with respect to the weights and h, which is beneficial as the estimate for h is effectively averaged by the gradient descent algorithm and is hence more accurate than simply setting h to the value of the loss at any iteration in particular, whilst the minimum of the loss for the network weights, W, remains unchanged. This technique can be trivially applied for every subsequent loss function described in this chapter, and is used in all of our numerical experiments. Our algorithm is described in further detail in Algorithm 2. Note that Algorithm 2 could also be initialised with the weights obtained by training the network with a mean squared error loss function, if these were already available.

Algorithm 2 Maximum error backpropagation method

```
Input: Training data pairs ($x^{(j)}$, $y^{(j)}$ for $j = 1, ..., N$)
  Randomly initialise weight tensor and $h$.
  for $i = 1, .., N_{iter}$ do
    Set $k = \arg\max_{j \in [1,...,N]} |y^{(j)} - \hat{y}(x^{(j)})|$
    Use gradient of loss function to update $W$ and $h$ ($W \leftarrow W + \eta \nabla_W |y^{(k)} - \hat{y}(x^{(k)})| + (h - |y^{(k)} - \hat{y}(x^{(k)})|)^2$ $h \leftarrow h + \eta \frac{\partial}{\partial h}(h - |y^{(k)} - \hat{y}(x^{(k)})|)^2$)
  end for
Output: Weight tensor and $h$
```

Algorithm 2 is more costly than the standard back propagation method, since the proposed method costs $\mathcal{O}(N \cdot N_{iter})$, compared to a standard stochastic gradient descent cost of $\mathcal{O}(N_{iter})$. The algorithm is amenable to parallelisation, since at each step the N evaluations of the absolute error can be made simultaneously. However, the largest GPU architectures have several thousand cores, so for datasets with millions of data points Algorithm 2 would not be tractable. Note also that the N_{iter} required for convergence in both algorithms is not necessarily the same, as this depends on the variance of the gradient at each step.

9.3.2 Scalability improvement

We propose the use of minibatch stochastic gradient descent to reduce the computational cost of the algorithm [53], whereby a randomly selected subset of size M of the training data is selected at each step and used to evaluate the maximum absolute error loss, Eqn. 9.1. This procedure is described in further detail in Algorithm 3.

Using minibatches reduces the cost of the proposed algorithm to $\mathcal{O}(M \cdot N_{iter})$, which is a potentially vast improvement when $N >> M > 1$. However, we are now only minimising

Algorithm 3 Maximum error backpropagation method, using minibatches

```
Input: Training data pairs ($x^{(i)}$, $y^{(i)}$ for $i = 1, ..., N$)
  Randomly initialise weight tensor and $h$.
  for $i = 1, .., N_{iter}$ do
    Generate set, $B$, of $M$ random numbers, sampled without replacement between 1 and $N$
    Set $k = \arg\max_{j \in B} |y^{(j)} - \hat{y}(x^{(j)})|$
    Use gradient of loss function to update $W$ and $h$ ($W \leftarrow W + \eta \nabla_W |y^{(k)} - \hat{y}(x^{(k)})| + (h - |y^{(k)} - \hat{y}(x^{(k)})|)^2$ $h \leftarrow h + \eta \frac{\partial}{\partial h}(h - |y^{(k)} - \hat{y}(x^{(k)})|)^2$)
  end for
Output: Weight tensor and $h$
```

an *approximation* of Eqn. 9.1. Fortunately, the true loss function can be approximated well for reasonably small minibatch sizes.

The probability of selecting the true maximum of the absolute error in a minibatch by random sampling without replacement is $\frac{M}{N}$. The probability that the maximum error point selected in the minibatch is the i-th largest error in the training set is

$$P(i) = \frac{\binom{N-i}{M-1}}{\binom{N}{M}}. \tag{9.3}$$

Then to find the expectation of i we calculate

$$\mathbb{E}(i) = \sum_{i=1}^{i=N-M+1} i \frac{\binom{N-i}{M-1}}{\binom{N}{M}} = \frac{N+1}{M+1}. \tag{9.4}$$

In the case that $\frac{N}{M} >> 1$ we find that the expression for the expected percentile reduces to

$$\frac{\mathbb{E}(i)}{N} \approx \frac{1}{M}, \tag{9.5}$$

so for large N we find that using a minibatch of size M is equivalent to minimising the $\frac{1}{M}$-th percentile of the empirical cumulative distribution function of the error for the whole training dataset. The variance of the percentile is

$$\text{Var}\,(\frac{i}{N}) = \frac{M(N-M)(1+N)}{N^2(1+M)^2(2+M)}, \tag{9.6}$$

which becomes

$$\mathrm{Var}\,(\frac{i}{N}) \approx \frac{1}{M^2} \tag{9.7}$$

in the large N limit. Therefore we see that the minibatch technique performs best when the size of the training set is large, but it is also necessary to increase the minibatch size to avoid the gradient having a large variance.

Now, let us consider the case when the $k-1$ data points in the minibatch with the largest error are ignored, i.e. we minimise the k-th largest error in the minibatch. The probability that the k-th largest error point selected in the minibatch is the i-th largest error in the training set is

$$P(i) = \frac{\binom{N-i}{M-k}\binom{i-1}{k-1}}{\binom{N}{M}}. \tag{9.8}$$

In Nagaraja [122] the order statistics are given for uniform distributions sampled without replacement. This allows us to find the expectation of i, which is given by

$$\mathbb{E}(i) = \frac{k(1+N)-1-M}{(1+M)}. \tag{9.9}$$

Therefore, by minimising the k-th largest error in a minibatch of size M in the $\frac{N}{M} >> 1$ limiting case, one is actually minimising the $\frac{k}{M}$-th percentile of the empirical cumulative distribution function of the error for the whole training dataset. This reduces to Eqn. 9.4 when $k=1$, as expected. If desired this can be checked after training by passing the entire dataset through the model once and checking the identified value of h against the data (this check will not be too costly if $N_{iter}M >> N$). The variance of i is

$$\mathrm{Var}\,(i) = \frac{k(M-k+1)(N+1)(N-M)}{(M+1)^2(M+2)}, \tag{9.10}$$

which reduces to Eqn. 9.7 when the appropriate limits are taken. This provides valuable insight - when $k > 1$ is minimised it is necessary to increase M slightly to maintain constant variance in the gradient.

All the above results assume the minibatch is constructed by sampling without replacement. If the minibatch is constructed by sampling with replacement then the order statistics for sampling with replacement should be used instead.

Minimising the empirical percentiles of the error on the training set, with the minibatch approximation of Eqn. 9.1, allows us to control the training of the neural network but does not by itself provide a statistical guarantee on performance on the test set. To statistically guarantee performance of the model it is therefore necessary to use the techniques in Section 3.3.2, and in particular the a posteriori frequentist analysis.

9.3.3 Incertitude in training data

Algorithm 2 and Algorithm 3 are described for use with crisp training data. However one of the main advantages of the interval predictor model framework is that training data with incertitude (i.e. interval training data or fuzzy data) fits coherently into the paradigm [108]. An example of incertitude in training data is a common defence against the adversarial attack model from Madry et al. [112]. The proposed attack model places each training data point in an uncertain hyper-sphere (ℓ_2 ball). Typically, in the context of uncertainty quantification, incertitude is characterised with intervals (ℓ_∞ ball). However, both cases are convex sets and therefore the conceptual challenge of accommodating this training data is similar. Since the neural network model is more complex than that proposed in Lacerda and Crespo [108], the computations required to accommodate interval data are also more complex.

For the case of interval imprecision in the output variables (i.e. pairs $x^{(i)}$ and $[\underline{y}^{(i)}, \bar{y}^{(i)}]$ are observed), Eqn. 3.19 can be modified as follows:

$$\arg\min_{W,h} [h : \max(|\bar{y}^{(i)} - \hat{y}(x^{(i)})|, |\underline{y}^{(i)} - \hat{y}(x^{(i)})|) < h \;\forall\; i], \tag{9.11}$$

which can be written in simplified form if the width of interval $[\underline{y}^{(i)}, \bar{y}^{(i)}]$ is constant for all data points. The optimisation program in Eqn. 9.11 can be solved by minimising the loss

$$\mathcal{L}_{\text{output incertitude}} = \max_i \left(\max(|\bar{y}^{(i)} - \hat{y}(x^{(i)})|, |\underline{y}^{(i)} - \hat{y}(x^{(i)})|)\right), \tag{9.12}$$

with respect to the weights, W, where h becomes the value of the loss at the minimum.

For interval incertitude in the input training data the situation is more complex. Since the sum of squares approach used in Lacerda and Crespo [108] is not directly applicable, the algorithm with neural networks will be more costly. If the pairs $[\underline{x}^{(i)}, \bar{x}^{(i)}]$ and $[\underline{y}^{(i)}, \bar{y}^{(i)}]$ are observed then one must solve

$$\arg\min_{W,h} [h : \max_{x \in [\underline{x}^{(i)}, \bar{x}^{(i)}]} (|\bar{y}^{(i)} - \hat{y}(x)|, |\underline{y}^{(i)} - \hat{y}(x)|) < h \;\forall\; i], \tag{9.13}$$

where the nested optimisation in the constraints significantly increases the complexity of the algorithm. Several strategies for efficiently solving such programs were described in Section 2.1.2, and will be re-iterated here.

One approach to solving this problem would be to attempt to brute force the nested optimisation (i.e. discretise along the upper 'edge' of the incertitude box). However if the incertitude is large or the dimensionality of the training data is high, then this becomes impractical. Another possibility is assuming the prediction of the neural network is approximately linear locally, and using the gradient of the neural network with respect to the inputs (which is known analytically) to find an approximate solution to the nested optimisation problem. This is similar to the approaches proposed in Kurakin et al. [107] and Goodfellow et al. [78], where the gradient is used to search within a set close to the original training data for points which maximise the loss function of the neural network. The crucial difference is that in the formulation proposed in this chapter only the surface of the set must be searched, since the aim is to enclose the whole set in the interval neural network.

Therefore we propose that Eqn. 9.13 is best solved by minimising

$$\mathcal{L}_{\text{input incertitude}} = \max_i \max\left(|\bar{y}^{(i)} - (\hat{y}(\frac{\bar{x}^{(i)} + \underline{x}^{(i)}}{2}) - \epsilon^{(i)})|, |\underline{y}^{(i)} - (\hat{y}(\frac{\bar{x}^{(i)} + \underline{x}^{(i)}}{2}) + \epsilon^{(i)})|\right), \tag{9.14}$$

with respect to the parameters W, where $\epsilon^{(i)} = |\frac{\bar{x}^{(i)} - \underline{x}^{(i)}}{2} \cdot (\text{sign}(\nabla_x \hat{y}(\frac{\bar{x}^{(i)} + \underline{x}^{(i)}}{2})) \circ \nabla_x \hat{y}(\frac{\bar{x}^{(i)} + \underline{x}^{(i)}}{2}))|$ ($\circ$ denotes component-wise multiplication of vectors), and h becomes the value of the loss at the minimum. This loss will provide an accurate solution to Eqn. 9.13 when the output of the neural network ($\hat{y}(x)$) is locally linear for a Taylor series expansion in the training data intervals, such that $|\hat{y}(x + \delta x) - (\hat{y}(x) + \delta x \cdot \nabla_x \hat{y}(x))| < \omega$, where ω is an arbitrarily small constant representing the accuracy of the solution and δx is a constant vector at the length scale of the interval width. Of course, higher order Taylor expansions can be used to construct more complex loss functions, or the assumption of monotonicity can be made $(\max_{x \in [\underline{x}^{(i)}, \bar{x}^{(i)}]} \hat{y}(x) = \hat{y}(\frac{\bar{x}^{(i)} + \underline{x}^{(i)}}{2} + \frac{\bar{x}^{(i)} - \underline{x}^{(i)}}{2} \circ \text{sign}(\nabla_x \hat{y}(\frac{\bar{x}^{(i)} + \underline{x}^{(i)}}{2}))))$.

This approach provides a computationally feasible approximate solution to many practical problems involving the ℓ_∞ ball and ℓ_2 ball uncertainty models.

9.3.4 Multi-output neural networks

A key advantage of neural networks over other machine learning techniques is the ease with which correlation between model outputs can be expressed. For example, this is applicable when the output layer is a one hot encoder for classification tasks, or an image for computer vision tasks. It is also of use for multi-task learning [7, 186]. So far Algorithm 2 and Algorithm 3 have been described in the context of supervised learning from data with only one output dimension. We generalise the algorithms in the previous sections to multi-output neural networks by predicting an ℓ_p ball, with radius h, around the output of the neural network in the output space. To recap Section 2.1.2, note that $p = \infty$ corresponds to no correlation between outputs (intervals, or hypercubes), and $p \to 0$ corresponds to the case of completely correlated outputs. p becomes a hyper-parameter which can be optimised to express the dependence between outputs in the proposed model. A weighted norm can be used to form more complex shapes like super-ellipsoids, or hyper-rectangles (as opposed to hyper-cubes).

To train the interval neural network the optimisation program

$$\arg\min_{W,h} [h : \left(\sum_j \left| \frac{y_j^{(i)} - \hat{y}_j(x^{(i)})}{\hat{\sigma}_j} \right|^p \right)^{\frac{1}{p}} < h \ \forall \ i] \tag{9.15}$$

should be solved, where the weights $\hat{\sigma}$ are normalised such that $\|\hat{\sigma}\|_2 = 1$, which is ensured by setting $\hat{\sigma}_i = \frac{\sigma_i}{\sqrt{\sum_j \sigma_j^2}}$, where σ_i are parameters to be found during training.

In practice, training takes place by replacing the absolute error in Algorithm 3 with the appropriate ℓ_p distance in the output space, i.e.

$$\mathcal{L}_{\text{multi-output}} = \max_i \left(\sum_j \left| \frac{y_j^{(i)} - \hat{y}_j(x^{(i)})}{\hat{\sigma}_j} \right|^p \right)^{\frac{1}{p}}, \tag{9.16}$$

which reduces to the case of a ℓ_p ball when $\sigma_i = 1 \ \forall \ i$.

For example, if the analyst believes there is no dependency between outputs they might minimise Eqn. 9.16 with $p = \infty$ for minibatches of training data. The network would then predict intervals (hyper-rectangles) with radius h. Explicitly, the training loss for neural networks predicting hyper-rectangles is given by:

$$\mathcal{L}_{\text{hyper-rectangle}} = \max_i \max_j \left| \frac{y_j^{(i)} - \hat{y}_j(x^{(i)})}{\hat{\sigma}_j} \right|. \tag{9.17}$$

The trained network makes interval predictions with centre $\hat{y}_j(x)$ and half-width $h\hat{\sigma}_j(x)$.

9.3.5 Heteroscedastic interval uncertainty

So far the interval neural networks discussed (i.e. from solving Eqn. 3.19) have made predictions with constant interval width, or constant convex set width in the case of multi-output models. There may be some situations where a richer description of uncertainty is desired. Therefore in this section we describe how to generalise the results from the previous sections to the case of non-constant width interval prediction. Rather than solving the original interval neural network optimisation program (Eqn. 3.18), we propose a modified model:

$$\operatorname*{arg\,min}_{W,h} \left[h : \frac{|y^{(i)} - \hat{y}(x^{(i)})|}{\hat{\sigma}(x^{(i)})} < h \ \forall \ i\right], \tag{9.18}$$

where the neural network provides $\hat{y}(x^{(i)})$ (the central line prediction of the interval), and $\hat{\sigma}(x^{(i)})$ (the interval half-width), and the other symbols have the same meanings as in Eqn. 3.19. In order for the optimisation program to yield a plausible interval neural network it is required that $\hat{\sigma}(x^{(i)}) > 0$ and $\mathbb{E}_x(\hat{\sigma}(x)) = 1$. These constraints can be enforced by setting $\hat{\sigma}(x^{(i)}) = \frac{\sigma(x^{(i)})}{\mathbb{E}_x(\sigma(x))}$, where $\sigma(x^{(i)})$ is output from a neural network layer with positive only activation function (e.g. ReLU or Softplus, or in the case of a multi-output neural network, as in the previous section, Softmax). Then the neural network can be trained by minimising the loss given by

$$\mathcal{L}_{\text{heteroscedastic}}(y^{(i)}) = \max_i \frac{|y^{(i)} - \hat{y}(x^{(i)})| \mathbb{E}_x(\sigma(x))}{\sigma(x^{(i)})}, \tag{9.19}$$

again h is obtained from the minimum value of the loss function. The loss is evaluated on minibatches, and therefore $\mathbb{E}_x(\sigma(x))$ is computed using the Monte Carlo estimator of the expectation on the minibatch. The trained network makes interval predictions with centre $\hat{y}(x)$ and half-width $h\hat{\sigma}(x)$ (the normalising factor $\mathbb{E}_x(\sigma(x))$ should be precomputed and stored).

9.4 Numerical examples

All experiments were timed on TensorFlow on a Google Colaboratory session equipped with an NVIDIA Tesla T4. In all experiments, TensorFlow's ADAM optimiser was used with exponential gradient decay, i.e. learning rate = initial learning rate $*$ decay rate$^{\frac{\text{global step}}{\text{decay steps}}}$ [100].

9.4.1 Simple numerical example

Description

In order to illustrate the developed techniques we will demonstrate the interval neural network on a modified version of a simple problem from Campi et al. [35]. We train a neural network in TensorFlow with 1 hidden layer containing 10 neurons with hyperbolic tangent activation on 1250 samples from the following test function:

$$y = 0.3 * (15 * x * \exp(-3 * x) + w * x) \tag{9.20}$$

where w is a normal distributed random variable with zero mean and standard deviation $\sigma = 0.025$. The data is generated by sampling from the input variable x uniformly between 0 and 1. We perform the following experiments:

1. We train a constant width neural network using the loss from Eqn. 9.1, using a minibatch size of $M = 200$;

2. We repeat the previous experiment with a minibatch size of $M = 20$ to demonstrate the effect of using a smaller minibatch size;

3. We train a neural network with heteroscedastic uncertainty using the loss from Eqn. 9.19;

4. We train a neural network using the mean squared error (MSE) loss as a comparison.

For clarity, the algorithm used is described in Algorithm 3. The hyper-parameters used are shown in Table 9.1. Note that an epoch is defined as one pass of the whole dataset through the model, so training runs with smaller batch sizes require more iterations to complete the same number of training epochs, and hence will require more training time. These optimiser hyper-parameters were tuned manually by inspecting the loss curves, and

Experiment	1 Const. Width	2 Const. Width	3 Heteroscedastic	4 MSE
Minibatch size, M	200	20	200	200
Initial learning rate	0.1	0.1	0.1	0.1
Learning rate decay rate	0.96	0.96	0.96	0.96
Number of training epochs	6000	6000	6000	6000
Decay steps	100	100	100	100

Table 9.1: Hyper-parameters used in the numerical experiments for the simple analytical function.

the Minibatch size was chosen to be large enough to benefit from the properties discussed in 9.3.2. The weights were initialised using the TensorFlow defaults (Glorot uniform initializer [75] for the kernel and zeros for the bias), and h was initialised at zero.

Results

The training loss curves for Experiments 1, 2, 3 and 4 are shown in Figures 9.1, 9.3, 9.5 and 9.7 respectively. The trained neural networks for Experiments 1, 2, 3 and 4 are shown in Figures 9.2, 9.4, 9.6, and 9.8 respectively.

Using a train–test split ratio of 0.2, and the a posteriori frequentist analysis approach from Section 3.3.2 we calculate bounds on the violation probability, $\bar{v}$ and $\underline{v}$, from

$$P(V(\hat{z}_N) < \bar{v} \cap V(\hat{z}_N) > \underline{v}) = 10^{-3} \tag{9.21}$$

for each trained interval neural network (with the test set size, $N_t = 250$). In this case the solution $\hat{z}_N$ is the obtained weights and model width of the interval neural network. The results are summarised in Table 9.2, which also displays the model half-width h for each trained network and the number of bound violating test points, N_v.

As expected, the number of violating test points, N_v, and hence the bounds on the violation probability, $V(\hat{z}_N)$, are higher in Experiment 2 than Experiment 1, as the minibatch size, M, is lower. In addition, we observe that as expected, the model half-width, h, is much lower in Experiment 3 than in Experiment 1. This indicates that the model in Experiment 1 is far too simple for the dataset, which we know to be true because in reality the training data contains heteroscedastic additive noise. For comparison, we observe that the neural network trained with the mean squared error loss function (Experiment 4), has a root mean squared error on the test set of 4.3×10^{-3}, and a fitted model which is comparable

Experiment	1	2	3	4
	Const. Width	Const. Width	Heteroscedastic	MSE
Test points, N_t	250	250	250	250
Bound violating test points, N_v	1	5	2	N/A
$\bar{v}$	3.6×10^{-2}	6.4×10^{-2}	4.4×10^{-2}	N/A
$\underline{v}$	4.0×10^{-6}	3.0×10^{-3}	1.8×10^{-4}	N/A
Model half-width, h	1.5×10^{-2}	1.0×10^{-2}	9.7×10^{-3}	N/A
Root mean squared error	N/A	N/A	N/A	4.3×10^{-3}
Runtime (s)	66	525	75	68

Table 9.2: Results from the numerical experiments with the simple analytical function.

to those in the interval model experiments (since if the strong assumption is made of a fitted Gaussian probability density then 99.7% data points would fall within 3 standard deviations of the mean).

9.4.2 Simple numerical example with uncertain training data

Description

In order to demonstrate the developments in Section 9.3.3, we train an interval neural network on a modified version of the previous example, where the training data consists of ℓ_∞ balls (intervals). The centre of the intervals $(x^{(i)}, y^{(i)}) = (\frac{\bar{x}^{(i)}+\underline{x}^{(i)}}{2}, \frac{\bar{y}^{(i)}+\underline{y}^{(i)}}{2})$ is generated by Eqn. 9.20. The incertitude in both the inputs and outputs will be given by the interval radius,

$$\frac{\bar{y}-\underline{y}}{2} = \frac{\bar{x}-\underline{x}}{2} = \frac{1}{160*(|x-0.5|+0.1)}. \tag{9.22}$$

In order to allow for heteroscedasticity we train the network with the loss from Section 9.3.3 with the scaling in Section 9.3.5, i.e. we minimise $\mathcal{L}_{\text{experiment}} + (\mathcal{L}_{\text{experiment}} - h)^2$ where

$$\begin{aligned}\mathcal{L}_{\text{experiment}} = \max(&\mathcal{L}_{\text{heteroscedastic}}(\bar{y}^{(i)}) + \frac{\bar{x}^{(i)}-\underline{x}^{(i)}}{2} \cdot \text{abs}(\nabla_x \mathcal{L}_{\text{heteroscedastic}}(\bar{y}^{(i)})),\\ &\mathcal{L}_{\text{heteroscedastic}}(\underline{y}^{(i)}) + \frac{\bar{x}^{(i)}-\underline{x}^{(i)}}{2} \cdot \text{abs}\left(\nabla_x \mathcal{L}_{\text{heteroscedastic}}(\underline{y}^{(i)})\right)\end{aligned} \tag{9.23}$$

where abs is the component-wise absolute value, σ and ϵ take the same meanings as in

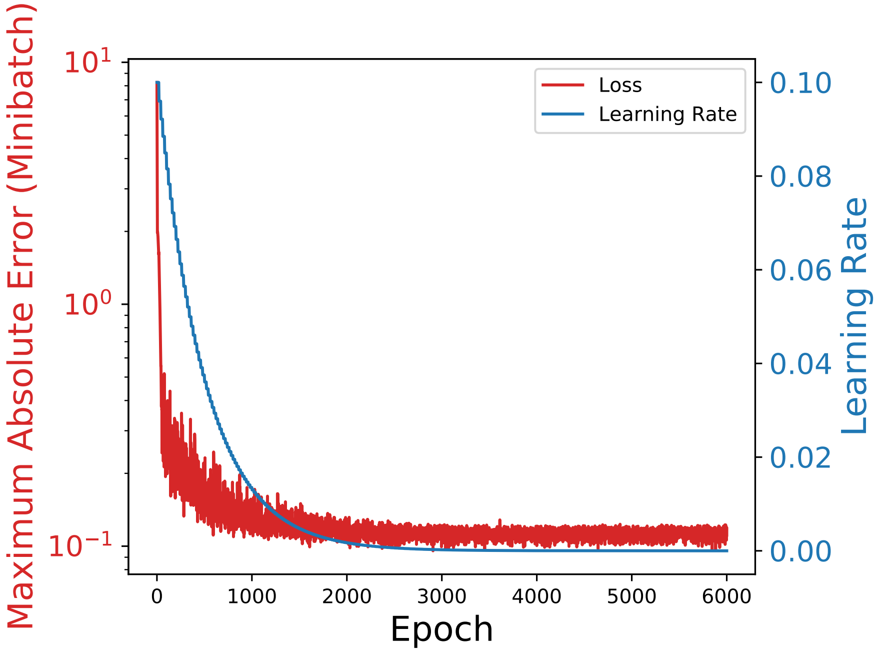

Figure 9.1: Plot of convergence of the neural network for Experiment 1.

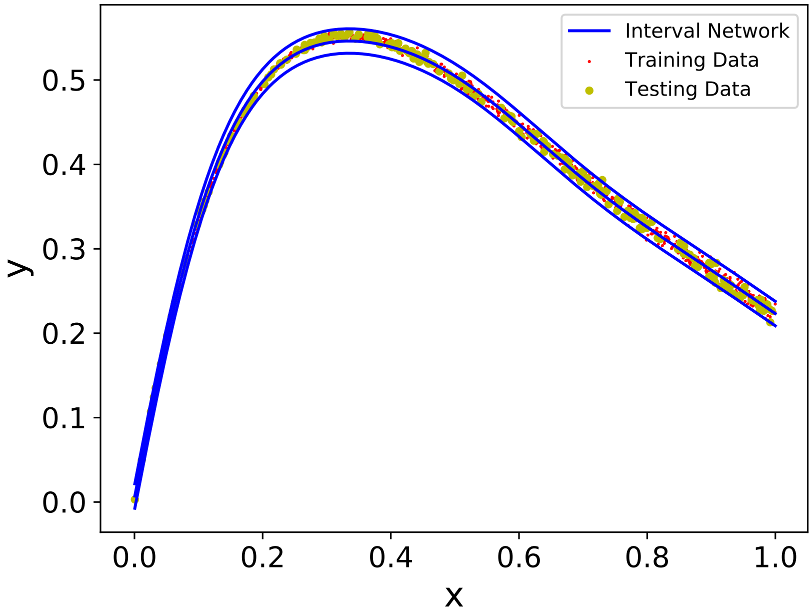

Figure 9.2: Plot of trained interval neural network for Experiment 1. Training set shown in red, test set shown in yellow.

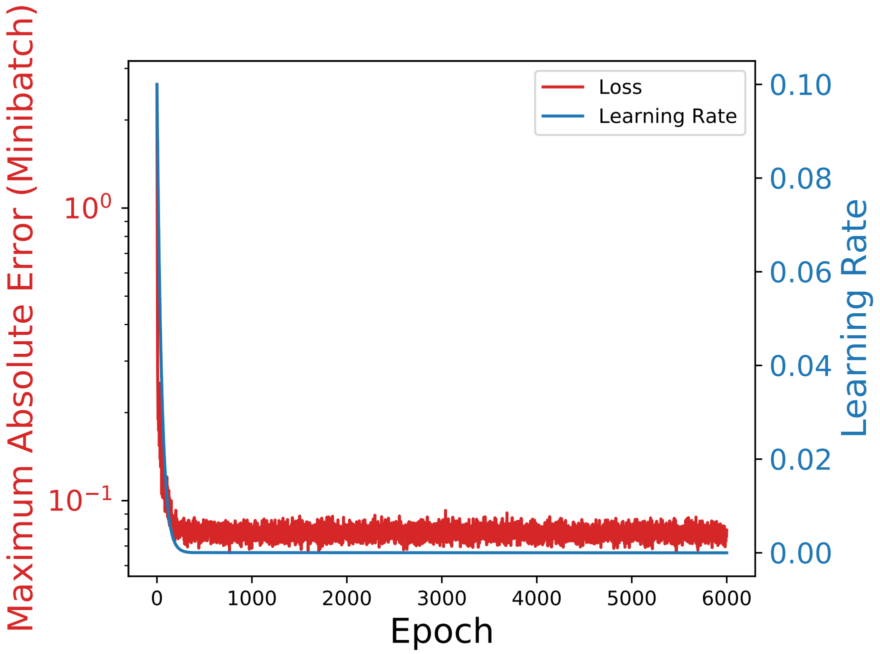

Figure 9.3: Plot of convergence of the neural network for Experiment 2.

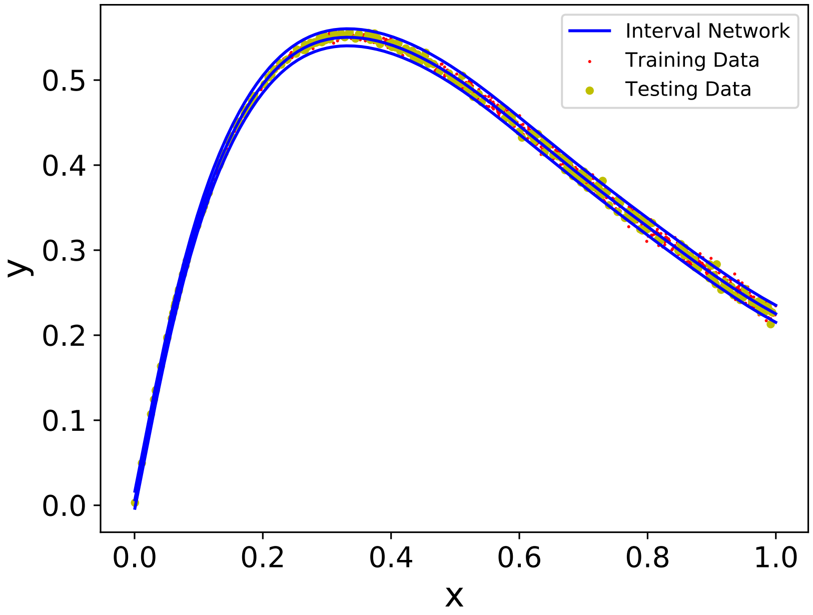

Figure 9.4: Plot of trained interval neural network for Experiment 2. Training set shown in red, test set shown in yellow.

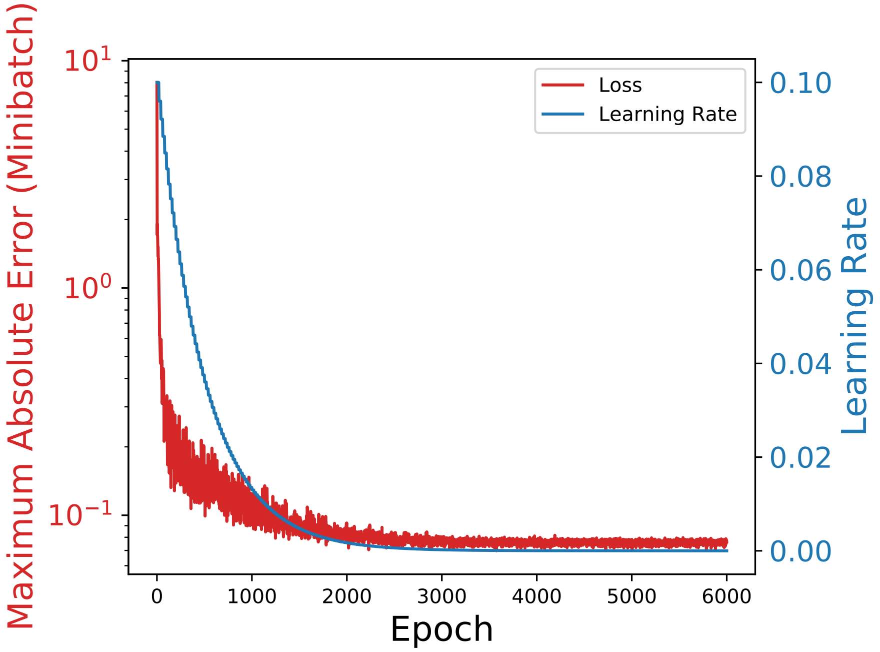

Figure 9.5: Plot of convergence of the neural network for Experiment 3.

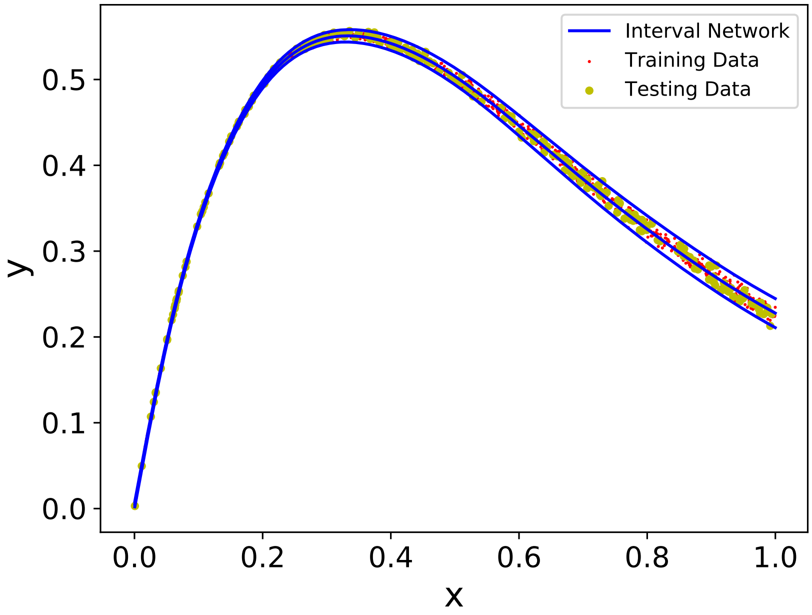

Figure 9.6: Plot of trained interval neural network for Experiment 3. Training set shown in red, test set shown in yellow.

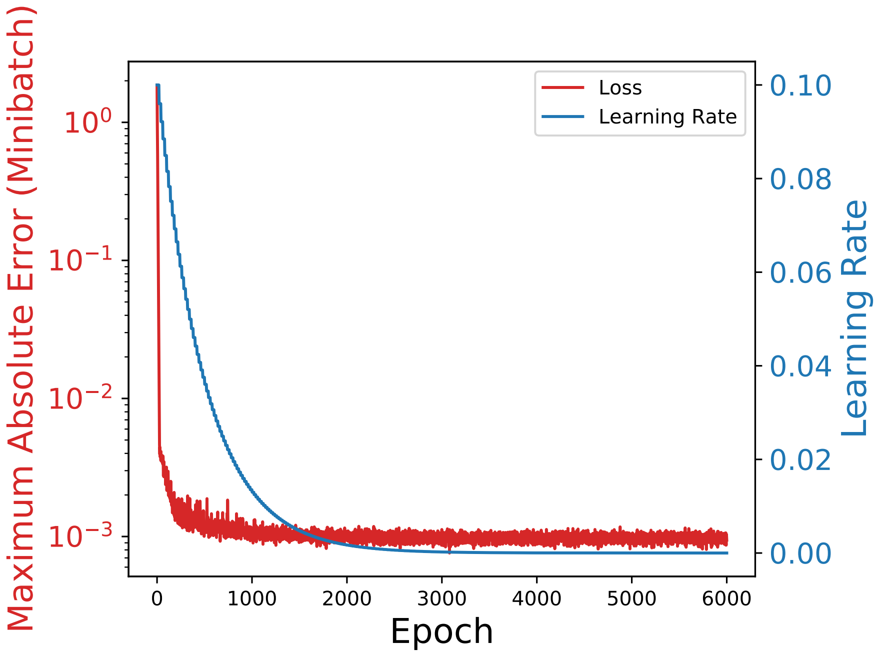

Figure 9.7: Plot of convergence of the neural network for Experiment 4 (mean squared error loss).

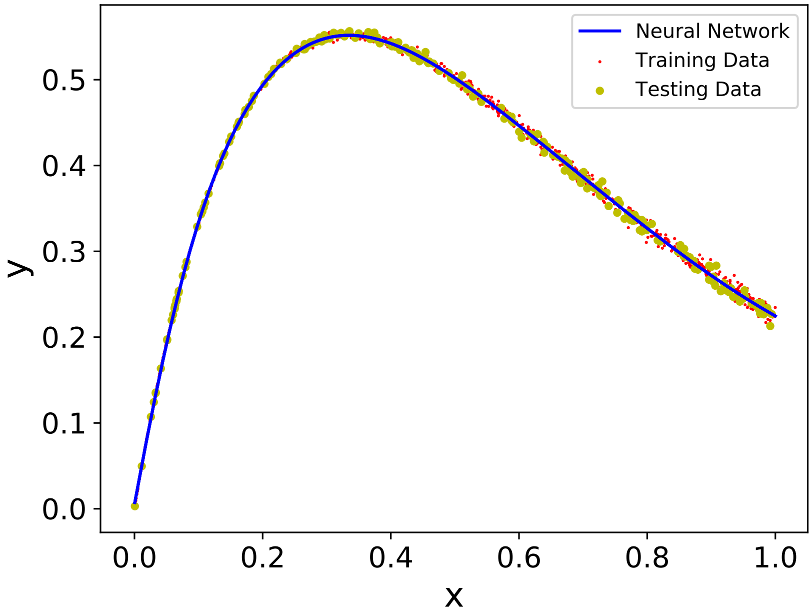

Figure 9.8: Plot of trained interval neural network for Experiment 4 (mean squared error loss). Training set shown in red, test set shown in yellow.

Experiment	1 Const. Width	2 MSE
Minibatch size, M	200	200
Initial learning rate	0.1	0.1
Learning rate decay rate	0.96	0.96
Number of training epochs	6000	6000
Decay steps	100	100

Table 9.3: Hyper-parameters used in the numerical experiments with interval training data.

previous chapters, and the gradient of the loss, $|\nabla_x \mathcal{L}_{\text{heteroscedastic}}|$, is evaluated at the centre of the intervals $(\frac{\bar{x}^{(i)}+\underline{x}^{(i)}}{2})$. The same neural network architecture was used as in the previous example (10 neurons in hidden layer). In order to make a comparison, another network with two hidden layers with 10 and 20 neurons was used. This is summarised in Table 9.3.

In this case the interval network could not be compared with a traditional neural network, as a traditional neural network would not be able to represent the set inclusion constraint required to train with interval data.

Results

The training loss curves and trained single interval neural network are shown in Figure 9.9 and Figure 9.10. The corresponding plots of the neural network with two hidden layers are shown in Figure 9.11 and Figure 9.12. Using a train–test split ratio of 0.2, and the a posteriori frequentist analysis approach from Section 3.3.2 we calculate bounds on $V(\hat{z}_N)$ with confidence 0.999. Although the single-layer interval neural network encloses the expected proportion of data based on the minibatch size, the interval appears overly large in places. This indicates that the chosen neural network may be too simple and hence the complexity (number of neurons) of the model could be increased, as in the neural network with two hidden layers. The results are summarised in Table 9.4.

9.4.3 Multi-output test function

Description

In order to test the multi-output loss function proposed in Section 9.3.4, we train an interval neural network with the loss function Eqn. 9.17 on a test function from Coveney [44], which

Experiment	1 Single Layer	2 Two Layers
Test points, N_t	250	250
Bound violating test points, N_v	1	2
$\bar{v}$	3.6×10^{-2}	4.4×10^{-2}
$\underline{v}$	4.0×10^{-6}	1.8×10^{-4}
Model half-width, h	0.066	0.058
Runtime (s)	182	191

Table 9.4: Results from the numerical experiments with interval training data.

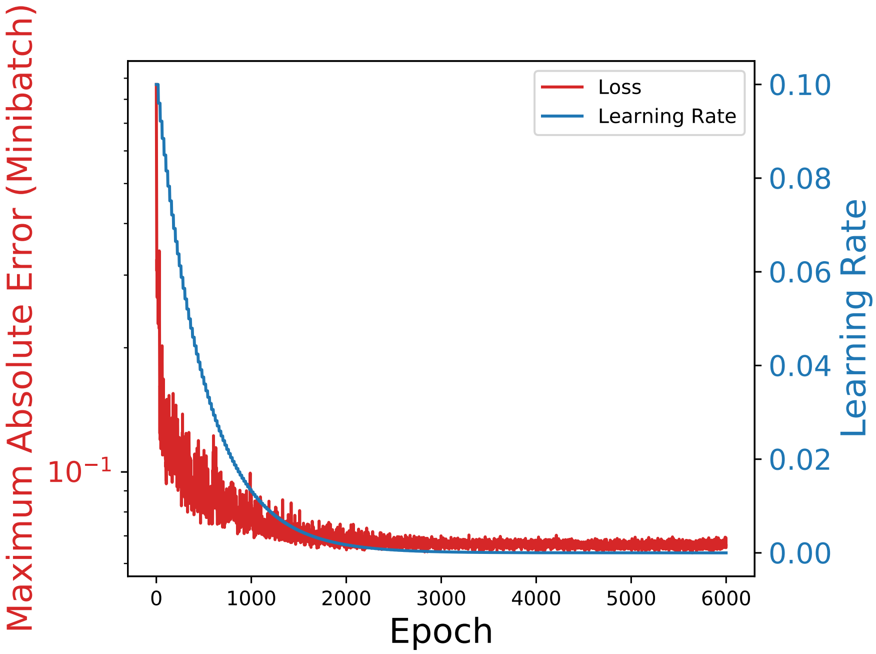

Figure 9.9: Plot of convergence of single hidden layer interval neural network trained on uncertain data.

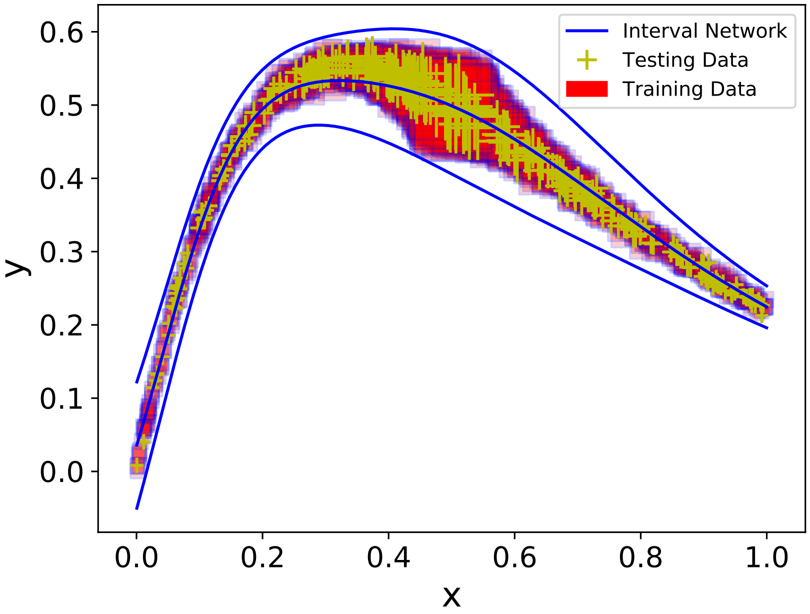

Figure 9.10: Plot of trained single hidden layer interval neural network trained on uncertain data. Training set shown in as red squares, test set shown as yellow crosses.

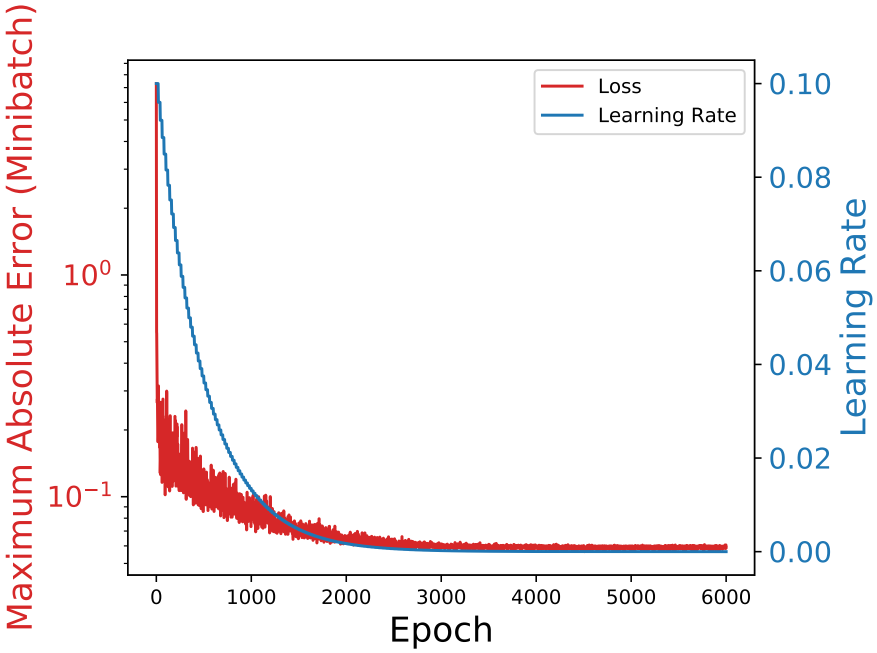

Figure 9.11: Plot of convergence of the interval neural network with two hidden layers trained on uncertain data.

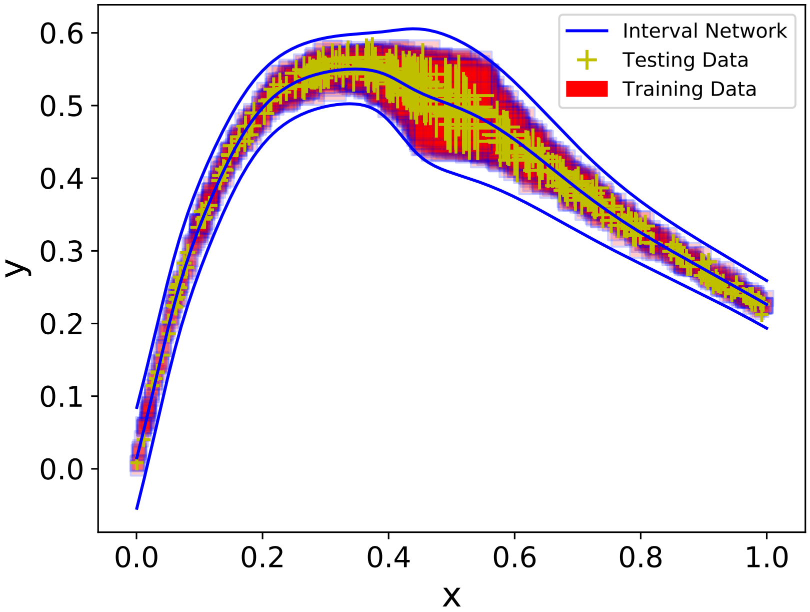

Figure 9.12: Plot of trained interval neural network with two hidden layers trained on uncertain data. Training set shown in as red squares, test set shown as yellow crosses.

Experiment	1 Const. Width	2 MSE
Minibatch size, M	200	200
Initial learning rate	0.01	0.01
Learning rate decay rate	0.99	0.99
Number of training epochs	10000	10000
Decay steps	200	200
ℓ_2 regularisation scale	1.5×10^{-3}	1.5×10^{-3}

Table 9.5: Hyper-parameters used in the numerical experiments for the multi-output test function.

was used to test multi-output emulators. The test function is given by

$$y_1 = 3x_1^3 + \exp(\cos(10x_2)\cos^2(5x_1)) \; + \exp(\sin(7.5x_3)) + w_1 \tag{9.24}$$

and

$$y_2 = 2x_1^2 + \exp(\cos(10x_1)\cos^2(5x_2)) \; + \exp(\sin(7.5x_3^2)) + 1.5w_2, \tag{9.25}$$

where w_1 and w_2 are uniformly distributed random numbers between 0 and 1. The model is trained on 1000 samples from the test function, made by sampling each component of x uniformly between 0 and 1, with a 0.2 train test split ratio. The neural network has 1 hidden layer with ReLU activation and 100 neurons. The hyper-parameters are summarised in Table 9.5. The TensorFlow default initialisers are used, except for σ which is initialised to ones.

Results

The plots of residuals for the network outputs are shown in Figures 9.14 and 9.15. The training loss curve is shown in 9.13. Using a train–test split ratio of 0.2, and the a posteriori frequentist analysis approach from Section 3.3.2 we calculate bounds on $V(\hat{z}_N)$ with confidence 0.999 for the trained interval neural network. The model half-widths were $h\hat{\sigma}_1 = 0.64$ and $h\hat{\sigma}_2 = 0.83$. Encouragingly, the model has identified a noise in each output similar to the true value from the test function. This is comparable with the result obtained by training the same network with the MSE loss. The results are summarised in Table 9.6.

Experiment	1 Const. Width	2 MSE
Test points, N_t	200	200
Bound violating test points, N_v	8	N/A
$\bar{v}$	1.0×10^{-1}	N/A
$\underline{v}$	1.0×10^{-2}	N/A
Model half-width output 1, $h\hat{\sigma}_1$	0.66	N/A
Model half-width output 1, $h\hat{\sigma}_2$	0.85	N/A
Root mean squared error output 1	N/A	0.34
Root mean squared error output 1	N/A	0.50
Runtime (s)	115	88

Table 9.6: Results from the numerical experiments with the multi-output test function.

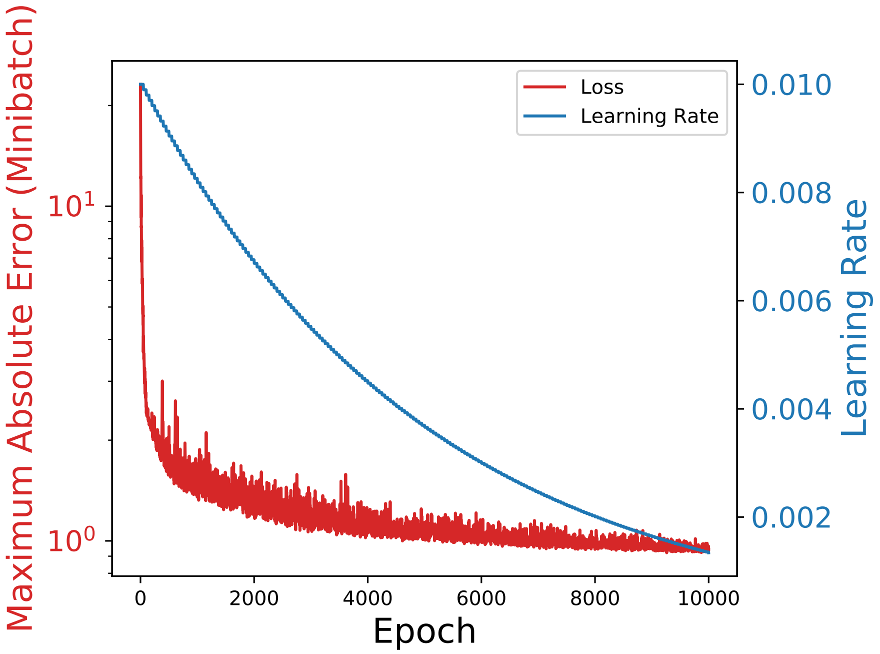

Figure 9.13: Plot of convergence of the multi-output neural network.

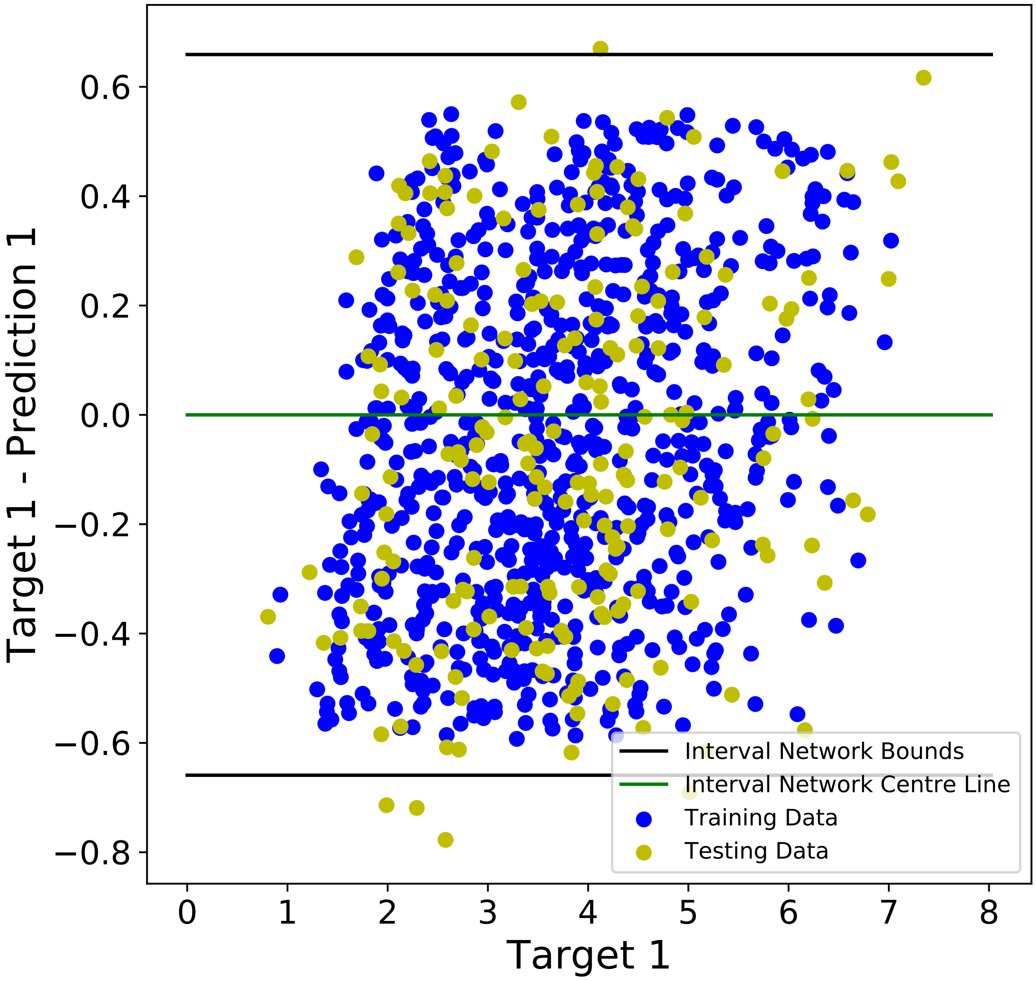

Figure 9.14: Plot of residuals for output 1 of multi-output interval neural network. Training set shown in as blue, test set shown in yellow.

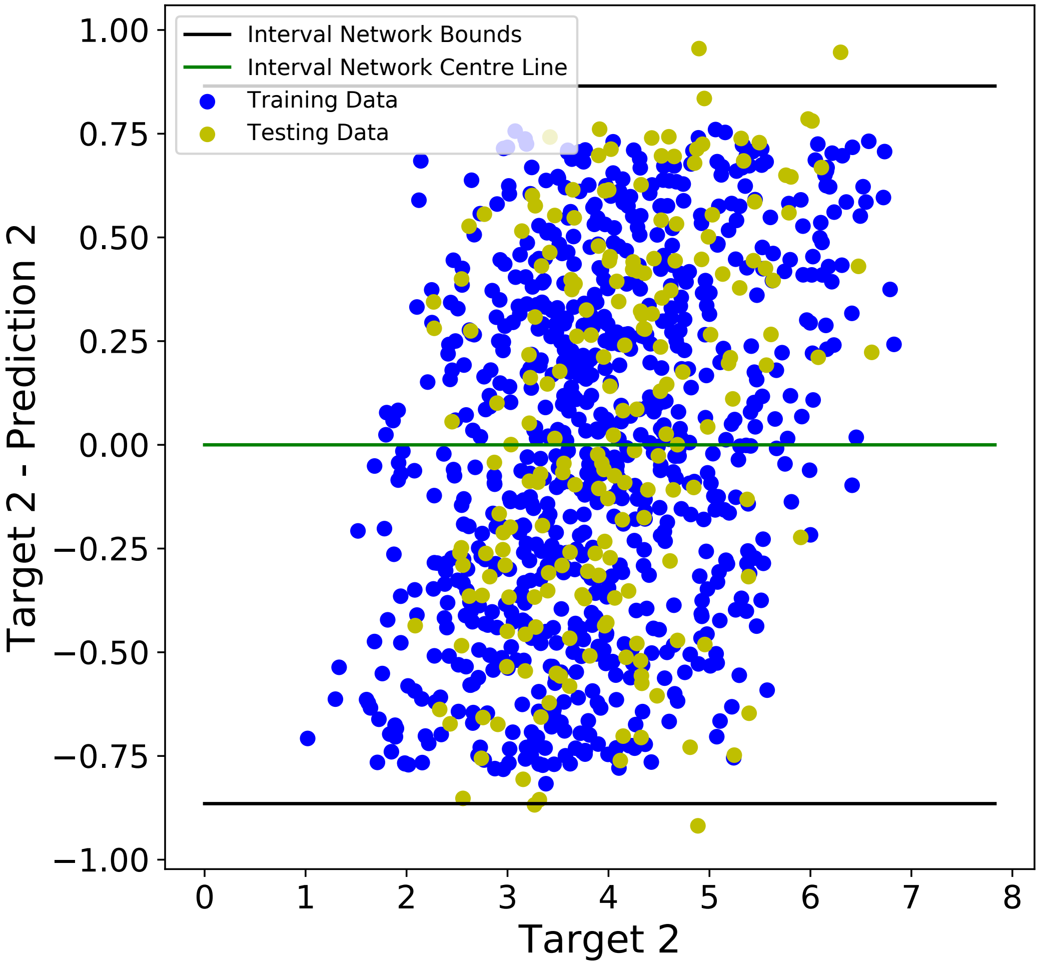

Figure 9.15: Plot of residuals for output 2 of multi-output interval neural network. Training set shown in as blue, test set shown in yellow.

Experiment	1 Const. Width	2 MSE
Minibatch size, M	200	200
Initial learning rate	0.01	0.01
Learning rate decay rate	0.9	0.9
Number of training epochs	15000	15000
Decay steps	1000	1000
ℓ_2 regularisation scale	N/A	N/A

Table 9.7: Hyper-parameters used in the numerical experiments for the concrete test dataset.

9.4.4 Realistic engineering test case

Description

The compressive strength of concrete is a nonlinear function of age and ingredients. Yeh [184] provides a database with 1030 experimental measurements of the compressive strength of concrete as a function of age and composition in kg/m^3 (cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, fine aggregate).* No information is provided about incertitude in the measurements, and therefore we are forced to process the data as it is given.

We wish to obtain robust bounds for the compressive strength of the concrete. This can be used for a worst case structural reliability analysis calculation. We replicate the architecture from Yeh [184] with our proposed algorithm and train a neural network with 1 hidden layer containing 8 neurons with hyperbolic tangent activation functions on the normalised dataset (transformed to have mean zero and unit variance).

We apply Algorithm 3 with the constant width loss from Eqn. 9.1 and $M = 200$. The weights are again initialised with the TensorFlow defaults. The hyper-parameters are summarised in Table 9.7.

Results

Annotated plots of the convergence for the upper and lower bounds (i.e. the maximum error at each step) are shown in Figure 9.16. The absolute error for the upper and lower bounds (i.e. the ‘residuals’) is plotted in Figure 9.17, and corresponds to an error width of $h = 14.6$ MPa, so the bounds had width 29.2 MPa. Using a train–test split ratio of 0.2, and

*Copyright Prof. I-Cheng Yeh

Experiment	1 Const. Width	2 MSE
Test points, N_t	206	206
Bound violating test points, N_v	8	N/A
$\bar{v}$	1.0×10^{-1}	N/A
$\underline{v}$	9.7×10^{-3}	N/A
Model half-width, h (MPa)	14.6	N/A
Root mean squared error (MPa)	N/A	5.85
Runtime (s)	145	146

Table 9.8: Results from the numerical experiments for the concrete test dataset.

the a posteriori frequentist analysis approach from Section 3.3.2 we calculate bounds on $V(\hat{z}_N)$ with confidence 0.999 for the trained interval neural network. The results compare favourably with other machine learning techniques [181]. The results are summarised in Table 9.8.

9.4.5 Outaouais benchmark dataset

Description

The Outaouais dataset was introduced in the Evaluating Predictive Uncertainty Challenge [145]. The dataset is for a regression problem with 37 features and 1 target variable. The dataset consists of 20000 training examples and 9000 test examples.

To predict the target, a heteroscedastic interval neural network was trained with Eqn. 9.19. The network architecture had two hidden layers with 150 and 20 neurons with ReLU activation functions. The weights were initialised with the TensorFlow defaults.

This was compared to a heteroscedastic maximum likelihood perceptron network (heteroscedastic MLP) [40], trained with the same network architecture and hyper-parameters. The hyper-parameters for both models are summarised in Table 9.9.

Results

Using the test dataset with the a posteriori frequentist analysis approach from Section 3.3.2, we calculate bounds on $V(\hat{z}_N)$ with confidence 0.999 for the trained interval neural net-

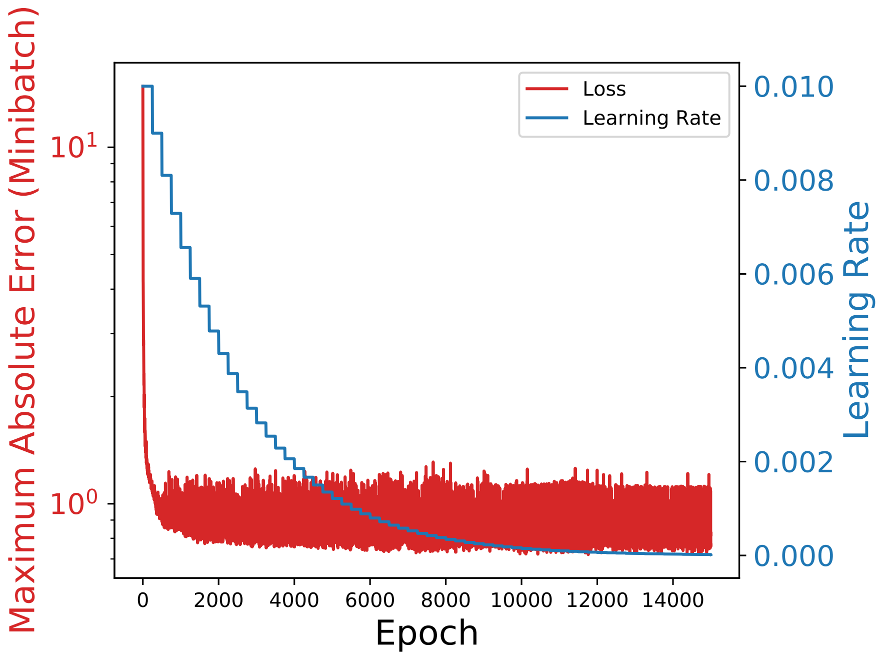

Figure 9.16: Plot of convergence of the interval neural network to predict concrete compressive strength.

Experiment	1 Heteroscedastic Interval Network	2 Heteroscedastic MLP
Minibatch size, M	200	200
Initial learning rate	0.001	0.01
Learning rate decay rate	0.995	0.995
Number of training epochs	2000	2000
Decay steps	200	200
ℓ_2 regularisation scale	N/A	N/A
Dropout rate	0.01	N/A

Table 9.9: Hyper-parameters used in the numerical experiments for the Outaouais dataset.

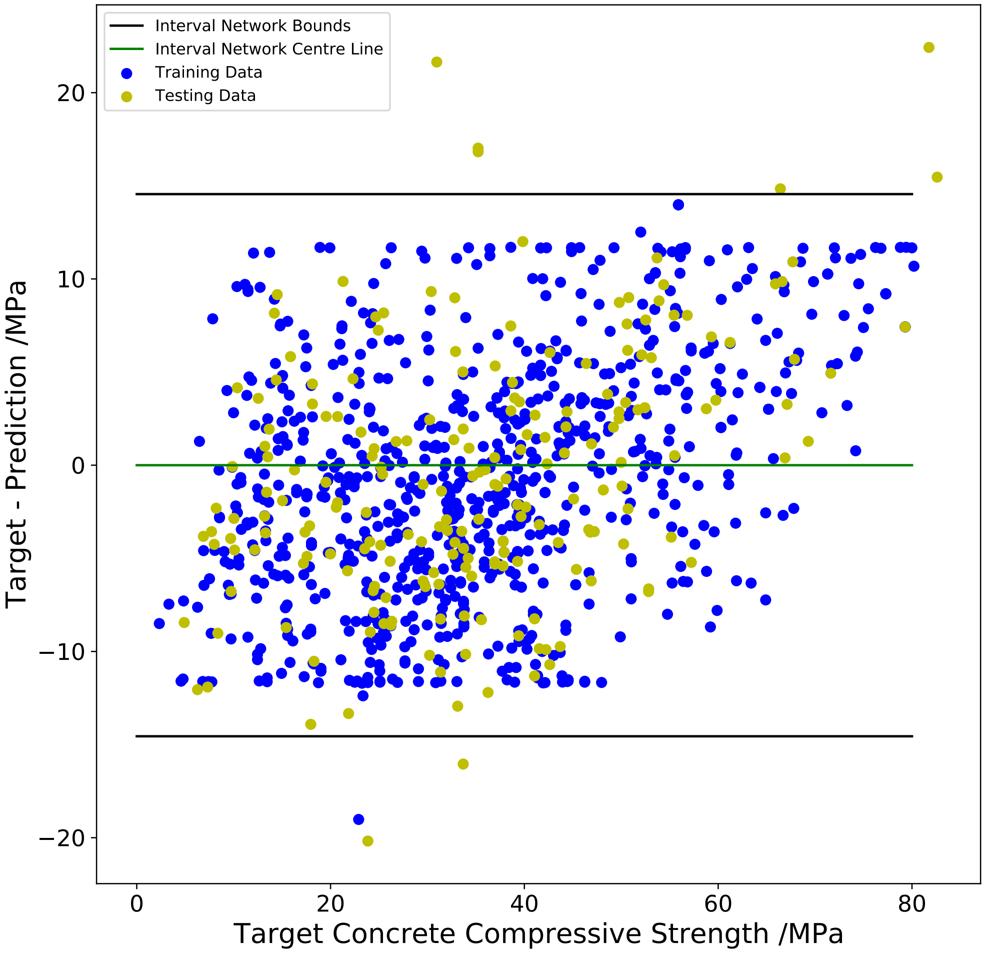

Figure 9.17: Plot of residuals (difference of predictions and targets) for interval neural network to predict concrete compressive strength. Model central line shown in green, and bounds shown in black. Training set shown in blue, test set shown in yellow.

Experiment	1 Heteroscedastic Interval Network	2 Heteroscedastic MLP
Test points, N_t	9000	9000
Bound violating test points, N_v	671	N/A
$\bar{v}$	6.6×10^{-2}	N/A
$\underline{v}$	8.3×10^{-2}	N/A
Model half-width, h	0.29	N/A
Normalised mean squared error, nMSE	N/A	0.038
Runtime (s)	691	705

Table 9.10: Results from the numerical experiments for the Outaouais dataset. The data variance used to compute the nMSE metric was 0.55.

work. The results are summarised in Table 9.10. The confidence bound on $V(\hat{z}_N)$ of the heteroscedastic interval network is superior to that of the heteroscedastic MLP when Chebyshev's inequality is used to produce a confidence bound from the mean squared error of the MLP.

9.5 Chapter summary

In this chapter, we have demonstrated how to create neural networks which quantify their uncertainty with interval predictions. In order to achieve scalability, the proposed technique relies upon techniques developed for modern deep learning applications, such as minibatch gradient descent. The proposed approach converges reliably and is not restricted to a specific architecture. Crucially, we avoid using explicit set inclusion relationships in the training process, which usually cause computational difficulties for practitioners of interval methods.

Since the model is not Bayesian, it is unnecessary to specify prior distributions, or to use complex variational inference implementations. Instead, the uncertainty is modelled using an interval which contains at least a specific proportion of the true output with near certainty.

The main contribution of this chapter is to provide a computationally feasible alternative to Bayesian models of uncertainty in neural networks, by allowing the neural network to be trained from data specified by ℓ_2 or ℓ_∞ balls, which the network is forced to include in its prediction interval. The theoretical contributions of this chapter could be applied to many convex models of uncertainty, and hence useful domain specific models could be derived from the work presented in this chapter.

Chapter 10

Conclusion

The research in this thesis set out to describe how uncertainty quantification can be accurately and reliably achieved for engineering systems, when the available data is of poor quality or if a limited quantity of data is available. Computational techniques for performing this simulation in a feasible computational time were discussed. The developed uncertainty quantification techniques were discussed in the context of reliability engineering for structures in the civil nuclear industry, however the potential applications of the developed techniques are much wider than this. This thesis is mostly focused on computational and statistical techniques, and hence the code to create interval predictor models, a major contribution of the thesis, has been made available as open source software. In this chapter, the main contributions proposed in this thesis will be summarised and recommendations for future research will be made.

10.1 Summary of conclusions

In Chapter 1, the motivation behind this thesis was explained and the direction of the research was introduced.

Chapter 2 presented a review of uncertainty models which model joint uncertainty of unknown variables (e.g. *generative* probabilistic models, convex sets and probability boxes). Techniques for general computation with such models are described. The construction of such models with or without experimental data was discussed.

Chapter 3 discussed techniques to create uncertainty models, where the uncertainty in one variable is dependent on the uncertainty in other variables. These models include

Bayesian regression, neural networks, Gaussian processes and interval predictor models. Techniques to validate the performance of these models is discussed.

Chapter 4 presented a state of the art review of techniques for reliability analysis, where the probability of failure of a system under the influence of uncertainty is calculated. Reliability measures were discussed for probabilistic models, convex set models and imprecise probability models. Efficient computational techniques were discussed for performing the simulation required to calculate the reliability of expensive black-box models of systems.

In Chapter 5, the analysis of a concrete containment from the nuclear industry was presented. The structural failure of the containment under pressure is modelled using an analytic equation derived from structural engineering principles. The properties of the containment were represented by random variables where only an upper bound on the coefficients of variation of the distribution parameters was available. Sensitivity analysis was applied to study the effect of changing these distribution parameters on the probability of failure of the system.

In Chapter 6, equations were provided to solve the problem described in Chapter 4 analytically, where the system's parameters are represented by probability boxes, for systems where the performance is described by a strength load relationship. This was achieved by using interval analysis to generalise the traditional analytical probabilistic equations used in such calculations. Then these developments were applied to analytically calculate the reliability of a containment structure, without using Monte Carlo simulation.

Chapter 7 described how the interval predictor models introduced in Chapter 3 could be used as a metamodel to obtain rigorous bounds on the failure probability calculated by Monte Carlo simulation, as described in Chapter 4, whilst reducing the computational time required for the simulation. Techniques to create interval predictor models which are better suited to modelling performance functions were described. For example, the interval predictor model can be forced to model the limit state function more accurately, and the basis used in the interval predictor model can be iteratively pruned to prevent overfitting. A case study is presented, where the performance of interval predictor models is compared to Kriging.

In Chapter 8, the results from Chapter 7 were generalised for the case where the system's parameters are described by probability box variables. Several sampling techniques were proposed and their efficiencies compared, including re-weighting based estimators and focal element propagation techniques. The performance of the technique was studied for test cases including a cantilever beam, a non-linear oscillator and a finite element model of a small satellite.

Chapter 9 demonstrated how non-convex interval predictor models could be trained for arbitrarily large datasets, with multiple prediction variables. This was achieved by developing new loss functions which can be minimised using minibatch stochastic gradient descent to reduce the computational complexity of training. The proposed training technique is applicable to cases where the training data is imprecise. The technique was applied to benchmark datasets, and the prediction of the compressive strength of concrete.

10.2 Recommendations

While traditional Bayesian methods address the issue of scarce and limited data in some sense, it is clear that the framework of imprecise probability offers greater flexibility when limited prior knowledge is available, or data is imprecise. Furthermore, the existing metamodelling techniques used to enhance the efficiency of Monte Carlo simulations require many assumptions to be made regarding the computational model. This is something that the proposed techniques avoid, either by use of analytic computation, or by use of interval predictor models. Hence, rigorous bounds can now be obtained on the failure probability of systems, or other quantities predicted by interval predictor models. This is something that will no doubt be of use in highly regulated engineering domains, where the performance of safety critical systems must be accurately quantified. Therefore, the presented research opens up many interesting research directions for future work.

The analytic probabilistic safety analysis approach for probability boxes presented in Chapter 6 lends itself very well to industrial application, because it does not require complex and time consuming simulations, and many systems can be expressed as a parallel or series combination of components. In its current state, it is likely that the work could be applied to the study of many similar systems in structural engineering. The wider application of imprecise probabilities in industrial probabilistic safety analysis is an exciting prospect. Similar approaches could be developed for other analytic probabilistic relationships used in engineering, as one simply has to intervalise the relevant expressions used in traditional probabilistic safety analysis. For example, more complex calculations are required to calculate the relevant failure probabilities of backup power systems in nuclear power plants. Currently the approach presented considers either independence or complete lack of knowledge on dependencies of failure events. It would be desirable if the approach could be modified to consider partial knowledge of dependencies between failure events.

The computational efficiency of the reliability analysis in Chapter 7 and Chapter 8 could be greatly enhanced if more efficient techniques were available to train interval predictor models, as metamodels or otherwise. Other metamodel techniques, e.g. AK-MCS [56], use active learning to reduce the number of samples required from the full model by specifying that more samples should be collected on the limit state surface of the computational model. By analogy, it would be useful if the interval predictor model could iteratively specify in which regions additional accuracy is required, and then collect samples accordingly. Recent progress in scenario optimisation indicates that such techniques may be viable, e.g. importance sampling estimators for estimating the reliability of a solution to a scenario program [12] and the FAST algorithm [38]. Furthermore, the iterative scenario approach proposed in Garatti and Campi [72], which reduces the number of samples required to guarantee the solution of a scenario program, appears to be particularly effective in numerical examples where the solution must be guaranteed with high probability — corresponding to rare failure probabilities in the algorithm proposed in this thesis.

The development of techniques to train interval neural networks for imprecise data in Chapter 9 is an attempt to incorporate interval incertitude into modern machine learning architectures, but no attempt was made to include Bayesian machine learning techniques in the framework.[*] Further development in this area could produce a variational method for training Bayesian neural networks on imprecise data, which would yield probability boxes to describe the posterior distribution of the network's weights. Chapter 9 describes how a loss function can be intervalised for imprecise data using Taylor expansions. It is also known that loss functions can be intervalised using interval arithmetic [137]. Therefore, one could also attempt to intervalise the evidence lower bound loss function from Bayesian neural networks. A similar algorithm could also be developed when the prior distribution for the weights was not precisely known.

It would be desirable to extend the approach to the problem of classification; currently the approach presented in Chapter 9 has only been applied to regression. In fact, preliminary results in Mirman et al. [117], indicate that the max error loss function can be applied to classification problems for a particular class of neural networks where the intervals can be propagated analytically through the network, i.e. without using a Taylor series linearisation of the loss, resulting in a model equivalent to a homoscedastic interval neural network for classification. However, in Mirman et al. [117] no theoretical justification is made for the

[*] Clearly, using weight regularisation in the networks for the numerical experiments has an obvious Bayesian maximum *a posteriori* interpretation, but this was not the focus of the experiments.

empirical finding that the proposed models generalise to data not seen in training; in this thesis the presentation of interval neural networks in the context of non-convex scenario optimisation provides such an explanation.

Bibliography

[1] T. Alamo, R. Tempo, A. Luque, and D. R. Ramirez. Randomized methods for design of uncertain systems: Sample complexity and sequential algorithms. *Automatica*, 52: 160–172, 2015. doi: 10.1016/j.automatica.2014.11.004.

[2] D. Altieri and E. Patelli. An efficient method for estimating conditional failure probabilities. In *The 8th International Workshop on Reliable Engineering Computing (REC2018), Liverpool, UK*, pages 163–168, 2018. URL `https://rec2018.uk/papers/REC2018-18_D.Altieri,%20E.%20Patelli__An%20efficient%20method%20for%20estimating%20conditional%20failure%20probabilities.pdf`.

[3] D. A. Alvarez. On the calculation of the bounds of probability of events using infinite random sets. *International journal of approximate reasoning*, 43(3):241–267, 2006. doi: 10.1016/j.ijar.2006.04.005.

[4] D. A. Alvarez. *Infinite random sets and applications in uncertainty analysis.* PhD thesis, Ph. D. Thesis, Leopold-Franzens Universität Innsbruck, Innsbruck, Austria, 2007.

[5] D. A. Alvarez, J. E. Hurtado, and J. Ramírez. Tighter bounds on the probability of failure than those provided by random set theory. *Computers & Structures*, 189: 101–113, 2017. doi: 10.1016/j.compstruc.2017.04.006.

[6] American Society of Civil Engineers. Seismic design criteria for structures, systems, and components in nuclear facilities (ASCE/SEI 43-05). 2005.

[7] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008. doi: 10.1007/s10994-007-5040-8.

[8] S.-K. Au and J. L. Beck. Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic engineering mechanics*, 16(4):263–277, 2001. doi: 10.1016/S0266-8920(01)00019-4.

[9] S.-K. Au and E. Patelli. Rare event simulation in finite-infinite dimensional space. *Reliability Engineering & System Safety*, 148:67–77, 2016. doi: 10.1016/j.ress.2015.11.012.

[10] M. S. Balch. A corrector for probability dilution in satellite conjunction analysis. In *18th AIAA Non-Deterministic Approaches Conference*, page 1445, 2016. doi: 10.2514/6.2016-1445.

[11] M. S. Balch, R. Martin, and S. Ferson. Satellite conjunction analysis and the false confidence theorem. *Proceedings of the Royal Society A*, 475(2227):20180565, 2019. doi: 10.1098/rspa.2018.0565.

[12] J. Barrera, T. Homem-de Mello, E. Moreno, B. K. Pagnoncelli, and G. Canessa. Chance-constrained problems and rare events: an importance sampling approach. *Mathematical Programming*, 157(1):153–189, 2016. doi: 10.1007/s10107-015-0942-x.

[13] M. Beer and E. Patelli. Editorial: Engineering analysis with vague and imprecise information. *Structural Safety Special Issue: Engineering Analyses with Vague and Imprecise Information*, 52, Part B(0):143, 2015. ISSN 0167-4730. doi: 10.1016/j.strusafe.2014.11.001. Engineering Analyses with Vague and Imprecise Information.

[14] Y. Ben-Haim. Convex models of uncertainty: Applications and implications. *Erkenntnis*, 41(2):139–156, 1994. doi: 10.1007/BF01128824.

[15] Y. Ben-Haim and I. Elishakoff. *Convex models of uncertainty in applied mechanics*, volume 25. Elsevier, 2013. doi: 10.1137/1035076.

[16] J. Berger. *The robust Bayesian viewpoint (with discussion). Robustness of Bayesian Analysis.* Amsterdam: North Holland, 1984.

[17] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer Science & Business Media, 1985. doi: 10.1007/978-1-4757-4286-2.

[18] J. O. Berger, E. Moreno, L. R. Pericchi, M. J. Bayarri, J. M. Bernardo, J. A. Cano, J. De la Horra, J. Martín, D. Ríos-Insúa, B. Betrò, et al. An overview of robust Bayesian analysis. *Test*, 3(1):5–124, 1994. doi: 10.1007/BF02562676.

[19] M. Berveiller, B. Sudret, and M. Lemaire. Stochastic finite element: a non intrusive approach by regression. *European Journal of Computational Mechanics/Revue Européenne de Mécanique Numérique*, 15(1-3):81–92, 2006. doi: 10.3166/remn.15.81-92.

[20] Z. W. Birnbaum. On the importance of different components in a multicomponent system. Technical report, Washington Univ Seattle Lab of Statistical Research, 1968.

[21] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

[22] S. Boyd and L. Vandenberghe. *Convex optimization.* Cambridge University Press, 2004.

[23] S. Brandt, M. Broggi, J. Hafele, C. G. Gebhardt, R. Rolfes, and M. Beer. Meta-models for fatigue damage estimation of offshore wind turbines jacket substructures. *Procedia engineering*, 199:1158–1163, 2017. doi: 10.1016/j.proeng.2017.09.292.

[24] J. Braverman, J. Xu, B. Ellingwood, C. Costantino, R. Morante, and C. Hofmayer. Evaluation of the seismic design criteria in ASCE/SEI standard 43-05 for application to nuclear power plants. Technical report, US Nuclear Regulatory Commission, Job Code N, 2007.

[25] K. Breitung. The geometry of limit state function graphs and subset simulation: Counterexamples. *Reliability Engineering & System Safety*, 182:98–106, 2019. doi: 10.1016/j.ress.2018.10.008.

[26] S. Bubeck et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015. doi: 10.1561/2200000050.

[27] C. G. Bucher and U. Bourgund. A fast and efficient response surface approach for structural reliability problems. *Structural safety*, 7(1):57–66, 1990. doi: 10.1016/0167-4730(90)90012-E.

[28] R. J. Budnitz, P. J. Amico, C. A. Cornell, W. J. Hall, R. P. Kennedy, J. W. Reed, and M. Shinozuka. Approach to the quantification of seismic margins in nuclear

power plants. Technical Report NUREG/CR-4334; UCID-20444, Lawrence Livermore National Lab., CA (USA), 1985.

[29] G. Calafiore and M. C. Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46, 2005. doi: 10.1007/s10107-003-0499-y.

[30] G. C. Calafiore. Random convex programs. *SIAM Journal on Optimization*, 20(6): 3427–3464, 2010. doi: 10.1137/090773490.

[31] M. C. Campi and A. Caré. Random convex programs with l_1-regularization: sparsity and generalization. *SIAM Journal on Control and Optimization*, 51(5):3532–3557, 2013. doi: 10.1137/110856204.

[32] M. C. Campi and S. Garatti. The exact feasibility of randomized solutions of uncertain convex programs. *SIAM Journal on Optimization*, 19(3):1211–1230, 2008. doi: 10.1137/07069821X.

[33] M. C. Campi and S. Garatti. Wait-and-judge scenario optimization. *Mathematical Programming*, 167(1):155–189, 2018. doi: 10.1007/s10107-016-1056-9.

[34] M. C. Campi, G. Calafiore, and S. Garatti. Interval predictor models: Identification and reliability. *Automatica*, 45(2):382–392, 2009. doi: 10.1016/j.automatica.2008.09.004.

[35] M. C. Campi, S. Garatti, and F. A. Ramponi. Non-convex scenario optimization with application to system identification. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 4023–4028. IEEE, 2015. doi: 10.1109/CDC.2015.7402845.

[36] M. C. Campi, S. Garatti, and F. A. Ramponi. A general scenario theory for nonconvex optimization and decision making. *IEEE Transactions on Automatic Control*, 63(12): 4067–4078, 2018. doi: 10.1109/TAC.2018.2808446.

[37] A. Carè and E. Camporeale. Chapter 4 - regression. In E. Camporeale, S. Wing, and J. R. Johnson, editors, *Machine Learning Techniques for Space Weather*, pages 71 – 112. Elsevier, 2018. ISBN 978-0-12-811788-0. doi: 10.1016/B978-0-12-811788-0.00004-4.

[38] A. Carè, S. Garatti, and M. C. Campi. Fastfast algorithm for the scenario technique. *Operations Research*, 62(3):662–671, 2014. doi: 10.1287/opre.2014.1257.

[39] A. Care, S. Garatti, and M. C. Campi. Scenario min-max optimization and the risk of empirical costs. *SIAM Journal on Optimization*, 25(4):2061–2080, 2015. doi: 10.1137/130928546.

[40] G. C. Cawley, N. L. Talbot, and O. Chapelle. Estimating predictive variances with kernel ridge regression. In *Machine Learning Challenges Workshop*, pages 56–77. Springer, 2005. doi: 10.1007/11736790˙5.

[41] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10(4):377–409, Dec 1993. ISSN 1432-0444. doi: 10.1007/BF02573985.

[42] J. Collet. Some remarks on rare-event approximation. *IEEE Transactions on Reliability*, 45(1):106–108, 1996. doi: 10.1109/24.488924.

[43] C. N. S. Commission et al. *Probabilistic Safety Assessment (PSA) for Nuclear Power Plants*. Canadian Nuclear Safety Commission, 2005.

[44] S. Coveney. samcoveney gp_emu_uqsa: First official release. Dec. 2016. doi: 10.5281/zenodo.215521.

[45] L. G. Crespo, D. P. Giesy, and S. P. Kenny. A unifying framework to uncertainty quantification of polynomial systems subject to aleatory and epistemic uncertainty. *Reliable Computing*, 17(2):97–127, 2012.

[46] L. G. Crespo, S. P. Kenny, and D. P. Giesy. Interval predictor models with a linear parameter dependency. *Journal of Verification, Validation and Uncertainty Quantification*, 1(1), 2016. doi: 10.1115/1.4032070.

[47] L. G. Crespo, S. P. Kenny, and D. P. Giesy. Staircase predictor models for reliability and risk analysis. *Structural Safety*, 75:35–44, 2018. doi: 10.1016/j.strusafe.2018.05.002.

[48] K. Csilléry, M. G. Blum, O. E. Gaggiotti, and O. François. Approximate bayesian computation (abc) in practice. *Trends in ecology & evolution*, 25(7):410–418, 2010. doi: 10.1016/j.tree.2010.04.001.

[49] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos. Distributional reinforcement learning with quantile regression. *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, 2017.

[50] M. De Angelis. *Efficient Random Set Uncertainty Quantification by means of Advanced Sampling Techniques.* PhD thesis, 2015.

[51] M. de Angelis, E. Patelli, and M. Beer. Advanced line sampling for efficient robust reliability analysis. *Structural safety*, 52:170–182, 2015. doi: 10.1016/j.strusafe.2014.10.002.

[52] A. Decadt, G. de Cooman, and J. De Bock. Monte carlo estimation for imprecise probabilities: Basic properties. *arXiv preprint arXiv:1905.09301*, 2019.

[53] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao. Optimal distributed online prediction. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 713–720, 2011.

[54] F. A. Diaz De la O. *Gaussian process emulators for the analysis of complex models in engineering.* PhD thesis, Swansea University, 2011.

[55] C. Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[56] B. Echard, N. Gayton, and M. Lemaire. Ak-mcs: an active learning reliability method combining kriging and monte carlo simulation. *Structural Safety*, 33(2):145–154, 2011. doi: 10.1016/j.strusafe.2011.01.002.

[57] I. Elishakoff and M. Ohsaki. *Optimization and anti-optimization of structures under uncertainty.* World Scientific, 2010. doi: 10.1142/p678.

[58] S. Eslami, D. J. Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018. doi: 10.1126/science.aar6170.

[59] G. Feng, E. Patelli, M. Beer, and F. P. Coolen. Imprecise system reliability and component importance based on survival signature. *Reliability Engineering & System Safety*, 150:116–125, 2016. doi: 10.1016/j.ress.2016.01.019.

[60] S. Ferson, V. Kreinovich, L. Ginzburg, D. S. Myers, and K. Sentz. Constructing probability boxes and dempster-shafer structures. Technical report, Technical report, Sandia National Laboratories, 2003.

[61] S. Ferson, W. L. Oberkampf, and L. Ginzburg. Validation of imprecise probability models. *Int. J. Reliab. Saf*, 3(1-3):3–22, 2009. doi: 10.1504/IJRS.2009.026832.

[62] S. Ferson, J. ORawe, and M. Balch. Computing with confidence: imprecise posteriors and predictive distributions. In *Vulnerability, Uncertainty, and Risks: Quantification, Mitigation, and Management*, pages 895–904. 2014. doi: 10.1061/9780784413609.091.

[63] T. Fetz. Efficient computation of upper probabilities of failure. In D. M. F. Christian Bucher, Bruce R. Ellingwood, editor, *Safety, Reliability, Risk, Resilience and Sustainability of Structures and Infrastructure*, pages 493–502. 12th International Conference on Structural Safety & Reliability, TU-Verlag Vienna, 2017.

[64] T. Fetz and M. Oberguggenberger. Imprecise random variables, random sets, and monte carlo simulation. *International Journal of Approximate Reasoning*, 78:252–264, 2016. doi: 10.1016/j.ijar.2016.06.012.

[65] P. I. Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[66] S. Freitag, W. Graf, and M. Kaliske. Recurrent neural networks for fuzzy data. *Integrated Computer-Aided Engineering*, 18(3):265–280, 2011. doi: 10.3233/ICA-2011-0373.

[67] J. Friedman, T. Hastie, and R. Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001. doi: 10.1007/BF02985802.

[68] Y. Gal. *Uncertainty in deep learning.* PhD thesis, PhD thesis, University of Cambridge, 2016.

[69] Y. Gal. Heteroscedastic dropout uncertainty. `https://github.com/yaringal/HeteroscedasticDropoutUncertainty`, 2016.

[70] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[71] Y. Gal, J. Hron, and A. Kendall. Concrete dropout. In *Advances in Neural Information Processing Systems*, pages 3581–3590, 2017.

[72] S. Garatti and M. C. Campi. Complexity-based modulation of the data-set in scenario optimization. In *2019 18th European Control Conference (ECC)*, pages 1386–1391, June 2019. doi: 10.23919/ECC.2019.8796160.

[73] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. Eslami, and Y. W. Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018.

[74] A. Gelman, H. S. Stern, J. B. Carlin, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 2013. doi: 10.1201/b16018.

[75] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL `http://proceedings.mlr.press/v9/glorot10a.html`.

[76] W. J. Gomes. Structural reliability analysis using adaptive artificial neural networks. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering*, 2019. doi: 10.1115/1.4044040.

[77] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[78] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *ICLR 2015*, 2014.

[79] B. J. Goodno and J. M. Gere. *Mechanics of Materials*. Cengage Learning, 2016.

[80] P. Green and S. Maskell. Estimating the parameters of dynamical systems from big data using sequential monte carlo samplers. *Mechanical Systems and Signal Processing*, 93:379–396, 2017. doi: 10.1016/j.ymssp.2016.12.023.

[81] E. R. Hansen. A generalized interval arithmetic. In *International Symposium on Interval Mathematics*, pages 7–18. Springer, 1975. doi: 10.1007/3-540-07170-9˙2.

[82] J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, pages 282–290. AUAI Press, 2013.

[83] L. Huang, B.-L. Zhang, and Q. Huang. Robust interval regression analysis using neural networks. *Fuzzy sets and systems*, 97(3):337–347, 1998. doi: 10.1016/S0165-0114(96)00325-9.

[84] X. Huang, J. Chen, and H. Zhu. Assessing small failure probabilities by ak–ss: an active learning method combining kriging and subset simulation. *Structural Safety*, 59:86–95, 2016. doi: 10.1016/j.strusafe.2015.12.003.

[85] Z. Huang and B. He. Volume of unit ball in an n-dimensional normed space and its asymptotic properties. *Journal of Shanghai University (English Edition)*, 12(2): 107–109, Apr 2008. ISSN 1863-236X. doi: 10.1007/s11741-008-0204-1.

[86] R. L. Iman. *Latin Hypercube Sampling.* New York: John Wiley & Sons Ltd, 2014. ISBN 9781118445112. doi: 10.1002/9781118445112.stat03803. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118445112.stat03803`.

[87] H. Ishibuchi, H. Tanaka, and H. Okada. An architecture of neural networks with interval weights and its application to fuzzy regression analysis. *Fuzzy Sets and Systems*, 57(1):27–39, 1993. doi: 10.1016/0165-0114(93)90118-2.

[88] E. T. Jaynes. *Probability theory: The logic of science.* Cambridge University Press, 2003.

[89] C. Jiang, W. Li, X. Han, L. Liu, and P. Le. Structural reliability analysis based on random distributions with interval parameters. *Computers & Structures*, 89(23): 2292–2302, 2011. doi: 10.1016/j.compstruc.2011.08.006.

[90] C. Jiang, J. Zheng, and X. Han. Probability-interval hybrid uncertainty analysis for structures with both aleatory and epistemic uncertainties: a review. *Structural and Multidisciplinary Optimization*, pages 1–18, 2017. doi: 10.1007/s00158-017-1864-4.

[91] S. Kaplan, V. M. Bier, and D. C. Bley. A note on families of fragility curves - is the composite curve equivalent to the mean curve? *Reliability Engineering & System Safety*, 43(3):257–261, 1994. doi: 10.1016/0951-8320(94)90029-9.

[92] K. Kapur and L. Lamberson. *Reliability in engineering design*. Wiley, 1977.

[93] D. R. Karanki, H. S. Kushwaha, A. K. Verma, and S. Ajit. Uncertainty analysis based on probability bounds (p-box) approach in probabilistic safety assessment. *Risk Analysis*, 29(5):662–675, 2009. doi: 10.1111/j.1539-6924.2009.01221.x. URL `https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1539-6924.2009.01221.x`.

[94] I. Kaymaz. Application of kriging method to structural reliability problems. *Structural Safety*, 27(2):133–151, 2005. doi: 10.1016/j.strusafe.2004.09.001.

[95] D. Kececioglu. *Robust engineering design-by-reliability with emphasis on mechanical components & structural reliability*, volume 1. DEStech Publications, Inc, 2003.

[96] A. Kendall and Y. Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017.

[97] R. P. Kennedy. Performance-goal based (risk informed) approach for establishing the sse site specific response spectrum for future nuclear power plants. *Nuclear Engineering and Design*, 241(3):648–656, 2011. doi: 10.1016/j.nucengdes.2010.08.001.

[98] R. P. Kennedy, C. Cornell, R. Campbell, S. Kaplan, and H. Perla. Probabilistic seismic safety study of an existing nuclear power plant. *Nuclear Engineering and Design*, 59(2):315–338, 1980. doi: 10.1016/0029-5493(80)90203-4.

[99] J. H. Kim, I.-K. Choi, and J.-H. Park. Seismic evaluation by using composite standard deviation. In *Transactions of the Korean Nuclear Society Autumn Meeting*, Jeju, Korea, 10 2010.

[100] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[101] A. Kolmogoroff. Confidence limits for an unknown distribution function. *The annals of mathematical statistics*, 12(4):461–463, 1941. doi: 10.1214/aoms/1177731684.

[102] A. N. Kolmogorov. *Foundations of the Theory of Probability*. Chelsea Publishing Company, New York, 1950.

[103] V. Kreinovich, J. Beck, C. Ferregut, A. Sanchez, G. Keller, M. Averill, and S. Starks. Monte-carlo-type techniques for processing interval uncertainty, and their engineering applications. In *Proceedings of the workshop on reliable engineering computing*, pages 15–17, 2004.

[104] V. Kreinovich, W. Luther, and E. D. Popova. Special issue on "uncertainty modeling and analysis with intervals: foundations, tools, applications". *Soft Computing*, 17(8): 1315–1317, Aug 2013. ISSN 1433-7479. doi: 10.1007/s00500-013-1004-z.

[105] D. G. Krige. A statistical approach to some basic mine valuation problems on the witwatersrand. *Journal of the Southern African Institute of Mining and Metallurgy*, 52(6):119–139, 1951.

[106] J. Kruschke. *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan*. Academic Press, 2014.

[107] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. *ICLR 2017*.

[108] M. J. Lacerda and L. G. Crespo. Interval predictor models for data with measurement uncertainty. In *American Control Conference (ACC), 2017*, pages 1487–1492. IEEE, 2017. doi: 10.23919/ACC.2017.7963163.

[109] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

[110] T. D. Le Duy, D. Vasseur, L. Dieulle, C. Bérenguer, and M. Couplet. Representation of parameter uncertainty with evidence theory in probabilistic risk assessment. In *Proceeding of the Workshop on the Theory of Belief Functions, Brest, France*, 2010.

[111] X. Liu, L. Yin, L. Hu, and Z. Zhang. An efficient reliability analysis approach for structure based on probability and probability box models. *Structural and Multidisciplinary Optimization*, 56(1):167–181, 2017. doi: 10.1007/s00158-017-1659-7.

[112] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *ICML 2017 Workshop on Principled Approaches to Deep Learning*, 2017.

[113] K. Makino and M. Berz. Taylor models and other validated functional inclusion methods. *International Journal of Pure and Applied Mathematics*, 6:239–316, 2003.

[114] S. Maskell. A Bayesian approach to fusing uncertain, imprecise and conflicting information. *Information Fusion*, 9(2):259–277, 2008. doi: 10.1016/j.inffus.2007.02.003.

[115] R. E. Melchers and A. T. Beck. *Structural reliability analysis and prediction.* John Wiley & Sons, 2018.

[116] D. Meyer. Notes on maximization of inner products over norm balls, 2017. URL `http://www.1-4-5.net/~dmm/ml/sgn.pdf`.

[117] M. Mirman, T. Gehr, and M. Vechev. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pages 3575–3583, 2018.

[118] M. Modarres and I. S. Kim. *Handbook of Nuclear Engineering*, chapter Deterministic and Probabilistic Safety Analysis, pages 1739–1812. Springer US, Boston, MA, 2010. doi: 10.1007/978-0-387-98149-9˙15.

[119] M. Modarres, M. P. Kaminskiy, and V. Krivtsov. *Reliability engineering and risk analysis: a practical guide.* CRC press, 2016. doi: 10.1016/S0951-8320(02)00008-X.

[120] V. Montgomery. *New statistical metliods in risk assessment by probability bounds.* PhD thesis, Durham University, 2009.

[121] R. E. Moore, R. B. Kearfott, and M. J. Cloud. *Introduction to interval analysis.* Society for Industrial and Applied Mathematics, 2009. doi: 10.1137/1.9780898717716.bm.

[122] H. Nagaraja. Order statistics from discrete distributions. *Statistics: a journal of theoretical and applied statistics*, 23(3):189–216, 1992. doi: 10.1080/02331889208802365.

[123] R. M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012. doi: 10.1007/978-1-4612-0745-0.

[124] N. S. Nedialkov, V. Kreinovich, and S. A. Starks. Interval arithmetic, affine arithmetic, Taylor series methods: Why, what next? *Numerical Algorithms*, 37(1-4):325–336, 2004. doi: 10.1023/B:NUMA.0000049478.42605.cf.

[125] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, pages 841–848, 2002.

[126] NIST/SEMATECH. e-handbook of statistical methods. `http://www.itl.nist.gov/div898/handbook`, 2012.

[127] J. Oakley and A. O'Hagan. SHELF: The Sheffield Elicitation Framework (Version 2.0). School of Mathematics and Statistics, University of Sheffield, 2010. URL `http://www.tonyohagan.co.uk/shelf/`.

[128] I. Osband. Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout. In *Proceedings of the NIPS* 2016 Workshop on Bayesian Deep Learning*, 2016.

[129] H. Panayirci. Efficient solution for galerkin-based polynomial chaos expansion systems. *Advances in Engineering Software*, 41(12):1277–1286, 2010. doi: 10.1016/j.advengsoft.2010.09.004.

[130] E. Patelli. *Handbook of Uncertainty Quantification*, chapter COSSAN: A Multidisciplinary Software Suite for Uncertainty Quantification and Risk Management, pages 1–69. Springer International Publishing, Cham, 2016. ISBN 978-3-319-11259-6. doi: 10.1007/978-3-319-11259-6˙59-1.

[131] E. Patelli, H. J. Pradlwarter, and G. I. Schuëller. Global sensitivity of structural variability by random sampling. *Computer Physics Communications*, 181(12):2072–2081, 2010. doi: 10.1016/j.cpc.2010.08.007.

[132] E. Patelli, H. M. Panayirci, M. Broggi, B. Goller, P. Beaurepaire, H. J. Pradlwarter, and G. I. Schuëller. General purpose software for efficient uncertainty management of large finite element models. *Finite elements in analysis and design*, 51:31–48, 2012. doi: 10.1016/j.finel.2011.11.003.

[133] E. Patelli, D. A. Alvarez, M. Broggi, and M. d. Angelis. Uncertainty management in multidisciplinary design of critical safety systems. *Journal of Aerospace Information Systems*, 12(1):140–169, 2014. doi: 10.2514/1.I010273.

[134] E. Patelli, M. Broggi, S. Tolo, and J. Sadeghi. Cossan software: A multidisciplinary and collaborative software for uncertainty quantification. In *2nd ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering*, number UNCECOMP 2017, 2017. doi: 10.7712/120217.5364.16982.

[135] E. Patelli, G. Feng, F. Coolen, and T. Coolen-Maturi. Simulation methods for system reliability using survival signature. *Reliability Engineering & System Safety*, 167: 327–337, 2017. doi: 10.1016/j.ress.2017.06.018.

[136] E. Patelli, S. Tolo, H. George-Williams, J. Sadeghi, R. Rocchetta, M. de Angelis, and M. Broggi. Opencossan 2.0: an efficient computational toolbox for risk, reliability and resilience analysis. In *Proceedings of the joint ASCE ICVRAM ISUMA UNCERTAINTIES conference*, 2018. URL `http://icvramisuma2018.org/cd/web/PDF/ICVRAMISUMA2018-0022.PDF`.

[137] R. E. Patiño-Escarcina, B. R. C. Bedregal, and A. Lyra. Interval computing in neural networks: One layer interval neural networks. In *International Conference on Intelligent Information Technology*, pages 68–75. Springer, 2004. doi: 10.1007/978-3-540-30561-3˙8.

[138] M. Pellissetti, H. Keler, F. Laudarin, D. Altieri, and E. Patelli. Statistical analysis of impact forces and permanent deformations of fuel assembly spacer grids in the context of seismic fragility. In *Transactions, SMiRT-24*, 2017.

[139] G. Perret, S. Rahman, and D. Wicaksono. Deterministic sampling and gaussian process metamodel approach applied to a station black out accident model. In *ANS Best Estimate Plus Uncertainty International Conference (BEPU 2018)*, 2018.

[140] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. Numeric recipes in c: the art of scientific computing, 1992.

[141] N. Prinja and D. Shepherd. Theory and practice. *Professional Engineering Publishing*, 2003.

[142] N. Prinja, K. Kamatam, and J. Curley. Predicting pre-stressed concrete containment capacity. *SMiRT, Div-: Paper ID*, 628, 2011.

[143] N. K. Prinja, A. Ogunbadejo, J. Sadeghi, and E. Patelli. Structural reliability of pre-stressed concrete containments. *Nuclear Engineering and Design*, December 2016. doi: 10.1016/j.nucengdes.2016.11.036.

[144] Z. Qiu, D. Yang, and I. Elishakoff. Probabilistic interval reliability of structural systems. *International Journal of Solids and Structures*, 45(10):2850–2860, 2008. doi: 10.1016/j.ijsolstr.2008.01.005.

[145] J. Quinonero-Candela, C. E. Rasmussen, F. Sinz, O. Bousquet, and B. Schölkopf. Evaluating predictive uncertainty challenge. In *Machine Learning Challenges Workshop*, pages 1–27. Springer, 2005. doi: 10.1007/11736790˙1.

[146] R. Rackwitz. Principles and methods for a practical probabilistic approach to structural safety. *Sub-Committee for First Order Reliability Concepts for Design Codes of the Joint CEBCECM-CIB-FIB-IABSE Committee on Structural Safety*, 1975.

[147] R. Rocchetta and E. Patelli. Stochastic analysis and reliability-cost optimization of distributed generators and air source heat pumps. In *System Reliability and Safety (ICSRS), 2017 2nd International Conference on*, pages 31–35. IEEE, 2017. doi: 10.1109/ICSRS.2017.8272792.

[148] C. M. Rocco and J. A. Moreno. Fast monte carlo reliability evaluation using support vector machine. *Reliability Engineering & System Safety*, 76(3):237–243, 2002. doi: 10.1016/S0951-8320(02)00015-7.

[149] S. Ross. *A first course in probability*. Pearson, 2014.

[150] R. Y. Rubinstein and D. P. Kroese. *Simulation and the Monte Carlo method*, volume 10. John Wiley & Sons, 2016.

[151] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[152] L. Rüschendorf. Fréchet-bounds and their applications. *Advances in probability distributions with given marginals*, pages 151–187, 1991. doi: 10.1007/978-94-011-3466-8˙9.

[153] J. Sadeghi. PyIPM, 2018. URL `https://github.com/JCSadeghi/PyIPM`.

[154] J. Sadeghi, M. de Angelis, and E. Patelli. Frequentist history matching with interval predictor models. *Applied Mathematical Modelling*, 61:29–48, September 2018. doi: 10.1016/j.apm.2018.04.003.

[155] A. Saltelli, S. Tarantola, F. Campolongo, and M. Ratto. *Sensitivity Analysis in Practice: A Guide to Assessing Scientific Models.* John Wiley & Sons, 2004. doi: 10.1002/0470870958.

[156] F. J. Samaniego. *System signatures and their applications in engineering reliability*, volume 110. Springer Science & Business Media, 2007. doi: 10.1007/978-0-387-71797-5.

[157] R. Schöbi. *Surrogate models for uncertainty quantification in the context of imprecise probability modelling.* PhD thesis, ETH Zurich, 2017.

[158] R. Schöbi and B. Sudret. Structural reliability analysis for p-boxes using multi-level meta-models. *Probabilistic Engineering Mechanics*, 48:27–38, 2017. doi: 10.1016/j.probengmech.2017.04.001.

[159] G. I. Schuëller and R. Stix. A critical appraisal of methods to determine failure probabilities. *Structural Safety*, 4(4):293–309, 1987. doi: 10.1016/0167-4730(87)90004-X.

[160] S. Shao and Y. Murotsu. Structural reliability analysis using a neural network. *JSME International Journal Series A Solid Mechanics and Material Engineering*, 40(3): 242–246, 1997. doi: 10.1299/jsmea.40.242.

[161] B. Spencer, J. Petti, and D. Kunsman. Risk-informed assessment of degraded containment vessels. Technical report, 2006.

[162] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[163] J. Stolfi and L. H. de Figueiredo. An introduction to affine arithmetic. *Trends in Applied and Computational Mathematics*, 4(3):297–312, 2003.

[164] B. Sudret. Global sensitivity analysis using polynomial chaos expansions. *Reliability engineering & system safety*, 93(7):964–979, 2008. doi: 10.1016/j.ress.2007.04.002.

[165] B. Sudret. Meta-models for structural reliability and uncertainty quantification. In *"Asian-Pacific Symposium on Structural Reliability and its Applications"*, pages 1–24, 2012.

[166] H.-l. Sun and W.-x. Yao. The basic properties of some typical systems' reliability in interval form. *Structural Safety*, 30(4):364–373, 2008. doi: 10.1016/j.strusafe.2007.05.003.

[167] C. Sundararajan. Expert opinion in probabilistic structural mechanics. In *Probabilistic Structural Mechanics Handbook*, pages 261–279. Springer, 1995. doi: 10.1007/978-1-4615-1771-9˙12.

[168] C. R. Sundararajan. *Probabilistic structural mechanics handbook: theory and industrial applications*. Springer Science & Business Media, 2012. doi: 10.1007/978-1-4615-1771-9.

[169] A. Tarantola. *Inverse problem theory and methods for model parameter estimation*, volume 89. SIAM, 2005. doi: 10.1137/1.9780898717921.

[170] S. Tarantola, D. Gatelli, and T. A. Mara. Random balance designs for the estimation of first order global sensitivity indices. *Reliability Engineering & System Safety*, 91 (6):717–727, 2006. doi: 10.1016/j.ress.2005.06.003.

[171] S. Tolo, T. V. Santhosh, G. Vinod, U. Oparaji, and E. Patelli. Uncertainty quantification methods for neural networks pattern recognition. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2017. doi: 10.1109/SSCI.2017.8285163.

[172] S. Tolo, X. Tian, N. Bausch, V. Becerra, T. Santhosh, G. Vinod, and E. Patelli. Robust on-line diagnosis tool for the early accident detection in nuclear power plants. *Reliability Engineering & System Safety*, 186:110–119, 2019. doi: 10.1016/j.ress.2019.02.015.

[173] M. C. Troffaes. Imprecise Monte Carlo simulation and iterative importance sampling for the estimation of lower previsions. *International Journal of Approximate Reasoning*, 101:31–48, 2018. doi: 10.1016/j.ijar.2018.06.009.

[174] J. K. Uhlmann. *Dynamic map building and localization: New theoretical foundations.* PhD thesis, University of Oxford Oxford, 1995.

[175] US Nuclear Regulatory Commission. Guidance on the treatment of uncertainties associated with pras in risk-informed decision making. *NUREG-1855*, March 2009.

[176] US Nuclear Regulatory Commission. Probabilistic risk assessment and regulatory decision making: Some frequently asked questions. *NUREG-2201*, September 2016.

[177] V. N. Vapnik and A. Y. Chervonenkis. *On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities*, pages 11–30. Springer International Publishing, Cham, 2015. ISBN 978-3-319-21852-6. doi: 10.1007/978-3-319-21852-6˙3.

[178] F. A. Viana and R. T. Haftka. Probability of failure uncertainty quantification with kriging. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA*, page 1853, 2012. doi: 10.2514/6.2012-1853.

[179] E. A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, pages 153–158. Ieee, 2000. doi: 10.1109/ASSPCC.2000.882463.

[180] S. Wang, Y. Chen, A. Abdou, and S. Jana. Mixtrain: Scalable training of formally robust neural networks. *arXiv preprint arXiv:1811.02625*, 2018.

[181] S. Wenkel. Revisiting machine learning datasets - concrete compressive strength // simonwenkel.com, 2019. URL `https://www.simonwenkel.com/2018/08/08/revisiting_ml_datasets_concrete_compressive_strength.html`.

[182] Wikipedia contributors. Sipta — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=SIPTA&oldid=884781926`, 2019. [Online; accessed 15-April-2019].

[183] S. S. Wilks. Determination of sample sizes for setting tolerance limits. *The Annals of Mathematical Statistics*, 12(1):91–96, 1941. doi: 10.1214/aoms/1177731788.

[184] I.-C. Yeh. Modeling of strength of high-performance concrete using artificial neural networks. *Cement and Concrete research*, 28(12):1797–1808, 1998. doi: 10.1016/S0008-8846(98)00165-3.

[185] L. Zadeh. On the validity of dempsters rule of combination, memo m 79/24. *Univ. of California, Berkeley*, 74, 1979.

[186] C. Zhang and Z. Zhang. Improving multiview face detection with multi-task deep convolutional neural networks. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1036–1041. IEEE, 2014. doi: 10.1109/WACV.2014.6835990.

[187] H. Zhang, R. L. Mullen, and R. L. Muhanna. Interval monte carlo methods for structural reliability. *Structural Safety*, 32(3):183–190, 2010. doi: 10.1016/j.strusafe.2010.01.001.

[188] H. Zhang, Y. Li, J. Xiao, and T. Jordan. Preliminary study of condensation heat transfer uncertainty quantification using cfd code gasflow-mpi. In *ANS Best Estimate Plus Uncertainty International Conference (BEPU 2018)*, 2018.