Numerical Modelling and Uncertainty Quantification of Biodiesel Filters

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

Peter O. Hristov

August 2, 2018

Abstract

Numerical Modelling and Uncertainty Quantification of Biodiesel Filters

Peter O. Hristov

This dissertation explores the design and analysis of computer models for filters used to separate water from biodiesel. Regulations concerning air pollution and increasing fossil fuel scarcity mandate the transition towards biofuels. Moreover, increasingly stringent standards for fuel cleanliness are introduced continually. Biodiesel exhibits strong affinity towards water, which makes its separation from the fuel challenging. Water in the fuel can cause problems, ranging from reduced performance to significant damage to the equipment. A model of the filter is needed to substitute costly or impractical laboratory experiments and to enable the systematic studies of coalescence processes. These computational experiments provide a means for designing filtration equipment with optimal separation efficiency and pressure drop.

The coalescence process is simulated using the lattice Boltzmann modelling framework. These models offer several advantages over conventional computational fluid dynamics solvers and are commonly used for the simulation of multiphase flows. Different versions of lattice Boltzmann models in two and three dimensions are created and used in this work.

Complex computer models, such as those employed in this dissertation are considered expensive, in that their running times may prohibit any type of code analysis which requires many evaluations of the simulator to be performed. To alleviate this problem, a statistical metamodel known as a Gaussian process emulator is used. Once the computational cost of the model is reduced, uncertainty quantification methods and in particular sensitivity and reliability analyses are used to study its performance.

Tools and packages for industrial use are developed in this dissertation to enable the practical application of the studies conducted in it.

Declaration

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed .... (candidate)
Date ....

STATEMENT 1
This thesis is the result of my own investigations, except where otherwise stated. Where correction services have been used, the extent and nature of the correction is clearly marked in a footnote(s).
Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed .... (candidate)
Date ....

Acknowledgements

I must say that during my PhD, I met many excellent people who have either given me ideas, helped me solve a problem or have offered support and encouragement. As a result, this short acknowledgement is by no means exhaustive and I apologize beforehand to anyone that I have missed from it.

Firstly, I would like to thank my primary supervisor Dr Alejandro Díaz DelaO, or as I have always called him - Alex. He not only invited me to work with him on the project, but has taught me how to do research, be critical of my work, write and present about it. Alex has supported me with ideas, advice and patience and for the past four years has not just been a supervisor, but a friend as well!

I would also like to express my gratitude towards Dr Krzysztof Kubiak who has supported me in developing most of the software tools used for this work and has offered priceless advice and guidance through my work.

This project would not have been possible without the valuable support of Dr Umer Farooq who has had the patience and expert knowledge to help me steer my efforts in a meaningful direction. I would also like to say a big thank you to everyone from Parker Hannifin with whom I had the pleasure to work and exchange ideas - Phil Core, Premilla Nair and Lana Ferrari. I am also grateful to Parker Hannifin for the financial support they have provided for the project.

I would like to thank Dr Mark Wilson and Dr Anas Batou for reading this slightly-longer-than-expected thesis on such short notice and providing very useful comments and suggestions.

My time in the Risk Institute would not have been the same without the following people: Alfredo Garbuno Iñigo, who has managed to bring complex mathematics down to engineering level on more than one occasion; Dr Edoardo Patelli, who has provided support with computational issues and has proven to be a great football team mate; Dr Peter Green, Prof Ivan Au, and Dr Marco De Angelis who have provided advice and guidance on different topics. Thank you for making my time as a PhD unforgettable!

Through my time in the university, I had the opportunity to demonstrate for a wide variety of modules and I would like to thank everyone involved in the process for the excellent opportunity. I really appreciate the support of the technical team in the School of Engineering who helped in many different ways - Steve Bode, Andy Lyons and Jim Evans.

I have received so much from people outside the university too. Firstly, I would like to deeply thank my best friend and closest supporter - my wonderful wife, Plami - for putting up with all the late night work, travelling and absent-mindedness on my part, and for not giving up believing in me! Thank you, my love! I would also like to thank my loving family, Ognyan, Silvana, Polina and Lachezar who encouraged me to pursue a PhD and kept believing in me.

Last, but by no means least, I would like to thank The One to whom I owe everything. The One who has brought me thus far, will keep leading me and will always be by my side - to my Lord be the glory, the honour and the power, forever!

Contents

List of Figures

List of Tables

List of Acronyms

ARD	Automatic relevance determination
BC	Boundary condition
BGK	Bhatnagar-Gross-Krook
BLAS	Basic linear algebra subprograms
BPR	Bypass ratio
BTE	Boltzmann transport equation
CA	Cellular automata
CCDF	Complementary cumulative distribution function
CDF	Cumulative distribution function
CFD	Computational fluid dynamics
CLSVOF	Coupled level set and volume of fluid
CLT	Cross-laminated timber
CO	Carbon monoxide
CO2	Carbon dioxide
CPU	Central processing unit
CT	Computer tomography

DBSCAN	Density-based spatial clustering for applications with noise
DMC	Direct Monte Carlo
EC-LBE	Enhanced collision lattice Boltzmann equation
EI	Expected improvement
EOS	Equation of state
FE	Finite element
FHP	Frisch-Hasslacher-Pomeau
FPR	Fan pressure ratio
GA	Genetic algorithm
GB	Gigabyte
GP	Gaussian process
GPE	Gaussian process emulator
GPSS	Gaussian process subset simulation
GPU	Graphics processing unit
GSA	Global sensitivity analysis
GUI	Graphical user interface
HC	Hydrocarbon
HPP	Hardy-de Pazzis-Pomeau
IFT	Interfacial tension
IPE	Individual prediction error
LAPACK	Linear algebra package
LB	Lattice Boltzmann
LBE	Lattice Boltzmann equation
LBGK	Lattice Bhatnagar-Gross-Krook
LBM	Lattice Boltzmann method
LGCA	Lattice gas cellular automata
LHS	Latin hypercube sampling
LS	Level set

LSA	Local sensitivity analysis
MB	Megabyte
MCMC	Markov chain Monte Carlo
MCMP	Multi-component multiphase
MLE	Maximum likelihood estimation
MRT	Multi-relaxation time
NN-LBE	Nonlinear lattice Boltzmann equation
NOx	Nitrous oxides
OpenMP	Open Multi-processing
OPR	Overall pressure ratio
PCA	Principal component analysis
PDE	Partial differential equation
PDF	Probability density function
PI	Pair interaction
PSD	Pore size distribution
PSO	Particle swarm optimisation
QF	Quality factor
QL-LBE	Quasi-linear lattice Boltzmann equation
QQ	Quantile-quantile
RBDO	Reliability-based design optimisation
RMSE	Root-mean-square error
RVE	Representative volume element
SA	Sensitivity analysis
SBO	Surrogate-based optimisation
SCMP	Single component multiphase
SEM	Scanning electron microscopy
SLST	Sea-level static thrust
SPH	Smoothed particle hydrodynamics

TC	Thermodynamic consistency
UK	United Kingdom
ULSD	Ultra-low sulfur diesel
UPE	Uncorrelated prediction errors
VOF	Volume of fluid

CHAPTER 1

Introduction

1.1 Motivation

Mankind's existence depends on energy. More precisely, it depends on satisfying certain energy requirements. Food and sunlight are the prime sustainers of life on Earth. People, however, require a lot more energy than is naturally provided. Fire and artificial light are two classic examples. In modern days the demand is even higher - the use of fuel for different modes of transport, electricity for kitchen appliances and battery power for hand-held devices. Historically, the world has depended on fossil fuels since antiquity, but their use became ubiquitous throughout the Industrial revolution. Steam trains and ships, looms and whole factories were bound by the constant and stable supply of coal. Later on, in the mid-1800s the first oil wells were discovered and thus petroleum and all of its derivatives came into light[1]. A large portion of the world's electricity, production, trans-

[1] Oil was utilised much earlier, but its mass production and use began around 1854.

portation and heating still relies on burning some type of fossil fuel. However, none of the traditional sources of energy will last forever. Some studies [1, 2] give projected time scales for the depletion of fossil fuel reserves. These sources do not give a definitive depletion date, but they both point to the same conclusion: the depletion of natural resources is imminent. Even if fossil fuels were not running out their effect on climate change is demonstrable [3].

These facts have already been recognized and the world is systematically moving toward renewable energy sources. These include solar power, wind harvesting and tidal energy generation, among others. It is in this context that biofuels have gained popularity. A biofuel is a fuel which is produced by means different than geological processes, distinguishing them from fossil fuels. There are several common types of biofuel, but this research is mainly focused on *biodiesel* and the challenges posed by its efficient use and widespread adoption.

1.2 Biodiesel

Biodiesel is a diesel fuel. This is to say that in its use, it is ignited through compression rather than by a spark source. In order to present a more complete discussion about biodiesel, the role of petroleum diesel, also known as petrodiesel, is briefly examined.

Petrodiesel is by far the most common diesel fuel. Engines running on diesel are widely used due to their superior fuel efficiency and torque, when compared to similarly sized gasoline engines. These characteristics make them a suitable choice for powering heavy vehicles, ships and electric generators.

Biodiesel inherits these qualities and more. The main method for producing biodiesel from animal or vegetable fats is called *transesterification*. Essentially, transesterification is the catalysed reaction of the animal or vegetable fat with alcohol (most commonly ethanol or methanol) to form esters - crude biodiesel.

This product is then refined to arrive at pure biodiesel, designated B100. This fuel can be used in diesel engines with caution, due the solvent properties of B100 which can damage the tank and fuel lines. However, when mixed with petrodiesel, biodiesel can be readily used in unmodified engines. Currently B20 (20 % biodiesel and 80 % petrodiesel) is in widespread use with significant benefits. Biodiesel possesses many advantages including:

- Renewability - biodiesel is a by-product of agricultural or recycled resources.
- Low emissions - biodiesel contains no sulphur, has low CO, HC and greenhouse gases emissions, and has approximately no CO_2 footprint, as most CO_2 emitted during its use is balanced out by the CO_2 absorbed in its production.
- High lubricity - it helps keep the engine wear down.
- Good solvent properties - it clears build up and deposits from tanks and fuel lines.

For more information on the history, development and characteristics of biodiesel, the reader is referred to [4].

1.3 The problem

The process of transesterification produces a substance consisting of long-chain fatty acid esters [4]. These esters are highly hygroscopic, which leads to biodiesel having an increased affinity towards water, compared to regular diesel. The amount of water dissolved in the fuel depends on a number of factors [5]. Water can enter the fuel during any stage of its life cycle - production, storage, transportation and use. The contamination is most severe during production, but at the end of this stage the majority of water is mechanically separated from the

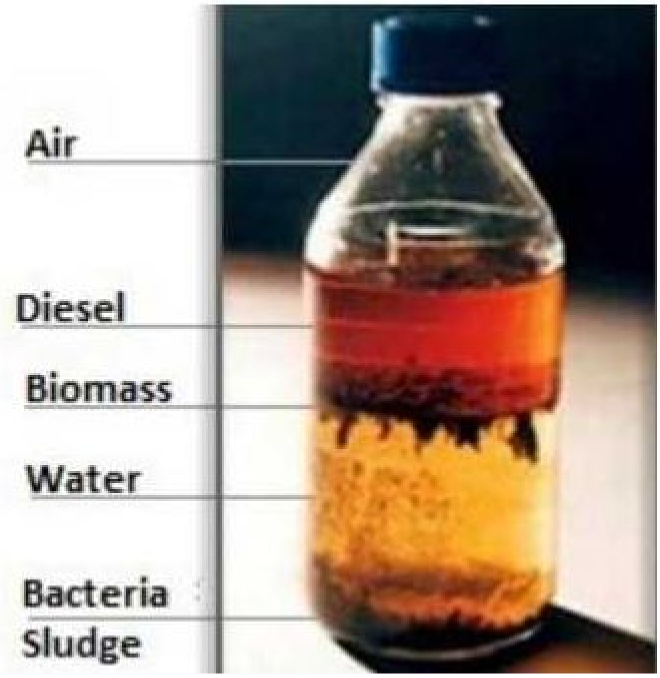
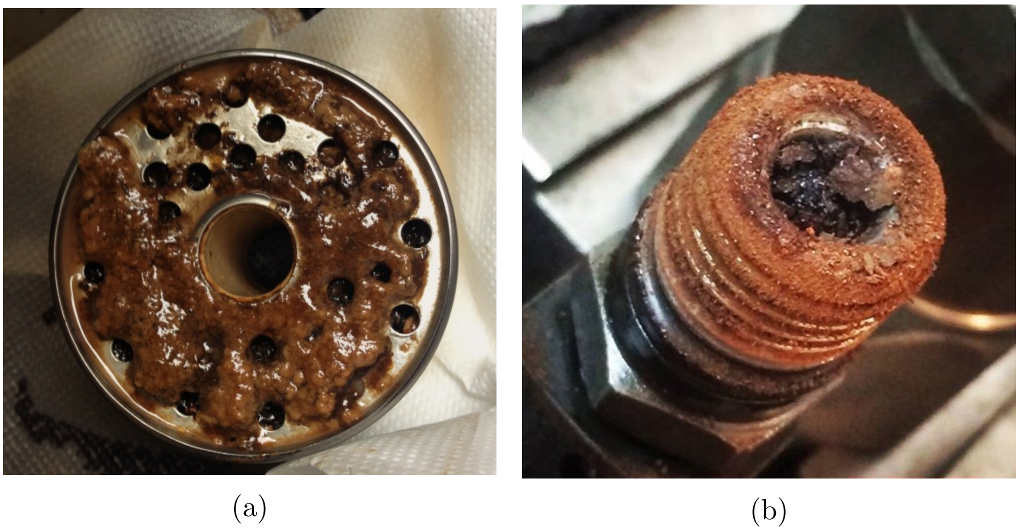


Figure 1.1: Diesel contaminated with water. The lighter diesel can be seen on top followed by the heavier water. Biological growth is promoted at the fuel/water interface with some the sludge sunk to the bottom of the specimen (Source: www.ehlsolutions.com).

fuel, typically using a centrifuge [6]. *Dissolved* water, which remains after the initial cleaning is not harmful for the equipment if it stays dissolved. However, once the fuel becomes saturated with water, it can spontaneously form an emulsion of droplets. Large temperature oscillations can also cause the moisture to condense out [5]. For further details about the mechanics of emulsion formation [7]. The type of water content that is *not dissolved* in the biodiesel is called *free water*. This is the type that can cause issues in engines using the fuel. According to [8], 40 % of the problems with biodiesel stem from free water contamination and further 25 % are as a result of microbial growth, which is also largely attributed to free water content. The biological growth promoted by the presence of free water in the fuel can form sludge in the fuel tank. This slime, shown in Figure 1.2(a) is highly corrosive to metals and can be transported with the flow



(a) (b)

Figure 1.2: Damage caused by water-contaminated fuel. (a) microbial infestation of a filter (source: www.seamanship.ie); (b) corroded and worn out injector (source: www.injectorsdirect.com).

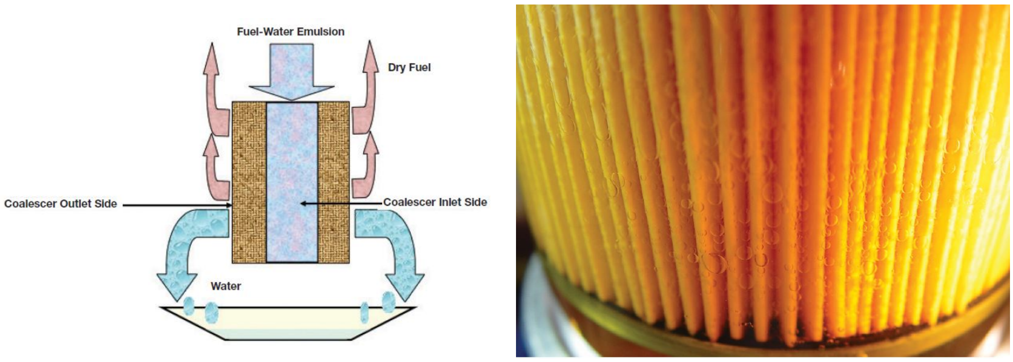
of fuel, plugging lines and filters. If no growth is observed and water remains in the fuel it can crystallise at low temperatures, forming solid particles that can damage the fuel system [9]. In the event where microscopic droplets of water make it to the injection system, they can have corrosive and abrasive effects on the injectors and other components. This problem is particularly pronounced with compression-ignition engines, because they operate at very high pressures, giving the droplets solid-like characteristics [10]. An example of the corrosive and abrasive action of water on an injector is shown in Figure 1.2(b). Finally, if the water makes it to the cylinders and takes part in the working cycle, it can affect the engine performance, decreasing the available power and efficiency, and causing excess emission of smoke from the engine.

1.4 Aspects of filtration

Since free water can form at any point right up to the biodiesel being delivered to the engine, one of the most reliable ways to ensure fuel cleanliness is to separate as much water as possible immediately prior to injection. This is typically done with the use of coalescing filters. Generally, these consist of a coalescer, which promotes the merging of micro droplets to increase their size, and a separator, which is responsible for the segregation of the enlarged water droplets from the fuel. Figure 1.3(a) shows a schematic representation of a coalescer, whereas Figure 1.3(b) depicts the enlarged water droplets leaving the media.

A variable that plays an important role in the stability of water emulsions in biodiesel is the *interfacial tension* (IFT) between the two components. IFT is the force that is exerted along the interface of two immiscible fluids. Lower IFT values result in finer, more stable water emulsions that are harder to coalesce or separate. The value of IFT between water and biodiesel is typically significantly lower than that between water and petrodiesel. Pure biodiesel, that is B100, is still considered problematic for everyday use mainly due to its solvent power. This is why biodiesel is most widely used as blend with petrodiesel. At present, diesel fuel in the UK contains up to 7 % biodiesel, with pressure from the government to raise this proportion to 20 % by 2020[2]. Due to environmental concerns the basis fuel used in biodiesel blends is the so called *ultra-low sulphur diesel* (ULSD) [8]. As the name suggests, ULSD differs from regular diesel by its sulphur content. It also has reduced lubricity and IFT for water. Due to the present blending ratios between ULSD and biodiesel the properties of the former dominate the overall behaviour of the blend. Filtering water out of such a mixture is already challenging, but there is one last ingredient that complicates matters even further: *additives*. Additives are a variety of chemicals that enhance different aspects of the fuel blend. Examples include corrosion protectors, lubricity enhancers, additives to increase the cetane number, to improve low temperature operation and many others. In addition to lowering IFT even further, some additives contain surfactants which form a shell around water droplets inhibiting coalescence and

[2] Private correspondence with Parker-Hannifin.



(a)

(b)

Figure 1.3: Coalescing filters. (a) a schematic representation of the coalescence process; (b) enlarged water droplets leaving a pleated coalescer (source: Stanfel [10]).

improving emulsion stability [11].

Porous fibrous filters are devices employing non-woven mesh components, consisting of nano-microscopic fibres, which act as coalescers and separators for water droplets. These filters can be manufactured via different methods, such as *electrospinning* or *melt-blowing* [12] and can handle industrial applications. Electrospun webs have small pores, large surface area-to-mass ratio making them preferable for filtration purposes. The construction of non-woven filters and the dependence of their performance on a number of factors is experimentally explored in the literature.

Even though the filters are complex, multi-parameter systems with time-dependent behaviour, a single figure of merit - the *quality factor* (QF) [13] - is widely used as a metric for assessing their performance. The quality factor is given by

$$QF = \frac{-\ln(1-\eta)}{\Delta p} \tag{1.1}$$

In Eq. 1.1, ΔP is the pressure drop across the filter and η is its separation efficiency defined as

$$\eta = 1 - \frac{C_o}{C_i} \tag{1.2}$$

where C_i and C_o are the input and output water concentrations, respectively. An increase in the value of QF indicates a filter with better performance. Both separation efficiency and pressure drop depend on many properties that can be broadly classified into three categories: *fibre properties*, *filter properties* and *flow properties*. For example the fibre *diameter*, *orientation* and *surface properties* [14] represent the first group. Filter *depth*, *porosity* and *pore size distribution* [15] are typical of the second category. Finally, *droplet size distribution*, *interfacial tension*, *flow velocity* and *pressure* [11, 14] are also identified as important in determining the filter properties. The presence of droplets of different sizes in the same flow, suggests that more complex filters are needed to ensure high percentage of separated water. For layered media, secondary factors such as drainage layers with particular properties [16], mixture of fibre materials and diameters [15, 17–20] and structures [21] become important. Furthermore, gradient porosities [22] and variable wettability [23] may be desirable, despite the increase complexity of the filter. The role of variable wettability in efficient filtration is widely debated [15, 17], but its importance is acknowledged.

Experimental work is useful for studying the macroscopic behaviour of the filter, but cannot provide a complete understanding about the processes that underlie this behaviour. Most practical filters are composed of fibres in the nanometre and micrometre ranges making their handling and observation difficult. This limitation is reflected in the fact that the practically testable combinations of filter properties may be a long way off of the optimal configuration. Additionally, due to the low number of such set-ups, they may not provide enough information about the performance of the filter in extreme and/or uncertain conditions. Moreover, laboratory experiments are not always representative of real working conditions [24]. On the other hand a computer, given an accurate model of the filter as some black-box function of its input properties can examine a large number of combinations of such inputs. Depending on the requirements of the analyst it can either return a figure of merit for each of them, enabling an optimisation procedure to be carried out, or allow the whole flow field to be examined in depth.

1.5 Computational modelling and uncertainty quantification

1.5.1 Computer models

The use of models, also called simulators or computer codes [25], in science in engineering has become ubiquitous. An increasing number of areas of research depend on computational modelling to some extent. Computer platforms with superior performance emerge all the time and new supercomputers are built to support massive computations. Nano-scale technologies, efficient algorithms and user-friendly interfaces make computer simulations available to a growing number of researchers. Substituting physical experiments for computer simulations has a number of advantages. On one side, performing typical computational experiments is much less expensive than setting up physical tests. Furthermore, simulations allow the investigations of systems and scenarios that would otherwise be physically impossible (*e.g.* molecular dynamics, large scale astrophysics) or unethical (*e.g.* disease and defects studies) to carry out.

Computer codes capture complex system behaviour through the collection of simpler sub-models. The end simulator represents a complex interaction among internal modules and for this reason its output for a specific input is not known prior to running the code. Thus, all the analyst sees is a set of inputs and corresponding outputs. Such simulators are called *black-box* models. That is the user does not typically have any explicit knowledge or the access to modify the inner workings of the code, such as the details of numerical algorithms or the underlying equations. More formally, the model provides some input/output mapping through a function $\eta(\cdot)$, that is

$$y = \eta(\mathbf{x}) \tag{1.3}$$

The input of the simulator resides in an *input space* that can be multidimensional, that is $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$, with d being the number of dimensions. For the purposes of this work the output will be considered one-dimensional. Computer models fall into two general categories determined by the behaviour of their outputs. *Stochastic* models produce output values which vary even when the same input is supplied to them. In contrast, *deterministic* simulators always output the same value whenever the same input is fed in. The multiphase flow which characterizes water-in-fuel emulsions is modelled using the lattice Boltzmann method (LBM). This family of methods contains deterministic models possessing a number of advantages, particularly desirable for multiphase flow simulation. More details on LBM will be given in Chapter 2.

Regardless of the nature of the code, simulators which model complex phenomena with high fidelity are usually computationally expensive. That is, a single input-output evaluation of the given code takes long enough to prohibit statistically meaningful ensembles of experiments to be run in a practical amount of time. This is particularly relevant in industrial settings, where important and costly decisions must be made relatively quickly. Even though great improvements have been made in that regard with the introduction of powerful computers, the trend is that more complex codes are developed when more powerful platforms become available. This way, there is a parity maintaining the meaning of the term *expensive computer code*.

1.5.2 Coalescence models

Coalescence processes are of great practical importance and have been studied for a long time. The dynamics behind droplet-droplet and droplet-fibre coalescence are very complex and thus all analytical models rely on simplifying assumptions. The advent of various numerical fluid methods, in combination with powerful computers, provided a means for analysts to study coalescence dynamics in more depth and relax some of the assumptions.

Different authors approach coalescence modelling from different perspectives. Here, a few prior works are reviewed along with the modelling approach they use, key findings and limitations. It should be noted that this review is not exhaustive and contains some coalescence studies which are similar, but not identical to the work in this dissertation.

Starting from macro scale simulations, Krasinski et *al.* [26] explore optimal geometries for pleated diesel filters. The authors simulate the flow in the housing of the filter and into the pleats using a commercial CFD package. No mention is made about designing an optimal filter media. Also from a macroscopic point of view, the authors of [27] investigate the effect of bed saturation of the performance of coalescing filters. The authors model the filter in its saturated state using volume-averaged theory [28]. The authors do not specify how the saturation in the filter was reached, but instead model the performance of the saturated bed. A key part of their approach is an exhaustive list of assumptions for the modelling framework. Some of these assumptions, such as an *incompressible filter bed* and *rigid, stationary fibres* are still applied in modern simulations today. Others, concerning flow and emulsion properties can be addressed due to the use of more sophisticated models, including LBM presented in this dissertation.

The series of works by Mino et *al.* address coalescence filtration on the pore level. For instance, the work in [29] uses the *coupled level set and volume of* *fluid* method[3] to model the coalescence of oil emulsion in water through a single, straight pore. The authors determine that coalescence is taking place, but since the simulation only encompasses a single pore, secondary dispersion occurs at the outlet. This phenomenon has also been observed experimentally [21], and is an undesirable by-product of high pressure drop across the filter. The work in [30] expands on the previous efforts to investigate the wettability, flux and fibre diameter effect on filter performance in multi-pore, two-dimensional beds. The model of choice for this work is the *phase field* method. The authors report that hydrophobic fibres facilitate the formation of bridging structures[4], which may lead to the formation of secondary dispersions. The series of articles is concluded by [31] in which a free-energy LBM is used to model the performance of a two-dimensional filter and to study the effect of wettability, porosity and fibre diameter on filter performance. The authors conclude that large pores promote the coalescence formation of larger droplets, but also allow smaller one to pass through. To provide a solution to this problem they analyse a bi-layer filter with smaller pores in the inlet layer and larger ones in the outlet layer. All three articles are concerned with modelling two-dimensional flows with highly ordered fibres in small simulation domains. Computational fluid dynamics solvers are inherently expensive to run, inhibiting simulations with sizes of practical interest to be performed. A way to develop efficient filter models is presented in Chapter 3 and Chapter 5.

The effect of orientation of fibres in the media is investigated numerically by [32]. The authors of the paper use the *finite volume method* to solve the flow equations. They conclude that particular orientation of the fibres improve the filter performance for small droplet emulsions, but deteriorate it for larger ones. However, the results are only compared with analytical methods instead of experiments. Similar technique is used in [33] for the design of a model of mist

[3] This and other methods mentioned in this section will be outlined in the next chapter.

[4] The article investigates oil-in-water emulsions and so hydrophobic is seen as oleo-philic.

filter. The authors couple the macroscopic solver with a *volume-of-fluid* tool to track droplet interfaces and use a separate mechanism for collision tracking. The performance of their code is also found to be in good agreement with analytical tools.

Finally, the work by [34] investigates coalescence dynamics with the use of LBM. The author validates LBM against the finite-volume and volume-of-fluid methods.

1.5.3 Uncertainty quantification

The motivation to introduce computer codes and use them instead of physical experiments was that simulations can be performed considerably more easily than physical experiments. Models should be able to process an arbitrarily large number of input configurations to produce a map of the output space. This map can then be used to quantify the uncertainty in the model output stemming from using variable inputs (*uncertainty analysis*) or to infer the importance of different inputs to the variation of the output (*sensitivity analysis*). However, for complex models constructing this map is not generally possible due to the long running times of the model. This situation necessitates a solution to increase the efficiency with which model evaluations are performed. A common way to do this is through the use of *surrogate models*. A surrogate model, also known as a *metamodel* is an approximation to the output of the simulator, which is much less computationally expensive to use. Without loss of generality, given n input configurations, for which the simulator has been evaluated, there will also be n output values. The metamodel uses this information to capture the behaviour of the original code. The selection of the n points, referred to as the *training sample* is particularly important so as to maximize the information about the simulator with a minimal computational effort. Once built, the surrogate model can be used to infer the output of the original code at input configurations not included in the training sample. There exist a large number of metamodels. Popular examples include Taylor series expansion [35], response surfaces [36], neural networks [37], radial basis functions [38] and support vector machines [39].

Another widely used method is Gaussian process emulation (GPE) [25]. An *emulator* is a statistical surrogate model which quantifies the uncertainty in any predictions it makes about the behaviour of the simulator. For this and other reasons that will be outlined in Chapter 3, GPE and its extensions are amongst the most popular metamodels in engineering. This is also why Gaussian process emulators will be used in this research.

1.5.4 Optimisation and reliability analysis

An important part of engineering computational experiments is *optimisation*. Finding the set of points which maximize (minimize) a performance function subject to a set of constraints is extensively studied in many fields of science. In the case of engineering, the output of the model is the performance function and the constraints are usually bounds in the input space. When there are two or more performance functions to be jointly optimised, the process is referred to as *multiobjective optimisation*. In particular, the research in this dissertation explores ways to maximize the separation efficiency of the filter bed, whilst keeping the pressure drop across it at a minimum. When adding nano-scale fibres both metrics increase [17] and therefore finding the optimum ratio between micro and nano fibres is of particular importance. Searching the input space in this manner using either experiments or an expensive model is infeasible, but could be substantially improved with the use of metamodels. The advantages GPE offers over other methods are used to perform efficient and reliable optimisation [40].

A procedure very similar to optimisation is *reliability analysis* in which the goal is to identify those portions of the input domain, $\mathcal{X}$ which cause the output of the model, y to exceed a predefined critical value, y_c. Formally

$$F = \{\mathbf{x} : \eta(\mathbf{x}) > y_c\} \tag{1.4}$$

where F is used to denote the *failure domain*, by analogy with real systems which *fail* if a given parameter exceeds a critical value. Typically F is several orders of magnitude smaller than $\mathcal{X}$ and its identification, through direct Monte Carlo (DMC), requires the code to be run many times. Even though some techniques [41] have improved upon DMC estimations, they may still be prohibitively expensive to use with complex models. Developing a metamodel that represents the simulator well, directly reduces this cost.

1.5.5 Further use of metamodels

As discussed above, metamodels can be used to make inferences about the output of the simulator at a cost much lower than that of the original code. This advantage can be used for more than purely computational studies aiming at querying the simulator for information. Any model is built on assumptions and simplifications and must therefore be tested for compliance with experimental data. Moreover, complex models usually have a large set of parameters and attempting to tune each of them by running the model would incur a considerable time penalty. The process of matching model predictions to observed data is called *calibration*. There are many calibration techniques, but a popular approach to use with GPE is *history matching* [42]. Formal calibration is not part of this research, but is listed as a future development. Some experimental work can be found in Section 5.5.

1.6 About this research

In light of the imminent transition to sustainable energy generation imposed worldwide, there is a need for an improved and reliable level of biofuel cleanliness to ensure safe and efficient operation of engines using it. Consequently, a systematic approach for the development of optimal coalescing filter beds is required.

The research contained in this dissertation comprises two areas of significance. From an academic standpoint, modelling the coalescence of water droplets in filter beds, leads to a better understanding about the physical process itself. Developing the means to practical coalescence modelling, will result in an improved quality of evaluation and prediction of filter performance without the need for costly experiments. At the same time, in an industrial setting separation of droplets larger than approximately 10 µm has already reached efficiencies close to 100 % [5]. However, with the increased use of biodiesel and ULSD in higher pressure engines, the droplet sizes fall closer to the 5 µm range. Thus, the introduction of new regulations for fuel cleanliness is likely, the effect of which will be directly reflected on the design and production of coalescing filters. As mentioned above, experimental research points to the conclusion that smaller droplets are more efficiently filtered with smaller fibres, the use of which is penalized with a higher pressure drop.

As one of the leading filtration equipment manufacturers, Parker Hannifin can benefit from having a means to systematically design and develop optimal filtration systems to reliably separate micro droplets. The filters developed by Parker Hannifin, shown in Figure 1.4, are used in fields such as land haulage, maritime transport, and others, all requiring reliable and sustainable solutions for fuel cleanliness.

[5] Private correspondence with Parker Hannifin.

(a) (b)

Figure 1.4: Filters from the Parker Hannifin array of products. (a) a heavy duty ULSD/biodieselfilter; (b) a marine diesel engine filter.

Aim

The aim of this research is to model the process of coalescence and separation of water emulsions from biodiesel using non-woven filters, to diagnose and quantify modelling uncertainty and to enable reliability analysis using the computer code.

Objectives

In order to achieve the aim stated above the following objectives are identified and addressed in this dissertation

1. The development of a multiphase fluid model to enable the simulation of coalescence processes. The selection of the modelling approach and environment, as well as an initial validation study are also included here.

2. The development of an accurate computational representation and generation of non-woven media. Accuracy of representation of experimentally observed media will enable the use of the model in validation and evaluation studies, whereas parametrization at the fibre level will allow the

generation and optimisation of non-woven beds.

3. The alleviation of scalability issues arising from the simulations of full filters through the use of metamodelling techniques. Reducing the cost of evaluating the model will permit statistical analyses about its predictions to be obtained and the uncertainty associated with using the model to be formally quantified.

4. The development of novel methods for efficient reliability analysis, which account for the computational cost of the model. Designing the filter bed with a minimal probability of failure or predicting it for an existing coalescer is traditionally computationally expensive and is addressed in this work.

Deliverables

As part of the industrial strand of the project a number of tools have been developed, which form the project's legacy. These are

1. A cross-platform code for coalescence simulation.

2. A parametric geometry tool for the computational generation of non-woven filter beds, utilising a graphical user interface to facilitate user interaction.

1.7 Thesis Layout

The layout of the thesis is as follows. Chapter 2 discusses methods for modelling multiphase flow and introduces the lattice Boltzmann method theoretically. Chapter 3 provides an introduction to metamodelling and in particular to Gaussian process emulation and relevant concepts.

Chapters 4 and 5 discuss practical aspects of the development and use of LBM for emulsion modelling in filtration. Results from LBM simulation and GP emulation are presented in Chapter 5. An overview of experimental work is also given there.

The ideas presented in Chapter 3 are applied to sensitivity analysis in Chapter 6. A novel reliability analysis method is presented in Chapter 7. Chapter 8 provides a short introduction to flow through porous media and focuses on the development of a three-dimensional filter bed generation tool. Finally, Chapter 9 draws some conclusions about the project and gives suggestions for future work.

CHAPTER 2

Lattice Boltzmann models for multiphase flow

This chapter discusses the process of modelling the flow of water-in-oil emulsions through the filter structure. This process can be divided into two major branches - modelling the multiphase flow and modelling the fibrous, non-woven mesh. The computational representation of the filter bed is a topic in its own right and will be addressed in Chapter 8.

Experimental and computational studies of filtration mechanisms and performance date back decades and have seen the development of a number of models and techniques with varying accuracy. This chapter introduces the lattice Boltzmann model (LBM) for multiphase flows. First, LBM is compared to other computational fluid dynamics (CFD) models. Second, the origins of LBM are discussed in some detail, demonstrating the evolution to its present state. Finally, several competing approaches within the LBM framework are juxtaposed, introducing the modelling choice for this research.

2.1 Modelling multiphase flow

The flow of any fluid is characterized by its velocity at any point in the volume of interest. This velocity field can be solved for, using the *Navier-Stokes* equations [43]

$$\frac{\partial \rho}{\partial t} + \rho(\nabla \cdot \mathbf{u}) = 0 \tag{2.1}$$

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = \nabla \cdot \mathbf{T} + \mathbf{F}_B \tag{2.2}$$

where ρ is the fluid density, $\mathbf{u}$ is its velocity, $\mathbf{T}$ is a stress tensor and $\mathbf{F}_B$ is a body force term. The expression in Eq. 2.1 is called the *continuity* equation and represents the conservation of mass in a fluid volume. The momentum equation in Eq. 2.2 is a statement of Newton's second law of motion applied to the infinitesimal fluid volume. In this form, Eq. 2.1 and Eq. 2.2 constitute an *underdetermined* system of equations, since there are three components of velocity, $\mathbf{u} = \left[u, v, w\right]^T$ and nine components to the stress tensor

$$\mathbf{T} \equiv \left\{ \begin{matrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{matrix} \right\}$$

and only four equations. To solve this problem, a particular form for the stress tensor must be assumed. Since the fluids of interest in this work have non-negligible viscosity, the stress tensor will be modelled as

$$\mathbf{T} = -p\mathbf{I} + \mu\left[\nabla \mathbf{u} + (\nabla \mathbf{u})^T\right] \tag{2.3}$$

where p is pressure, $\mathbf{I}$ is the identity tensor and μ is dynamic viscosity. Eq. 2.3 is known as a *constitutive relation*. Moreover, the analysis is carried out on liquids that are assumed to be incompressible. This addition, combined with Eq. 2.3

modifies Eq. 2.1 and Eq. 2.2 to

$$\nabla \cdot \mathbf{u} = 0 \tag{2.4}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \frac{-1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u} + \frac{\mathbf{F_B}}{\rho} \tag{2.5}$$

where the only undefined term is the kinematic viscosity, ν.

Eq. 2.4 and Eq. 2.5 are partial differential equations which can be solved numerically in a variety of ways. For instance a number of *difference methods* can be used. Three such widespread techniques are the *finite difference*, *finite element* and *finite volume* methods. For all of them, the simulation domain is discretised in elements of finite size and local. The finite difference method approximates the partial differential equations governing the system. In contrast, the finite element method provides a local approximation to the solution of the governing equation in each element. Finally, the finite volume works by conserving the flux of properties in and out of each cell. The distinction between them is subtle and a summary can be found in any graduate level CFD book, such as [44]. A related approach is the so-called *spectral element* method [45], which approximates the original equations in each element with a different set of basis functions, giving it more desirable convergence properties. For the multiphase formulation the Navier-Stokes equations have to be solved for each component. This can be done by any of the methods above coupled with a proper method for tracking the interface between components. This interface is of particular importance, as discussed in Section 1.4, since the performance of the filter depends strongly on interfacial and surface tension among components and surfaces. Several ways to track the interface between fluids exist. Among others, these are the *volume-of-fluid* (VOF) [46] and the *level set* (LS) [47] methods, as well as a combination of both (CLSVOF) [48]. VOF calculates the fraction of a grid cell occupied by each component and assigns the location of the interface to cells that are filled with both components. The LS method is a more general approach for tracking and analysing of curves and surfaces. It works by intersecting a *level-set function* with a plane and tracking the evolution of the resultant curve [49]. The last two solvers to be mentioned here are the *phase field* method [50] and *smoothed particle hydrodynamics* (SPH) [51]. The phase field method is based on the Ginzburg-Landau free energy functional and uses a compositional order parameter to distinguish between fluid components. SPH is a mesh free approach which uses integral kernel functions to propagate changes in the field variables along a collection of particles. Both methods deserve their separate sections, but are out of the scope of the current work and thus the interested reader is advised to refer to the above sources for more detail.

2.2 Lattice Boltzmann modelling of multiphase flow

Typical Navier-Stokes solvers are based on a *top-down* logic, where the macroscopic PDEs are discretized and solved on a finite mesh. The lattice Boltzmann model (LBM) takes on a different approach referred to as *bottom-up* whereby statistical information about ensembles of fluid particles is used to reach the Navier-Stokes equations. Because of this, phenomena that have inherently molecular nature, such as surface and interfacial tension, and pressure are directly obtained from the model without further additions. For the same reason, LBM is often referred to as a *mesoscopic* approach, since it occupies the scales between microscopic molecular dynamics models and macroscopic flow solvers. In the following sections LBM is discussed in more detail. A short comparison between LBM and conventional CFD methods is given in Table 2.3 at the end of this chapter.

2.2.1 LBM preliminaries

Physical aspects

Given a collection of particles[1] in a system, one can theoretically define certain properties for each particle. For instance at a specific moment in time all particles will have well-defined positions, $\mathbf{x}$ and momenta $\mathbf{p}$, that is they all have coordinates in *phase space*. If one could know and store this information and had knowledge of the forces $\mathbf{F}$ acting upon the system at any moment in time, one could then predict the state of the system at any point in the future. Realistically, such a scenario is unimaginable since any natural fluid be it even a rarefied gas has an extremely large number of particles in volumes of practical interest. It makes sense, when dealing with very large numbers, to use different approaches to obtaining information about the investigated fluid. The *kinetic theory of gases* is one such approach. According to it, the gas is viewed as a large collection of particles moving randomly in the system. Then, instead of trying to characterize individual particles, a statistical distribution, $f(\mathbf{x}, \mathbf{p}, t)$ is introduced, such that $f(\mathbf{x}, \mathbf{p}, t)d\mathbf{x}d\mathbf{p}$ captures the *likely* number of particles in an infinitesimal phase space volume - $d^3\mathbf{x}d^3\mathbf{p}$. The exponent in these expressions simply signifies that both position and momentum have three coordinates per particle. Another way to look at it is to say that $f(\mathbf{x}, \mathbf{p}, t)$ gives the probability of finding a particle at position $\mathbf{x} \pm \frac{1}{2}d\mathbf{x}$ and momentum $\mathbf{p} \pm \frac{1}{2}d\mathbf{p}$. This concept is similar to the notion of density and is in fact closely related to it. In the classical sense of the word, density is taken to mean mass density, $\rho(\mathbf{x}, t)$ as in Eq. 2.1, for example. Then, $f(\mathbf{x}, \mathbf{p}, t)$ can be seen as the density of particles at a given position, having certain momentum (or equivalently velocity). Kinetic theory goes on to assert that having these distribution functions is sufficient to work out the macroscopic properties of interest. As already defined, the distribution function depends on

[1] Here the term "particle" is meant to represent an object of a scale much smaller than that of the system.

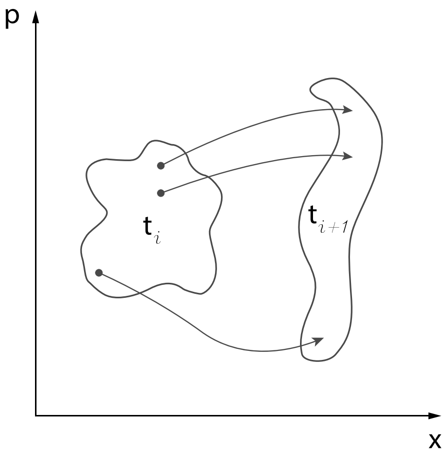


Figure 2.1: Intuition behind Liouville's theorem.

phase space variables and time. After a small amount of time has passed since some initial moment, the function has the form

$$f(\mathbf{x} + \frac{\mathbf{p}}{m}\Delta t, \mathbf{p} + \mathbf{F}\Delta t, t + \Delta t)$$

Due to Liouville's theorem (see Figure 2.1) which states that

Liouville's Theorem. *A volume enclosed by a surface in phase space remains constant as the surface moves through phase space.*

it can be seen that

$$f(\mathbf{x} + \frac{\mathbf{p}}{m}\Delta t, \mathbf{p} + \mathbf{F}\Delta t, t + \Delta t) = f(\mathbf{x}, \mathbf{p}, t)$$

since f is related to properties in a volume. This relation only holds true for cases where no collisions occur between particles. If they do collide then the change of f will be non-zero and can thus be attributed entirely to some functional of collisions

$$
\begin{aligned}
&f(\mathbf{x}+\frac{\mathbf{p}}{m}\Delta t,\mathbf{p}+\mathbf{F}\Delta t,t+\Delta t)-f(\mathbf{x},\mathbf{p},t)=0 && \text{without collisions}\\
&f(\mathbf{x}+\frac{\mathbf{p}}{m}\Delta t,\mathbf{p}+\mathbf{F}\Delta t,t+\Delta t)-f(\mathbf{x},\mathbf{p},t)=\left.\frac{\partial f}{\partial t}\right|_{collision} && \text{with collisions}
\end{aligned}
\tag{2.6}
$$

Taking limits, finding the total differential and rearranging Eq. 2.6 gives

$$
\frac{\partial f}{\partial t}+\nabla_x f\cdot\frac{\mathbf{p}}{m}+\nabla_p f\cdot\mathbf{F}=\Omega \tag{2.7}
$$

where Ω is an abbreviation for the two-body collision term and the subscript in ∇ identifies the variable with respect to which the gradients of f are taken. Eq. 2.7 is called the Boltzmann transport equation (BTE) and is due to Ludwig Boltzmann [52]. It describes the evolution of a system away from its equilibrium state. By virtue of statistical reasoning it can be seen that taking moments of f recovers some of the macroscopic variables in Eq. 2.4 and Eq. 2.5. Marginalizing $f(\mathbf{x},\mathbf{p},t)$ with respect to momentum, $\mathbf{p}$ is equivalent to accounting for the contributions of all particles regardless of their velocity and gives the density of the fluid

$$
\rho(\mathbf{x},t)=\int_{\mathbf{p}} f(\mathbf{x},\mathbf{p},t)d\mathbf{p} \tag{2.8}
$$

Similarly, taking the first moment of the distribution function with respect to $\mathbf{p}$ gives the momentum density of the fluid

$$
\rho(\mathbf{x},t)\mathbf{u}(\mathbf{x},t)=\int_{\mathbf{p}} \mathbf{p}f(\mathbf{x},\mathbf{p},t)d\mathbf{p} \tag{2.9}
$$

Higher moments correspond to energy density.

The form of the collision term is involved and has been studied since Boltzmann himself introduced the equation. The complexity arises from the fact that collisions have a statistical nature and cannot be captured precisely due to their dependence on higher order interactions [53]. Furthermore, in order to make conclusions about a statistical property, one must have knowledge of the way this property is distributed. In order to provide closure to the problem, Boltzmann applied the assumption of molecular chaos (*stosszahlansatz* - literally meaning shock number approach from German). This assumption states that the velocities of two particles entering a collision event are uncorrelated and independent of position. Statistically, this is equivalent to asserting that the joint distribution function of the particles is equal to the product of the marginal distribution functions. Despite these assumptions, the collision operator is still intractable and in order to be of practical use needs to be approximated. One of the most widespread and simple approximations is the one given by Bhatnagar, Gross and Krook [54], called the *BGK collision operator* after its authors

$$\Omega = \frac{1}{\tau}(f - f^{eq}) \tag{2.10}$$

The intuition behind Eq. 2.10 is that particles relax to an equilibrium distribution function due to momentum exchange if they are allowed to collide. This relaxation happens at a time scale τ called *relaxation time*. As will be seen in later sections, τ has a very important role in LBM, determining the viscosity of the modelled fluid and dictating the onset of instabilities in simulations. Special attention needs to be paid to the equilibrium distribution function, f^{eq}. Although it is not derived in the present work, this isotropic function is the well-known *Maxwell-Boltzmann* distribution

$$f^{eq} = \rho\left(\frac{1}{2\pi RT}\right)^{3/2} \exp\left(\frac{-|\boldsymbol{\xi} - \mathbf{u}|^2}{2RT}\right) \tag{2.11}$$

In Eq. 2.11 R is the gas constant which, together with the temperature T (with context-dependent units) represents the *specific internal energy density*, $RT/2$ of the fluid. This energy includes contributions from various molecular degrees of freedom in the fluid *e.g.* translation, rotation and vibration. Furthermore, $\boldsymbol{\xi}$ is the molecular velocity in phase space. The Maxwell-Boltzmann distribution and the factor $RT/2$ are instrumental to the field of statistical mechanics where they appear constantly (*e.g.* in the probability partition function).

The discussion about the physical aspects relevant to this work is closed by a brief overview of the methodology adopted to recover the Navier-Stokes equations from the Boltzmann equation. This procedure is referred to as *Chapman-Enskog* analysis and is covered in detail in [55, 56]. The mass conservation given in Eq. 2.1 is obtained by directly integrating Eq. 2.7 in velocity space

$$\frac{\partial}{\partial t}\int f d\boldsymbol{\xi} + \nabla_x \int f \cdot \boldsymbol{\xi} d\boldsymbol{\xi} + \mathbf{F} \cdot \nabla_p \int f d\boldsymbol{\xi} = \int \Omega d\boldsymbol{\xi}$$

where the equivalent notation $\boldsymbol{\xi} = \mathbf{p}/m$ is used. By Eq. 2.8 and Eq. 2.9 as well as the relation $\nabla_p \int f d\boldsymbol{\xi} = 0$ one recovers the mass conservation equation exactly. The momentum equation, Eq. 2.2 requires a similar procedure in addition to choosing an appropriate stress tensor. In order to include dissipative terms, such as viscosity the already mentioned Chapman-Enskog perturbation analysis must be conducted to first order [55].

Computational aspects

Lattice Boltzmann models have a long pedigree in computational simulation of fluids. Even though this work will not examine the different stages of the development of LBM and its predecessor lattice gas cellular automata (LGCA) in great detail, some important features will be mentioned.

The development starts with *cellular automata* (CA) first studied by John von Neumann and Stanislaw Ulam at Los Alamos in the 1940s [57]. A cellular automaton (CA) is a binary entity occupying a node in a lattice domain, whose state at a discrete future time, $t+1$, depends on the state of the automaton and those of its neighbours at the present time t, and a simple updating rule. Figure 2.2 shows a sample of updating rules for a CA with neighbourhood $n = 3$.

As the name suggests, LGCA is a type of a cellular automaton in that it uses a set of rules and neighbour interactions to reproduce complex phenomena. The idea behind LGCA is simple - the model occupies a prescribed regular lattice and obeys certain rules for its evolution. More formally, the process can be written as $\mathcal{E} = \mathcal{S} \circ \mathcal{C}$, where $\mathcal{E}$, $\mathcal{S}$ and $\mathcal{C}$ are the evolution, streaming and collision operators, respectively and $\circ$ denotes composition. LGCA assigns *occupation numbers*, n_i at each node in the lattice with

$$
\begin{aligned}
n_i(\mathbf{x}, t) &= 0 \qquad \text{absence of particle at node } \mathbf{x} \text{ at time, t} \\
n_i(\mathbf{x}, t) &= 1 \qquad \text{presence of particle at node } \mathbf{x} \text{ at time, t}
\end{aligned}
$$

The governing equation of LGCA is

$$\Delta n_i = C(\mathbf{n}) \tag{2.12}$$

where $C(\mathbf{n})$ is the collision operator $\mathcal{C}$. Each lattice site has a number of discrete velocities (a feature unique to the LGCA/LBM paradigm) associated with it, according to which particles propagate to different nodes as time advances. Also encoded in Eq. 2.7 is the ability of particles to collide. These collisions must respect the conservation of mass and momentum at the lattice site prior to and after the event. Strict rules and principles exist to enforce these requirements [53, 58]. This model can never simulate realistic molecular dynamics regardless of the lattice resolution. However, in accordance with statistical mechanics, different microscopic systems can give rise to the same macroscopic behaviour, which is exactly the case with LGCA.

Different LGCA models exist and have been used in the past, with some of the more famous being HPP [59] and FHP [60] named after their inventors, as well

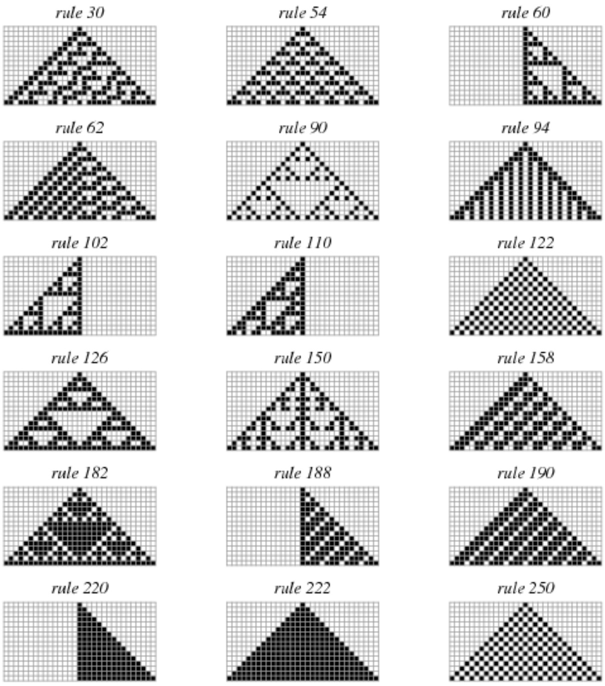
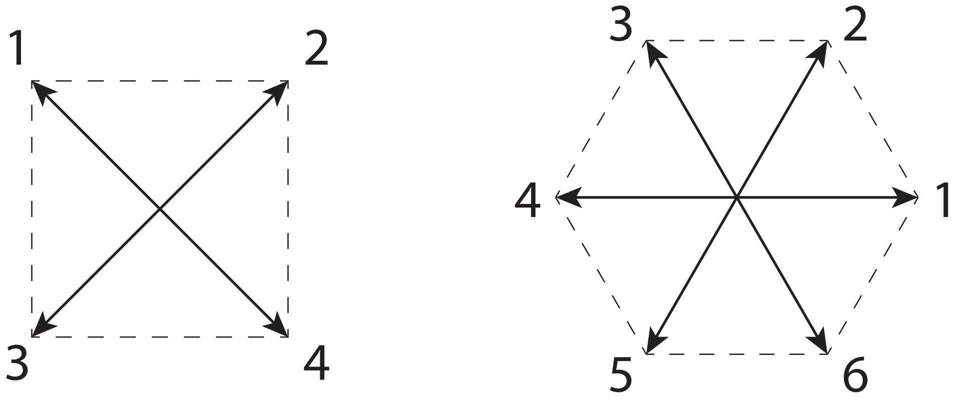


Figure 2.2: Updating rules for CA with neighbourhood, n=3. Rule 30 is sometimes used as a random number generator (from Wolfram [63]).

as PI [61], named after its underlying principle of operation [58]. Each setting is characterised by its own lattice, as shown in Figure 2.3, but all models assume particle mass of $m = 1$ and a unit of length defined as one lattice unit - $d \equiv 1$. The FHP model was the first to enable the near full recovery of the Navier-Stokes equations and macroscopic variables. However, it was not capable of simulating 3D fluids, due to insufficient isotropy of the lattice tensors. This limitation was partially resolved [2] with the introduction of the four-dimensional face-centred hypercube (FCHC) lattice [62]. The LGCA models had a number of problems, which sometimes led to inaccurate and unstable simulations. The more serious issues were

- Lack of Gallilean relativity - due to velocity discretization of the lattice.

[2] The cost of simulating 3D fluids in LGCA was so great, that it came to be one of the reasons for developing LBM.



(a) HPP

(b) FHP

Figure 2.3: Lattices in for different LGCA models.

- Velocity dependence on pressure - due to velocity discretization on the lattice.
- Statistical noise - due to the Boolean nature of the method.
- Low Reynolds number constraints - due to heavily reduced *collision space* *i.e.* small number of allowable collisions.
- Exponential complexity - due to form of the collision look up table and the number of velocities in 3D lattices.
- Spurious invariants - quantities other than mass, momentum and energy conserved due to discrete velocity space.

In an attempt to rectify these problems, the Lattice Boltzmann Equation (LBE) models were developed. This modelling approach has also seen several important advancements. The original method was the *non-linear LBE* (NN-LBE) developed in [64] to solve the problems associated with statistical noise. It introduced the notion of a distribution function

$$f_i = \langle n_i \rangle$$

where $\langle\cdot\rangle$ denotes an ensemble average. A way to interpret this change is that the new operator uses the time history at the lattice to perform calculations, instead of individual occupation numbers. The evolution equation of the model is a straightforward translation of Eq. 2.12 with the new concept f_i.

$$\Delta f_i = C(f_i) \tag{2.13}$$

The NN-LBE *is* non-linear in the form of the multi-body collision operator $C(f_i)$. This implementation resolved the statistical noise issues and made the method practically usable on general purpose computers. Even though the rest of the issues from LGCA were carried over to NN-LBE, it provided a solid foundation for research to gradually resolve each of them. The most significant practical issue was the exponential cost in performing 3D simulations. This problem was addressed by the successor of NN-LBE in the hierarchy of LBM - the *quasilinear LBE* (QL-LBE) [65]. This approach is based on a "linearised" collision operator and has the form

$$\Delta f_i = A_{ij}(f_i - f_i^{eq}) \tag{2.14}$$

In Eq. 2.14, A_{ij} is the Hessian of the original collision operator that emerges from its Taylor expansion about a global equilibrium distribution function. The non-linear portion of the operator is encoded in the term f^{eq} [53]. The simplification this method brings is significant, since the Hessian is a square, $b \times b$ matrix with b being the number of discrete velocities in the lattice. The original transition matrix[3] of LGCA and the collision operator of NN-LBE have complexity $\mathcal{O}(2^b)$, compared to $\mathcal{O}(b^2)$ for QL-LBE. Following from the quasilinear LBE, two more improvements were introduced. The first was the *enhanced collision* LBE (EC-LBE) [66], which solves the problems associated with low Reynolds numbers and violation of Galilean invariance. This is achieved by redefining the scattering

[3] The transition matrix is a lookup table that defines the outcome of collisions.

matrix A_{ij} in a way that no longer associates with the transition matrix of LGCA. The motivation behind this decision is the fact that there is no need for using LGCA's boolean collision nature when the rest of the evolution equation is no longer boolean.

The last descendant in the pedigree is the lattice Boltzmann model that is in widest use today and has been applied to a wide variety of problems. The authors of the EC-LBE model showed that the viscosity in simulations depends only on a single parameter - the leading eigenvalue of the scattering matrix. This is tantamount to having a diagonal matrix for A_{ij}, and furthermore with identical non-zero elements. This then leads to replacing A_{ij} altogether with a *single relaxation time*, ω for an evolution equation of the form

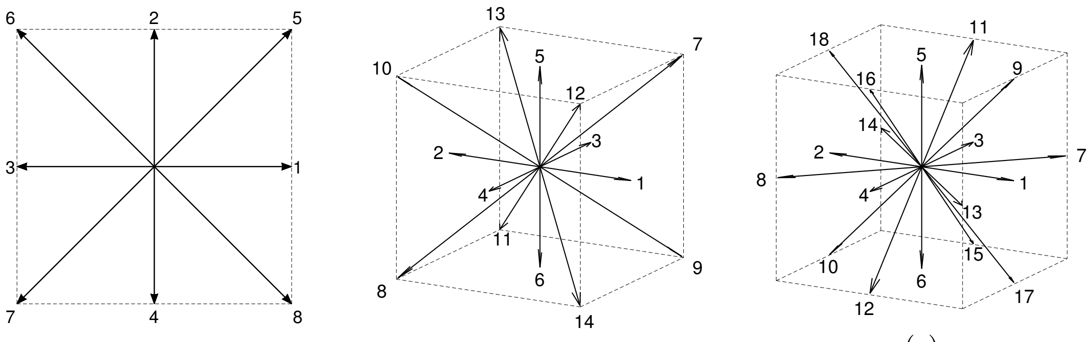
$$\Delta f_i = -\frac{1}{\tau}(f_i - f_i^{eq}) \tag{2.15}$$

The collision operator in Eq. 2.15 is closely reminiscent of that in Eq. 2.10 and because of this the model governed by Eq. 2.15 is called the *lattice Bhatnagar-Gross-Krook* (LBGK) model. It was proposed by a number of authors [67–69]. As LBGK is the principal workhorse of the LBM family it will be discussed in a little more detail. Since now there is only one relaxation time scale all hydrodynamic modes decay at the same rate. This is a problem which was resolved by requiring the local equilibrium function, f_i^{eq} to satisfy the original distribution function moments

$$\sum_i f_i^{eq} = \sum_i f_i \quad = \rho(\mathbf{x}, t) \tag{2.16}$$

$$\sum_i \mathbf{c}_i f_i^{eq} = \sum_i \mathbf{c}_i f_i = \rho(\mathbf{x}, t)\mathbf{u}(\mathbf{x}, t) \tag{2.17}$$

The relations in Eq. 2.16 and Eq. 2.17 are the discrete counterparts of Eq. 2.8 and Eq. 2.9. There, $\mathbf{c}_i$ is the discrete lattice velocity in the i^{th} direction. It is



(a) (b) (c)

Figure 2.4: Common LBM lattices. (a) D2Q9 (b) D3Q15; (c) D3Q19.

not a feature unique to LBGK, but was skipped over in previous models as they were not examined in depth. A unique characteristic of the velocities in LBGK, however, is the fact that they are not restricted to be the same in all directions as is the case in earlier models. This precise difference leads to the so called *multi-energy model*, allowing Eq. 2.16 and Eq. 2.17 to be satisfied. The corresponding equilibrium distribution function is given by

$$f_i^{eq}(x,t) = w_i \rho \left[1 + \frac{\mathbf{c}_i \cdot \mathbf{u}}{c_s^2} + \frac{(\mathbf{c}_i \cdot \mathbf{u})^2}{2c_s^4} - \frac{|\mathbf{u}|^2}{2c_s^2} \right] \tag{2.18}$$

where w_i and c_s are determined by the lattice and correspond to weights and speed of sound, respectively. Both of these parameters depend on the structure of the lattice and are given in Table 2.1 for the most commonly used two- and three-dimensional models. The graphical representation of each lattice is shown in Figure 2.4. In order to conform to the accepted convention and to indicate the importance of the LBGK model it shall be referred to as LBM for the remainder of this work. Furthermore, its evolution equation will be considered explicitly

$$f_i(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t) = f_i(\mathbf{x}, t) - \frac{1}{\tau} \left[f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t) \right] \tag{2.19}$$

where τ is the relaxation time. In addition to the macroscopic conserved quanti-

Table 2.1: Parameter values for common LBM lattices

Lattice	c_s	$\mathbf{c}_i$	w_i	i
D2Q9	$1/\sqrt{3}$	$[0, 0]$	$4/9$	$i = 0$
		$[\pm1, 0]; [0, \pm1]$	$1/9$	$i = 1, 3; 2, 4$
		$[\pm1, \pm1]$	$1/36$	$i = 5 \ldots 8$
D3Q15	$1/\sqrt{3}$	$[0, 0, 0]$	$2/9$	$i = 0$
		$[\pm1, 0, 0]; [0, \pm1, 0]; [0, 0, \pm1]$	$1/9$	$i = 1, 2; 3, 4; 5, 6$
		$[\pm1, \pm1, \pm1]$	$1/72$	$i = 7 \ldots 14$
D3Q19	$1/\sqrt{3}$	$[0, 0, 0]$	$12/36$	$i = 0$
		$[\pm1, 0, 0]; [0, \pm1, 0]; [0, 0, \pm1]$	$2/36$	$i = 1, 2; 3, 4; 5, 6$
		$[\pm1, \pm1, 0]; [\pm1, 0, \pm1]; [0, \pm1, \pm1]$	$1/36$	$i = 7 \ldots 10;$ $11 \ldots 14; 15 \ldots 18$

ties in Eq. 2.16 and Eq. 2.17, the equation of state relating pressure, P to density is given by

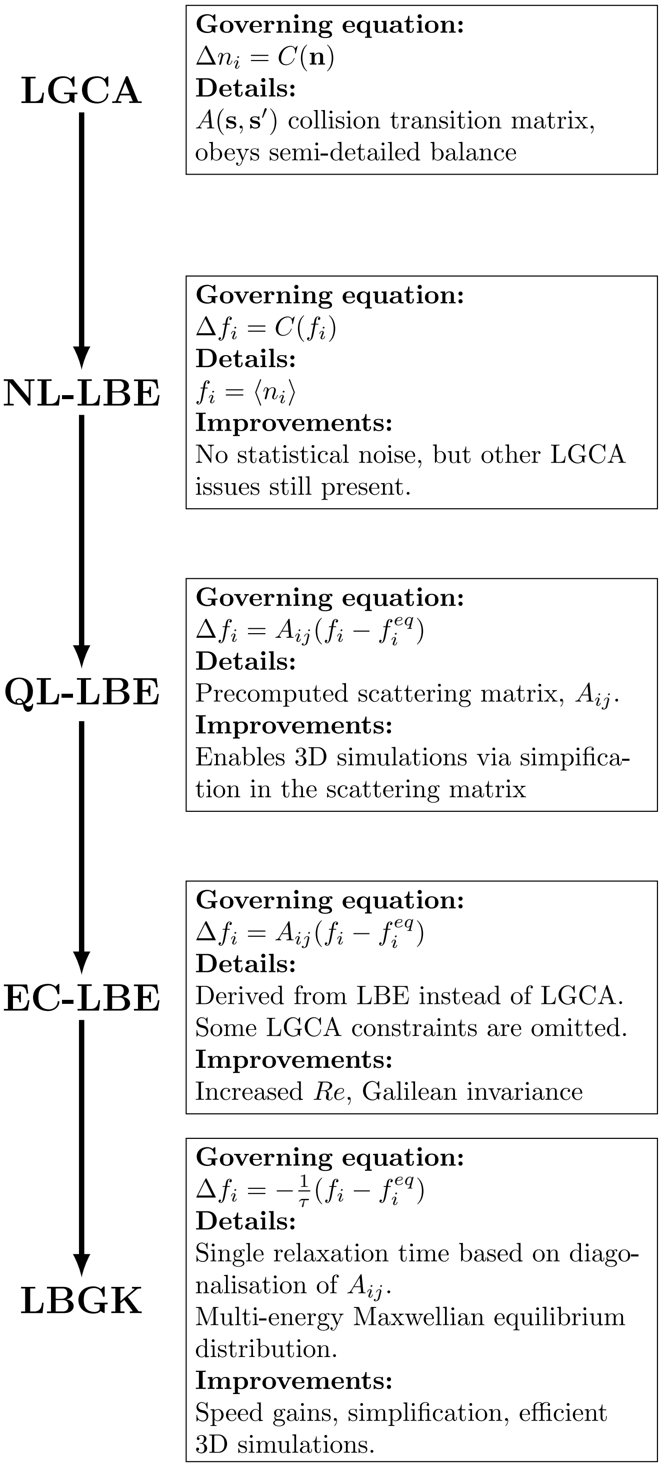
$$p = c_s^2 \rho \tag{2.20}$$

and the fluid viscosity, ν is given in terms of the relaxation time, τ and the lattice speed of sound, c_s

$$\nu = c_s^2 \big(2\tau - 1 \big) \tag{2.21}$$

Both properties emerge from the Chapman-Enskog procedure as outlined in [55].

This brief look into the history of LBM is based on [53] and [58]. Figure 2.5 presents an outline of the chronology discussed here. More details about conditions that different parameters must satisfy, as well as various analyses, including the Chapman-Enskog procedure for recovering the Navier-Stokes equations from LBM can be found in [55, 56].



LGCA

Governing equation:
$\Delta n_i = C(\mathbf{n})$
Details:
$A(\mathbf{s}, \mathbf{s}')$ collision transition matrix, obeys semi-detailed balance

NL-LBE

Governing equation:
$\Delta f_i = C(f_i)$
Details:
$f_i = \langle n_i \rangle$
Improvements:
No statistical noise, but other LGCA issues still present.

QL-LBE

Governing equation:
$\Delta f_i = A_{ij}(f_i - f_i^{eq})$
Details:
Precomputed scattering matrix, A_{ij}.
Improvements:
Enables 3D simulations via simpification in the scattering matrix

EC-LBE

Governing equation:
$\Delta f_i = A_{ij}(f_i - f_i^{eq})$
Details:
Derived from LBE instead of LGCA. Some LGCA constraints are omitted.
Improvements:
Increased Re, Galilean invariance

LBGK

Governing equation:
$\Delta f_i = -\frac{1}{\tau}(f_i - f_i^{eq})$
Details:
Single relaxation time based on diagonalisation of A_{ij}.
Multi-energy Maxwellian equilibrium distribution.
Improvements:
Speed gains, simplification, efficient 3D simulations.

Figure 2.5: LBM development.

2.2.2 LBM for multiphase flow

In their original formulation, lattice Boltzmann models simulate the flow of single component fluids. Additional treatment is needed to enable the simulation of multiphase flow. There exist more methods giving multiphase modelling capabilities to LBM, than there are members of the original family of LGCA and derivatives as outlined in Section 2.2.1. A rich taxonomy of these approaches can be found in §1 of [70] as well as in [71] and [72]. More on the topic can be found in [53].

Among the myriad of methods available to LBM users, three approaches stand out. That is, they are most commonly the methods of choice when simulating multiphase flows in LBM. These are the *chromodynamic models* [73, 74], *pseudo-potential model* [75, 76] and the *free energy method* [77], given in chronological order.

The chromodynamic model is based on defining separate distribution functions, f_{ic} for each colour c. In the model, "colour" is simply used to refer to separate components of the fluid. Each component is governed by a separate LBM relation in the form of Eq. 2.15

$$\Delta f_{ic} = -\frac{1}{\tau_c}(f_{ic} - f_{ic}^{eq}) + S_{ic} \tag{2.22}$$

where S_{ic} is a heuristic source term capturing component interactions, originally given in the form

$$S_{ic} = A\frac{f(G)}{h(G)}$$

where $f(G)$ and $h(G)$ are particular functions of the colour gradient between fluids and A is a free parameter corresponding to interfacial tension, which is one of the strengths of the model. The influence of interfacial tension on filter performance is mentioned in Section 1.4. This approach suffers from several practical issues, including dependence of interfacial tension of the orientation of the interface and computationally demanding colour redistribution step used to promote phase separation.

As already mentioned, the chromodynamic approach is somewhat heuristic in its definition of interaction forces between components. In the introduction to Section 2.2 it was pointed out that interfacial and surface tension phenomena have a molecular nature and for this reason LBM is particularly well suited to simulating them. The pseudo-potential model is aimed at utilizing the physical root of fluid interactions to model them. Akin to the evolution of LBM, this method builds on the chromodynamic model by re-expressing the source term as

$$S_{ic} = \mathbf{F}_c \cdot \mathbf{c}_i$$

where $\mathbf{F}_c$ is the pairwise interaction force between components. This research utilizes the pseudo-potential modelling strategy and thus further definitions and details are discussed in Section 2.3. Anticipating the discussion there, some of the advantages of the model are represented by its relative simplicity and ease of implementation and by the fact that it encapsulates spontaneous phase separation. Two of the main limitations include the reduced set of functional forms of S_{ic} for thermodynamic consistency and lack of local momentum conservation.

Finally, the free energy approach was inspired by the consistency issues experienced by the pseudo-potential method. The model is based on a prescribed free energy density functional defined as

$$\Psi(\rho) = \int \frac{1}{2} k \big[(\nabla \rho)^2 + \psi(\rho) \big] dV$$

where the sum splits into the energy required to support density gradients and the bulk free energy. The method has been shown to be in excellent agreement with thermodynamic theory and despite a few disadvantages is also a widely used method in the simulation of multiphase flows [31]. A summary of some of the

Table 2.2: Comparison of common multiphase LBM methods.

Method	**Advantages**	**Disadvantages**
Chromodynamic	Enables multicomponent simulations.	Empiric nature.
	Free parameter interfacial tension (IFT).	Spurious dependence of IFT on interface orientation.
		Computationally demanding calculations.
		Spurious interface velocities.
Pseudo-potential	Physical basis.	Thermodynamic consistency issues.
	Spontaneous separation.	Lack of local momentum conservation.
	Versatile.	Spurious interface velocities.
	Easy to implement.	
	Efficient.	
Free energy	Improved thermodynamic consistency (TC).	Lack of full TC.
	Physical basis	Spurious interface velocities.

advantages and disadvantages of the three methods discussed in this section is given in Table 2.2.

2.3 The Shan-Chen model

Inspecting Eq. 2.19, it can be seen that the original LBM model only accounts for nearest neighbour interactions at each node of the lattice. In order to simulate emulsions, the Shan-Chen model prescribes that long range interactions between different fluid components are taken into account. The pseudo-potential method can be split into two models emerging from the same mathematical definition.

The *single component multiphase* (SCMP) approach models two phases of the same fluid, whereas the *multi-component multiphase* (MCMP) treatment simulates two immiscible fluids and potentially their different phases. It should be noted that MCMP is the original method with SCMP being a simplification and thus MCMP is considered first. Each modelling choice is discussed below and the main advantages and disadvantages are outlined. The present research utilizes both methods for different purposes as will be discussed in Chapter 5.

2.3.1 Multi-component multiphase models

The multi-component multiphase model models immiscible fluids as separate computational fields, having distinct distribution functions f_{ic}. Consequently, anticipating the more detailed treatment below, the two (or more) components have different equilibria, densities and viscosities. Their velocities can also be taken separately, but it stands to reason that having a single bulk velocity is more meaningful in predicting the behaviour of the flow. The way the two components of the emulsion interact was alluded to in the overview of Section 2.2.2. A force acting between the components defines either an attractive or a repulsive interaction as will be seen in due course.

Moving to formal treatment of the model, Eq. 2.19 acquires one more degree of freedom

$$f_{ic}(\mathbf{x} + \mathbf{c}_i \Delta t, t + \Delta t) = f_{ic}(\mathbf{x}, t) - \frac{1}{\tau_c}[f_{ic}(\mathbf{x}, t) - f_{ic}^{eq}(\mathbf{x}, t)] \tag{2.23}$$

where the subscript $c = \{1 \ldots n\}$ with n being the number of the simulated components. Eq. 2.23 is solved independently for each component, which makes it easy to implement. The density and velocity of each fluid are obtained in the usual way, solving Eq. 2.16 and Eq. 2.17 for the components. Having each velocity separately is not very meaningful since it is the bulk flow that needs to be analysed. The so-called composite velocity is given by

$$\mathbf{u}' = \frac{\sum_c \frac{1}{\tau_c} \sum_i f_{ic}\mathbf{c}_i}{\sum_c \frac{1}{\tau_c}\rho_c} \tag{2.24}$$

In the literature $\mathbf{u}'$ is called *composite macroscopic velocity.* The physical meaning behind the expression for the composite velocity is a weighted sum of momenta of the two components.

So far, the MCMP model features almost completely separate fluids. In order to promote interaction between them the *interaction potential* term is introduced as follows

$$\mathbf{F}_{c\bar{c}}(\mathbf{x},t) = -G_{c\bar{c}}\psi_c(\mathbf{x},t)\sum_i w_i\psi_{\bar{c}}(\mathbf{x}+\mathbf{c}_i,t)\mathbf{c}_i \tag{2.25}$$

where $c \neq \bar{c}$ denote any two components of the emulsion, w_i are the weights defined in Table 2.1 [4] and ψ_c is the interaction potential function. Additionally, $G_{c\bar{c}} \equiv G_{c\bar{c}}(\mathbf{x},\bar{\mathbf{x}})$ is an interaction strength parameter between two distinct nearest-neighbour sites, $\mathbf{x}$ and $\bar{\mathbf{x}}$. The surface tension between the components is controlled by $G_{c\bar{c}}$, which also determines the nature of the interaction as attractive for positive values and repulsive for negative ones. More generally, G is expressed in a matrix form as

$$\mathbf{G} = \begin{bmatrix} G_{c\bar{c}} & G_{cc} \\ G_{\bar{c}\bar{c}} & G_{c\bar{c}} \end{bmatrix}$$

where the terms on the diagonal are equal as they represent pairwise interactions and the off-diagonal terms control the phase separation within each component. Special attention should be given to the interaction potential. This function has the role of effective mass or free energy in the model. It has several forms used in the literature [78]. However, the only form of ψ that ensures thermodynamic

[4] The interaction summation is taken over non-zero indices i only.

consistency (with Maxwell equal area rule) is the following [79]

$$\psi_c \propto \exp(-1/\rho) \tag{2.26}$$

In the scenario where only single phase per component is considered, there is a little more freedom in choosing ψ_c. In the present research it is taken as $\psi_c = \rho_c$, but other forms are possible. As will be seen in Section 2.3.2, this choice leads to instabilities in the SCMP model where different phases have large density ratios. It should be noted here, that the *density* of each component in MCMP models, $\rho_c(\mathbf{x}, t)$ denotes the concentration of that component at site $\mathbf{x}$ and time t. As such, density values vary from 0 to 1 and the sum of the densities of both components is equal to 1.

There are a number of strategies for incorporating the force term in Eq. 2.25 and other forces into the model [80, 81]. Here the original *velocity shift* method is used

$$\mathbf{u}_c = \mathbf{u}' + \tau_c \frac{\mathbf{F}_{c\bar{c}}}{\rho_c} \tag{2.27}$$

Increasing $G_{c\bar{c}}$ decreases the width of the diffusive interface between components, reducing the solubility between them. A limit to this increase can be seen in Eq. 2.27, where a sufficiently high $\mathbf{F}_{c\bar{c}}$ can lead to the velocity violating the small Mach number assumptions encoded in the model, rendering it unstable.

Lastly, having the interaction force in the model also changes its equation of state, Eq. 2.20, to

$$p = c_s^2 \sum_c \rho_c + \frac{c_s^2}{2} \sum_c G_{c\bar{c}} \psi_c \psi_{\bar{c}} \tag{2.28}$$

As with other ingredients of LBM Eq. 2.28 is not the only non-ideal equation of state. Other possibilities include the Carnahan-Starling, Peng-Robinson and more [82].

2.3.2 Single component multiphase models

As mentioned in Section 2.2.2, the SCMP model simulates different phases of the same substance, *i.e.* it models truly multiphase fluids. Such flows are common in nature and can be used to study phenomena such as cavitation and free surface flows. In the study of multi-component flows, such as the one of water-in-fuel emulsions, SCMP can be used as a surrogate[5] for the more expensive MCMP model. Carefully choosing different parameter values of the model can lead to stable and informative simulations. Using SCMP in this way brings a general limitation - it can only simulate binary liquids, that is an emulsion with two components. This makes intuitive sense as there are only two well-determined fluid phases of substances - liquid and gas. Despite it being a general disadvantage, the present research is only concerned with two components, making SCMP a good "prototyping" tool.

The authors of the MCMP model introduced SCMP [76] for studying phase transitions in fluids. The idea behind it is essentially the same as in MCMP with a few distinctions. Firstly, the model is built with a single distribution function, similar to the original LBM. This has an immediate impact on the computational cost of simulations as there are half as many calculations to be made for each conserved property as in MCMP. A disadvantage of this is that the diffusion between distinct fluids takes place on a much shorter time scale than in MCMP, simply because the two phases are not hard coded as separate. Moreover, the interaction strength is based on the parameter G_{cc} controlling intra-component potentials. In Eq. 2.25, ψ_c, now denoted as ψ due to the lack of different components has the form

$$\psi(\rho) = \rho_0 \left[1 - exp \left(\frac{-\rho}{\rho_0} \right) \right] \tag{2.29}$$

[5] This is not to be confused with surrogate models in uncertainty quantification. They are discussed in Chapter 3.

where ρ_0 is an initial density whose value depends on other parameters and is taken here as $\rho_0 = 1$. The thermodynamic consistency arguments raised in the previous section are also valid here, but the choice of ψ is dictated mostly by practical matters. The third distinction between the two pseudo-potential models is the meaning of the parameter G_{cc}. In SCMP it has a temperature-like nature, which determines the phase separation of the fluid. It has negative values for attraction and positive for repulsion. For the remainder of this work G_{cc} will be shortened to G for clarity and will indicate the use of SCMP in the context. A good feature of SCMP is the existence of analytic solutions for the critical values of G leading to phase separation and the density, ρ, at this point. The critical values can be found as follows. Including interactions, as in Eq. 2.28 the non-ideal equation of state reads

$$p = c_s^2 \rho + c_s^2 \frac{G}{2} \psi(\rho)^2 \tag{2.30}$$

It is known from thermodynamic theory, that the critical point is an inflection

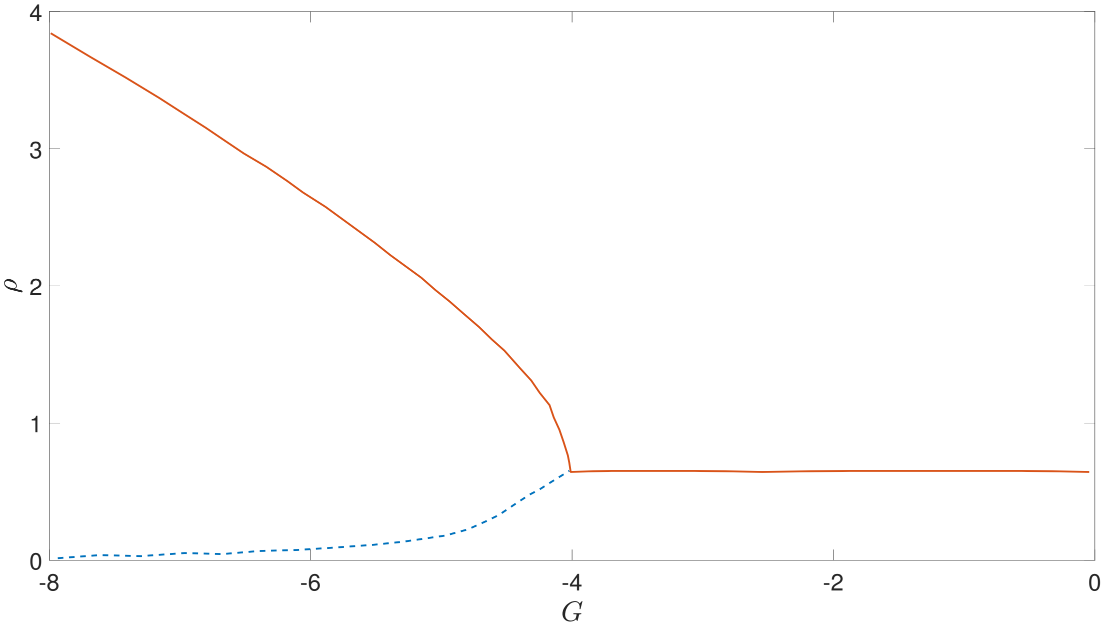


Figure 2.6: Coexistence curve for SCMP. The fluid undergoes spontaneous phase separation at $G = G^* = -4$ for effective mass of the form in Eq. 2.29.

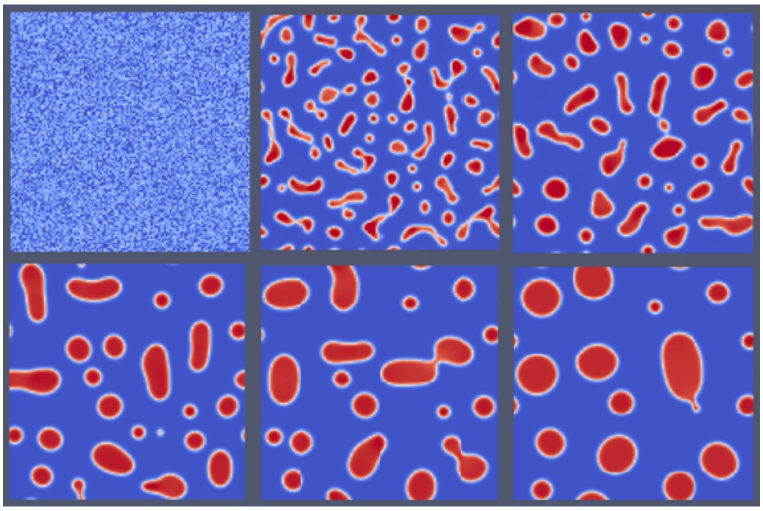


Figure 2.7: Spontaneous separation of a fluid in liquid and vapour phase. $G = -6$. Time increases from left to right and from top to bottom.

point *i.e.* the first two derivatives of the isotherm vanish [83]. Differentiating Eq. 2.30 twice with respect to ρ and setting both derivatives to 0, the critical values of G and ρ needed for phase separation are obtained as $G^* = -4$ and $\rho_c = \ln(2)$. Thus initializing SCMP simulations with $G < G^*$ guarantees the phase separation of a liquid in light and heavy phases. The phase curve of the SCMP model is shown in Figure 2.6. The separation process is illustrated in Figure 2.7. It is interesting to note that neither value depends on any of the lattice-associated variables, making them independent of the grid structure and dependent instead on the form of the interaction potential in Eq. 2.29.

The SCMP model has some problems as discussed in the review by Chen et *al.* [79]. The more practical of these issues is the presence of *spurious velocity* streams in the vicinity of the phase interface, illustrated in Figure 2.8 and the erroneous dependence between some properties. The former problem stems from the discretization of the lattice and can lead to the velocity exceeding its stable values, thus leading to instabilities. The strength of the spurious currents increases as the density ratio becomes larger and as the relaxation time decreases. It has been shown in practice that using different equations of state can reduce

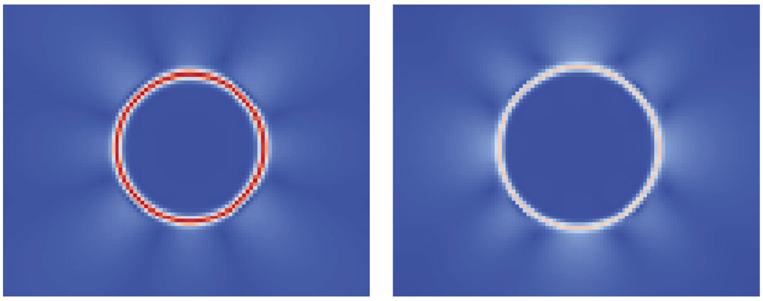
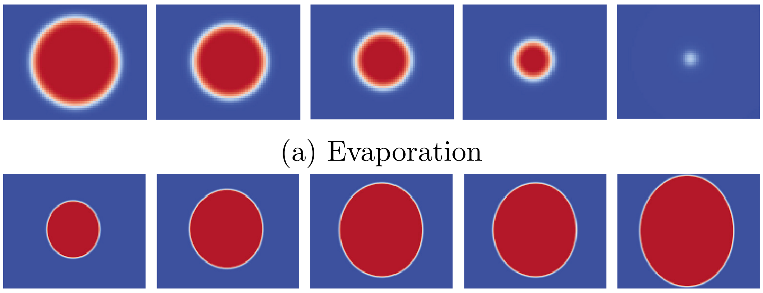


Figure 2.8: Velocity contours of a stationary droplets. The spurious currents of the interface can be see on both the horizontal velocity (left) and vertical velocity (right) contours.

the magnitude of the streams [84]. A by-product of the presence of spurious currents is the accelerated *evaporation* and *condensation* of the liquid phase through the diffuse interface. These phenomena, depend strongly on G (more pronounced for low values) and limit the minimum droplet sizes that can be sustained in the simulation, further distancing computation from reality. Examples of evaporation and condensation are shown in Figure 2.9. The second drawback, namely the dependence among physical properties refers to the inability of the different parameters in SCMP to be changed independently, which may limit its usefulness as a simulation tool. Particularly important and unwanted dependences are those of interfacial tension on density and relaxation time. This deficiency is problematic in the current research since, as pointed out a number of times before, the role of interfacial tension is of particular interest. In order to alleviate this problem one can change the forcing scheme as discussed by [81] and introduce multi-range interaction potential [85, 86], which allows the next-to-nearest neighbours to contribute to the interaction potential in both a repulsive and attractive way.



(a) Evaporation

(b) Condensation

Figure 2.9: Density contours of stationary droplets. Evaporation occurs below certain drop diameter, whereas condensation takes place above a critical size. Both mechanisms depend on the interaction parameter G. Time increases from left to right.

2.3.3 Boundary conditions

The ability to simulate free stream flows is important, but not very useful in practice. All meaningful experiments, physical or computational, are concerned with the flow *over bodies*. Here, *over* is taken to collectively mean around, through and so forth. In order to enable such simulations, there needs to be a way to indicate the presence of the object in the fluid field and to have adequate models to dictate how the fluid behaves in the vicinity of the object. This is done through the use of *solid boundary conditions*. To add to this, some investigations require large spatial domains to develop an accurate model[6]. At the same time computer simulations are limited by the amount of available processing power and using very large domains may render meaningful simulations prohibitively slow to run and requiring large amount of storage. Another type of boundary conditions is used to address these issues, referred to as *open boundary conditions*. A comprehensive overview of the topic is given in §5 of [55].

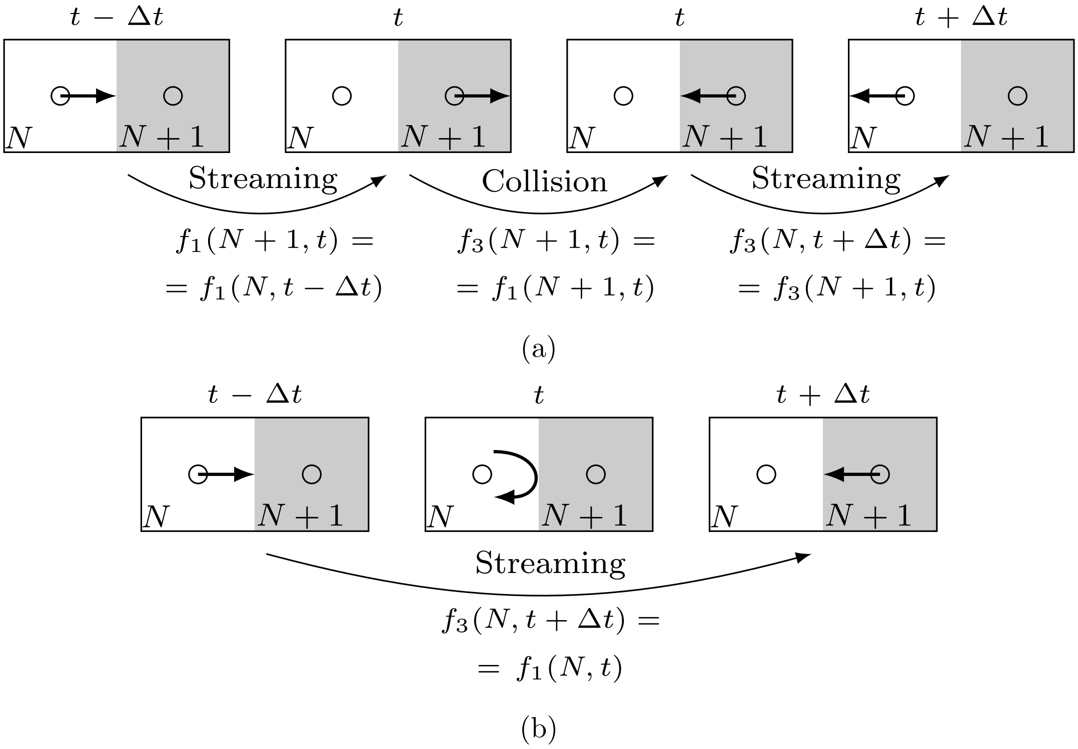
[6] See Section 5.3 for such experiments.

Boundary conditions for solids

There are many ways to model the presence of solid bodies in the fluid domain. In fact, the ease with which complex solid structures can be incorporated in a simulation is one of the reasons for the widespread use of LBM. All that needs to be done to include the geometry is to designate the required nodes as *solid*. In this way arbitrarily convoluted structures can be modelled directly from image pixels.

In this research the focus is on *no-slip* boundary conditions (BC), whereby the flow velocity near the wall of the solid is $\mathbf{u} = \mathbf{0}$. Looking ahead to filtration modelling the fluid is in a continuum state, which justifies using a no-slip BC. Two main ways exist to model no-slip boundaries - the bounce-back and the halfway bounce-back methods. The former works by reflecting the distribution functions during a collision step if a solid node is encountered. Referring to the D2Q9 model in Figure 2.4, if there is a solid wall to the right, f_1, f_5 and f_8 at time t will be sent back to f_3, f_7 and f_6 at $t + \Delta t$. The other approach is the halfway bounce-back, where the distribution functions are reversed during the streaming step, effectively reflecting before they reach the solid node. Despite their names, both conditions simulate the wall as lying *approximately* halfway between the solid and fluid nodes. A schematic of both implementations is shown in Figure 2.10, where N is the last fluid node before the solid located at $N + 1$.

An important aspect of modelling fibrous structure is the ability to tune their affinity to the filtered component. This allows for the modelling of *contact angle* between drops and solids - a phenomenon arising due to surface and interfacial tensions between emulsion components and solids and one which plays an important role in filtration. A way to model this, is through a force expression similar



(a)

(b)

Figure 2.10: A schematic representation of (a) full-way; (b) half-way bounce back boundary conditions. Full-way implementations modify the collision step, whereas half-way implementations modify the streaming step. Figure adapted from [55].

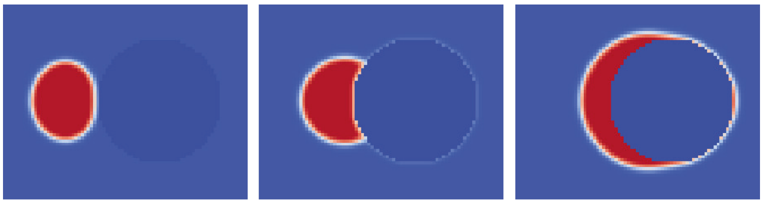


Figure 2.11: Density contours of a droplet impinging on a fibre. Three different contact angles in the η approach are shown. From left to right: $\eta = 0 \to \theta_c = 180°$, $\eta = 0.265 \to \theta_c = 90°$, $\eta = 1 \to \theta_c = 0°$.

to Eq. 2.25, $\mathbf{F}_{cs}$ quantifying the interaction between component c and the solid

$$\mathbf{F}_{cs}(\mathbf{x}, t) = -G_{cs}\psi_c(\mathbf{x}, t) \sum_i w_i s(\mathbf{x} + \mathbf{c}_i, t)\mathbf{c}_i \tag{2.31}$$

where $s(\cdot)$ is an indicator function having the value $s(\cdot) = 1$ if the location in its argument is a solid and $s(\cdot) = 0$ otherwise. This method is used in [78, 87]. Another approach to modelling adhesion is by assigning a fictitious density, ρ_w : $\rho_c \leq \rho_w \leq \rho_{\bar{c}}$ to the solid nodes which is then used to update the distributions at surrounding nodes, thus creating an interaction between fluid and solid. A *wall affinity* parameter, η is introduced defined as

$$\eta = \frac{\rho_w - \rho_c}{\rho_c - \rho_{\bar{c}}} \tag{2.32}$$

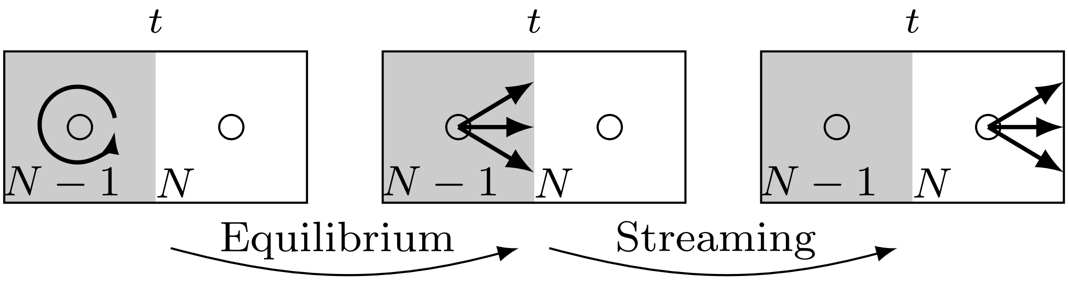
Setting, $\eta \in [0, 1]$ allows for the density of the solid nodes to be changed accordingly. In fact, this method stems from the free energy formulation for multiphase flows, mentioned in Section 2.2, but is used in different models [84, 88, 89]. This method is generally easier to implement and is flexible enough to simulate the full range of contact angles, $\theta_c \in [0, 180]°$. An example of this is shown in Figure 2.11.

Open boundary conditions

The Lattice Boltzmann models support a number of different boundary conditions. The simplest type are the periodic boundaries. In this scenario, the lattice domain wraps around on itself. To illustrate the point, consider a two-dimensional domain. If periodic boundaries are applied in one direction the field becomes topologically identical to a cylinder. If periodic conditions are used in both directions the domain will be identical to a torus. Having both boundaries as periodic is rare and is mainly used for demonstration purposes. In practice, periodic boundaries are useful when simulating flows that develop over large regions, *e.g.* Poiseuille flow [78].

Boundary conditions which allow the specification of pressure, velocity and other macroscopic flow variables, also exist. Pressure boundaries allow the simulation of flows driven by a pressure gradient and velocity inlet boundaries simulate a scenario in which the flow is already in motion by the time it reaches the computational space. The latter condition is useful for steady-state flows as it is usually used to set a uniform velocity in the domain. An extensive treatment for the specification of pressure and velocity boundaries with second order accuracy is given in [90]. The authors make an important remark about corners, *i.e.* the intersection of solid-like boundaries. This research uses the a simplified Dirichlet velocity inlet with a non-equilibrium portion of the distribution function, *i.e.* $f_i - f_i^{eq} = 0$. A schematic representation of this boundary condition for the left edge of the domain is shown in Figure 2.12. As before, N is the first fluid node after the inlet located at $N-1$ in this case.

The open boundary at the outlet of the domain is particularly important as it allows the simulation of *semi-infinite* domains at a manageable computational cost. A necessary condition for these boundaries is that they preserve the properties of the fluid across them as if the nodes immediately after the boundary are still in the domain. The authors of [91] provide a useful discussion of outflow



$$f_i(N-1,t) = \\ = f_i^{eq}(\rho_{N-1}, \mathbf{u}_{N-1}, t)$$

$$f_{1,5,8}(N,t) = \\ = f_{1,5,8}(N-1,t)$$

Figure 2.12: Simplified Dirichlet inlet boundary condition. The non-equilibrium distribution functions are set to zero. This inlet is suitable for the simulation of steady state flows.

BC, including issues related to their implementation. The majority of the work in this research was carried out using the *Neumann* outflow boundary condition, which sets the gradients of all macro- and microscopic variables to zero [91].

Details about other types of boundary conditions, not in the scope of this dissertation, can be found in [53, 58, 78].

2.3.4 Additional aspects

To complete the topic of this chapter, a few additional matters need to be mentioned, that are important when performing LBM simulations. This includes some of the advantages and disadvantages of LBM as compared to traditional CFD methods, as well as some useful extensions to the LBM model.

Comparison to other models

A short summary of some of the advantages that LBM possesses over traditional Navier-Stokes solvers, for modelling multiphase flow are outlined in Table 2.3.

Table 2.3: Models and requirements for simulating multiphase flow in complex media

Requirement	**LBM**	**CFD**
Fully resolved multiphase flow.	Components emerge from interaction forces.	Separate N-S equations are solved for each fluid in addition to coupling terms.
Interface modelling.	Dynamically evolving interface driven by energy minimisation and mesoscopic interactions.	Additional tools to track the interface.
Complex geometry modelling.	Boolean specification of solid nodes. Straightforward fluid-solid interactions and contact angle modelling.	Mesh adaptation required for complex geometries. Macroscopic boundary conditions.
Surface properties.	Affinity of solids to liquids modelled with a single parameter	Separate models are needed.
Incorporation of body forces.	A variety of methods.	A variety of methods.
Others.	First order PDE solver. Local interactions lead to efficient parallel implementation.	Second order PDE solver. Field-wide communications may lead to inefficiency.

Errors in LBM

Any numerical method has finite accuracy and lattice Boltzmann models are no exception. Clear understanding and appreciation of its limitations in computational sense helps keep one's expectations reasonable and leads to stable and meaningful results. The errors discussed here apply to LBM in general and are separate from the deficiencies pertaining to multiphase extensions.

In addition to the inaccuracies stemming from the use of discrete grids, there are five principal sources of error relevant to LBM [92]:

1. *Mach number error*: When recovering the incompressible Navier-Stokes momentum equations from LBM, their form includes gradients in the density of order $\mathcal{O}(|\mathbf{u}|^2)$. These gradients are non-physical for an incompressible fluid and lead to error terms $\mathcal{O}(|\mathbf{u}|^3)$ [93]. Since the Mach number, $Ma = |\mathbf{u}|/c_s$ is proportional to speed, these errors are typically given in terms of $Ma \to \mathcal{O}(Ma^3)$. Thus, to keep the assumption of incompressible flow, Ma in the simulation must be kept sufficiently low. Other ways to keep this error small is by increasing the spatial resolution and by having the relaxation time, τ as close as possible to its limiting value, $\tau = 0.5$ (refer to Eq. 2.21). This holds other problems as τ also defines how quickly the local velocities go to their equilibrium values.

2. *Finite-difference error*: LBM solves flow variable fields on a finite resolution lattice, which leads to non-smooth variations in these fields. Referring back to Eq. 2.19, its finite difference nature can be clearly seen. Thus increasing the spatial resolution, decreases field discontinuities and improves the accuracy of the results. It has to be noted that the computation time increases with the number of lattice nodes.

3. *Boundary error*: The two most common solid boundary conditions, the full bounce-back and the half-way bounce-back also have finite accuracy. The

former is accurate to first order, whereas the latter to second order [53] due to the effectively reduced lattice size.

4. *Spatial rounding error*: This issue is an artefact of the nature of the computational grid and appears with the inclusion of solid boundaries. Any spatial dimension can only be represented as an integer (*i.e.* multiple of 1, the lattice spacing) version of the original. Thus objects with fractional lengths, curves and in the special case circles are only accurate at a relatively high lattice resolutions. Additionally, for surfaces not aligned with the grid this leads to a "staircase" appearance and again is remedied at high resolutions.

A more thorough discussion on errors and stability in LBM can be found in [94].

Extensions to the model

There are many extensions to LBM to suit specific simulation needs in different fields. Here, a few commonly used variants for the simulation of incompressible, multiphase flows are mentioned.

In Section 2.2.1 a move from the transition matrix of EC-LBE to a single relaxation time parameter gave rise to LBGK. Some of the limitations of this simplification were also mentioned. A modification to the LBGK was made almost simultaneously with its derivation. The *multiple relaxation time* (MRT) lattice Boltzmann equation was introduced [95]. The model was developed to address some of the deficiencies of LBGK, the chief among which was the instability of LBGK at small τ. This translates to an inability to simulate small-viscosity (high Reynolds number) flows reliably. The principal difference between LBGK and MRT-LBE is the reintroduction of a full relaxation matrix, Λ. The use of a full matrix addresses another issue of LBGK - it allows different properties to be adjusted independently. Multiphase MRT-LBM models with good performance can be found in [96, 97].

Another improvement is the so-called *explicit forcing*, where any force terms are included directly in the LBE, instead of using *e.g.* velocity shift [98]. This improvement was built on top of the MRT-LBE and achieves large viscosity ratios.

2.4 Chapter summary

Simulating multicomponent and multiphase flows is important not only from scientific, but also from industrial point of view. This chapter introduced several modelling approaches that have traditionally been used for the simulation of multiphase flows. The historical development that led to the widely used BGK lattice Boltzmann method was also presented. The most common LBM modifications for modelling multiphase flows, were discussed with an emphasis on the Shan-Chen models.

In Chapter 5 some of the aspects discussed here will be revisited in the light of practical considerations and the application of LBM for filter modelling will be demonstrated. First, however, a way to ameliorate the computational cost of LBM simulations is presented in the next chapter.

CHAPTER 3

Gaussian process emulation

3.1 Surrogate modelling

Computer codes are frequently used in science and engineering to model physical processes. These codes are deterministic in their nature, meaning that there is no uncertainty about the output of the model once it has been observed. Furthermore this condition guarantees that the output of the model will be exactly the same as long as the input configuration remains unchanged. Regardless of this, the inputs of any model vary according to some, possibly joint probability distribution, which drives the output values to form a distribution of their own. Unless the model has a tractable expression to encode the relationship between its input and output, the output distribution will not be known prior to running the model. Theoretically, one can sample the input distribution many times and run the computer code at each sampled point to obtain the output distribution. In practice, whether or not this is possible is determined by the complexity of the computer model. This complexity has two main components - long running times and lack of closed form expression for the model. For any useful simulator, the latter is extremely common, mainly due to the complex nature of the process being modelled. Computational cost on the other hand depends on many factors, such as programming environment, hardware capability and input configuration. For the purpose of this dissertation, simulators whose running times are long enough to prohibit any type of analysis which requires a large number of model runs will be referred to as *expensive*[1]. For such codes the procedure of obtaining a distribution of output values as described above is limited by a computational budget. The computer code employed in this research to simulate coalescence is indeed expensive and a surrogate model is used to enable the execution of different analyses.

Let the values of the model inputs come from a joint probability distribution, denoted $p(\mathbf{x})$. Then, the collection of inputs can be treated as multivariate random variable, $\mathbf{X} \in \mathbb{R}^d$, where d is the dimensionality of the input space $\mathcal{X}$. For instance, consider the simple model $y = \eta(x) = 15x\sin(15x)$, shown in Figure 3.1(a). In this case, $d = 1$ and the input is assumed to be uniformly distributed in $[0, 1]$. If, additionally, $\eta(x)$ is considered to be expensive to evaluate (which it is not in this case), one is more likely to be presented with a finite collection of input-output pairs, $\mathcal{D} = [(x_1, y_1), \ldots, (x_n, y_n)]$ such as those shown in Figure 3.1(b), rather than a continuous curve. Remember, that the simulator is deterministic and the point values shown are equal to the output of the code with no uncertainty. This can be restated as a bounding condition on the form of the function, *i.e.* $\eta(x)$ must be such as to exactly pass through the points on Figure 3.1(b). This condition is not sufficient to learn the form of the function, however. The five values of y uniquely determine a fourth order

[1] There are no immediate implications about monetary cost here. However, if the analyst is paying to use a given computational platform, as is often the case, then a monetary value can be attached to using the code.

polynomial, $p_4(x) = \beta_0 + \sum_{i=1}^{4} \beta_i x^i$ with coefficients β_i for $i = \{0, \dots, 4\}$. This polynomial, shown as a solid line in Figure 3.1(c) is clearly very different from the original function. In fact, an infinite number of functions that pass through those points, satisfying the deterministic condition can be generated just from the polynomial family. A comparison between a few different polynomials is shown in Figure 3.1(c), where the original fourth order polynomial is compared to p_5, p_8 and p_{17}. Taking into account other types of functions leads to the notion that the code output can lie anywhere in the shaded region on Figure 3.1(d), where the area was trimmed to $y \in [-40, 40]$ only for visualisation purposes. Since there is a degree of uncertainty in the behaviour of the output, at locations different from those of the observed five points, it can be treated as *random*. This is due to the fact that the value of the code output can only be known once the code has been run, but to obtain the exact shape of $\eta(x)$ requires the code to be evaluated at an infinite number of points in the input space. Here, randomness does not mean that the function itself is random. Formally this is reflected by changing the notation of the mapping to $Y = \eta(\mathbf{X})$, where Y is a random variable with a probability distribution function $p(y)$.

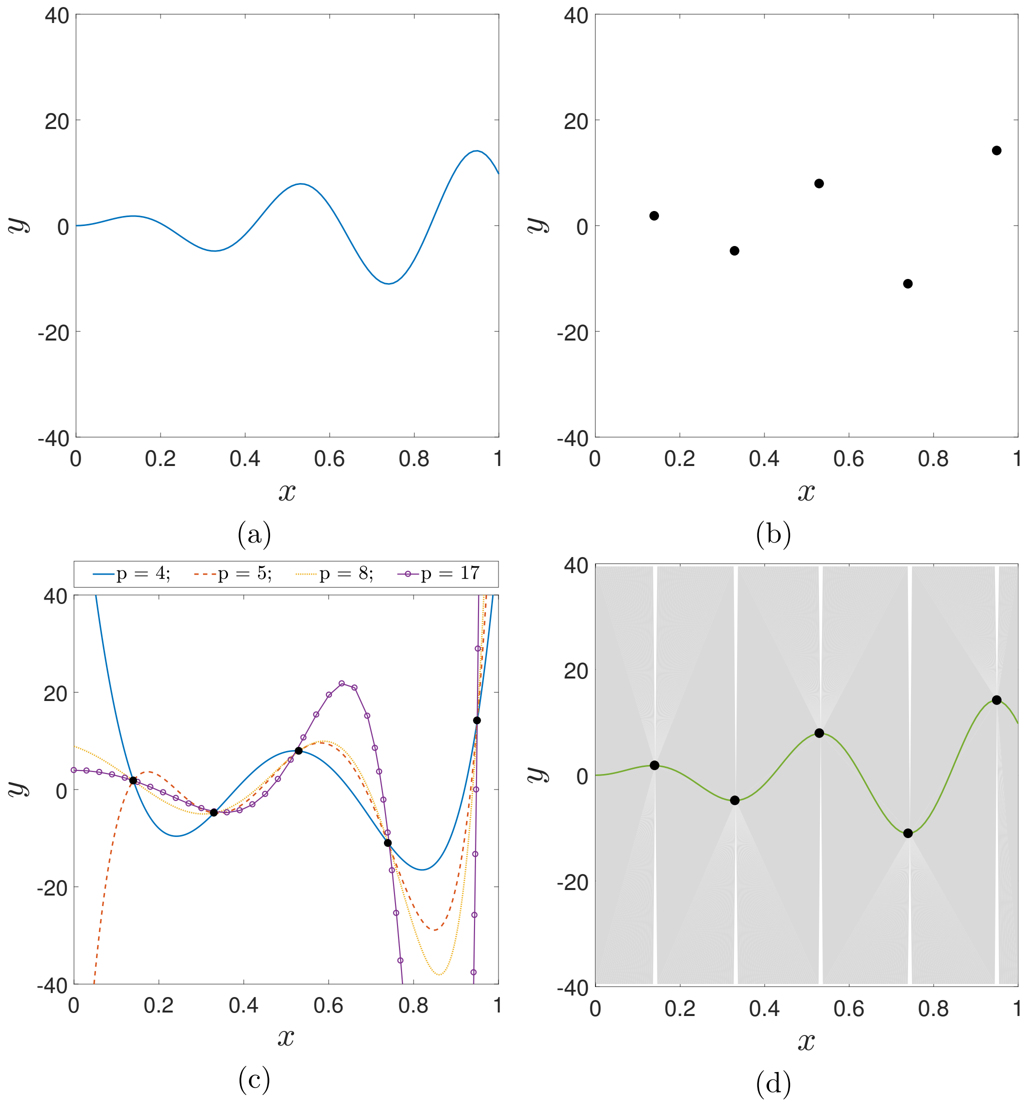


Figure 3.1: A demonstration of code output uncertainty. The original function is shown in (a). The sample in (b) is interpolated with polynomials of different orders as shown in (c). The shaded region in (d) demonstrates the uncertainty about the true function, when the particular sample is observed.

In order to alleviate the cost of expensive computer models, one can think of a stochastic representation of the output, which works as an approximation. These approximations are called surrogate models or metamodels [99, 100]. There exist a large variety of surrogate models that are used with different simulators.

Historically, one of the original metamodelling approaches is the *response surface method* [36]. This technique was first used with physical experiments and is based on the theory of linear regression. It is fairly widespread and is still in use today, albeit its usage has seen a decline in favour of more sophisticated and accurate methods, as will be seen in the literature review of Chapter 7. Other common metamodels include Taylor expansions [35], artificial neural networks [37], radial basis functions [38], support vector machines [39] and polynomial chaos expansions [101]. Each of these surrogate models, has its advantages and disadvantages as outlined in [99]. However, none of these methodologies is statistical in its nature and is thus unable to provide an estimation of the uncertainty introduced while using the surrogate without employing additional tools.

In contrast, the *Gaussian process emulator* (GPE) is a statistical surrogate model [102]. The term *emulator* is used because the GPE not only provides a *prediction* about the value of the code's output at a previously untested input combination, but also gives a closed-form expression for the uncertainty in the output of the real model at this point. Thus at any unobserved point the GPE provides a *distribution* for the simulator's output, which can consequently be used to compute sample statistics and can guide decisions in optimisation or emulator refinement. The Gaussian process emulator is used in the remainder of this work and is the subject of this chapter.

3.2 Gaussian random processes

A random variable X is said to have a normal distribution if the probability density function (PDF) of $X = x$, denoted $p(x)$ is given by

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right] \tag{3.1}$$

where μ and σ^2 are parameters of the distribution denoting its mean value and variance. The normal distribution, expressed as $\mathcal{N}(\mu, \sigma^2)$ and shown in Figure 3.2(a) is commonly used in science, both because of its frequent natural occurrence and due to the implications of the central-limit theorem, which states that the distribution of the means of samples converges to a normal distribution as the number of samples approaches infinity. A generalisation of the *univariate* Gaussian distribution which is defined over scalars, is the *multivariate* Gaussian distribution function defined over vectors and shown in Figure 3.2(b). A random vector, $\mathbf{X} \in \mathbb{R}^d$ is considered to have a multivariate normal distribution, if the probability of $\mathbf{X} = \mathbf{x}$ is given by

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right] \tag{3.2}$$

where $\mathbf{\Sigma} = \sigma^2\mathbf{C}$ is a covariance matrix, given by the product of the variance and correlation matrix. The meaning of $\boldsymbol{\mu}$ is the same as above, but here it is a vector, $\boldsymbol{\mu} \in \mathbb{R}^d$. The operators $|\cdot|$ and $(\cdot)^T$ denote the determinant of a matrix and the transpose of an array, respectively. The Gaussian distribution possesses a number of desirable properties which consolidates their widespread use. Let $\mathbf{X} \sim \mathcal{N}(\mu, \mathbf{\Sigma})$, that is $\mathbf{x}$ is a vector whose elements have probability given by Eq. 3.2. Also, let $\mathbf{x}$, $\boldsymbol{\mu}$ and $\mathbf{\Sigma}$ be partitioned as follows

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} \quad \mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{11} & \mathbf{\Sigma}_{12} \\ \mathbf{\Sigma}_{21} & \mathbf{\Sigma}_{22} \end{bmatrix}$$

The following properties are stated without proof. For formal proofs, which are beyond the scope of this dissertation, see [103].

1. The *marginal* distributions are Gaussian

$$
\begin{aligned}
\mathbf{x}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}) \\
\mathbf{x}_2 &\sim \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22})
\end{aligned}
$$

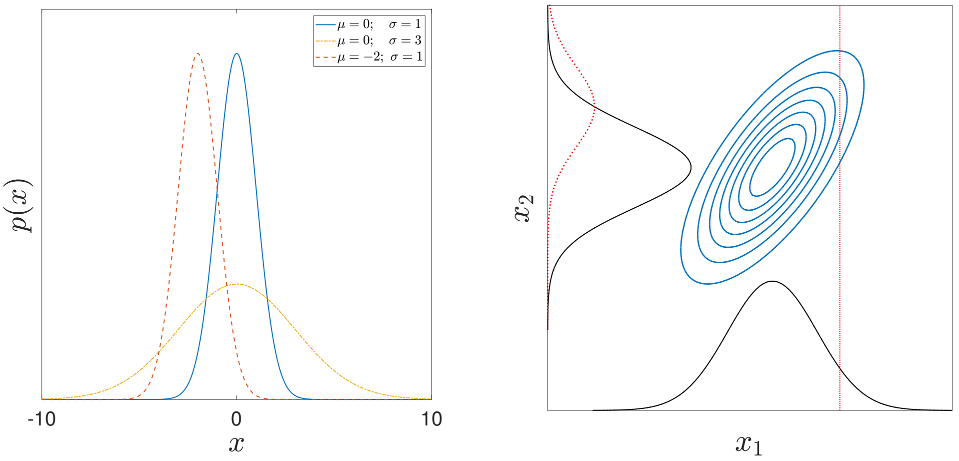
2. The *conditional* distributions are Gaussian:

$$
\begin{aligned}
\mathbf{x}_1|\mathbf{x}_2 &\sim \mathcal{N}(\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2), \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}) \\
\mathbf{x}_2|\mathbf{x}_1 &\sim \mathcal{N}(\boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}(\mathbf{x}_1 - \boldsymbol{\mu}_1), \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12})
\end{aligned}
$$

3. If $\mathbf{x}_1$ and $\mathbf{x}_2$ are also independent, any linear combination of the two is Gaussian:

$$
\begin{bmatrix} a & b \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} a & b \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \begin{bmatrix} a & b \end{bmatrix} \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \right)
$$

A visual representation of the marginalization and conditioning properties is shown in Figure 3.2(b). There, the two marginal distributions on x_1 and x_2, plotted in black solid line, clearly exhibit Gaussian form. Furthermore, the red line in Figure 3.2(b) depicts a scenario in which the value of x_1 is known precisely. The resulting conditional distribution $p(x_2|x_1)$, shown as a red dotted curve is also Gaussian. As mentioned in Section 3.1, the aim of surrogate modelling is to learn the input/output mapping provided by the computer model. To achieve this, a generalization of the Gaussian distribution, referred to as a *Gaussian process* (GP) is employed. A GP is a generalization in that it is defined in a functional space of infinite dimension, just as the multivariate Gaussian distribution is defined over a finite vector space. Thus, a GP provides a distribution



(a)

(b)

Figure 3.2: Univariate and bivariate Gaussian distributions. The effect of varying the parameters in Eq. 3.1 is shown in (a). The properties of a Gaussian distribution under marginalization and conditioning are depicted in (b) as black solid curves, and dotted red line and curve, respectively.

over random functions. An important clarification must be made here. When the term *random function* is used it is implied that, instead of a random number or a vector of numbers, each draw from a Gaussian process is a *randomly chosen* deterministic function. Even though it is chosen probabilistically, the function itself is not stochastic. Formally a Gaussian process is defined as:

Definition 1. *Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the d-dimensional input space, then $Z(\mathbf{x})$ for $\mathbf{x} \in \mathcal{X}$ is a Gaussian Process if for any subset $\{\mathbf{x}_1 \dots \mathbf{x}_m\} \subseteq \mathcal{X}$, $[Z(\mathbf{x}_1) \dots Z(\mathbf{x}_m)] \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.*

Just as the Gaussian distribution is fully characterized by a mean and covariance matrix, the GP is characterized by its mean and covariance functions, $m(\cdot)$ and $c(\cdot, \cdot)$, respectively. The mean function can be any real function of $\mathbf{x} \in \mathcal{X}$, but the covariance function or *kernel*, must produce a symmetric positive semi-definite covariance matrix, $\mathbf{C}$, with elements $C_{ij} = c(\mathbf{x}_i, \mathbf{x}_j)$.

3.3 Gaussian process emulation

Gaussian process emulation is concerned with building statistical approximations to the output of expensive computer codes. This means that, besides providing an interpolator, a Gaussian process emulator (GPE) is capable of quantifying the uncertainty about the interpolating function which arises from the fact that the output of the simulator has not been observed at all locations of interest. Being statistical, the construction of GPEs can be approached from either classical or Bayesian standpoints. The main difference between the two is that the Bayesian method attempts to account for all sources of uncertainty, including uncertainty stemming from parameter estimation. In a Bayesian setting, the uncertainty about parameters is represented through probability distributions. This results in a full probabilistic model that takes into account uncertainty in the parameters and in the data generating process. From a classical perspective the surrogate is constructed as an interpolator of the data, which generates predictions as a linear combinations of observations. Additionally, model parameters are seen as unknown, but fixed quantities that can be estimated from the data. In this setting, the surrogate is commonly known as *Kriging* or the *best linear unbiased predictor* [104, 105]. Fundamental differences between the Bayesian and classical approaches exist, and their superiority is still subject to debate [106]. This research adopts the technical formulation of the Bayesian framework, but retains the conceptual simplicity of the classical approach.

3.3.1 Emulator construction

Having modelled the output of the simulator as a random variable makes it possible to express the uncertainty in the output due to lack of data as a Gaussian

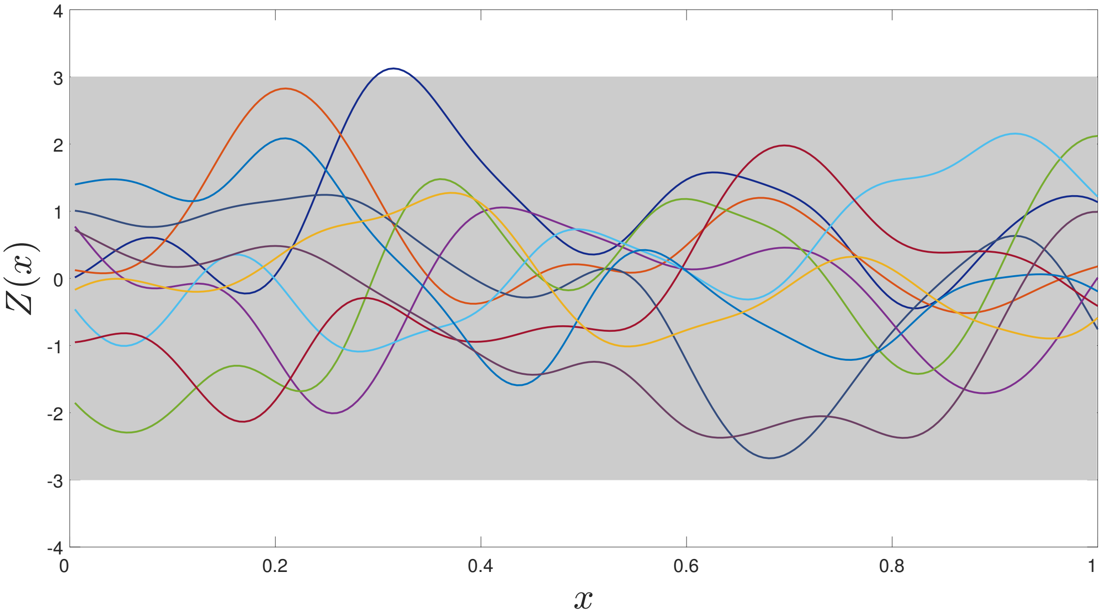


Figure 3.3: Random draws from a zero mean GP, $Z(x)$. Shaded area indicates the interval $\pm 3\sigma$ containing 99.7% of the realisations.

process. The main assumption is that the output of the simulator is of the form

$$\eta(\mathbf{x}) = h(\mathbf{x})^T \boldsymbol{\beta} + Z(\mathbf{x}) \tag{3.3}$$

where $h(\cdot) \in \mathbb{R}^q$ is any real-valued function that maps $\mathbb{R}^d \to \mathbb{R}^q$, $\boldsymbol{\beta}$ is a vector of unknown coefficients and $Z(\mathbf{x})$ is a Gaussian process with zero mean and covariance $\sigma^2 c(\mathbf{x}, \mathbf{x}'; \boldsymbol{\psi})$, where σ^2 is a scale parameter and $\boldsymbol{\psi}$ is a parameter specifying the behaviour of the correlation function, $c(\cdot, \cdot; \boldsymbol{\psi})$. The form assumed in Eq. 3.3 implies that the global trend of the output is captured by $h(\mathbf{x})^T \boldsymbol{\beta}$ whilst the local variations are modelled through the Gaussian process.

The form assumed for the computer model in Eq. 3.3 places a GP prior on the output and allows all properties of Gaussian distributions to be used. A set of example realisations from a GP prior is shown on Figure 3.3. In particular since any finite collection of points from a Gaussian process is distributed as a multivariate Gaussian, following Property 2 in Section 3.2, it can be seen that the distribution of the computer code output at unobserved input values, $\mathbf{x}^*$, *given* the data already obtained must follow a normal distribution

$$\eta(\mathbf{x}^*)|\mathcal{D} \sim \mathcal{N}(m(\mathbf{x}^*), c(\mathbf{x}^*, \mathbf{x}^*)) \tag{3.4}$$

with some mean and covariance functions. In Eq. 3.4 $\mathcal{D} = \{\mathbf{x}_i, y_i = \eta(\mathbf{x}_i)\}_{i=1}^n$ are n evaluations from the original code on which to condition future predictions. Different strategies for optimal sampling are discussed in Section 3.3.2. Remembering that

$$\mathbf{y}|\boldsymbol{\beta}, \sigma^2 \sim \mathcal{N}(\mathbf{H}\boldsymbol{\beta}, \sigma^2\mathbf{C}) \tag{3.5}$$

where the only undefined term is $\mathbf{H} = (h(\mathbf{x}_1), \ldots, h(\mathbf{x}_n))^T$, leads to the notion that in order to obtain a distribution for the output of the simulator unconditional on the GPE parameters in Eq. 3.3, their values need to be estimated from data. The regression coefficients $\boldsymbol{\beta}$ and the scale parameter, σ^2 can be estimated analytically. The parameters of the correlation function, $\boldsymbol{\psi}$, whose role in Eq. 3.5 is implied through the presence of $\mathbf{C}$ can be estimated through direct search using optimisation algorithms. This is discussed in Section 3.3.4.

From Eq. 3.5 the likelihood of $\boldsymbol{\beta}$ and σ^2 can be expressed as

$$p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{n/2}|\mathbf{C}|^{1/2}} \exp\left[-\frac{(\mathbf{y}-\mathbf{H}\boldsymbol{\beta})^T\mathbf{C}^{-1}(\mathbf{y}-\mathbf{H}\boldsymbol{\beta})}{2\sigma^2}\right] \tag{3.6}$$

At this point a fully classical construction (*e.g.* for Kriging) proceeds by maximising the expression in Eq. 3.6 with respect to both $\boldsymbol{\beta}$ and σ^2. However, this approach neglects the fact that these parameters can also come from some distribution. Instead, the posterior distribution of the parameters given the data is computed here. It is common to assume a weak prior for the joint distribution of $\boldsymbol{\beta}$ and σ^2

$$p(\boldsymbol{\beta}, \sigma^2) \propto \frac{1}{\sigma^2} \tag{3.7}$$

This prior is used to allow the data to have a greater effect on the predictions from the emulator. In practice, other types of priors, including informative priors can be used to incorporate expert knowledge and other information. From Bayes' rule the posterior parameter distribution is

$$p(\boldsymbol{\beta}, \sigma^2|\mathbf{y}) \propto p(\mathbf{y}|\boldsymbol{\beta}, \sigma^2)p(\boldsymbol{\beta}, \sigma^2) \tag{3.8}$$

Eq. 3.6 can be reparameterized from a distribution for $\mathbf{y}$ to one for $\boldsymbol{\beta}$ yielding a normal posterior distribution and an estimator

$$\hat{\boldsymbol{\beta}} = (\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H}^T)^{-1}\mathbf{H}^T\mathbf{C}^{-1}\mathbf{y} \tag{3.9}$$

Multiplying Eq. 3.6 and Eq. 3.7, separating from the distribution of $\boldsymbol{\beta}$ and combining terms results in a inverse-gamma posterior distribution for σ^2 with an unbiased estimator

$$\hat{\sigma}^2 = \frac{\mathbf{y}^T(\mathbf{C}^{-1} - \mathbf{C}^{-1}\mathbf{H}(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1}\mathbf{H}^T\mathbf{C}^{-1})\mathbf{y}}{n - q - 2} \tag{3.10}$$

Combining Eq. 3.5, Eq. 3.9 and Eq. 3.10 gives a posterior predictive distribution for the code output, conditional on the observed data and parameter estimators as

$$\eta(\mathbf{x}^*)|\mathcal{D}, \hat{\boldsymbol{\beta}}, \hat{\sigma}^2 \sim \mathcal{N}(m(\mathbf{x}^*), c(\mathbf{x}^*, \mathbf{x}^*)) \tag{3.11}$$

with posterior predictive mean given by

$$m(\mathbf{x}^*) = h(\mathbf{x}^*)\hat{\boldsymbol{\beta}} + \mathbf{t}(\mathbf{x}^*)^T\mathbf{C}^{-1}(\mathbf{y} - \mathbf{H}\hat{\boldsymbol{\beta}}) \tag{3.12}$$

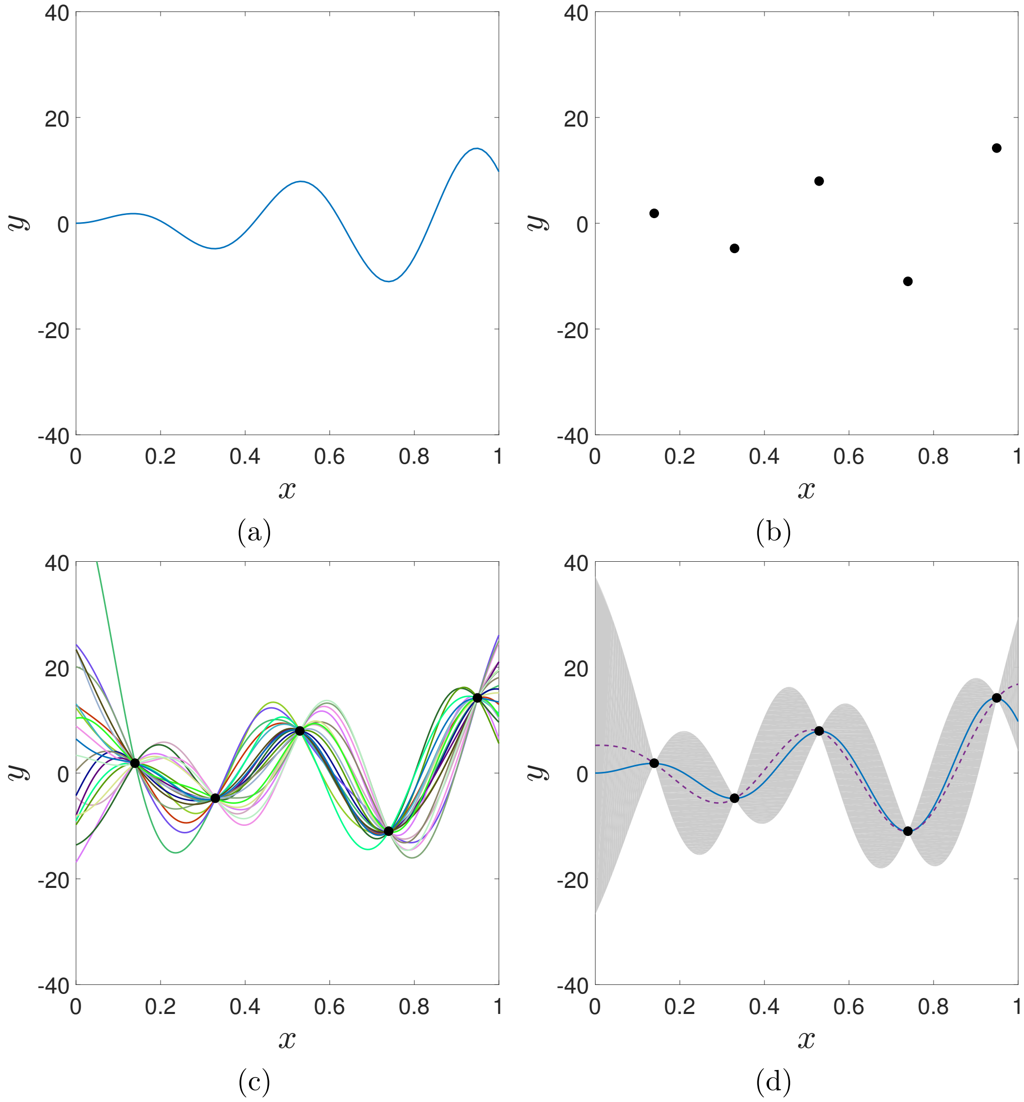
and posterior predictive covariance

$$c_2(\mathbf{x}^*, \mathbf{x}'^*) = \hat{\sigma}^2 \Big[c_1(\mathbf{x}^*, \mathbf{x}'^*) + (h(\mathbf{x}^*)^T - \mathbf{t}(\mathbf{x}^*)^T \mathbf{C}^{-1}\mathbf{H})(\mathbf{H}^T\mathbf{C}^{-1}\mathbf{H})^{-1} \times \\ \times (h(\mathbf{x}'^*)^T - \mathbf{t}(\mathbf{x}'^*)^T\mathbf{C}^{-1}\mathbf{H})^T \Big] \tag{3.13}$$

with c_1 given by

$$c_1(\mathbf{x}^*, \mathbf{x}'^*) = c(\mathbf{x}^*, \mathbf{x}'^*) - \mathbf{t}(\mathbf{x}^*)^T\mathbf{C}^{-1}\mathbf{t}(\mathbf{x}'^*) \tag{3.14}$$

Eq. 3.14 implies that the prior uncertainty, given by $c(\mathbf{x}^*, \mathbf{x}'^*)$ is reduced after observing the data. In Eq. 3.12 to Eq. 3.14, $\mathbf{C} \in \mathbb{R}^{n\times n}$ is such that $C_{ij} = c(\mathbf{x}_i, \mathbf{x}_j; \hat{\boldsymbol{\psi}})$; $\mathbf{t}(\cdot) \in \mathbb{R}^n$ such that $\mathbf{t}(\cdot) = (c(\cdot, \mathbf{x}_1; \hat{\boldsymbol{\psi}}), \ldots, c(\cdot, \mathbf{x}_n; \hat{\boldsymbol{\psi}}))^T$. The covariance function in Eq. 3.13 is composed from the covariance of the model distribution conditional on the model parameters (Eq. 3.14) and terms produced during parameter estimation. After the emulator is constructed, its posterior predictive mean can be evaluated at a much lower cost than that associated with obtaining realisations from the simulator. This is due to the fact that the expression in Eq. 3.12 does not depend on $\eta(\cdot)$. Additionally, the covariance function, Eq. 3.13 provides the uncertainty about the value of each new point sampled from the mean. Figure 3.4 revisits the example from Figure 3.1. Conditioning on the data, Figure 3.4(c), demonstrates how the GP prior, from Figure 3.3 is combined with observations to eliminate the uncertainty at these points. In the limiting case, shown in Figure 3.4(d), the GPE provides both a surrogate for the function and plausible uncertainty bounds. More details on the general procedure of constructing the GPE can be found in [100] from a classical perspective and [107, 108] from a Bayesian standpoint.



(a)

(b)

(c)

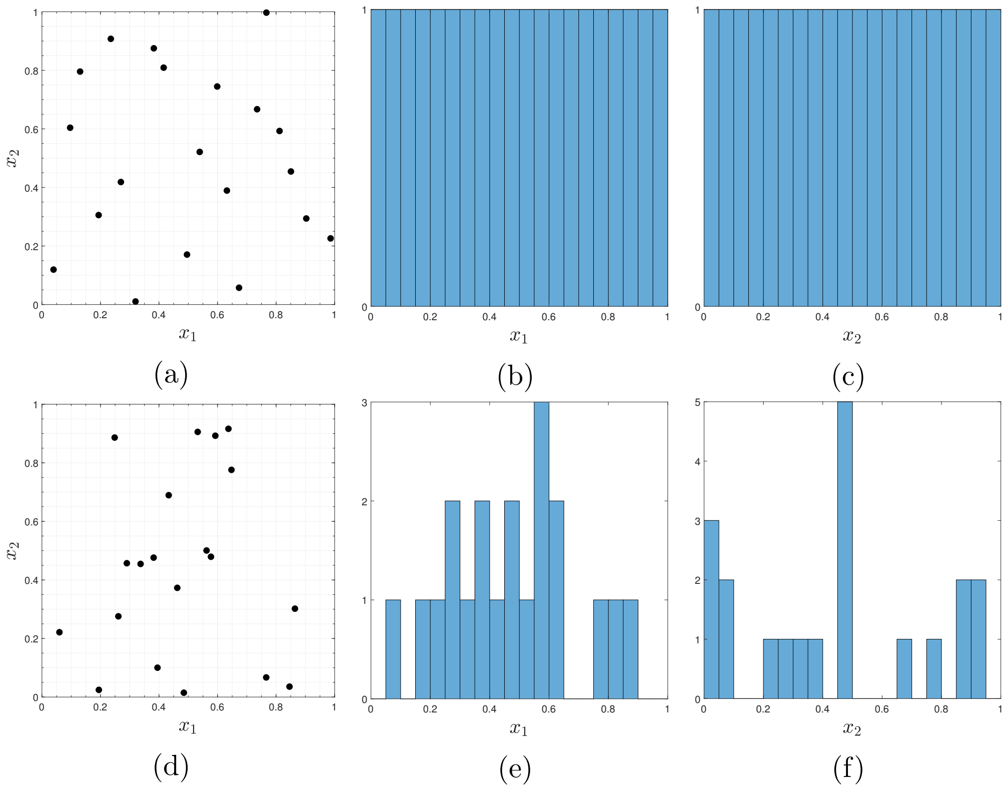
(d)

Figure 3.4: A simple demonstration of GPE for the quantification of code output uncertainty.

3.3.2 Training an emulator

The study of *design of experiments* (DoE) explores optimal strategies in selecting combinations of predictor values, to determine the variation in the response of a dependent variable. Different DoE methods were first devised in the context of physical experimentation, but have commonly been applied to computational studies under the name *design and analysis of computer experiments* [109].

Since the cost of running the simulator is likely to restrict the number of available evaluations, n, the *training* sample should be chosen such that it explores the input space uniformly, that is to say the training design is *space-filling* in all dimensions. Since the predictive mean of the GPE in Eq. 3.12 is a linear combination of distances between all training points and the prediction site, a space-filling design ensures that any unobserved point in the input space is not too far away from the observed data. There are several ways to construct a sampling plan which conforms to these criteria. Notes on widespread approaches, can be found for example in [100]. One possible strategy, which yields reasonable results and is not too computationally expensive is the *maximin, stratified Latin Hypercube Sampling* (LHS) [110]. The term *maximin* refers to the sample, chosen among other similar samples, which minimizes the maximum distance between points. LHS offers superior distribution of data points as compared to random sampling. Figure 3.5 juxtaposes the two strategies in terms of exploration of individual variable ranges and in distribution of samples. Comparing the histograms in Figure 3.5(b) to Figure 3.5(e) and in Figure 3.5(c) to Figure 3.5(f) demonstrates the ability of LHS to distribute the samples evenly along each dimension. More on the use of space-filling sampling plans can be found in [111]. Another aspect of obtaining the training sample for the GPE is to choose its size. There are different opinions on this matter as briefly stated in Section 7.1.2. A commonly used informal rule states that the number of samples, n is related to the dimensionality of the problem, d as $n \approx 10d$. This rule is supported by analyses carried out in [112], where the authors provide quantitative reasoning about the validity of the rule and deem it a “good approximation”. As discussed in the following sections, the sample size cannot be determined simply by a rule of thumb, but depends on the model, choice of kernels, $c(\cdot, \cdot)$ and global trend functions $h(\cdot)$.



(a) (b) (c)

(d) (e) (f)

Figure 3.5: Comparison between an LHS ((a)-(c)) and uniform random ((d)-(f)) sample. The number of data points in each case is $n = 20$. The Latin hypercube sample fills each dimension uniformly.

3.3.3 Trends and kernels

Trends

As mentioned before, from the model structure in Eq. 3.3 it can be deduced that the role of the global trend term, $h(\mathbf{x})^T\boldsymbol{\beta}$ is to provide an offset for the zero-mean GP, $Z(\mathbf{x})$. Referring back to Section 3.3.1, $h(\mathbf{x})$ can be any real-valued function. For example, it can be a polynomial p_{q-1} of order $q-1$, or another analytic function. The role of $\boldsymbol{\beta}$ is to place appropriate weights onto each of the q terms, determining their influence. For problems with multidimensional input spaces $h(\mathbf{x})$ can contain any number of monomials for different input variables, without the need for all dimensions to participate in the weighted sum. The combined regression term is used to encapsulate beliefs about the general shape of the model output. These can either be provided by human experts, or can be inferred from input/output plots. Such exploratory analysis will be used Section 5.6. Choosing an appropriate form for $h(\mathbf{x})$ can be especially useful when emulating models with high dimensional inputs with a small number of observed points. Another reason for incorporating a functional global trend is to improve the quality of the predictions away from the observed data. For most covariance functions, Eq. 3.12 will return to the mean, $h(\mathbf{x})^T\boldsymbol{\beta}$ if there are no observations within a distance determined by $\boldsymbol{\psi}$. This point is demonstrated in Figure 3.6, which shows predictions of aircraft drag versus flight angle of attack and compares constant to a quadratic trend. In this example, theory suggests that the relationship is quadratic allowing this knowledge to be used in the GPE. Whenever such intuition about the emulated model is lacking, $h(\mathbf{x})$ is taken to be a constant with $h(\mathbf{x}) = 1$ and $\beta \in \mathbb{R}$, charging $Z(\mathbf{x})$ in Eq. 3.3, with the responsibility of capturing the behaviour of the underlying function [99].

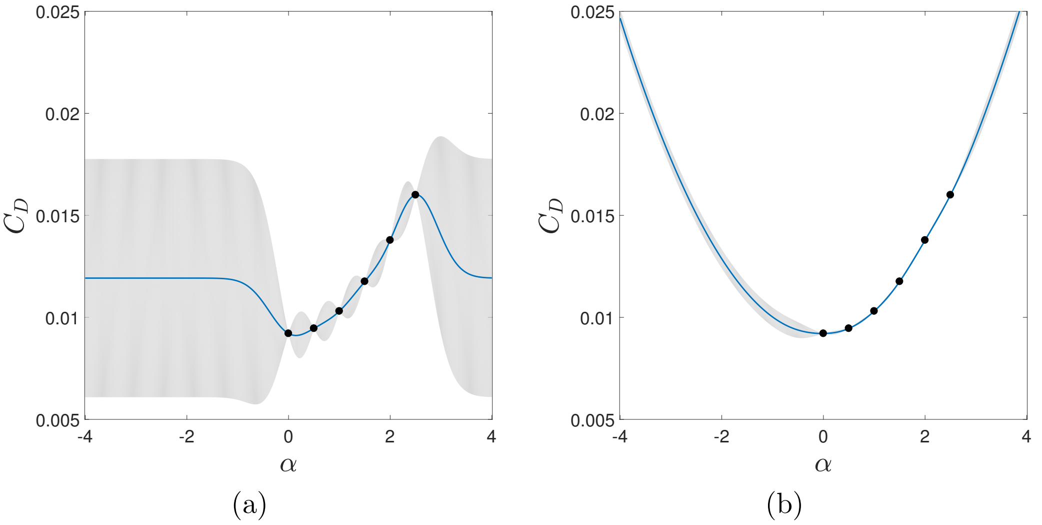


Figure 3.6: GPE for a drag polar with a constant trend in (a) and a quadratic trend in (b). The quality of the predictions away from the data improves for the quadratic term.

Kernels

The choice of covariance function directly affects the structure of the data and its information content. There are many available kernels, but the requirements for a given function to form a valid kernel are more strict than those imposed on $h(\mathbf{x})$.

Definition 2. *Any valid kernel produces a covariance matrix,* $\mathbf{C} \in \mathbb{R}^{n\times n}$ *which is both symmetric, that is* $\mathbf{C} = \mathbf{C}^T$ *and positive semi-definite, that is* $a^T\mathbf{C}a \geq 0$, $\forall a \in \mathbb{R}^n$.

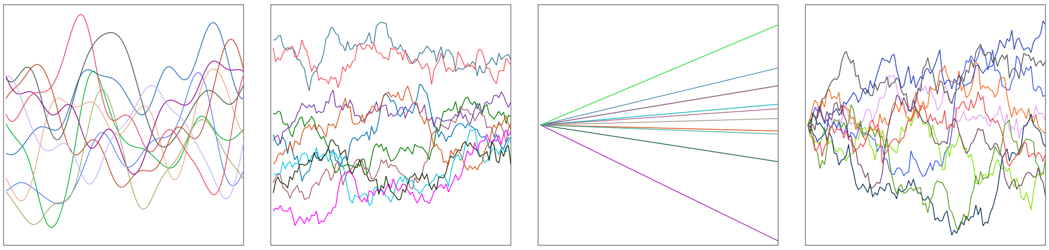
Similar to the global trend, the kernel can also reflect prior knowledge about the model to be emulated. Covariance functions are most commonly divided into two classes: *stationary* and *non-stationary*. Stationary kernels are those whose output depends on the distance between points, rather than on their location in $\mathcal{X}$. This means that the kernel can be written as $c(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\psi}) = c(|\mathbf{x}_i - \mathbf{x}_j|; \boldsymbol{\psi})$. Non-stationary kernels are those whose value may depend on distance between points, but also on their position. Furthermore, when emulating models with

Table 3.1: Some commonly used kernels.

Name	$c(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\psi})$
Exponentiated quadratic	$\exp\left(-\sum_{k=1}^{k=d}\frac{\lvert x_{ik}-x_{jk}\rvert^2}{\psi_k^2}\right)$
Matérn 3/2	$(1+\sqrt{3r})\exp\left(-\sqrt{3r}\right)$ $r=\sum_{k=1}^{k=d}\frac{(x_{ik}-x_{jk})^2}{\psi_k^2}$
Dot product	$\mathbf{x}_i^T\mathbf{x}_j$
Brownian	$\min(\mathbf{x}_i, \mathbf{x}_j)$

multidimensional input, kernels can have a single correlation parameter, $\boldsymbol{\psi} = \psi \in \mathbb{R}$ with the same value in each dimension, or be $\boldsymbol{\psi} \in \mathbb{R}^d$ which is referred to as *automatic relevance determination* (ARD). ARD means that depending on the covariance function, the value of $\psi_i : i \in \{1, \ldots, d\}$ indicates the importance of the i^{th} input to the variability of the output[2].

Commonly used kernels include the *exponentiated quadratic* and Matérn family for the stationary class, and *Brownian* and *dot product* kernels for the non-stationary one. The functional form of these kernels is given in Table 3.1. Sample draws from GP with each of the four kernels are shown in Figure 3.7. More information on these and other types of covariance functions can be found in [100, 102].



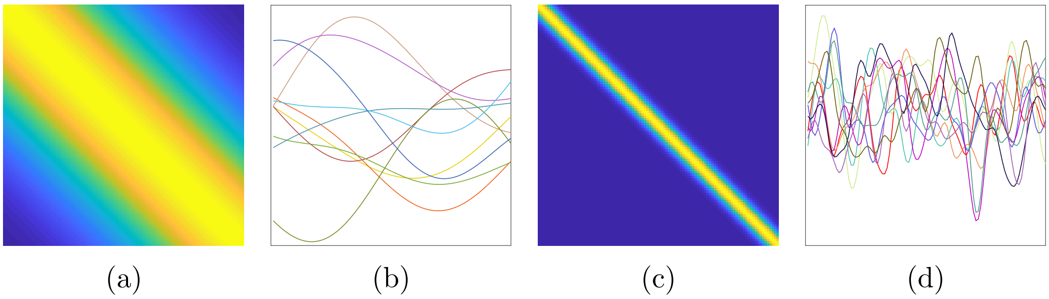
(a) (b) (c) (d)

Figure 3.7: Random samples from a GP with different kernels. (a) exponentiated quadratic; (b) Matérn 3/2; (c) linear; (d) Brownian.

[2] This matter is the principal aim of sensitivity analysis, discussed in Chapter 6.

3.3.4 Kernel parameter estimation

The estimates of $\boldsymbol{\beta}$ and σ^2 can be obtained assuming the value of $\boldsymbol{\psi}$ to be known. In reality this is not the case and instead $\boldsymbol{\psi}$ needs to be estimated from the data. Correlation parameters play an important role in the predictive capability of the GPE and their accurate estimation is very important. Figure 3.8 demonstrates, the effect on varying $\boldsymbol{\psi}$ for an exponentiated quadratic kernel on the covariance matrix, $\mathbf{C}$ and the shape of the realisations of the GP prior. For relatively large values of ψ, even points that are far apart in space are strongly correlated as seen by the wide yellow band in Figure 3.8(a), resulting in the smooth functions from the GP shown in Figure 3.8(b). Conversely, as ψ decreases, the correlation is only preserved between points that are close in $\mathcal{X}$ and hence on the diagonal of C in Figure 3.8(c), producing the irregular functions from Figure 3.8(d). The process of estimating the hyperparameters of the GPE bears the largest computational expense in constructing the emulator[3] and thus many approaches have been designed to improve the efficiency of the process. For instance, the posterior mode of the parameter distribution given data can be used, following [113]. Another method is to use *cross-validation* [107]. As stated earlier, this work adopts a classical perspective on the matter and estimates $\boldsymbol{\psi}$ via *maximum likelihood esti-*



(a) (b) (c) (d)

Figure 3.8: Random samples and covariance matrices of a GP with different correlation parameters. (a)-(b) $\psi = 0.5$, (c)-(d) $\psi = 0.05$. High correlation values are shown in yellow and low ones in purple.

[3] That is, after generating observations from the original code.

mation (MLE). This method provides estimates for the parameter values, which are *most likely* to have generated the data and is commonly used in classical statistics. A drawback of MLE is that it only provides a point estimate for $\boldsymbol{\psi}$ which might not correspond to a robust parameter value. In practice this issue can be identified and corrected for, using well-known diagnostics (described in Section 3.3.5). The estimation proceeds as follows. The optimal value of $\boldsymbol{\psi}$ is taken as the one that maximizes Eq. 3.6. To simplify the algebra and improve stability [103], the natural logarithm of the likelihood is taken, resulting in the *log-likelihood*

$$\begin{aligned}\ln[f(\mathbf{y}|\boldsymbol{\beta},\sigma^2)] = &-\frac{n}{2}\ln(2\pi)-\frac{n}{2}\ln(\sigma^2)-\frac{1}{2}\ln|\mathbf{C}|- \\ &-\frac{(\mathbf{y}-\mathbf{H}\boldsymbol{\beta})^T\mathbf{C}^{-1}(\mathbf{y}-\mathbf{H}\boldsymbol{\beta})}{2\sigma^2}\end{aligned} \tag{3.15}$$

Substituting Eq. 3.9 and Eq. 3.10 in Eq. 3.15, simplifying and removing constant terms, yields the *concentrated* log-likelihood:

$$\ln[f(\mathbf{y}|\boldsymbol{\beta},\sigma^2)] \approx -\frac{n}{2}\ln(\hat{\sigma}^2)-\frac{1}{2}\ln|\mathbf{C}| \tag{3.16}$$

The concentrated log-likelihood is used in computational implementations of MLE to increase the efficiency of the procedure and has been shown to produce likelihood values close to those obtained from the full log-likelihood [100]. In order to maximize the expression in Eq. 3.16 it could either be differentiated with respect to $\boldsymbol{\psi}$ or a direct search algorithm can be employed to find optimal values. The latter approach is preferred here, essentially casting the estimation as an optimisation problem. There is a wide range of available numerical solvers to search the likelihood, such as the Nelder-Mead simplex algorithm [114], simulated annealing [115], particle swarm optimisation and genetic algorithms [116], among others. Markov chain Monte Carlo [117] methods are also widely used in the literature.

A *genetic algorithm* (GA) is used in this implementation, as it is a method commonly applied in optimising functions with complex landscapes. In comparison, a local method such as the Nelder-Mead simplex could converge on a local optimum. Simulated annealing is also a global search method, but it requires an initial point, which could prove difficult to specify for complex likelihoods or higher-dimensional problems. In contrast GA only requires search boundaries. A caveat of this method is its computational cost which increases with both the number of dimensions d and the number of training sample points, n. The first dependence is due to the increased number of the fitness function (Eq. 3.16) evaluations required to explore search spaces with higher dimensionality. At every evaluation, the correlation matrix, $\mathbf{C}$ is inverted to provide estimates for $\boldsymbol{\beta}$ and σ^2. In practice, this operation scales as $\mathcal{O}(n^3)$ giving the dependence on n.

A problem independent of the search algorithm used for MLE, is the stability of $\mathbf{C}$. For large n at small d, $\mathbf{C}$ can become *ill-conditioned*, and hence non-positive semi-definite. Similar problems can occur at certain values of $\boldsymbol{\psi}$ and are used to navigate the search. However, for inappropriately constructed training data the space defined by $\boldsymbol{\psi}$ can become difficult to negotiate, prohibiting reliable parameter estimation. Different methods of *regularising* $\mathbf{C}$, improving its conditioning have been explored in literature (e.g. using a *nugget* [118], penalised likelihood [119] and others).

3.3.5 Diagnostics

As pointed in the discussion above, Gaussian process emulators construct an approximation to the output of the simulator. This requires some assumptions and introduces additional uncertainty. Unless the surrogate correctly represents the simulator output any inferences made using it will be invalid. Therefore the surrogate needs to be tested and validated. Typically, this is done by using the emulator to predict the response of the code for a set of input configurations, $\mathbf{x}^*$

unobserved during the training process. For benchmark tests, where the model under consideration is not expensive in reality, a visual check can be performed of the quality of the predictions. However, this approach becomes infeasible for functions with more than two inputs or real expensive codes. To help with assessing the quality of the GPE predictions, there exist a variety of methods which provide both point-wise metrics and more complete diagnostics. A comprehensive discussion on the topic is provided in [120]. Here, only the methods used in the project are presented. A common point-wise figure of merit is the *root mean squared error* (RMSE)

$$D^{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \mathbb{E}[\eta(x_i^*)|\mathbf{y}])^2} \tag{3.17}$$

where m is the number of test points and the expected value of the posterior predictive distribution, $\mathbb{E}[\eta(\mathbf{x}_i^*)|\mathbf{y}]$ is given in functional form in Eq. 3.12. This notation is adopted to consolidate the notion that the GPE is a probabilistic model with well-defined moments. The RMSE provides a summary of the quality of the surrogate, but its value can be easily affected by a small number of large errors, leading to erroneous conclusions. Moreover it only features information about the mean of the GPE and ignores its variance. A diagnostic which utilizes the complete information provided by the GPE is the set of *individual prediction errors* (IPE)

$$D_i^I = \frac{y_i - \mathbb{E}[\eta(x_i^*)|\mathbf{y}]}{\sqrt{\mathbb{V}[\eta(x_i^*)|\mathbf{y}]}} \tag{3.18}$$

where $\mathbb{V}[\eta(\mathbf{x}_i^*)|\mathbf{y}]$ is the variance of the posterior predictive distribution and is defined in Eq. 3.13. Using Eq. 3.18, the distribution IPE can be visualised and studied to detect problems with the emulator. The test sample size m, is mainly dictated by the cost of the computer model, but can be determined by other considerations.

In the following sections and chapters RMSE, IPE and other appropriate diagnostics are used continually to ensure that the quality of the GPE is acceptable.

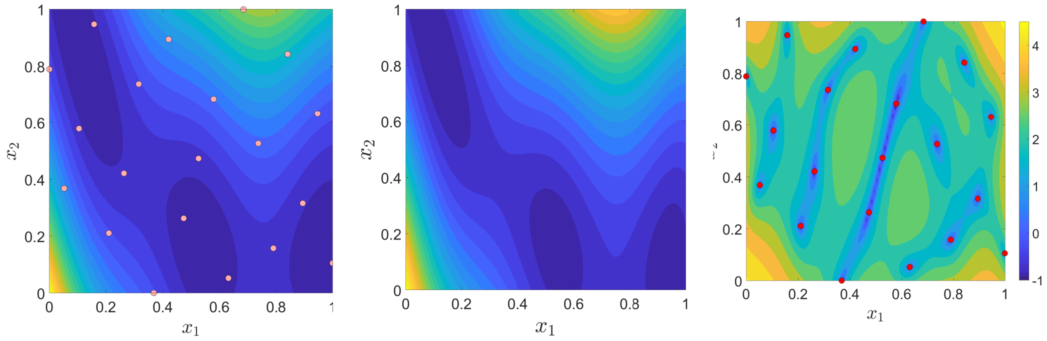
3.4 Benchmark studies

In this section, two examples are provided in order to test the GPE code developed for this research. Several implementations and toolboxes exist. However, a GPE code was written and tested in order to acquire a deeper understanding of the theoretical and practical challenges associated with emulation. MATLAB® was chosen as the programming environment for the Gaussian process emulator. This choice was made, because MATLAB® offers efficient implementation of linear algebra algorithms, through the use of the BLAS and LAPACK libraries.

To gain an insight in the way GP emulation works, a simple model, known as the *Branin function* is emulated first. It has the following analytical expression

$$\eta(x_1, x_2) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1-t)\cos(x_1) + s \tag{3.19}$$

where a, b, c, r, s and t are known constants. Even though the Branin function is not expensive to evaluate by any standard, it is a good representation of a typical non-linear engineering model. It is also two-dimensional, which allows visual inspection of the quality of the emulator to be carried out. Figure 3.9 depicts the results of an emulation. Figure 3.9(a) shows the level sets of the true function overlaid with the data points used to build the GPE. The sample size was $n = 20$, chosen according to the guidelines discussed in Section 3.3.2. The level sets of the posterior predictive mean of the GPE for Eq. 3.19 are shown in Figure 3.9(b). Visually there is very little difference between the two plots. The largest discrepancies between prediction and reality occur far away from training data and in particular, near the corners of the domain. One of the major advantages of using a GPE is the fact that it provides a closed-form expression



(a)

(b)

(c)

Figure 3.9: Emulation of the Branin function. (a) real function; (b) mean prediction; (c) predictive variance in $\log_{10}$ scale. Values scale from purple (low) to yellow (high).

of uncertainty associated with predictions. The posterior predictive variance for the Branin function is shown in Figure 3.9(c). It can be seen that the variance goes to zero in the vicinity of the training points, shown as red dots. This reflects the fact that the simulator is deterministic and there is no uncertainty about its output, once it has been observed. In order to examine the quality of the GPE beyond its visual appearance, more objective diagnostics are employed. After constructing the GPE, $m = 50$ points were obtained to validate it. The mean squared error based on this sample is $D^{RMSE} = 1.56$. This figure is useful when the performance of different metamodels is considered. The correlation between the responses of the function and the predictions from the emulator for the validation sample is shown in Figure 3.10(a). Points lying on the 45° dashed line indicate perfect correspondence between observations and predictions. The error bars correspond to the 95% credible interval for each site. As can be seen the majority of the validation data is close to the line. A few exceptions occur, but it should be observed that the predictive uncertainty for these captures the dashed line, potentially flagging problems associated with lack of data in those regions of the input domain. Another validation tool, the individual prediction errors, introduced in Section 3.3.5 is shown on Figure 3.10(b). Since the posterior

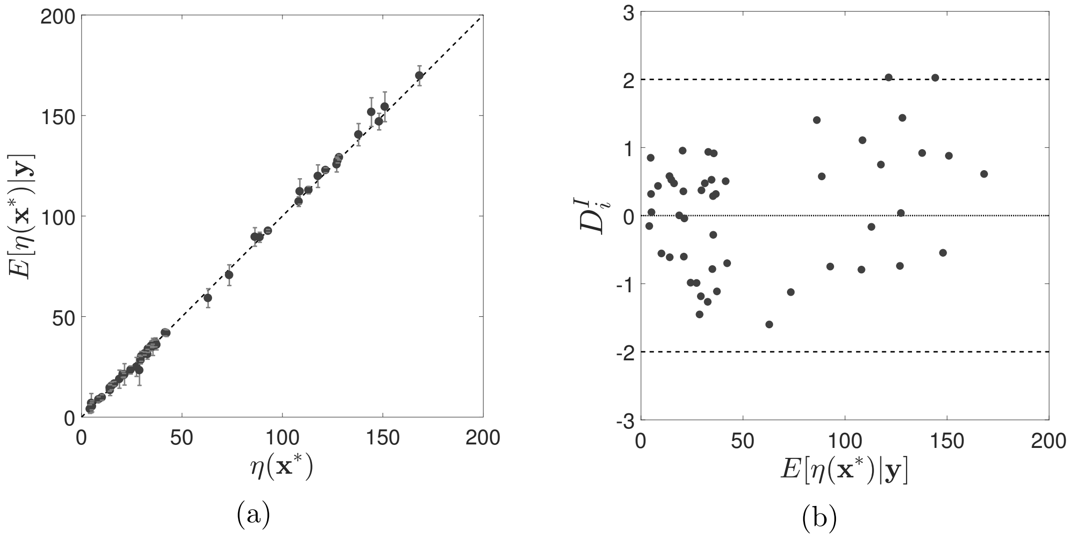


Figure 3.10: Diagnostics for the Branin functions emulator. (a) correlation between observations and predictions; (b) IPE.

predictive distribution of the emulator is Gaussian[4], D_i^I in Eq. 3.18 should have a standard normal distribution. Therefore the majority of the IPE should be distributed in the interval $[-2, 2]$ with no discernible patterns, such as in this case.

Most practical engineering models have input dimensionality higher than two. The next benchmark problem presents an empirical wing weight model for a light aircraft [121]. Such models are used in the conceptual design stage of aircraft and as the product gains an increasing amount of detail, more complex computational tools substitute the empirical approximations. Here the analytical expression is not important as the function is treated as black box. However, knowledge of design parameters incorporated in the model is useful. These variables are given in Table 3.2 and those having a geometric representation are shown in Figure 3.11.

The model was emulated using a constant trend and an exponentiated quadratic ARD kernel. The GPE was trained and validated on samples of $n = 100$ and

[4] Due to parameter estimation, it technically has a Student-t distribution with $n - q$ degrees of freedom. However, the size of the training sample, n is usually large enough to assume a Gaussian distribution.

Table 3.2: Design variables, corresponding ranges and GPE correlation lengths for the conceptual wing weight model.

Parameter	**Min**	**Max**	$\boldsymbol{\psi}$	**Remark**
tc	0.08	0.18	0.037	Airfoil thickness to chord ratio.
N_z	2.5	6	0.041	Ultimate load factor.
A	6	10	0.068	Aspect ratio, $A = b^2/S$.
S	150	200	0.081	Wing area (ft^2).
Λ	-10	10	0.085	Wing sweep at quarter chord (deg).
W_{dg}	1700	2500	0.106	Gross design weight (lb).
λ	0.5	1	0.244	Taper ratio, $\lambda = c_t/c_r$.
W_p	0.025	0.08	0.549	Weight of paint on the wing (lb).
W_{fw}	220	300	1.000	Weight of fuel in the wing (lb).
q	16	45	1.000	Dynamic pressure (lbf/ft^2).

$m = 100$ data points, respectively obtained from the model. Looking at the ranges for each variable, it can be seen that there is a relative difference of on the order of 10^5 between the largest and smallest ranges. When this is the case, the input space is usually normalized to the $[0,1]^d$ hypercube. This procedure stabilizes the search for optimal values of $\boldsymbol{\psi}$. This approach, combined with the use of ARD kernels, indicates the relative importance of each input variable to the variation in the output. In the case of the exponentiated quadratic kernel a smaller value of the correlation length corresponds to a more “active” input and vice-versa. The values of $\boldsymbol{\psi}$ for the wing model GPE are given in Table 3.2. This emulation by-product ranks airfoil thickness, load factor and aspect ratio as the three most important parameters in sizing the wing. An in-depth justification of this ranking is out of the scope of this work, but it is not too far from reality. Naturally, these results are only an indication of importance and should not be used to substitute full sensitivity analyses as will be presented in Chapter 6. As before, the correlation between observed data and predictions, together with IPE were used to assess the quality of the GPE. As seen in Figure 3.12, both diag-

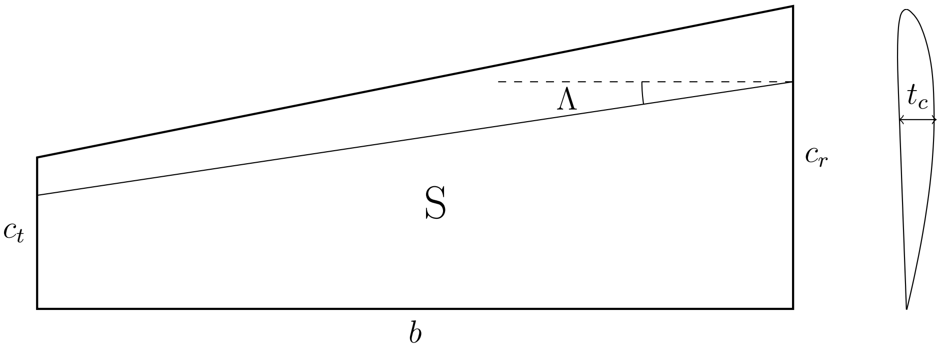
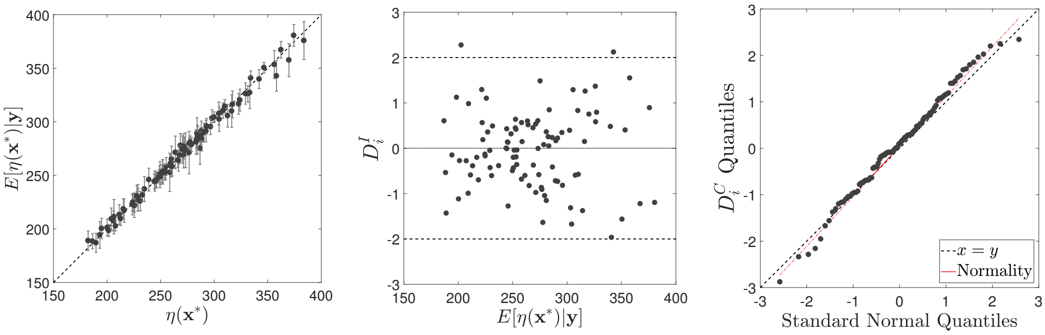


Figure 3.11: Wing weight problem nomenclature.

nostics suggest that the emulator is an accurate representation of the simulator. The RMSE for the model is $D^{RMSE} = 4.26$. In order to summarize the individual prediction errors, the authors of [120] use the *Mahalanobis distance* between the simulator output and the emulator mean

$$D^{MD} = (\mathbf{y}^* - \mathbb{E}[\eta(\mathbf{x}^*)|\mathbf{y}])^T \mathbb{V}[\eta(\mathbf{x}^*, \mathbf{x}'^*)|\mathbf{y}](\mathbf{y}^* - \mathbb{E}[\eta(\mathbf{x}^*)|\mathbf{y}]) \tag{3.20}$$

The Mahalanobis distance, D^{MD} is weighted by the posterior covariance matrix of the GPE, as shown in Eq. 3.20 to take into account the fact that predictions are correlated. The Mahalanobis distance should follow an F distribution with degrees of freedom $\nu_1 = m$ and $\nu_2 = n - q$. For the wing model GPE with the selected validation sample, $D^{MD} = 71.44$, which is an extreme value of the underlying probability density. This is a potential indication of correlation length estimation problem or violation of model assumptions. To investigate the matter further, it is useful to decompose D^{MD} in a set of uncorrelated errors to explore. Figure 3.12(c) shows a quantile-quantile (QQ) plot of uncorrelated errors obtained via a Cholesky decomposition of Eq. 3.20. As stated above the emulator structure imposes a standard normal distribution on the residuals and a QQ plot allows this assertion to be verified visually. In Figure 3.12(c) the red line indicates the normality of the data - residuals distributed normally, should exhibit no curvature



(a) (b) (c)

Figure 3.12: GPE diagnostics for the wing weight function. (a) correlation plot; (b) IPE; (c) UPE quantile-quantile plot.

with respect to the line. The black dashed line indicates perfect correlation between the GPE residuals and data coming from standard normal distribution. The QQ plot does not exhibit any obvious problems, even though the outliers on both ends might indicate some local emulator-simulator disparities. Further diagnostics can be performed as discussed in [120].

3.5 Chapter summary

This chapter introduced the concept of emulation. The theory behind Gaussian process emulators and the relevant notation were discussed. Using two illustrative examples, it was demonstrated how Gaussian process emulation can be used to address the computational cost of obtaining information from computer codes, by constructing a statistical approximation to their output.

The GPEs, introduced in this chapter, are a key ingredient in the uncertainty quantification analyses performed in this dissertation. Emulators for the lattice Boltzmann model are constructed in Chapter 5. They are then used in Chapter 6 to enable sensitivity analysis and again in Chapter 7 where a novel approach for reliability analysis is presented.

The next chapter introduces the development of LB codes for coalescence modelling. There, the need for a metamodelling approach to practical filtration studies becomes evident in a discussion about computational cost of LBM.

CHAPTER 4

Numerical modelling

4.1 Code development

The lattice Boltzmann model is a special discretization of the Boltzmann transport equation as discussed in Chapter 2, Section 2.2. This discretized version allows computers to solve the evolution equations forming the model. The next step in the development of the model is its implementation, the topic of the present chapter. As will be discussed below, the model needs to be built in a suitable environment to ensure its efficient operation. Even though Chapter 3 introduced a way of reducing the cost of computationally expensive models, the methods outlined there are aimed at dealing with the inherent cost of computer models and not with that incurred from inefficient development.

4.1.1 C Implementation

The importance of discussing matters related to implementation is warranted by the large amount of computational power accessible to users nowadays, which may lead to inefficient coding practices. Moreover, the increase in future clock speeds and core numbers is slowing down, due to various limitations [122], which emphasizes the need for lean coding.

Nowadays there is a large number of programming languages with different paradigms. These range from machine level languages (such as Assembly), low-level (C) through mid- (C++) and high-level languages (Ruby, Python) to environments built for specific purposes (MATLAB, R). High-level, interpreted (see below for definition) languages are usually easier to work with, but bring along significant efficiency penalties. On the other hand, machine level coding is often unnecessary and can have an impact on efficiency through its primitive constructs. Therefore, the language of choice for implementing LBM in this dissertation is C [123]. This language has a number of advantages including:

- *Low-level language* - C instructions are closely linked to the way the CPU works allowing it to be highly efficient for complex computations.

- *Compiled language* - programs written in C compile into an machine-readable executable file and are ready to run by the end user. This results in faster execution times compared to *interpreted* languages where commands are translated in runtime.

- *High portability* - C is widely used in operating system and kernel development and is thus native to most platforms. This ensures that a code written in C will be more accessible.

- *Structured implementation* - C is an imperative language which supports code structures making the code more readable and easier to debug.

- *Extensible capabilities* - There are many libraries available to C, which extend its low level capabilities.

Furthermore, LBM does not require complex language functionalities to run. In its core, the code consists of summations (taking moments) and thus C is an ideal language for its implementation. One of the useful extensions mentioned in the list above is the capability of C to use the OpenMP (Open Multi-Processing) interface, which enables parallel processing to be carried out by the program. OpenMP operates the *fork-join* model, whereby the main or *master* thread (process) divides into two or more branches that are shipped for execution to multiple CPUs and the output of each is then collected back in the main thread, joining the results of the computation. Using multi-threading, the performance of the code usually increases. There are applications in which the overhead from managing data transfer between threads outweighs the gains of having multiple processors. Therefore, judgement has to be exercised about where and how to parallelize the code. In the case of LBM, calculations are done independently on each lattice node and data from neighbouring nodes is needed only in computing the interaction forces as shown in Section 2.3. In an extreme case of parallelisation the graphical processing unit (GPU) of the computer is used because it has a large number of small processors [124].

The results of a small benchmark experiment are given in Table 4.1. It must be noted that this benchmark should be viewed vertically, that is one should not compare the performance of the two processors, but rather its variation with the change in the number of cores. The Intel® i5 is a desktop CPU with a much smaller bandwidth and transfer rates than the server-optimised Intel® Xeon.

The model is coded in a way that allows the user to specify simulation parameters via text files. All constants and values related to the mathematical discretization of the code (lattice velocities, weights, etc.) are not immediately accessible from outside the source code, but can be altered should the need arise.

In practice the source code uses models that work for most applications and thus there should be no need for them to be changed or supplemented.

The time durations in Table 4.1 may not seem long in absolute sense, but when scaled to represent domains of practical interest and durations of a few seconds in real time, they can span days. For example, a 2D MCMP simulation in a domain of size 500 × 700 lattice nodes, representing bed thickness of 0.55 mm and set to run for 10×10^6 time steps (ts) takes approximately 52 hours on the aforementioned Intel® Xeon. This simulation represents just over 16 seconds of real time. More details on this and other examples will be given in Chapter 5. On the other hand, even if the simulations of interest had durations similar to those in Table 4.1, a sensitivity analysis like the one that will be presented in Chapter 6, requires on the order of 10 000 code evaluations per input variable. With just three input variables it would take nearly 19 hours to complete the study, with the resources at hand. Therefore the need for a Gaussian process emulation, described in Chapter 3 becomes evident.

Table 4.1: Run times of an SCMP LBM code with 200 × 200 nodes for 1000 time steps.

	Time	
Number of Cores	**Intel® i5-4570**	**Intel® Xeon E5-2670**
1	96.731	32.142
2	82.613	17.122
4	77.747	8.608
8	-	4.639
16	-	2.520
20	-	2.236

4.1.2 Code capabilities

The simulator has to have certain capabilities in order to be useful for the present research. These are not *model features*, that is features intrinsic to the modelling framework (for a list of these refer to Chapter 2, Section 2.4), but consist of functionalities that enable the analysis of filter beds. The first one is the ability of the model to simulate arbitrary values of water concentration. Since this research is about filtering *free* water from the fuel, concentration is taken as the volumetric ratio of water droplets to bulk fuel. In theory variable concentrations can be achieved by initializing the simulations with a large upstream domain which is seeded with water droplets to match the desired concentration. This approach however scales poorly as it adds unnecessary elements to the computational domain and slows down simulations. Instead, in practice water droplets are dynamically initialized in the domain immediately in front of the filter bed. The interval of initialization provides the means of changing the concentration of water in the bulk medium. For stability reasons the droplets seeded during runtime have wider diffuse interface which normalizes in a few time steps.

To increase the code utilisation this research also explored the idea of simulations that can be restarted. The idea is to enable the analyst performing the simulation to potentially reuse results from stable runs. Suppose, for instance, a particular simulation that was set up to run for $t_{run} = 50000$ time steps in the hope that water droplets will be observed at the outlet of the filter. Now suppose, that in the post processing visualisation the analyst discovers that the droplets had not reached the outlet and another $t_{add} \approx 20000$ ts will be required to achieve this. Instead of running $t = t_{run} + t_{add} \approx 70000$ts the investigator can only run the code for $t = t_{add}$. This is achieved by storing the density and velocity fields at each site of the lattice at regular intervals of predefined number of steps. In order to prevent this procedure from loading the memory excessively the number of steps for writing information should be set as high as practically possible. It should be noted that this *backup* method is not recommended as a short cut for restarting divergent simulations.

Finally, in order to visualise the results of any given simulation the code outputs a collection of .vti files which can be used and analysed with open source software such as ParaView [125]. The values of any field can be written to the output at any time rate. It is worth noting however that writing to files is one of the major bottlenecks in the code execution and in storage memory. Therefore the time between frames should be appropriately chosen for the length of the simulation.

4.2 Introducing units

4.2.1 Dimensionless numbers

As any other computer simulation tool lattice Boltzmann models work in a non dimensional regime. Computers only understand numbers and it is therefore part of post-processing to assign correct units to the variables of interest. Analysing and working with units is part of the domain of *dimensional analysis*, which is quite extensive and has its own formalisms. In this work, that part of dimensional analysis is used which states that equations must have matching units and that certain conditions must be satisfied for correct simulations. When simulating flows one has to ensure that certain *non-dimensional* numbers or groups match. These are most often the Reynolds number and the Mach number, given in Eq. 4.1 and Eq. 4.2 respectively.

$$Ma = \frac{|\mathbf{u}|}{a} \tag{4.1}$$

$$Re = \frac{|\mathbf{u}|d}{\nu} \tag{4.2}$$

where $|\mathbf{u}|$ is the flow speed, a is the speed of sound, d is a characteristic length and ν is kinematic viscosity. When simulating incompressible flow (generally accepted as flows having $Ma < 0.3$) the Mach number is not of immediate importance. Moreover, the flow of liquids is indeed very nearly incompressible. Thus, even though no attempts will be made to match it, the Mach number in LBM will be used as an indicator for potential stability issues as outlined in Chapter 2.

The situation with Re is different. Eq. 4.2 gives the general form of the Reynolds number. However when analysing *multiphase* flow through *porous media*, some of the terms change their physical meaning. This has led different authors to define different expressions for the Reynolds number. Table 4.2 summarizes some popular suggestions. The authors of [126] present a collection of expressions typically used to calculate Re in porous media. Another expression found in [127] is the one denoted Re_6. It is also important to recognize that the flow is not uniform and this may have an effect on its behaviour. An expression for Reynolds number in multiphase flow is also given in Table 4.2. It becomes obvious that capturing the important parameters in this scenario is non-trivial. A real filtration example is given in [22].

Table 4.2: Comparison between different Re expressions. Values for dependent variables are approximately based on Parker products.

Porous media	**Values**	**Multiphase flow**	**Values**
$Re_1 = \frac{u_0 D_p}{\nu}$	1.45×10^{-2}	$Re_m = \frac{\lvert u_0 - u_d \rvert d_d}{\nu}$	1.75×10^{-3}
$Re_2 = \frac{u D_p}{\nu} \frac{1}{1-\phi}$	1.73×10^{-1}		
$Re_3 = \frac{u d_t}{\nu}$	3.75×10^{-2}		
$Re_4 = \frac{u_0 r_t}{\nu}$	8.75×10^{-3}		
$Re_5 = \frac{u_0 k \beta}{\nu}$	No reliable data for β		
$Re_6 = \frac{u_0}{(1-\phi) S_v \nu}$	3.78×10^{-1}		

In Table 4.2 u_0 is the superficial or face velocity, defined as $u_0 = \dot{Q}/A$, where $\dot{Q}$ is the volumetric flow rate and A is the cross-sectional area of the filter bed. The value of u_0 different from u which is the *interstitial* velocity in the filter. Furthermore, D_p is a representative diameter typically taken as some percentile of the fibre diameter distribution obtained experimentally; ϕ is the filter bed porosity; r_t and d_t are pore throat radius and diameter, respectively, again taken from an empirical distribution. In Re_4, k is the air permeability of the bed and β is called the *non-Darcy coefficient*[1]. In Re_6, S_v is the fibre surface area per unit volume. In the multiphase Reynolds numbers u_d is the velocity of the droplet and d_d is its diameter. There are other expressions for multiphase Re which use a combination of dependent variable values for the components, weighted by their volume fractions, instead of a single value as in Eq. 4.2. They are not discussed here. Values for many of the above variables are either empirically estimated or have to be obtained experimentally. This leads to higher degree of uncertainty in calculating Re. However all expressions lie well below 2, which according to [127] is the criteria for Darcy flow and the development of turbulence. Furthermore, according to [128], for $Re_m << 1$ droplets and bubbles remain spherical and the flow is classified as Stokes flow.

4.2.2 Dimensional analysis for LBM

Starting with velocity, there are a number of locations where it can be measured, as mentioned in Section 4.2.1. The most reliable value to be used is the *face velocity* u_0, as it comes directly from flow rate which is easily measured. For a typical, flat (*i.e.* non-pleated) filter the face velocity ranges from 3 mm/s to 13 mm/s. The kinematic viscosity of a mixture, ν_m is a complicated function of the viscosities of the components and can only be estimated [129, 130]. However, typical concentrations of water in the fuel are around ζ_w =1500 ppm to ζ_w =20 000 ppm, which results in $\nu_m \approx \nu_{biodiesel} \approx 4\,\mathrm{mm^2/s}$ and $\nu_m \approx \nu_{ULSD} \approx 2\,\mathrm{mm^2/s}$. This value varies considerably with fuel composition and temperature, but it is representative of typical ULSD/biodieselin normal operating conditions. Finally, a characteristic length is taken to be the median size of fibre in the coalescer bed, d_f. This value, again, varies substantially with position and distribution as will be explained in Chapter 5, Section 5.2 and in Chapter 8, but for initial calculations

[1] High speed flows in porous media are referred to as non-Darcy flows

a figure obtained from analysis of SEM images is used as $d_f \approx 2\,\mu\text{m}$ to $20\,\mu\text{m}$. [2] Using Eq. 4.2, this gives a reference value of $Re \approx [2.00\times10^{-3}-7.00\times10^{-2}]$, which fits well with the values in Table 4.2. Due to its simplicity and reliability this Reynolds number will be used in the following analysis. The procedure starts by selecting three independent, *primary conversion* variables. Here, *length*, *density* and *time* are chosen. The physical values for the three are the density of water, ρ_w, the fibre diameter, d_f and relaxation time, τ. In coalescence filtration processes the bed depth is relatively large compared to the fibre diameter. Nevertheless, resolution high enough to capture separate fibres is required for simulation, which means that the number of lattice nodes across fibres has to be as low as practically possible to keep the simulation cost under control. This is why, a representative value for the fibre diameter[3] $d_f = 8\,\mu\text{m}$ is discretized with 4 lattice units. The fibre diameter for the LBM simulation thus becomes, $\hat{d}_f = 4$ lu. Hereinafter, the symbol $\hat{}$ will be used to denote non-dimensional properties in the model. It must be noted here that the choice of fibre size completely arbitrary so long as it does not violate any physical principles and does not fail in simulating the flow of interest. The ratio between physical and LBM units is called a *conversion factor* and it has the units of the physical property. The conversion factor for length is thus, $C_L = 2 \times 10^{-6}\text{m}$.

Density is chosen as the second independent variable, because it is convenient to work with from a dimensional point of view and because there is certain degree of freedom in choosing its LBM value. From the discussion in Section 2.3.1 it becomes clear that this choice is based on practical simulation considerations. In the same section, the density of the water component is chosen as, $\hat{\rho}_w = 0.999$. The corresponding conversion factor is $C_\rho = 1001\ \text{kg/m}^3$. To calculate the conversion factor for time, the procedure is carried out in reverse *i.e.* from LBM to physical units. This is due to the sensitivity of the simulation to the relaxation

[2] All values obtained from Parker Hannifin laboratory work.
[3] Private correspondence with Parker Hannifin.

time. As outlined in Section 2.3.1 this parameter is set to, $\tau_w = \tau_{bd} = 1$ for both components.

A final set of ingredients that are not immediately obvious, but have to be specified nonetheless are the time and space discretization steps. Since LBM works on a regular lattice and in equal time steps, it is useful to take $\Delta\hat{x} = \Delta\hat{t} = 1$. Then, their physical values will be $\Delta x = C_L$ and $\Delta t = C_t$. By definition the Reynolds numbers between reality and simulation have to match, that is $Re = \hat{Re}$. From Section 2.2, the viscosity in the simulation is given by

$$\hat{\nu} = \hat{c}_s^2(2\tau - 1) \tag{4.3}$$

Then comparing units with physical viscosity and rearranging

$$\begin{aligned} \nu &= \hat{\nu}\frac{C_L^2}{C_t} && (4.4) \\ C_t &= \frac{\hat{\nu}}{\nu}C_L^2 && (4.5) \\ C_t &= \frac{\hat{c}_s^2(2\tau - 1)}{\nu}C_L^2 \end{aligned}$$

Once all primary conversion factors are calculated, the values of the secondary factors can be uniquely defined. A summary of the results of the analysis is given in Table 4.3

A few things must be mentioned about Table 4.3. Firstly, it should only be seen as a guide. The conversion factor definitions will change if the user chooses a different set of primary variables. What is more, sometimes there are more than one way to construct the conversion factors using the same primary variables. The initial values can also be changed under some general rules. The meaning of the values in columns **LBM theory** and **LBM practice** is important and they should be analysed together. The latter column contains values for different variables which are used in practical LBM due to various restrictions. It can be

Table 4.3: Conversion factors and sample values. The table includes optimal and corrected values for some variables. Not all properties are used in the simulations.

Property	**Value**	**Unit**	**Factor**	**LBM theory**	**LBM practice**
d_f	8.00×10^{-6}	m	C_L	2.33×10^{-1}	4.00×10^{0}
u_0	7.00×10^{-3}	m/s	$\frac{C_L}{C_t}$	2.92×10^{-4}	1.00×10^{-2}
τ_w	–	-	–	17.64×10^{0}	1.00×10^{0}
τ_{bd}	–	-	–	17.64×10^{0}	1.00×10^{0}
ζ_w	1.60×10^{3}	ppm	–	–	–
ρ_w	1.00×10^{3}	kg/m^3	C_ρ	–	9.99×10^{-1}
ρ_{bd}	0.84×10^{3}	kg/m^3	–	–	1.00×10^{-3}
ρ_m	0.84×10^{3}	kg/m^3	–	–	–
ν_w	1.00×10^{-6}	m^2/s	C_ν	5.71×10^{0}	1.67×10^{-1}
ν_{bd}	4.00×10^{-6}	m^2/s	–	5.71×10^{0}	1.67×10^{-1}
ν_m	3.99×10^{-6}	m^2/s	–	–	–
t	1.00×10^{0}	s	$\frac{C_L^2}{C_\nu}$	2.40×10^{7}	–
g	9.81×10^{0}	m/s^2	–	1.70×10^{-8}	–
f	9.81×10^{3}	N/m^3	$\frac{C_\rho C_L}{C_t^2}$	1.70×10^{-8}	–
σ	1.30×10^{-3}	N/m	$\frac{C_\rho C_L^3}{C_t^2}$	1.81×10^{-1}	–
p	1.93×10^{5}	Pa	$\frac{C_\rho C_\nu}{C_t}$	3.34×10^{-1}	–
Re	1.40×10^{-2}	–	–	1.40×10^{-2}	4.80×10^{-1}
We	3.94×10^{-6}	–	–	5.52×10^{-6}	–

seen that there is no exact match between Reynolds numbers in simulation and reality. As illustrated previously the physical values of the Reynolds number are unclear too and vary noticeably. However, both simulated and real numbers indicate that the flow is in a turbulence-free regime where droplets remain spherical. If in theory, one requires an exact match between Re and $\hat{Re}$ the values in **LBM theory** provide a means for that. It should be read as follows. If the characteristic dimension, $\hat{d}_f$ and the relaxation time, τ are to be kept fixed at their practical values, then the face velocity needed for matching Reynolds numbers can be taken from **LBM theory**. This is also valid for any of the other Reynolds-defining properties. One strong motivation to disregard perfect correspondence between Reynolds numbers is simulation time. Reading from **LBM theory**, in order to match Re simulations must be set up such that 1 second of physical time is represented by 24 million time steps. For simulations with realistically sized domains this is infeasible in both time and memory requirements. It should be said here that the properties in Table 4.3 are strongly interlinked and changing one value is likely to influence multiple others. For this reason, the code features a dimensional analysis facility to aid the user in choosing parameter values. Additionally, since there are repeating variables, with distinct values for each component (*e.g.* density, viscosity), engineering judgement must be applied as to which component value should be used in the calculation of conversion factors. For example, it is of interest to see enlarged *water* droplets exiting the coalescer fall under gravity. This is why the density of water is used when working out the force density in the simulation. Secondly, the cells filled with a hyphen indicate either a repetition or values that do not have direct correspondence between the physical system and the model.

Finally, the last row of the table is dedicated to the Weber number, given by

$$We = \frac{\rho u^2 d}{\sigma} \tag{4.6}$$

characterising the ratio between inertial and surface tension forces in multiphase flows. The difference between the two values is insignificant and can be controlled by τ_c and $G_{c\bar{c}}$. The same issues described for Re concern We in the current research.

4.3 Detection of water droplets

Computer simulations can be run one-at-a-time and the resulting property fields can be visualized in a post-processor for inspection by the analyst. However, when the design of the filter is considered, there is a need for running series of computational experiments and using the results to perform various analyses. In this case instead of analysing the whole flow field, some information is extracted as an output in the form of predefined figures of merit. These could be velocity, flow rate and pressure at a given position. Or as it is the case in the current research, size distribution of droplets. There is no immediate need to visualize the results of the whole simulation[4] and inspect them by hand. All that is needed in the end of each run is a predefined characteristic. If the quantity of interest is droplet size distribution in the outlet of the filter, a reliable method for determining the time when a droplet leaves the mesh is needed. As mentioned above this process can be performed manually, but at a considerable post-processing cost. At the same time simulations are typically time consuming to run and their time cycle does not necessarily match that of the analyst. Therefore an automatic droplet/fibre separation and characterization algorithm is required.

This can be achieved with the use of a carefully positioned *barrier*, which acts as a trigger switch for the algorithm calculating the volumes of droplets and inferring their size. An illustration of this idea is shown in Figure 4.1.

The barrier algorithm works by checking the presence of water density compo-

[4] The output data from realistically sized simulations representing physical time of a few seconds, can be as large as several hundred gigabytes.

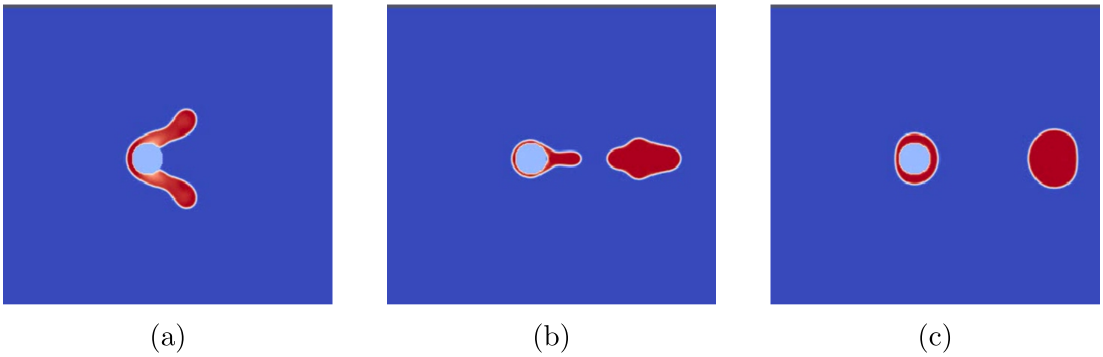


Figure 4.1: A droplet separating from a fibre. The barrier algorithm is able to correctly detect separations with simple dynamics.

nent across the barrier at regular interval of time steps. Once a detection of water component has occurred (Figure 4.1(a)), the program calculates the volume of water across the barrier. It then compares this with the volume calculated in the previous step and if there is a decrease in the value, the algorithm interrupts the simulation. Assuming that the barrier is placed at the correct distance behind the mesh, a reduction in the volume of water component would indicate the retraction of the water finger Figure 4.1(b) that remained attached to the fibre, suggesting separation (Figure 4.1(c)). There are several issues that arise from the use of this basic algorithm. Firstly, the routine results in the calculation of volume instead of size. These two quantities can only be used interchangeably if there is one droplet escaping the mesh at a time, which is highly unlikely. Secondly, the volume calculated by the algorithm is very sensitive to the placement of the barrier. If for instance it is placed too close to the mesh, the water finger still attached to the fibre might not retract fully resulting in its inclusion in the escape volume. Finally, there is no guarantee that a reduction in volume means that separation has occurred.

A possible improvement is to employ two barriers - a stationary one and an active one. The static barrier is still in the role of a switch determining when to start the volume calculation. The active barrier moves with the water component until it senses a reduction in volume between the two barriers. At this point the algorithm calculates the total volume behind the active line. This approach only partially addresses the issue arising from barrier placement. Furthermore, as outlined in Section 2.3.3 the domain is often truncated to save on computational time and there is usually limited space behind the mesh for the drop to reach steady state, which may inhibit the correct operation of the algorithm. Further issues begin to appear when there is a multitude of droplets.

In order to not only have a robust separation identification, but also to decouple the volume and size, clustering is used. This enables not only the reliable distinction between drop and fibre (*i.e.* separation), but also provides information about separate droplets upon request. Clustering [103] is a technique frequently used in the fields of machine learning and data mining to learn about objects and their features, and how they fall into groups (clusters). There is a large number of clustering algorithms, which have different advantages. To be considered a candidate for meeting the current objective, the algorithm has to be able to *discover* an *unknown* number of clusters. This criterion rules out the very widely used *k-means* method [131]. In this research, an algorithm for *Density-Based Spatial Clustering for Applications with Noise* (DBSCAN) [132] is used. DBSCAN allows the user to reliably discover clusters by specifying the minimum number of *neighbours* each data point should have along with the size of the *neighbourhood* in order to be considered a cluster. It has been used for clustering low-dimensional datasets [133] with very high accuracy. For the current application the clusters of interest are the droplets flowing out of the filter. Despite the fact that it might not be immediately obvious, droplets in LBM are *clusters* of lattice nodes occupied by the water component. These nodes have well-defined coordinates, occupying relatively small number of dimensions (at most 3) and have a precisely defined number of neighbours, and neighbourhood sizes. The meaning of all these terms in the LBM case is illustrated in Figure 4.2. Since LBM uses a regular lattice, the

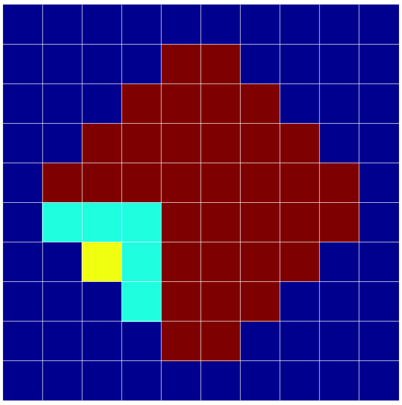


Figure 4.2: DBSCAN concepts applied to a water droplet in LBM (red). For each node (yellow) of a valid cluster, the neighbourhood (cyan) contains at least three other nodes.

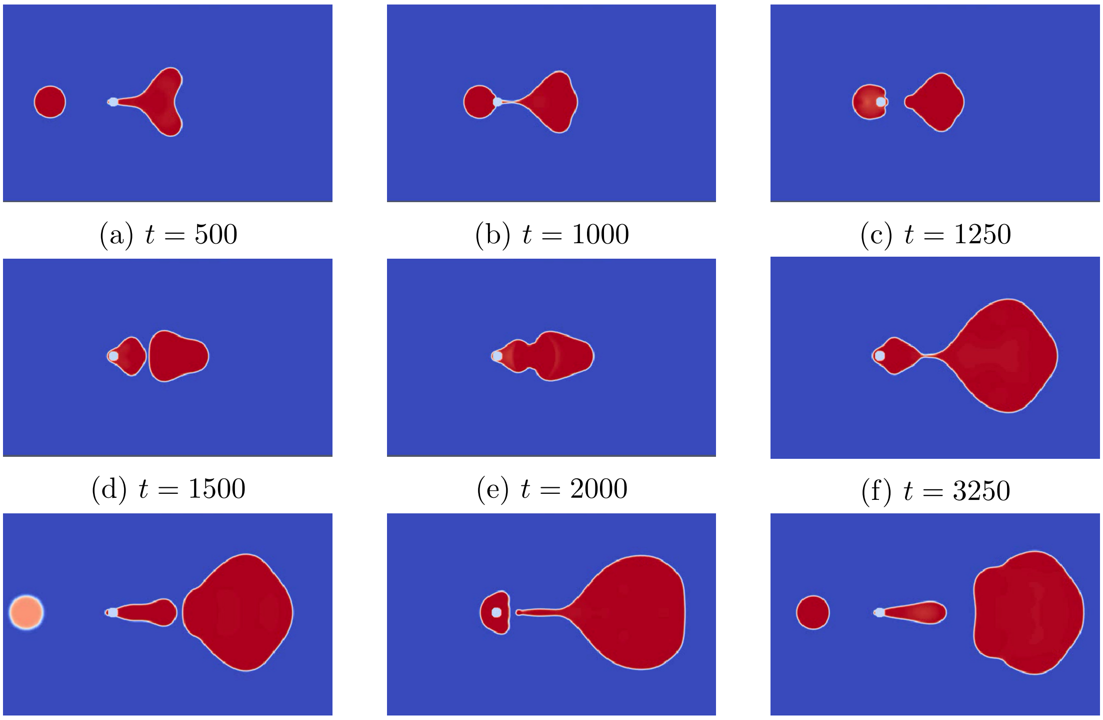
neighbourhood of each node has radius $r_n = 1$ and consists of exactly $n_n = 3^d - 1$ neighbours, where d is the number of dimensions in the simulation. In fact this relationship provides the second parameter required by DBSCAN. Refer again to Figure 4.2. In order for a cluster of water nodes to be classed as a droplet, it needs to be approximately spherical as this is the shape of the physical droplets. This means that at any resolution, the smallest droplet *must* consist of *at least* 4 water nodes arranged in a 2×2 grid[5]. Consequently, this limits the minimum number of neighbours to 3. If the user wishes to specify different minimum number of neighbours, $n_{\min(n)}$ for example to impose a minimum droplet size, the neighbourhood will have

$$n_n = n^d_{\min(n)} - 1 \tag{4.7}$$

For the current application $n_{\min(n)} = 3$ resulting in maximum resolution. The details of the detection are outlined in Algorithm 1. The nomenclature of symbols, not defined previously is as follows. The barrier at location x_b is a line or a plane (depending on the simulation dimensionality) spanning the domain in the other directions and is used to initialize DBSCAN upon detection of the water

[5] It could technically consist of one node, but such a droplet will evaporate quickly.

component. The distance counter, n_d and its limit value, $\bar{n}_d$ store a number of distances up to a maximum, respectively, between the centroid of each cluster $\mathbf{c}_k$ and the centroid of the fibres c_F. If the distances, d_{k_t} are increasing with each call to DBSCAN, then the droplet in the given cluster is departing the mesh and can be considered separated. For brevity, the full notation d_{k_t} with explicit dependence on time has been shortened to d_k where this does not cause confusion. The arrays $\mathbf{F}$ and $\mathbf{D}$ hold the locations of lattice nodes constituting the fibres and droplets, respectively. Similarly, $\mathbf{C_F}$ and $\mathbf{C_D}$ are tall arrays containing the cluster indices of each (x, y) tuple in the subscripted arrays. Also, in Algorithm 1, $\mathbb{I}(\cdot)$ is the indicator function which returns 1 whenever the logical statement in its argument is true and 0 otherwise. The switch, s_d is used to determine whether DBSCAN should be started or another pass of the barrier should be performed. This procedure can be used to detect separation and end the simulation (as in Algorithm 1), or to take "snapshots" of the outlet of the filter and calculate droplet size distributions. Only a minor modification is required to the pseudo code, where instead of returning from the computation the routine continuously outputs droplet volume-based diameters. The value of $\bar{n}_d$ can be decreased for the second application, mainly to speed up the computations. A visual demonstration of the detection method at work is shown in Figure 4.3 and Figure 4.4. In the simulation that produced these particular results, a single fibre was challenged with dynamically generated droplets and the simulation was run until separation. More details about the set-up will be provided in Section 5.3. The frames in Figure 4.3 illustrate a scenario in which the droplet separates initially, but due to its enlarged volume, it goes back and reattaches to the fibre. An algorithm that is incapable of differentiating between the droplet and the fibre will be defeated into detecting the false-positive separation. As outlined in Algorithm 1 the procedure tracks the centroid of the clusters and continually calculates its distance from the fibres. Figure 4.4 shows the variation in the distance with each call to the



(a) $t = 500$

(b) $t = 1000$

(c) $t = 1250$

(d) $t = 1500$

(e) $t = 2000$

(f) $t = 3250$

(g) $t = 3500$

(h) $t = 3750$

(i) $t = 5250$

Figure 4.3: False separation of a droplet. The algorithm correctly identifies several false positive separations at and neglects them, leaving the simulation running until true separation is achieved at $t = 5250$.

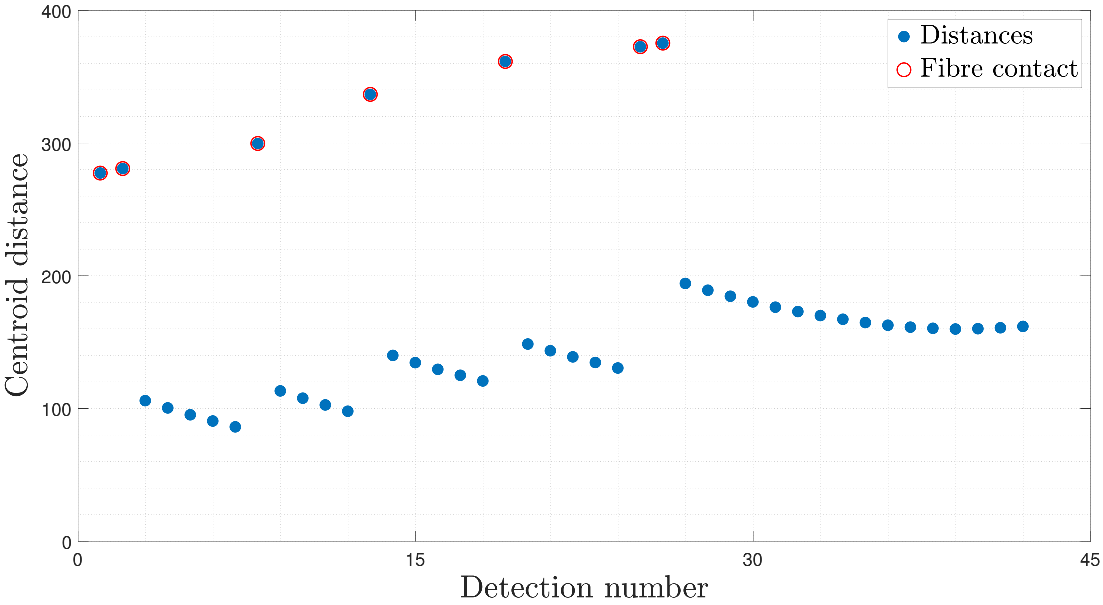


Figure 4.4: Variation of distance between fibre and drop with time. The droplet is considered separated when the distance has been increasing for the last $\bar{n}_d$ calls to the algorithm

algorithm, illustrating the distance rule.

4.4 Chapter summary

In this chapter important aspects of the development of the computational tool used to simulate coalescence processes were outlined. The significance of working with a suitable software as well as hardware was briefly discussed and the choice of C as a primary development language was justified. Some of the built-in capabilities of the code were introduced and will be discussed further in the next chapter. This chapter also explored the introduction of physical units in LBM simulations and some of the challenges of the process when multiphase flow through complex media is concerned. A contribution made in the current chapter is, among others, the introduction of a droplet detection algorithm, based on the principles of spatial clustering. The next chapter introduces LBM experiments, developed on the basis of concepts formed in the current discussion.

Algorithm 1 An illustration of the DBSCAN-based detection for 2D LBM

```
Require: An LBM simulation set-up, barrier switch position $x_b$, distance counter
    limit, $\bar{n}_d$.
Extract horizontal, $x$ and vertical, $y$ coordinates of solid lattice nodes.
$\mathbf{F} = \{(x_1 \ldots x_n)^T, (y_1 \ldots y_n)^T\}$ s.t. $\rho_{x_i,y_i} = \rho_s \ \forall i \in 1 \ldots n$.
$\mathbf{C}_{F_i} = 0 \ \forall i \in 1 \ldots n$.
Calculate centroid of $F$, $c_F = \left[x_{c_F}, y_{c_F}\right]$
Assign step size $t_{DBSCAN}$ for clustering.
$s_d = 0$.
for $t < t_{max}$ do
    Perform LBM calculations.
    if $t$ mod $t_{DBSCAN} = 0$ then
        if $s_d = 1$ then
            $\mathbf{D} = \{(x_{d_1} \ldots x_{d_m})^T, (y_{d_1} \ldots y_{d_m})^T\}$ s.t. $\rho_{x_{d_i},y_{d_i}} = \rho_w \forall i \in 1 \ldots m$.
            Divide $\mathbf{D}$ in $k$ clusters and assign indices to $\mathbf{C_D}$.
            for all $k \in \mathbf{C_D}$ do
                $c_k = \left[x_{c_k}, y_{c_k}\right]$
                $V_k = \sum_{xy} \mathbb{I}(\rho_{x_{d_i},y_{d_i}} \geq \rho_w)$ of $C_{\mathbf{D}_k}$
                $d_k = \sqrt{(c_k - c_F)^2}$
                if $d_{k_t} > d_{k_{t-1}}$ then
                    $n_d = n_d + 1$
                end if
            end for
            if $n_d > \bar{n}_d$ then
                return
            end if
        else
            $s_d = \left(\sum_y \rho_{x_b,y} > 0\right)$
        end if
    end if
end for
```

CHAPTER 5

Lattice Boltzmann experiments

After introducing the theory behind lattice Boltzmann modelling in Chapter 2 and matters related to its computational implementation and capabilities in Chapter 4, this chapter provides a discussion of how the model was used in numerical experiments. Important aspects of setting up LBM simulations, as well as a collection of preliminary studies are considered. The chapter presents the hierarchical modelling approach adopted in this dissertation, going from small-scale, two-dimensional studies to three-dimensional experiments. Later on in the chapter, Gaussian process emulation, introduced in Chapter 3, is used with LBM to reduce its cost and enable further analyses.

5.1 Interfacial tension experiments

Special attention is given to interfacial tension due to its importance in emulsion filtration. According to Laplace's law, the difference between the outside and the inside pressure of a drop or bubble is given by $\Delta p = 2\frac{\sigma}{r}$. That is, the difference is proportional to the interfacial tension and inversely proportional to the radius (*i.e.* proportional to curvature). This relationship suggests the slope in a Δp versus r^{-1} plot gives the interfacial tension.

For the MCMP model, outlined previously in Chapter 2, the pressure is related to the density via the EOS in Eq. 2.28 and repeated here for convenience

$$\hat{p} = \hat{c}_s^2 \sum_c \hat{\rho}_c + \frac{\hat{c}_s^2}{2} \sum_{c \neq \bar{c}} G_{c\bar{c}} \psi_c \psi_{\bar{c}} \tag{2.28}$$

Formally, $\hat{p}$ is a function of the lattice speed of sound, $\hat{c}_s$, the density of each component, $\hat{\rho}_c$, the interaction strength parameter, $G_{c\bar{c}}$ and the interaction potential for each component, ψ_c *i.e.* $\hat{p} = \hat{p}(\hat{c}_s, \hat{\rho}_c, G_{c\bar{c}}, \psi_c)$. However, it is typical to use the relation $\psi_c = \hat{\rho}_c$ and the values of $\hat{\rho}_c$ are somewhat arbitrary and dictated by simulation stability. The speed of sound, $\hat{c}_s$ is also fixed for a given lattice. Thus, given the choice of model and some general functional relationships, $\hat{P}$ can be seen as $\hat{P} = \hat{P}(G_{c\bar{c}})$. In the MCMP model several experiments were run to establish the connection between the parameter $G_{c\bar{c}}$ and interfacial tension. These consisted of simulating stationary circular droplets of a high density component suspended into a low density one. It has to be noted that $G_{c\bar{c}}$ is directly related to the stability of simulations, such that too low of a value leads the droplet dissolving in the surrounding medium. On the other hand if the value is too high, the simulation becomes unstable and diverges from equilibrium. $G_{c\bar{c}}$ also has an effect on the width of the interface between components with small $G_{c\bar{c}}$ resulting in a wide interface and vice versa. Practically, one would like to have as sharp an interface as possible.

The experiments were run for $G_{c\bar{c}} = \{2.0, 2.5, 3.0, 3.5\}$ and $\hat{r}_d = \{3, 5, 8, 10, 12, 15, 20, 30, 40, 50\}$ lu to cover the whole range of relevant drop sizes. The size of the domain was set to 200×200 lu to make sure the droplets are

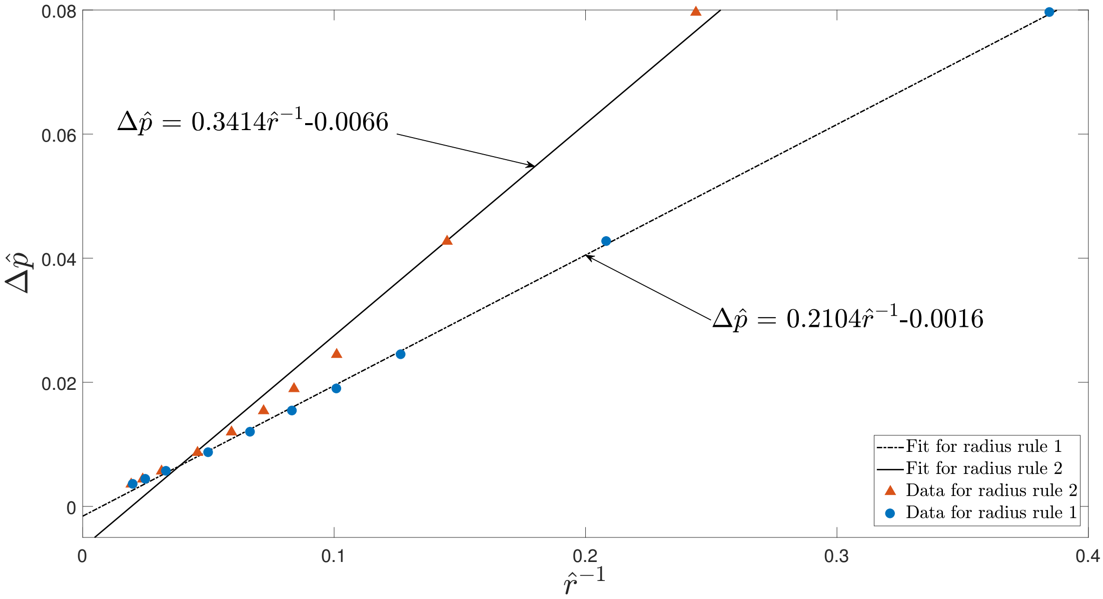


Figure 5.1: A typical Laplace plot used to determine the interfacial tension between components.

sufficiently far away from the boundaries. It was found the values of $G_{c\bar{c}} < 2.0$ lead to a fast droplet dissolution, whereas $G_{c\bar{c}} > 4.0$ leads to simulation instabilities. A typical Laplace plot is shown in Figure 5.1. There, $G_{c\bar{c}} = 3.5$ resulting in a "line of best fit" with a slope $\hat{\sigma} = 0.2105$. There are several parameters that go into the plot and whose values must be obtained. First, the radius of the droplet needs to be measured. Looking at a plot of the densities of both components along a line spanning the domain and going through the middle of the droplet, as shown in Figure 5.2, it is apparent that there are no clear boundaries for the droplet. This is due to the presence of a finite width diffusive interface. Determining the radius can be done in a few ways outlined in [78]. The first approach is to identify the point at which the density values of the two components cross. In Figure 5.2 this point is denoted with a circle. Another, more conservative approach is to measure the density of the drop in the surrounding fluid (dashed line in Figure 5.2) and identify the region over which the droplet component density (solid line) is larger. The span of the droplet according to this rule is indicated with squares.The first rule is used in the remainder of this work, because it gives line intercepts closer to 0, as seen in Figure 5.1.

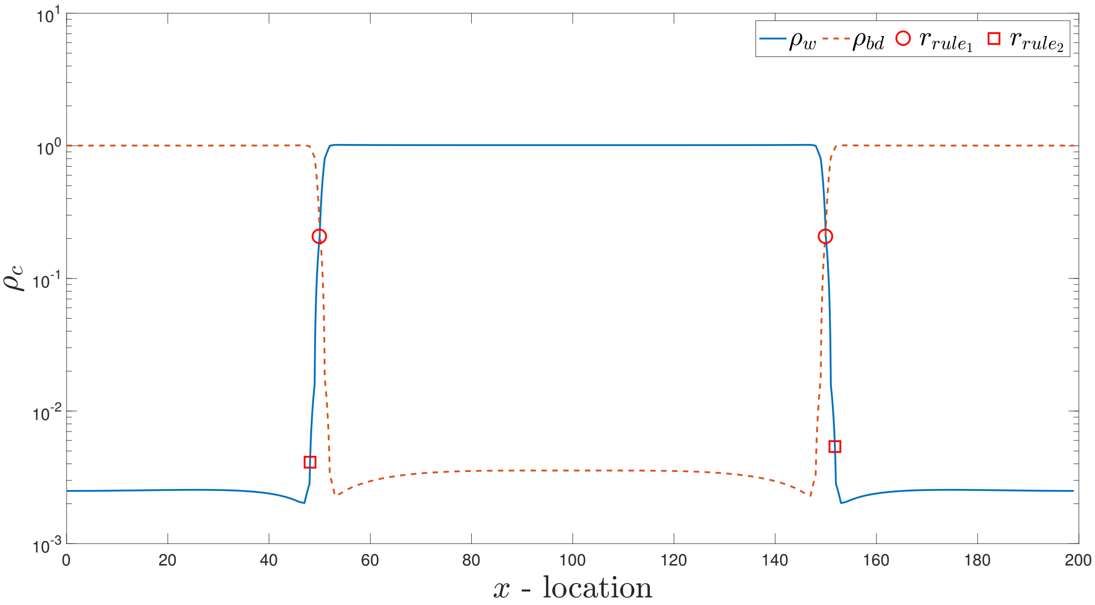


Figure 5.2: Density profiles of both components of a drop with initial radius $r_0 = 50$. Comparison between the sizes produced by the rules used to measure radius.

Once the radius is established, experiments at different values of $G_{c\bar{c}}$ can be run to establish its relation to interfacial tension. For the purpose, a total of 40 simulations were performed. In each simulation 12 000 time steps were allowed for the droplet to reach equilibrium before the measurements were taken. The results are summarized in Figure 5.3. When fitting the first order polynomials to the data for relatively small values of $G_{c\bar{c}}$, care must be taken with the interpretation of the coefficients. When $G_{c\bar{c}}$ is small, the diffuse interface between components is wider, as mentioned above. This leads to the region shaded in blue marked "*Mixing Region*". Running simulations with parameter values in this region leads to the two components being mixed. Similarly, having $G_{c\bar{c}}$ at a value high enough to place the Laplace gradient in the "*Unstable Region*" leads to excessive velocity fluctuations and divergence.

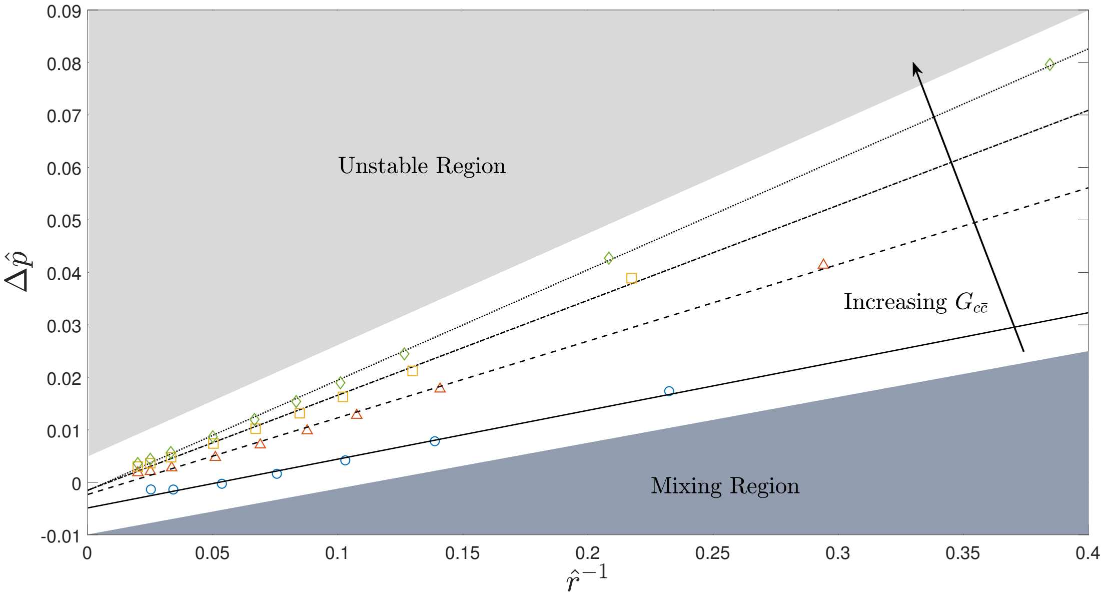
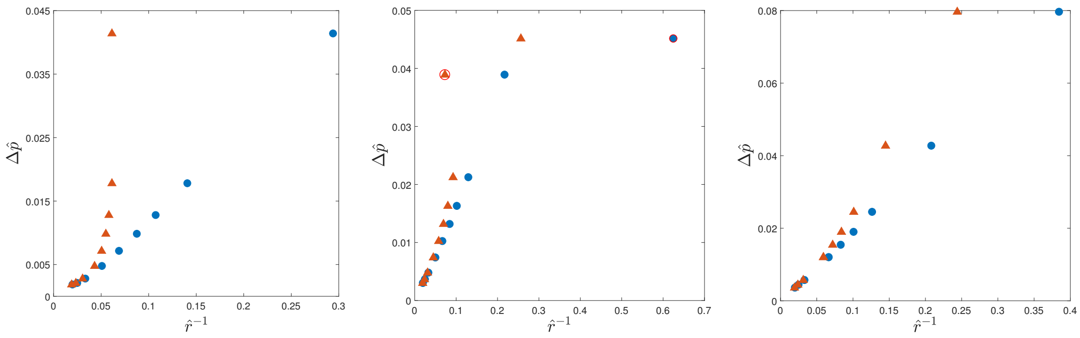


Figure 5.3: Evolution of interfacial tension for varying interaction strengths. The steady reduction of droplet size with decreasing $G_{c\bar{c}}$ can be observed by connecting data points corresponding to the same reciprocal radius.

If at the same time the initial droplet radius is small, the effect of the interface becomes non-negligible. The effects of this phenomenon can be observed in Figure 5.4. Especially, in Figure 5.4(a) it can be seen that there are 9, instead of 10 data points respectively, because the smallest droplet with $\hat{r}_d = 3$ have evaporated. Furthermore in Figure 5.4(b) there are 10 points, but the last one lies far away from the otherwise linear trend of the group, due to the same effect. An additional argument for the use of the first radius calculation rule, mentioned above, can also be seen in Figure 5.4, where the data obtained via the second rule exhibits a higher degree of non-linearity. Finally, all pressure differentials should be positive.



(a) $G_{c\bar{c}} = 2.5$ (b) $G_{c\bar{c}} = 3.0$ (c) $G_{c\bar{c}} = 3.5$

Figure 5.4: Laplace plots for different values of the interaction strength parameter $G_{c\bar{c}}$. Points marked with a circle are loosely considered to be outliers and are not used in obtaining the slope of the respective lines. Points correspond to Rule 1 and triangles to Rule 2.

It is important to understand the overall effect of the interaction strength parameter $G_{c\bar{c}}$ on interfacial tension. The relationship between the two for the conducted experiments is shown in Figure 5.5. The blue dots indicate the interfacial tensions of each of the experiments that were performed. Least squares regression (up to third degree) was used with the data to summarize the trend. As expected the higher the order of the fitting polynomial the better its fit to the data. The sum of squared errors, J_i, for the i^{th} degree polynomial was, $J_1 = 1.48 \times 10^{-4}$, $J_2 = 8.22 \times 10^{-6}$ and $J_3 = 2.18 \times 10^{-32}$. These figures are however not definitive as to the quality of the regression, because higher order polynomials will generally overfit the data. To check this, another two simulations with $G_{c\bar{c}} = 3.8$ and $G_{c\bar{c}} = 4.0$ were run independently of the previous four experiments. Their values of $\hat{\sigma}$ are plotted as red squares on Figure 5.5. As can be seen, the cubic fit does not generalize as well as the square one even though the former has a much smaller J_i.

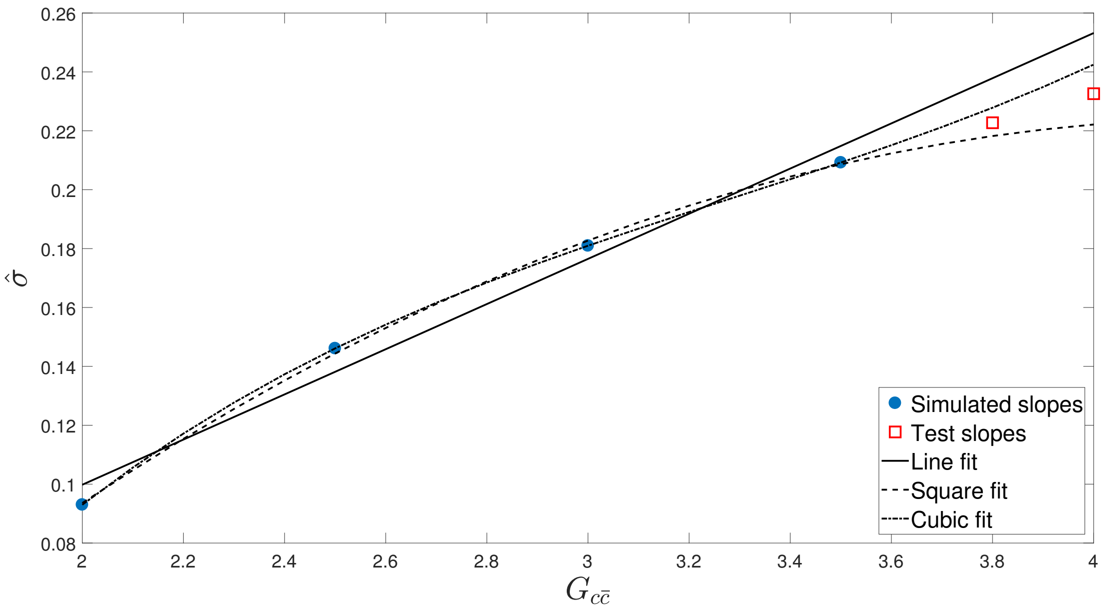


Figure 5.5: Surface tension changes with $G_{c\bar{c}}$ approximately quadratically, up to $G_{c\bar{c}} = 4.0$. There is no significant increase in σ after $G_{c\bar{c}} = 3.5$.

In order to relate the dimensional analysis outlined in Section 4.2.2 to the interfacial tension experiments in the present section, some challenges in matching experimental IFT values are examined. For water in biodiesel or ULSD emulsions, IFT ranges from 7 mN/m to 20 mN/m [8][1]. Looking at the form of the IFT conversion factor in Table 4.3, it depends on the conversion factors for density, C_ρ, length, C_L and time C_t. As already mentioned the values of some of the conversion factors based on matching Reynolds numbers are impractical. Among these is C_t which was derived from relaxation time and viscosity in Eq. 4.4. This derivation leads to simulations with a very high number of time steps and small velocities. When simulating small droplets at slow speeds problems like increased solubility and abrupt changes in droplet flow begin to occur. The time conversion factor can be redefined to come from a predefined simulation speed $\hat{u}_0$ as $C_t = C_L/C_u$ where $C_u = u_0/\hat{u}_0$. Naturally, this violates the Reynolds number requirement, but as discussed above this violation, provided it is not too large,

[1] In addition to private correspondence with Parker Hannifin.

is not a issue. From Table 4.3, it can be shown that $\hat{d}_f$ and $\hat{u}_0$ have opposite effects on $\hat{\sigma}$ and C_ρ can be taken as a proportionality constant. Taking the fibre diameters and face velocities from Section 4.2.2 one can determine the IFT values obtainable in practical simulations. Combinations of LBM velocities and lattice sizes to represent an 8 µm fibre are shown in Figure 5.6. There, it can be seen that as u_0 decreases ((a) to (f)), achieving suitable σ becomes increasingly difficult, as indicated by the shrinking grey region. As shown in Table 4.3 $\hat{u}_0$ can be set at an experimentally tested value, $\hat{u}_0 = 1 \times 10^{-2}$, but this depends on the size of simulated droplets. For instance at $G_{c\bar{c}} = 3.5$ a droplet with diameter of 6 lattice units was successfully simulated in a flow with $\hat{u}_0 = 2.5 \times 10^{-3}$. If this set-up represents $u_0 = 5$mm/s and IFT at the lower bound of its range *i.e.* 7 mN/m, the physical droplet diameter should be $d_d \approx 25$ µm as compared to the target minimum drop diameter, $d_d \approx 5$ µm.

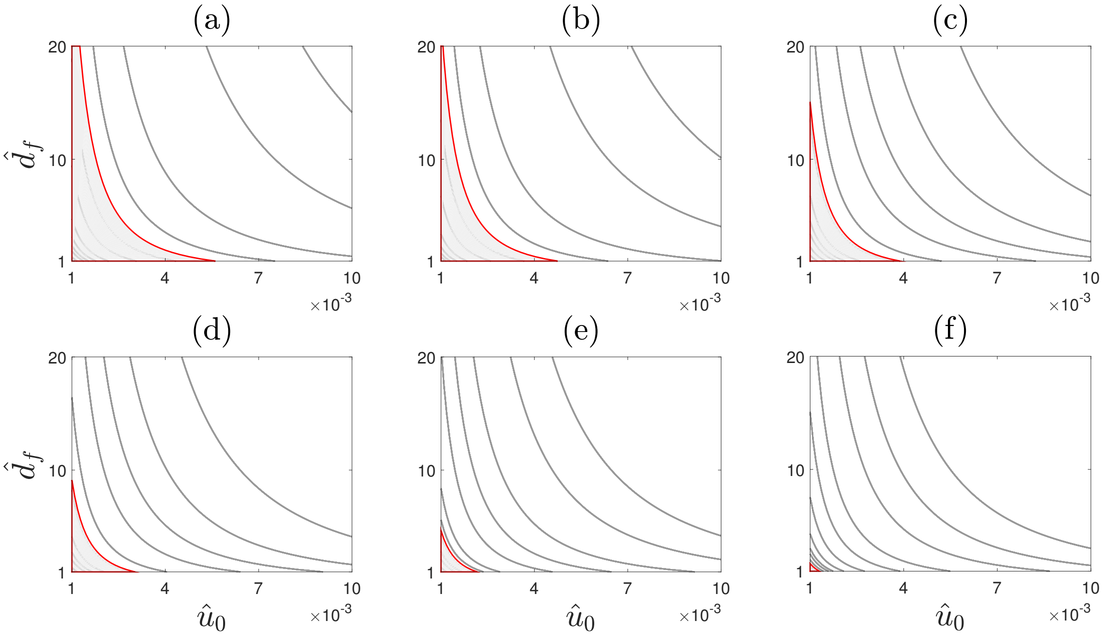


Figure 5.6: IFT contours. Combinations of lattice velocities and fibre sizes to achieve $\sigma = 9$ mN/m are shown as shaded regions. Individual subfigures represent distinct velocity values - (a) $u_0 = 13$ mm/s; (b) $u_0 = 11$ mm/s; (c) $u_0 = 9$ mm/s; (d) $u_0 = 7$ mm/s; (e) $u_0 = 5$ mm/s; (f) $u_0 = 3$ mm/s. $G_{c\bar{c}} = 3.5$ everywhere.

Extending the spectrum of IFT values to smaller droplets and physical velocities, requires simulations to be run at lower speeds without compromising stability. Another way to tune IFT independently of physical parameters is to change the interaction strength parameter, $G_{c\bar{c}}$. As σ is related to $G_{c\bar{c}}$ through C_σ and $\hat{P}$, using larger interaction potentials can directly increase the IFT. Using the original Shan-Chen velocity shift scheme for force incorporation prohibits $G_{c\bar{c}} > 4$ to be used as seen in Figure 5.3.

5.2 Filter experiments - preliminaries

In the initial stages of the development of the computational framework, simulations were kept as simple as physically meaningful to avoid unnecessary time costs. Full LBM simulations can be very computationally demanding and performing them without clear understanding of the important underlying principles was deemed inefficient. Instead, emphasis was placed on two-dimensional (2D) test cases, discussed in Section 5.3. These set-ups can be thought of as representations of infinitesimally thin slices of the filter mesh. Certain properties of the mesh are lost, such as its depth, fibre orientation and curvature. Despite this, they provide useful insight into the micro dynamics of filtration and understanding of the role of some variables as well as a means of efficiently developing full scale three-dimensional simulations introduced in Section 5.4.

Experiments involving single drop and fibre, multiple dynamically generated droplets interacting with a single fibre and multi-body interactions were performed. The first two act as guides in understanding various aspects of the model and to clarify ranges for different variables. In addition, those simple simulations were used in early studies of how the LB filter model can be coupled with Gaussian process emulators, described in Chapter 3, and used as a test bed for different uncertainty quantification methods, described in Chapter 6 and Chapter 7.

5.2.1 Input variables and their distributions

The performance of the coalescer depends on many variables characterising bed, fluid and flow properties. As outlined in Chapter 1, the influence of some of these variables has been observed in physical experiments and inferred through empirical modelling. However, due to the nature of experimentation, limited number of values for each variable can be tested, introducing uncertainty in the effect of the variable on the performance metrics. One of the goals of this work is to examine and if possible to corroborate expert knowledge by performing simulations with values from across the range of each variable. Lattice Boltzmann simulations also allow for other implicit factors and interactions among variables to be examined.

In order to construct the model, the initial set of input variables was reduced to allow simple, relatively fast simulations to be run. In addition, the SCMP model was used at that point to examine its applicability as a less expensive surrogate for multicomponent flow models.

For single fibre simulations, input variables included fibre diameter, d_f, drop diameter, d_d and droplet-fibre contact angle, θ_c. In addition, multi-drop experiments investigated the effect of dynamic generation of droplets at different simulation times. For both groups, the output was taken to be the volume of the heavy (liquid) phase that successfully detached from the fibre and was carried with the flow to the domain outlet. Uniform distributions for all inputs were used, due to the fact that these simulations were synthetic and there is no data for single-fibre experiments. A summary of the distributions is given in Table 5.1.

An important factor to be taken into account is the validity of input data for the model. As mentioned previously, both fibre and droplet diameters are modelled probabilistically and thus plausible distributions describing them need to be identified.

Simulations with multiple fibres and droplets represent two-dimensional sys-

Table 5.1: Inputs and their distributions for single-fibre LBM simulations.

Input	Distribution static	Distribution dynamic	Remark
$\hat{d}_d$	$\mathcal{U}(48, 80)$	$\mathcal{U}(16, 125)$	Droplet diameter (lu).
$\hat{d}_f$	$\mathcal{U}(20, 40)$	$\mathcal{U}(6, 48)$	Fibre diameter (lu).
θ_c	$\mathcal{U}(130, 180)$	$\mathcal{U}(130, 180)$	Contact angle (deg).

tems with N bodies having at least $2N$ positional and N scale parameters - a total of $3N$ degrees of freedom. The only shape which is defined by a single parameter and is realistic as fibre cross section is the circle. Filter media has a typical percentage of voids, also known as *porosity* of ε from 50 % to 90 %. These porosity levels, combined with typical median fibre sizes from 2 µm to 20 µm, result in $N = [\mathcal{O}(10^2) - \mathcal{O}(10^4)]$ and corresponding number of parameters. Therefore, positions and sizes are defined as coming from a probability distribution and are considered random up to some parameters. Here, it is important to note that this does not mean the model is stochastic, only its inputs are. To remind the reader, this means that even though they are defined probabilistically, if the same sample of fibres and droplets is fed into two different simulation their outcome will be exactly the same. The introduction of stochastic properties is discussed in some more detail is Section 5.3.4. To summarize, variables representing the bulk properties of media and emulsion, such as distribution of fibre and droplet diameters, respectively denoted as d_f and d_d, as well as porosity, ε, porosity gradient, $\Delta\varepsilon$ and thickness, t were considered. Flow variables such as velocity, water concentration and interfacial tension were also included.

In order to increase the plausibility of the model, droplet and fibre distributions were inferred by obtaining a set of data from experimental studies [134] and fitting probability density curves, using the maximum likelihood estimation to compute their parameters. For droplets, the volumetric distribution of their diameters [135] was used, whereas for fibres, data from the analysis of scanning electron

Table 5.2: Variables in a 2D filter model.

Input	Symbol	Active variable	Remarks
Fibre diameter	d_f	2 µm to 20 µm	Median values.
Media thickness	t	1 mm to 5 mm	Along flow direction.
Porosity	ε	50 % to 90 %	Inferred from *bulk density*.
Porosity gradient	$\Delta\varepsilon$	2 % to 12 %	Linear variation across filter thickness.
Contact angle	θ_c	0 deg to 180 deg	Constant through filter.

microscopy (SEM)[2] images was provided. All fitted distributions were ranked by their negative log-likelihood, $(-\ln\mathcal{L})$ for the particular data set and those with the smallest $(-\ln\mathcal{L})$ were chosen as representative. For the experiments in Section 5.3.4 truncated normal, log-normal and Student-t distributions were all identified as good candidates. Since the support of these distributions is either infinite or semi-infinite, truncation was applied to prevent extreme values of both fibres and droplets appearing in the simulation. The controlled variables for both fibre and droplet size distributions were their medians. All other parameters were modelled using uniform distributions with appropriate bounds. Table 5.2 summarizes the properties of the filter bed, whereas Table 5.3 gives those of the fluid flow.

5.3 Filter experiments - 2D

5.3.1 Contact angle experiments

Prior to using the code to simulate drop-fibre interactions, a set of experiments was run to determine the relationship between the contact angle, θ_c and the wall affinity parameter, η. The experiments, a selection of which are shown on

[2] More on SEM in Chapter 8.

Table 5.3: Variables in a 2D fluid model.

Input	**Symbol**	**Active variable**	**Remarks**
Droplet diameter	d_d	10 µm to 150 µm	Volume median values.
Flow velocity	u_x	3 mm/s to 10 mm/s	Face velocity[3] based on volumetric flow rate.
Water concentration	c	1500 ppm to 20 000 ppm	Based on experimental observations.
Interfacial tension	σ	9 mN/m to 20 mN/m	Figures for ULSD and biodiesel.

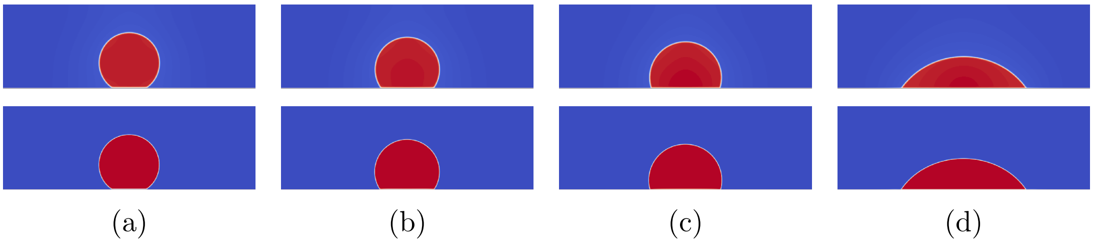


Figure 5.7: Droplets resting on a substrate with different wettability. Top - $G_{c\bar{c}} = 2.0$, bottom - $G_{c\bar{c}} = 3.5$. The value of η in each column is (a) $\eta = 0.2$, (b) $\eta = 0.3$, (c) $\eta = 0.4$, (d) $\eta = 0.7$.

Figure 5.7 comprised of simulating a static droplet relaxing on a substrate with a given value of η. For each experiment the resulting contact angle was calculated as

$$\theta_c = 2\tan^{-1}\left(\frac{2H}{D}\right) \tag{5.1}$$

where H and D are the height and base of the droplet. The convention for these two parameters, as well as for θ_c is shown on Figure 5.8. The result of the experiments, carried out for different values of $G_{c\bar{c}}$, is shown in Figure 5.9, where experimental data is plotted as separate markers and solid lines represent least-squares fit for θ_c at each value of $G_{c\bar{c}}$. Second order polynomials were used to provide trends for all interaction parameters, except $G_{c\bar{c}} = 2.0$, which had a cubic fit. It can be seen that the line of best fit for $G_{c\bar{c}} = 4.0$ diverges from the rest

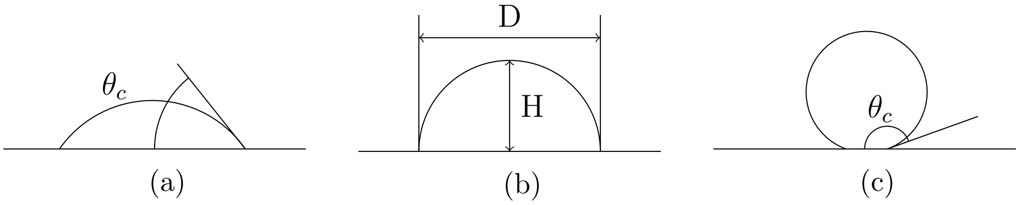


Figure 5.8: Contact angle convention and nomenclature. The angle measures through the water from the substrate surface to the liquid-liquid interface. Three distinct contact angles are shown corresponding to (a) hydrophilic substrate with $\theta_c < 90°$, (b) neutral substrate with $\theta_c = 90°$ and (c) hydrophobic substrate with $\theta_c > 90°$.

of the lines for $\eta > 0.6$. This is due to the fact that simulations for those values of η were unstable and were thus excluded from the fit. The divergence of the simulations for $G_{c\bar{c}} = 4.0$ supports the conclusions made in Section 5.1, where it was determined that such high value for the interaction parameter can indeed lead to instabilities in the simulation.

5.3.2 Single drop single fibre experiments

The single fibre, single drop experiment was used to test the performance of the Gaussian process emulator, discussed in Chapter 3. The uncertainty quantification techniques that will be studied in detail in Chapter 6 and Chapter 7 are also tested with this set-up. The simulation uses an SCMP formulation and comprises a heavy phase droplet submerged in a light phase medium impacting with a hydrophobic fibre. The simulation was set to run with $G = -6$, resulting in two distinct phases with densities, $\rho_l = 2.65$ for the liquid phase and $\rho_g = 0.076$ for the gaseous phase, as shown on Figure 2.6. Two separate relaxation times were used for each phase, with values $\tau_l = 0.65$ and $\tau_g = 1.0$. These values were found to give a sufficiently stable simulation. It should be noted here that, even though the dimensional analysis in Chapter 4 was carried out for the MCMP model, a similar procedure is applicable to the SCMP formulation.

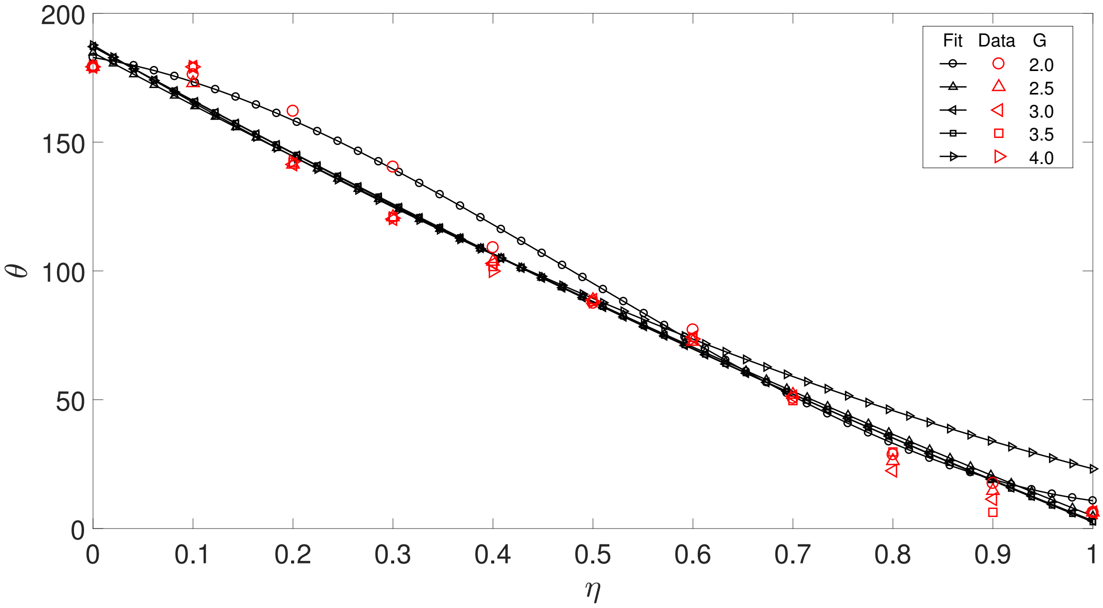
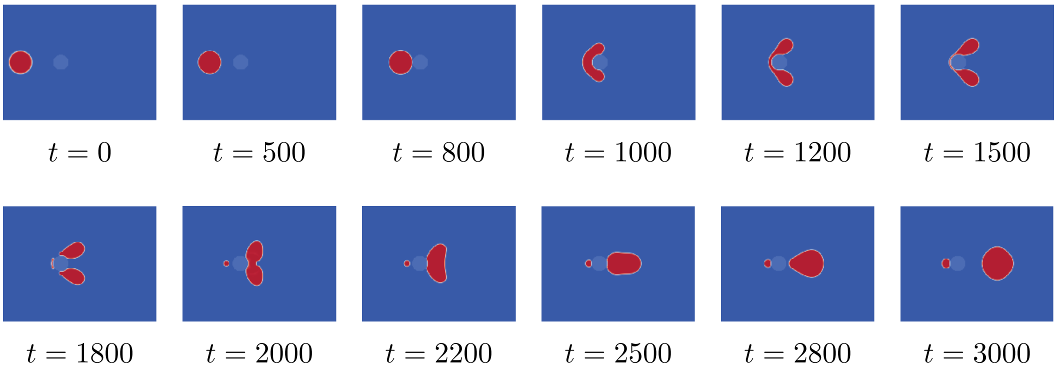


Figure 5.9: Relationship between contact angle, θ_c and wall affinity, η. The markers represent simulation data. The least-squares fit for each experiment is shown as a solid line. The divergence of the fit for $G_{c\bar{c}} = 4.0$ is due to missing data, caused by unstable simulations for $\eta > 0.6$ at this interaction strength.



$t = 0$ $t = 500$ $t = 800$ $t = 1000$ $t = 1200$ $t = 1500$

$t = 1800$ $t = 2000$ $t = 2200$ $t = 2500$ $t = 2800$ $t = 3000$

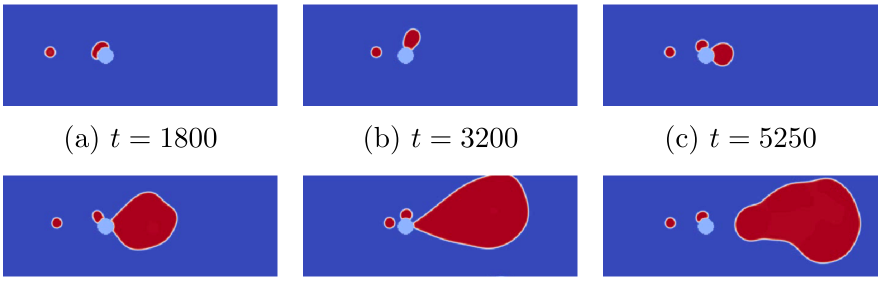
Figure 5.10: A selection of steps from the single component multiphase model used for preliminary emulation.

The domain was 400 lu long and 200 lu wide and is shown in Figure 5.10. Simulations were performed with velocity inlet and Neumann outlet boundaries in the flow direction and periodic boundaries in the transverse direction. The inlet and outlet boundary conditions allowed the domain to be simulated as having large upstream and downstream portions without physically including them.

Each simulation was run until some volume of water was classed as separated by the algorithm introduced in Section 4.3. The simulation was run 100 times to provide data for the Gaussian process emulator. A single execution of the code took from 60 s to 250 s running on 25 Intel® Xeon® E5-2670 processors. The code input distributions for each input variable are shown in Table 5.1 in the **Distribution static** column. The emulation of this code's output is discussed in Section 5.6.

5.3.3 Multi-drop, single fibre experiments

The next logical step in the development of the model was to enable the dynamic generation of droplets in the domain and to verify the stability of the simulation. A code set-up with dynamically seeded droplets is very important to the functionality of the end tool as it permits the user to change the water concentration in the system.


(a) $t = 1800$ (b) $t = 3200$ (c) $t = 5250$

(d) $t = 22700$ (e) $t = 37900$ (f) $t = 41300$

Figure 5.11: A selection of steps from the dynamic drop experiments. The enlarged droplet separates in (f) at $t \approx 41000$ ts, corresponding to approximately 0.82 seconds.

In the simulation each droplet is generated as a region of heavy phase according to the following equation [79]

$$\rho(x,y) = \frac{1}{2}\left[(\rho_c + \rho_{\bar{c}}) - (\rho_c - \rho_{\bar{c}})\tanh\left(\frac{2\left(\sqrt{(x-x_c)^2 + (y-y_c)^2} - r\right)}{W}\right)\right] \quad (5.2)$$

where the undefined terms are x and y denoting the coordinates within the domain, x_c and y_c which are the x and y coordinates of the centre of the droplet, respectively and W, the thickness of the diffuse interface of the droplet. The tanh function provides a smooth transition from light to heavy phase, which improves the stability of the simulation. Eq. 5.2 is used to initialize droplets in all models used in this dissertation.

Experiments with this arrangement were run with different seed times, effectively changing the concentration of water. The input distributions used in the simulation are given in Table 5.1 in the column titled **Distribution dynamic**. The domain size and LBM settings were identical to those used for the static droplet studies. The results from these experiments were also used to test the separation algorithm as for some input configurations the separating water droplet exhibited very complex dynamics. A selection of time steps is shown on Figure 5.11.

The output of interest was once again the volume of water leaving the fibre. In order to provide data for the GPE, 100 simulation runs were performed. Discussion and results of the emulation of this arrangement are discussed in Section 5.6.

5.3.4 Multi-body experiments

Once the performance of the model was successfully verified for single fibre scenarios, more fibres were introduced in the domain. This step was taken as a final preliminary study in the modelling of the filter, before transitioning to three-dimensional simulations. These experiments are referred to as *multi-body* experiments and are carried out using the MCMP model. Despite the associated increase in running time the MCMP formulation increases the stability of simulations and significantly reduces the effects of evaporation and condensation. The term multi-body indicates the fact that multiple water droplets interact with multiple fibres. In two dimensions both entities are represented with circles and all volumetric properties are calculated on an area basis. The simulation set-up can be seen as an infinitesimal slice of the real control volume. This is equivalent to having one of the three dimensions of a real filter to be degenerate. Different properties can be investigated, either all at once or in subsets. The motivation behind conducting computer experiments, in which parameters are varied one at a time is twofold. First, it serves to ensure the simulator can handle set-ups of realistic size as opposed to illustrative studies, such as those presented above. Second, it allows to check assumptions in the design of experiments.

Even though the model is two-dimensional, the number of parameters is significant. Therefore, some variables, such as velocity and IFT were fixed in favour of varying others, such as fibre and droplet size distributions and porosity, which are deemed by experts to have a significant effect on filter performance. Deciding which input variables are *important* in the sense of their influence on the predicted performance of the model, is the aim of sensitivity analysis and will

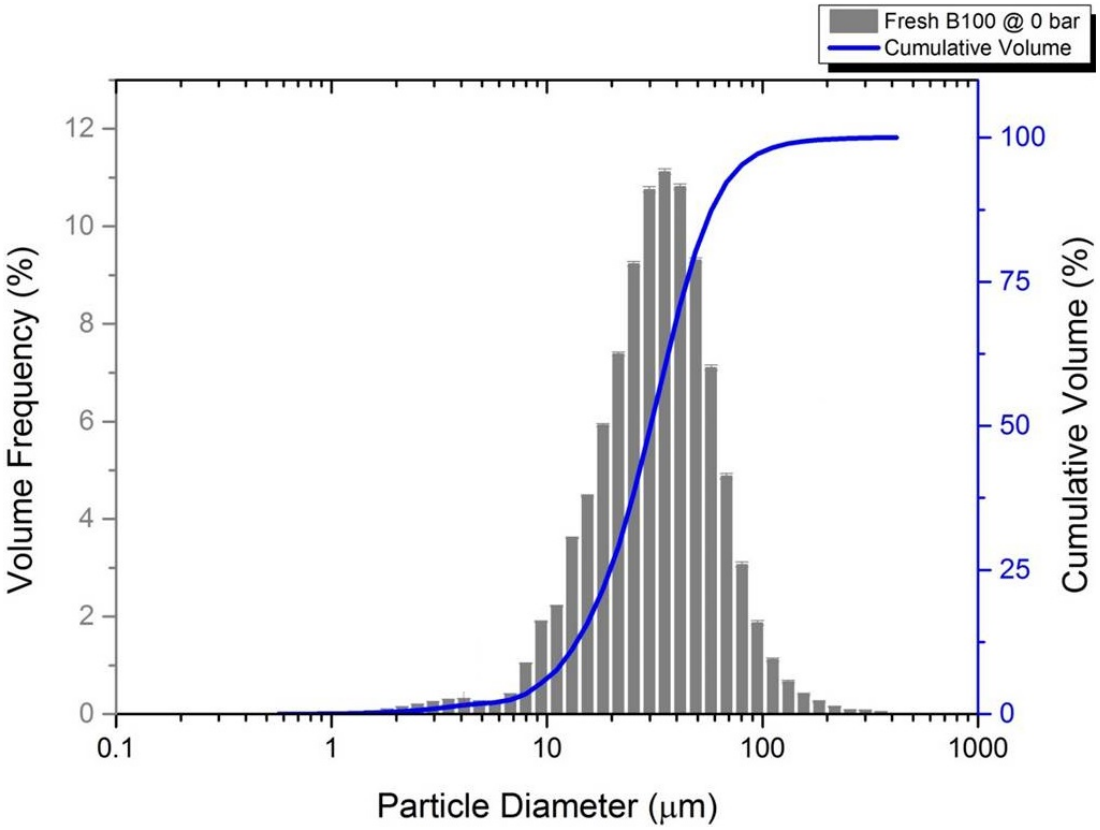


Figure 5.12: Influent droplet size distribution with $D_{v_{50}} \approx 25$.

be discussed in Chapter 6. The goal is to eventually run simulations with all parameters varying simultaneously and use uncertainty quantification methods to study the filtration process.

The ranges of different flow and filter properties are given in Table 5.3 and Table 5.2, respectively. As mentioned in Section 5.2, normal and log-normal distributions were fitted to data from SEM and particle sizing experiments. Figure 5.12 shows a typical influent droplet distribution obtained in laboratory setting. It can be seen that the distribution is predominantly uni-modal, with the exception of a small peak around 4 µm. The distribution of water droplets is thus reasonably well described by its volumetric median, $D_{v_{50}}$.

A sample of the fitted log-normal water droplet distributions is shown in Figure 5.13. No experimental data was available for some of the desired water chal-

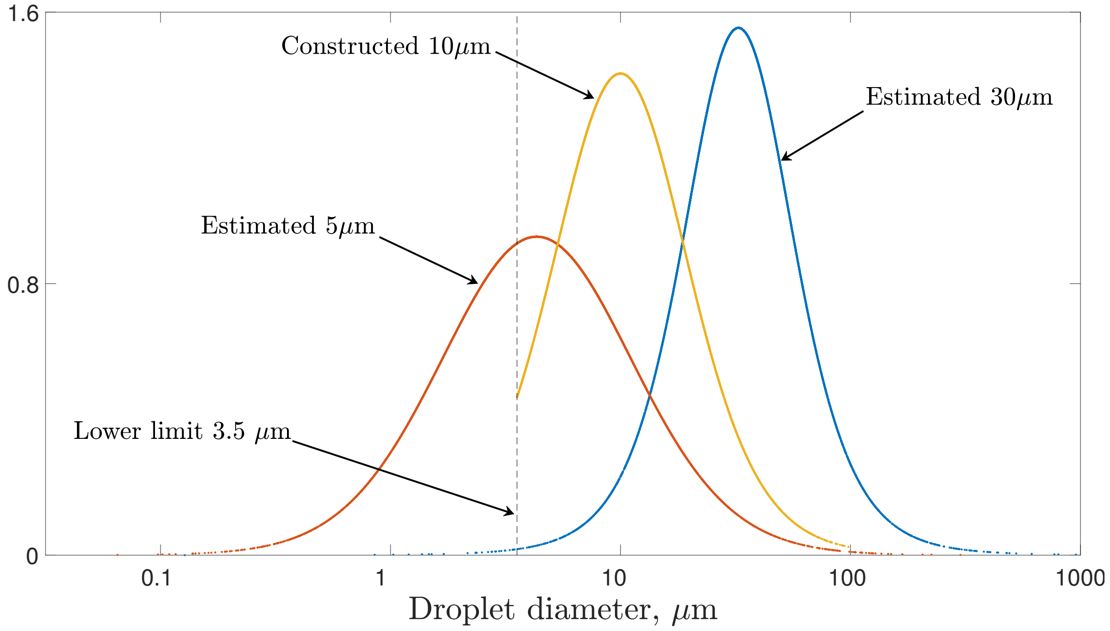


Figure 5.13: Influent droplet size distributions. The curves labelled as estimated were fitted to existing experimental data, whereas the one denoted as constructed was built based on parameters of the other two distributions. All droplets studies use distributions truncated at limits ensuring simulation stability.

lenges and thus they had to be constructed. To achieve this, the parameters of the closest two distributions bounding the range of interest were used to estimate those of the new distribution. The two curves estimated from data for 5 µm and 30 µm were used to construct the distribution for 10 µm droplets. This case is shown on Figure 5.13.

In multi-body studies one can give meaning to the terms separation efficiency and pressure drop. This is due to the presence of multiple fibres which obstruct the flow and present an obstacle for the water emulsion. In the experiments presented in this section, separation efficiency is the primary output of interest. This figure of merit is modelled as the shift in the median of the droplet size distributions from inlet to outlet. Modelling pressure drop requires the use of special boundary conditions and it was therefore decided that it will be included directly in future three-dimensional studies, as outlined in Chapter 9, Section 9.3.

One of the variables that was fixed in these experiments was the *water concentration.* To conform to experimental procedures a value of 2500 ppm was chosen. To match this concentration between reality and simulation, the method outlined below is used. The volumetric flow rate is given by

$$Q = \mathbf{u}A_{\perp} \tag{5.3}$$

where $\mathbf{u}$ is the inlet velocity and $A_{\perp}$ is the projection of the area of the inlet perpendicular to $\mathbf{u}$. In three dimensions, flow rate has units of m^3/s. Translated to two dimensions this relation reads

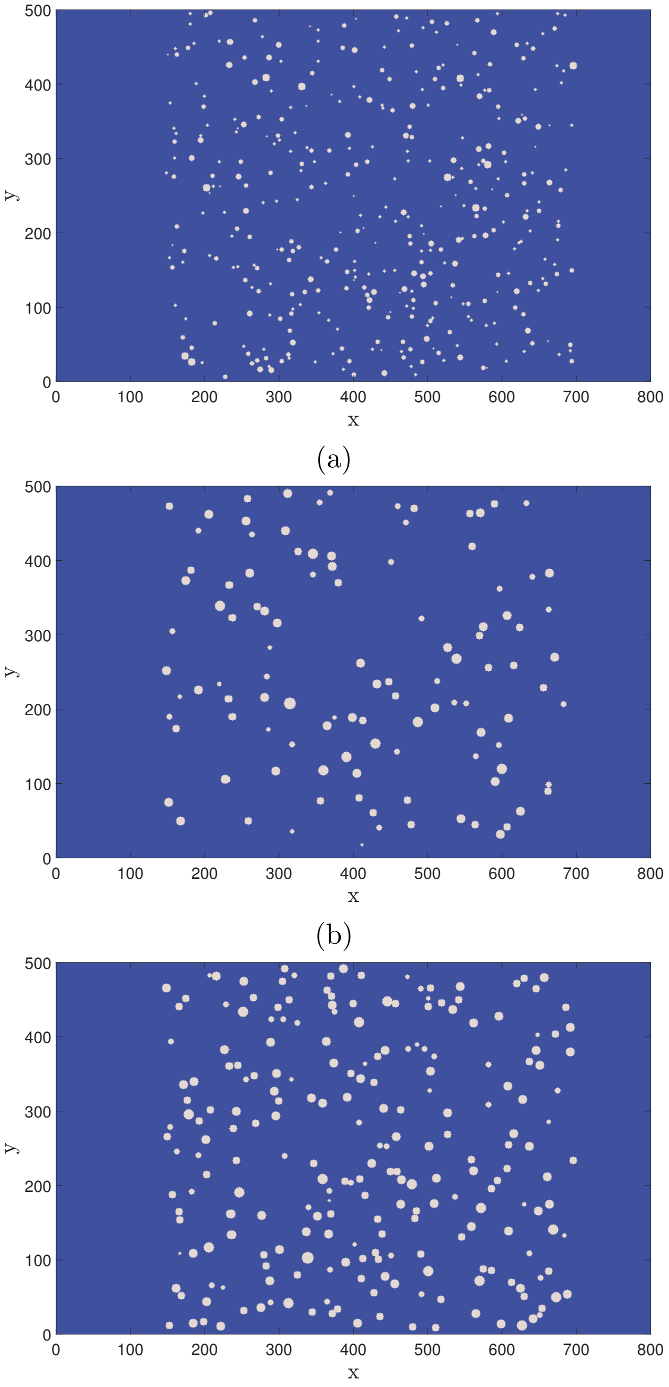
$$Q' = \mathbf{u}w \tag{5.4}$$

where w is the *width* of the computational domain. Note that w is not specified as perpendicular to the flow velocity. This is because the simulations are set up having only horizontal velocity, $\mathbf{u} = [u, 0]$. This assumption is not incorrect since the simulations deal with flat filters and any cross components of velocity will generally be in the third, gravity-oriented dimension omitted here. Concentration values are typically specified on per-unit-volume basis, so for flow scenarios using volumetric values is intuitive. Once the inflow of diesel is established, the concentration can be used to calculate the inflow of water at every time step of the simulation. This average value is combined with pre-sampled droplet diameters to determine the seed rate of virtual droplets.

The size of the simulation domain was primarily determined by available computational resources. Since the simulation was set up as 1-to-1 scale in size at the micron level, 1 mm bed thickness, translated into a domain of length 1000 lattice units. The width was set at 500 lu. Output files containing density and velocity magnitude in the field, for a single time step were approximately 10 MB each. Simulations were run for 10 000 000 time steps. Using the procedure outlined in Section 4.2 this duration corresponds to approximately 17 s of physical time. Writing output files every 2000 times steps results in 50 GB of data. At the same time laboratory experiments show that typical coalescing filters reach their steady operation after several minutes from initial start. When simulations are run solely for the purpose of performing uncertainty quantification, instead of visual inspection, the volume of output files is expected to decrease as there will be no need to store information about the whole domain. Another issue is CPU time. A simulation with a domain and duration similar to those described above takes about 48 hours to complete on 25 Intel® Xeon® E5-2670 processors.

To alleviate the problems discussed above, simulations were carried out on a reduced domain of size 550 × 500 lu. The detection algorithm outlined in Section 4.3 was used to construct the outlet distributions. An example of a two-dimensional filter mesh with porosity, $\varepsilon \approx 96\%$ is shown in Figure 5.14. The contact angle throughout the bed was set at $\theta_c = 130°$. A selection of snapshots from a simulation with droplets from a distribution with $D_{v_{50}} = 10\,\mu\text{m}$ is shown in Figure 5.15. The velocity fields clearly indicate the regions of flow of accelerated fluid around the fibres. Water droplets tend to follow these paths, which get modified as the bed becomes saturated with water. A comparison between the influent and effluent water droplet size distributions is shown Figure 5.16. The filter was challenged with 450 droplets during the simulation, 88 of which left through the outlet. The reason behind this drop was the short time scale of the simulation in physical time. Despite this, a shift in the median between the two distributions can be seen. Furthermore, Figure 5.16 indicates the presence of 3 droplets with sizes approximately 50 µm - 60 µm which were not among those fed to the filter. These facts show that coalescence does occur in the bed. Experiments with saturated beds are also possible in the simulator, which are expected to decrease the time to steady-state operation of the filter.

Results from the experiment discussed in this section were used to train a *time*

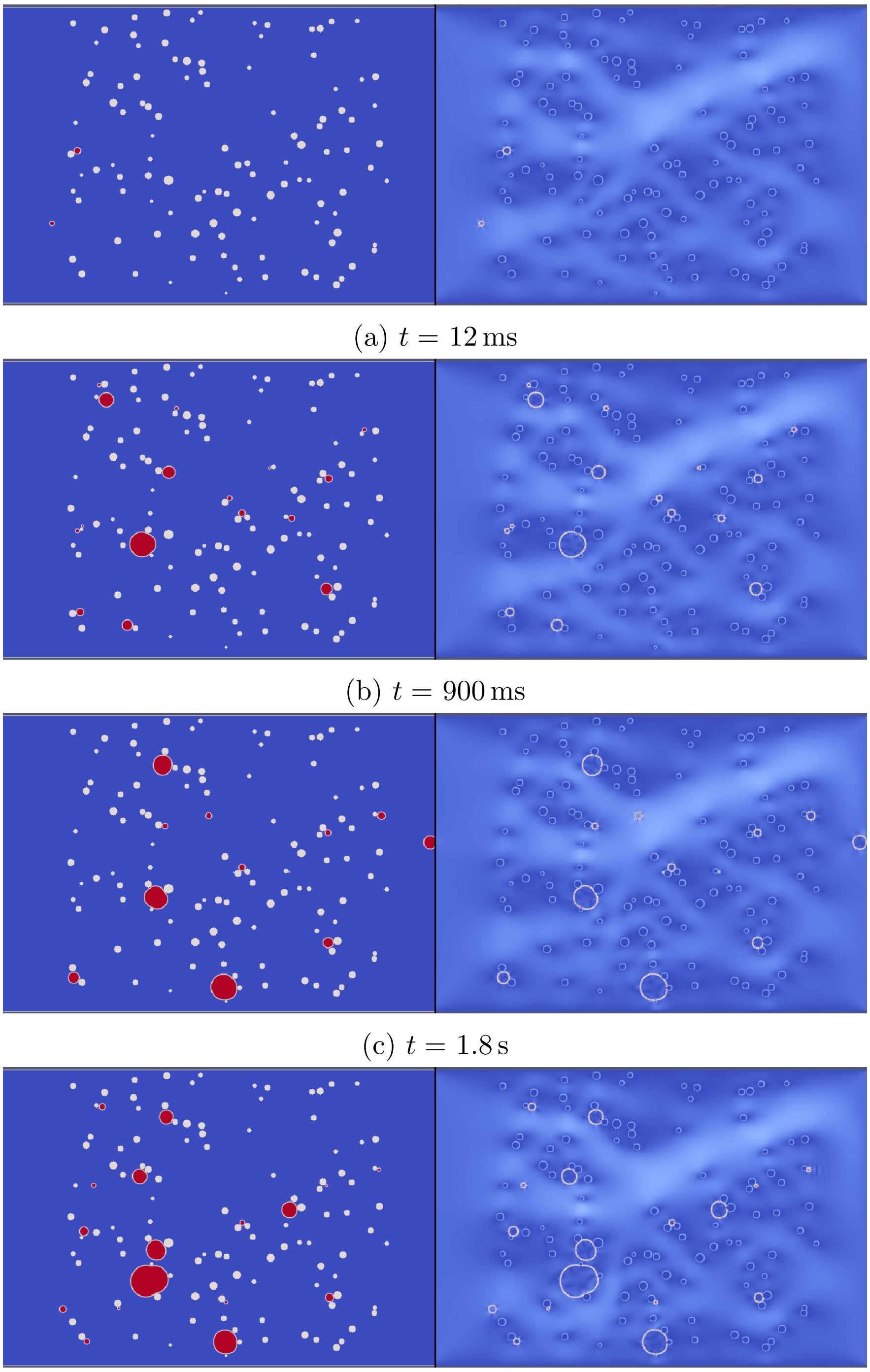


(a)

(b)

(c)

Figure 5.14: Mesh with fibres from (a) 5 µm and (b)-(c) 10 µm distributions. Fibre sizes are sampled from the underlying distribution and their positions are uniformly distributed. Mesh porosity is (a) $\varepsilon \approx 96\%$, (b) $\varepsilon \approx 96\%$ and (c) $\varepsilon \approx 92\%$.



(a) $t = 12\,\mathrm{ms}$

(b) $t = 900\,\mathrm{ms}$

(c) $t = 1.8\,\mathrm{s}$

(d) $t = 2.4\,\mathrm{s}$

Figure 5.15: Simulation with fibres and droplets from a 10 µm distribution and porosity, $\varepsilon \approx 0.96$. Density and velocity profiles are compared. Droplets coalesce in the bed and gradually migrate towards the outlet of the bed.

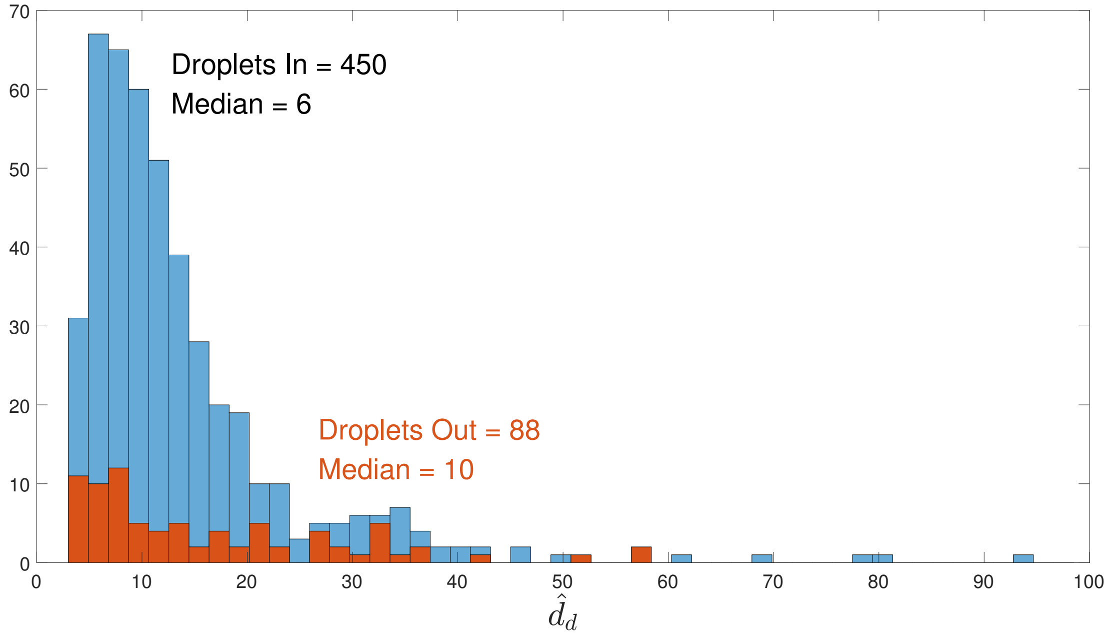


Figure 5.16: Comparison between influent and effluent water droplet size distributions. The number of droplets in each histogram as well as the sample median are shown. A shift in the median is observed at the outlet.

dependent emulator. A discussion about how emulation can be used with LBM to reduce the cost of experiments is provided in Section 5.6.

5.4 Filter experiments - 3D

Filters are inherently three-dimensional objects which exhibit important characteristics in all three dimensions. As mentioned before, one of the parameters of interest to industry is the thickness of the bed. A thinner bed provides a smaller pressure drop and reduced material usage, but may lead to suboptimal coalescence. In order to model the effects of filter thickness, as well as to include experimentally tested features, such as fibre orientation and gravity effects, a three-dimensional model is required. Transitioning from two to three dimensions is relatively easy in LBM. Computational time and memory increases are also associated with such a transition.

Some verification studies were performed as part of the current research. Figure 5.17 shows an example of an SCMP simulation where droplets impact a simple mesh. The fibres were arranged in two layers, such as to first promote coalescence among drops and second to provide a means of drainage. The first layer has $\theta_c = 90°$, whereas the second is hydrophobic with $\theta_c = 160°$. The qualitative differences between two- and three-dimensional simulations become apparent, also raising a question. How does one efficiently represent non-woven bed geometries in the computational domain. This is not only important from a standpoint of analysis of existing filters, but also in a design context. This topic will be discussed in Chapter 8, where the development of a tool for the virtual generation of non-woven beds is presented as the industrial legacy of the project. Future steps of this research are envisaged to make full use of three-dimensional LBM. This is discussed in Section 9.3. Prior to this, the three-dimensional code is used to demonstrate one possible way to qualitatively validate the LBM simulator.

5.5 Validation - 3D

In order to ensure that the LB model employed to simulate coalescence processes is faithful to reality it should be validated. Validation and the related *statistical calibration* are subjects in themselves which are included in the outlook of the project in Chapter 9, Section 9.3. Despite this, one of the established methods to carry out this process, called Bayesian calibration, is directly related to the work presented in this dissertation. In particular Gaussian process emulation is widely used in this area [136] and thus results presented here are directly applicable.

The idea behind the validation was to have a single fibre submerged in diesel with a droplet deposited onto the fibre. The medium was then given a flow velocity and the detachment mechanism of the droplet was observed. A schematic

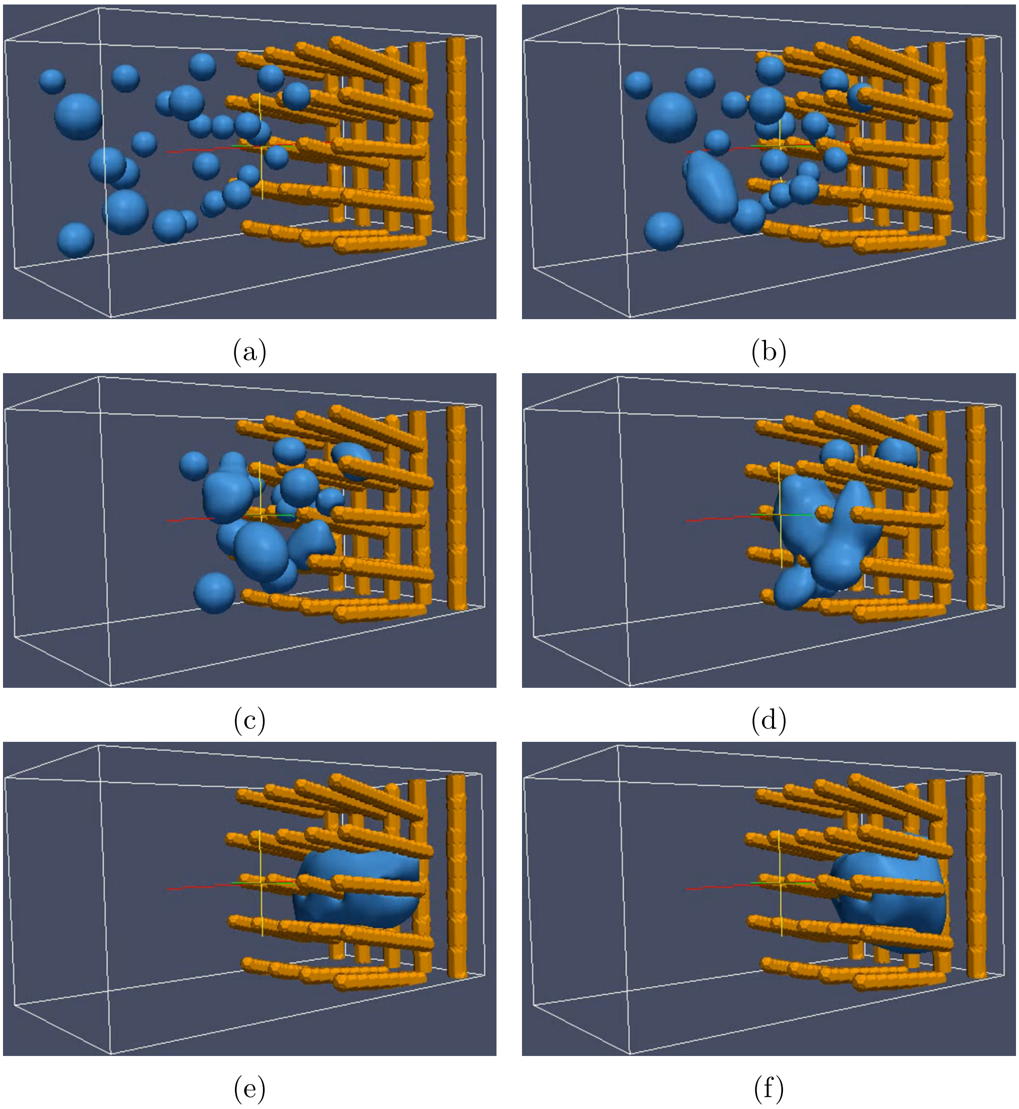
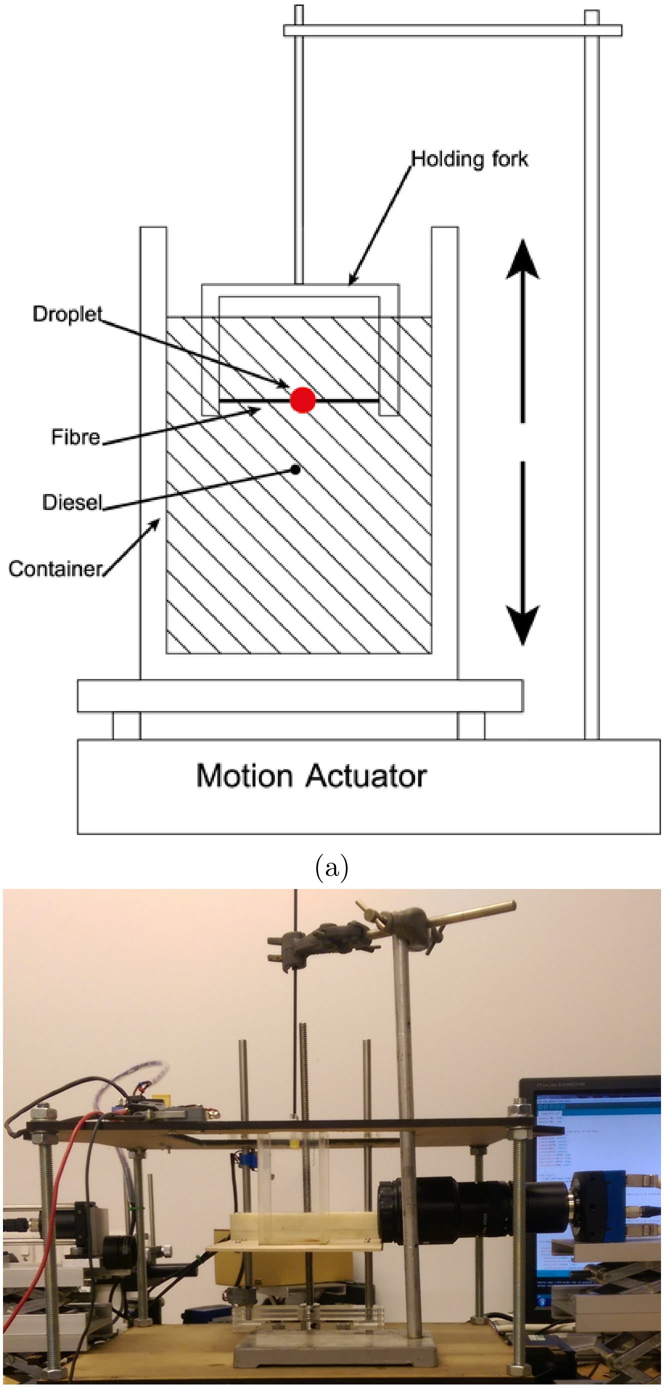


Figure 5.17: Examples from a study used as a demonstration for the capability of LBM to simulate three-dimensional arrangements. The fibres are arranged specifically to promote coalescence and drainage.

of the set-up is shown in Figure 5.18(a) and the finished arrangement is depicted in Figure 5.18(b). The flow of diesel was simulated by moving the container it was in, while keeping the fibre and drop stationary. To achieve this a simple validation rig was designed and built to perform the experiment. Mixed materials and techniques, including rapid prototyping methods were used to manufacture the rig. Actuation was provided by a stepper motor, whose speed was controlled through an Arduino® board. The experiment was filmed with a Mikrotron® MotionBLITZ® high-speed camera with a magnifying lens. The medium for the experiment was ULSD and the container was manufactured from acrylic to minimise reactions with the fuel. The fibre used in the experiment was a 20 µm fibre provided by Parker Hannifin. A microscope picture of the fibre is shown in Figure 5.19(a). The droplet was deposited on the fibre using a medical syringe with a hydrophobic needle. Deposition was done by hand and thus the droplet size was approximately 1 mm. A set of results from the experiment is shown in Figure 5.20, where the panel on the left shows simulation results and that on the right, a magnification of the experiment. The larger diameter of the deposited drop allowed the Reynolds numbers between simulation and experiment to be matched exactly. Observation revealed that this match provides a good agreement between separation dynamics at a simulated velocity, $\hat{u}_0 = 0.00625$.

5.6 GPE for the lattice Boltzmann model

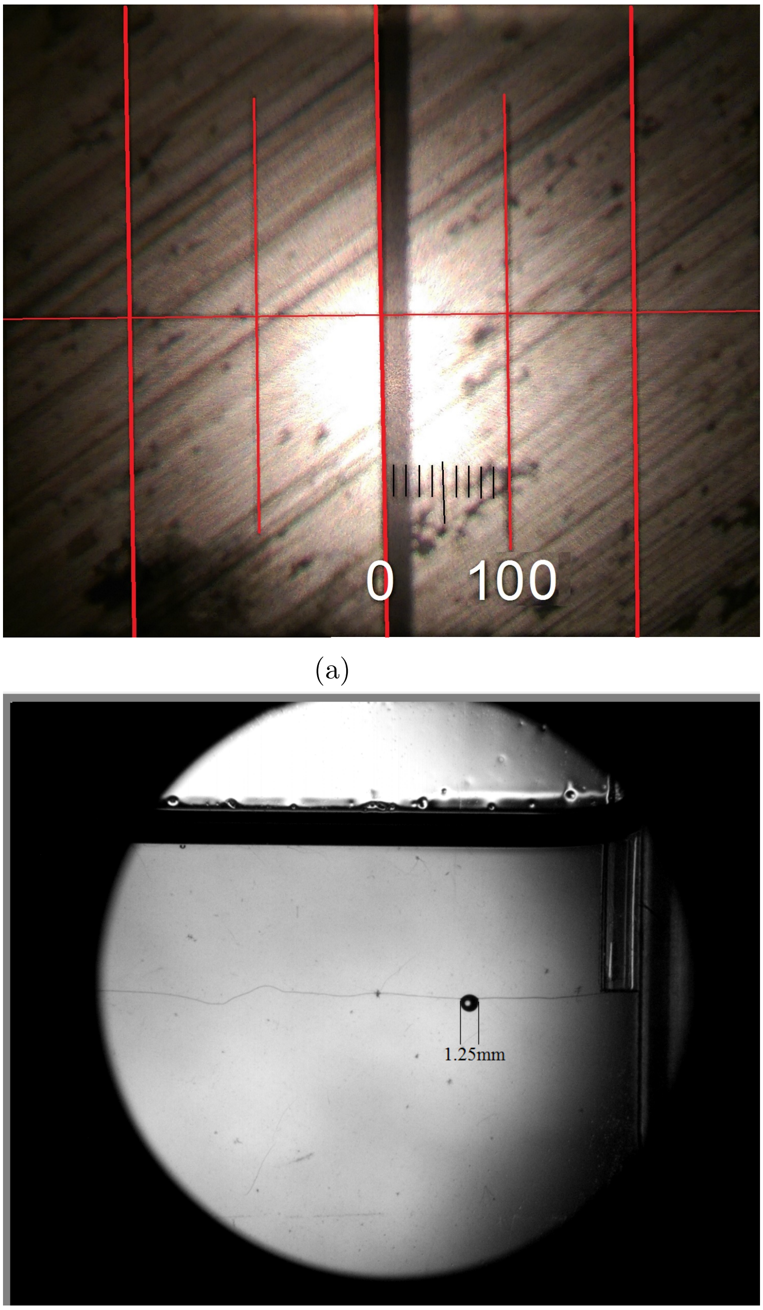
The ultimate goal of the research is to improve the understanding of filtration systems using computational approaches. It is therefore necessary to develop accurate and efficient computer models of the processes taking place in the filter. As mentioned before, an accurate model is likely to be computationally expensive and thus an affordable approximation, such as the Gaussian process emulator is necessary to enable practical analyses. The GPE has been shown to be capable



(a)

(b)

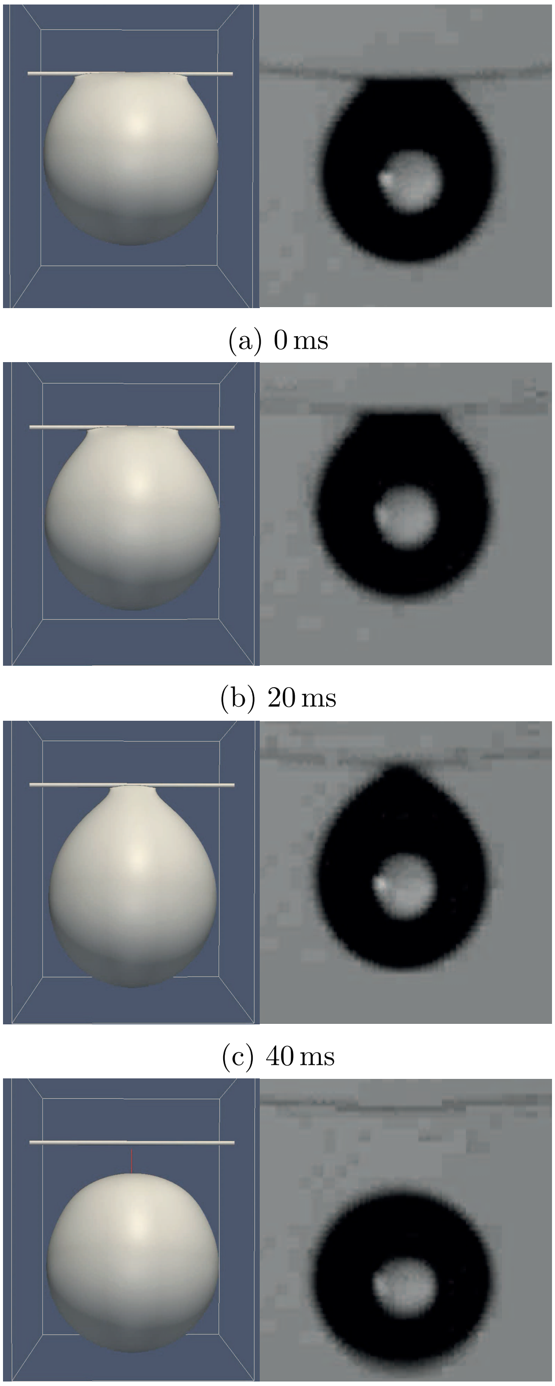
Figure 5.18: Schematic (a) and final view (b) of the experimental rig. The camera and light used to film the experiment can also be seen in (b).



(a)

(b)

Figure 5.19: A magnification of the fibre (a) and droplet (b) used in the validation studies. The fibre is visible in the field of view as a dark band with a width of approximately 20 µm (scale indicates 100 µm). The droplet measures 1.25 mm.



(a) 0 ms

(b) 20 ms

(c) 40 ms

(d) 60 ms

Figure 5.20: The comparison between the computational and physical experiments showed a degree of agreement that provided validation confidence. The simulations were carried out with matching Reynolds numbers and $\hat{u}_0 = 0.00625$ lu.

of emulating realistic, albeit benchmark engineering functions. In this section the lattice Boltzmann model is considered as the underlying model for approximation.

Fluid dynamics models typically return values of the variables of interest at every point in the computational domain. This effectively renders them *multi-output* models. Such simulators are not uncommon in science in engineering, particularly in climate modelling and other natural sciences. Emulation for such models is explored in [137–139]. It is also possible to develop an *array* of uni-variate emulators for each output of the code as suggested in [42]. However for CFD simulations the output of the model at adjacent cells will in general be highly correlated and this approach may not capture the simulator accurately. A way to overcome this problem is explored in [140], where *principal component analysis* (PCA) is used to eliminate the correlations between output values and reduce its dimensionality, before building separate emulators. In some CFD codes a set of end values is of interest, rather than the whole domain. As mentioned in Section 4.3 in the case of LBM, the values of interest are the distribution of droplet sizes at the filter outlet, and more importantly its median, and the overall pressure drop for a given filter geometry. This leaves several options:

1. Treat the pressure as a mean, point value uncorrelated to the median of the droplet size distribution and emulate them separately.

2. Treat the pressure as a mean, point value correlated to the median of the droplet size distribution and emulate them using a multivariate GPE.

3. Emulate the whole droplet distribution at the output and a point pressure value, using a multivariate emulator.

The option using more than a point value for pressure is not included above as it is associated with problems related to choosing the appropriate regions over which to measure the pressure and brings marginal benefits, if any. Emulating the whole distribution may also be unnecessary due to the fact that it is customary to use the shift in distribution medians from inlet to outlet to characterize the separation efficiency of the filter. The effect of emulating the entire droplet size distribution at the outlet can be explored in the future and is included in the outlook from the project in Section 9.3. Treating the pressure and droplet distributions as independent from each other has little physical footing [19] and should not be considered an option[4]. Therefore, a point pressure and the median of the droplet size distributions of the LBM filter simulator can be emulated using a multivariate GPE.

5.6.1 Single droplet experiments

Initially, the simple one fibre, one drop examples, described in Section 5.3 were developed for testing with the GPE. The goal of these studies, was to understand the structure of the problem, typical simulation times and how to interface LBM and GPE efficiently. The details of the simulation set-ups are given in Section 5.3. Since there is only one fibre in the field, pressure drop is not well defined and thus the output of interest was considered to be the volume of water separated from the fibre. The need for a separation event imposed some limitations on the input values of the simulations, summarized in Table 5.1.

For the first study, a single input, the droplet diameter was considered. The input/output relationship has a relatively complex shape as depicted on Figure 5.21(a). Taking this into account, the model was emulated with the Matérn 3/2 kernel as it provides the opportunity of accommodating more irregular responses. Following the sample size recommendation made in Chapter 3, $n = 20$ samples were chosen for training and $m = 30$ for validation. The correlation between predictions and observed values are shown in Figure 5.21(b). There are a few predictions that lie more than two standard deviations from the correspond-

[4] PCA can be used in this case to enable the use of univariate emulators, but one needs to be careful with the physical interpretation of results.

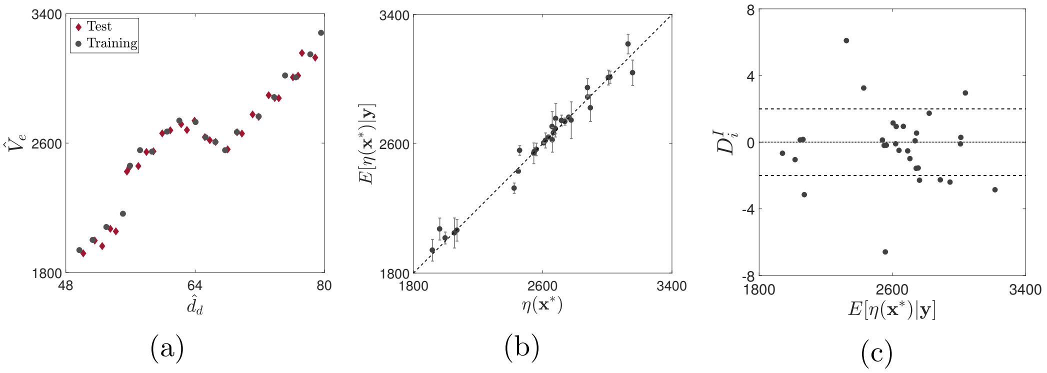
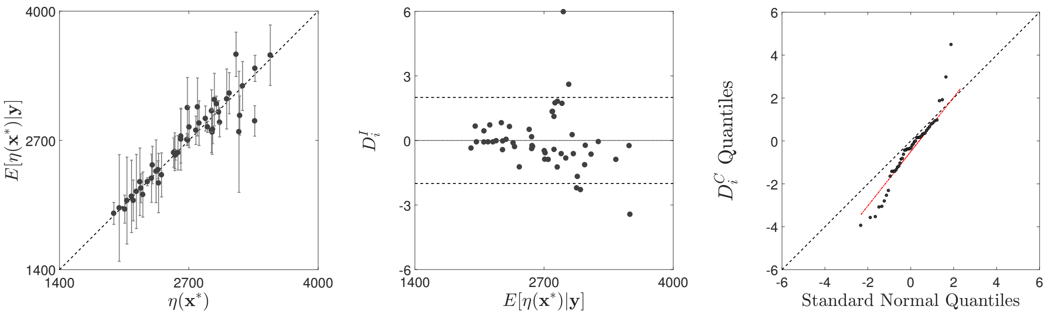


Figure 5.21: Model structure and GPE diagnostics for the single input LBM set-up.

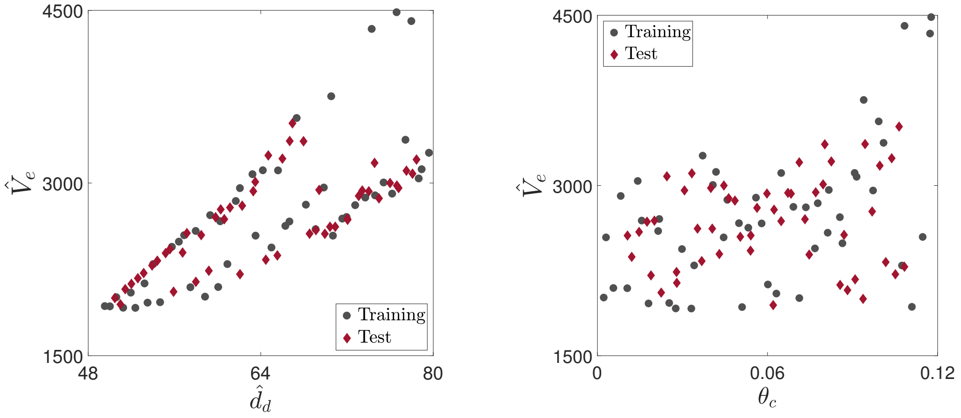
ing observations, particularly towards the middle of the range. Inspecting the IPE, it becomes clear that the emulator experiences difficulties in representing the simulator in that exact region. Going back to Figure 5.21(a), it can be seen that there is a pronounced jump in the values of the response around $V_e = 2600$ lu, which is the most likely reason for the problems. Having only one input variable to explain the behaviour of the output is the most probable root cause for the jump in V_e.

The contact angle between the fibre and droplet was added to examine its effect on the emulation outcome. The LBM code was run at $n = 50$ and $m = 50$ input configurations for training and validation, respectively. The GPE set-up from the single input problem were reused. Results from the emulation are shown in Figure 5.22. The majority of the predicted values correlate well with observations for the same inputs. As seen in Figure 5.22(b) the IPE only seem to exhibit local problems with a single extreme value. The Mahalanobis distance for this set of predictions is $D^{MD} = 108.63$, which lies in the 99th percentile of its distribution. To examine this issue the QQ plot of the uncorrelated residuals is presented in Figure 5.22(c). The data exhibits a considerable amount of curvature in its extreme values, which indicates a problem. To uncover the possible reason for the discrepancy between emulator and simulator, the structure of the input/output



(a) (b) (c)

Figure 5.22: Diagnostics for the initial two-input LBM emulation.



(a) (b)

Figure 5.23: LBM input versus output values. The bifurcation and increasing spread in (a) can be seen clearly.

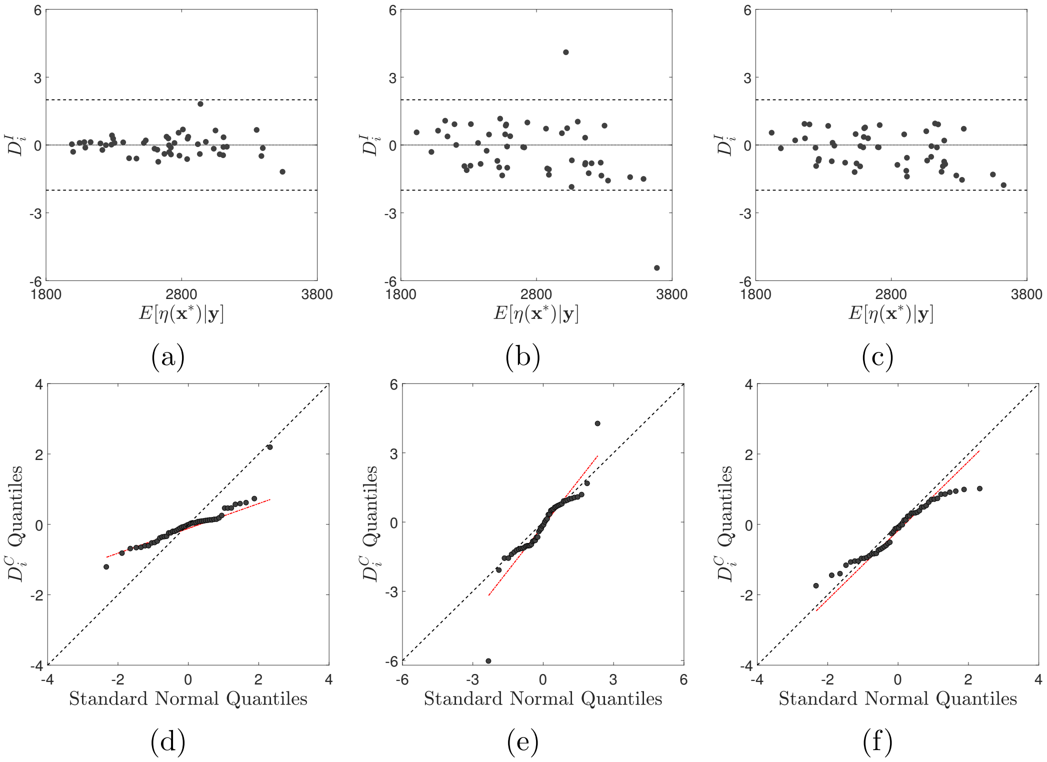
data is examined. Plotting the output values with respect to each input variable, as shown in Figure 5.23 suggests that θ_c does not influence the output as strongly as d_d. Additionally, there is a bifurcating behaviour visible in Figure 5.23(a) for this variable. The structure of the input/output data can be exploited for the specification of global trend for the emulator. As seen in Figure 5.23 there is a roughly linear relationship in the data. However, it could be argued that due to the bifurcation a square trend can also be used. Several different polynomials were tested for $h(\mathbf{x})$. Comparison of D^{MD} values and their CDF in the underlying distribution for different forms of $h(\mathbf{x})$ are given in Table 5.4. The corresponding IPE and QQ plots are shown in Figure 5.24. It becomes apparent that the quality

Table 5.4: Values of mean squared error and Mahalanobis distance for different global trends.

Function	D^{RMSE}	D^{MD}	$F(D^{MD})$
$h(\mathbf{x}) = 1$	143.0	108.6	0.998
$h(\mathbf{x}) = [1, d_d]$	126.8	11.3	0.000
$h(\mathbf{x}) = [1, d_d, \theta_c]$	238.6	98.0	0.992
$h(\mathbf{x}) = [1, \mathbf{x}, d_d^2]$	202.6	28.3	0.037

of emulation is influenced strongly by the choice of global trend. None of the functions solves all problems with the LBM GPE, as confirmed by the marginal improvement in D^{MD}. However, there is a marked difference with respect to the constant trend GPE. Analysing Figure 5.24, it can be seen that the spread of IPE increases, signifying an increasingly accurate estimation of the variability of the output values. This is traded-off for a higher degree of non-linearity in the QQ plots, suggesting the residuals are not normally distributed. Last, but not least, there are some outliers in all QQ diagnostics, which combined with the input/output structure may identify a non-stationary model.

Finally, the fibre diameter, d_f, was added to the model inputs. The emulation was set-up with a linear trend in the first input. The number of training and validation points was kept the same as for the two-input LBM experiment. A few different kernels were compared and it was found that the exponentiated quadratic performed better than the Matérn functions. The added input, increased the dimensionality of the likelihood function used to search for optimal correlation parameter values. The topology of the likelihood proved to be multimodal as evidenced by the range of parameter values identified as optimal by the genetic algorithm. Such issues emphasise one of the main caveats of the use of MLE. A box plot visualising the range of values for the three correlation parameters is shown in Figure 5.25 together with associated diagnostics. In the case of multimodal likelihood resulting in multiple optimal parameter values, the median



(a)

(b)

(c)

(d)

(e)

(f)

Figure 5.24: IPE and QQ plots for the uncorrelated residuals of emulators with different global trends. Panels (a),(d) - linear in d_d; panels (b),(e) - linear in both inputs; panels (c),(f) - quadratic in d_d.

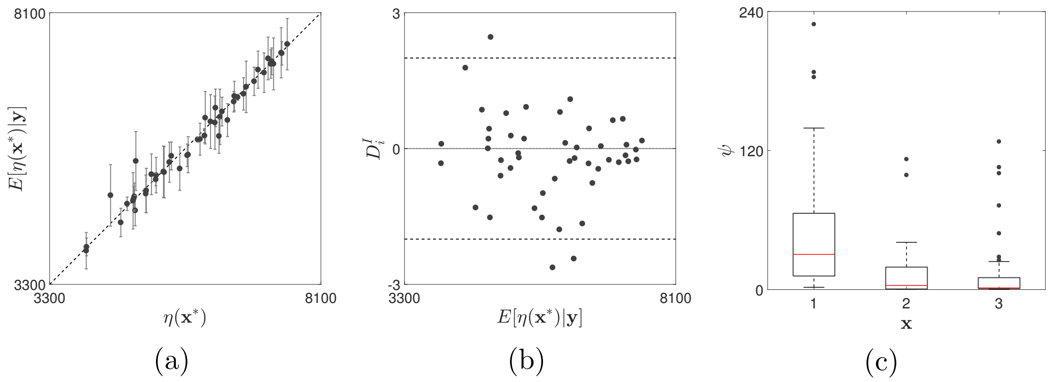
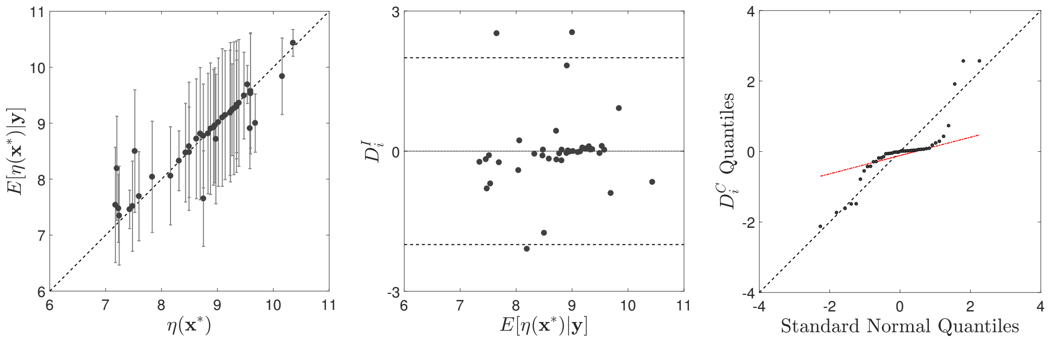


Figure 5.25: Emulator validation for the three-input LBM model. The variation in the optimal correlation parameter values is shown in (c).

of all for each correlation parameter is used as optimal. This resulted in $D^{RMSE} = 151.0$ and $D^{MD} = 54.8$ with $F(D^{MD}) = 0.725$.

5.6.2 Multiple droplet experiments

As described in Section 4.1 and Section 5.2 it is important for the simulator to be able to generate droplets in real time. From uncertainty quantification point of view this means that the GPE needs to be capable of representing the cumulative effect of droplets attaching to the fibre, coalescing and departing to the outlet of the domain. The set-up for the *dynamic* emulation studies was similar to the one described above in that it featured a single fibre and had the volume of water departing the fibre as an output. Each simulation was run until a *true separation* took place, utilising the DBSCAN detection algorithm described in Section 4.3. Dynamic seeding allows smaller droplet diameters to be used, that still lead to separation. The GPE was trained on $n = 50$ and validated $m = 50$ data points. This set of simulations resulted in an output sample with a range in $\mathcal{O}(10^4)$. To reduce the variability of the response it was transformed by taking its natural logarithm. This pre-processing step is widely used in emulation to make the response more amenable to analysis [40]. The corresponding diagnostics are shown on Figure 5.26. All three plots suggest that the predictive variance has



(a) (b) (c)

Figure 5.26: Emulator diagnostics for the three-input LBM model with dynamically generated droplets. The output data is in log scale.

been overestimated. This has likely been caused by the large scale irregularities in the output of the model, compensated by the GPE with an increased measure of uncertainty. The complex shape of the QQ plot for the uncorrelated residuals suggests that the simulator may need to be treated as a non-stationary process.

5.6.3 Emulation for multi-body experiments

The output of the multi-body Lattice Boltzmann simulations can also be viewed as a time series. That is, the change in the median of the effluent side droplet size distribution can be assumed to vary with time with no regard to specific features of the filter. In such a case the GPE can be used to effectively extend the duration of the simulation by providing a means for a look-ahead. For instance, if a particular simulation was run for 5,000,000 time steps, the GPE can provide a prediction of what the output will be at, for example, $t = 8,000,000$.

In order to emulate a time series two changes are introduced to the emulation approach used above. First, due to the more irregular nature of time series, the smooth exponentiated square kernel was substituted for the *exponential* kernel

$$c(\mathbf{x}_i, \mathbf{x}_j|\boldsymbol{\psi}) = \exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|}{\boldsymbol{\psi}} \right) \tag{5.5}$$

The kernel in Eq. 5.5 is the covariance function of the Ornstein-Uhlenbeck process [102], which allows for more irregular functions to be approximated. The second modification concerns the way data is treated by the GPE. A widely used approach to pre-process time data for machine learning applications is the *sliding window* method [141]. In this approach, the input to the prediction model is taken to be the output of the data generation process at the current time. The target that is to be predicted is then the output of the model at the next time step. In the current study the GPE is charged with predicting the next droplet diameter, given the one observed at present. Figure 5.27 shows the diameters of droplets at the outlet of the mesh as a time series. There the solid line is the training data up to $t = 5,000,000$, the dashed line is the remainder of the observed data used for testing the GPE and the dash-dotted line is the prediction itself. It can be seen that the GPE cannot capture the exact shape of the time series, but it eventually manages to match the location of the peaks reasonably well. Importantly the prediction of the GPE result in a outlet distribution median very close to that obtained with the true observations, as shown in Figure 5.28. The prediction quality can be improved by training the GPE with multiple time series from different simulations.

5.7 Chapter summary

This discussion in this chapter presented some of the LBM models developed and used to study the processes of coalescence in the fibrous bed of the filter. It showed how the theory behind lattice Boltzmann models, presented in Chapter 2 can be applied in practice to investigate the filtration mechanisms. The chapter outlined the incremental development of increasingly more complex LBM simulators in scale and computational fidelity. The studies presented here utilised results from the dimensional analysis in Chapter 4 and applied the detection

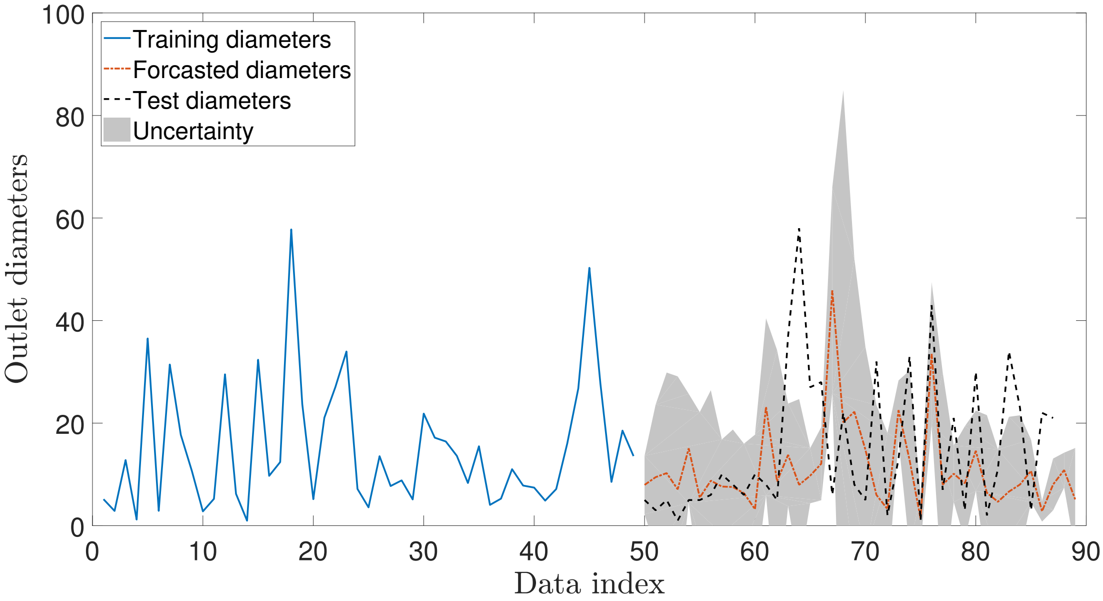


Figure 5.27: Diameters form the outlet droplet size distribution. The solid line in the first half of the time series is the data provided to the GPE for training. The predictions and test observations are shown in as a dash-dotted and dashed line, respectively.

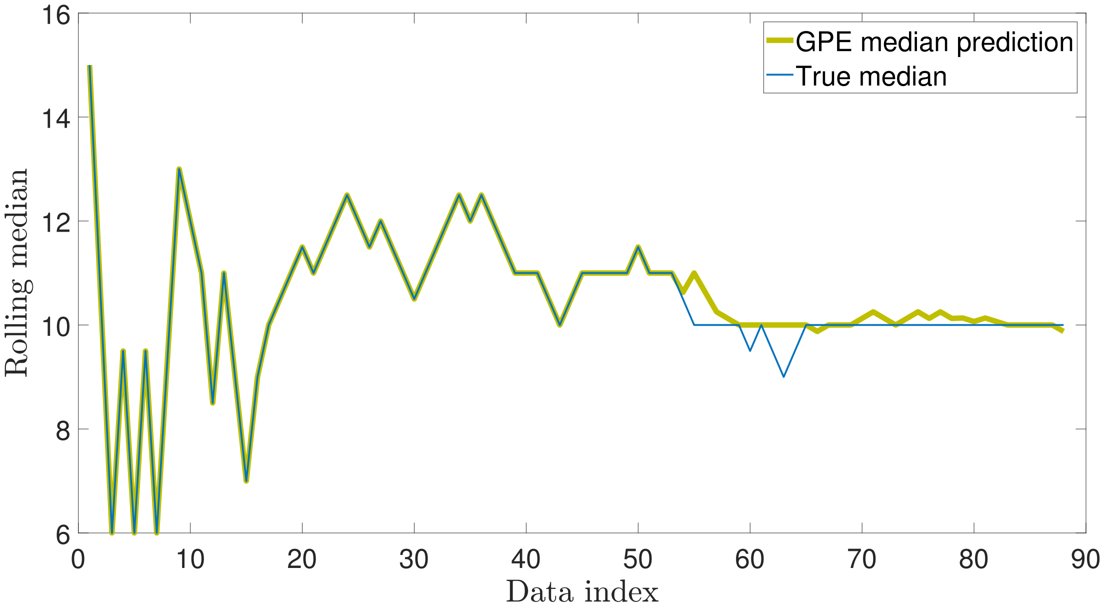


Figure 5.28: Median of the outlet droplet size distribution. The wide line is the GPE prediction, which interpolates the data in the training region. The prediction is close to the observation in the extrapolation region, but exhibits some characteristic oscillations.

algorithm introduced in the same chapter.

The current discussion showed how Gaussian process emulation, introduced in Chapter 3 can be used to reduce the computational cost of the filtration model in two different ways. These developments will be used in Chapter 6 and Chapter 7 to enable sensitivity and reliability analyses, respectively.

An experimental study was used to qualitatively validate the three-dimensional SCMP model. Formal calibration techniques employing GPE can be used to match the computer model to reality. The contributions made in this chapter form the foundation for further work on the project from both academic and industrial perspectives.

CHAPTER 6

Sensitivity analysis with GPE[1]

6.1 Introduction

As explained in preceding chapters, computer code output can be treated as a random variable, despite its deterministic nature. This is due to the fact that the true value of the code's output can be known only when the simulator is run. However, in order to know its value everywhere the code has to be evaluated infinitely many times, which is impossible. Moreover, input parameters are usually associated with a joint probability distribution, $p(\mathbf{x})$ turning them into a multivariate random variable, $\mathbf{X}$. Thus the output, originally $y = \eta(\mathbf{x})$ can be seen as a random variable $Y = \eta(\mathbf{X})$, with its own probability distribution, $p(y)$.

Very often the different inputs do not influence the output equally. Part of the computer-based investigation of the physical process is determining the influence

[1] The ideas developed in this chapter and their application in Section 6.3 have been published as Hristov, P.O., DiazDelaO, F.A., et *al.* (2017), "Probabilistic sensitivity analysis to understand the influence of micromechanical properties of wood on its macroscopic response", *Composite Structures*, **181**, 229–239

of the uncertainty in the different inputs, or sets thereof, on the variability of the output. This process is known as probabilistic *sensitivity analysis* (SA).

There are two main types of sensitivity analyses, namely local (LSA) and global (GSA). LSA is concerned with determining the effect of small, local perturbations in the parameter value around a given base point. A very common local sensitivity approach is the one based on derivatives, namely $\partial Y/\partial X_i$, where X_i is the i^{th} input variable. The local method is not used in the current investigation, since it fails to capture the effect of the inputs when their values are arbitrarily chosen from the entire input domain [142].

In order to fully explore the input space, GSA relies on a number of points carefully spread according to some experimental design. There exists a variety of GSA techniques such as function decomposition in main and higher order effects, regression coefficients and variance-based methods, among others (see for example [142]).

This chapter presents the theory behind GSA and how it can be made more efficient through the use of Gaussian process emulators. The ideas described can be applied to any computer model, since both GSA and GPE are general purpose methods. This fact is demonstrated through the use of three different case studies. First, a sensitivity analysis for the lattice Boltzmann model is presented in Section 6.2. Second, in Section 6.3, the influence of microscopic wood parameters on its macroscopic response is studied via the means of probabilistic sensitivity analysis. Finally, a noise and fuel consumption model of a general passenger aircraft is analysed in Section 6.4. Relevant conclusions are drawn in Section 6.5.

6.1.1 Variance-based sensitivity analysis

Since the models of interest are deterministic, the variance of the output random variable will be entirely due to the uncertainty in the input values. This means that if one could learn the exact, true values of the inputs, the variance of Y would be reduced to 0. This leads to the notion that fixing one of the inputs X_i at a given value x_i and re-running the code will result in Y having a lower variance. Let $\mathbb{V}_{\sim i}[Y|X_i]$ be the conditional variance of Y, taken over all factors, except X_i (denoted $X_{\sim i}$) and given $X_i = x_i$. This conditional variance can be used as a measure of how influential the fixed parameter is. A severe drawback of this measure, however, is its dependence on the location of the point x_i. This problem could be resolved by taking the average of the conditional variance over all possible values of x_i, that is $\mathbb{E}_i[\mathbb{V}_{\sim i}[Y|X_i]]$. It is a well known fact in Probability theory that the variance of a random variable Y can be decomposed as:

$$\mathbb{V}[Y] = \mathbb{E}_i[\mathbb{V}_{\sim i}[Y|X_i]] + \mathbb{V}_i[\mathbb{E}_{\sim i}[Y|X_i]] \tag{6.1}$$

Eq. (6.1) gives another important diagnostic - $\mathbb{V}_i[\mathbb{E}_{\sim i}[Y|X_i]]$ - *the first order effect* of X_i on Y. The associated normalised sensitivity measure, also known as a *Sobol' index* [143] is:

$$S_i = \frac{\mathbb{V}_i[\mathbb{E}_{\sim i}[Y|X_i]]}{\mathbb{V}(Y)} \tag{6.2}$$

The Sobol' index S_i can take values in $[0, 1]$. A high value of the Sobol' index for the given variable, means that it is important, i.e. if it is fixed, there will be a significant reduction in the variance of Y.

Many practical models are of the so-called *non-additive* class. That is, the effect that individual inputs have on the variance cannot be separated to account for 100% of the output variance. Instead, interactions between individual inputs or sets of inputs will play an important role. To capture such effects, the *higher-order Sobol' indices* can be constructed:

$$S_p = \frac{\mathbb{V}_p[\mathbb{E}_{\sim p}[Y|\mathbf{X}_p]]}{\mathbb{V}(Y)} \tag{6.3}$$

where $p \subset \{1, \ldots, d\}$ is a set of indices of all inputs under investigation. It can be shown that:

$$\sum_{i=1}^{d} S_i + \sum_{i<j}^{d} S_{ij} + \sum_{i<j<k}^{d} S_{ijk} + \ldots + S_{12\ldots d} = 1 \tag{6.4}$$

where the summation is carried out over all d dimensions, which means that summing over all Sobol' indices recovers the full variance. Analysis of all main effects of the model inputs and their respective interactions will result in Eq. (6.4) having $2^d - 1$ terms. This means that with relatively low number of inputs, the summation components become too many to investigate individually. To alleviate this issue, Homma and Saltelli [144] introduced the *Total Sobol' index*:

$$S_{T_i} = \left(1 - \frac{\mathbb{V}_{\mathbf{X}_{\sim i}}[\mathbb{E}_{X_i}[Y|\mathbf{X}_{\sim i}]]}{\mathbb{V}(Y)}\right) \tag{6.5}$$

This measure captures the effect of the $i-th$ input and all of its interactions, by fixing all other inputs. Note that $S_i \leq S_{T_i}$, due to interactions between inputs. Equality can only arise in a perfectly additive model. An input is said to be truly non-influential if and only if $S_{T_i} = 0$. In [145] the authors have argued that a good, albeit non-exhaustive characterization of the input influences is given by the set of first order and total Sobol' indices. This is further discussed with the results presented in Section 6.3.

6.1.2 Sobol' indices with GPE

As mentioned previously, the calculation of Sobol' indices requires the evaluation of both conditional and unconditional expectations and variances. These operations are associated with the calculation of a number of integrals. In order to evaluate them, the integrals can be approximated by direct Monte Carlo (DMC) simulation. Since this is the case, a relatively large sample size is required to achieve reasonably accurate estimation results. This is often a problem because, coupled with the computational complexity of most scientific and engineering codes, extensive sampling results in a very costly sensitivity analyses. Using GPEs as inexpensive approximations to the output of the code, together with the use of parallel computers, enables DMC based analyses to be performed within reasonable time periods. The unconditional variance of the simulator output can be written as:

$$\mathbb{V}[Y] = \mathbb{E}[Y^2] - \mathbb{E}[Y]^2 \tag{6.6}$$

When using the emulator, the simulator output Y in Eq. (6.6) is substituted with the posterior mean of the emulator $\mathbb{E}[\eta(\mathbf{X})|\mathcal{D}]$ (Eq. (3.12)). Then, the Monte Carlo approximations of the terms in Eq. (6.6) are given by:

$$\hat{\mathbb{E}}[Y] = \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}[\eta(\mathbf{x}^{(n)})|\mathcal{D}] \tag{6.7}$$

$$\hat{\mathbb{V}}[Y] = \frac{1}{N-1}\sum_{n=1}^{N}\mathbb{E}[\eta(\mathbf{x}^{(n)})|\mathcal{D}]^2 - \hat{\mathbb{E}}[Y]^2 \tag{6.8}$$

Here, only the estimators for the first-order and the *total* conditional variances are given, since any higher-order variances could be calculated from their definitions in Eq. (6.2) and in Eq. (6.5).

$$\hat{\mathbb{V}}[\mathbb{E}[Y|X_i]] = \frac{1}{N-1}\sum_{n=1}^{N}\mathbb{E}[\eta(x_i^{(n)}, \mathbf{x_{\sim i}}^{(n)})|\mathcal{D}]\mathbb{E}[\eta(x_i^{(n)}, \mathbf{x}'_{\sim\mathbf{i}}{}^{(n)})|\mathcal{D}] - \hat{\mathbb{E}}[Y]^2 \tag{6.9}$$

$$\hat{\mathbb{V}}[\mathbb{E}[Y|\mathbf{X}_{\sim i}]] = \frac{1}{N-1}\sum_{n=1}^{N}\mathbb{E}[\eta(x_i^{(n)}, \mathbf{x}'_{\sim\mathbf{i}}{}^{(n)})|\mathcal{D}]\mathbb{E}[\eta(x_i'^{(n)}, \mathbf{x}'_{\sim\mathbf{i}}{}^{(n)})|\mathcal{D}] - \hat{\mathbb{E}}[Y]^2 \tag{6.10}$$

where $\mathbf{x}$ and $\mathbf{x}'$ come from two distinct sets of values for $\mathbf{X}$, each of size $N \times d$. The full algorithm for the calculation of the first order and total Sobol' indices is given in [146]. In some instances, analytical expressions from the GPE are available for all quantities of interest [147], but these rely on some modelling assumptions and hence sampling the posterior of the GPE directly is preferred. Another approach is the one described in [148], based on constructing a GPE using an analysis-of-variance kernel, which results in decomposing the predictive mean in a sum of effects for each input.

6.2 Three-input lattice Boltzmann model

This section presents results from a sensitivity analysis with the three-dimensional LBM model, introduced in Chapter 5 and emulated in Chapter 3. The reasons for performing the analysis as outlined so far in this chapter, are two. Firstly, due to its relatively small set of input variables, the model allows the importance of the inputs as indicated by the Sobol' indices to be checked for physical meaning. Secondly, even though this version of the code has only three inputs, it is still computationally expensive and a direct Monte Carlo-based analysis would be too costly to perform. Finally, a full set of Sobol' indices for the problem results in only seven effects, which allows all interactions to be visualised and studied individually. The following results were obtained based on 15 000 samples from the distributions shown in Table 5.1. Figure 6.1 shows the first order and total Sobol' indices for the LBM model. The diameter of the droplet is ranked most important, being responsible for 93% and 99% of the variability of the input for S_i and S_{T_i}, respectively. Inspecting Figure 6.1(b), an interesting observation can be made about the other two inputs. Despite the fact that their absolute values do not change significantly, the relative change in the importance of the droplet diameter and fibre contact angle is noticeable. The two inputs exhibit approximately 317% and 1643% increase, for d_f and θ_c, respectively. These figures warrant a closer look at the interactions between the inputs. Higher order Sobol' indices can be obtained through a procedure similar to the one described in

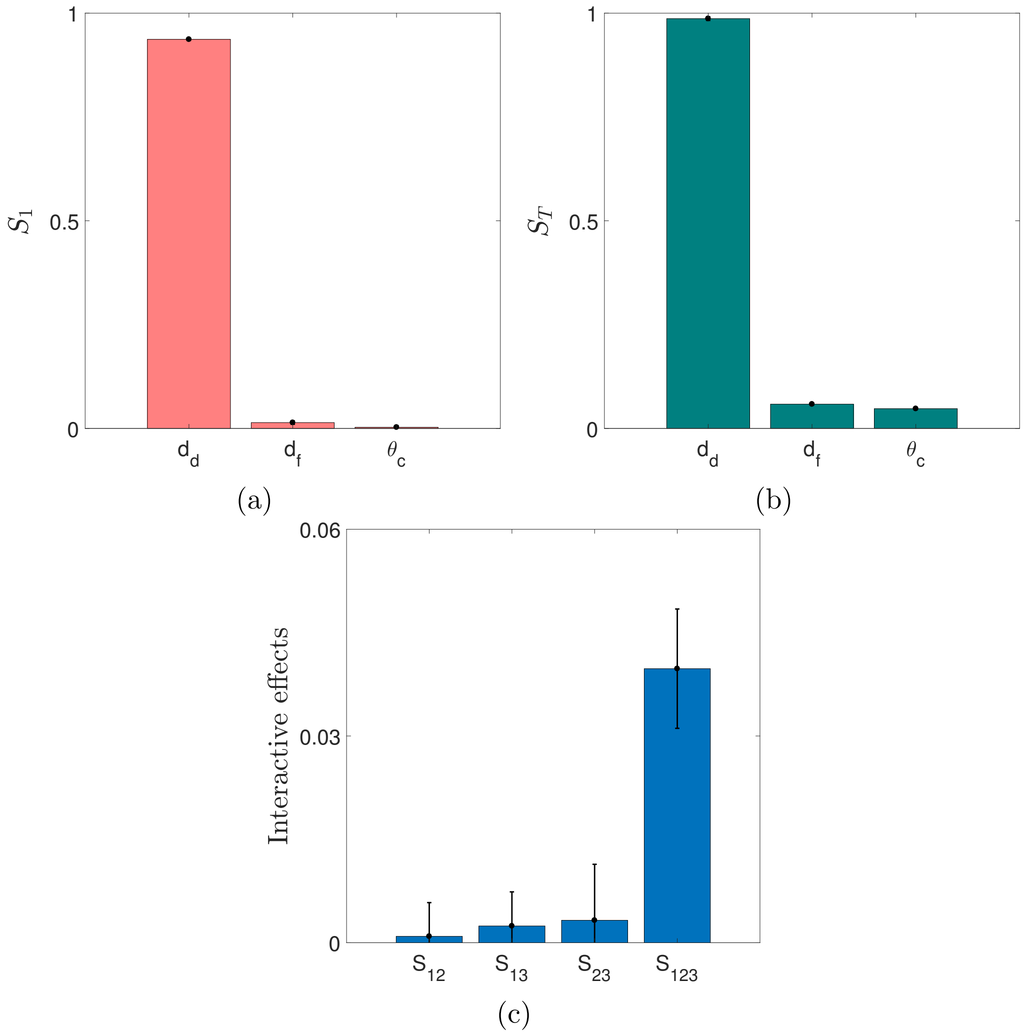
Section 6.1 for the first order effects. Once the conditional variance, defined as

$$V_{ij} = \mathbb{V}_{ij}[\mathbb{E}_{\sim ij}[Y|X_i, X_j]], \text{ for } i \neq j \tag{6.11}$$

is calculated, the second order effect is given by

$$S_{ij} = \frac{V_{ij}}{\mathbb{V}[Y]} - S_i - S_j, \text{ for } i \neq j \tag{6.12}$$

The highest order index, in this case S_{123} is given by the relationship in Eq. 6.4. Figure 6.1(c) shows the three second order indices, S_{12}, S_{13}, S_{23}, and the third order index S_{123}. To simplify the notation, the indices for each input were set as 1 - d_d, 2 - d_f, 3 - θ_c, thus for example, S_{13} corresponds to the interaction between d_d and θ_c. In Figure 6.1(c) the interaction between all three inputs is appreciable meaning that the variability of the output cannot be attributed to any single input, even though the droplet diameter has the largest quantitative effect. This fact should not come as a surprise, because the inputs of the model were selected to represent the most influential parameters in the filtration set-up and as such their influence on the model output was expected to show coupling. For instance, examining S_{123} it could be reasoned that a larger droplet will be held more strongly to a more hydrophilic fibre with larger surface area. In terms of modelling decisions, the results suggest that the model can not be simplified any further without risking the introduction of a bias in the output. Finally, in all three sub-figures the error bars indicate ±2 predictive standard deviations from the GPE. In Figure 6.1(a) and Figure 6.1(b) the errors are too small to be seen due to the scale of the Sobol' indices bars.



(a)

(b)

(c)

Figure 6.1: Sensitivity indices for the LBM set-up. (a) First order; (b) total; (c) higher order. The notation was simplified by setting 1 - d_d, 2 - d_f 3 - θ_c in (c). The error bars indicate ±2 standard deviations in the results, obtained from sampling the GP predictive distribution 1000 times.

6.3 Analysis of microscopic wood properties

The following analysis was carried out whilst the LBM code was being developed. The reason for this was to develop a better understanding of the theoretical and practical aspects of sensitivity analysis. The included results, which demonstrate the applicability of SA to larger models, were published as a journal article.

6.3.1 Problem description

In recent years, considerable attention has been paid to the investigation of wood at multiple length scales. These efforts are owed to the fact that wood exhibits many good properties, desirable in the building industry. In particular, cross-laminated timber (CLT) panels, shown in Figure 6.2 have been under special consideration.

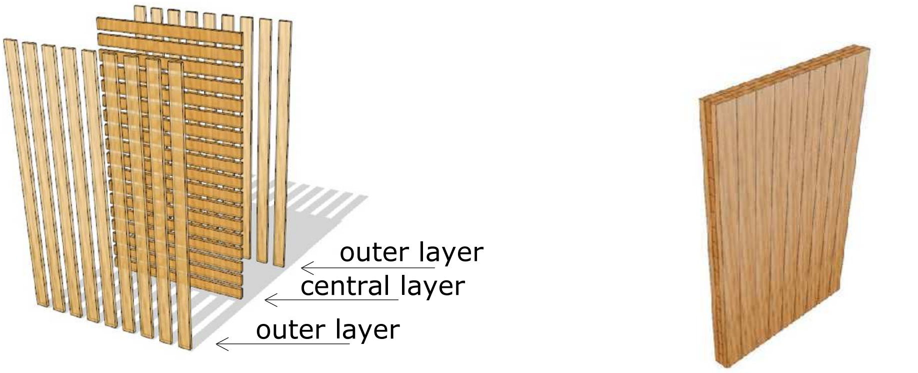
In softwoods, the typical material of choice in CLT fabrication, wood fibres can be divided into *early-wood* and *late-wood*. The *early-wood* fibres are characterised by large diameters and thin cell-walls, whereas *late-wood* fibres are composed of narrow diameters with much thicker cell-walls. At the scale of a few millimeters wood is represented by the growth rings, typically found in the cross-section cut through the trunk of a tree. Wood shows a highly hierarchical structure across length scales. For the purposes of this study, thirteen parameters in three scales were considered. A summary of the parameters and their ranges is given in Table 6.1.

6.3.2 Multi-scale modelling

A computational homogenisation approach is adopted to capture the hierarchical nature of wood at different length scales. The general procedure consists of building a material model for wood by homogenising the three material scales described in the previous section (at the level of the microfibril, wood fibres and

Table 6.1: A summary of the parameters considered in the study. The scale and definition for each parameter are also provided. E-W and L-W stand for early-wood and late-wood, respectively.

Notation	Name	Scale	Range	Unit
f_{cc}	Cellulose crystallinity	nm	[45.0 – 60.0]	%
f_h	Fraction of hemicellulose	nm	[25.0 – 29.0]	%
f_c	Fraction of cellulose	nm	[30.0 – 50.0]	%
MFA	Microfibril angle	nm	[0.0 – 22.0]	deg
L_{cc}	Length of crystalline	nm	[26.5 – 36.4]	nm
T_p	E-W tangential dimension	μm	[28.0 – 30.0]	μm
R_p	E-W radial dimension	μm	[37.0 – 40.0]	μm
t_{cp}	E-W thickness	μm	[3.1 – 4.3]	μm
T_v	L-W tangential dimension	μm	[25.0 – 27.0]	μm
R_v	L-W radial dimension	μm	[31.0 – 37.0]	μm
t_{cv}	L-W thickness	μm	[4.3 – 8.0]	μm
θ	Cell wall angle	μm	[10.0 – 27.5]	deg
P_{ew}	E-W fraction in growth ring	mm	[67.0 – 80.0]	%



(a) Stacked layers.

(b) Glued layers.

Figure 6.2: Schematic representation of a CLT panel [149].

growth rings).

Multi-scale models enable specifying the relationships between physical variables observed at different length scales. These are of particular importance in the study of heterogeneous materials with hierarchical microstructures in which the macroscopic response of the material can be predicted from the information coming from the microscopic (or lower) level.

This multiscale methodology has proven to be successful in reproducing the mechanical behaviour of materials at several length scales.

Each spatial scale was modelled using a finite element model with meshes depicted in Figure 6.3. A typical finite element mesh of the representative volume element (RVE) chosen to describe the mechanical response to loading of the growth ring is shown in Figure 6.3(d). The turquoise colour represents the portion of material calculated by the computational homogenisation of the *early-wood* RVE shown in Figure 6.3(c), whereas the light brown colour shows the material obtained by the homogenisation of the *late-wood* RVE shown in Figure 6.3(b). The periodic repetition of the growth rings forms the base material for the macroscopic or structural scale (in this case, the CLT panels). This scale is modelled using the finite element mesh depicted in Figure 6.3(e) and 6.3(f). The computational homogenisation procedure is implemented in the commercial software ANSYS [150]. Set up in this way, the model can predict the response of any (macroscopic) timber structure (that is, the structural scale). For further details on the present homogenisation approach, the reader is referred to Saavedra Flores et *al.* [151].

This study, focuses on two types of structural configurations. First, a timber plate, shown in Figure 6.3 (e) is analysed. The plate is subject to four-point bending along the length and width of the panel. These analyses provide the longitudinal and transverse Young's moduli for the wood, E_0 and E_{90}, respectively. Second, a CLT plate, depicted in Figure 6.3(f) is analysed. The CLT plate is subject to three-point bending along the strong direction, in-plane shear loading and compression parallel to wood fibres in the outer layers. These experiments provide the bending stiffness K_{bend}, the in-plane shear stiffness K_{sh} and the axial stiffness K_{comp} of the CLT plate. In addition, the macroscopic density of the material ρ, is computed. The above six macroscopic parameters (ρ, E_0, E_{90}, K_{bend},

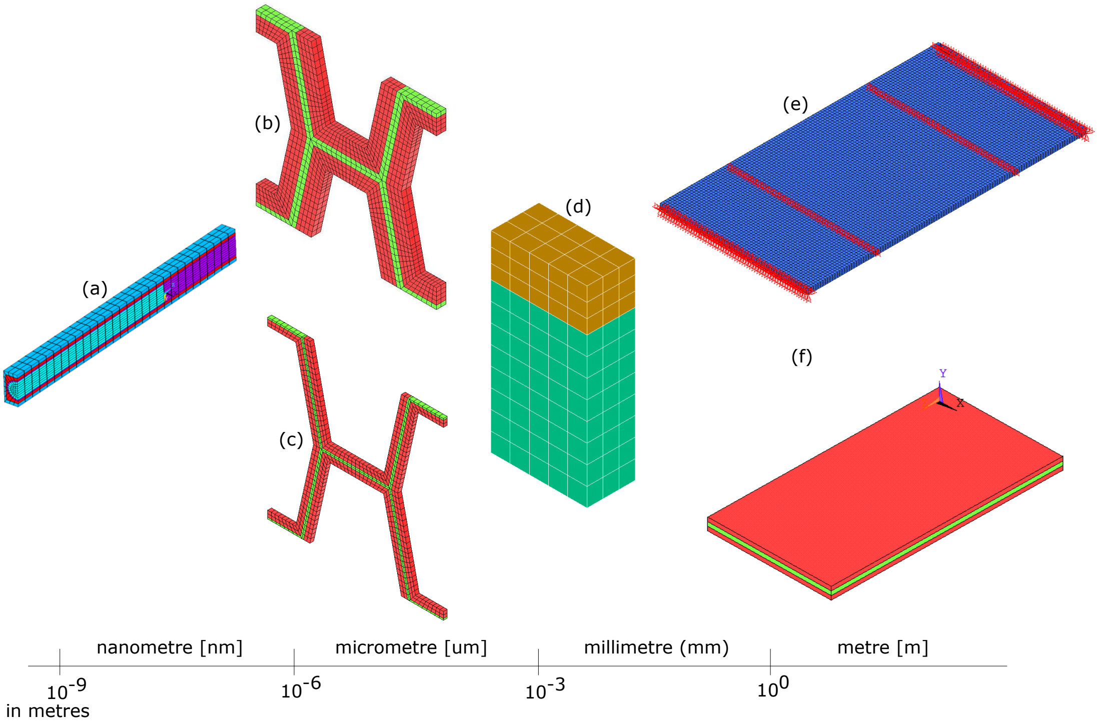


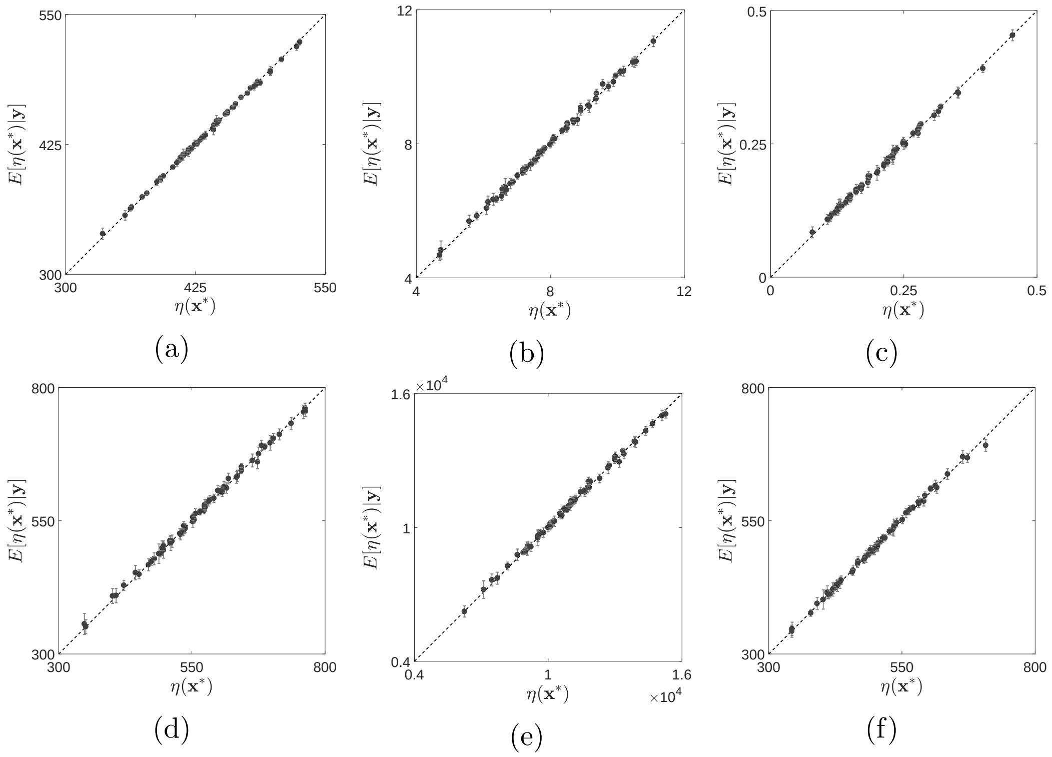
Figure 6.3: Finite element meshes of the RVEs and structures analysed in this paper along with their corresponding length scales. (a) RVE associated with the *microfibril* scale (for the sake of clarity, only one half of the RVE is shown here); (b) RVE associated with the modelling of *late-wood* fibres; (c) RVE associated with *early-wood* fibres; (d) growth ring RVE; (e) 4-cm-thick layer subject to four-point bending; (f) 12-cm-thick CLT panel (consisting of three 4-cm-thick layers). For the sake of clarity, the FE mesh has been hidden. Figure adapted from [149].

K_{comp} and K_{sh}) are selected because of their relevance in the day-to-day practice of analysis and design of timber structures, particularly in the context of CLT structures.

The micromechanical properties of wood can be uncertain due to the lack of knowledge or because of measurement errors at such small length scales. As these properties are crucial to develop reliable predictive models, the uncertainty in their values must be taken into account. The sensitivity analyses presented in this section is part of the systematic uncertainty quantification effort for multiscale wood models, began in Saavedra Flores et al. [152, 153].

6.3.3 Micro-macro sensitivity analysis

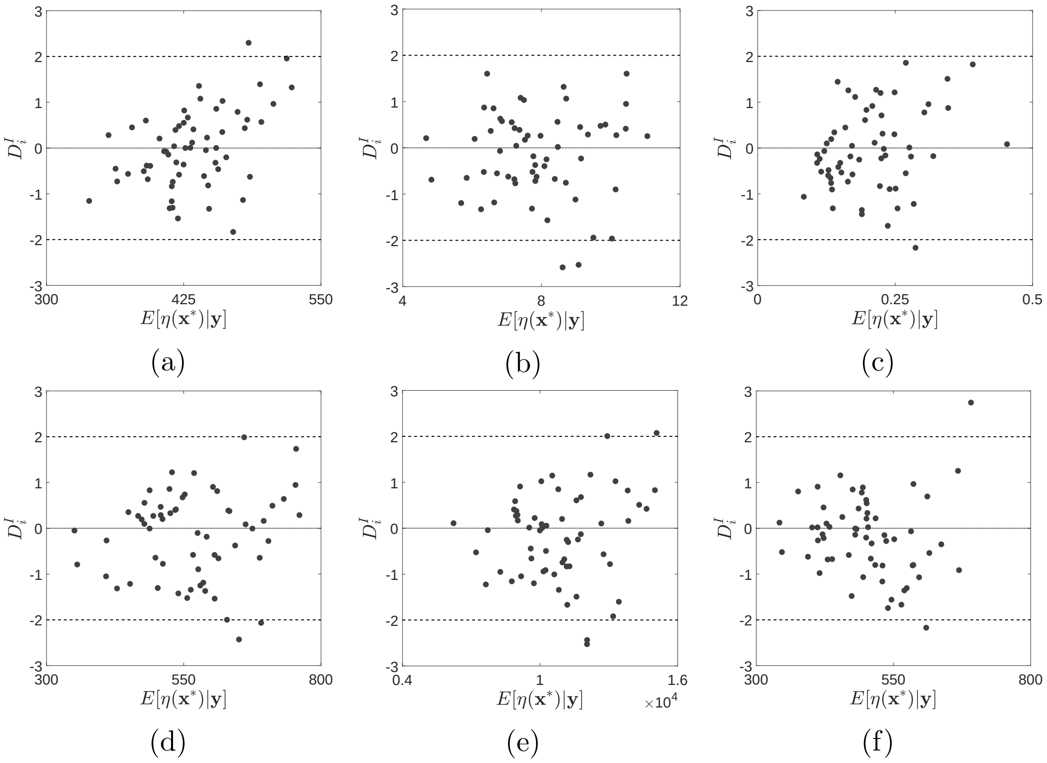
The investigation of relations between the 13 microscopic properties and the 6 macromechanical parameters described in the previous section is termed *micro-macro* analysis. The 6 macro parameters are analysed independently by fitting one Gaussian process per parameter. Therefore, the black-box function is of the form $M_j = \eta(m_1, \ldots, m_{13})$, where the M_j is the j^{th} macro parameter and $m_1, \ldots, m_{13}$ are the micro parameters. Since the multiscale FE model is expensive, it should only be run as many times as necessary. Following the discussion in Chapter 3, it is common to use a sample of size ten times the dimensionality of the input, to train the GPE. Therefore, 130 uniformly distributed points were selected via a Latin hypercube sampling (LHS). Another 60 LHS points were chosen as a validation set to check the quality of the GPE. Thus the code was run a total of 190 times to supply data for the analysis. Table 6.1 gives the ranges of each micromechanical parameter. The values were constrained in order to match available experimental data. The properties are assumed to be stochastically distributed as uniform random variables because they are susceptible to considerable variations when measured experimentally [151]. To validate the GPE, individual prediction errors (IPE), introduced in Section 3.3.5 are used. Figure 6.4 shows the validation results for all 6 macro parameters. It can be seen that there is a close correspondence between predictions and observations. Each point also displays the 95% credible interval, which is based on the posterior predictive variance Eq. 3.13. The individual prediction errors plotted against prediction values are shown in Figure 6.5. All but a few of the errors lie within the desired boundaries, which together with the plots in Figure 6.4 suggest that the emulator is a valid representation of the simulator. A convergence study was carried out to determine a suitable sample size for sensitivity analysis. Figure 6.6



(a) (b) (c)

(d) (e) (f)

Figure 6.4: Simulated versus emulated values. The prediction is given by the posterior mean and the 95% credible interval (error bars) is given by the posterior variance of the emulator.



(a)

(b)

(c)

(d)

(e)

(f)

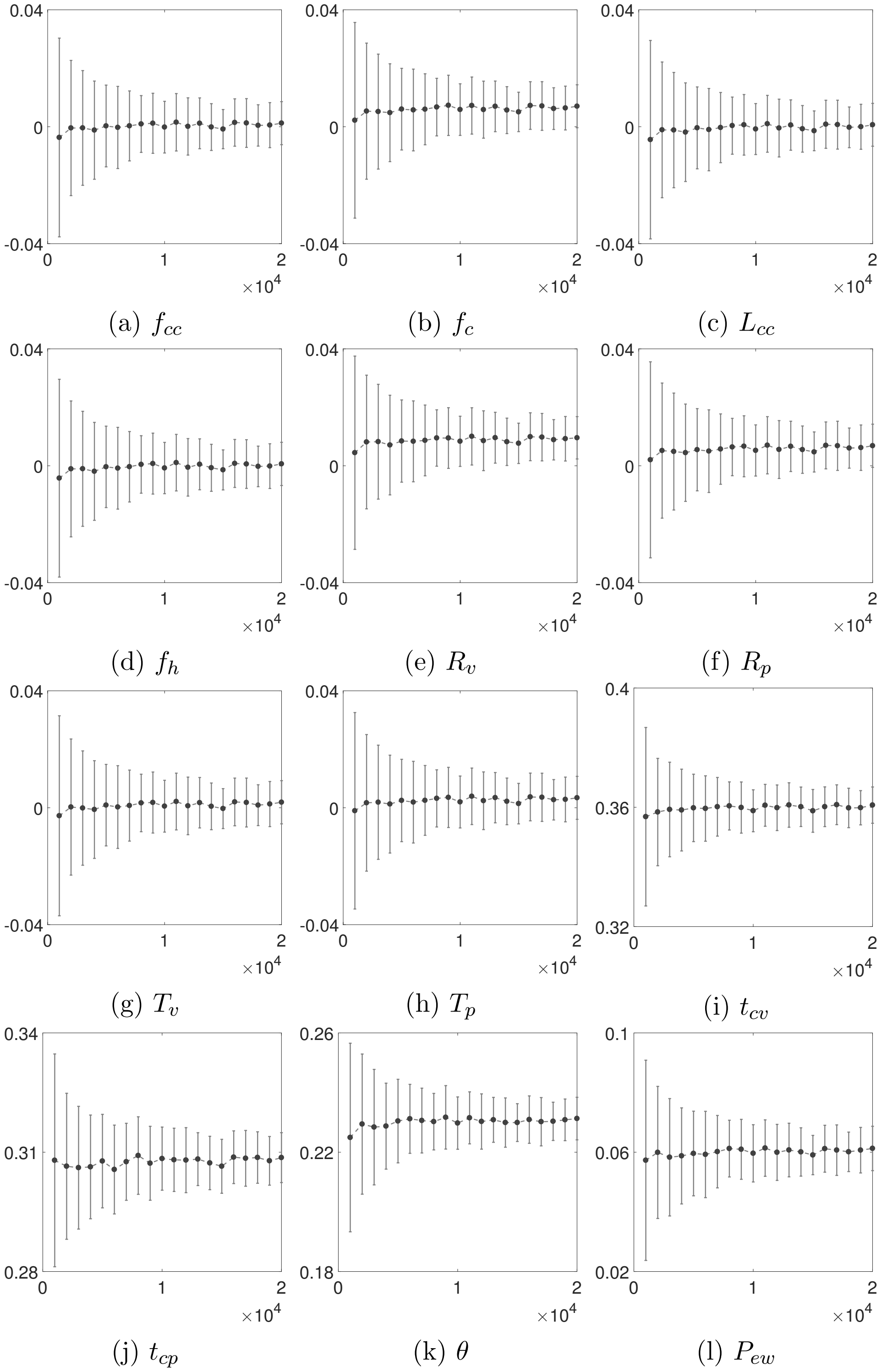
Figure 6.5: Individual prediction errors for all six macro parameters. The emulator value has a *Student-t* posterior distribution and the errors should lie within the interval $[-2, 2]$ with 95% confidence.

shows the mean and one standard deviation of the first order indices for density. The appropriate sample size is selected as the one after which there is no appreciable change in the index' standard deviation. In this case, $N = 10\,000$ points per variable were sampled. Inspecting Figure 6.7, which shows the same study based on the total indices, confirms the validity of the choice. Figure 6.8 shows a set of bar graphs representing the first order and total Sobol' indices of the 13 micromechanical parameters for density, longitudinal and transverse Young's modulus, in rows (a) - (c), respectively. Figure 6.9 follows the same logic and depicts the Sobol' index ranking for bending, compression and shear stiffness in rows (a) - (c), respectively.

The indices represent a ratio of variances so they can not (in theory) be negative. However some estimated values that are close to 0 are negative, due to the fact that all integrals are estimated using sums (see Eq. (6.9)). It is useful therefore to have a measure of confidence in the estimations. The fact that the GPE is only an approximation to the output of the real code can be accounted for by sampling the whole posterior distribution, instead of just its mean. The error bars on Figures 6.8 and 6.9 depict ±2 sample standard deviations obtained from sampling the emulator 1000 times. These measures give a 95% confidence interval for the indices and reflect the validity in the predictions from the GPE shown in Figures 6.4 and 6.5

6.3.4 Discussion

In general, the same trends in the first-order and total Sobol' indices are observed in both Figures 6.8 and 6.9, which represents weak interaction among parameters. Interaction among input variables is indicated as the relative increase in the total Sobol' indices as compared to the first order terms. It is noted that this increase quantifies that part of the response variability which cannot be written off as a simple superposition of input effects. Weak interactions is not to say



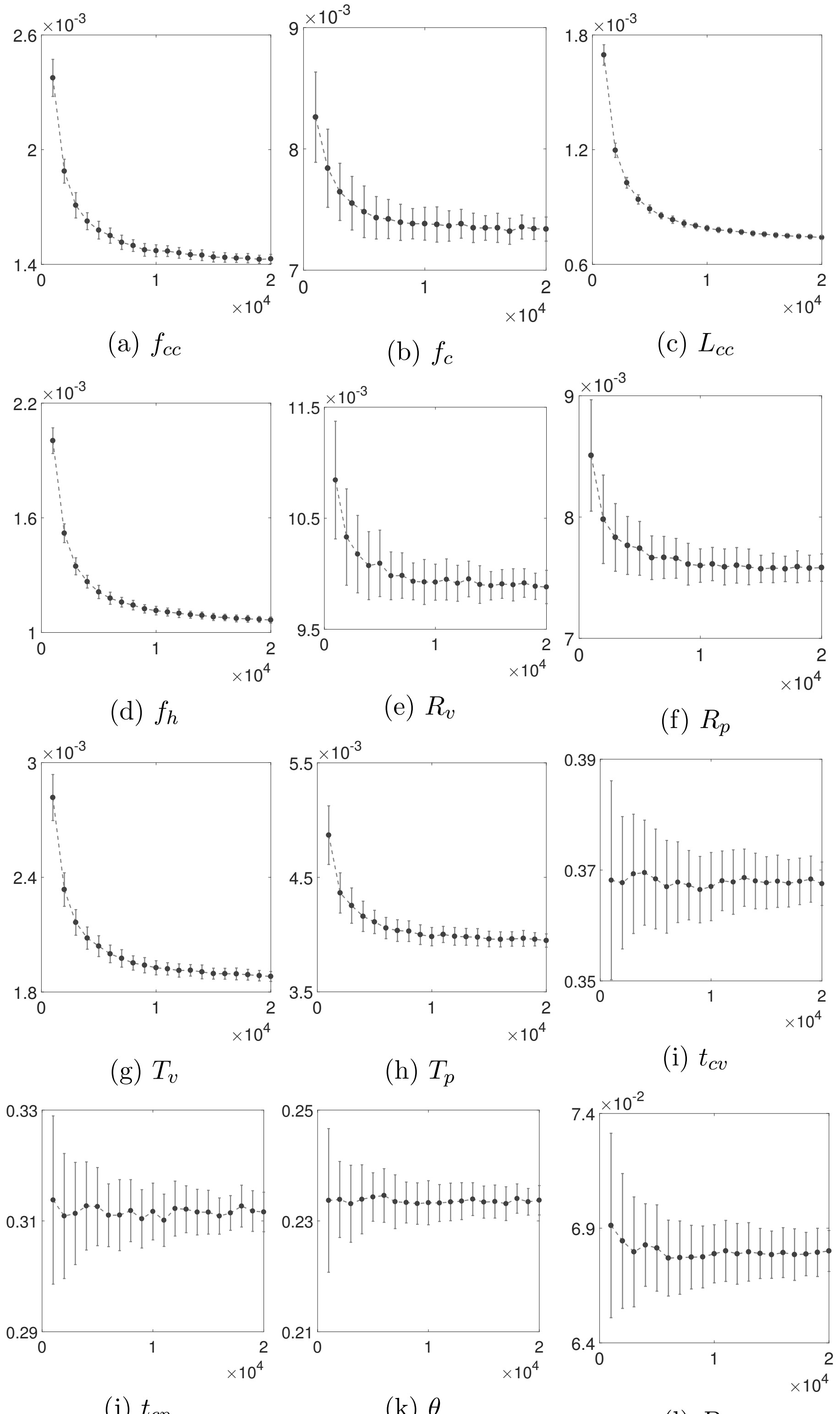
(a) f_{cc} (b) f_c (c) L_{cc}

(d) f_h (e) R_v (f) R_p

(g) T_v (h) T_p (i) t_{cv}

(j) t_{cp} (k) θ (l) P_{ew}

Figure 6.6: Convergence study for the first order parameter effects on density. Line and error bars are mean and ±1 standard deviation of the indices distribution estimated by sampling the emulator mean.



(a) f_{cc}

(b) f_c

(c) L_{cc}

(d) f_h

(e) R_v

(f) R_p

(g) T_v

(h) T_p

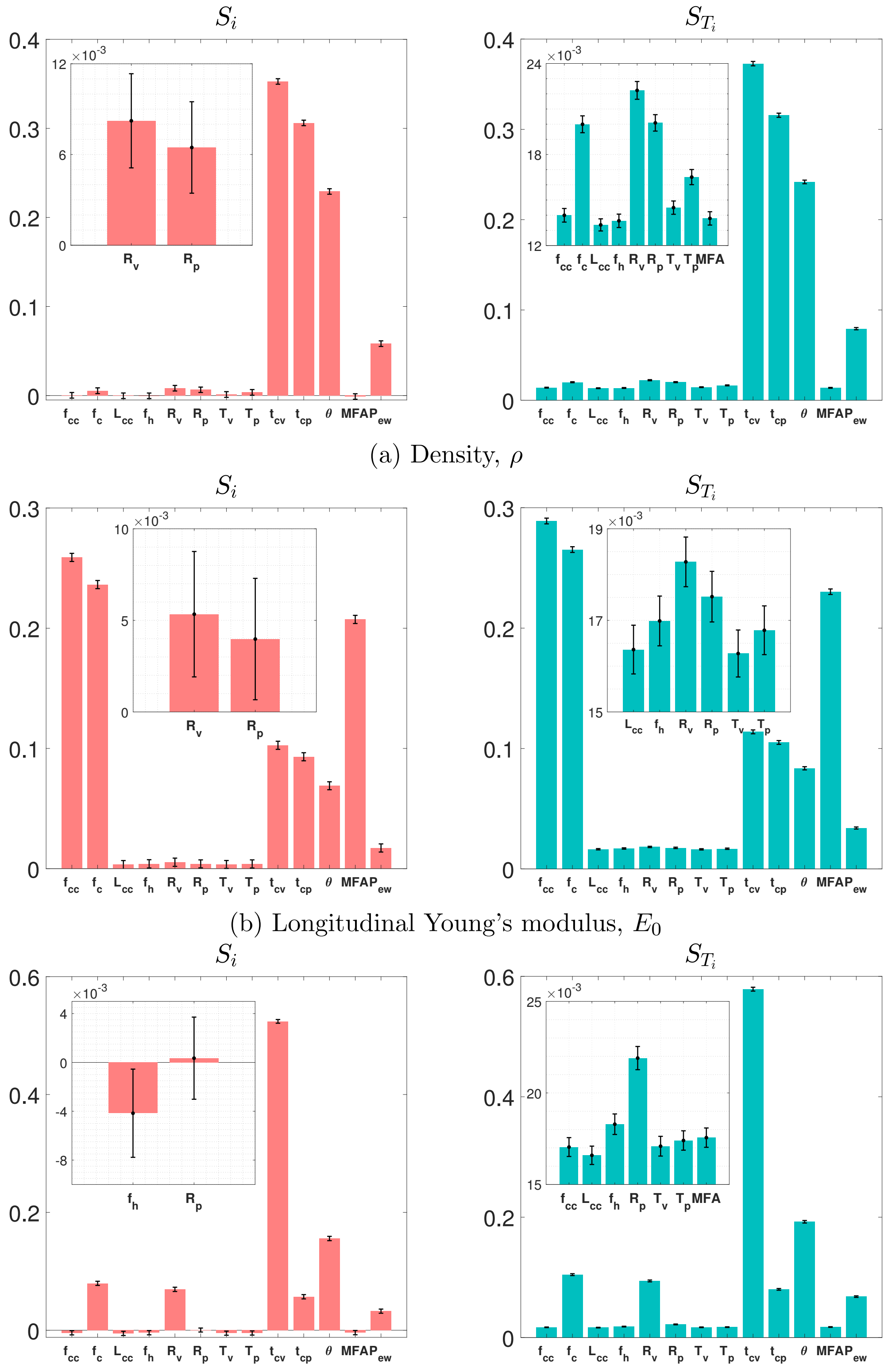
(i) t_{cv}

(j) t_{cp}

(k) θ

(l) P_{ew}

Figure 6.7: Convergence study for the total parameter effects on density. Line and error bars are mean and ±1 standard deviation of the indices distribution estimated by sampling the emulator mean. All non-influential values are overestimated at small sample sizes.

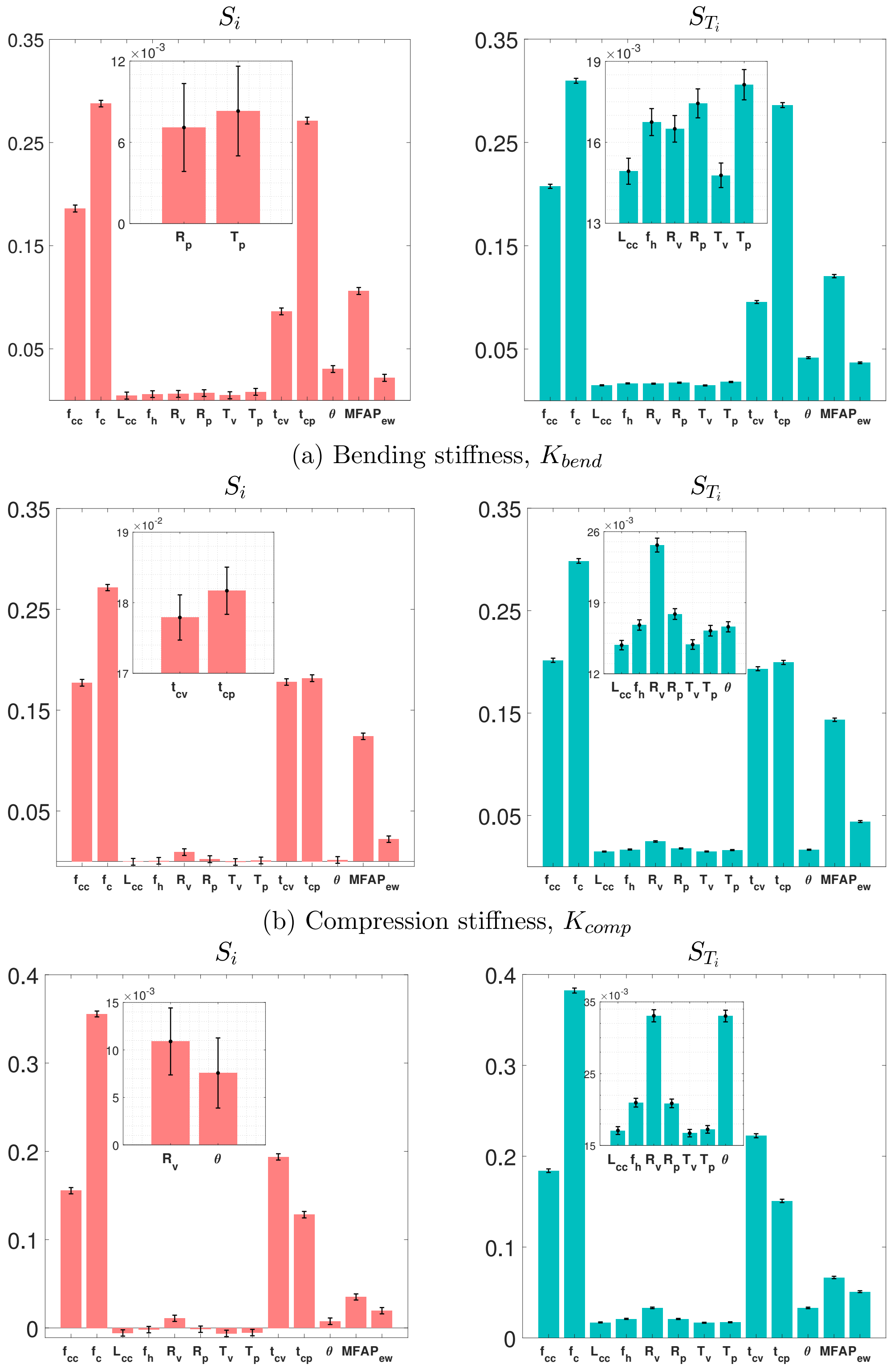


(a) Density, ρ

(b) Longitudinal Young's modulus, E_0

(c) Transverse Young's modulus, E_{90}

Figure 6.8: First and total Sobol' indices for density, longitudinal and transverse Young's moduli. Error bars show ±2 standard deviations of the indices obtained from the GP posterior. The insets show magnification of some sets of indices which could change importance due to errors.



(a) Bending stiffness, K_{bend}

(b) Compression stiffness, K_{comp}

(c) Shear stiffness, K_{sh}

Figure 6.9: First and total Sobol' indices for bending, compression and shear stiffness. Error bars show ±2 standard deviations of the indices obtained from the GP posterior. The insets show magnification of some sets of indices which could change importance due to errors.

that their relative magnitude with respect to the corresponding first order effect is small, but rather that there are no major changes in the ordering of the inputs by importance. The insets in each figure show a magnification of those indices which can change order in the overall importance ranking due to their quantified uncertainty. Most of the affected parameters have relatively low Sobol' indices and are thus simply a demonstrator of the fact that sensitivity ranking is a probabilistic measure and should not be taken to have a fixed numerical value. In general, parameters that were identified as important kept their positions after the inclusion of uncertainty. For the sake of clarity only two first order Sobol' indices are show in the inset. On the other hand all total indices whose error bars could not be clearly distinguished are shown in the insets in the left column of Figures 6.8 and 6.9. It is worth mentioning that the results presented in Section 4 depend on the assumed parametric distribution (see Table 6.1). This however, does not hold for the general methodology, which is independent of the modelling assumptions and can be applied to a wide variety of problems, as demonstrated in this chapter.

Even though the strong influence of the cellulose content on the overall stiffness was expected, neither the difference between the influence of the cellulose content on the shear deformation mode on the bending and compression behaviour, nor its numerical quantification, were evident. This represents the main justification of carrying out the present sensitivity analysis.

Some parameters have first and total Sobol' indices close to 0. The relevance of identifying such non-influencing parameters is that they can be removed from the modelling process in order to develop simpler and much more efficient models. A summary of the Sobol' indices is given in Table 6.2 and Table 6.3.

Table 6.2: First order sensitivity indices for all six macro parameters based on 1000 samples of the GPE distribution.

Parameter	ρ	E_0	E_{90}	K_{bend}	K_{comp}	K_{sh}
f_{cc}	0.0003	0.2589	-0.0044	0.1860	0.1771	0.1554
f_c	0.0054	0.2363	0.0796	0.2878	0.2716	0.3556
L_{cc}	-0.0003	0.0033	-0.0054	0.0047	-0.0004	-0.0057
f_h	-0.0003	0.0040	-0.0042	0.0061	0.0006	-0.0019
R_v	0.0082	0.0053	0.0694	0.0063	0.0092	0.0109
R_p	0.0065	0.0040	0.0003	0.0071	0.0022	-0.0015
T_v	0.0011	0.0035	-0.0050	0.0050	-0.0005	-0.0062
T_p	0.0035	0.0039	-0.0047	0.0083	0.0009	-0.0053
t_{cv}	0.3527	0.1026	0.5241	0.0863	0.1779	0.1937
t_{cp}	0.3061	0.0931	0.0568	0.2713	0.1817	0.1283
θ	0.2292	0.0690	0.1557	0.0304	0.0014	0.0076
MFA	-0.0011	0.2073	-0.0040	0.1061	0.1240	0.0351
P_{ew}	0.0583	0.0172	0.0324	0.0219	0.0220	0.0195

Table 6.3: Total sensitivity indices for all six macro parameters based on 1000 samples of the GPE distribution.

Parameter	ρ	E_0	E_{90}	K_{bend}	K_{comp}	K_{sh}
f_{cc}	0.0140	0.2890	0.0170	0.2075	0.2015	0.1840
f_c	0.0200	0.2652	0.1046	0.3097	0.2985	0.3823
L_{cc}	0.0134	0.0164	0.0166	0.0149	0.0148	0.0171
f_h	0.0136	0.0170	0.0183	0.0167	0.0168	0.0210
R_v	0.0222	0.0183	0.0941	0.0165	0.0247	0.0331
R_p	0.0201	0.0175	0.0219	0.0174	0.0179	0.0209
T_v	0.0145	0.0163	0.0171	0.0148	0.0149	0.0167
T_p	0.0165	0.0168	0.0174	0.0181	0.0162	0.0172
t_{cv}	0.3728	0.1139	0.5787	0.0955	0.1933	0.2225
t_{cp}	0.3156	0.1050	0.0801	0.2861	0.1994	0.1508
θ	0.2418	0.0836	0.1924	0.0416	0.0166	0.0330
MFA	0.0138	0.2302	0.0176	0.1206	0.1435	0.0666
P_{ew}	0.0793	0.0340	0.0681	0.0367	0.0441	0.0509

6.4 Aircraft climb-cruise engine matching

The case study in this section was presented as a problem by Airbus UK, at the Uncertainty Quantification and Management study group with industry, organized by Innovate UK. Results from this case study were included in the dissertation as they allowed sensitivity analysis results to be verified against expert industrial knowledge. The use of sensitivity analysis for the climb-cruise engine matching problem helped demonstrate the applicability of methods and tools developed by the author, to industrial scale challenges.

6.4.1 Problem description

Modern aircraft are expected to operate within very stringent performance and regulatory limits. They generate noise, heat, gaseous emissions and other by-products that severely affect the environment [154]. Aircraft sound is generated by two main sources, namely engines and aerodynamic interactions. In addition to noise limitations, increasingly demanding gaseous emission regulations are coming into effect worldwide. These impose bounds on the amount of greenhouse gases produced by an aircraft. The source of such gases is the power train, but their generation depends on a collection of factors beyond those defining the engine itself. Finally, for an aircraft to be profitable to its user, it needs to have as low a fuel consumption as possible. These demands necessitate a highly structured approach to aircraft design, acknowledging the complex nature of interactions and dependencies between different systems. For the purposes of this study, the aircraft is defined as a combination of different wings and engines, which are in turn defined by the parameters given in Table 6.4. Four *figures of merit* to the performance of the aircraft are considered. These are sideline noise, flyover noise, NO_X emissions and block fuel.

Whilst the modelling process is multi-organisational and multi-disciplinary, for the purposes of this study, it has been simplified to a chain of coupled analyses implemented in a tool called AirCADia [155].

Table 6.4: Inputs and outputs for the environmental case study, with respective parent system and ranges

Inputs				
Notation	**Name**	**System**	**Range**	**Unit**
S_w	Wing area	Airframe	$[1300, 1400]$	ft^2
A	Aspect ratio	Airframe	$[9, 11]$	-
SLST	Static thrust	Engine	$[26000, 32000]$	
FPR	Fan pressure ratio	Engine	$[1.5, 1.8]$	
OPR	Overall pressure ratio	Engine	$[30, 40]$	
BPR	Bypass ratio	Engine	$[6, 8]$	-

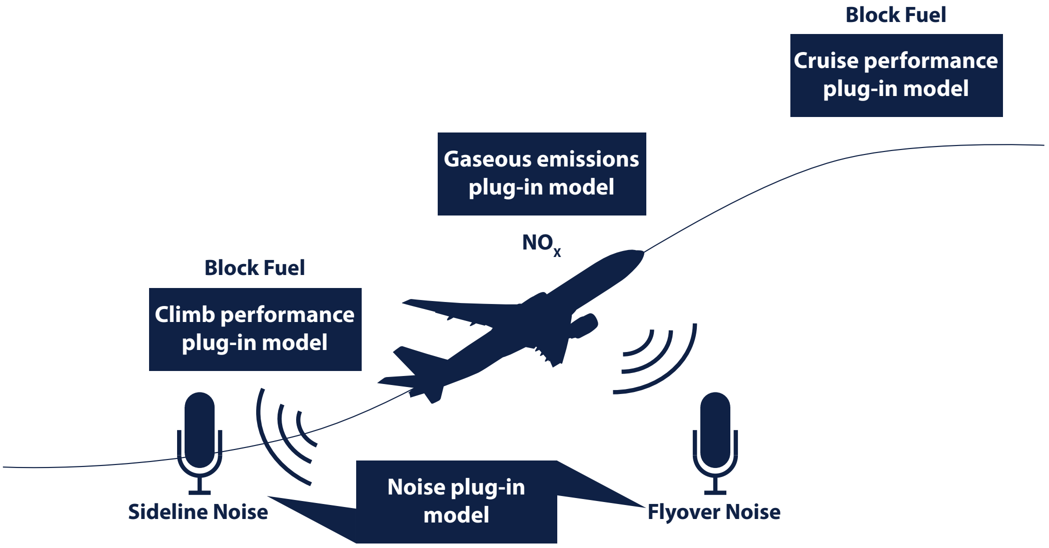
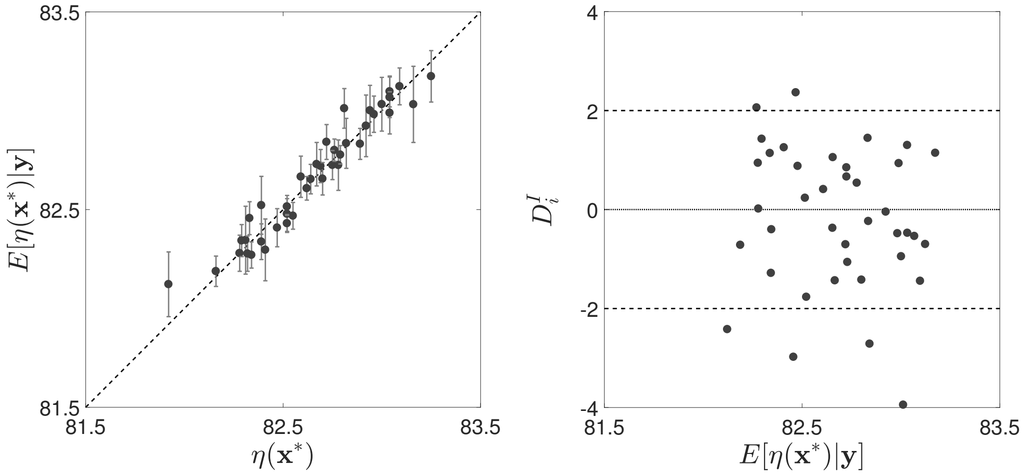

Block Fuel
Cruise performance plug-in model
Gaseous emissions plug-in model
NO_x
Block Fuel
Climb performance plug-in model
Sideline Noise
Noise plug-in model
Flyover Noise

Figure 6.10: Conceptualisation of the case study and model plug-ins for each figure of merit.

6.4.2 Sensitivity analysis

Given the setting and model, the overall aim of the case study was to narrow the set of possible aircraft configurations to a selection of feasible designs using uncertain, multi-dimensional decision criteria. The goal of performing sensitivity analysis was to discover the parameters that strongly contribute to the variation in the figures of merit, potentially helping to inform decisions about how the design sets can be changed efficiently. Even though AirCADia is a simplified model, its computational cost makes it challenging to use the code directly for Monte-Carlo-based analysis. A GPE was used to alleviate this.

Figure 6.11 and Figure 6.12 show a set of diagnostics for the GP emulators built for flyover noise and block fuel, respectively, based on $n = 60$ training and $m = 40$ validation runs of AirCADia. A good agreement can be observed between emulator and simulator in Figure 6.11(a) and Figure 6.12(a). Some local problems can be observed in both IPE plots, Figure 6.11(b) and Figure 6.12(b).



(a)

(b)

Figure 6.11: GPE diagnostics for flyover noise. (a) Correlation between predicted and observed data and 95% credible interval; (b) IPE for the emulator.

These can be attributed to regions where the data was insufficient to represent the simulator accurately. Therefore, after validation, the test dataset was added to the training data and the parameters β and σ^2, re-estimated. Overall, the GPE was credited with sufficient predictive capabilities to be used as a substitution for the model in sensitivity analysis.

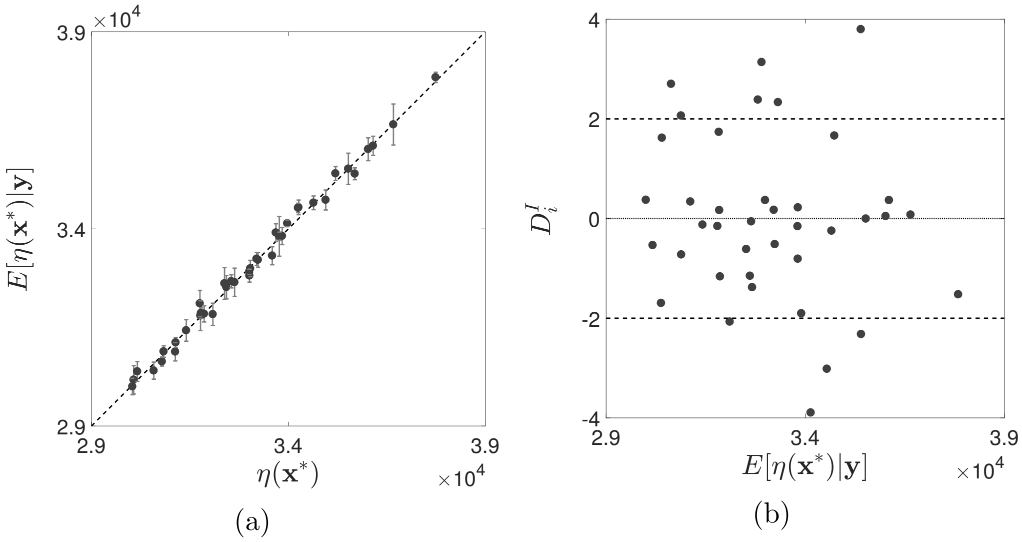
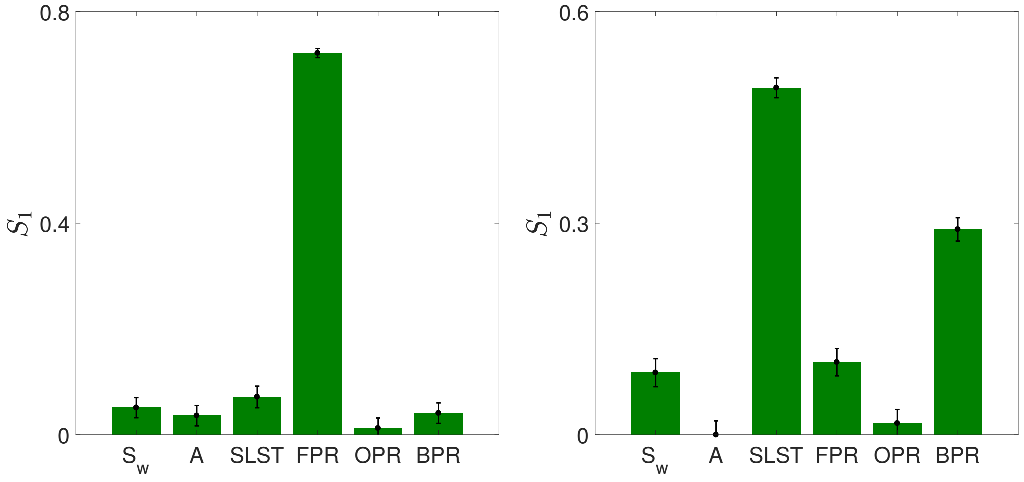


Figure 6.12: GPE diagnostics for block fuel. (a) Correlation between predicted and observed data and 95% credible interval; (b) IPE for the emulator.

In this analysis the inputs and outputs are treated as independent and all inputs are assumed to have a uniform distribution with ranges given in Table 6.4. The first and total Sobol' indices were estimated based on 10000 samples per input.

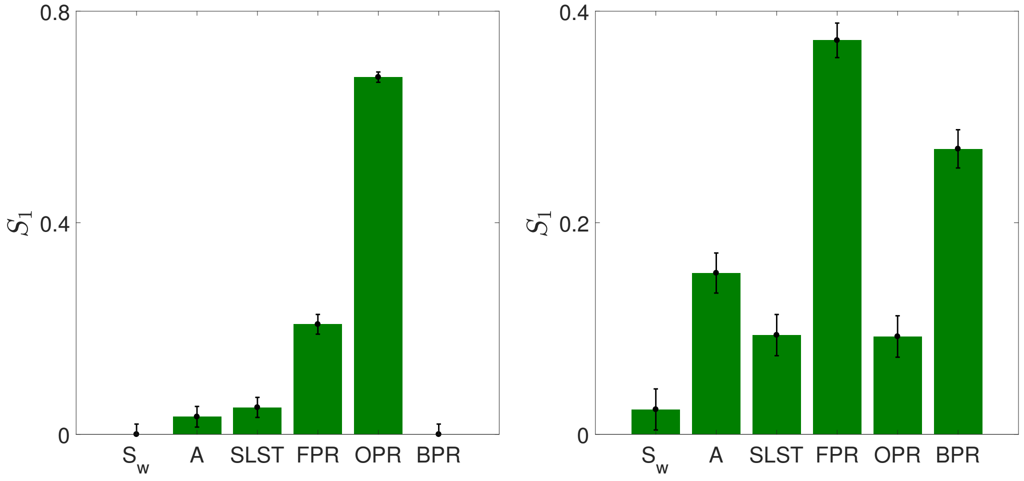
Figure 6.13 and Figure 6.14 show the relative importance of the six model parameters to each of the outputs, through the first order Sobol indices. The importance of acknowledging the uncertainty introduced by the GPE is evident in Figure 6.13 and. For example in Figure 6.13(a) the error bars indicate that wing area, aspect ratio, static thrust and bypass ratio can switch their order in the importance ranking. In general however, the uncertainty of the GPE has an appreciable effect on inputs with low to moderate Sobol' indices and does not contribute to significant shift in the relative importance of the variables.



(a)

(b)

Figure 6.13: First order Sobol' indices. (a) Flyover noise; (b) Sideline noise. Error bars indicate ±2 standard deviations in the results, obtained from sampling the GP predictive distribution 1000 times.



(a)

(b)

Figure 6.14: First order Sobol' indices. (a) NO_X emissions; (b) block fuel. Error bars indicate ±2 standard deviations in the results, obtained from sampling the GP predictive distribution 1000 times.

The total Sobol indices where also computed. As it can be seen in Figure 6.15 and Figure 6.16, the relative order of importance does not change and in most cases there is little absolute change. This fact indicates that interactions between input variables do not contribute significantly to the variability of the output. The numerical values of the first and total order indices are given in Table 6.5, to help appreciate the magnitude of the contribution from interactions. There, some notable exceptions from the above statement can be seen.

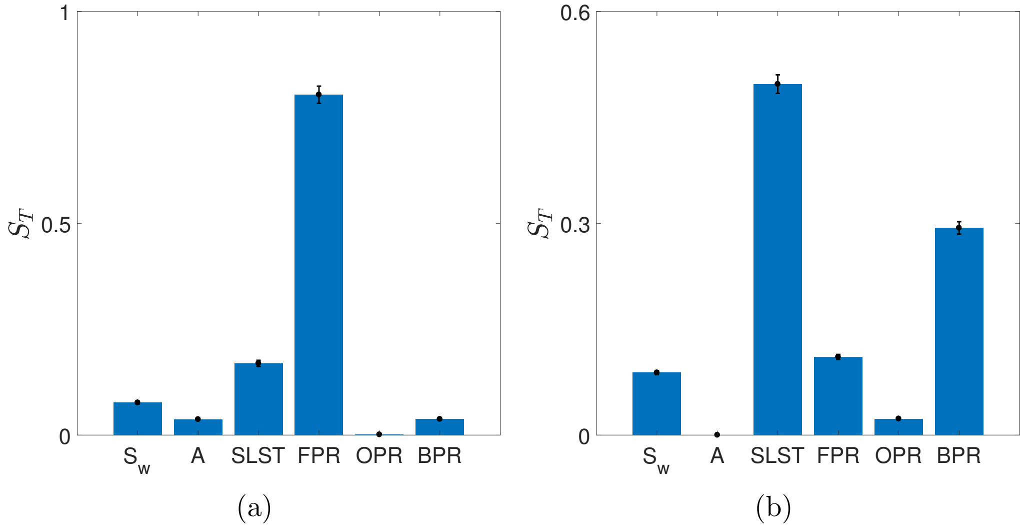


Figure 6.15: Total Sobol' indices. (a) Flyover noise; (b) Sideline noise. Error bars indicate ±2 standard deviations in the results, obtained from sampling the GP predictive distribution 1000 times.

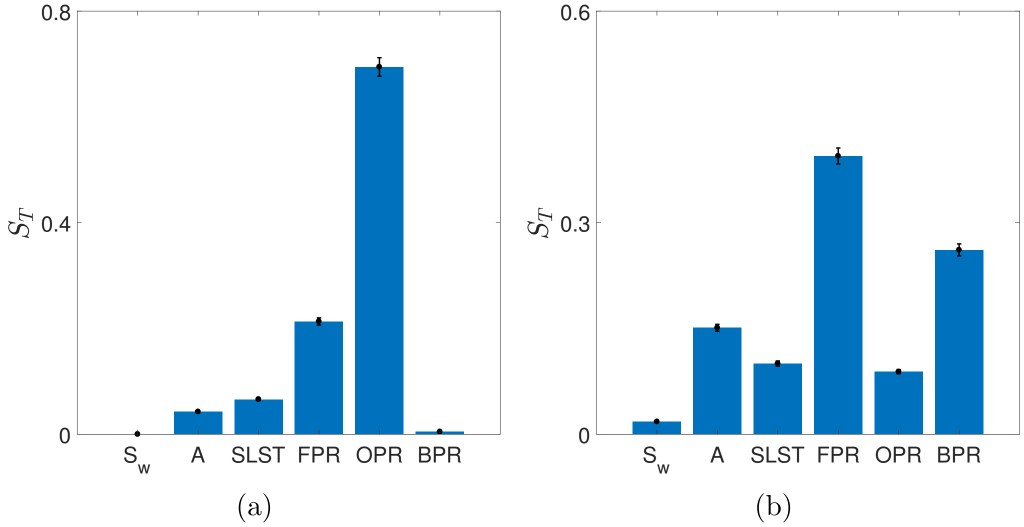


Figure 6.16: Total Sobol' indices. (a) NO_X emissions; (b) Block fuel. Error bars indicate ±2 standard deviations in the results, obtained from sampling the GP predictive distribution 1000 times.

Table 6.5: First order and total Sobol' indices for the climb-cruise match case study using AirCADia.

Input	Flyover noise		Sideline noise		$\mathrm{NO_X}$ emissions		Block fuel	
	S_i	S_{T_i}	S_i	S_{T_i}	S_i	S_{T_i}	S_i	S_{T_i}
S_w	0.051	0.077	0.088	0.088	0.000	0.000	0.024	0.018
A	0.036	0.037	0.000	0.000	0.033	0.043	0.153	0.151
SLST	0.072	0.169	0.492	0.497	0.051	0.066	0.094	0.100
FPR	0.722	0.803	0.103	0.111	0.208	0.213	0.372	0.394
OPR	0.008	0.000	0.016	0.023	0.675	0.694	0.087	0.089
BPR	0.039	0.038	0.291	0.294	0.000	0.005	0.260	0.261

6.5 Chapter summary

This chapter contributes the next step in the analysis of the LB model, through the development and use of a general-purpose method for affordable sensitivity analysis. With the help of Gaussian process emulation for the lattice Boltzmann model, described in Chapter 5, an investigation of the influence of flow and fibre parameters on the escape volume of water is made possible. This investigation is accomplished with the computation of first and total Sobol' indices for each parameter. These two indicators measure the influence of the inputs on the uncertain model response. In general an input variable is considered to be non-influential (or with little impact) on the model response if both Sobol' indices are close to zero. If, in addition there are considerable relative changes between the first and total indices, an examination of the effect of interactions for the inputs under consideration may be needed as demonstrated in Section 6.2. There it was found that all investigated parameters interact in a non-trivial manner.

To demonstrate the versatility of the approach described in this chapter three different models were analysed without any modification to the used technique. In each case, the uncertainty introduced through the use of a GPE was efficiently accounted for in the estimation of the Sobol' indices.

The uncertainty quantification framework is developed further in the next chapter, where a novel method for reliability analysis is presented. This final ingredient in the uncertainty quantification suite developed in this dissertation helps to significantly reduce the time required to perform reliability analysis.

CHAPTER 7

A novel method for reliability analysis[1]

7.1 Introduction

In this chapter a novel efficient algorithm for performing reliability analysis with complex computer codes is introduced. The method termed GPSS (after Gaussian processes and subset simulation) is aimed at computationally expensive codes and its purpose is to deliver a flexible yet practical tool for reliability analysis. The method is an innovative combination between metamodelling and optimisation techniques outlined in this chapter and Chapter 3. For the benefit of the reader, this chapter reminds about some of the concepts and nomenclature introduced previously.

[1] The ideas developed in this chapter have been submitted for publication as Hristov, P.O., DiazDelaO, F.A., Farooq, U., Kubiak, K.J. "Adaptive Gaussian process emulators for efficient reliability analysis", *Applied Mathematical Modelling*

7.1.1 Problem statement

Reliability analysis, in the most general sense, is concerned with the calculation of a probability of failure, p_F via the identification of the failure domain, $F = \{x : \eta(x) > y^*\}$ for a *performance function* $\eta(\cdot)$. This function contains all the available information for a complex engineering system (*e.g.* stresses, loads). The probability of failure can thus be computed, in principle, by solving

$$p_F = \int_F \pi(\mathbf{x}) d\mathbf{x} \tag{7.1}$$

where $\pi(x)$ is the joint probability distribution for x. The modeling of x as a random variable is justified given the complexity of modern, realistic engineering systems. Without loss of generality, the performance function $\eta(\cdot)$ is considered to be a mapping from its input space $\mathcal{X} \subset \mathbb{R}^d$, to a scalar on the real line, $y \in \mathbb{R}$. In typical engineering applications the explicit form of this mapping is not known and it is considered to be a *black-box*. Black-box models are ubiquitous in engineering in the form of *e.g.* finite element solvers, computational fluid dynamics tools, climate models among others. A common feature is their computational intensity. For a typical system, the failure region F is significantly smaller than the input domain and is considered a rare event. The most straightforward method to evaluate the multidimensional integral in Eq. 7.1 is direct Monte Carlo (DMC) sampling. The DMC estimator, $\hat{p}_F$ is unbiased and independent of the number of input dimensions, d, but requires a large number of samples to ensure that F is well populated to allow a reasonably accurate estimation of its properties. This drawback, coupled with a computationally expensive model renders a DMC-based reliability analysis infeasible. A method widely used in engineering for performing reliability analysis is *subset simulation* (SuS) [41]. Despite the significant improvement SuS brings over DMC, it can still be quite expensive when evaluating the simulator directly. In order to address the computational cost of the simulator, one can build a less expensive approximation to the code output. In this chapter the Gaussian process emulator, introduced in Chapter 3 will be used as a basis to the development of a novel method for reliability analysis. The presented discussion shows that using the emulator directly may lead to inaccurate and misleading results, and proposes a way of rectifying this problem.

7.1.2 Existing methodologies

In order to develop an original algorithm, a very extensive literature review was carried out and is presented here. It should, however, be noted that the size of reliability analysis literature is considerable, and thus this review focuses on approaches which aim to improve the efficiency of reliability analysis via the use of a surrogate model. Methods using Gaussian process emulators are examined in more detail. Most of the reviewed work relies on sequential sampling for the gradual improvement of the surrogate in the vicinity of the failure domain. This idea is heavily influenced by the field of surrogate-based optimisation (SBO), where an initial surrogate for a function is built from an experimental design and is used to search for optima. The schools of optimisation and reliability analysis are in fact so closely related that they form one of the most important aspects of product development - reliability based design optimisation (RBDO). Therefore, relevant methods from SBO and RBDO are reviewed in this section. A number of authors provide comparison between different surrogate models. For instance Simpson, [156] compares Gaussian process emulators with second order response surface models; Turner, [157] juxtaposes GPE and B-spline approximations; Queipo et *al.* compare the performance of polynomial regression, GPE and radial basis function surrogates in [158]; finally Wang and Shan, [159] provide an overview of the metamodels generally suitable for sequential improvement. A structured and model-independent approach is presented by Audet et *al.* in [160] where the authors develop a framework for managing the use of surrogates for optimisation.

A natural bridge between optimisation and reliability analysis is provided by Li and Au, [161], where they use SuS to perform optimisation, arguing that both the region in the vicinity of the global optimum and the failure domain represent a rare event. Some authors, such as Ranjan et *al.* [162] and Picheny et *al.* [163] have explored surrogate-based optimization strategies in the context of accurate target region estimation. They suggest that the emulator should only be refined in the regions of interest and that in order to obtain accurate region estimates a good quality surrogate is all that is needed. This idea can be directly applied to reliability analysis by recognizing that the failure domain is bounded by a limit state - a target region. An opposing view is given by Zhu and Du, [164] where the authors work directly with the reliability estimates as a measure of analysis quality. The approach proposed in this work agrees with and extends the former idea.

In general, most approaches build a ground level surrogate from some initial design. It is observed that optimisation and reliability algorithms have this step in common since both are aiming at a good global surrogate at first. To that effect, an A-optimal design is used by Osio and Amon, [165] and orthogonal arrays are chosen by Bang et *al.* [166]. A maximin design is adopted by Kuczera and Mourelatos in [167], while Dubourg, [168] uses a hypersphere-filling sampling plan. Some additional design strategies for metamodelling are presented by Jones and Johnson, [169]. The proposed approach uses the well-regarded Latin hypercube sampling (LHS) by McKay et *al.* [110], which is almost ubiquitously utilized in surrogate modelling. The details of the initial sampling plan are discussed in Section 7.3.1. In general, most methods follow a framework which is composed of a sampling rule, utility function and a stopping criterion amongst other specific details. Lee and Jung, [170] use constraint boundary sampling to select improvement points. A combination between MCMC sampling and k-means clustering is used by Dubourg et *al.* [171] to select new data points. In [172], Jin et *al.*

compare different metrics for the same purpose. An approach based on the first order reliability method is used by Su et *al.* in [173] and a probabilistic classification function is presented by Zhang et *al.* [174]. In [175], Bect et *al.* derive a stepwise uncertainty reduction methodology based on expected improvement (EI) [40], but formulated from a Bayesian risk perspective. A weighted integrated mean squared error is presented in [163]. Other methods that use EI or EI-based strategies are those in [176–178]. Other previously used utility functions include the U-function [177, 179], the improved U-function [180], a least improvement function [181] and a random-set-based function in [182]. A taxonomy of popular infill criteria is given by Sasena,[183]. All approaches based on a utility function, except those by Wen et *al.* [178] and Tong et *al.* [180] search the entire input space for a candidate point that maximizes that function and add it to the training plan for the next iteration of the algorithm. Furthermore, many of the aforementioned strategies rely on a pre-generated population of samples (e.g. [177]). This could prove to be suboptimal because at the time of generation nothing about the limit state was known and thus there is no guarantee the information these samples provide about the failure domain will be accurate. From a sampling point of view, the majority of the methods generate one point at a time, which, given the dynamical nature of sequential sampling could either miss important regions of the domain, or slow down convergence. These issues are addressed in the present chapter.

The other major part of all adaptive algorithms is the stopping condition. This ranges from the use of reliability indices [171, 173] through error in the estimation of the failure probability [164, 177, 179–181] and forms of measure of the discrepancy between the emulator predictions and code observations [163, 170, 174, 182] to thresholds on the learning function [162, 176, 178]. In this paper, a stopping condition which relies implicitly on the similarity between the surrogate and the model in the failure domain is proposed to terminate the learning process.

Finally, Gaussian process emulation is not the only surrogate used in reliability analysis. Among others, general response surfaces [184–187], neural networks [188–190], support vector machines [191, 192], radial basis functions [172], polynomial chaos expansions [193] and moving least squares [167] have been used in different ways. However, this work focuses on GPE because it not only predicts the output at an untried data point, but, as already mentioned in Chapter 3, provides a measure of uncertainty in the prediction. This quality is used by the proposed algorithm as discussed in following sections.

7.2 Subset simulation

As established in Chapter 1, Parker Hannifin wish to conduct reliability analysis to estimate the probability of failure of their filters. This section provides an abstract formulation of the solution as expressed by subset simulation. One very important problem in engineering is the estimation of the probability of failure of a system, p_F, given in Eq. 7.1. In the context of numerical simulations failure can be defined as the scenario where a response variable (output) of the model, exceeds some threshold of acceptable system behaviour. The output, y is related to the input variables, $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$, via some mapping provided by the simulator $y = \eta(\mathbf{x})$. Thus, the failure domain, as defined in the beginning contains the values of $\mathbf{x}$ which cause the system response, y to exceed a critical value y^*

$$F = \{\mathbf{x} : \eta(\mathbf{x}) > y^*\} \tag{7.2}$$

Estimating p_F is associated with sampling from F. Usually, for a well-designed system the true value of p_F is very small, that is, F is a *rare event.* Also, a typical model has a high dimensional input space and often the failure domain of that space is disjoint, so sampling from it poses a significant challenge. SuS [41] expresses the failure event as contained in a nested sequence of less-rare events.

$$F \subset F_m \subset F_{m-1} \subset \ldots \subset F_1 \tag{7.3}$$

where F_1 is a relatively frequent event. Given that sequence, it can be shown that the probability of the rare event F can be expressed as a product of larger conditional probabilities:

$$p_F \equiv \mathbb{P}(F) = \mathbb{P}(F_1) \cdot \mathbb{P}(F_1|F_2) \cdot \ldots \cdot \mathbb{P}(F|F_m) \tag{7.4}$$

Beginning from the unconditional level F_0, the algorithm "probes" the input space $\mathcal{X}$ via direct Monte Carlo sampling. Then, based on the values of y by the simulator, it constructs the first intermediate failure threshold, $y_1^* < y^*$, defining an intermediate or relaxed failure domain, F_1. SuS then populates F_1 with N_s samples, using an MCMC algorithm. The generation and population of intermediate, equiprobable levels ($\mathbb{P}(F_i|F_{i-1}) = p_0$) continues until a predefined number of samples lie in the true failure domain F. At the end of the algorithm an estimate of the complementary cumulative distribution function (CCDF) of the response function is generated [194].

7.3 Proposed approach

This section introduces the proposed method for efficient reliability analysis. The method is a novel combination between Gaussian process emulation and subset simulation, and as such is termed *Gaussian process subset simulation* (GPSS). All distinct steps of the process are described in detail below and a pseudocode is provided as Algorithm 2

7.3.1 Initialization

In order to reliably use the emulator it needs to be of sufficient quality around the estimated failure regions. This is not usually the case with GPE approximations built using data from the high probability regions (according to the input PDF). For the physical model it is assumed that the critical failure threshold, y^* is known (given *a priori*) and sensible (i.e. there is a set of values of $\mathbf{x}$ for which $\eta(\mathbf{x}) > y^*$). The proposed algorithm starts by building a GPE based on a set of data points, selected according to a space-filling strategy, LHS in this case. The interested reader is pointed to [158] and [195] for discussion on the suitability of LHS designs for surrogate modelling. In [196] the authors choose the number of training points as the one needed to define a unique quadratic response surface for the output of the model. The initial number of samples in the present work is chosen to conform to the already introduced rule that the training sample should have 10 times as many points as there are dimensions in the model. This rule was introduced in Chapter 3 and used in Chapter 6. The GPE can be validated to check if there are any large discrepancies between the emulator and simulator as in [120]. This is done to ensure that the initial approximation is of a reasonable overall quality.

It is very likely that the original threshold is not attained on the first iteration since the GPE learns about the simulator output from samples that do not constitute a rare event. This is to say that it is very likely that

$$F_1 = \{\mathbf{x} : \mathbb{E}[\eta_1(\mathbf{x})|\mathcal{D}] > y^*\} = \varnothing \tag{7.5}$$

where F_ℓ is the failure domain according to the ℓ^{th} emulator ($\ell = 1$ above), $\eta_\ell(\mathbf{x})|\mathcal{D}$ is the emulator output conditional on the training runs and $\mathbb{E}[\cdot]$ is its predictive mean, whose functional form is given in Eq. 3.12. For the sake of conciseness, note that the conditioning on $\hat{\boldsymbol{\theta}}$ has been dropped. Even though the emulator provides an affordable prediction for any point in the input space, using Monte Carlo to discover domains with small probability may prove to be prohibitively slow. This is because the emulator generates predictions by computing the distances between training and test points and among all test points. The size of the Monte Carlo sample required to get a reliable estimate of the features of the failure domain is usually large, resulting in an inefficient use of the GPE. Thus, it is proposed to use SuS to sample from the posterior predictive mean of the emulator. As outlined in Section 7.2, SuS converges when a predefined number of data points lie in the failure domain. If it is indeed the case that $F_1 = \varnothing$, the algorithm will be unable to naturally converge due to all candidate samples being rejected. Then, an alternative "failure level", $\tilde{y}_1^*$ could be set for which $\mathbb{P}(\mathbb{E}[\eta_1(\mathbf{x})|\mathcal{D}] > \tilde{y}_1^*) > 0$. This approach gives rise to an intermediate emulator "failure" domain with respect to $\tilde{y}_1^*$, denoted as $\tilde{F}_1$. The purpose here is to sample from $\tilde{F}_1$ rather than to estimate the probabilities of failure (conditional or otherwise). The emulator is still a very efficient approximation of the code and thus in theory one can search the input space exhaustively. However, a preference is given to SuS as it leads to a quick and reliable convergence, especially in high dimensions. The algorithm can be ran with a fairly large number of samples, N_s (a lot less than if exhaustive search is performed) at each level in order to ensure that even truly rare subregions of $\tilde{F}_1$ are populated.

7.3.2 Level generation

The above discussion raises the question of how to calculate $\tilde{y}_\ell^*$ such that SuS converges, whilst simultaneously exploring potentially interesting regions of the input space. Several different rules were conceived and tested with benchmark problems. The rules are presented in Table 7.1 along with some observations about each one. A graph of each level using the Goldstein-Price function (which is presented in Section 7.4.1) is shown in Figure 7.1. The criterion that was found to give satisfactory results is presented in Eq. 7.6 ("Rule 5" in Table 7.1). Essentially, the idea is to select the current threshold as the average of all training responses that lie above the previous critical level and then take the midpoint between the resulting quantity and the value of the previous level.

$$\tilde{y}_\ell^* = \begin{cases} \frac{\sum_{i=1}^n y_i}{n} & \text{if } \ell = 1 \\ \frac{1}{2}\left[\frac{1}{n}\sum_{i=1}^n y_i \mathbb{I}(y_i) + \tilde{y}_{(\ell-1)}^*\right] & \text{if } \ell > 1 \end{cases} \tag{7.6}$$

In Eq. 7.6 and Table 7.1, y_i are the n training responses and $\mathbb{I}(y_i)$ is an indicator function returning 1 if $y_i > \tilde{y}_{(\ell-1)}^*$ and 0 otherwise. This is not to be confused with $\mathbb{I}_F$ in Algorithm 2, where the success condition is that the argument set lies within F. The criterion ensures that the current threshold will lie above the previous one, whilst safeguarding against setting it too high and discarding regions that may lead to system failure. It is noted that this choice provides a robust and balanced performance to GPSS, but the analyst could give preference to either quick convergence or longer exploration by selecting any of the rules in Table 7.1.

7.3.3 Sample selection

When SuS converges, it populates $\tilde{F}_\ell$ with points whose predicted response lies above the current failure level $\tilde{y}_\ell^*$. However, it would be incorrect to only use samples from the GPE, since it is an approximation to the simulator's output. A mechanism to correct this has to be part of the algorithm. Among all of these samples some additional point/s at which to run the model have to be selected before fitting the next emulator prediction. Progressively adding new data to the GPE training set will represent the failure regions with increasing accuracy. However, since each evaluation of the model is expensive, care needs to be taken to choose new points in a way that maximizes the amount of information gained

Table 7.1: Different level definition rules.

Rule	Expression	Remarks
1	$\tilde{y}^*_\ell = \left.\frac{\max(y_i)+\min(y_i)}{2}\right\vert_\ell$	Slow convergence; useful for highly non-linear outputs.
2	$\tilde{y}^*_\ell = \left.\frac{y_i+y_j}{2}\right\vert_\ell$ for $y_i \leq y_j;\ i = j+1$	Equiprobable rule; slow convergence.
3	$\tilde{y}^*_1 = \frac{1}{N}\sum_{i=1}^N y_i$ $\tilde{y}^*_\ell = \tilde{y}^*_{\ell-1} + a$	Linear rule; non-informative; a is a positive constant.
4	$\tilde{y}^*_1 = \frac{\sum_{i=1}^N y_i}{N}$ $\tilde{y}^*_\ell = \frac{1}{N+1}\left[\sum_{i=1}^N y_i \mathbb{I}(y_i) + \tilde{y}^*_{(\ell-1)}\right]$	Adaptive rule; gives less weight to previous level.
5	$\tilde{y}^*_1 = \frac{\sum_{i=1}^N y_i}{N}$ $\tilde{y}^*_\ell = \frac{1}{2}\left[\frac{\sum_{i=1}^N y_i \mathbb{I}(y_i)}{N} + \tilde{y}^*_{(\ell-1)}\right]$	GPSS adaptive rule; gives more weight to previous level.

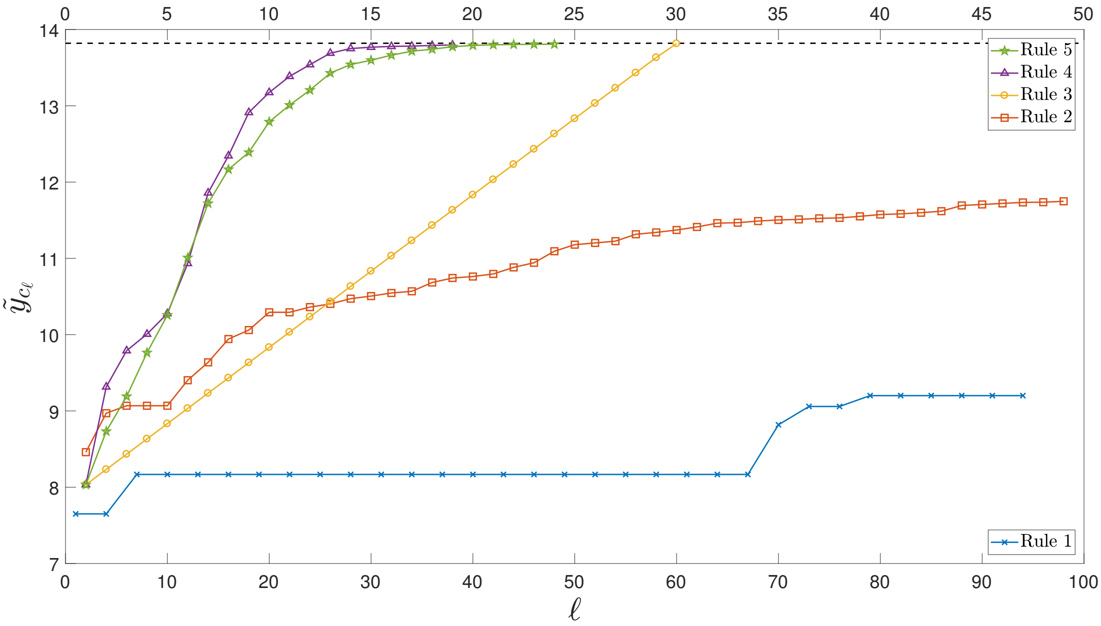


Figure 7.1: Comparison between alternative failure levels for the Goldstein-Price function.

from sampling at their locations. Thus a condition is needed which dictates how to choose candidate points to be considered when augmenting the design, $\mathcal{D}$. The posterior predictive variance of the emulator is proposed as an indicator of the local quality of the approximation. Since the GPE is an interpolator, at any given training point $\mathbb{V}[\eta_\ell(\mathbf{X} \in \mathcal{D}_\ell)|\mathcal{D}_\ell] = 0$. Therefore, if at any level of the GPE improvement the value of the posterior predictive variance for a sample point lies below a certain level ε, that point should not be considered a candidate for design improvement. This strategy prevents the algorithm from suggesting new samples in regions where the emulator is an accurate enough approximation of the output, thus saving potentially appreciable time for code evaluations. The correct value of ε is problem dependent since it scales with the output of the simulator. Of course it could be set arbitrarily small, but attention needs to be paid to the trade-off between final emulator accuracy and computational cost. Let $\mathbf{X}_{cand}$ be the candidate population. Once it has been determined, the point that satisfies some criterion can be selected as an improvement point. A widespread choice for making the selection is the *expected improvement* (EI) criterion [40]. EI is a strategy that balances exploitation of the emulator mean and exploration of the design space.

$$\begin{aligned}\mathbb{E}[I(\mathbf{x})]_\ell =& \left|(y_c - \mathbb{E}[\eta_\ell(\mathbf{x})|\mathcal{D}_\ell])\right|\Phi\left(\frac{y_c - \mathbb{E}[\eta_\ell(\mathbf{x})|\mathcal{D}_\ell]}{\mathbb{V}[\eta_\ell(\mathbf{x})|\mathcal{D}_\ell]}\right)+ \\ &+\mathbb{V}[\eta_\ell(\mathbf{x})|\mathcal{D}_\ell]\phi\left(\frac{y_c - \mathbb{E}[\eta_\ell(\mathbf{x})|\mathcal{D}_\ell]}{\mathbb{V}[\eta_\ell(\mathbf{x})|\mathcal{D}_\ell]}\right)\end{aligned} \tag{7.7}$$

where the subscript ℓ denotes information regarding the ℓ^{th} emulator and $\mathbb{V}[\cdot]$ is the posterior variance of the ℓ_{th} emulator given in its functional form in Eq. 3.13. The symbols $\Phi(\cdot)$ and $\phi(\cdot)$ denote the cumulative and probability distribution functions of a standard normal random variable, respectively. Let $\mathbf{X}_{add}$ denote the points to be added to the candidate population, such that

$$\mathbf{X}_{add} = \underset{\mathbf{X}_{cand}}{\operatorname{argmax}}\, \mathbb{E}\big[I\big(\mathbf{X}_{cand}\big)\big]_\ell \tag{7.8}$$

Expected improvement can be run on each point in $\mathbf{X}_{cand}$ and the one that maximizes it will be selected as the one that is expected to bring the greatest improvement in the quality of the next level GPE. However, applying expected improvement directly to $\mathbf{X}_{cand}$ poses a risk of neglecting subregions in the intermediate emulator failure domain. Unless $\tilde{F}_\ell$ is not disjoint, the presence of separate modes has to be accounted for. This could be achieved by first identifying the structure of $\tilde{F}_\ell$, detecting any modes and calculating EI on the samples in each mode. A clustering algorithm is used to discover the separate failure sub-domains. GPSS utilizes DBSCAN, a clustering algorithm introduced in Chapter 4, for problems with lower dimensionality and *k-means* for application with larger number of dimensions [131]. The reason behind this choice is a practical one. As mentioned in Section 4.3, DBSCAN does not need the number of clusters to be specified *a priori*, but relies on Euclidean distance between points - a concept that is not well defined in higher dimensions - to form clusters. On the other hand k-means relies on a heuristic such as the *gap criterion* [197] to find an optimal k. DBSCAN was found to provide more robust and faster clustering then k-means for up to $d = 5$. Note that the modular nature of GPSS allows for these of other robust clustering algorithms to be used at the analyst's discretion. It is noted that clustering has been used for parallelisation [178] and as a sample strategy [171, 198] in related work. Ultimately, the design plan for the emulator at the next level is composed of the current design plus the new points, formally $\mathcal{D}_{\ell+1} = \mathcal{D}_{\ell} \cup \{\mathbf{X}_{add}, \eta(\mathbf{X}_{add})\}$, where $\mathbf{X}_{add}$ is given in Eq. 7.8.

7.3.4 GPE architecture

A feature that is frequently absent from the discussion of other approaches is the *re-training* of the GPE upon augmenting the design $\mathcal{D}$. As mentioned in Chapter 3 there is a number of smoothness parameters $\boldsymbol{\psi}$ that need to be estimated from the data. This consists of two major steps which could pose some challenges. Firstly, an optimisation routine is used to search the likelihood of the model parameters for its optimum. This can be achieved with both local and global optimizers, which in this order represent increasing accuracy and computational cost. The second problem is that each optimisation run requires the inversion of the correlation matrix $\mathbf{C}$ in Eq. 3.12 and Eq. 3.13, which can become ill-conditioned as points begin to cluster around the failure domain. In GPSS, the first challenge is tackled by re-estimating the smoothness parameters after a predefined number of iterations, using a global search. The rest of the time the mean β and the process variance σ^2 are estimated by re-fitting the GPE to the new design with the existing hyperparameters. A formal procedure to do this is presented in [199]. To solve the second issue, regularisation techniques such as the use of a *nugget* [118] are utilized. An alternative approach is outlined in [160].

7.3.5 Stopping condition

As outlined above, the GPE can be used to reflect the failure regions of the true function with an arbitrary precision. However to keep the procedure efficient it needs to be refined just enough for the purposes of the underlying analysis. In order to stop the iterative generation of prediction the following rule is proposed. Consider running SuS with the true function. By design SuS will stop generating new levels once a sufficient number of samples from the last level lie in the failure domain, F. It follows that a necessary and sufficient criterion for the accuracy of the emulator is the ability of SuS to generate the same number of samples, $N_{\tilde{F}_\ell}$ that belong to the failure domain and have $\mathbb{V}[\eta_\ell(\mathbf{x})|\mathcal{D}] < \varepsilon$. If this condition is satisfied, there will be no distinction between emulated and simulated samples.

7.4 Numerical experiments

In this section, the performance of the algorithm is demonstrated with two numerical problems, before being applied to the main focus of this investigation: estimating the probability of failure of the biodiesel filter. The function in Section 7.4.1 features a truly rare event for an efficiency test. The problem in Section 7.4.2 tests the ability of GPSS to deal with disjoint failure domains and Section 7.4.3 presents an example of GPSS being applied to the simulator developed in Chapter 5, which, as has been shown, is computationally expensive.

7.4.1 Goldstein-Price function

The Goldstein-Price function originates from the field of optimisation as a benchmark problem [200]. Its original form is given by

$$\begin{aligned}\eta(\mathbf{x}) =&[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 + 14x_2 + 6x_1x_2 + 3x_2^2)] \\ &\times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2)]\end{aligned} \tag{7.9}$$

The function is traditionally evaluated in the interval $x_i = [-2, 2]$ for $i = 1, 2$. For the purposes of this chapter, the natural logarithm of the Goldstein - Price function is used and the input domain is rescaled as $[0, 1]^2$. The former transformation is carried out to reduce the range of the output values, originally $[3, 1.02 \times 10^6]$. This step also enables the analyst to choose a more intuitive value of the variance threshold, ε. The log-transformation of the output is not uncommon in emulation practice. The critical level, $y^* = 13.82$ was chosen to yield a fairly small failure probability, $p_F = 9.13 \times 10^{-5}$.

The initial GPE was trained with $n = 20$ LHS points. To add to the challenge of discovering the rare event the starting sample for x_2 was deliberately generated in $[0, 0.7]$. The variance threshold was set to be, $\varepsilon = 10^{-5}$. Figure 7.2 shows a selection of GPSS steps from $\ell = 1$ to completion. Each sub-figure shows the current intermediate threshold, $\tilde{y}_\ell^*$ in red and the failure domain, $\tilde{F}_\ell$ in black. The region that the algorithm samples from at each iteration is shaded in grey. Notice that in all stages the sampling domain chosen by SuS covers the general region of the failure domain. In sub-figures $\ell = 6$ through $\ell = 9$ the algorithm samples in more than one location. This is due to the fact that in this particular case the mean of the emulator, β was estimated such that predictions away from the training points had relatively high values. Once enough information was obtained in the vicinity they disappeared from the sampling region, as seen in $\ell = 11$ onwards. It is also worth mentioning that a small black pocket is seen on $\ell = 11$, meaning that the emulator discovered a false failure region. The reason is as outlined above. Finally, the last sub-figure contains an inset showing a magnification of the sampling and failure regions. The results presented above are from a GPSS run using Rule 5 in Table 7.1. Figure 7.3 shows a sequence similar to that in Figure 7.2, this time using Rule 4 in Table 7.1. The much faster advancement toward F is obvious and a sampling sequence akin to a path can be seen in the upper left corner. This is a result of the smaller weight that each previous $\tilde{y}_\ell^*$ carries in determining the current one. The inset was removed as it is identical to the one in Figure 7.2. The estimated mean probability of failure, $\bar{p}_F = 9.57 \times 10^{-5}$ was based on 100 runs of SuS on the improved GPE. The coefficient of variation and relative error based on the same sample were $\delta_{p_F} = 23.1\%$ and $\Delta p_F = 4.8\%$, respectively. A more complete idea about the quality of the approximation can be obtained from Figure 7.4, showing a comparison between the CCDF curves for SuS and GPSS. The results obtained with Rule 4 point to two important remarks. Firstly, the importance of supplying simulator information to the GPE, instead of relying on a one-iteration emulator is clearly demonstrated. Secondly, the existence of different level-generation rules provides means of benchmarking results obtained with a particular set-up. These rules can be used to tune the algorithm to the studied problem. The performance of the algorithm was tested for smaller values of p_F, by setting $y^* = 13.803$. The small increase in the critical level results in the probability of failure being reduced to $p_F = 2.43 \times 10^{-7}$. GPSS converged in 44 iterations, sampling the simulator a total of 70 times, with $\delta_{p_F} = 34.3\%$ and $\Delta p_F = 1.8\%$.

7.4.2 Mixture of Gaussians

This function was created to test the robustness of the GPSS algorithm. The function has the form:

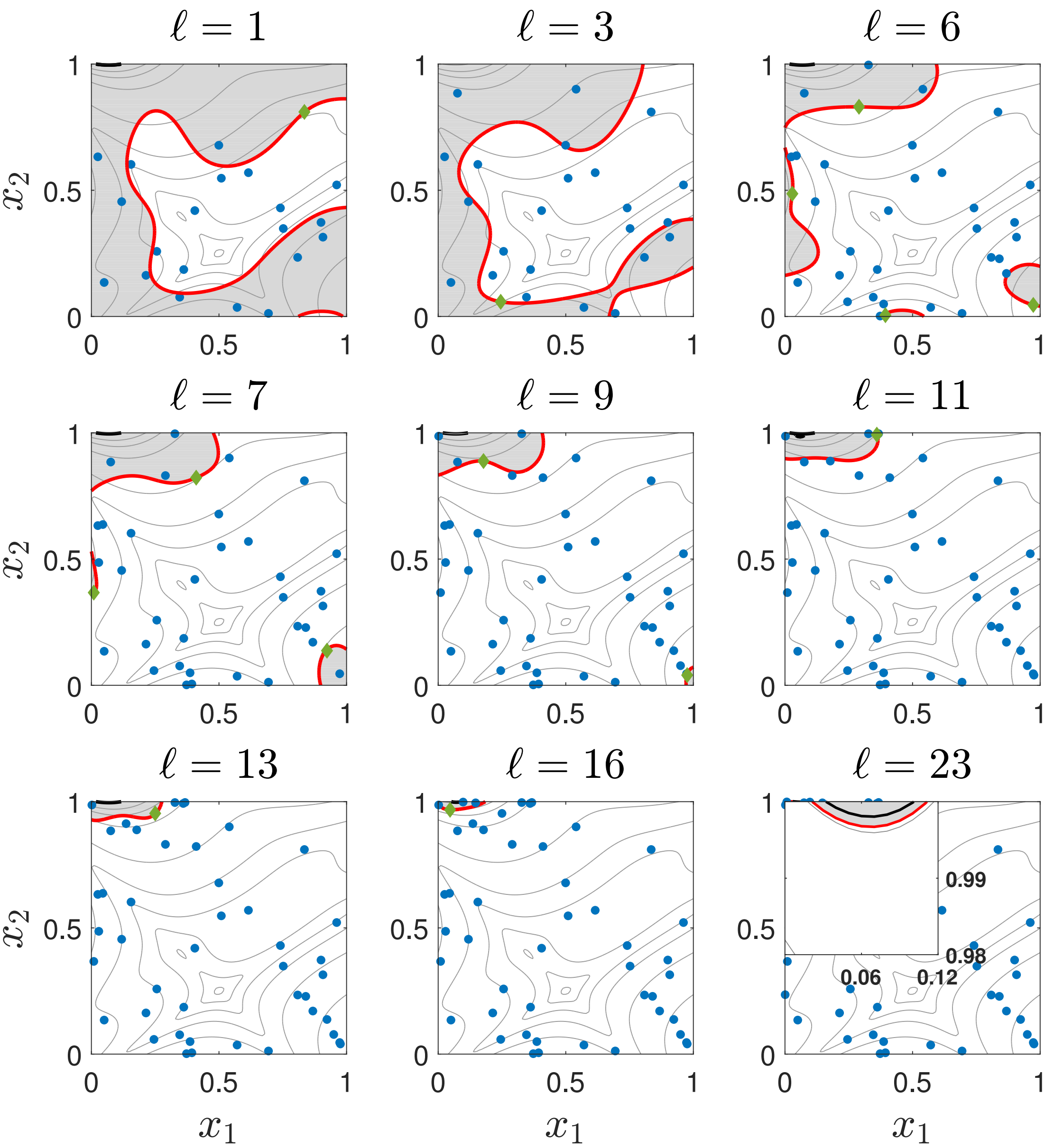


Figure 7.2: Progression of GPSS using Rule 5 for the modified Goldstein-Price function. The titles in each sub-figure denote the algorithm iteration. Red contours correspond to $\tilde{y}^*_\ell$ and black contours show $\tilde{F}_\ell$. New data points from the shaded regions are plotted as diamonds. *Inset in last tile:* Failure domain (black) and last GPSS level (red). The number of training samples in each tile is $n = \{20, 23, 30, 34, 39, 42, 44, 47, 53\}$.

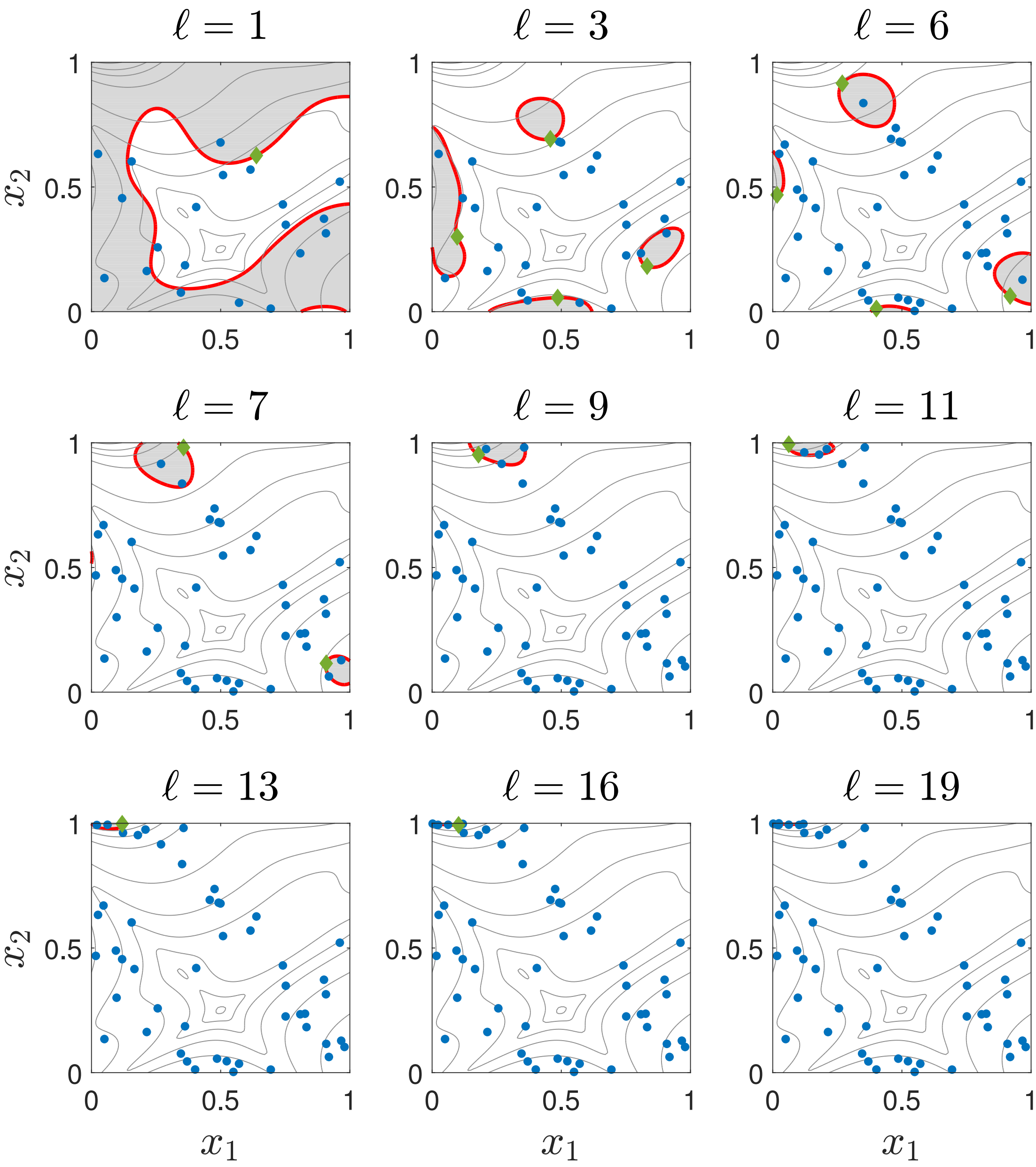


Figure 7.3: Progression of GPSS using Rule 4 for the modified Goldstein-Price function. The titles in each sub-figure signify the level of the algorithm. Red contours correspond to $\tilde{y}_\ell^*$ and black contours show $\tilde{F}_\ell$. New data points are plotted as diamonds. The number of training samples in the tiles is $n = \{20, 25, 37, 41, 45, 47, 49, 52, 54\}$.

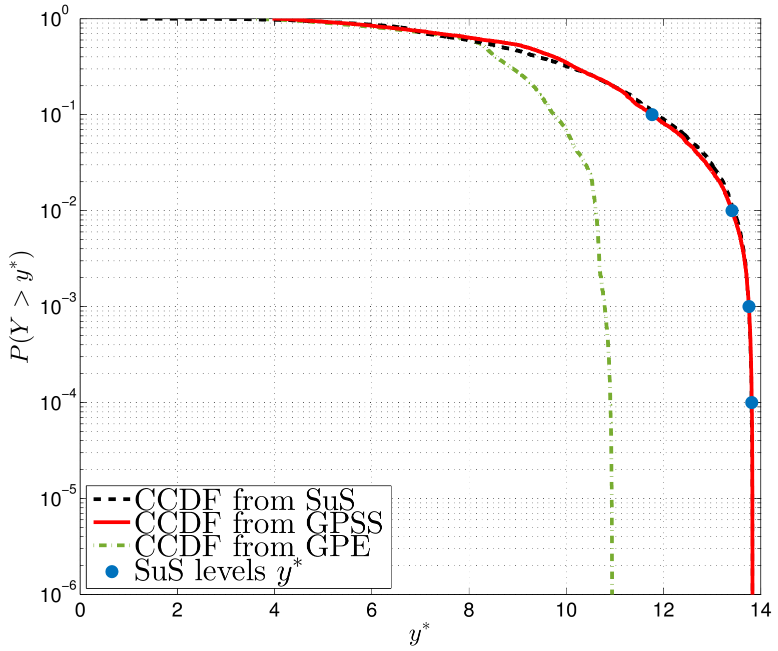


Figure 7.4: Complementary CDF of the modified Goldstein-Price function according to the GPE (green), GPSS (red) and SuS (black). The blue circles show the intermediate levels of SuS. Each level was populated with 3000 samples.

$$\begin{aligned}\eta(\mathbf{x}) &= \frac{10^2}{2}\left[a\phi(\mathbf{x}_A) + b\phi(\mathbf{x}_B) + c\phi(\mathbf{x}_C) + d\phi(\mathbf{x}_D)\right] \qquad (7.10)\\ \mathbf{x}_A &= 10\left(\mathbf{x} - 1/4\right)\\ \mathbf{x}_B &= 10\left(\mathbf{x} - 3/4\right)\\ \mathbf{x}_C &= \left[10\left(x_1 - 3/4\right), 10\left(x_2 - 1/4\right)\right]\\ \mathbf{x}_D &= \left[10\left(x_1 - 1/3\right), 10\left(x_2 - 5/6\right)\right]\end{aligned}$$

In Eq. 7.10 $\mathbf{x} = [x_1, x_2] \in [0,1]^2$ and $\phi(\cdot)$ is the standard normal PDF. The fractions in the expressions for $\mathbf{x}_A \ldots \mathbf{x}_D$ are location parameters and can be chosen arbitrarily. The constants a, b, c and d are used to select how many peaks belong to the failure domain. For this experiment $a = b = 1$, $c = 1.1$ and $d = 0.85$, meaning that peak d is just outside of failure level set at $y^* = 7.9$. The associated failure probability is $p_F = 7.27 \times 10^{-3}$. The GPE was initially trained with $n = 20$ LHS points. The variance threshold was set to, $\varepsilon = 10^{-5}$. Figure 7.5 shows the process of discovery of the failure domain. Each plot shows the contour values of the intermediate levels $\tilde{y}_\ell^*$ in Eq. 7.6 as a red line. The level of y^* is indicated with a black line, but is absent in the first row of Figure 7.5 as it was not accessible at these GPE stages. During most of the process points were added in all three modes. However, from levels $\ell = 15$ and $\ell = 16$ to the end, the algorithm could not improve the approximations in modes a and b, respectively, and stopped sampling from them. This is reflected in the last four tiles of Figure 7.5 where the green diamonds show the new samples in each consecutive frame - it can be seen that samples are only being added to mode c. This feature of the algorithm is useful in the presence of highly disjoint failure domains, where the local quality of the GPE can increase independently and resources will not be wasted where they are not needed. As before, some statistics about the probability of failure were calculated from 100 runs of SuS. The mean was found to be $\bar{p}_F = 7.31 \times 10^{-3}$.

The corresponding c.o.v. and relative error were $\delta_{p_F} = 14.1\%$ and $\Delta p_F = 2.7\%$, respectively. It is interesting to note that the same figures based on 100 runs of direct Monte Carlo simulation with 100000 samples were $\delta_{p_F}^{DMC} = 3.7\%$ and $\Delta p_F^{DMC} = 0.7\%$. Figure 7.6 shows the CCDF curve for the failure probability estimated with SuS, the one obtained via GPSS and the one calculated relying on the unimproved GPE. It can be seen, that subset simulation was unable to find failure points using only the GPE.

7.4.3 A simplified lattice Boltzmann model

The following section deals with one of the main research objectives and allows the analyst to take industrially relevant decisions about the reliability of the designed system. The example problem in this section is derived from the model developed in Chapter 5. The present set-up was introduced in [201] and is shown on Figure 7.7. It comprises a 2D simulation of a heavy phase droplet (red) submerged in a light phase medium (dark blue) and impacting with a hydrophobic fibre (light blue). Details about the code and its emulation have already been discussed in Section 5.3 and Section 5.6, respectively.

After validating the GPE, the test set was added to the training sample and the GPE was refitted, keeping the MLE values of the correlation lengths calculated on the original $n = 50$ samples. The critical level for the analysis was set equal to the value of the maximum training point, $y^* = 7452$ lattice units (lu). The natural logarithm of the code output was taken, $y = \ln(y)$, such that the transformed threshold was $y^* = 8.916$. GPSS was run with $\epsilon = 10^{-4}$ and $N_s = 3000$. Figure 7.8 shows the CCDF from the original emulator as compared to that obtained after GPSS. It could be seen that using the GPE alone results in an overconfident estimate of the reliability of the system with $\bar{p}_F = 8.42 \times 10^{-4}$, based on 100 runs of SuS. The same analysis run on GPSS returned $\bar{p}_F = 5.71 \times 10^{-3}$. The coefficient of variation of two estimates was 15.6% and 10.5% for the GPE and GPSS

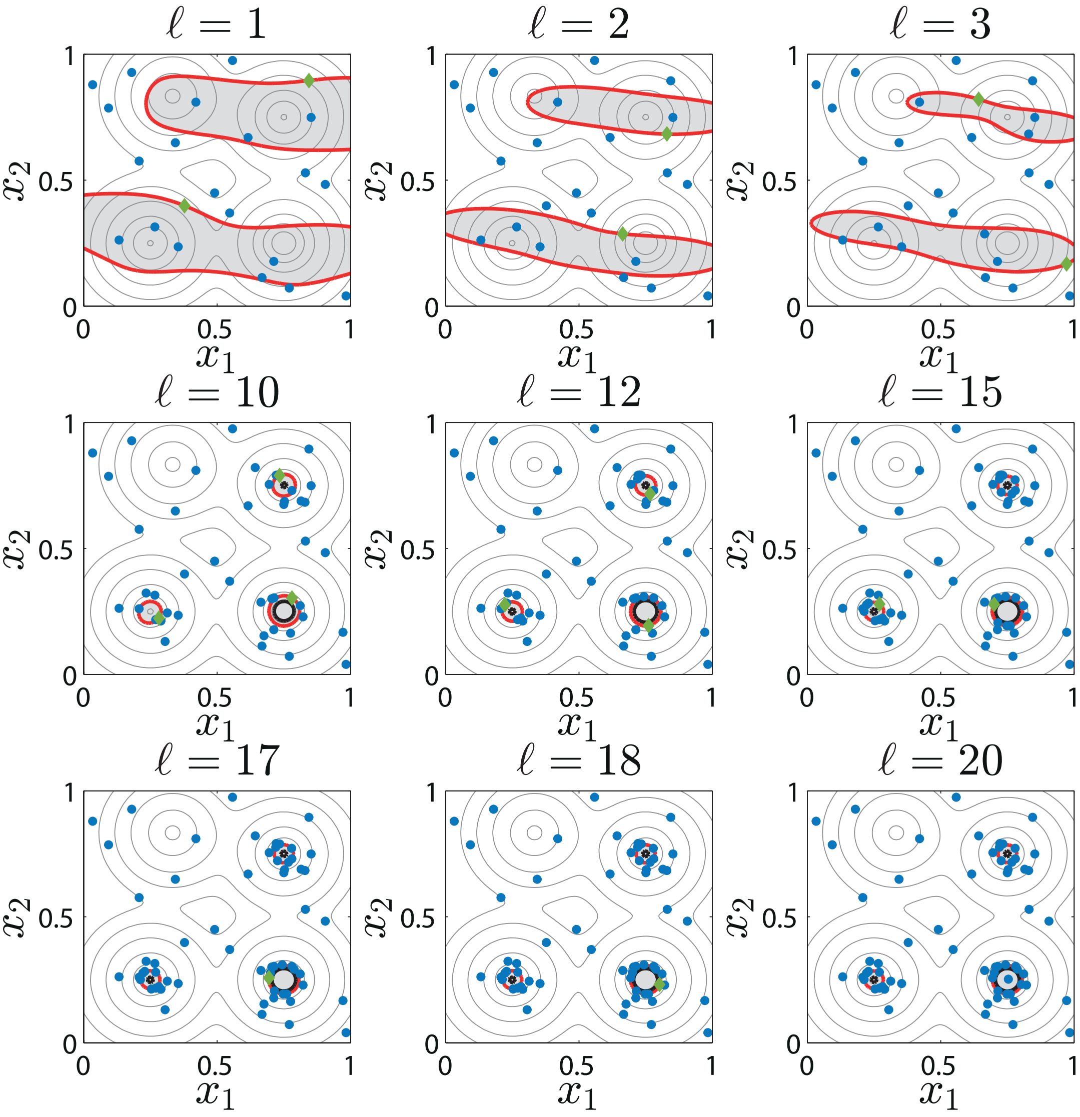


Figure 7.5: The performance of GPSS on the mixture function using Rule 4. The titles in each sub-figure signify the level of the algorithm. Red contours correspond to $\tilde{y}^*_\ell$ and black contours show $\tilde{F}_\ell$. New data points are plotted as diamonds. The number of training samples in each tile is $n = \{20, 22, 24, 44, 50, 59, 62, 63, 64\}$.

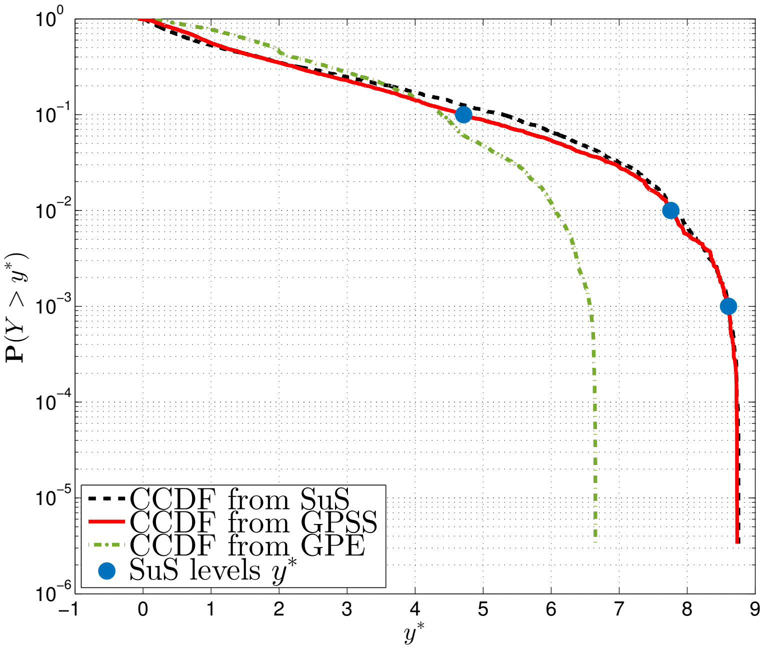


Figure 7.6: Complementary CDF of the mixture function according to the GPE (green), GPSS (red) and SuS (black). The blue circles show the intermediate levels of SuS. Each level was populated with 3000 samples.

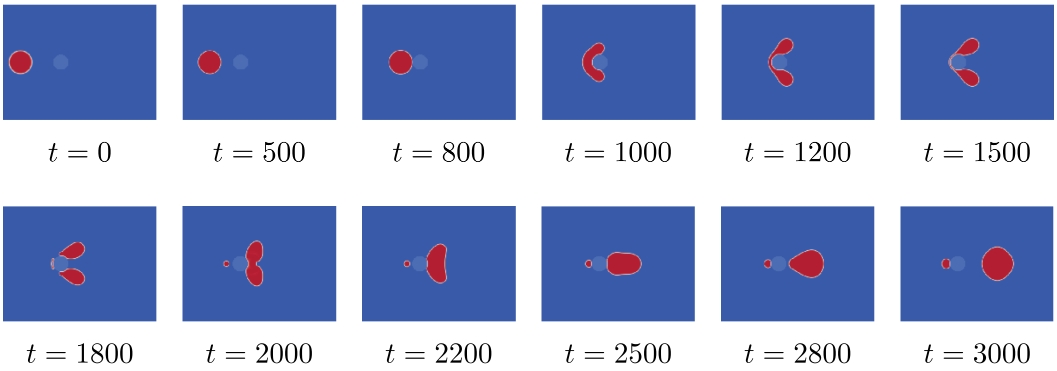


Figure 7.7: A selection of steps from the SCMP model used for preliminary emulation.

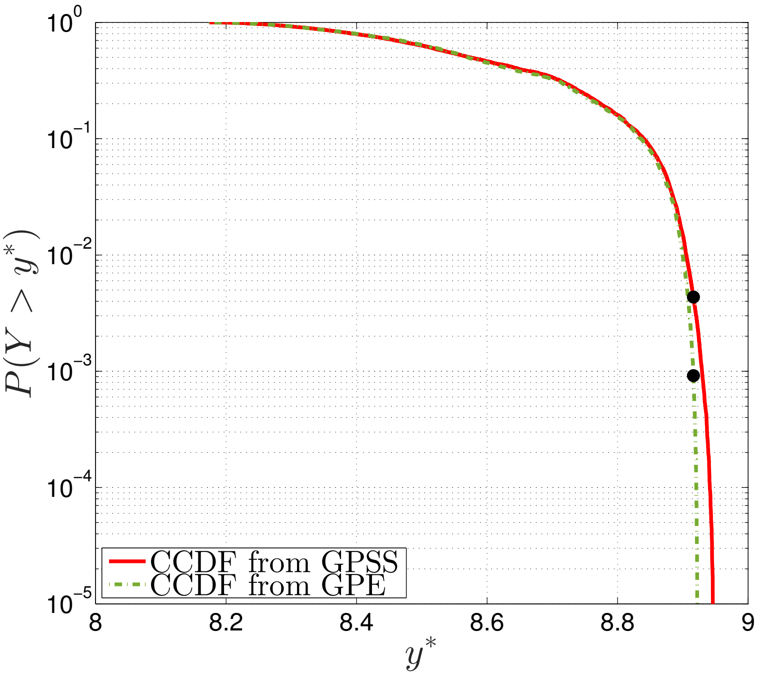


Figure 7.8: Complementary CDF of the LBM code according to the GPE (dash-dot) and GPSS (solid). The black circles show the probability of exceeding y^* estimated from the two surrogates.

respectively. The computational cost of the code prohibits direct Monte Carlo or subset simulation to be performed directly with the code. It should be noted that the small difference in p_F estimated using only the GPE and as opposed to GPSS is only due to the way the physical problem was set up. This LBM simulation is a synthetic example, whose output does not have a critical threshold in reality. Thus, defining $y^* = \max(\mathbf{y})$ to be the maximum of the training responses was a way to ensure that GPSS is tasked with providing information about an input region, the emulator has not seen during training and was not based on prior knowledge. It could be reasoned, that the closeness between the two p_F estimations is due to a smooth response of the LBM above y^*, which was accurately estimated by the GPE at $\ell = 1$. An interesting observation can be made on the features of the failure domain F. The estimates from the GPE and

GPSS differ not only quantitatively, but also qualitatively. Figure 7.9 shows a comparison between the samples lying in F according to the GPE and GPSS. The GPE identifies one continuous region in $\mathcal{X}$, which physically corresponds to failure occurring for big droplets challenged by moderately-sized fibres with varying wettability. On the other hand, when using GPSS to refine the emulator, three largely disjoint regions of the failure domain were identified. The left panel in Figure 7.9 provides some physical insight into the simulation dynamics. For instance, the bottom right cluster of failure points is the largest and corresponds to large droplets impacting fibres below the median value. For a large contact angle (*i.e.* a hydrophobic fibre), the drop splits and leaves as two smaller droplets. In contrast for $\theta_c \approx 160°$ it closes around the fibre and then detaches. The bottom left cluster of samples, corresponds to a situation where the contact angle between fibre and droplet is large enough for the droplet to split in two and go around the fibre. Finally, the top left cluster corresponds to more hydrophilic (still, $\theta_c > 130°$) fibres where the droplets initially attaches to the fibre, but has enough momentum to eventually split and depart. The region around $d_f = 0.7$, $\eta = 0.4$ which SuS populates with failure points when using the initial GPE can be ascribed to a local extrapolation issue, resolved by GPSS.

7.5 Chapter summary

The computational intensity of LBM may be sufficient to prohibit the execution of reliability analysis. Motivated by this challenge, the current chapter contributes a novel surrogate-model-based method for reliability analysis, called GPSS. The algorithm combines tested methods, namely subset simulation and Gaussian process emulation into an efficient tool for reducing the cost of reliability analysis of complex computer codes. Using the GPE as a naive substitute for the simulator results in the incorrect characterisation of the failure domain, where as

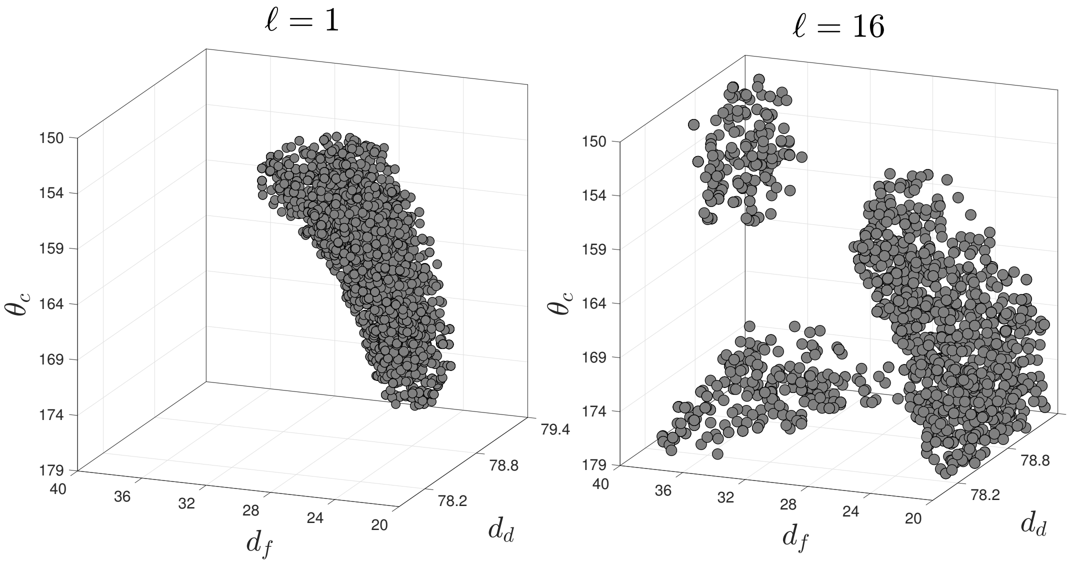


Figure 7.9: Samples in $\tilde{F}_\ell$ from the GPE ($\ell = 1$) and after GPSS ($\ell = 16$). GPSS not only estimates the probability of failure, but also provides an identification of the failure domain. The multimodality of F is clearly seen after $\ell = 16$ iterations. The size of the markers indicates the value of the response at that data point.

using SuS directly with the code is prohibitively time consuming if the underlying model is computationally expensive. The details of GPSS were presented and discussed outlining the current incarnation of the algorithm, which offers robust performance. By nature GPSS remains a flexible approach which the analyst can tailor to the problem-specific needs. To this end several level-generation rules were provided.

GPSS was used with the LBM code presented in Chapter 5 to not only calculate the probability of an excessive amount of water leaving the domain, but also to identify input configurations which lead to this event occurring. In this example the volume of water leaving the domain was chosen as the variable of interest for reliability analysis. In reality, the size of the droplets is more important as the primary task of the coalescer is to produce fewer larger droplet than those in the inlet. Another metric used in reliability analysis is the pressure drop, where a system with a small pressure drop is desirable. The outlook of the research envisions the analysis of systems where pressure differentials are incorporated.

This chapter introduced the last ingredient in the uncertainty quantification suite presented in this dissertation. It comprises the Gaussian process emulator (Chapter 3 and Chapter 5), a sensitivity analysis method (Chapter 6) and GPSS.

The topic of the next chapter is the industrial legacy of the research project, a tool for the virtual generation of non-woven media.

Algorithm 2 GPSS

```
Generate a space-filling plan, $\mathbf{X} \in \mathbb{R}^{n \times d}$ and form $\mathcal{D}_1 = \{\mathbf{X}_i, y_i = \eta(\mathbf{X}_i)\}$ for $i = 1 \dots n$
Train a GPE $\eta_1(\cdot)|\mathcal{D}_1, \hat{\boldsymbol{\theta}} \sim GP(m_1(\cdot), C_1(\cdot,\cdot))$ and validate it.
Assign values to all parameters for GPSS - $(p_0, N_s, y^*, \varepsilon)$
$\ell \leftarrow 1$
$N_{\tilde{F}_\ell} \leftarrow 0$
while $N_{\tilde{F}_\ell} < p_0 N_s$ do
    if $\ell = 1$ then
        $\tilde{y}^*_\ell = \frac{1}{n}\sum_{i=1}^{n} y_i$
    else
        $\tilde{y}^*_\ell = \frac{1}{2}\left[\frac{1}{n}\sum_{i=1}^{n} y_i \mathbb{I}(y_i) + \tilde{y}^*_{(\ell-1)}\right]$
    end if
    if $\tilde{y}^*_\ell > y^*$ then
        $\tilde{y}^*_\ell = y^*$
    end if
    $\tilde{F}_\ell = \{\mathbf{X}_i : \eta_\ell(\mathbf{X}_i)|\mathcal{D}_\ell, \hat{\boldsymbol{\theta}} > \tilde{y}^*_\ell\}, \forall i$
    Populate $\tilde{F}_\ell$ using SuS $\rightarrow \mathbf{X}_{\tilde{F}_\ell}$
    Form a candidate sample $\mathbf{X}_{cand} = \{\mathbf{X}_{\tilde{F}_\ell} : \mathbb{V}[\eta_\ell(\mathbf{X}_{\tilde{F}_\ell})|\mathcal{D}_\ell, \hat{\boldsymbol{\theta}}] > \varepsilon\}$
    Calculate $N_{\tilde{F}_\ell} = \sum \mathbb{I}_F(\mathbf{X}^{\mathsf{c}}_{cand})$
    Group $\mathbf{X}_{cand}$ in $k$ clusters $\{\mathbf{X}^{(1)}_{cand}, \dots, \mathbf{X}^{(k)}_{cand}\}$
    $\mathbf{X}_{add} \leftarrow \varnothing$
    for $j$ in $1 \dots k$ do
        $\mathbf{X}_{add} \leftarrow \mathbf{X}_{add} \cup \operatorname{argmax}_{\mathbf{X}^{(j)}_{cand}} \mathbb{E}\left[I(\mathbf{X}^{(j)}_{cand})\right]_\ell$
    end for
    $\mathcal{D}_{\ell+1} \leftarrow \mathcal{D}_\ell \cup \{\mathbf{X}_{add}, \eta(\mathbf{X}_{add})\}$
    Train a GPE $\eta_{\ell+1}(\cdot)|\mathcal{D}_{\ell+1}, \hat{\boldsymbol{\theta}} \sim GP(m_{\ell+1}(\cdot), C_{\ell+1}(\cdot,\cdot))$
    $\ell = \ell + 1$
end while
```

CHAPTER 8

MeshGen: software development for industry

8.1 Introduction

Lattice Boltzmann models, introduced in Chapter 2, can readily incorporate complex geometries in the simulation domain. This makes the use of techniques such as computer tomography (CT) [202] and its derivatives a desirable method for translating the physical geometry into the simulation. Computer tomography is a method in which a three-dimensional object or media is virtually reconstructed from sections generated by a penetrating wave. Using CT the non-woven bed can be represented as a collection of voxels (three dimensional pixels) in the simulation LBM domain. This method has been applied on a number of occasions in combination with LBM [203–205].

Another very widely used approach for imaging of micro structures is scanning electron microscopy (SEM) [206]. In SEM, a narrow beam of electrons is focused on the analysed surface which reflects a portion of the electrons, depending on its geometry. By scanning the surface at multiple locations the topology of the media can be reconstructed. Mukherjee et *al.* [207], used SEM and LBM to model the flow in porous fuel cells.

The methods above provide an accurate representation of the analysed sample, but lack the ability to represent those characteristics of the media that are used during manufacturing. Geometries produced with the CT or SEM methods can be used to predict the performance of existing filter meshes and to validate the fluid simulation tools. However, they can not be efficiently utilised to design and optimise new media, which is one of the main purposes behind computational studies. Despite this fact, scans of different existing media specimens can be post-processed and information about distributions of the properties of the media extracted. Once obtained, this statistical information can be supplied to a software which treats fibres as separate entities and generates them in the computational domain from the provided information. In this way, different characteristics of the bed can be studied through the analyses presented in Chapter 6 and Chapter 7, and optimised for using formal methods.

In Section 8.2 and Section 8.3 of this chapter different ways of virtual generation of non-woven beds are explored, which corresponds directly to the second research objective listed in Section 1.6. Following this discussion, in order to address the industrial aspect of the problem, a graphical user interface (GUI) software package is introduced in Section 8.4. This tool is designed to facilitate the work of the industrial practitioner in the design and analysis of new media and corresponds to the second deliverable from the research; see Section 1.6.

8.2 Analysis of non-woven fibrous media

Parker Hannifin uses SEM in the analysis of non-woven media. In contrast to tomographic methods, SEM does not provide a three-dimensional model of the

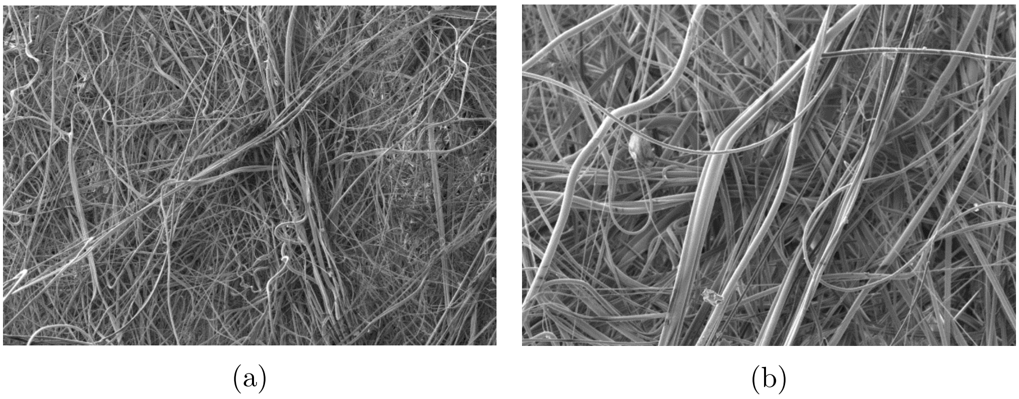


Figure 8.1: SEM images of an meltblown media at different magnifications - (a) 100x magnification, (a) 400x magnification. Fibres with different morphologies can be observed (source: Parker Hannifin Manufacturing UK).

media, but instead a detailed projection of the surface is obtained through the reflection of electrons off fibres. Example SEM images with different magnification are shown in Figure 8.1. It should be noted that the scans provide images of more than one layer of the media, giving it a three-dimensional-like appearance. The depth of the scan is controlled by adjusting the power of the electron beam [208]. Information about a variety of geometric features can be extracted from SEM images, including fibre size [209], curvature [210] and orientation [211]. Due to its multi-level depth, however, reliable measurements of porosity cannot be obtained. Consider the scenario shown in Figure 8.2, where an SEM image is processed into a black-and-white map of the fibre layers. The porosity, ε can be calculated from its standard definition

$$\varepsilon = \left(1 - \frac{V_{solid}}{V_{total}}\right) \times 100\% \tag{8.1}$$

where V_{solid} is the volume of the bed occupied by fibres and V_{total} is its total volume. In Figure 8.2 the two volumes are replaced by areas, due to the two-

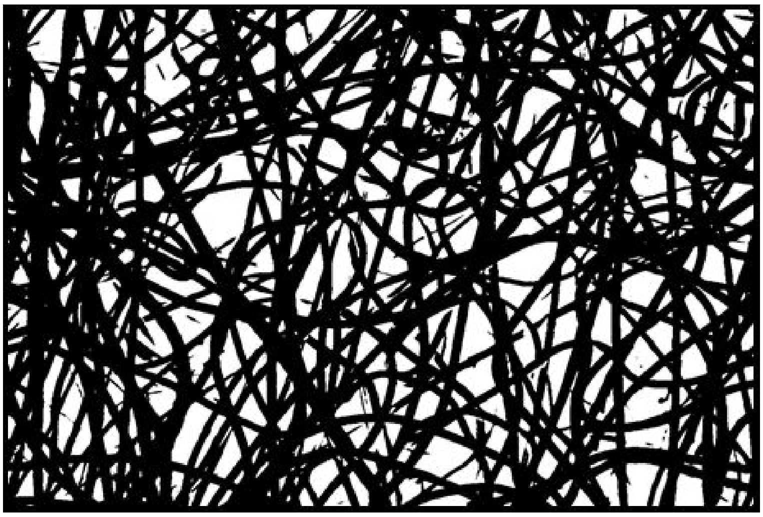


Figure 8.2: A post-processed SEM image. A threshold was used to assign values black or white memberships to each pixel (source: Parker Hannifin Manufacturing UK).

dimensional nature of the image and are calculated via summations as

$$V_{solid} = \sum_{x=1}^{w} \sum_{y=1}^{h} \mathbb{I}_b(p(x,y)) \tag{8.2}$$

$$V_{total} = w \times h \tag{8.3}$$

where w and h are the width and height of the image, respectively and $\mathbb{I}_b$ is the indicator function which is equal to one if the pixel at a specific location, $p(x,y)$ is black and zero otherwise. Performing this analysis results in a porosity value of $\varepsilon \approx 40\%$, which is a significant deviation from practical values. In order to measure the porosity of a porous bed one can determine the solidity (fraction of solids) of the specimen, denoted by ϕ as

$$\phi = \frac{\rho_{fabric}}{\rho_{fibre}} \tag{8.4}$$

where $\rho_{fabric} = m/t$ is the density of fabric taken as the ratio between its weight per unit area, m and its thickness, t. The density of the fibre, ρ_{fibre} is assumed to be equal to the bulk density of the material it was made of. Once ϕ is known the porosity is calculated as $\varepsilon = (1 - \phi) \times 100\%$. Other methods for measuring ε exist as well, as discussed in [212].

Another commonly used technique to predict filter media performance is characterising its *pore size distributions* (PSD). Characterisation of pore openings within the media can be performed both experimentally and computationally, with an outcome dependent on the type of analysis and technique used in the process [213]. A widespread approach for obtaining PSD from image data is by determining the largest sphere/circle that fits into each pore and contains no solid material [214]. This method provides what is known as a *geometric pore size distribution*. The accuracy of this method depends on the morphology of the pores and in particular, its deviation from spherical/cylindrical shape. Another method, proposed by [215] achieves a closer match with experimental results. A method based on random walk is presented in [216]. All of these methods analyse the PSD after the media has been generated virtually. Creating a virtual mesh with a prescribed PSD may prove useful for analysing its influence on the separation efficiency. To the best knowledge of the author of this dissertation there are no announced efforts or proposed algorithms to this end.

8.3 Modelling of non-woven media

8.3.1 Empirical modelling

Attempts to analyse the flow through complex media have been made long before computational models were available. The earliest analytic expressions were derived to predict characteristics of the flow through packed beds, such as sand and soil [217]. One of the earliest models of this is Darcy's law, which relates the pressure gradient across a packed bed, denoted ∇p to different properties of the flow and media. The law reads

$$\nabla p = -\frac{\mu}{k}(u_0 + \rho g) \tag{8.5}$$

where μ is the fluid dynamic viscosity, ρ is its density, u_0 is the superficial flow speed, g is the gravity constant and k is the *permeability* of the media. Darcy's law is one of the fundamental equations for flow through porous medium. A major limitation of Eq. 8.5 is the fact that the whole bed is described by a single parameter, k, which needs to be empirically determined. A way to determine the value of the permeability of the bed is through the Kozeny-Carman equation [217]

$$k = \frac{1}{KS_0^2}\frac{\varepsilon^3}{(1-\varepsilon)^2} \tag{8.6}$$

where K is a factor to account for the tortuosity of the bed, S_0 is the media surface area per unit volume and ε is the porosity of the mesh. The issue with Eq. 8.6 is that it attempts to define one unknown property of the filter (k) in terms of other unknown properties. The Kozeny-Carman equation also assumes the bed is formed of bundled tubes of particular orientation. Finally, experimental work has been used to validate Eq. 8.6 for beds with $\varepsilon < 80\%$ [217].

As an attempt to extend the analytic tool set introduced above the so called *cell model theory* was developed. In cell model theory, the mesh is represented as an array of circular cylinders, each enclosed in a cell of fluid. The development was based on a flow past a single fibre and eliminated the requirement for $\varepsilon < 80\%$. Cell model theory is still widely used for preliminary filter design. The interested reader is referred to [217].

8.3.2 Computational modelling

The empirical approaches described above rely on a number of assumptions and approximations, some of which are quite far from reality when analysing non-woven webs. From one side, the study of filter beds can benefit from techniques, such as SEM, which supplies information about the mesh properties. From another side, the availability of computational power makes it possible to directly simulate the non-woven beds, instead of relying on approximations. An early example of such work is the method presented in [218], where the authors assume the filter is formed of fibrous layers of thickness twice that of individual fibres. They than draw a number of straight lines of random length and orientation, using Monte Carlo sampling and place them in the filter domain.

Sampling and depositing fibres sequentially is a widely used approach in modelling filter media. The principle is that fibre properties are sampled from underlying distributions and used to form fibres. The way fibre morphology is defined varies between implementations. For instance, the authors of [219] present a model in which electrospun fibres are represented as ellipses of fixed aspect ratio and uses fibre number density as a proxy for porosity. Different augmentations of the simple random sampling exist. A method for simulating electrospun meshes is described in [220]. The authors simulate each fibre as a straight cylinder and deposit fibres sequentially until a desired porosity is obtained. In this approach the fibres are lowered one on top of the others to prevent the crossover between them. The direction of each fibre is determined through a procedure referred to as *μ-randomness* with a predetermined orientation distribution function. Another method based on the one above is described in [221], with the modification that fibres are allowed to cross each other.

Different approaches to modelling the bed altogether also exist. For example, the authors of [222] adopt a method, whereby after analysing image data of the non-woven a stochastic geometric model is employed to represent the media. In their application, the Poisson line process is used because fibres are assumed to be circular cylinders. The parameters of the model are chosen such that the simulated porosity and fibre diameter distribution match those of the real sample.

The model is then used in an optimisation routine. In [223] an SEM image is transformed into a virtual bed, by tracing the centrelines of each fibre in the mesh and applying thickness to those centrelines. The authors use a threshold approach to pre-process the SEM image, extracting only the top layer of the mesh. Methods similar to this one are useful for analysing existing media, but since they do not generate separate fibres they cannot be used in the design process.

8.4 The MeshGENerator

Practical usability of any tool and procedure is of paramount importance when industry is concerned. The virtual mesh generation tool, called MeshGen was designed and implemented during the course of this PhD project with that purpose in mind. At the same time, the ease of use should not compromise the technical capabilities of the routine. Thus a balance between performance and convenience must be achieved. The tool is based on the idea of sequential modelling of the fibres. Its implementation is discussed in more detail below.

8.4.1 Description and generation

As already discussed in Chapter 4 and demonstrated in Chapter 5, the lattice Boltzmann model, used for the purposes of this dissertation, reads in plain text or .vti files containing the location of solid nodes and outputs .vti files which can be read with open-source platforms such as ParaView [125]. These facts dictate the main input/output features of MeshGen: it should be able to turn information supplied in a human-friendly manner to a .vti file, which can either be fed into an LBM simulation or be used to visualise and analyse the mesh itself.

In the main mode of operation, each fibre in MeshGen consists of a straight line and a circular cross section. Fibres are defined by the following features, addressed in more detail below:

1. **Starting position** - three positive spatial coordinates. *e.g.* $\mathbf{x} = [20, 50, 15]$.

2. **Direction** - three component vector containing the cosine of the angle between the positive unit vector for each axis and the fibre. For example, $\mathbf{r} = [0, 1, 0]$ represents a fibre with $\theta_y = \cos^{-1}(1) = 0°$, where θ_y is the angle between the fibre and the positive y-axis.

3. **Length** - a scalar defining the length of the fibre from the starting position along the its direction.

4. **Radius** - a scalar determining the constant width of the fibre transverse to its length.

The features above give a considerable amount of freedom in the definition of the final mesh, while preserving a clear set of parameters that can be studied and optimised. The package supports both deterministic and stochastic definitions for each property. For the purposes of demonstration, the parameters will be defined stochastically[1]. To build each fibre a starting position within the initially defined domain is chosen from a suitable distribution. Similarly, a direction and a length for the fibre are also generated. The process then proceeds by constructing each fibre in the mesh from spherical primitives, concatenated along the previously selected direction. The mesh is constructed by adding fibres with given properties until the desired porosity of the bed is achieved. This approach is favoured over a prior computation of the volume of fibres, as it may result in significant errors due to discretization effects on the rectangular lattice of the domain. Fibres are allowed to cross over as this is often observed in SEM images of meltblown meshes.

Meltblowing [12] can also produce curved fibres as seen in Figure 8.1. The use of spherical fibre elements allows the geometry to be readily extended from

[1] For meshes generated from deterministic values with some added uncertainty see the results in Section 5.4

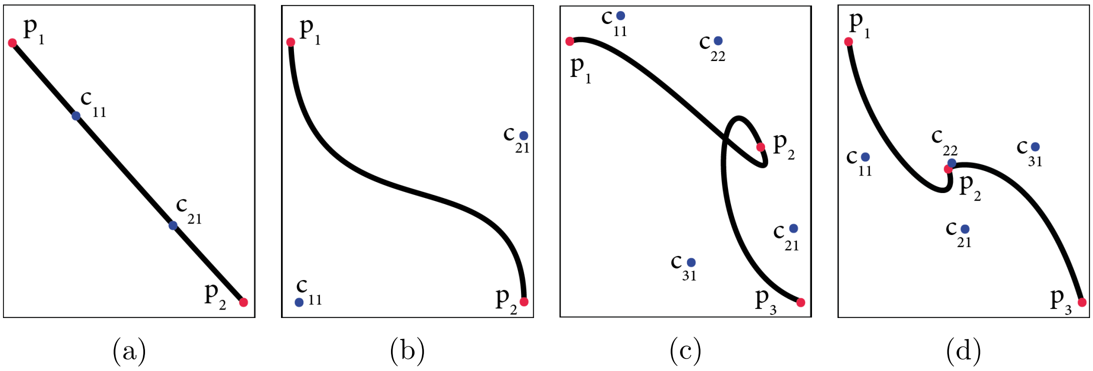


Figure 8.3: Third order Bézier curves. The locations of the two control points have a strong influence on the end shape of the curve. They can (a) reproduce a straight line; (b) achieve a relatively complex shape. The curves can also be appended as seen in (c) and(d). In all four panels the composite notation c_{ij} is used to denote the i^{th} control point on the j^{th} curve.

straight fibres to ones with a degree of curvature. This is achieved by representing each fibre as a Bézier curve [224]. Different types of Bézier curves exist, but a widely used choice is the third order form, described by

$$P(t) = (1-t)^3 p_1 + 3(1-t)^2 t c_1 + 3(1-t) t^2 c_2 + t^3 p_2 \tag{8.7}$$

where $t \in [0, 1]$ is the normalized location along the curve, p_i are the end points of the curve located at $t = 0$ for $i = 1$ and $t = 1$ for $p = 2$. Additionally, c_i are the *control points* of the curve. Their influence can be seen in Figure 8.3. Another useful property of Bézier curves is that they can be appended to each other with smooth transition from one curve to the next. This can be done to achieve a curve with greater flexibility than that of its constituents. The characterisation and measurement of the radius of curvature of fibrous beds is a challenging endeavour and is subjected to different uncertainties [210]. Therefore there is an ongoing effort to facilitate the transfer of information from experimental measurements to MeshGen.

8.4.2 Implementation and Capabilities

In order to enable the efficient generation of large-scale media samples, the low-level routines which build the fibres and write the information to files were developed in C [123]. This choice was made due to the fact that both of the above procedure involve the extensive use of cyclic operators, such as *for* and *while* loops. With C being a lower-level, compiled language, execution of such constructs is considerably faster than in interpreted languages. Even though C offers good efficiency in terms of basic computing, it requires a significant amount of preprocessing to be used for such operations as probability distribution construction and sampling, array handling and GUI development. Refer to Chapter 4 for a discussion of other advantages and disadvantages of the C language. These two facts motivated the choice of MATLAB® as a high-level wrapper around the core algorithm. One feature of MATLAB®, which is particularly important in this case is the *MEX* compiler. This functionality provides integration between MATLAB® and C, such that the strengths of both languages can be utilised.

For the purposes of reproducibility MeshGen can export the information of the generated bed with details on each fibre. This information can be passed on to an optimisation algorithm for adjustment and later loaded back into the package for visualisation. In the package the user is able to control all four geometric properties, as well as the wall affinity of the fibres (as described in Chapter 5) and the target porosity of the bed. A snapshot of the software immediately after start-up is shown on Figure 8.4. At this point the user is presented with the choice of generating a new mesh, in which case they must specify the size of the domain to enable all other options, or loading mesh information from an existing file. Once either of these options is taken the Generate button becomes available allowing the user to generate the mesh and export it to a .vti file for further use.

If a new set of fibres is to be generated, after specifying the domain size, the user can either manually input information for each feature of the geometry or

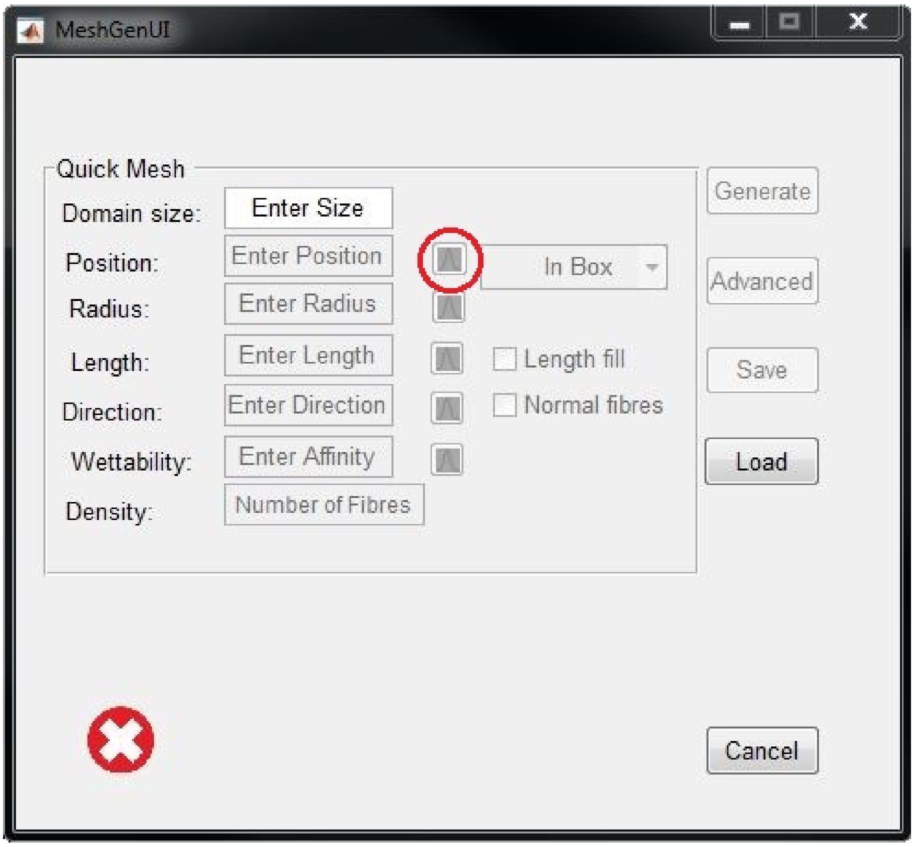
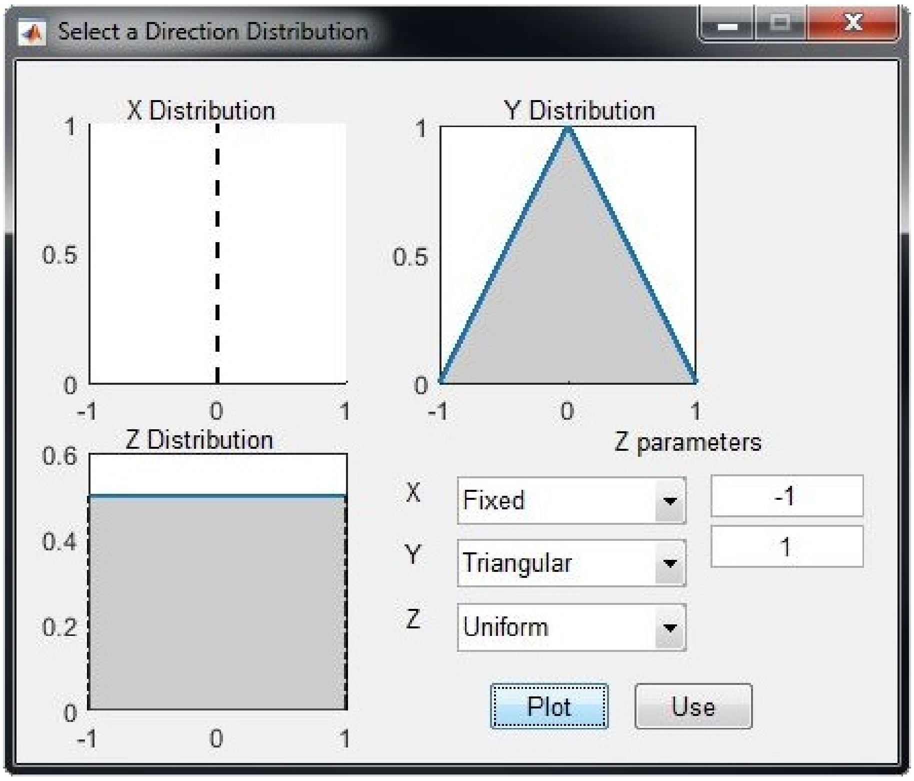


Figure 8.4: MeshGen upon start-up. The user is required to input the size of the domain to enable all other options. They can also load mesh information from an existing file. Each option supports a stochastic definition through the Distribution selector, circled in red.

select them thorough the Distribution selector, accessible via the button with a Gaussian distribution icon, circled in red on Figure 8.4. Each property has its own Distribution selector. Three dimensional characteristics, such as position and direction offer a distribution for each coordinate as seen on Figure 8.5, whereas scalar-valued features, such as radius and length have a single distribution, shown on Figure 8.6. The user has control over the type of the distribution and its parameters. Three distributions are implemented in the current version of MeshGen: uniform, triangular and truncated Gaussian. Furthermore the user can also choose to fix any of the properties through the Distribution selector. After each distribution is defined, the user is required to plot them to verify their choice and to confirm the selection to the software. Once valid information for all relevant coordinates is selected the user can proceed by pressing the Use button which records their choice and takes them back to the main window. MeshGen allows the user to have a mixture of probabilistic and deterministic properties. The state of each property is displayed in its box with a number representing fixed, deterministic value and a distribution abbreviation with its parameter values denoting a the use of a distribution. After selecting all necessary information, MeshGen appears as shown on Figure 8.7. The software indicates that all information has been properly supplied by displaying a green tick mark and enabling the Generate button. Upon pressing the Generate button, MeshGen uses MATLAB® to sample the distribution of each property a number of times and then passes this information on to the C core which generates the fibres and writes the output file. Once generation is finished the user is presented with a summary of the results as shown in Figure 8.8. The results of the generation can be visualised in ParaView. The output of MeshGen for the setting presented in Figure 8.7 is shown in Figure 8.9. There, the effect of having the direction along the x-axis set to zero, as shown in Figure 8.5, is visible in that the fibres appear to form layers in the x-direction.



Select a Direction Distribution

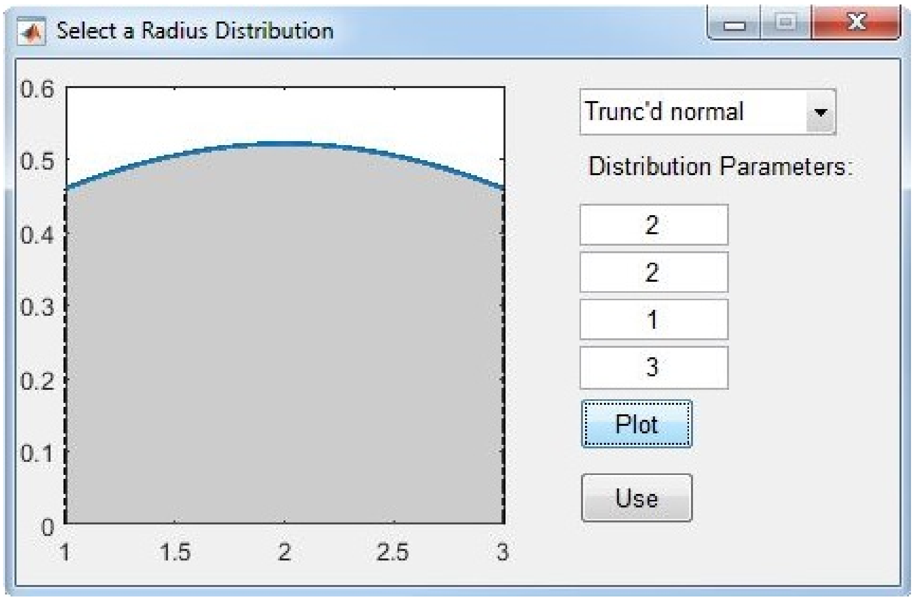
Z parameters

X Fixed -1

Y Triangular 1

Z Uniform

Figure 8.5: A Distribution selector for vector-valued properties. The user can select the type of distribution for each coordinate, its parameters, plot it to verify it is correct and proceed using it.



Select a Radius Distribution

Trunc'd normal

Distribution Parameters:

2

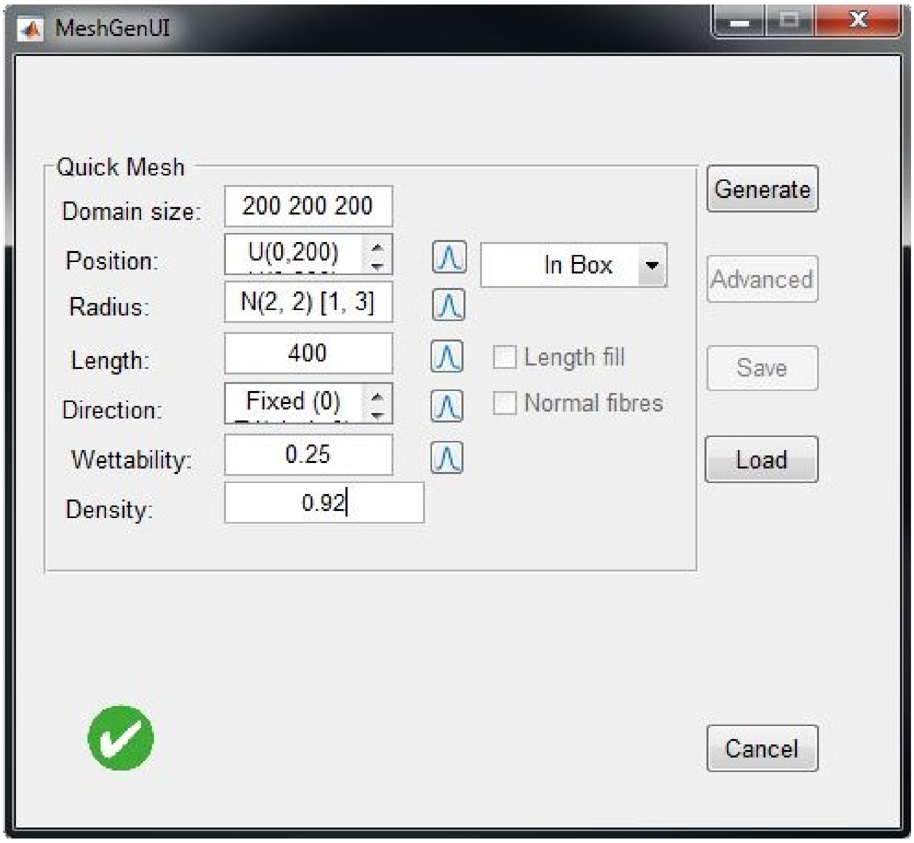
2

1

3

Figure 8.6: A Distribution selector for scalar-valued properties. The user can select the type of distribution, its parameters, plot it to verify it is correct and proceed using it.

As mentioned above MeshGen is also capable of generating curved fibres. This can be achieved through the Advanced options. A sample of media with curved fibres is shown in Figure 8.10. Another capability accessible in the Advanced set of tools is the generation of combined meshes from existing files. This feature is particularly useful when simulating beds with a porosity gradient. In that case separate files can be generated and then combined to form one mesh with the desired properties. A sample of the output of this process in shown in Figure 8.11. There the porosity of the bed increases from 85% to 96% and the fibre diameter also increases from 2 to 6 µm. This particular mesh was generated with fibres stemming from the bottom of the domain and oriented at ±45° angles from the y-axis.



Domain size: 200 200 200

Position: U(0,200) In Box

Radius: N(2, 2) [1, 3]

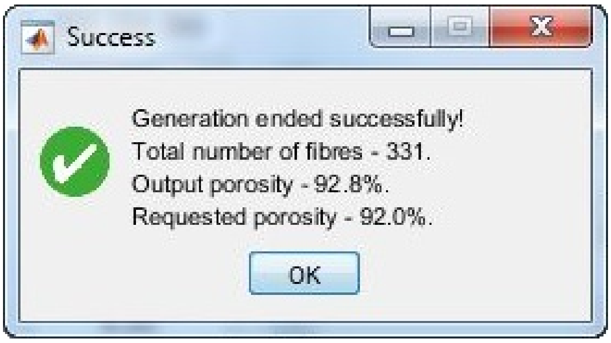
Length: 400 □ Length fill

Direction: Fixed (0) □ Normal fibres

Wettability: 0.25

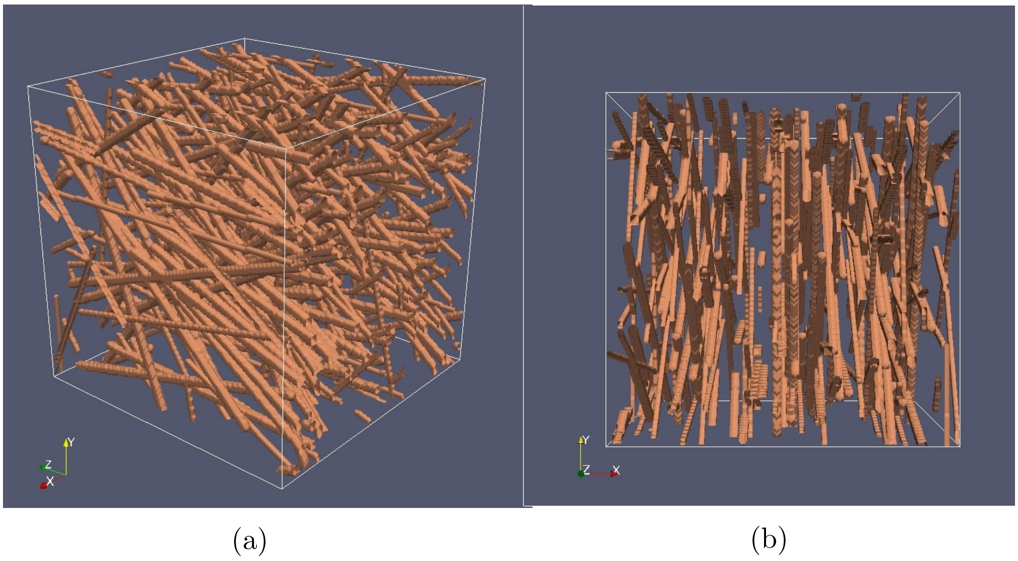
Density: 0.92

Figure 8.7: The state of MeshGen with all information correctly supplied and prior to generation. A mixture of fixed and distribution-defined properties can be seen.



Generation ended successfully!
Total number of fibres - 331.
Output porosity - 92.8%.
Requested porosity - 92.0%.

Figure 8.8: A summary of the fibre generation presented to the user. The Requested and Output porosities may differ slightly due to the way fibre information is sampled from distributions.



(a) (b)

Figure 8.9: Output from MeshGen for the set-up in Figure 8.7. An overview is shown in (a). The direction in the x-axis was set to 0 in Figure 8.5 and the effect can be clearly seen in (b) - the media looks layered along the x-axis.

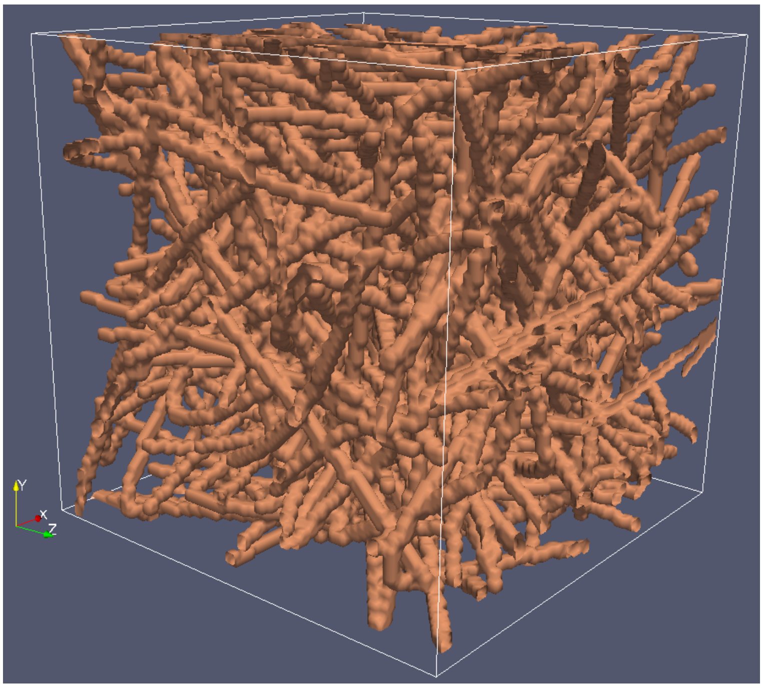
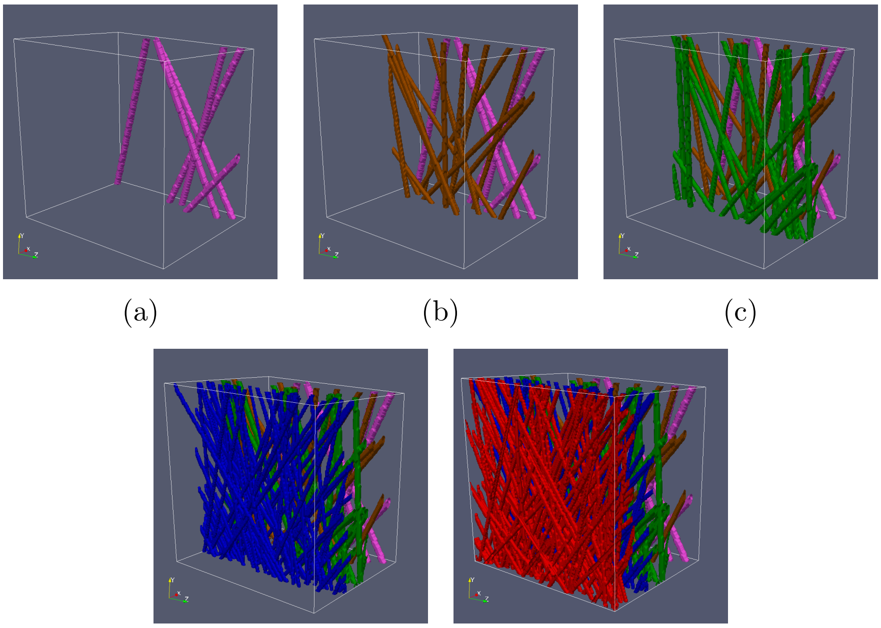


Figure 8.10: A sample mesh with curved fibres. The porosity of the specimen is 82%.



(a) (b) (c)

(d) (e)

Figure 8.11: A mesh with gradient in both porosity and fibre diameter. The slice at the outlet, shown in (a) has fibre diameter of 6 lattice units and porosity of 96%. The next two slices, shown in (b) and (c) have diameter of 4 lattice units and porosity of 94% and 92%, respectively. Finally, the layers at the inlet, depicted on (d) and (e) have fibre diameter of 2 lattice units and porosity of 88% and 85%, respectively.

8.5 Chapter summary

This chapter reviewed some of the existing techniques for modelling complex media and how these models can be incorporated in the lattice Boltzmann framework. MeshGen, a virtual non-woven media generator was introduced. MeshGen is a tool developed in C+MEX and MATLAB$^{®}$, providing a combination between high-level capabilities and efficiency. The package treats the media on two levels - as a porous bed capturing properties such as bulk density and thickness, and as a collection of fibres, allowing it to be used in analysis and optimisation routines. MeshGen is part of the research project industrial legacy, emphasizing not only the academic, but also the practical significance of the research in this dissertation.

Some of the planned improvements in MeshGen include the capability of seamlessly generating fibre beds with multiple diameter distributions, enhanced control over the generation of curved fibres and the incorporation of tools for modelling precise pore size distributions. These and other developments will provide a basis for ongoing communication and collaboration between Parker Hannifin and academia.

CHAPTER 9

Conclusion

The research in this dissertation was motivated by the depletion of natural resources and the subsequent increasingly stringent regulations for air pollution and biofuel cleanliness. The computational aspects of developing optimal filtration systems were discussed in detail in this dissertation. The work carried out as part of the present research project is within the realm of computational fluid dynamics and uncertainty quantification, and has both industrial and academic significance. Throughout this work, computational, statistical and experimental aspects of the design of filtration equipment were covered. This chapter serves to recapitulate the main findings and contributions of the research and to provide future research directions.

9.1 Summary of completed work

After the motivation and introduction to the scope of the work in Chapter 1, Chapter 2 introduced the lattice Boltzmann modelling framework. The physical foundations of LBM were introduced and a history of its development was presented. The chapter also discussed how LBM can be used for the simulation of multiphase flows and several commonly used paradigms were compared. Different boundary conditions in the model, its main sources of error, as well as its advantages over conventional computational fluid dynamics methods were discussed.

In Chapter 3, the concept of Gaussian process emulation was introduced, together with relevant supporting theory and nomenclature. Aspects of constructing and optimising GPEs and validating their quality were discussed. The GPE code developed specifically for the purposes of this dissertation was benchmarked with illustrative examples.

Chapter 4 discussed important aspects of model development, efficient implementation and dimensional analysis for LBM. There, a discussion about introducing units into models of multiphase flow through complex media was provided. Some assumptions about the process and their justifications were shown. In the same chapter a method for the detection of droplets in LBM simulations was proposed and demonstrated. This method is based on spatial clustering and provides a way of efficiently analysing the output of the simulation.

The main body of numerical experiments was presented in Chapter 5. There, a discussion about the incremental development of the filter model was provided, together with information about different input characteristics and distributions. An SCMP model was used for single fibre simulations, but was found inadequate for multi-body studies, where it was substituted for the MCMP formulation. In these multi-body experiments, it was found that beds with porosities as high as 96 % lead to a shift in the droplet size distribution in the filter outlet. However, such filters also let some of the finer droplets escape unchallenged. Later in the chapter, a way to validate LBM was also shown. The model-development work was concluded with the construction of GPEs for the lattice Boltzmann filter model. Aspects of GP emulation discussed in Chapter 3 were applied in practice to reduce the computational cost of the filter simulator.

In Chapter 6 the development of a uncertainty quantification framework for the filter model was begun, with an introduction to sensitivity analysis. It was shown that preforming sensitivity analysis directly with the filter model results in prohibitively long running times. To alleviate this the GPEs introduced and built in previous chapters were employed to reduce the computational cost of the model. The GPE allows this to be achieved, whilst accounting for the uncertainty introduced when using an approximation. The capability of the presented SA framework to work with models having high-dimensional input and output, as well as its generality was demonstrated through the use of two more industrially-scaled case studies.

In Chapter 7 a novel method for efficient reliability analysis was proposed. The approach, termed GPSS, provides a way of carrying out practical reliability computations, whilst ensuring that the underlying model is used in an optimal way. Initially, GPSS was benchmarked with two problems specifically constructed to challenge different capabilities of the algorithm. Later on the GPSS was applied to estimate the probability of failure of the filter model, and to provide an indication of input configurations leading to this failure.

Finally, in Chapter 8 a tool for the virtual generation of non-woven filter beds was presented. The tool, called MeshGen, is capable of generating meshes with a wide range of fibre and filter morphologies. MeshGen was constructed in a way which provides efficient operation and ease of use. The tool will be provided to Praker Hannifin to supplement further studies in the filed.

9.2 Summary of contributions

The main contributions of this dissertation are the following:

- **Detection algorithm.** An algorithm for the detection of escaping water droplets in filter simulations was proposed and applied to numerical experiments. The method provides two main advantages to the analyst. First, it eliminates the need for post-simulation analysis to extract and measure effluent droplet distributions. Second, for large simulations, the method enables the reduction in memory usage by eliminating the need for information about the whole flow field to be stored (Chapter 4).

- **Efficient filter models.** A systematic way to develop efficient filtration models from both software and hardware perspective was demonstrated throughout this dissertation. The combination of suitably selected programming environments, computational models, and hardware-enabling paradigms, such as OpenMP, were shown to provide reasonable computational requirements. In addition the developed filter models feature the ability to restart simulations, potentially having a significant impact on their usability (Chapter 4 and Chapter 5).

- **Model validation.** A way to validate the lattice Boltzmann model for a single fibre, single drop interactions was conceived. An experimental rig was designed, built and used to carry out the study, which showed good correspondence between the produced lattice Boltzmann model and experiments (Chapter 5).

- **Cost reduction of sensitivity analysis.** The GPE was used with LBM to enable the execution of the inherently costly sensitivity analysis. The GPE provides a statistical approximation to the output of the LBM filter model which can be used as an inexpensive substitution and provides a closed form expression for the metamodelling uncertainty. It was demonstrated how the GPE can be used to provide affordable computation of Sobol' indices with associated error estimates (Chapter 6).

- **Efficient reliability analysis.** A novel method for performing inexpensive reliability analysis with computationally demanding models was proposed. An important problem which had to be addressed in the new algorithm was how can the approximation provided by the GPE be efficiently supplied with information from the model. This challenge was resolved by integrating subset simulation and Bayesian optimisation methods with the GPE to minimize the time expenditure of reliability studies (Chapter 7).

- **A method and tool for virtual mesh generation.** The need to have virtual filter bed representations amenable to analysis and optimisation, necessitated the development of an efficient, yet practical software tool. A package, called MeshGen was developed for the industrial partner through the course of this PhD. The tool provides full compatibility with the developed LBM codes and has a user-friendly front end which maximizes productivity. (Chapter 8).

9.3 Research outlook

The work conducted in this dissertation leads to several research directions to be considered in the future.

Firstly, the development of larger, three-dimensional filter models is seen as a direct future step, following the modelling hierarchy presented in this chapter. Three-dimensional models will be capable of simulating the effect of several important filter parameters, such as mesh thickness and fibre orientation, among others. The transition to this type of LB models will be facilitated by the use of MeshGen. From an academic perspective, the two-dimensional model can be further used to develop better understanding of coalescence dynamics.

Secondly, the ability to simulate the pressure differential across the bed can be investigated. This can be achieved through the incorporation of pressure boundaries [225, 226]. As discussed in Chapter 1, pressure drop is one of the indicators of bed saturation and is frequently experimented with. Enabling the simulation of pressure drop across the filter bed will enable the use of the model in multidisciplinary design optimisation and reliability-based design optimisation, with GPSS.

Thirdly, the three-dimensional models can be used in large-scale validation studies. This can be done by either performing dedicated experiments or using existing experimental data. At the same time the results from these studies can be used to underpin model calibration efforts. There exist a number of techniques to perform calibration [42, 136] as discussed in Chapter 1 and Chapter 5.

Fourthly, emulation for the LB filter model can be approached from a number of different directions. For instance, multi-output [139], time-dependent [137] and distributional emulation may be used, among others. Future efforts related to emulation will reflect directly on the practical usefulness of the filter model.

Finally, the virtual mesh generator can be augmented with further capabilities. One such feature is the generation of beds with fibre properties coming from two or more distributions, which reflects experimental efforts [17]. Furthermore, the ability to generate filter beds with a specified pore size distribution can be added an extension to the tool. MeshGen can also benefit from an enhanced user interface for the generation of beds with curved fibres.

All of these developments are seen as potential basis for future collaboration with Parker Hannifin in the development of optimal filtration equipment.

9.4 Published work

The following literature and awards were generated from the present dissertation.

9.4.1 Journal papers

1. P. Hristov, F. DiazDelaO, E. Saavedra Flores, C. Guzmán, U. Farooq, Probabilistic sensitivity analysis to understand the influence of micromechanical properties of wood on its macroscopic response, Composite Structures 181 (2017) 229–239.

9.4.2 Conference papers

1. P. O. Hristov, F. A. Diazdelao, K. J. Kubiak, U. Farooq, Reliability Analysis and Surrogate Modelling of Biodiesel Filters, in: Proceedings of the 26th European Safety and Reliability Conference (ESREL), Glasgow, 2016, pp. 390–395.

2. P. Hristov, F. DiazDelaO, K. Kubiak, U. Farooq, Adaptive emulation-based reliability analysis, in: Proceedings of the 2nd International Conference on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP), Rhodes, 2017, pp. 198–211.

9.4.3 Awards

1. "Best Image from a PhD student", ESREL 2016 Images of Risk visualisation competition. Refer to Figure 9.1.

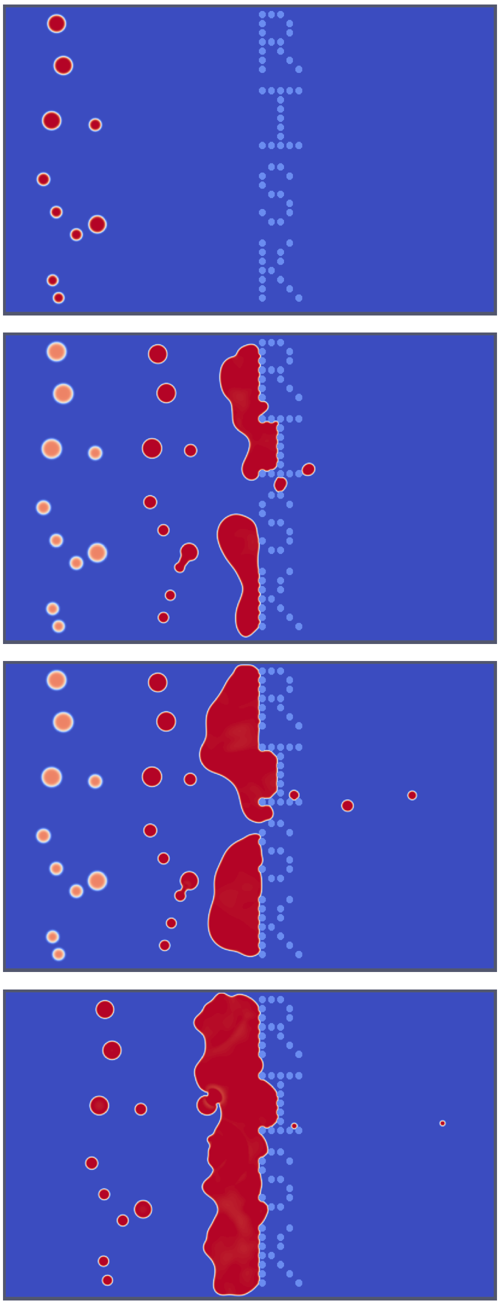


Figure 9.1: Improper fibre (light blue) arrangement can lead to water (red) being released into the fuel stream (blue) and filter clogging. Water emulsified in diesel could have detrimental effect on engine performance and result in potentially costly damage to the system. Lattice Boltzmann models allow simulation of multicomponent flows through complex media which enables the design optimization of porous fibrous filters.

9.5 Work under review

At the time of submission of this dissertation, the following work has been submitted and is currently under review.

9.5.1 Journal papers

1. P. O. Hristov, F. A. DiazDelaO, K. J. Kubiak, U. Farooq, Adaptive Gaussian Process Emulators for Efficient Reliability Analysis, Under Review.

Bibliography

[1] S. Shafiee, E. Topal, When will fossil fuel reserves be diminished?, Energy Policy 37 (2009) 181–189.

[2] I. Capellán-Pérez, M. Mediavilla, C. de Castro, Ó. Carpintero, L. J. Miguel, Fossil fuel depletion and socio-economic scenarios: An integrated approach, Energy 77 (2014) 641–666.

[3] J. Wang, L. Feng, X. Tang, Y. Bentley, M. Höök, The implications of fossil fuel supply constraints on climate change projections: A supply-side analysis, Futures 86 (2017) 58–72.

[4] A. Sarin, Biodiesel: Production and Properties, The Royal Society of Chemistry, 2012.

[5] P. B. L. Fregolente, L. V. Fregolente, M. R. Wolf Maciel, Water Content in Biodiesel, Diesel, and Biodiesel-Diesel Blends, Journal of Chemical & Engineering Data 57 (2012) 1817–1821.

[6] T. Sparks, G. G. Chase, Other Separation Processes and Equipment, in: Filters and Filtration Handbook, Elsevier, 2016, Ch. 6, pp. 361—-382.

[7] P.-Z. Yang, D.-Z. Zhao, D.-M. Hua, Preparation of diesel microemulsion, Shiyou Xuebao, Shiyou Jiagong/Acta Petrolei Sinica (Petroleum Processing Section) 22 (2006) 80–84.

[8] C. Yang, S. Lee, S. Larsen, M. Madsen, ULSD/Biodiesel blend and its effect on fuel/water separation, in: American Filtration & Separation Society Annual Conference, Valley Forge, 2008, pp. 1–12.

[9] D. Wilfong, A. Dallas, C. Yang, P. Johnson, K. Viswanathan, M. Madsen, B. Tucker, J. Hacker, Emerging challenges of filtration, Filtration 10 (2) (2010) 105–115.

[10] C. Stanfel, Fuel filtration: Protecting the diesel engine, Filtration & Separation 46 (3) (2009) 22–25.

[11] C. Yang, S. Larsen, S. Wagner, Understanding Emulsified Water Filtration from Diesel Fuels, in: 8th International Filtration Conference, 2007, pp. 1–5.

[12] I. M. Hutten, Processes for Nonwoven Filter Media, in: Handbook of Nonwoven Filter Media, Elsevier, 2016, pp. 276–342.

[13] R. Bharadwaj, A. Patel, S. Chokdeepanich, D. Ph, G. G. Chase, Oriented Fiber Filter Media, Journal of Engineering Fibers and Fabrics 3 (Special Issue: Filtration) (2008) 29–34.

[14] S. U. Patel, G. G. Chase, Gravity orientation and woven drainage structures in coalescing filters, Separation and Purification Technology 75 (2010) 392–401.

[15] C. Shin, G. Chase, The effect of wettability on drop attachment to glass rods, Journal of Colloid and Interface Science 272 (2004) 186–190.

[16] S. U. Patel, P. S. Kulkarni, S. U. Patel, G. G. Chase, The effect of surface energy of woven drainage channels in coalescing filters, Separation and Purification Technology 87 (2012) 54–61.

[17] C. Shin, G. G. Chase, Water-in-oil coalescence in micro-nanofiber composite filters, AIChE Journal 50 (2004) 343–350.

[18] C. Shin, G. Chase, D. Reneker, Recycled expanded polystyrene nanofibers applied in filter media, Colloids and Surfaces A: Physicochemical and Engineering Aspects 262 (2005) 211–215.

[19] C. Shin, G. G. Chase, D. H. Reneker, The effect of nanofibers on liquid-liquid coalescence filter performance, AIChE Journal 51 (2005) 3109–3113.

[20] C. Shin, G. G. Chase, Separation of WaterinOil Emulsions Using Glass Fiber Media Augmented with Polymer Nanofibers, Journal of Dispersion Science and Technology 27 (2006) 517–522.

[21] G. Viswanadam, G. G. Chase, Waterdiesel secondary dispersion separation using superhydrophobic tubes of nanofibers, Separation and Purification Technology 104 (2013) 81–88.

[22] S. Agarwal, V. von Arnim, T. Stegmaier, H. Planck, A. Agarwal, Effect of Fibrous Coalescer Geometry and Operating Conditions on Emulsion Separation, Industrial & Engineering Chemistry Research 52 (2013) 13164–13170.

[23] P. S. Kulkarni, S. U. Patel, G. G. Chase, Layered hydrophilic/hydrophobic fiber media for water-in-oil coalescence, Separation and Purification Technology 85 (2012) 157–164.

[24] N. Arnault, G. Monsallier, Diesel Fuel Cleanliness at Filter Outlet in Vehicle Real Life Conditions, in: SAE 2013 World Congress {&} Exhibition, no. 2013–01–1696, 2013.

[25] A. O'Hagan, Bayesian analysis of computer code outputs: A tutorial, Reliability Engineering and System Safety 91 (10-11) (2006) 1290–1300.

[26] A. Krasinski, L. Gradon, Comprehensive system for diesel fuel cleaning, Filtration + Separation 53 (1) (2016) 33–36.

[27] S. Andan, S. I. Hariharan, G. G. Chase, Continuum Model Evaluation of the Effect of Saturation on Coalescence Filtration, Separation Science and Technology 43 (8) (2008) 1955–1973.

[28] J. M. Saleh, Transport phenomena in porous media, in: Fluid Flow Handbook, McGraw Hill Professional, 2002.

[29] Y. Mino, Y. Kagawa, T. Ishigami, H. Matsuyama, Numerical simulation of coalescence phenomena of oil-in-water emulsions permeating through straight membrane pore, Colloids and Surfaces A: Physicochemical and Engineering Aspects 491 (2016) 70–77.

[30] Y. Mino, Y. Kagawa, H. Matsuyama, T. Ishigami, Permeation of oil-in-water emulsions through coalescing filter: Two-dimensional simulation based on phase-field model, AIChE Journal 62 (7) (2016) 2525–2532.

[31] Y. Mino, A. Hasegawa, H. Shinto, H. Matsuyama, Lattice-Boltzmann flow simulation of an oil-in-water emulsion through a coalescing filter: Effects of filter structure, Chemical Engineering Science 177 (2018) 210–217.

[32] S. Fotovati, H. V. Tafreshi, B. Pourdeyhimi, Influence of fiber orientation distribution on performance of aerosol filtration media, Chemical Engineering Science 65 (2010) 5285 – 5293.

[33] R. Mead-Hunter, A. J. King, G. Kasper, B. J. Mullins, Computational fluid dynamics (cfd) simulation of liquid aerosol coalescing filters, Journal of Aerosol Science 61 (2013) 36 – 49.

[34] K. Vengadasalam, Computational modelling of coalescence filtration process, Ph.D. thesis, University of Leeds (2016).

[35] M. Abramowitz, I. Stegun, Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables, Applied mathematics series, Dover Publications, 1965.

[36] G. E. P. Box, K. B. Wilson, On the experimental attainment of optimum conditions, Journal of the Royal Statistical Society. Series B 13 (1) (1951) 1–45.

[37] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural networks 2 (5) (1989) 359–366.

[38] M. J. D. Powell, in: J. C. Mason, M. G. Cox (Eds.), Algorithms for Approximation, Clarendon Press, 1987, Ch. Radial Basis Functions for Multivariable Interpolation: A Review, pp. 143–167.

[39] V. Vapnik, Statistical Learning Theory, John Wiley & Sons, 1998.

[40] D. Jones, M. Schonlau, W. Welch, Efficient global optimization of expensive black-box functions, Journal of Global optimization 13 (4) (1998) 455–492.

[41] S. K. Au, J. L. Beck, Estimation of small failure probabilities in high dimensions by subset simulation, Probabilistic Engineering Mechanics 16 (4) (2001) 263–277.

[42] I. Andrianakis, I. R. Vernon, N. McCreesh, T. J. McKinley, J. E. Oakley, R. N. Nsubuga, M. Goldstein, R. G. White, Bayesian History Matching

of Complex Infectious Disease Models Using Emulation: A Tutorial and a Case Study on HIV in Uganda, PLoS Computational Biology 11 (1) (2015) e1003968.

[43] J. Anderson, Fundamentals of Aerodynamics, McGraw-Hill Education, 2010.

[44] J. Anderson, Computational Fluid Dynamics, Computational Fluid Dynamics: The Basics with Applications, McGraw-Hill Education, 1995.

[45] A. T. Patera, A spectral element method for fluid dynamics: Laminar flow in a channel expansion, Journal of Computational Physics 54 (3) (1984) 468–488.

[46] J. E. Pilliod, E. G. Puckett, Second-order accurate volume-of-fluid algorithms for tracking material interfaces, Journal of Computational Physics 199 (2004) 465–502.

[47] M. Sussman, A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell, M. L. Welcome, An Adaptive Level Set Approach for Incompressible Two-Phase Flows, Journal of Computational Physics 148 (1999) 81–124.

[48] M. Sussman, E. G. Puckett, A Coupled Level Set and Volume-of-Fluid Method for Computing 3D and Axisymmetric Incompressible Two-Phase Flows, Journal of Computational Physics 162 (2000) 301–337.

[49] A.-K. Tornberg, B. Engquist, A finite element based level-set method for multiphase flow applications, Computing and Visualization in Science 3 (1-2) (2000) 93–101.

[50] Y. Mino, Y. Kagawa, H. Matsuyama, T. Ishigami, Permeation of oil-in-water emulsions through coalescing filter: Two-dimensional simulation based on phase-field model, AIChE Journal 62 (7) (2016) 2525–2532.

[51] Z.-B. Wang, R. Chen, H. Wang, Q. Liao, X. Zhu, S.-Z. Li, An overview of smoothed particle hydrodynamics for simulating multiphase flow, Applied Mathematical Modelling 40 (23-24) (2016) 9625–9655.

[52] K. Sharp, F. Matschinsky, Translation of Ludwig Boltzmann's Paper On the Relationship between the Second Fundamental Theorem of the Mechanical Theory of Heat and Probability Calculations Regarding the Conditions for Thermal Equilibrium Sitzungberichte der Kaiserlichen Akademie d, Entropy 17 (2015) 1971–2009.

[53] S. Succi, The lattice Boltzmann equation: for fluid dynamics and beyond, Oxford university press, 2001.

[54] P. L. Bhatnagar, E. P. Gross, M. Krook, A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems, Physical Review 94 (3) (1954) 511–525.

[55] T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, E. M. Viggen, The Lattice Boltzmann Method, Graduate Texts in Physics, Springer International Publishing, 2017.

[56] J. Lätt, Hydrodynamic limit of lattice Boltzmann equations, Ph.D. thesis, University of Geneva (2007).

[57] J. von Neumann, The general and logical theory of automata., in: Cerebral mechanisms in behavior; the Hixon Symposium., Wiley, Oxford, England, 1951, pp. 1–41.

[58] D. A. Wolf-Gladrow, Lattice-Gas Cellular Automata and Lattice Boltzmann Models - An Introduction, Springer Berlin Heidelberg, 2005.

[59] J. Hardy, Y. Pomeau, O. de Pazzis, Time evolution of a twodimensional

model system. I. Invariant states and time correlation functions, Journal of Mathematical Physics 14 (12) (1973) 1746–1759.

[60] U. Frisch, B. Hasslacher, Y. Pomeau, Lattice-Gas Automata for the Navier-Stokes Equation, Physical Review Letters 56 (14) (1986) 1505–1508.

[61] R. Nasilowski, A cellular-automaton fluid model with simple rules in arbitrarily many dimensions, Journal of Statistical Physics 65 (1-2) (1991) 97–138.

[62] D. d'Humires, P. Lallemand, U. Frisch, Lattice Gas Models for 3D Hydrodynamics, Europhysics Letters (EPL) 2 (4) (1986) 291–297.

[63] S. Wolfram, Cellular automaton fluids 1: Basic theory, Journal of Statistical Physics 45 (3-4) (1986) 471–526.

[64] G. R. McNamara, G. Zanetti, Use of the boltzmann equation to simulate lattice-gas automata, Physical Review Letters 61 (20) (1988) 2332–2335.

[65] F. J. Higuera, J. Jiménez, Boltzmann Approach to Lattice Gas Simulations, Europhysics Letters (EPL) 9 (7) (1989) 663–668.

[66] F. J. Higuera, S. Succi, R. Benzi, Lattice gas dynamics with enhanced collisions, EPL (Europhysics Letters) 9 (4) (1989) 345–349.

[67] S. Chen, H. Chen, D. Martnez, W. Matthaeus, Lattice Boltzmann model for simulation of magnetohydrodynamics, Physical Review Letters 67 (27) (1991) 3776–3779.

[68] J. M. V. A. Koelman, A Simple Lattice Boltzmann Scheme for Navier-Stokes Fluid Flow, Europhysics Letters (EPL) 15 (6) (1991) 603–607.

[69] Y. H. Qian, D. D'Humières, P. Lallemand, Lattice BGK Models for Navier-Stokes Equation, Europhysics Letters (EPL) 17 (6) (1992) 479–484.

[70] T. Inamuro, T. Ogata, S. Tajima, N. Konishi, A lattice Boltzmann method for incompressible two-phase flows with large density differences, Journal of Computational Physics 198 (2) (2004) 628–644.

[71] H. W. Zheng, C. Shu, Y. T. Chew, A lattice Boltzmann model for multiphase flows with large density ratio, Journal of Computational Physics 218 (1) (2006) 353–371.

[72] H. Liu, Q. Kang, C. R. Leonardi, S. Schmieschek, A. Narváez, B. D. Jones, J. R. Williams, A. J. Valocchi, J. Harting, Multiphase lattice Boltzmann simulations for porous media applications, Computational Geosciences 20 (4) (2016) 777–805.

[73] A. K. Gunstensen, D. H. Rothman, S. Zaleski, G. Zanetti, Lattice boltzmann model of immiscible fluids, Phys. Rev. A 43 (1991) 4320–4327.

[74] D. Grunau, S. Chen, K. Eggert, A lattice Boltzmann model for multiphase fluid flows, Physics of Fluids A: Fluid Dynamics 5 (10) (1993) 2557–2562.

[75] X. Shan, H. Chen, Lattice Boltzmann model for simulating flows with multi phases and components, Physical Review E 47 (3) (1993) 1815–1819.

[76] X. Shan, H. Chen, Simulation of nonideal gases and liquid-gas phase transitions by the lattice Boltzmann equation, Physical Review E 49 (4) (1994) 2941–2948.

[77] M. Swift, E. Orlandini, W. Osborn, J. Yeomans, Lattice Boltzmann simulations of liquid-gas and binary fluid systems, Physical Review E 54 (5) (1996) 5041–5052.

[78] M. C. Sukop, D. T. Thorne, Lattice Boltzmann Modelling - An Introduction for Geoscientists and Engineers, 2nd Edition, Springer, 2006.

[79] L. Chen, Q. Kang, Y. Mu, Y.-L. He, W.-Q. Tao, A critical review of the pseudopotential multiphase lattice Boltzmann model: Methods and applications, International Journal of Heat and Mass Transfer 76 (2014) 210–236.

[80] Z. Guo, C. Zheng, B. Shi, Discrete lattice effects on the forcing term in the lattice Boltzmann method, Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 65 (4) (2002) 1–6.

[81] A. Kupershtokh, D. Medvedev, D. Karpov, On equations of state in a lattice Boltzmann method, Computers & Mathematics with Applications 58 (5) (2009) 965–974.

[82] P. Yuan, L. Schaefer, Equations of state in a lattice Boltzmann model, Physics of Fluids 18 (4) (2006) 042101.

[83] P. Atkins, J. de Paula, Atkins' Physical Chemistry, OUP Oxford, 2010.

[84] J. R. Castrejón-Pita, K. J. Kubiak, A. A. Castrejón-Pita, M. C. T. Wilson, I. M. Hutchings, Mixing and internal dynamics of droplets impacting and coalescing on a solid surface, Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 88 (2) (2013) 1–11.

[85] S. Chibbaro, G. Falcucci, G. Chiatti, H. Chen, X. Shan, S. Succi, Lattice Boltzmann models for nonideal fluids with arrested phase-separation, Physical Review E 77 (3) (2008) 036705.

[86] G. Falcucci, S. Ubertini, S. Succi, Lattice Boltzmann simulations of phase-separating flows at large density ratios: the case of doubly-attractive pseudo-potentials, Soft Matter 6 (18) (2010) 4357.

[87] H. Huang, D. T. Thorne, M. G. Schaap, M. C. Sukop, Proposed approximation for contact angles in Shan-and-Chen-type multicomponent multiphase lattice Boltzmann models, Physical Review E 76 (6) (2007) 066701.

[88] D. Iwahara, H. Shinto, M. Miyahara, K. Higashitani, Liquid Drops on Homogeneous and Chemically Heterogeneous Surfaces: A Two-Dimensional Lattice Boltzmann Study, Langmuir 19 (21) (2003) 9086–9093.

[89] A. R. Davies, J. L. Summers, M. C. T. Wilson, On a dynamic wetting model for the finite-density multiphase lattice Boltzmann method, International Journal of Computational Fluid Dynamics 20 (6) (2006) 415–425.

[90] Q. Zou, X. He, On pressure and velocity flow boundary conditions and bounceback for the lattice Boltzmann BGK model, Physics of Fluids 9 (6) (1996) 1591–1598.

[91] Q. Lou, Z. Guo, B. Shi, Evaluation of outflow boundary conditions for two-phase lattice Boltzmann equation, Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 87 (6) (2013) 1–16.

[92] E. W. Llewellin, LBflow: An extensible lattice Boltzmann framework for the simulation of geophysical flows. Part II: usage and validation, Computers and Geosciences2 36 (2010) 123–132.

[93] J. D. Sterling, S. Chen, Stability Analysis of Lattice Boltzmann Methods, Journal of Computational Physics 123 (1) (1996) 196–206.

[94] R. A. Brownlee, A. N. Gorban, J. Levesley, Stability and stabilization of the lattice Boltzmann method, Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 75 (3) (2007) 1–17.

[95] D. D'Humieres, I. Ginzburg, M. Krafczyk, P. Lallemand, L.-S. Luo, Multiple-relaxation-time lattice Boltzmann, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 360 (1792) (2002) 437–451.

[96] M. E. McCracken, J. Abraham, Multiple-relaxation-time lattice-Boltzmann model for multiphase flow, Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 71 (3) (2005) 036701.

[97] K. N. Premnath, J. Abraham, Three-dimensional multi-relaxation time (MRT) lattice-Boltzmann models for multiphase flow, Journal of Computational Physics 224 (2) (2007) 539–559.

[98] M. L. Porter, E. T. Coon, Q. Kang, J. D. Moulton, J. W. Carey, Multicomponent interparticle-potential lattice Boltzmann model for fluids with large viscosity ratios, Physical Review E - Statistical, Nonlinear, and Soft Matter Physics 86 (3) (2012) 1–8.

[99] A. Keane, P. Nair, Computational Approaches for Aerospace Design, John Wiley & Sons, Derby, 2005.

[100] A. Forrester, A. Sobester, A. Keane, Engineering Design via Surrogate Modelling: A Practical Guide, 1st Edition, John Wiley & Sons, 2008.

[101] A. O'Hagan, Polynomial Chaos: A Tutorial and Critique from a Statisticianís Perspective (2011).

[102] C. E. Rasmussen, C. K. I. Williams, Gaussian processes for machine learning., Vol. 14, MIT Press, 2006.

[103] C. M. Bishop, Pattern recognition and machine learning., Information science and statistics, New York : Springer, 2006., 2006.

[104] J. Sacks, W. J. Welch, T. J. Mitchell, H. P. Wynn, Design and Analysis of Computer Experiments, Statistical Science 4 (1989) 409–423.

[105] N. Cressie, The origins of kriging, Mathematical Geology 22 (3) (1990) 239–252.

[106] J. O. Berger, M. J. Bayarri, The Interplay of Bayesian and Frequentist Analysis, Statistical Science 19 (2004) 58–80.

[107] J. Oakley, Bayesian uncertainty analysis for complex computer codes, Ph.D. thesis, University of Sheffield (1999).

[108] W. Becker, Uncertainty Propagation through Large Nonlinear Models, Ph.D. thesis, University of Sheffield (2011).

[109] T. Santner, B. Williams, W. Notz, The Design and Analysis of Computer Experiments, Springer Series in Statistics, Springer New York, 2010.

[110] M. D. McKay, R. J. Beckman, W. J. Conover, Comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (2) (1979) 239–245.

[111] L. Pronzato, W. G. Müller, Design of computer experiments: space filling and beyond, Statistics and Computing 22 (3) (2012) 681–701.

[112] J. L. Loeppky, J. Sacks, W. Welch, W. J. Welch, Choosing the Sample Size of, Technometrics 51 (4) (2008) 366–376.

[113] R. G. E. Haylock, Bayesian inference about outputs of computationally expensive algorithms with uncertainty on the inputs, Ph.D. thesis, University of Nottingham (1997).

[114] J. A. Nelder, R. Mead, A simplex method for function minimization, The Computer Journal 7 (4) (1965) 308–313.

[115] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, Science 220 (4598) (1983) 671–680.

[116] D. Whitley, A genetic algorithm tutorial, Statistics and Computing 4 (2).

[117] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, E. Teller, Equation of State Calculations by Fast Computing Machines, The Journal of Chemical Physics 21 (6) (1953) 1087–1092.

[118] P. Ranjan, R. Haynes, R. Karsten, A Computationally Stable Approach to Gaussian Process Interpolation of Deterministic Computer Simulation Data, Technometrics 53 (4) (2011) 366–378.

[119] H. Song, K. K. Choi, D. Lamb, A Study on Improving the Accuracy of Kriging Models by Using Correlation Model/Mean Structure Selection and Penalized Log-Likelihood Function, in: 10th World Conference on Structural and Multidisciplinary Optimization, Orlando, 2013, pp. 1–10.

[120] L. S. Bastos, A. O'Hagan, Diagnostics for Gaussian Process Emulators, Technometrics 51 (4) (2009) 425–438.

[121] D. Raymer, Aircraft Design: A Conceptual Approach, Fifth Edition, American Institute of Aeronautics and Astronautics, Inc., Washington, DC, 2012.

[122] S. Lloyd, Ultimate physical limits to computation, Nature 406 (2000) 1047–1054.

[123] B. W. Kernighan, D. M. Ritchie, The C Programming Language, 2nd Edition, Prentice Hall Professional Technical Reference, 1988.

[124] N. Delbosc, Real-Time Simulation of Indoor Air Flow Using the Lattice Boltzmann Method on Graphics Processing Unit, Ph.D. thesis, University of Leeds (2015).

[125] U. Ayachit, The ParaView Guide: A Parallel Visualization Application, Kitware, 2015.

[126] Z. Zeng, R. Grigg, A Criterion for Non-Darcy Flow in Porous Media, Transport in Porous Media 63 (1) (2006) 57–69.

[127] R. Holdich, Fundamentals of Particle Technology, Midland Information Technology and Publishing, 2002.

[128] M. M. Awad, Two-phase flow, in: S. N. Kazi (Ed.), An Overview of Heat Transfer Phenomena, InTech, 2012, Ch. 11.

[129] W. Gambill, How to estimate mixtures viscosities, Chemical Engineering 66 (5) (1959) 151–152.

[130] D. S. Viswanath, T. Ghosh, D. H. Prasad, N. V. Dutt, K. Y. Rani, Viscosity of Liquids, 2007.

[131] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, A. Wu, An efficient k-means clustering algorithm: analysis and implementation, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (7) (2002) 881–892.

[132] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, in: Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, 1996, pp. 226–231.

[133] P. O. Hristov, F. A. DiazDelaO, K. J. Kubiak, U. Farooq, Adaptive Gaussian Process Emulators for Efficient Reliability Analysis, Under Review.

[134] M. Bampi, A. D. P. Scheer, F. de Castilhos, Application of near infrared spectroscopy to predict the average droplet size and water content in biodiesel emulsions, Fuel 113 (2013) 546–552.

[135] Horiba Scientific, A Guidebook To Particle Size Analysis, Tech. rep., Horiba Scientific (2014).

[136] M. C. Kennedy, A. O'Hagan, Bayesian calibration of computer models, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63 (2001) 425–464.

[137] S. Conti, A. O'Hagan, Bayesian emulation of complex multi-output and dynamic computer models, Journal of Statistical Planning and Inference 140 (3) (2010) 640–651.

[138] R. Paulo, G. García-Donato, J. Palomo, Calibration of computer models with multivariate output, Computational Statistics & Data Analysis 56 (12) (2012) 3959–3974.

[139] J. Rougier, Efficient Emulators for Multivariate Deterministic Functions, Journal of Computational and Graphical Statistics 17 (4) (2008) 827–843.

[140] K. Wang, T. Chen, S. T. Kwa, Y. Ma, R. Lau, Meta-modelling for fast analysis of CFD-simulated vapour cloud dispersion processes, Computers & Chemical Engineering 69 (2014) 89–97.

[141] R. Hyndman, G. Athanasopoulos, Forecasting: principles and practice, OTexts, 2018.

[142] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, S. Tarantola, Global sensitivity analysis. The primer, John Wiley & Sons, 2008.

[143] I. M. Sobol, Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates, Mathematics and Computers in Simulation 55 (1-3) (2001) 271–280.

[144] T. Homma, A. Saltelli, Importance measures in global sensitivity analysis of nonlinear models, Reliability Engineering & System Safety 52 (1) (1996) 1–17.

[145] A. Saltelli, S. Tarantola, F. Campolongo, M. Ratto, Sensitivity analysis in practice: A guide to assessing scientific models, Wiley, 2004.

[146] A. Saltelli, Making best use of model evaluations to compute sensitivity indices, Computer Physics Communications 145 (2) (2002) 280–297.

[147] J. E. Oakley, A. O'Hagan, Probabilistic sensitivity analysis of complex models: A Bayesian approach, Journal of the Royal Statistical Society. Series B: Statistical Methodology 66 (3) (2004) 751–769.

[148] N. Durrande, D. Ginsbourger, O. Roustant, L. Carraro, ANOVA kernels and RKHS of zero mean functions for model-based sensitivity analysis, Journal of Multivariate Analysis 115 (2013) 57–67.

[149] E. I. Saavedra Flores, R. M. Ajaj, I. Dayyani, Y. Chandra, R. Das, Multi-scale model updating for the mechanical properties of cross-laminated timber, Computers and Structures 177 (2016) 83 – 90.

[150] ANSYS, Parametric Design Language Guide. Release 16.1, ANSYS, Inc., Canonsburg, PA 15317, http://www.ansys.com, 2016.

[151] E. I. Saavedra Flores, I. Dayyani, R. Ajaj, R. Castro-Triguero, F. DiazDelaO, R. Das, P. González Soto, Analysis of cross-laminated timber by computational homogenisation and experimental validation, Composite Structures 121 (2015) 386 – 394.

[152] E. I. Saavedra Flores, F. A. DiazDelaO, M. I. Friswell, R. M. Ajaj, Investigation on the extensibility of the wood cell-wall composite by an approach based on homogenisation and uncertainty analysis, Composite Structures 108 (2014) 212–222.

[153] E. I. Saavedra Flores, F. A. DiazDelaO, R. M. Ajaj, M. I. Friswell, G. F. Fernando, Mathematical modelling of the stochastic mechanical properties

of wood and its extensibility at small scales, Applied Mathematical Modelling 38 (15–16) (2014) 3958–3967.

[154] J. Penner, D. Lister, D. Griggs, D. Dokken, Aviation and the Global Atmosphere: A Special Report of the Intergovernmental Panel on Climate Change, Cambridge University Press, 1999.

[155] M. Guenov, M. Nunez, A. Molina-Cristóbal, V. C. Datta, A. Riaz, Aircadia an Interactive Tool for the Composition and Exploration of Aircraft Computational Studies At Early Design Stage, 29th Congress of the International Council of the Aeronautical Sciences (2014) 1–12.

[156] T. W. Simpson, Comparison of Response Surface and Kriging Models in the Multidisciplinary Design of an Aerospike Nozzle, Tech. Rep. February, NASA Langley (1998).

[157] C. Turner, Generic sequential sampling for metamodel approximations, in: ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Chicago, 2003, pp. 1–10.

[158] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, P. Kevin Tucker, Surrogate-based analysis and optimization, Progress in Aerospace Sciences 41 (1) (2005) 1–28.

[159] G. G. Wang, S. Shan, Review of Metamodeling Techniques in Support of Engineering Design Optimization, Journal of Mechanical Design 129 (4) (2007) 370.

[160] C. Audet, J. E. Dennis, D. W. Moore, A. Booker, P. D. Frank, A Surrogate-Model-Based Method for Constrained Optimization, in: 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, 2000, pp. 1–11.

[161] H.-S. Li, S.-K. Au, Design optimization using Subset Simulation algorithm, Structural Safety 32 (6) (2010) 384–392.

[162] P. Ranjan, D. Bingham, G. Michailidis, Sequential Experiment Design for Contour Estimation From Complex Computer Codes, Technometrics 50 (4) (2008) 527–541.

[163] V. Picheny, D. Ginsbourger, O. Roustant, R. T. Haftka, N.-H. Kim, Adaptive Designs of Experiments for Accurate Approximation of a Target Region, Journal of Mechanical Design 132 (7) (2010) 071008.

[164] Z. Zhu, X. Du, Reliability Analysis With Monte Carlo Simulation and Dependent Kriging Predictions, Journal of Mechanical Design 138 (12) (2016) 121403.

[165] I. G. Osio, C. H. Amon, An engineering design methodology with multistage Bayesian surrogates and optimal sampling, Research in Engineering Design 8 (4) (1996) 189–206.

[166] I. K. Bang, D. S. Han, G. J. Han, K. H. Lee, Structural optimization for a jaw using iterative Kriging metamodels, Journal of Mechanical Science and Technology 22 (9) (2008) 1651–1659.

[167] R. C. Kuczera, Z. P. Mourelatos, On Estimating the Reliability of Multiple Failure Region Problems Using Approximate Metamodels, Journal of Mechanical Design 131 (12) (2009) 121003.

[168] V. Dubourg, Adaptive surrogate models for reliability analysis and reliability-based design optimization, Ph.D. thesis, Blaise Pascal University (2011).

[169] B. Jones, R. T. Johnson, Design and Analysis for the Gaussian Process

Model, Quality and Reliability Engineering International 23 (2007) 517–543.

[170] T. H. Lee, J. J. Jung, A sampling technique enhancing accuracy and efficiency of metamodel-based RBDO: Constraint boundary sampling, Computers and Structures 86 (13-14) (2008) 1463–1476.

[171] V. Dubourg, B. Sudret, J. M. Bourinet, Reliability-based design optimization using kriging surrogates and subset simulation, Structural and Multidisciplinary Optimization 44 (5) (2011) 673–690.

[172] R. Jin, W. Chen, A. Sudjianto, On sequential sampling for global metamodelling in engineering design, in: DETC ASME 2002 Design Engineering Technical Conferences And Computers and Information in Engineering Conference, Montreal, 2002, pp. 1–10.

[173] G. Su, B. Yu, Y. Xiao, L. Yan, Gaussian Process Machine-Learning Method for Structural Reliability Analysis, Advances in Structural Engineering 17 (9) (2014) 1257–1271.

[174] L. Zhang, Z. Lu, P. Wang, Efficient structural reliability analysis method based on advanced Kriging model, Applied Mathematical Modelling 39 (2) (2015) 781–793.

[175] J. Bect, D. Ginsbourger, L. Li, V. Picheny, E. Vazquez, Sequential design of computer experiments for the estimation of a probability of failure, Statistics and Computing 22 (3) (2012) 773–793.

[176] B. Bichon, S. Mahadevan, M. Eldred, Reliability-Based Design Optimization Using Efficient Global Reliability Analysis, in: 50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Palm Springs, 2009.

[177] B. Echard, N. Gayton, M. Lemaire, AK-MCS: An active learning reliability method combining Kriging and Monte Carlo Simulation, Structural Safety 33 (2) (2011) 145–154.

[178] Z. Wen, H. Pei, H. Liu, Z. Yue, A Sequential Kriging reliability analysis method with characteristics of adaptive sampling regions and parallelizability, Reliability Engineering & System Safety 153 (2016) 170–179.

[179] X. Huang, J. Chen, H. Zhu, Assessing small failure probabilities by AK-SS: An active learning method combining Kriging and Subset Simulation, Structural Safety 59 (2016) 86–95.

[180] C. Tong, Z. Sun, Q. Zhao, Q. Wang, S. Wang, A hybrid algorithm for reliability analysis combining Kriging and subset simulation importance sampling, Journal of Mechanical Science and Technology 29 (8) (2015) 3183–3193.

[181] Z. Sun, J. Wang, R. Li, C. Tong, LIF: A new Kriging based learning function and its application to structural reliability analysis, Reliability Engineering & System Safety 157 (2017) 152–165.

[182] X. Yang, Y. Liu, L. Gao, Unified reliability analysis by active learning Kriging model combining with random-set based Monte Carlo simulation method, Int. J. Numer. Meth. Engng 108 (2016) 1343–1361.

[183] M. J. Sasena, Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations, Ph.D. thesis, University of Michigan (2002).

[184] C. Bucher, U. Bourgund, A fast and efficient response surface approach for structural reliability problems, Structural Safety 7 (1) (1990) 57–66.

[185] N. Gayton, J. Bourinet, M. Lemaire, CQ2RS: a new statistical approach to the response surface method for reliability analysis, Structural Safety 25 (1) (2003) 99–121.

[186] S. Gupta, C. S. Manohar, An improved response surface method for the determination of failure probability and importance measures, Structural Safety 26 (2) (2004) 123–139.

[187] D. Q. Li, K. B. Shao, Z. J. Cao, X. S. Tang, K. K. Phoon, A generalized surrogate response aided-subset simulation approach for efficient geotechnical reliability-based design, Computers and Geotechnics 74 (2016) 88–101.

[188] H. M. Gomes, A. M. Awruch, Comparison of response surface and neural network with other methods for structural reliability analysis, Structural Safety 26 (1) (2004) 49–67.

[189] V. Papadopoulos, D. G. Giovanis, N. D. Lagaros, M. Papadrakakis, Accelerated subset simulation with neural networks for reliability analysis, Computer Methods in Applied Mechanics and Engineering 223-224 (2012) 70–80.

[190] N.-C. Xiao, M. J. Zuo, C. Zhou, A new adaptive sequential sampling method to construct surrogate models for efficient reliability analysis, Reliability Engineering & System Safety 169 (2018) 330–338.

[191] J. E. Hurtado, An examination of methods for approximating implicit limit state functions from the viewpoint of statistical learning theory, Structural Safety 26 (3) (2004) 271–293.

[192] J. M. Bourinet, F. Deheeger, M. Lemaire, Assessing small failure probabilities by combined subset simulation and Support Vector Machines, Structural Safety 33 (6) (2011) 343–353.

[193] B. Sudret, Meta-models for structural reliability and uncertainty quantification, in: Proc. 5th Asian-Pacific Symp. Struct. Reliab. (APSSRA 2012), Singapore, 2012, pp. 53–76.

[194] S. K. Au, Y. Wang, Engineering Risk Assessment with Subset Simulation, Wiley, 2014.

[195] D. E. Huntington, C. S. Lyrintzist, Improvements to and limitations of Latin hypercube sampling, Prob. Engng. Mech 13 (4) (1998) 245–253.

[196] B. J. Bichon, M. S. Eldred, L. P. Swiler, S. Mahadevan, J. M. McFarland, Efficient Global Reliability Analysis for Nonlinear Implicit Performance Functions, AIAA Journal 46 (10) (2008) 2459–2468.

[197] R. Tibshirani, G. Walther, T. Hastie, Estimating the number of clusters in a data set via the gap statistic, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63 (2002) 411–423.

[198] V. Dubourg, B. Sudret, F. Deheeger, Metamodel-based importance sampling for structural reliability analysis, Probabilistic Engineering Mechanics 33 (2013) 47–57.

[199] S. E. Gano, J. E. Renaud, J. D. Martin, T. W. Simpson, Update strategies for kriging models used in variable fidelity optimization, Structural and Multidisciplinary Optimization 32 (4) (2006) 287–298.

[200] L. C. W. Dixon, G. P. Szego, The global optimization problem: an introduction, Towards global optimisation 2 (1978) 1–15.

[201] P. O. Hristov, F. A. Diazdelao, K. J. Kubiak, U. Farooq, Reliability Analysis and Surrogate Modelling of Biodiesel Filters, in: Proceedings of the 26$^{\text{th}}$ European Safety and Reliability Conference (ESREL), Glasgow, 2016, pp. 390–395.

[202] G. T. Herman, Fundamentals of Computerized Tomography, Advances in Pattern Recognition, Springer London, London, 2009.

[203] U. Aaltosalmi, Fluid flow in porous media with the lattice Boltzmann method, Ph.D. thesis, University of Jyväskylä (2005).

[204] Y. Gao, Simulated fluid transport in gas diffusion layers of PEM Fuel Cells using lattice Boltzmann method and X-ray computed tomography, Ph.D. thesis, University of Liverpool (2012).

[205] J. Banhart, Advanced Tomographic Methods in Materials Research and Engineering, Monographs on the Physics and Chemistry of Materials, OUP Oxford, 2008.

[206] K. C. A. Smith, C. W. Oatley, The scanning electron microscope and its fields of application, British Journal of Applied Physics 6 (1955) 391–399.

[207] P. P. Mukherjee, Q. Kang, C.-Y. Wang, Pore-scale modeling of two-phase transport in polymer electrolyte fuel cellsprogress and perspective, Energy Environ. Sci. 4 (2011) 346–369.

[208] J. Goldstein, D. Newbury, P. Echlin, D. Joy, A. Romig, C. Lyman, C. Fiori, E. Lifshin, Scanning Electron Microscopy and X-Ray Microanalysis: A Text for Biologists, Materials Scientists, and Geologists, Springer US, 2012.

[209] E. Öznergiz, Y. E. Kiyak, M. E. Kamasak, I. Yildirim, Automated Nanofiber Diameter Measurement in SEM Images Using a Robust Image Analysis Method, Journal of Nanomaterials (2014) Article ID 738490.

[210] C.-L. Pai, M. C. Boyce, G. C. Rutledge, On the importance of fiber curvature to the elastic moduli of electrospun nonwoven fiber meshes, Polymer 52 (26) (2011) 6126–6133.

[211] G. G. Chase, V. Beniwal, C. Venkataraman, Measurement of uni-axial fiber angle in non-woven fibrous media, Chemical Engineering Science 55 (2000) 2151–2160.

[212] I. M. Hutten, Testing of Nonwoven Filter Media, in: Handbook of Nonwoven Filter Media, Elsevier, 2016, pp. 343–408.

[213] A. Jena, K. Gupta, Advanced Technology for Evaluation of Pore Structure Characteristics of Filtration Media to Optimize Their Design and Performance (2002).

[214] L. D. Gelb, K. E. Gubbins, Pore Size Distributions in Porous Glasses: A Computer Simulation Study, Langmuir 15 (2) (1999) 305–308.

[215] J. Becker, A. Wiegmann, V. Schulz, Design of fibrous filter media based on the simulation of pore size measures, in: Filtech, International Conference & Exibition for Filtration and Separation Technology, 2007, pp. 71–78.

[216] M. M. Tomadakis, T. J. Robertson, Pore size distribution, survival probability, and relaxation time in random and ordered arrays of fibers, The Journal of Chemical Physics 119 (3) (2003) 1741–1749.

[217] I. M. Hutten, Filtration Mechanisms and Theory, in: Handbook of Nonwoven Filter Media, Elsevier, 2016, pp. 53–107.

[218] H. Piekaar, L. Clarenburg, Aerosol filtersPore size distribution in fibrous filters, Chemical Engineering Science 22 (1967) 1399–1407.

[219] P. Danwanichakul, D. Danwanichakul, Two-Dimensional Simulation of Electrospun Nanofibrous Structures: Connection of Experimental and Simulated Results, Journal of Chemistry 2014 (2014) ID479139.

[220] B. Maze, H. Vahedi Tafreshi, Q. Wang, B. Pourdeyhimi, A simulation of unsteady-state filtration via nanofiber media at reduced operating pressures, Journal of Aerosol Science 38 (2007) 550–571.

[221] S. Hosseini, H. V. Tafreshi, 3-D simulation of particle filtration in electrospun nanofibrous filters, Powder Technology 201 (2010) 153–160.

[222] K. Schladitz, S. Peters, D. Reinel-Bitzer, A. Wiegmann, J. Ohser, Design of acoustic trim based on geometric modeling and flow simulation for non-woven, Computational Materials Science 38 (1) (2006) 56–66.

[223] W. Sambaer, M. Zatloukal, D. Kimmer, 3D modeling of filtration process via polyurethane nanofiber based nonwoven filters prepared by electrospinning process, Chemical Engineering Science 66 (2011) 613–623.

[224] G. Farin, J. Hoschek, M. Kim, Bézier Techniques, in: Handbook of Computer Aided Geometric Design, Elsevier, 2002, pp. 75–111.

[225] J. Bernsdorf, G. Brenner, F. Durst, Numerical analysis of the pressure drop in porous media flow with lattice Boltzmann (BGK) automata, Computer Physics Communications 129 (2000) 247–255.

[226] M. Junk, Z. Yang, Pressure boundary condition for the lattice Boltzmann method, Computers & Mathematics with Applications 58 (2009) 922–929.

[227] P. Hristov, F. DiazDelaO, E. Saavedra Flores, C. Guzmán, U. Farooq, Probabilistic sensitivity analysis to understand the influence of micromechanical properties of wood on its macroscopic response, Composite Structures 181 (2017) 229–239.

[228] P. Hristov, F. DiazDelaO, K. Kubiak, U. Farooq, Adaptive emulation-based reliability analysis, in: Proceedings of the 2$^{\text{nd}}$ International Conference

on Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP), Rhodes, 2017, pp. 198–211.