A Parallel Solution Adaptive Implementation of the Direct Simulation Monte Carlo Method

Stuart Jackson Wishart

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Department of Aerospace, Mechatronic and Mechanical Engineering
The University of Sydney
August 2004
Declaration

This thesis is submitted to The University of Sydney in fulfilment of the requirements for the degree of Doctor of Philosophy. This thesis is entirely my own work and, except where otherwise stated, describes my own research.

Stuart Wishart

August 30, 2004
Abstract

Stuart Wishart  
The University of Sydney  
August 2004

A Parallel Solution Adaptive Implementation of the Direct Simulation Monte Carlo Method

This thesis deals with the direct simulation Monte Carlo (DSMC) method of analysing gas flows. The DSMC method was initially proposed as a method for predicting rarefied flows where the Navier-Stokes equations are inaccurate. It has now been extended to near continuum flows. The method models gas flows using simulation molecules which represent a large number of real molecules in a probabilistic simulation to solve the Boltzmann equation. Molecules are moved through a simulation of physical space in a realistic manner that is directly coupled to physical time such that unsteady flow characteristics are modelled. Intermolecular collisions and molecule-surface collisions are calculated using probabilistic, phenomenological models. The fundamental assumption of the DSMC method is that the molecular movement and collision phases can be decoupled over time periods that are smaller than the mean collision time.

Two obstacles to the wide spread use of the DSMC method as an engineering tool are in the areas of simulation configuration, which is the configuration of the simulation parameters to provide a valid solution, and the time required to obtain a solution. For complex problems, the simulation will need to be run multiple times, with the simulation configuration being modified between runs to provide an accurate solution for the previous run’s results, until the solution converges. This task is time consuming and requires the user to have a good understanding of the DSMC method. Furthermore, the computational resources required by a DSMC simulation increase rapidly as the simulation approaches the continuum regime. Similarly, the computational requirements of three-dimensional problems are generally two orders of magnitude more than two-dimensional problems. These large computational requirements significantly limit the range of problems that can be practically solved on an engineering workstation or desktop computer.

The first major contribution of this thesis is in the development of a DSMC implementation that automatically adapts the simulation. Rather than modifying the simulation configuration between solution runs, this thesis presents the formulation of algorithms that allow the simulation configuration to be automatically adapted during a single run. These adaption algorithms adjust the three main parameters that effect the accuracy of a DSMC simulation, namely the solution grid, the time step and the simulation molecule number density. The second major contribution extends the parallelisation of the DSMC method. The implementation developed in this thesis combines the capability to use a cluster of computers to increase the maximum size of problem that can be solved while simultaneously allowing excess computational resources to decrease the total solution time. Results are presented to verify the accuracy of the underlying DSMC implementation, the utility of the solution adaption algorithms and the efficiency of the parallelisation implementation.
Acknowledgments

I would like to thank all of the people who encouraged and supported me during the undertaking of this research.

Firstly, I would like to thank all the members of the Aeronautical Engineering Department. To my supervisor, Dr. Doug Auld, who set me on this path of research. To my post-graduate colleagues Jeremy Randle, Dr. Hugh Stone and Dr. Nick van Bronswijk, who helped make the experience of this thesis more rewarding. Thanks also to the members of the ACFR who encouraged my research and provided me with good coffee.

I would also like to acknowledge the support of my employer BAE Systems Australia. While they have tried to tempt me away from my research with interesting projects, they have also been supportive of my completion.

My last and greatest thanks is for my family. To my parents and siblings for their encouragement and support throughout the entirety of this project. To my in-laws who kept my strength up with good food and the inquisitive questions. Lastly, I would like to thank my wife Clare who has been steadfast in her support. But most importantly for her love and companionship.

“The end of a matter is better than its beginning, and patience is better than pride.”

Ecclesiastes 7:8
## Contents

**Declaration** ............................... i

**Abstract** ..................................... ii

**Acknowledgments** ....................... iii

**Contents** .................................... iv

**List of Figures** ............................ viii

**List of Tables** ............................. xi

**Nomenclature** ............................. xii

1 **Introduction** ............................. 1
   1.1 Background and Motivation ......................... 1
   1.2 Problem Summary .................................... 2
   1.3 Principle Contributions ............................ 2
   1.4 Outline .............................................. 4

2 **The Direct Simulation Monte Carlo Method** ............................ 5
   2.1 Introduction ........................................ 5
   2.2 DSMC Theory ....................................... 5
   2.3 Method Outline ..................................... 6
   2.4 Implementation Issues ............................ 7
      2.4.1 Grid Representation ............................ 8
      2.4.2 Time Step Selection ............................ 9
      2.4.3 Number of Molecules per Cell .................. 10
      2.4.4 Ratio of Real to Simulation Molecules .......... 11
   2.5 Griding Methods .................................... 11
      2.5.1 Discussion ....................................... 17

3 **Movement Algorithms** .................. 18
   3.1 Introduction ....................................... 18
   3.2 Alternate Methods ............................... 19
      3.2.1 Physical Domain Methods ....................... 19
      3.2.2 Computational Movement Methods ............... 23
3.3 Computational Movement Algorithms ............................................................. 23
  3.3.1 Constant Time Step Subdivision Algorithm ........................................... 24
  3.3.2 Improved Time Step Subdivision Algorithm .......................................... 26
  3.3.3 Results ..................................................................................................... 27
3.4 Hybrid Algorithm ............................................................................................. 29
  3.4.1 Derivation................................................................................................ 30
  3.4.2 Robustness ............................................................................................... 32
  3.4.3 Simplifications ........................................................................................ 33
  3.4.4 DSMC Movement Implementation ......................................................... 33
  3.4.5 Results ..................................................................................................... 37
  3.4.6 Extension to 3D Geometries ................................................................. 37

4 Program Structure 44
4.1 Introduction ....................................................................................................... 44
4.2 Physical Models ................................................................................................ 44
  4.2.1 Collision Model ....................................................................................... 44
  4.2.2 Molecule-Surface Interactions ................................................................. 46
  4.2.3 Boundary Models .................................................................................... 46
  4.2.4 Flowfield Initialisation ............................................................................ 48
4.3 Grid Generation ................................................................................................ 48
  4.3.1 Geometry Definition ............................................................................... 48
  4.3.2 Grid Generation ....................................................................................... 50
4.4 Implementation ................................................................................................. 54
  4.4.1 Simulation Configuration Parameters ..................................................... 54
  4.4.2 Geometry Definition ............................................................................... 54
  4.4.3 Optimisations .......................................................................................... 54
  4.4.4 Programming and Storage Issues ............................................................ 56
  4.4.5 Solution Management ............................................................................. 57
  4.4.6 Flow Chart ............................................................................................... 58

5 Solution Adaption 60
5.1 Introduction ....................................................................................................... 60
5.2 Grid Adaption .................................................................................................. 60
  5.2.1 Adaption Methodology ........................................................................... 61
  5.2.2 Grid Resizing and Splitting ..................................................................... 64
  5.2.3 Parameter Tuning ................................................................................... 68
5.2.4 Implementation ....................................................................................... 74
5.3 Ratio of Real to Simulation Molecule Adaption .............................................. 74
  5.3.1 Adjusting Grid blocks with an Excessive Number of Molecules per Cell ................................................................. 77
5.4 Time Step Adaption ..................................................................................... 79
  5.4.1 Base Algorithm ..................................................................................... 79
  5.4.2 Implementation ..................................................................................... 81
5.5 Solution Adaption Implementation ................................................................. 82
  5.5.1 Molecule Injection ............................................................................. 82
  5.5.2 Steady Flow Problems ................................................................... 83
  5.5.3 Unsteady Flow Problems ................................................................... 85

6 Parallelisation .......................................................................................... 87
  6.1 Introduction........................................................................................ 87
  6.2 Physical Domain Decomposition Parallelisation......................................... 88
    6.2.1 Domain Decomposition Algorithm ............................................. 90
    6.2.2 Dynamic Domain Decomposition .......................................... 91
    6.2.3 Total Node Computational Load ................................................. 94
    6.2.4 Node Grid Structure ................................................................. 95
  6.3 Parallel Statistically Independent Runs ..................................................... 96
  6.4 Implementation .................................................................................. 98
    6.4.1 Sub-domains ................................................................................ 99
    6.4.2 Domain Decomposition ........................................................... 100
    6.4.3 Solution Adaption ................................................................. 103
    6.4.4 Parallel Libraries ................................................................. 105

7 Program Verification and Results .................................................................. 106
  7.1 Introduction ........................................................................................ 106
  7.2 Program Verification ........................................................................... 106
    7.2.1 Collision Rate Test .................................................................... 106
    7.2.2 Supersonic Leading-edge Test ............................................... 108
  7.3 Serial Results .................................................................................. 112
    7.3.1 Test Configuration .................................................................... 112
    7.3.2 Subsonic Test Results .............................................................. 113
    7.3.3 Supersonic Test Results ............................................................ 117
    7.3.4 Conclusions ........................................................................... 122
List of Figures

Figure 2.1 Serial DSMC flow chart .............................................................. 8
Figure 2.2 Example of a regular rectangular grid ......................................... 13
Figure 2.3 Example of an algebraic grid ...................................................... 13
Figure 2.4 Example of a variable resolution rectangular grid ....................... 14
Figure 2.5 Example of a body-fitted grid .................................................... 15
Figure 2.6 Example of an unstructured grid ................................................ 16
Figure 2.7 Example of a multi-block body-fitted grid .................................. 16
Figure 3.1 Types of molecule-surface interactions ....................................... 18
Figure 3.2 Geometric cell location vector definitions .................................... 21
Figure 3.3 General transformation between physical body-fitted coordinates and computational rectangular coordinates .............................. 24
Figure 3.4 Definition of cell boundary surface area vectors ............................ 25
Figure 3.5 Movement algorithm test grids .................................................. 27
Figure 3.6 Mach No. contours of flow past a horizontal flat plate ................. 28
Figure 3.7 Molecule movement error due to changes in cell aspect ratio and ‘skew’ ................................................................. 28
Figure 3.8 Geometry definition for an arbitrary quadrilateral cell ................. 30
Figure 3.9 Geometry definition for an arbitrary hexahedral cell ................. 38
Figure 3.10 Geometry definition for an arbitrary tetrahedral cell .......... 40
Figure 3.11 Subdivision of a hexahedral cell into six tetrahedral ............... 42
Figure 4.1 Variable sub-cell search method ................................................ 45
Figure 4.2 TFI grid with folding ................................................................. 51
Figure 4.3 Detail of TFI grid with folding .................................................. 51
Figure 4.4 Grid smoothing results of folded TFI grid ................................. 53
Figure 4.5 Parallel implementation optimised time step processing ............ 56
Figure 4.6 Full simulation flow chart ........................................................ 59
Figure 5.1 Grid split line calculation ............................................................ 67
Figure 5.2 Split contour calculation ............................................................ 68
Figure 5.3 Grid adaption test case 1 flow field ........................................... 69
Figure 5.4 Grid adaption test case 1 adapted grids ....................................... 71
Figure 5.5 Grid adaption test case 1, configuration B2 adapted flow field .... 71
Figure 5.6 Grid adaption test case 2 flow field ........................................... 72
Figure 5.7 Grid adaption test case 2 adapted grids ....................................... 73
Figure 5.8 Grid adaption test case 2, configuration C1 adapted flow field.................. 74
Figure 5.9 Grid adaption flow chart ........................................................................ 76
Figure 5.10 Steady flow adaption solution procedure flow chart ................................. 86
Figure 6.1 Physical domain decomposition flow chart ............................................. 89
Figure 6.2 Measured parallel efficiency of the PDD method ([19], Figure 17).......... 95
Figure 6.3 Extract of sub-domain grid movement flow chart ..................................... 96
Figure 6.4 Parallel statically independent runs flow chart ......................................... 97
Figure 6.5 Hybrid parallel flow chart .................................................................... 99
Figure 6.6 Domain decomposition test grids............................................................ 102
Figure 6.7 Domain decomposition test results ......................................................... 103
Figure 7.1 Supersonic leading-edge number density contours overlaying the results from [7] Figure 14.7, $Ma_\infty = 4.0$, $T_\infty = 300$K, $Kn = 0.0143$, $n = 10^{-20}$m$^3$............. 110
Figure 7.2 Supersonic leading-edge number density contours, $Ma_\infty = 4.0$, $T_\infty = 300$K, $Kn = 0.0143$, $n = 10^{-20}$m$^3$ .................................................. 110
Figure 7.3 Supersonic leading-edge temperature number contours, $Ma_\infty = 4.0$, $T_\infty = 300$K, $Kn = 0.0143$, $n = 10^{-20}$m$^3$ .................................................. 110
Figure 7.4 Supersonic leading-edge local Mach contours, $Ma_\infty = 4.0$, $T_\infty = 300$K, $Kn = 0.0143$, $n = 10^{-20}$m$^3$ .................................................. 111
Figure 7.5 Supersonic leading-edge pressure coefficient, $Ma_\infty = 4.0$, $T_\infty = 300$K, $Kn = 0.0143$, $n = 10^{-20}$m$^3$ .................................................. 111
Figure 7.6 Supersonic leading-edge skin friction coefficient, $Ma_\infty = 4.0$, $T_\infty = 300$K, $Kn = 0.0143$, $n = 10^{-20}$m$^3$ .................................................. 111
Figure 7.7 Vertical flat plate static, unadapted grid .................................................. 113
Figure 7.8 Local Mach number contours past a vertical flat plate, $Ma_\infty = 0.53$, $T_\infty = 300$K, $Kn = 0.043$, $n = 10^{-20}$m$^3$ .................................................. 113
Figure 7.9 Number density contours past a vertical flat plate, $Ma_\infty = 0.53$, $T_\infty = 300$K, $Kn = 0.043$, $n = 10^{-20}$m$^3$ .................................................. 114
Figure 7.10 Streamlines past a vertical flat plate, $Ma_\infty = 0.53$, $T_\infty = 300$K, $Kn = 0.043$, $n = 10^{-20}$m$^3$ .................................................. 114
Figure 7.11 Adapted grid, $Ma_\infty = 0.53$, $T_\infty = 300$K, $Kn = 0.043$, $n = 10^{-20}$m$^3$ .................................................. 114
Figure 7.12 Adapted time step multiple, $Ma_\infty = 0.53$, $T_\infty = 300$K, $Kn = 0.043$, $n = 10^{-20}$m$^3$ .................................................. 115
Figure 7.13 Ratio of average cell side length to local mean free path, $Ma_\infty = 0.53$, $T_\infty = 300$K, $Kn = 0.043$, $n = 10^{-20}$m$^3$ .................................................. 115
Figure 7.14 Ratio of maximum collision separation to local mean free path, $Ma_\infty = 0.53$, $T_\infty = 300$K, $Kn = 0.043$, $n = 10^{-20}$m$^3$ .................................................. 115
Figure 7.15 Static grid, ratio of average cell side length to local mean free path, Ma∞ = 0.53, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ................................. 116

Figure 7.16 Static grid, ratio of maximum collision separation to local mean free path, Ma∞ = 0.53, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ................................. 116

Figure 7.17 Local Mach number contours past a vertical flat plate, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 118

Figure 7.18 Temperature contours past a vertical flat plate, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 118

Figure 7.19 Number density contours past a vertical flat plate, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 118

Figure 7.20 Streamlines past a vertical flat plate, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 119

Figure 7.21 Adapted grid, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ........................... 119

Figure 7.22 Adapted time step multiple, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 119

Figure 7.23 Ratio of average cell side length to local mean free path, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 120

Figure 7.24 Ratio of maximum collision separation to local mean free path, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 120

Figure 7.25 Static grid, ratio of average cell side length to local mean free path, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 120

Figure 7.26 Static grid, ratio of maximum collision separation to local mean free path, Ma∞ = 5.0, T∞ = 300K, Kn = 0.043, n = 10^{-20} m^3. ...................................................... 121

Figure 7.27 PDD and PSIR parallelisation results. ...................................................... 124

Figure 7.28 Combined PDD and PSIR results. ...................................................... 125

Figure 7.29 Performance of combined PDD and PSIR method relative to PDD method. ...................................................... 126

Figure 7.30 PDD results with a constant number of molecules per node. ...................................................... 127
List of Tables

Table 5.1 Grid adaption test case 1 ................................................................. 69
Table 5.2 Grid adaption test case 1 results ...................................................... 70
Table 5.3 Grid adaption test case 2 ................................................................. 72
Table 5.4 Grid adaption test case 2 results ...................................................... 73
Table 6.1 Domain decomposition test results .................................................. 103
Table 7.1 Collision rate test gas species molecular properties .......................... 107
Table 7.2 Ratio of simulated collision rate to theoretical value .......................... 108
Table 7.3 PDD and PSIR parallelisation results .............................................. 124
Table 7.4 Combined PDD and PSIR results .................................................... 125
Table 7.5 PDD results with a constant number of molecules per node ............... 126
Nomenclature

\( a \) speed of sound
\( \mathbf{c} \) molecule position vector, computational space
\( \overline{\mathbf{c}} \) average molecule speed
\( c_f \) skin friction coefficient
\( C_P \) coefficient of pressure
\( d \) molecular diameter
\( F_N \) ratio real to simulation molecules
\( l_m \) cell side length
\( \overline{l}_m \) average cell side length
\( Ma \) Mach number
\( N_m \) number of molecules
\( k \) Boltzmann’s constant, \( 1.380658 \times 10^{-23} \text{ JK}^{-1} \)
\( Kn \) Knudsen number
\( m \) molecule mass
\( m_r \) reduced molecule mass
\( n \) number density
\( S_{sf} \) stream species, of species group, fraction
\( \overline{S} \) cell boundary surface area vector
\( t \) time
\( \overline{t}_{mc} \) average collision time
\( T \) temperature
\( \mathbf{v} \) molecule velocity vector, physical space
\( V_c \) cell volume
\( \mathbf{x} \) molecule position vector, physical space
\( x, y, z \) physical domain coordinates
\( \eta, \xi, \zeta \) computational domain coordinates
\( \lambda \) local mean free path
\( \nu \) collision frequency
\( \omega \) temperature exponent of the coefficient of viscosity

Subscripts

\( \text{avg} \) average
\( \text{HS} \) hard sphere molecule model
\( \text{max} \) maximum
\( \text{min} \) minimum
\( \text{ref} \) reference value
\( \text{rem} \) remainder
\( \infty \) freestream value
Chapter 1

Introduction

1.1 Background and Motivation

This thesis deals with the direct simulation Monte Carlo (DSMC) method of analysing gas flows. The DSMC method was initially proposed as a method for predicting rarefied flows where the Navier-Stokes equations are inaccurate and it has now been extended to near continuum flows. The method models gas flows using simulation molecules which represent a large number of real molecules in a probabilistic simulation to solve the Boltzmann equation. Molecules are moved through a simulation of physical space in a realistic manner that is directly coupled to physical time such that unsteady flow characteristics can be modelled. Intermolecular collisions and molecule-surface collisions are calculated using probabilistic, phenomenological models. The fundamental assumption of the DSMC method is that the molecular movement and collision phases can be decoupled over time periods that are smaller than the mean collision time.

Two obstacles to the wide spread use of the DSMC method as an engineering tool are in the areas of simulation configuration, which is the configuration of the simulation parameters to provide a valid solution, and the time required to obtain a solution. For complex problems\(^1\), the simulation will need to be run multiple times, with the simulation configuration being modified between runs to provide an accurate solution for the previous run’s results, until the solution converges. This task is time consuming and requires the user to have a good understanding of the DSMC method. Furthermore, the computational resources required by a DSMC simulation increase rapidly as the simulation approaches the continuum regime. Similarly, the computational requirements of three-dimensional problems are generally two orders of magnitude

---

\(^1\) The expression “complex problems” refers to problems that contain mixed subsonic and supersonic flows, transition regimes, boundary layers, slip planes or chemically interacting flows.
1.2 Problem Summary

more than two-dimensional problems. These large computational requirements significantly limit the range of problems that can be practically solved on an engineering workstation or desktop computer.

While there are less computationally expensive alternative methods available for solving near continuum flows, e.g. Navier-Stokes based programs, these methods do not work well rarefaction effects are present. One example of where rarefaction effects are significant is in the micro-electro-mechanical systems (MEMS) [53]. The design of MEMS components requires the simulation of gas flows around microscale structures. It is possible to adapt Navier-Stokes based solvers to solve these types of problems by the use of special boundary conditions. However, this is a time consuming task that requires a very high level of user knowledge, and is not applicable to a standard engineering tool.

For a program to be a useful engineering tool, it should be simple to set up and run, and provide accurate results in a timely manner. It should warn the user when results are expected to be inaccurate and ideally, it should automatically adapt the program parameters to obtain a more accurate solution. However, as Harvey and Gallis state in their review of DSMC validation studies, [25], “writing and running a DSMC code is a demanding task that requires considerable skill, care and experience.”. In the comprehensive presentation of the DSMC method [7] Bird states "The objective is to develop a code that requires only the specification of the boundaries ... and the flow conditions. The program should itself generate the grid and, ideally it should adapt the grid to its optimal form as the flow develops."

1.2 Problem Summary

The aim of this thesis is to develop a DSMC implementation that allows a non-expert user to efficiently solve arbitrary problems with the DSMC method in both serial and parallel environments.

1.3 Principle Contributions

This thesis addresses the issues related to the development of a parallel, solution adaptive direct simulation Monte Carlo implementation. The principle contributions of this thesis arise from the formulation of different solution adaption and parallelisation
1.3 Principle Contributions

The integration of these algorithms into a single DSMC implementation results in a significant improvement in performance. The contributions made are:

- A novel molecule movement algorithm is developed which allows the efficient calculation of a molecule’s trajectory through an arbitrary quadrilateral grid. This movement algorithm combines the simplicity and accuracy of calculating the molecule movement in physical space with the simple cell indexing afforded by tracking the molecules computational space position. The algorithm achieves this by performing the molecule movement in physical space and then transforming the molecule’s final position into computational space. The computational space position is then used to determine the molecule’s new cell index and whether any surface/boundary interactions took place over the molecule’s trajectory.

- A multi-block grid adaption algorithm is developed. Using the intermediate results of the solution, the grid adaption algorithm calculates the number of cells and the distribution required to ensure that the grid meets the DSMC cell size requirements. The algorithm determines whether the distribution of cell size across a block would be more efficiently represented by splitting the block into two or more blocks and creating the required splits. Furthermore, the algorithm adjusts the ratio of real to simulation molecules to ensure that the number of molecules per cell is sufficient to ensure that the correct collision rate is maintained.

- A parallelised time step adaption algorithm is developed. This algorithm allows the time step of each cell to be set to a locally optimal value while still maintaining the ability to efficiently synchronise the solution process in a parallel implementation. Additionally, the algorithm is formulated such that the calculation of a new time step distribution is performed in parallel.

- Two different but complementary parallelisation methods are integrated. This integration of complementary parallelisation methods allows the use of the optimal parallel configuration for the available computational resources. Furthermore, the parallel implementation allows the parallel distribution to be dynamically changed to account for changes in the solution.

- An improved implementation of the stop-at-rise algorithm is developed. This modified algorithm incorporates information relating to the total solution progress.
with the current solution performance to more determine whether it is efficient to perform a domain decomposition repartitioning.

1.4 Outline

Chapter 2 presents an outline of the direct simulation Monte Carlo method and summarises the implementation issues that effect the accuracy and validity of the method.

Chapter 3 presents an analysis of the different movement algorithms. A novel movement algorithm is developed which allows the trajectory calculation of a molecule through an arbitrary quadrilateral grid.

Chapter 4 details the program structure of the baseline serial implementation.

Chapter 5 develops the solution adaption procedures implemented to ensure that the simulation is configured to meet the requirements for an accurate simulation.

Chapter 6 discusses two different parallelisation methods and details their integration into the serial implementation.

Chapter 7 presents the verification and results of the direct simulation Monte Carlo implementation developed in the previous chapters.

Finally, Chapter 8 presents conclusions and directions for future research.