University of Strathclyde
Department of Mechanical Engineering

Hybrid Molecular-Continuum Modelling of Nano-Scale Flows

Matthew Karl Borg

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

2010

Declaration of author's rights

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

The copyright of this thesis belongs to the author under the terms of the United Kingdom Copyright Acts as qualified by University of Strathclyde Regulation 3.50. Due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

Matthew K. Borg
June 2010

Abstract

Nanofluidics is an emerging technology of the 21^{st} century, and Molecular Dynamics (MD) is perhaps the most important numerical simulation method available to capture liquid phenomena at the smallest scales. MD is, however, far too computationally intensive, limiting it to only small systems and relatively short time-scales. To alleviate the overwhelming cost of full MD simulations, we describe in detail the development of a new hybrid method, which combines MD with a continuum fluid formulation in a computational framework of arbitrary geometries and in parallel.

We implement a new measurement tool box which is required for sampling properties in MD simulations. We describe in detail a method for measuring coarse-grained property fields on the mesh of the MD simulation for passing as molecular-to-continuum state-based boundary conditions to the continuum solver in hybrid simulations. We also describe new methods for applying continuum-to-molecular state- and flux-based boundary conditions on an MD mesh, which are of non-periodic type. A new feedback-loop state controller is developed for steering field properties such as density, velocity and temperature towards their target values in arbitrary regions of a molecular dynamics simulation. These methods are tested rigorously by showing that, for studying flows in equilibrium, they may replace the common and often limited periodic boundary conditions in pure MD simulations. Furthermore, we present results for a new method which controls pressure gradients in nano-channels of arbitrary geometries. This method is used to investigate mixing in a three-inlet one-outlet complex nano-channel using MD, and droplet deformation and breakup in the throat of a two-inlet two-outlet extensional-shear flow nano-channel.

Molecular dynamics and computational fluid dynamics codes are coupled in the hybrid framework. We describe the spatial and time coupling parts of the coupling framework and perform hybrid simulations, using the domain-decomposition method, of simple shear flows. The results of this scheme are compelling, as generally good agreement is observed when results are compared with full MD solutions.

Acknowledgements

It is a pleasure to thank all those who helped make this thesis possible. I would first of all like to extend my gratitude and deepest appreciation to my supervisor and mentor, Professor Jason M. Reese. His expertise, understanding, patience and continuous support have set an example which I hope to emulate.

I am also grateful to a former colleague, Dr. Graham B. Macpherson who developed the Molecular Dynamics solver, expounded and developed further within this research. He has supported me in a number of ways including immense technical advice, numerous fruitful discussions and answers to many queries.

Thanks also go to Chris Greenshields of OpenCFD Ltd. and to the entire team behind the development of the OpenFOAM software, without which, my research would not have been possible.

I would also like to acknowledge Dr. Tom Scanlon, my second supervisor for his continuous interest and the informal discussions we had, and to all my other colleagues and friends from the Multiscale research group for making my years at the University so rewarding.

This work was funded by the James Weir Foundation, and supported by the University of Strathclyde, for which I am also very grateful.

Finally, this dissertation is dedicated to my family who have provided unconditional love, support, words of encouragement and sacrifice throughout the course of my doctorate. I owe all my success and achievements to you.

Contents

List of Figures

List of Tables

Nomenclature

Symbols

A	cross-sectional area
$\mathbf{a}$	molecule acceleration
b	bin index
C	continuum sub-domain
C→M	continuum-to-molecular boundary conditions
e	total energy
$\mathbf{f}$	molecule force vector
f	arbitrary face on cell
$g(r, \cdot)$	radial distribution function
i, j	arbitrary pair of molecules
k_b	Boltzmann constant
K	molecule kinetic energy
K_P	controller proportional constant
L	length
M	molecular sub-domain
M→C	molecular-to-continuum boundary conditions
m	molecule mass
$\dot{m}$	mass flux
N	number of molecules
n_x	number of x
$\hat{\mathbf{n}}$	normal unit-vector
$\mathbf{P}$	stress tensor
$p(\cdot)$	probability density function
P	arbitrary cell
p	normal pressure
$\mathbf{p}$	molecule momentum
$\mathbf{r}$	position vector
r_{cut}	potential cut-off radius

S	number of measurement samples
T	temperature
t	time
Δt	time interval
U	potential energy
$\mathbf{u}$	continuum velocity
$\mathbf{v}$	molecule velocity
V	volume
w	virial
ΔX	cell size
x, y, z	spatial Cartesian co-ordinates
Z	zone

Greek

α	pressure controller blending coefficient
$\dot{\gamma}$	strain rate
δ_s	slip length
ϵ	characteristic energy scale
ζ	relative error
η	shear viscosity
θ	solution coupling relaxation parameter
κ	thermal conductivity
λ	constrained-Lagrangian relaxation parameter
μ	dynamic viscosity
ρ	number density
σ	characteristic length scale
τ_T	time constant
ϕ	property field
χ	molecular velocity scaling parameter

Superscripts

$*$	reduced unit
$miss$	missing
$fict$	fictitious molecules
$real$	existing (real) molecules
req	required (target) value
meas	measured value

Subscripts

av	average
b	boundary
c	continuum
$cont$	control
f	arbitrary face
i, j	pair of molecule indices
k	time index
m	molecular
$mols$	molecules within cut-off sphere
n	new time
o	old time
P	arbitrary cell
r	reference
s	sample
w	write / wall value
x, y, z	spatial Cartesian co-ordinates
Z	zone

Acronyms

ACF	Autocorrelation function
AICA	Arbitrary Interaction Cells Algorithm
CFD	Computational Fluid Dynamics
CNT	Carbon Nanotube
FCC	Face Centred Cubic
LJ	Lennard-Jones
MD	Molecular Dynamics
MFBM	Mean Force Boundary Model
MPBM	Mean Position Boundary Model
NPBC	Non-Periodic Boundary Conditions
PBC	Periodic Boundary Conditions
RDF	Radial Distribution Function
VACF	Velocity Autocorrelation Function
WCA	Weeks Chandler Anderson

Chapter 1

Introduction

"Begin at the beginning,", the King said, very gravely,
"and go on till you come to the end: then stop"
- Lewis Carroll

1.1 Motivation

In his inspiring talk "There's Plenty of Room at the Bottom" [9], Nobel prize winner Richard Feynman looked into the future of miniaturisation in an era when computer hardware notably occupied entire rooms. Since then, science and technology has taken leaps forward, giving way to micro- and nano-technology in many disciplinary fields, and generally experiencing a paradigm shift from "bigger is better" to "smaller and better" [10]. Microelectromechanical systems (MEMS) refer to microscale machines and systems in the range of 100 nm to 100 μm, while nanotechnology addresses systems and devices with lengthscales of 1nm – 100nm. In this research we are mainly interested in the fluid mechanics of nanoscale flows, which is also referred to as 'nanoflows' or 'nanofluidics'.

Downscaling of engineering devices and components brings numerous benefits, such as smaller volume-sizes, lightweight components, lower material consumption, lower manufacturing costs, and high heat transfer. For example, in aerospace applications, smaller and lower weight products are necessary to reduce the mass of the payload; in the electronics industry, microscopically small and highly effective heat exchangers are required for fast chip/circuitry cooling; in medical or health-related applications, lab-on-a-chip devices will help to analyse diseases of patients in shorter time periods, and are much cheaper than conventional methods and use much smaller fluid volumes (e.g., blood samples, and expensive reagents); in other MEMS applications, micro gas-turbines or fuel cells which are smaller, more powerful and long-lasting than standard batteries may power other microscale components such as micro-actuators (e.g., pumps, valves and motors) and micro sensors. A considerable number of MEMS applications are already on the market, and it is expected that many futuristic microscale devices will continue to be developed.

The laws of physics relating to conventional macroscopic modelling (with which we are so familiar), differ, sometimes substantially, from experimental observations of micro and nano-systems. Modelling laws do not always scale down in proportion to size. Inertial forces and weight for example become negligible, while diffusion, and the effects of surface tension and electrostatics become dominant. Flows along microchannels exhibit differences to the standard Navier-Stokes model. Mass flow rates are under-predicted, necessitating modifications to the formulation to include slip at the solid bounding surfaces; and non-linear pressure gradients are observed in long channels. Furthermore, in channels smaller than 8 nm it has been shown through the use of molecular dynamics, that the cross-channel velocity profile no longer resembles the Hagen-Poiseuille quadratic solution [11].

The state-of-the-art in nanotechnology, an emerging technology of the 21st century, can be said to be at the same level that computers and information technology were back in the 1950's [12]. In nanotechnology, the lengthscales of matter approach the atomistic regime, and there are clearly more constraints and limitations than microtechnology. In particular, substantially high resolution tools for manipulation, manufacturing, visualisation and measurement are still lacking. In manufacturing, there are geometrical limitations on the size and features of nano-structures that can be constructed using top-down techniques, such as micro lithographic processes, etching and micro-machining. Any feature size less than around 10nm requires bottom-up methods. Carbon nanotubes (CNT) or nano-pipes are created to bypass these limitations by a process of chemical vapour deposition (CVD). A major research field is the creation of new engineering materials with enhanced mechanical, electrical, optical, thermal, magnetic and/or chemical properties. For example, nano-sized particles are placed in a solution in order to enhance its conductivity, and simultaneously to solve the issue of micro-particle settling or agglomeration [10]. CNTs are especially useful for nanofluidic applications due to their highly attractive characteristics, including the high speed imbibition of liquids: liquid velocities in CNTs, including water, have been shown experimentally to be approximately 10,000 times higher than what is predicted by the Hagen-Poiseuille formulation [13]. A particular useful societal application would be high-throughput water desalination through nano-membrane CNT systems. Other applications could include filtration/separation of industrial fluids, lab-on-a-chip components, DNA screening and sensing, nano-jet reactors and CNT-pipettes [14]. The latter is the nano-scale equivalent of syringe needles, which may be used to inject drugs into living biological cells, in a controlled manner. There are still many obstacles inherent in the manufacturing of CNTs, such as uncontrolled introduction of bends in the tubes, closed-ended CNTs, internal closures, catalytic particles trapping at the ends of tubes, and general inhomogeneity in structure and properties. In future, scientists expect to overcome these challenges using atomistic resolution measurement and manipulation tools.

In the coming decades, nanoscale technology will revolutionise many industries, and provide more interesting solutions to applications in most disciplinary fields, despite the challenges it currently faces. Expected futuristic applications include nano-robots, nano-explorers, or nano-submarines, that use some form of dynamic propulsion systems to explore biological fluids [15]. These will have swimming abilities, communication systems, actuators that perform some form of function (e.g., surgical operations), nano-sensors to provide live-imagery, nano-power supply (e.g., energy scavenging from surrounding environments) and on-board collection devices for foreign samples.

1.2 Computational Methods

A major research field in nanotechnology is developing new design theory and modelling techniques to probe phenomena at these scales. Continuum methods are engineering design tools renowned for their accuracy and computational efficiency when applied to macroscopic systems. However, when scaled down to model the behaviour of micro- and nano-scale systems, their applicability and accuracy becomes questionable. Notably, it is the lack of atomistic detail in the description of boundary conditions and constitutive relations which invalidates them. The nanoscale hydrodynamic regime can be accurately treated by accounting for its atomistic constituents — a collection of discrete particles.

In this thesis we focus on nanoliquids and employ the Molecular Dynamics (MD) simulation method, as it represents the only approach to modelling liquids from first principles. As with almost all discrete methods, MD is extremely computationally expensive and is therefore restricted to only relatively small systems and for small time scales. To alleviate this, a multi-scale approach may be constructed with the purpose of focusing MD in only small regions of the problem domain where it is required, and couple it with a continuum formulation in a real-time simulation. A second drawback of MD is that the microscopic and macroscopic results are overwhelmingly dependent on the model for the intermolecular potential. In other words, one can tweak the potentials to obtain a desired result. The art of molecular dynamics lies in choosing the correct potentials for the fluids that need to be simulated.

1.3 Liquids

The mechanisms through which mass, momentum and energy are transported in liquid flows are somewhat different to those in solids and gases, and may be described using either continuum or molecular models. Unlike solids, liquids and gases have more thermal energy and therefore the constituent molecules can move about freely. Molecular motion in dilute gaseous systems, for example, consists of long free-flight motion interrupted by very short binary collisions, in which momentum abruptly changes. The mean free path is defined as the average distance a molecule travels between these collisions. In liquids, molecules are generally closely packed, resulting in multiple and frequent collisions. Momentum transfer in liquids is therefore dependent mostly on the intermolecular force field created by neighbouring molecules. On average, the distance between molecules in gaseous systems is typically one order of magnitude greater than the diameter of the constituent molecules, while intermolecular distances approach the molecular diameter in dense liquids. Liquids are therefore generally incompressible fluids, although compressibility becomes increasingly important to model in nanoscale systems.

From the continuum point of view, there are basically three specific fluid mechanical requirements if the standard Navier-Stokes-Fourier governing equations are deemed valid for modelling liquids at the micro or nano-scale [16]:

1. A **Newtonian framework** defines that mass and energy are independently conserved, and the sum of all forces on a fluid element is equal to the rate of change of momentum. The Newtonian assumption is generally always valid.

2. The **continuum approximation** implies that space and time are indefinitely divisible. More simply, the local macroscopic variables such as density, velocity, stress and heat flux are represented by averages over small volume elements that contain a relatively large number of molecules. The size of the volume element however needs to be small enough to describe the spatial and temporal variations of properties. The continuum approximation is almost always met in the bulk of a dense liquid, but remains questionable in very small atomistic simulations, and close to interfaces.

3. **Thermodynamic equilibrium** (or quasiequilibrium) stipulates that molecules undergo collisions within distances which are much smaller than the characteristic lengthscale of the flow. Liquid systems are generally dense and molecules are almost always in a state of collision, thus the stress/strain-rate and heat-flux/temperature-gradient relations can be assumed to be linear. If a flow is in thermodynamic equilibrium, then the no-slip boundary conditions may also be applied. Departure from thermodynamic equilibrium is violated in many parts of micro- and nano-scale liquid systems, and an alternative to the no-slip Navier-Stokes equations needs to be sought. It is often more difficult for liquids than gases to determine if the flow is in non-equilibrium because there is no well-advanced molecular based theory or a Knudsen number (ratio of mean free path to characteristic flow length) to guide us. The questions that need to be asked are: when are the no-slip boundary conditions not valid anymore; or, when do the constitutive relations fail to be linear. For example, non-Newtonian behaviour in liquids has been observed when the strain-rate exceeds a threshold value $\dot{\gamma} = \partial u/\partial y \geq 2\tau^{-1}$, [17] where $\tau = \sqrt{(m\sigma^2)/\epsilon} = 2.16059 \times 10^{-12}$ s is the time characteristic for liquid argon. Such strain-rates are generally unachievable, and the constitutive relations can be assumed linear in many parts of the flow domain. The occurrence of slip next to interfaces in nanoscale liquid flows is perhaps a more important issue. Phenomenological methods have been adopted within the continuum representation, either by incorporating slip boundary conditions, or else by using ‘apparent viscosities’ [18].

1.4 Project objectives

In this thesis, our goal is to develop and implement numerical tools to simulate the phenomena occurring in nanofluidic systems. In particular, we enhance the capabilities of a molecular dynamics (MD) computational software for simulating flows in arbitrary geometries. Engineering-related nanoscale devices do not consist of merely cubic domains in equilibrium with infinite periodicity, but encompass complex three-dimensional geometries, bends, structural protrusions, defects in material, channel inlets and outlets of different sizes and shapes, multi-species fluids (hardly pure fluids in real cases), and pressure gradients across domains. In addition, since molecular dynamics is extremely computationally intensive, an important objective is to develop a hybrid scheme for coupled molecular dynamics and continuum-based formulations, again for arbitrary geometries and in parallel. The hybrid code aims to reduce the computational overhead of purely MD systems by applying MD only in regions where atomistic detail and phenomena are to be captured (e.g., next to interfaces to overcome the issues of slip boundary conditions, effective transport coefficients, and non-linear constitutive terms) while applying continuum hydrodynamic models in bulk regions of the computational domain. The coupling problem is universally challenging, and is described in this thesis.

1.5 The OpenFOAM code

Two codes have been developed in this research within OpenFOAM® (Open Field Operation and Manipulation) [19]. OpenFOAM is an open-source set of C++ libraries originally created for Computational Fluid Dynamics' (CFD) solvers. C++ offers a more attractive method for writing, modifying, and structuring code. The first code which we use is the gnemdFOAM solver[1], which was developed in this research Group for simulating non-equilibrium molecular dynamics, and is available to download freely from [19]. To this code we add new methods for applying non-periodic boundary conditions and carrying out field measurements in MD simulations. The second code is the hybridFOAM solver which we developed to couple the gnemdFOAM solver with a CFD solver (sonicLiquidFOAM) within one framework.

This thesis consists of the concept development, computer programming and testing. It describes the methods of our work as clearly as possible without listing any of the code, but presents the results which the code has produced. Lines of code are not reproduced in this thesis for the benefit of readers who are not familiar with the C++ language. However, the full code is included in the enclosed CD.

---

[1] In the recent version of OpenFOAM, v 1.6, the non-equilibrium MD code has been renamed to mdFoam.

1.6 Present contributions — thesis overview

The molecular dynamics code used in this research, was developed entirely by Macpherson [20], and we briefly describe it in Chapter 2. Our main additional contributions are outlined below.

1. In Chapter 3 we implement a flexible measurement tool-box for measuring properties in MD simulation runs, which includes a general time-framework for sampling and averaging per arbitrary measurement. We implement a new method for measuring coarse-grained field properties that are required for applying molecular-to-continuum boundary conditions in our hybrid model.

2. The development and implementation of non-periodic boundary conditions for MD simulations in arbitrary geometries and in parallel are described in Chapter 4. This includes cell-based continuum-to-molecular state-controllers, mass flux boundary models and force field boundary models. These boundary conditions are required for applying continuum-to-molecular boundary conditions in our hybrid model.

3. In Chapter 5 we test the new boundary conditions in simple and complex systems. We verify that the non-periodic boundary conditions may replace periodic boundary conditions in MD simulations of equilibrium, 1D uniform flows and shear flows. Furthermore, we propose and test a different method for applying pressure gradient flows in Poiseuille-type nano-channels. In particular, we show that the method can be applied to model mixing and droplet deformation under an extensional flow in complex nano-channels.

4. A new multi-mesh hybrid method is developed in Chapter 6 that combines MD with the Navier-Stokes continuum formulation in a general coupling framework. The framework handles spatial and temporal coupling in arbitrary geometries and in parallel. The hybrid solver is tested on simple start-up Couette flow problems, an oscillating shear flow and a shear flow with a complex coupling region.

1.6.1 Papers

The research results in this thesis are reported in the following published papers:

1. G. B. Macpherson, M. K. Borg and J. M. Reese, Generation of initial molecular dynamics configurations in arbitrary geometries and in parallel, *Molecular Simulation*, 33(1): 97–115, 2008.

2. M. K. Borg and J. M. Reese, "A hybrid particle-continuum framework", *6th International Conference on Nanochannels, Microchannels and Minichannels*. 2008. Darmstadt, Germany.

3. M. K. Borg, G. B. Macpherson and J. M. Reese, “Controllers for imposing continuum-to-molecular boundary conditions in arbitrary fluid flow geometries”, accepted, *Molecular Simulation*, 2010.

Chapter 2

Molecular Dynamics

Nothing exists except atoms and empty space;
everything else is opinion.
- Democritus (Greek philosopher, 460-370bc)

2.1 Introduction

Molecular Dynamics (MD) [21, 22] is a widely used computational technique to study the macroscopic behaviour and microscopic detail of a group of molecules which interact for a sufficient period of time. MD is frequently used in many disciplinary fields, including bio-chemistry (DNA, proteins, lipid membranes, and other bio-molecules), bio-physics (self-organisation of complex bio-molecules), bio-engineering (drug-analysis and transport), mechanical engineering (micro- and nano-scale flows, and non-equilibrium flows), and material science (phase transitions, fundamental macroscopic properties of matter, and rheology of non-Newtonian fluids). For liquids in particular, MD represents perhaps the only direct method of simulation from first principles [23].

In a molecular dynamics simulation, the time-evolution of the system is obtained by numerically integrating Newton's equation of motion for all system molecules, concomitant with potential energy functions applied between all pairs of molecules that determine the net forces. MD examines processes over time-scales that are far below the realm of those considered in conventional continuum fluid mechanics. Coupled with the demanding intermolecular force computation, an MD algorithm is therefore very computationally intensive, especially for a relatively large number of molecules and long time-scales. In this regard, parallelisation is crucial to speed up simulation runs. For example, standard personal computers may only handle systems up to $\mathcal{O}(10^4)$ molecules in serial. For a relatively short simulation run of $\mathcal{O}(1)$ ns, the computation would take approximately a few hours. High-performance computers, with multi-processor nodes can conduct parallel simulations of $\mathcal{O}(10^6)$ molecules in total over a few days. Possible future processing improvements may extend the application of MD simulations to larger length-scales and longer time-scales.

The application of boundary conditions in MD simulations is not fully developed in the literature. Periodic boundary conditions and explicit solid-wall boundary conditions are the two most common types generally applied to equilibrium and non-equilibrium systems. Although the former method has many limitations (§4.1), it allows the system to approach thermodynamic equilibrium in the direction in which periodicity is applied, while eliminating the undesired effects that arise in finite (non-periodic) systems. The wall boundary conditions are required for investigating fluids close to solids (e.g., contact angles, slip, adsorption, molecular layering, heat-transfer etc.), and may also provide the means for applying more accurate MD-derived boundary conditions in continuum Navier-Stokes simulations. Boundary conditions in MD are discussed in more detail in Chapter 4.

The purpose of any MD simulation is to compute the static and dynamic properties of a particular fluid system from the trajectories of the constituent molecules. This generally requires extracting time-averaged hydrodynamic fields of density, stress, temperature, velocity, pressure etc. or transport properties, for fluid mechanics simulations. We describe techniques and models for measurement in more detail in Chapter 3. The biggest drawback is the lack of experimental data with which the MD results may be validated, for example, when testing new boundary conditions.

2.1.1 Development of a new MD code

A new MD code was developed in previous work [20], which we adopt also in this thesis, and in which we want to test new algorithms such as the continuum-to-MD controllers, non-periodic boundary conditions in MD and hybrid MD-CFD methods. The MD code is implemented in OpenFOAM [19], an open-source C++ toolbox primarily used for CFD, and is available to download freely from [19].

Although there are available codes (e.g. LAMMPS, NAMD, AMBER, GROMACS and DL_POLY) which are widely used by the MD community, they do not offer a framework robust enough to handle arbitrary geometries[1] (where the walls of the channel are defined by a molecular wall) for simulating MD in more realistic systems, in addition to arbitrary shaped 'boundaries' of the non-periodic, non-wall type, necessitated also for hybrid MD-continuum systems. While the former can be argued to be plausible albeit immensely time-consuming to set up, the latter is still a limiting issue which may be meliorated by discretising the computational space by a collection of individual cells of arbitrary shape. OpenFOAM provides this framework. The implementation of our MD solver in the open-source architecture has therefore been performed relatively quickly, since it uses existing OpenFOAM libraries, such as the mesh[2], geometric fields, boundary conditions, particle

---

[1] The neighbour list, a conventional method for computing pair forces in complex geometries, has been shown in [20] to produce unfavourable features in non-equilibrium flows, interprocessor communication and spatially resolved properties.

[2] The mesh is vital for computing pair forces using a more robust linked cells algorithm [24], local measurement of properties and localised constraints in complex meshes.

tracking and particle cloud libraries which are all parallelised using open MPI [25] for communications. In addition, OpenFOAM facilitates coupling of the new MD solver with any of the available continuum solvers, since it is coded using the same numerics, structure and programming language.

The MD solver which is described in this chapter is similar to that presented in [20], which is for monatomic, multi-species Lennard-Jones molecules in arbitrary geometries and in parallel. During this present research, the MD code was improved elsewhere to include many new features, such as user-defined intermolecular potential models (e.g., Coulomb potential, Maitland Smith) and rigid polyatomic molecules (e.g., water TIP4P, and carbon-nanotubes).

2.1.2 Chapter overview

In Section 2.2 we give a brief overview of the molecular dynamics method in arbitrary geometries. Reduced units are presented in Section 2.3, while Section 2.4 describes the technique we use for initialising MD systems.

2.2 MD in arbitrary geometries

2.2.1 Potentials

Consider a domain of arbitrary geometry, defined by an unstructured arbitrary polyhedral mesh, similar to that used for standard finite-volume Computational Fluid Dynamics (CFD). The MD fluid occupies the domain geometry and consists of point masses for monatomic particles — which we will refer to as “molecules” — that interact through a pair-wise potential $U(r_{ij})$, where $r_{ij} = |\mathbf{r}_{ij}|$ and $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ is the separation vector between a pair of molecules (i, j). The common Lennard-Jones (LJ) 12-6 potential (see Figure 2.1) is used here, because of its simplicity and its thorough validation:

$$U_{LJ}(r_{ij}) = 4\epsilon\left(\left(\frac{r_{ij}}{\sigma}\right)^{-12} - \left(\frac{r_{ij}}{\sigma}\right)^{-6}\right), \tag{2.1}$$

where σ is the characteristic length scale and ϵ is the characteristic potential energy. The positions $\mathbf{r}_i$ and velocities $\mathbf{v}_i$ of molecules in the system evolve according to standard Newtonian dynamics:

$$m_i\mathbf{a}_i = \mathbf{f}_i, \tag{2.2}$$

where $\mathbf{a}_i = \ddot{\mathbf{r}}_i$ and m_i are the i^{th} molecule’s acceleration and mass respectively. The net force on a molecule i is computed from the sum of intermolecular pair-forces:

$$\mathbf{f}_i = \sum_{j=1(\neq i)}^{N} -\nabla U(r_{ij}), \tag{2.3}$$

where $-\nabla U(r_{ij}) = \mathbf{f}_{ij}$ is the intermolecular force between molecules i and j within the system of N molecules.

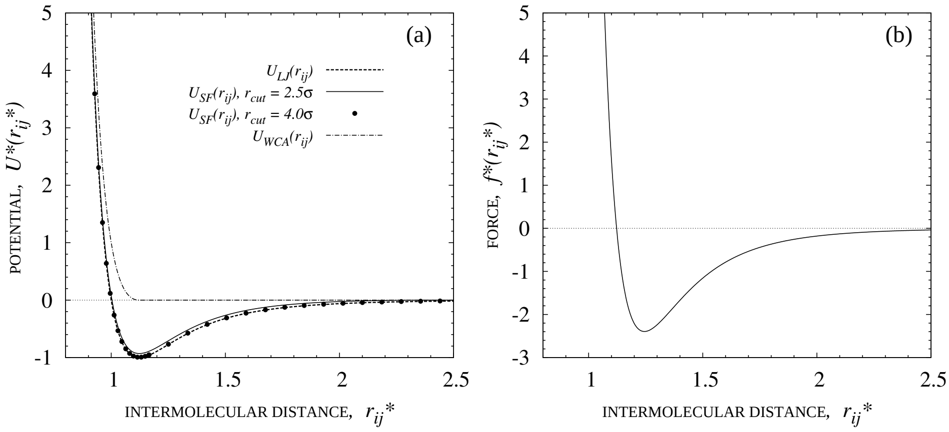


Figure 2.1: The Lennard-Jones (LJ) 12-6 (a) potential, $U(r_{ij})$ and (b) the corresponding shifted force $f_{ij} = -|\nabla U(r_{ij})|$ as a function of intermolecular distance. Graphs are presented in reduced units (§2.3).

The intermolecular force calculation is by far the most computationally consuming task of the MD algorithm. Therefore, it is a common implementation measure to truncate the LJ potential (2.1) at a radial cut-off r_{cut} and to shift it so that the potential is zero at $r_{ij} = r_{cut}$:

$$U_{SF}(r_{ij}) = \begin{cases} U_{LJ}(r_{ij}) - U_{LJ}(r_{cut}), & r_{ij} \leq r_{cut}, \\ 0, & r_{ij} > r_{cut}, \end{cases} \tag{2.4}$$

where $U_{LJ}(r_{cut})$ is the energy value at which the LJ potential is truncated:

$$\begin{aligned} U_{LJ}(r_{cut}) \quad = \quad & \left[\left(\frac{r_{cut}}{\sigma}\right)^{-12} - \left(\frac{r_{cut}}{\sigma}\right)^{-6}\right] \\ & -\frac{12 r_{cut}(r_{ij} - r_{cut})}{\sigma^2}\left[\left(\frac{r_{cut}}{\sigma}\right)^{-14} - \frac{1}{2}\left(\frac{r_{cut}}{\sigma}\right)^{-8}\right]. \end{aligned} \tag{2.5}$$

The corresponding force variation (see Fig 2.1(b)) for the LJ shifted potential is given in reduced units [20] by:

$$f^*(r_{ij}^*) = \frac{48\epsilon^*}{\sigma^{*2}}\left[\left(\frac{r_{ij}^*}{\sigma^*}\right)^{-14} - \frac{1}{2}\left(\frac{r_{ij}^*}{\sigma^*}\right)^{-8} - \frac{r_{cut}^*}{r_{ij}^*}\left(\left(\frac{r_{cut}^*}{\sigma^*}\right)^{-14} - \frac{1}{2}\left(\frac{r_{cut}^*}{\sigma^*}\right)^{-8}\right)\right] r_{ij}^*. \tag{2.6}$$

The pair-force vector on molecule i due to j is $\mathbf{f}_{ij} = f(r_{ij})\mathbf{r}_{ij}/r_{ij}$.

In this research we also use the Weeks Chandler Anderson (WCA) potential, which is defined as the repulsive part of the LJ potential. It is constructed from (2.4) by selecting $r_{cut} = 2^{1/6}\sigma$. Other potentials have not been investigated, but in the new molecular dynamics code of OpenFOAM version 1.6, any potential model may be coded, or inputted in a tabulated manner straightforwardly.

Any number of multi-species fluid and solid molecules may be simulated. This is implemented in the code by assigning an identification (id) number to each molecule. For $K \geq 1$ species in the system there should be $K(1+K)/2$ number of combinatorial interacting potentials defined. The pair-force on two molecules (i,j) is computed from the interaction potential which is assigned for that combination of ids.

2.2.2 Leapfrog numerical integration scheme

The Verlet Leapfrog algorithm [22] is used to integrate numerically the equations of motion of all molecules at each MD time-step, Δt_m. The choice of Δt_m is a trade-off between computational cost, accuracy, and robustness of the code. Small integrating time-steps may strongly conserve momentum and energy between molecules, albeit with an increase in computational cost. On the other hand, a large time-step may blow up the algorithm, as a result of a molecule moving into the deeply repulsive range of one or more molecules, leading to a cascading process of overlapping molecules. In this research we choose a value for Δt_m between 5.4×10^{-15} s and 1.08×10^{-14} s.

The Leapfrog algorithm is described in the following procedure for one MD time-step, which is then repeated until the end of the simulation run.

1. Update half the velocity,

$$\mathbf{v}_i(t + \Delta t_m/2) = \mathbf{v}_i(t) + 1/2\mathbf{a}_i(t)\Delta t_m. \tag{2.7}$$

2. Advance all molecules to their new positions,

$$\mathbf{r}_i(t + \Delta t_m) = \mathbf{r}_i(t) + \mathbf{v}_i(t + \Delta t_m/2)\Delta t_m. \tag{2.8}$$

   The motion of molecules within arbitrary meshes is handled by an efficient tracking scheme [26]. Each time a molecule crosses an internal face of the mesh during its motion, the molecule immediately updates and stores the new occupant cell. The process is repeated until the whole motion of the molecule is performed, which in general cases may cross multiple faces and cells during one MD time-step. If it collides with a boundary face, then the boundary prescribes an ‘action’ (see §4.5.1), depending on whether it is of type ‘cyclic’, ‘processor’, ‘specular wall’ etc.

3. The cell-occupancy data structure is updated. In this step, the link between cells of the mesh and the residing molecules is renewed, because these have been updated by the tracking scheme in the previous step. In addition, molecules at the boundary of the mesh are referred across periodic and processor boundaries [20]. This data structure is important for the intermolecular force calculation step, and the measurements of volume-based field properties (§3).

4. Compute the intermolecular forces on all the molecules, $\mathbf{f}_i(t + \Delta t_m)$ from equation (2.3), and subsequently the acceleration,

   $$\mathbf{a}_i(t + \Delta t_m) = \mathbf{f}_i(t + \Delta t_m)/m_i. \tag{2.9}$$

   This step utilises the Arbitrary Interacting Cells Algorithm (AICA) scheme [24] that computes the pair-forces in complex meshes, which is mainly a generalisation of the linked cells algorithm [21] (see Fig 2.2).

5. Update the second half of the velocity,

   $$\mathbf{v}_i(t + \Delta t_m) = \mathbf{v}_i(t + \Delta t_m/2) + 1/2\mathbf{a}_i(t + \Delta t_m)\Delta t_m. \tag{2.10}$$

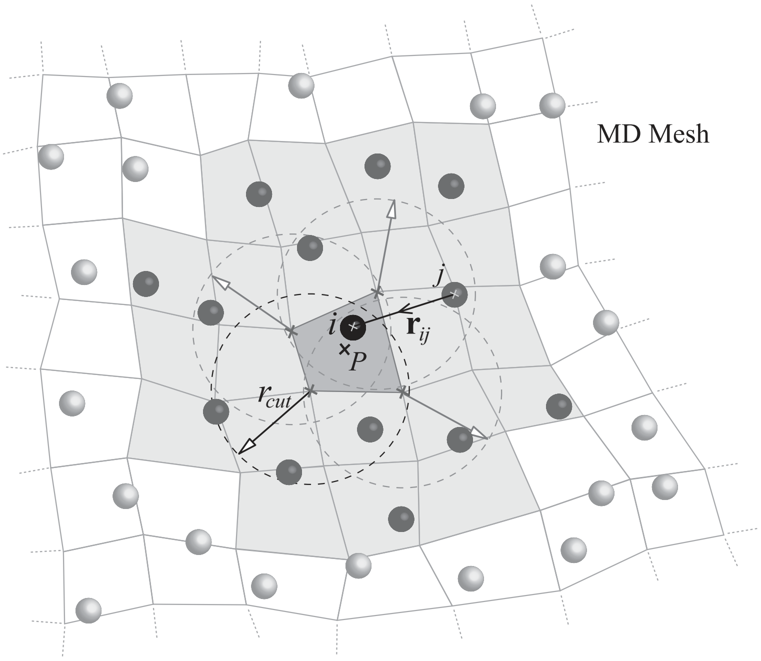


Figure 2.2: Schematic of the Arbitrary Interacting Cells Algorithm (AICA). A molecule i (black) which is located in a cell P (dark grey) interacts only with molecules j (dark grey) located within its own cell and within the direct interaction cell list (light grey). The list of interaction cells is constructed once, in the beginning of the simulation.

2.2.3 Boundary Conditions

Molecular dynamics simulations require boundary conditions to be applied at the extremities of the domain. In practice, the choice of boundary conditions will depend on the problem or case being simulated. Since a major part of this thesis is dedicated to the development of new boundary conditions in MD, we refer the reader to Chapter 4 for more in-depth information.

Periodic boundary conditions (PBCs) are by far the most frequently used method, in which the central simulation box (generally cubic) of molecules is surrounded by an infinite number of replicas of itself. These images of molecules are defined by transformations which are related to the symmetry of the central simulation box, and are updated every time-step to include also the motion of molecules occurring within the central box. A molecule in proximity of an arbitrary periodic boundary, therefore, interacts also with referred molecules transformed from the opposite end of the domain.

Wall boundary conditions are constructed by defining a solid lattice structure at a boundary region of the MD domain, and incorporating interactions of fluid-solid and solid-solid types. In some cases, it is also possible to replace the wall atoms by a specular or diffuse implicit wall boundary. This model reverses the perpendicular component of velocity so that colliding molecules remain confined within the domain. Walls of any arbitrary shape, roughness, and design features may be created using the lattice generating utility, as described in §2.4 and [20, 27].

Parallel processing

Our method for applying MD simulations in parallel is based on domain-decomposition (e.g., [28]), in which the mesh is partitioned along its internal faces into smaller sub-meshes. Each sub-mesh is then assigned to a different processor, and a replica of the MD code is used by each processor to solve the MD dynamics on its own mesh. The overall simulation proceeds in parallel on each processor, and inter-processor communication occurs at frequent and well-defined times of the run, via the open MPI library [25].

Processor boundary conditions are applied where inter-processor cuts are made, which are similar in construction to periodic boundary conditions. If a molecule collides with a processor boundary during its motion it is deleted from its current processor mesh, transmitted to the coupled boundary, and subsequently continues the rest of its trajectory on the new processor mesh. After the tracking step, molecules in close proximity of the boundary get referred across processors [20, 24], so that they interact with real molecules at the boundary (see Fig 2.3). While in this thesis we have used pair-intermolecular potentials (e.g., LJ, WCA), other many-body potentials and force fields available in the literature (e.g., AMBER, CHARMM) may also be implemented for parallel processing using this technique of constructing ‘halos’ of referred molecules at processor boundaries. For force fields that are not truncated, and that require molecules to interact with all other molecules in the system, the halo of referred molecules is large, and computational savings due to parallel processing is still achieved albeit minimal in comparison to serial processing when the minimum sum convention is used.

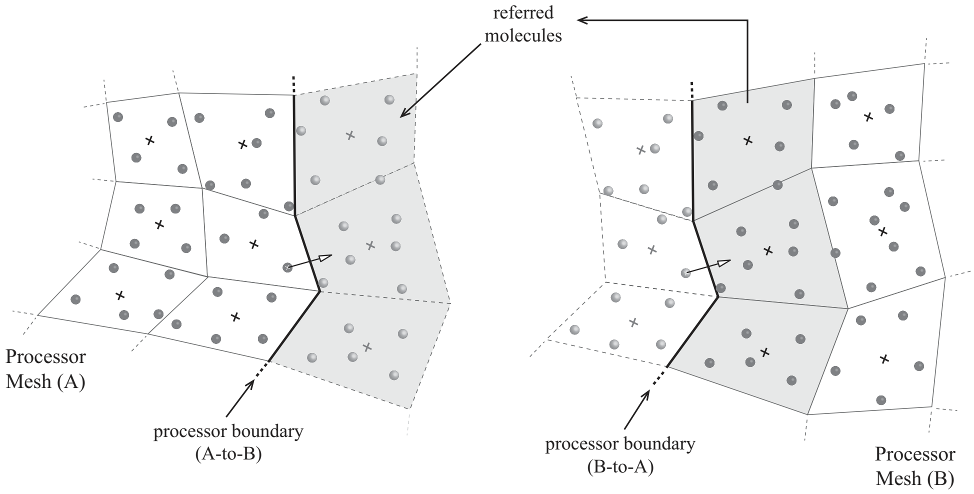


Figure 2.3: Schematic of processor boundary conditions applied in arbitrary geometries. Referred molecules are transferred across processor meshes from internal boundary regions using the cells on the mesh. The list of referred cells are constructed once only, in the beginning of the simulation.

2.2.4 Measurements

The main purpose of an MD simulation is to analyse the results it produces. This can be carried out in a number of ways. First there are the trajectories $\mathbf{r}_i(t)$ of molecules (or other molecular properties such as id number, velocity $\mathbf{v}_i(t)$, force $\mathbf{f}_i(t)$, acceleration $\mathbf{a}_i(t)$, etc.), which can be analysed in instantaneous picture or movie format using a visualisation software such as ParaView® (distributed with OpenFOAM). Second, the MD code can perform measurements of any required property during the simulation run using the new measurement tool-box which is described in Chapter 3. For volume-based hydrodynamic properties (such as density, velocity, temperature, pressure etc.) these are accumulated at the end of the Leapfrog time-step procedure, and averaged over longer periods of time. Fluxes are measured as and when they occur, in particular, during the tracking step. Different techniques may be adopted to measure volume based properties, such as using 1D or 2D distributions, 3D fields in arbitrary geometries, or zones. Other measurement properties may include transport coefficients and correlation functions.

2.3 Reduced Units

It is common practice to convert all MD quantities into reduced units [22], using only three fundamental normalisation parameters for length σ_r, mass m_r and energy ϵ_r. All other properties are variants of these parameters. Values for the normalisation parameters are typically chosen to correspond to the LJ characteristic values, σ, ϵ and m respectively. For argon these are $\sigma = 0.34 \times 10^{-9}$ m, mass $m = 6.69 \times 10^{-26}$ kg and $\epsilon = 120k_b = 1.65678 \times 10^{-21}$ J, where k_b is the Boltzmann constant. In this thesis, a reduced quantity is labelled by an asterisk (*) and may be converted back to SI units, as shown in Table 2.1 for some common quantities. We present most of our results in reduced units.

Table 2.1: Table of reduced units for properties carried out in this research. Reference values are taken to be the Lennard-Jones potential characteristics for Argon. A property q in SI units is converted into a reduced unit q^* by its reference property q_r, given in the last column of the table.

	Fundamental quantities	
length	$l^* = l/\sigma_r$	$\sigma_r = 0.34 \times 10^{-9}$ m
energy	$\epsilon^* = \epsilon/\epsilon_r$	$\epsilon_r = 1.65678 \times 10^{-21}$ J
mass	$m^* = m/m_r$	$m_r = 6.6904 \times 10^{-26}$ kg
	Derived quantities	
time	$t^* = t/t_r$	$t_r = (\sqrt{m_r \sigma_r^2/\epsilon_r}) = 2.16059 \times 10^{-12}$ s
force	$f^* = f/f_r$	$f_r = (\epsilon_r/\sigma_r) = 4.87288 \times 10^{-12}$ N
acceleration	$a^* = a/a_r$	$a_r = (m_r \epsilon_r/\sigma_r) = 7.28340 \times 10^{-13}$ m s^{-2}
velocity	$u^* = u/u_r$	$u_r = (\sqrt{\epsilon_r/m_r}) = 157.364$ m s^{-1}
density	$\rho^* = \rho/\rho_r$	$\rho_r = (\sigma^{-3}) = 1702.22$ m^{-3}
temperature	$T^* = T/T_r$	$T_r = (\epsilon_r/k_b) = 120$ K
pressure	$p^* = p/p_r$	$p_r = (\epsilon/\sigma^3) = 42.153 \times 10^6$ N m^{-2}
viscosity	$\eta^* = \eta/\eta_r$	$\eta_r = (\sqrt{\epsilon_r m_r}/\sigma_r^2) = 9.10753 \times 10^{-5}$ kg m^{-1} s^{-1}

2.4 Initialisation of MD systems in arbitrary geometries

In our simulations, we carry out a set of pre-processing steps in order to obtain initialised MD systems for use in subsequent MD runs. We describe these processes as follows (see Fig 2.4):

1. Define the domain geometry by creating a mesh for the case. One may use the blockMesh utility supplied in OpenFOAM, or convert meshes from external software using the mesh-conversion utilities [19]. An example of the latter is shown in §5.2.2 for complex meshes, using Pro/ENGINEER® as the domain drawing tool and GAMBIT® as the meshing tool.

2. Use pre-processing utilities to define zones of arbitrary shape in the domain (Appendix C). Zones are groups of cells (cell-zones) or groups of faces (face-zones) on the mesh and are required for many operations, including measurement of local properties (see Chapter 3) and imposing control or non-periodic boundary conditions in arbitrary sub-regions of the domain (see Chapter 4).

3. Generate initial molecular lattice structures for solids or liquids within individual cell-zones of the domain using the pre-processing utility in [20, 27]. This utility employs a filling algorithm of an expanding cube of unit cells, based on input properties such as density, lattice structure, bulk velocity, and temperature (see Appendix C.1).

4. Apply a short MD simulation so that the initial lattice structure relaxes to a fluid. Controllers for density, temperature and velocity are simultaneously applied in the

domain in order to accurately converge the target spatial properties of the fluid. The state-controllers are described in more detail in Section 4.4.

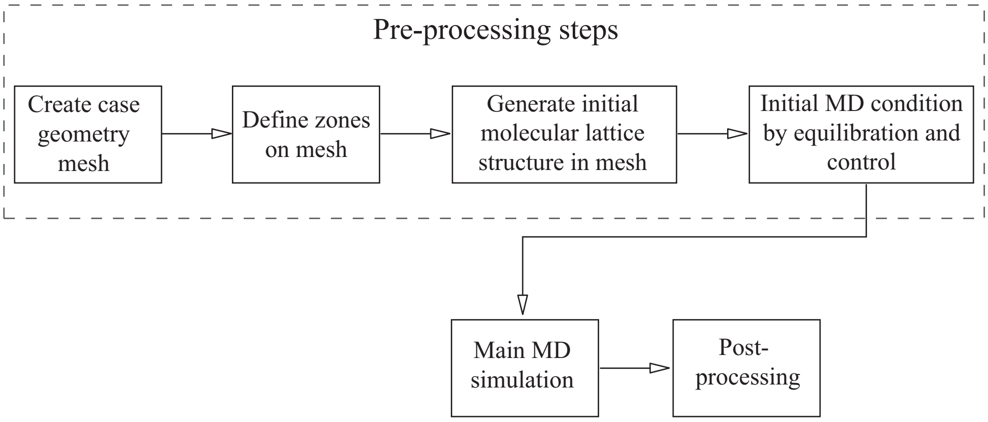


Figure 2.4: Route map for initialising MD systems.

Chapter 3

Measuring Properties in MD

> *It is probably true that the moment the simulation run*
> *is finished, you will remember something important*
> *that you have forgotten to calculate.*
> - Allen and Tildesley [22].

3.1 Introduction

An indispensable task in Molecular Dynamic simulations and hybrid MD-continuum simulations is extracting meaningful and accurate macroscopic measurements (with low statistical scatter) from the trajectories of molecules and their properties as they change in time. In view of this, we have implemented a flexible and manageable measurement "tool-box" in our MD code. To this we added a number of *models* for sampling properties such as thermodynamic and hydrodynamic continuum field properties (e.g., density, pressure, temperature, stress and velocity), or other measurements fundamental to MD (e.g., correlation functions, transport coefficients, and so on). The architecture allows the user to select any number of these available measurement models for a particular MD simulation. Models have been implemented in a general way, so that no extra coding is necessary across different MD cases. In future work, the tool-box may be extended to include new measurement models without the need to replicate any code which is used by the existing models.

Key to the measurement of an arbitrary property is the creation of a new general time-scheme. The scheme defines the times at which a particular property needs to be sampled, time-averaged, written out to the hard disk, and/or transferred to the continuum formulation (if it is a hybrid simulation). Properties of the time-scheme for an arbitrary measurement are chosen and modified by the user.

3.1.1 Measurements: during or after the run?

As hard-disk drive capacity shows a trend of exponential growth, an increasingly popular measurement technique is to store the Lagrangian fields (molecular positions, velocities, etc.) as lists of raw data on the hard-disk after every few MD time-steps of a simulation run. A separate post-processing step may then be carried out, which reads in the raw fields so that measurements of required properties may be computed. The objective of this technique is to focus the processing power on the demanding intermolecular force computations, and then measure properties at later stages without the need for expensive re-simulations. Furthermore, the domain may be re-meshed with a coarser cell-size after the simulation run is complete, in order to increase the statistical accuracy of the cell-based measurements[1].

Despite these advantages, we have noted many drawbacks of this technique. To start with, the sampling interval of a measurable property is limited by the size of the write interval, therefore frequent write-outs will demand greater hard-disk storage space, especially for larger systems. An example of a medium-sized MD system ($\mathcal{O}(10^5)$ molecules), requires $\mathcal{O}(10)$ megabytes per MD time-step, and at least $\mathcal{O}(10^2)$ gigabytes of storage space for $\mathcal{O}(10^4)$ write out steps. Moreover, in systems with variable number of molecules, and write intervals of a few MD time-steps, it is almost impossible to compute accurately fluxes of properties solely from the Lagrangian fields. Finally, coupling MD with a continuum formulation in a *direct* hybrid simulation may only be achieved if an "on-the-fly" measurement technique is adopted, rather than the aforementioned "simulate-then-measure" technique.

In view of these disadvantages, we mainly use the real-time measurement technique in our research, so that only reasonable storage space is required. We note that the computational effort of computing fields of data is still not as demanding as the intermolecular force step.

3.1.2 Chapter overview

We begin by demonstrating how the models are implemented within the MD code in Section 3.2. In Section 3.3 we describe the general time-scheme used to measure an arbitrary property. The formulations for measuring volume-based continuum properties are presented in Section 3.4, whereas Section 3.5 gives an overview of the different spatial techniques we use for measuring these properties within MD domains. Measuring fluxes and other MD properties is described in Section 3.7.

---

[1] The mesh is also used by the linked cells algorithm for computing truncated intermolecular forces in the MD simulation. Therefore sizes of cells must be small to avoid waste of computation [20].

3.2 The fieldPropertiesDict input file

The fieldPropertiesDict dictionary is an input file in which all measurement models are listed for a particular MD simulation case (see example in Fig 3.1). The dictionary is divided into five measurement sub-dictionaries: densityFields, velocityFields, temperatureFields, pressureFields, and otherFields. In an arbitrary sub-dictionary one may select any number of measurement models that correspond to the particular continuum property. The otherFields sub-dictionary include all those measurement models related to MD, such as correlation functions, pair-distribution functions, transport coefficients and average molecular property fields (mass, momentum, forces and energies). The reason for separating these measurement models into the state fields is so as to provide molecular-to-continuum (M→C) coupling *links* based on state-properties, for hybrid MD-continuum simulations.

3.3 Time-scheme

We present a variable time-scheme to measure a continuum property, Q, at regular time-intervals in a MD simulation run. The write-interval, Δt_w, is a time period after which Lagrangian fields and other data are written to the hard-disk. The total number of write-out steps for the duration of the MD simulation is $t_e - t_s/\Delta t_w$, where $t = (t_s, t_e)$ are the start and end times of the simulation.

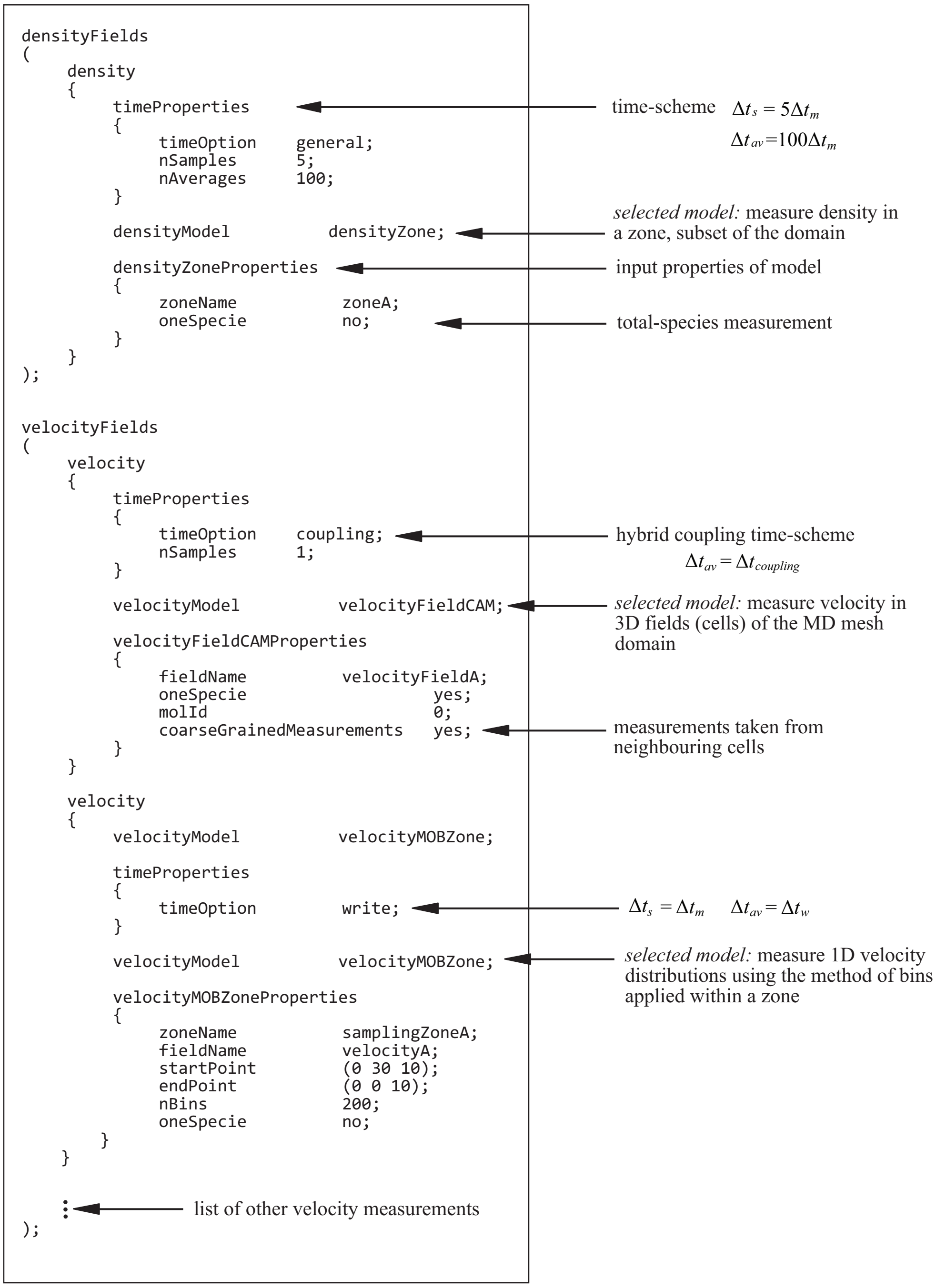
Q is measured using two sets of time-intervals: *sampling*, Δt_s, which defines the time at which instantaneous samples are taken and cumulatively stored to the field, and the *averaging* interval, Δt_{av}, which defines the time-period over which the added samples are time-averaged (see Fig 3.2). The relationship between the two periods is the number of samples, a positive integer given by:

$$S = \frac{\Delta t_{av}}{\Delta t_s}. \tag{3.1}$$

At an arbitrary time t of a simulation run, at which S samples of Q are taken during the previous Δt_{av}, the mean measurement may be defined by:

$$\langle Q(t) \rangle = \frac{1}{S} \sum_{k=1}^{S} Q(t_k), \tag{3.2}$$

where $Q(t_k)$ is the macroscopic variable sampled at an instantaneous time t_k, $(t - \Delta t_{av} < t_k \leq t)$. The time-average of an arbitrary *molecular* variable, q, may differ from (3.2), as follows:



```
densityFields
(
    density
    {
        timeProperties
        {
            timeOption    general;
            nSamples      5;
            nAverages     100;
        }

        densityModel          densityZone;

        densityZoneProperties
        {
            zoneName          zoneA;
            oneSpecie         no;
        }
    }
);


velocityFields
(
    velocity
    {
        timeProperties
        {
            timeOption    coupling;
            nSamples      1;
        }

        velocityModel          velocityFieldCAM;

        velocityFieldCAMProperties
        {
            fieldName         velocityFieldA;
            oneSpecie                  yes;
            molId                      0;
            coarseGrainedMeasurements   yes;
        }
    }

    velocity
    {
        velocityModel          velocityMOBZone;

        timeProperties
        {
            timeOption         write;
        }

        velocityModel          velocityMOBZone;

        velocityMOBZoneProperties
        {
            zoneName          samplingZoneA;
            fieldName         velocityA;
            startPoint        (0 30 10);
            endPoint          (0 0 10);
            nBins             200;
            oneSpecie         no;
       }
    }

    ⋮
);
```

time-scheme $\Delta t_s = 5\Delta t_m$

$\Delta t_{av} = 100\Delta t_m$

selected model: measure density in a zone, subset of the domain

input properties of model

total-species measurement

hybrid coupling time-scheme

$\Delta t_{av} = \Delta t_{coupling}$

selected model: measure velocity in 3D fields (cells) of the MD mesh domain

measurements taken from neighbouring cells

$\Delta t_s = \Delta t_m$ $\Delta t_{av} = \Delta t_w$

selected model: measure 1D velocity distributions using the method of bins applied within a zone

list of other velocity measurements

Figure 3.1: The fieldPropertiesDict input file. Examples of density and velocity measurements are shown in their respective sub-dictionaries.

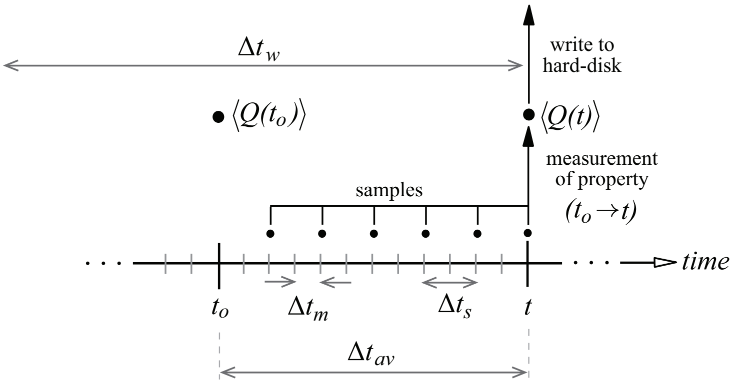


Figure 3.2: Measurement time-scheme for an arbitrary property Q. The time-line represents that of an MD simulation run, with time-step Δt_m and write interval Δt_w. Choice of Δt_s and Δt_{av} are dependent on the property being sampled.

$$\langle q(t)\rangle = \frac{\displaystyle\sum_{k=1}^{S}\sum_{i=1}^{N(t_k)} q_i(t_k)}{\displaystyle\sum_{k=1}^{S} N(t_k)}, \tag{3.3}$$

where $N(t_k)$ is the instantaneous number of molecules.

This is intended to create a general framework, so Δt_s and Δt_{av} may be selected arbitrarily, depending on the property being measured. It may sometimes be useful for example to set $\Delta t_s = \Delta t_m$, and $\Delta t_{av} = \Delta t_w$. The latter implies that once a mean property is computed it is stored immediately to disk. However if Q is to be used in a hybrid simulation, then we may want to set $\Delta t_{av} = \Delta t_{coupling}$, where $\Delta t_{coupling}$ is the coupling time-interval, while $\Delta t_w \geq \Delta t_{av}$.

The code for this time-scheme is implemented once, however, a time ‘clock’ is created independently for each measured property. We consider this an important and unique aspect of our measurement scheme, since different properties generally require variable timings for sampling statistically accurate means.

3.4 Models for volume-measurement

Typical hydrodynamic variables are commonly measured using “bins” or “cells” within local parts of the MD simulation domain. In this section, we present expressions used to measure macroscopic properties, such as number-density, velocity, temperature and pressure, from molecules that reside within an arbitrary cell P, of volume V_P, during a time period $t - \Delta t_{av} \to t$.

For multi-species fluids, a measurement can be computed either on a per-species or a total-species basis. Here we present all properties for total-species. When a per-species computation is performed, the computation excludes all other species molecules within the cell by using the molecule's identification (id) number.

3.4.1 Density

The number density is:

$$\langle (\rho_N)_P \rangle = \frac{\langle N_P \rangle}{V_P} = \frac{1}{SV_P} \sum_{k=1}^{S} N_P(t_k), \tag{3.4}$$

where $N_P(t_k)$ is the number of molecules occupying cell P at an instantaneous time.

The mass density is:

$$\langle (\rho_M)_P \rangle = \frac{1}{SV_P} \sum_{k=1}^{S} \sum_{i=1}^{N_P(t_k)} m_i. \tag{3.5}$$

3.4.2 Velocity

Two techniques for measuring velocity are described. The first method averages a set of instantaneous velocity measurements (as in equation (3.2)):

$$\langle \mathbf{u}_P \rangle = \left\langle \frac{\mathbf{p}_P}{M_P} \right\rangle = \frac{1}{S} \sum_{k=1}^{S} \frac{\sum\limits_{i=1}^{N_P(t_k)} m_i \mathbf{v}_i(t_k)}{\sum\limits_{i=1}^{N_P(t_k)} m_i}, \tag{3.6}$$

where $\mathbf{p}_P$ and M_P are the sums of momentum and mass respectively within P. The second technique involves averaging momentum and mass separately and computing the velocity at the end of the time-averaging period (as in Eqn (3.3)):

$$\langle \mathbf{u}_P \rangle = \frac{\langle \mathbf{p}_P \rangle}{\langle M_P \rangle} = \frac{\sum\limits_{k=1}^{S} \sum\limits_{i=1}^{N_P(t_k)} m_i \mathbf{v}_i(t_k)}{\sum\limits_{k=1}^{S} \sum\limits_{i=1}^{N_P(t_k)} m_i}. \tag{3.7}$$

Tysanner and Garcia [29] refer to these techniques as the Sample Average Measurement (SAM) and the Cumulative Average Measurement (CAM) respectively. A bias of velocity is shown to exist [29] using the SAM method (3.6) only when applied to fluids in non-equilibrium. The error, albeit minimal in some cases, is mainly due to the existence of correlations with other hydrodynamic variables. The CAM method, equation (3.7), is preferred because it measures the correct unbiased *mean* velocity over a particular time-interval in non-equilibrium systems. Since successive instantaneous samples have highly fluctuating numbers of molecules residing within the measurement cell P, the CAM method applies the correct weighting to the statistical averages.

3.4.3 Temperature

Temperature is measured within a cell P in a similar way to the Cumulative Average Measurement method (e.g., Ref. [30]):

$$\langle T_P \rangle = \frac{1}{3k_B} \frac{\displaystyle\sum_{k=1}^{S} \sum_{i=1}^{N_P(t_k)} m_i |(\mathbf{v}_i(t_k) - \mathbf{u}_P(t_k))|^2}{\displaystyle\sum_{k=1}^{S} N_P(t_k)}, \tag{3.8}$$

where $\mathbf{u}_P(t_k)$ is the instantaneous streaming velocity within the cell, $\mathbf{v}_i(t_k)$ is the (laboratory) velocity of a molecule, and $\mathbf{v}_i(t_k) - \mathbf{u}_P(t_k)$ is its peculiar or thermal velocity.

To improve the statistics of the temperature measurement, we time-average velocity $\langle \mathbf{u}_P \rangle$ within P, instead of using the cell-instantaneous velocity $\mathbf{u}_P(t_k)$. This is however ambiguous as the average velocity will only be known *after* the averaging time-period is completed, and not *during* the interval when it is in fact needed. We address this issue by performing a lagged velocity-temperature measurement, as depicted in Figure 3.3. Velocity is computed over shorter time-intervals than temperature: $\Delta t_{av}^{(u)} < \Delta t_{av}^{(T)}$. Therefore, the instantaneous temperature measurements always use the latest time-averaged velocity measurement.

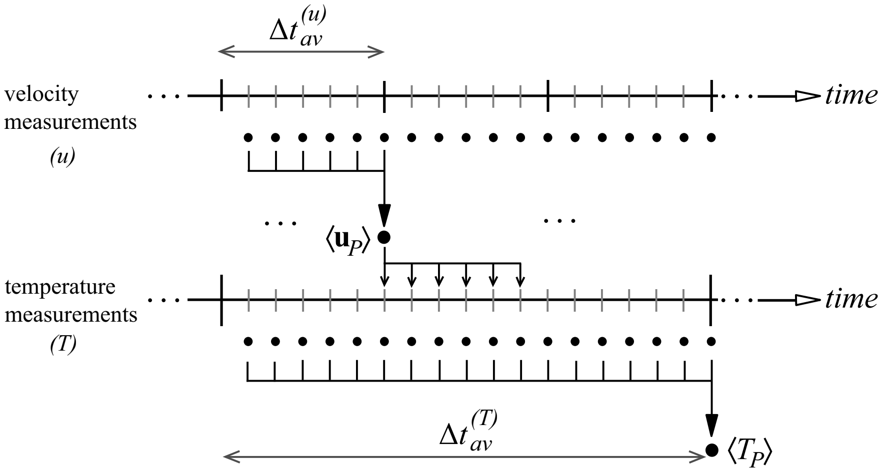


Figure 3.3: The proposed lagged velocity-temperature measurement scheme.

3.4.4 Pressure

The average pressure tensor is computed from the Irving-Kirkwood formula [31]:

$$\langle \mathbf{P}_P(t_k)\rangle = \frac{1}{V_P S}\sum_{k=1}^{S}\sum_{i=1}^{N_P(t_k)}\left\{ m_i(\mathbf{v}_i(t_k)-\mathbf{u}_P(t_k))(\mathbf{v}_i(t_k)-\mathbf{u}_P(t_k)) + \frac{1}{2}\sum_{j=1(\neq i)}^{N_{mols}(t_k)} f(r_{ij})\frac{\mathbf{r}_{ij}}{|\mathbf{r}_{ij}|}\mathbf{r}_{ij}\right\}, \tag{3.9}$$

where the vector-vector operation is an outer product that produces a second-rank tensor. Moreover, we employ the same velocity-pressure lagged measurement scheme that we used for temperature (see Fig 3.3) so that $\langle \mathbf{u}_P\rangle$ replaces $\mathbf{u}_P(t_k)$ in equation (3.9).

The normal pressure is the trace of the pressure tensor: $p = 1/3(P_{xx}+P_{yy}+P_{zz})$, and written as follows:

$$\langle p_P(t_k)\rangle = \frac{1}{3V_P S}\sum_{k=1}^{S}\sum_{i=1}^{N_P(t_k)}\left\{ m_i|(\mathbf{v}_i(t_k)-\mathbf{u}_P(t_k))|^2 + \frac{1}{2}\sum_{j=1(\neq i)}^{N_{mols}(t_k)} f(r_{ij})r_{ij}\right\}. \tag{3.10}$$

3.4.5 Energies

The time-averaged potential energy and kinetic energies per molecule within a cell P may be computed using the template equation (3.3):

$$\langle U_P\rangle = \frac{\sum_{k=1}^{S}\sum_{i=1}^{N_P(t_k)}\frac{1}{2}\sum_{j=1(\neq i)}^{N_{mols}(t_k)} U(r_{ij})}{\sum_{k=1}^{S} N_P(t_k)}, \tag{3.11}$$

$$\langle K_P\rangle = \frac{\sum_{k=1}^{S}\sum_{i=1}^{N_P(t_k)}\frac{1}{2}m_i|\mathbf{v}_i(t_k)|^2}{\sum_{k=1}^{S} N_P(t_k)}. \tag{3.12}$$

The total energy is then the sum of potential and kinetic terms, i.e. $\langle e_P\rangle = \langle U_P\rangle + \langle K_P\rangle$.

3.5 Techniques for measurement

We now describe different spatial techniques for measuring properties within an MD domain. The equations for common macroscopic properties were presented for an arbitrary cell P in Section 3.4 and are not repeated in this section.

3.5.1 Method of Zone (MOZ)

The method of zone (MOZ) technique measures an arbitrary property within a local subregion of the MD domain. The 'sampling zone', which consists of a group of connected cells (n^Z_{cells}) on the mesh, may be of any shape (e.g. spherical and cuboid regions), size, and location in the domain; the sampling region may be flexibly applied as a subset of the MD domain or made equal to the global domain. Notably, the measurement will return the spatial mean of the sampling region, and consequently no spatial variations of properties may be resolved within the zone using the MOZ.

The method is implemented by using the cell-occupancy data structure of the MD code. Every cell stores inexpensive links to all residing molecules, and the links get updated every MD time-step. Therefore, an instantaneous measurement involves looping over molecules that occupy a cell, and over all the cells that belong to the zone. This technique is optimal in terms of computational efficiency because it prevents looping over all molecules within the domain. Equations presented in Section 3.4 need to be altered to include this implementation. For example, the zonal number density is given by:

$$\langle (\rho_N)_Z \rangle = \frac{1}{S} \frac{\sum_{k=1}^{S} \sum_{P=1}^{n^Z_{cells}} N_P(t_k)}{\sum_{P=1}^{n^Z_{cells}} V_P}. \tag{3.13}$$

An example of the input requirements for this measurement model within the fieldPropertiesDict is shown in Figure 3.1.

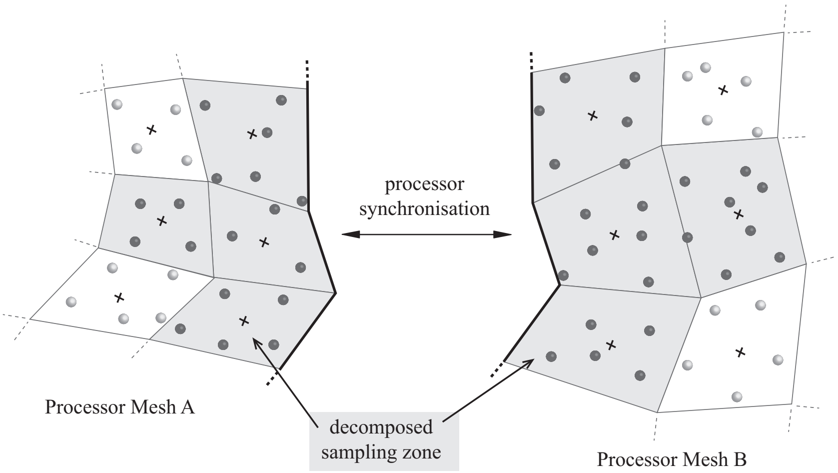


Figure 3.4: The method of zone (MOZ) for parallel processing.

Parallelisation

During parallel processing, the mesh (also the sampling zone) is decomposed on different processor meshes, as shown in Figure 3.4. An instantaneous measurement therefore calls for a parallel synchronisation step between all n_{proc} processors, at each sampling time-step.

For example the total number of molecules occupying zone Z, at an instantaneous time t_k is deduced from all processors by:

$$N_Z(t_k) = \sum_{p=1}^{n_{proc}} \sum_{P=1}^{n^Z_{cells}(p)} N_P(t_k). \tag{3.14}$$

3.5.2 Method of Bins (MOB)

The method of bins (or "binning method") is a common and useful technique for measuring distributions of properties within nanoflow applications, such as nano-pores and nano-channels, where the properties in the domain may vary spatially. The technique essentially partitions whole or part of the domain into bins or cells, and uses the volume-measurement expressions per cell, as presented in Section 3.4. We describe three MOB techniques for 1D, 2D and 3D fields.

1D method

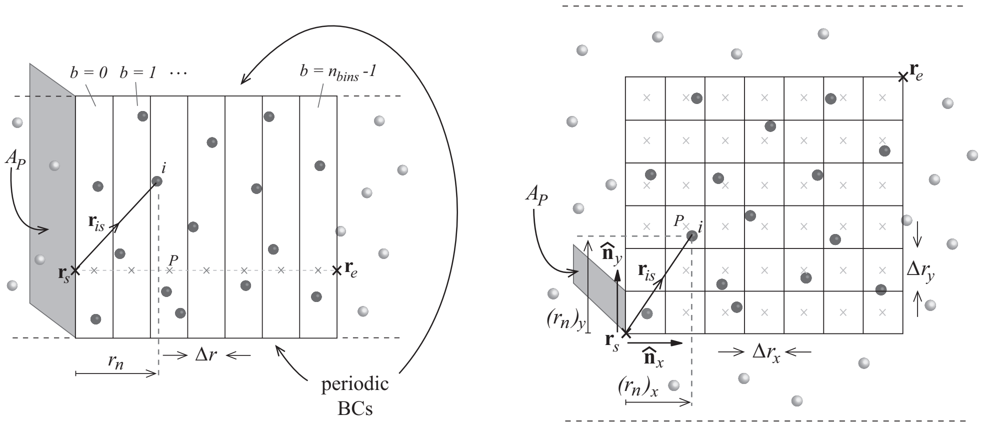
Consider a start point $\mathbf{r}_s$, an end point $\mathbf{r}_e$ and n_{bins} number of bins, as shown in Figure 3.5(a). The volume of one bin P is therefore equal to $V_P = \Delta r A_P$, where A_P is the cross-sectional area of the bins (typically of the domain) and $\Delta r = (|\mathbf{r}_s - \mathbf{r}_e|)/n_{bins}$ is the uniform width of one bin. An array (or histogram) is constructed for each measurement property that needs to be cumulatively sampled for the duration of Δt_{av}. For example, the velocity measurement in equation (3.7) requires two arrays, one for the numerator and one for the denominator. The size of a histogram is equal to n_{bins}, having bin indices running over $\{0, 1, \ldots, n_{bins} - 1\}$. At a sampling time instance, measurements of properties are appended within the appropriate bins. This is performed by looping through all molecules of the domain and computing for each arbitrary molecule i the bin index $b_i = \text{floor}(r_n/\Delta r)$, where r_n is the normal distance between $\mathbf{r}_i$ and $\mathbf{r}_s$:

$$r_n = (\mathbf{r}_i - \mathbf{r}_s) \cdot \frac{\mathbf{r}_e - \mathbf{r}_s}{|\mathbf{r}_e - \mathbf{r}_s|}. \tag{3.15}$$

If b_i lies within the bounds of the array, the properties of the molecule are cumulatively added to the bin entry of the respective histogram. At the averaging time instance, the cumulative properties in each bin are used to compute the mean properties by using the time-averaged expressions in Section 3.4. After the averaging is performed and stored to disk, all entries in the arrays are reset to zero, so that a new set of measurements can take place for $t \rightarrow \Delta t_{av}$.

2D method

The 2D method of bins is essentially an extension of the 1D MOB to plane or surface sampling. The square plane is defined by a starting (origin) point, $\mathbf{r}_s$, an end point, $\mathbf{r}_e$, a unit-vector, $\hat{\mathbf{n}}_x$, and two sets of numbers of bins, n^x_{bins} and n^y_{bins} that segment the plane in



(a) 1D (b) 2D

Figure 3.5: Schematic for sampling properties using the method of bins for (a) 1D and (b) 2D distributions. Note that the back and front planes are periodic for both cases.

the $x-$ and $y-$direction of the local co-ordinate system (Fig 3.5(b)). In this case the bin volumes are typically smaller, and are given by $V_P = A_P \Delta r_x = L_P \Delta r_x \Delta r_y$. Histograms of data consist of a matrix of size $n^x_{bins} \times n^y_{bins}$. The algorithm for measurement of properties proceeds in a similar fashion as described in the 1D MOB, however a bin is found using two indices, $[(b_i)_x, (b_i)_y]$, from two normal distances $(r_n)_x$ and $(r_n)_y$ between the molecule i at $\mathbf{r}_i$ and the starting point.

1D and 2D MOB in a Zone

The 1D and 2D MOB techniques are implemented independently of the mesh, so that measurements may easily be sampled at high resolutions without changing the mesh characteristics. Despite this, the method is not an optimal one, since a loop is carried out over all molecules within the domain to check if they contribute to a measurement bin or not.

Alternatively, the MOB and MOZ techniques may be merged, so that the distributions are sampled only within a user-defined sampling zone. As a result, the method is more efficient, since looping is performed only on those molecules that occupy the sampling zone, not the global domain. In addition, it is more appropriate for this combined method to sample local distributions, without the limitations of the system geometry, or the location of the periodic boundaries. An example of the input requirements for this measurement model within the fieldPropertiesDict is shown in Figure 3.1.

3D fields method

The 3D bin method requires the entire three-dimensional space of the MD domain to be partitioned into finite-volume cells. In light of this, we use the same mesh as that used by our MD component, which is required by the arbitrary interacting cells algorithm (AICA). Geometric-fields are used to store cell-properties, in a similar manner as hydrodynamic variables are defined in CFD formulations. A volume-based geometric-field is essentially an array that has its size equal to the number of cells on the mesh, n_{cells}, and its properties defined at cell-centres. These fields are populated with measurements (§3.4) similar to the 1D and 2D methods, yet avoiding the tedious step of finding the bin-index based on the global position of molecules. Instead, we use the robust cell-to-molecular occupancy data structure: links to molecules that occupy an arbitrary cell P are stored. All molecules occupying the cell contribute to the measurement within the same cell. After the averaging time-period is completed, the accumulated data within each cell-entry of the field is time-averaged and then the field is written out to hard disk in a format read by any 3D visualisation software (e.g., ParaView®). Furthermore, in a hybrid simulation, a sub-set of the field of averaged properties may serve as the target data for imposing molecular-to-continuum boundary conditions on the continuum CFD formulation. An example of the input requirements for this measurement model is shown in Figure 3.1.

Mesh properties

The MD mesh is used by three components of an MD simulation: (1) the AICA scheme, for computing intermolecular pair-forces within a cut-off sphere in unstructured meshes, (2) for measuring properties (as well used in the hybrid coupling scheme), and (3) for controlling macroscopic properties (see §4.4).

A compromise is required between three aspects: accuracy, computational cost, and resolution. The AICA scheme requires that the cells on the mesh are adequately small ($\Delta X \sim 2\sigma$) so as to minimise the number of computed pair-forces. Additionally, too small a cell size ($\Delta X < 1\sigma$) extends the number cells over which the algorithm must loop, thus resulting in a computational overhead. Resolution is a characteristic which also favours small cell-sizes, because properties may be better resolved spatially. However, their accuracy requires that enough sampling space and time-averaging periods are allowed so that the averaged cell-property converges to the mean with low statistical uncertainty. Computational efficiencies that are gained from using small cell-sizes in the AICA scheme, may in turn be forfeited to longer MD time-averaging periods in order to measure cell-properties within the required statistical error bars.

Coarse-grained 3D fields method

We devise a method of *coarse-graining* which aims to improve the statistical accuracy of measurements without largely affecting the spatial resolution. The method is a combination between the method of zones (MOZ) (§3.5.1) and the 3D fields method of bins. An arbitrary cell P on the mesh measures a property from molecules within its own cell P, in addition to molecules from other neighbouring cells that are within a specified range (or radius) from P. For practical purposes we use the intermolecular potential cut-off radius r_{cut} as the size of the 'spherical' measurement zone. After a property is computed in the local zone it is then assigned to the central cell P.

Important applications of this technique include direct measurements of properties that vary over short time-scales, and hybrid MD-continuum methods that require the fluctuations inherent in MD-derived cell-properties to be rectified sufficiently before being imposed as boundary conditions in the CFD formulation. The main advantage is that the method produces accurate mean measurements over shorter averaging time-intervals. It is therefore a tractable solution for controlling a state in small cell-sizes of a mesh, which rely on feedback measurements. The main concern is that large sampling volumes per cell may not capture properly the spatial resolution of measurements.

Parallelisation

Parallelisation for the 1D and 2D fields method of bins is solved using a similar synchronisation step described above for MOZ (§3.5.1), whereupon each processor reconstructs the full array of an arbitrary property, entry-by-entry. The 3D field method, with coarse-graining disabled, does not require a synchronisation step since measurements are taken within individual cells. At the end of the simulation, the fields written on the hard disk are reconstructed to the full fields by using typical OpenFOAM post-processing utilities. However, if the 3D fields method is applied with coarse-graining enabled, a similar syncronisation step is required in real-time simulation.

3.6 Fluxes

Macroscopic fluxes measured from molecules crossing faces of cells in the mesh have also been implemented as they may provide useful results, for example, in time-varying unsteady-state MD flows. In addition, future work may consider using real-time measurements of fluxes as a method of coupling with a continuum formulation (e.g., see [5, 8]). The method consists of using the robust particle-tracking algorithm [26], which keeps track of molecules as they cross faces of one cell or more in a time-step Δt_m.

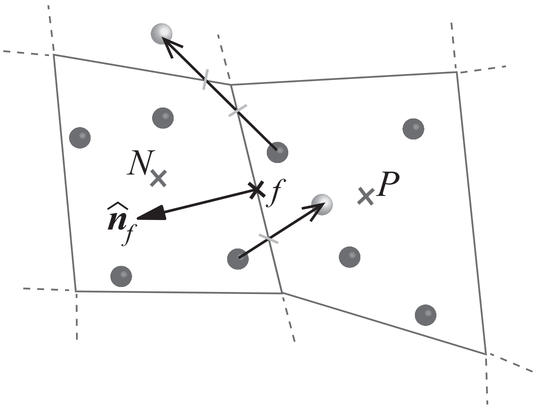


Figure 3.6: Fluxes measured from molecules crossing an arbitrary face f between two touching cells (P, N).

Consider two neighboring cells (P, N), of 3D arbitrary shape, that are connected together through a common face f (see Fig 3.6). The face normal vector $\hat{\mathbf{n}}_f$, points in the direction out of the owner cell (in this case cell P), and towards the neighbour cell (N). During the tracking scheme of the leapfrog algorithm it is possible to identify and keep count of all molecules that cross f, and the direction of crossing by $sgn(\mathbf{v}_i \cdot \hat{\mathbf{n}}_f)$, where,

$$sgn(x) = \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{if } x = 0, \\ -1, & \text{if } x < 0. \end{cases} \tag{3.16}$$

We define a positive flux as being a cross-over in the direction of the normal face unit-vector (owner-to-neighbour), and a negative flux in the opposite direction. This convention is kept for all our definitions. No contribution to the flux occurs if a molecule moves on the face f, that is, for the case when $sgn(\mathbf{v}_i \cdot \hat{\mathbf{n}}_f) = 0$.

3.6.1 Measured properties

We present expressions for computing the mass, momentum, and energy fluxes at an arbitrary face, f, of the mesh.

Mass flux

The average mass flux at f may be computed by,

$$\langle \dot{m}_f \rangle = \frac{1}{\Delta t_{av}} \sum_{i}^{\Delta N_f(t \to t_n)} m_i \; sgn(\mathbf{v}_i \cdot \hat{\mathbf{n}}_f), \tag{3.17}$$

where $\Delta N_f(t \to t_n)$ is the total number of molecules that cross face f during the time period $t \to t + \Delta t_{av}$. The mass flux density may be computed by $\langle \dot{m}_f \rangle / A_f$, where A_f is the area of face f.

Momentum flux

The average momentum flux at f may be computed by,

$$\langle \dot{\mathbf{p}}_f \rangle = \frac{1}{\Delta t_{av}} \sum_{i}^{\Delta N_f(t \to t_n)} m_i \mathbf{v}_i. \tag{3.18}$$

The $sgn(x)$ function is not used here since $\mathbf{v}_i$ establishes the direction of the flux. Similarly the momentum flux density may be computed, $\langle \dot{\mathbf{p}}_f \rangle / A_f$.

Energy flux

The mean kinetic, potential, and total energy fluxes may also be computed by,

$$\langle \dot{U}_f \rangle = \frac{1}{\Delta t_{av}} \sum_{i=1}^{\Delta N_f(t \to t_n)} U_i \; sgn(\mathbf{v}_i \cdot \hat{\mathbf{n}}_f), \tag{3.19}$$

$$\langle \dot{K}_f \rangle = \frac{1}{\Delta t_{av}} \sum_{i=1}^{\Delta N_f(t \to t_n)} K_i \; sgn(\mathbf{v}_i \cdot \hat{\mathbf{n}}_f), \tag{3.20}$$

$$\langle \dot{e}_f \rangle = \langle \dot{U}_f \rangle + \langle \dot{K}_f \rangle, \tag{3.21}$$

where $U_i = 1/2 \sum_{j \neq i}^{N_{mols}} U(r_{ij})$ and $K_i = 1/2 m_i |\mathbf{v}_i|^2$. Similarly the energy flux densities may be computed by dividing the energy fluxes by A_f.

3.6.2 Flux fields

The tracking algorithm is used to resolve fluxes at all faces of the mesh. During the averaging time-interval, mass, momentum and energy fluxes associated with molecular-face crossings are accumulated and stored in a geometric surface field, which is of size equal to the number of faces on the mesh, n_{faces}. At the end of the averaging time-period the expressions for fluxes (§3.6.1) are computed at all faces on the mesh. The time-scheme which we use for measuring fluxes is similar to that used by volume sampling, but ignores the sampling time, Δt_s. Fluxes are conservative measurements (once a molecule crosses a face it contributes to the flux at that face) while state measurements may be taken at uniform correlated/uncorrelated sampling times.

Arbitrary sampling surfaces can also be chosen on the mesh to define spatial flux averages. The net flux at a face-zone is computed by summing the fluxes computed at individual faces of the zone — that are taken from the readily sampled geometric-fields — and then divided by the number of faces in the zone n^Z_{faces}. Some examples include two dimensional planes that can be distributed in the streamwise direction (and normal to the flow) of a nanochannel, or else at boundaries of an enclosed volume (e.g., cubic or roughly-spherical shapes). The latter may, for example, be used to measure fluxes emanating from a cell, or for controlling fluxes within cells that are set from continuum target fluxes in a hybrid simulation. Furthermore, it can be shown that fluxes at faces of cells may also be used to reconstruct hydrodynamic variables at cell-centres, instead of the volume-based measuring methods, albeit with higher statistical scatter. A similar method of planes technique [32] has been proposed in the literature for measuring properties such as pressure [33] and heat flux [34, 35] at many uniformly distributed 2D planes in the domain. We avoid this method since the technique generally requires a large number of planes, which in our case introduces a high computational cost due to the large number of cells that are required. Furthermore, the method published in the literature is only applicable to 1D distributions applied to simple cuboid-shaped geometries.

3.7 MD properties

MD properties, other than state and flux hydrodynamic properties, may also be computed using the measurement tool-box; these include correlation functions and transport properties. These measurements may be selected in the otherFields part of the fieldPropertiesDict input file.

3.7.1 Radial distribution function

The Radial Distribution Function (RDF), is a pair correlation function that represents the mean local structure of the molecular fluid from a reference molecule, and is dependent on spatial co-ordinates and mean state, $g(\mathbf{r}, \rho, T)$. It is sometimes referred to as the probability density of finding another molecule at a radius $\mathbf{r}$ from an arbitrary molecule. In §4.5.5 we need the RDF in order to derive numerical force fields to be imposed at non-periodic fluid boundaries.

The RDF is derived for a monoatomic, isotropic fluid in equilibrium, so that it may be stored as a 1D distribution of radial dependency only, $g(r, \dots)$. Consider a reference cell P, in the middle of the MD domain, surrounded by a spherical cell-zone of radius $r_{max} \sim 5\sigma$ (see Figure 3.7). The 1D method of bins may be applied, so that a histogram is constructed of size equal to the number of bins, n_{bins}. The RDF is generally based on 'drawing' a series of concentric spherical shells, each of bin width $\Delta r = r_{max}/n_{bins}$, between the position of an arbitrary reference molecule at $\mathbf{r}_i$ ($r = 0$) and the outer shell ($r = r_{max}$). Subsequently, at each sampling time-instance, the number of molecules that reside within every spherical shell are counted and added to the respective bin entry of the distribution. The bin index is found using $b_i = \text{floor}(|(\mathbf{r}_j - \mathbf{r}_i)|/\Delta r)$, where $0 \leq b_i \leq n_{bins} - 1$.

In our method we compute the time- and cell-averaged RDF by summing instantaneous RDF's of all i molecules that occupy P throughout the averaging time period Δt_{av}. For the r^{th} bin of the distribution, the RDF value is,

$$\langle g(r) \rangle = \frac{1}{\langle \rho_Z \rangle V_{shell}(r)} \frac{\displaystyle\sum_{k=1}^{S} \sum_{i=1}^{N_P(t_k)} [N_j(r, t_k)]_i}{\displaystyle\sum_{k=1}^{S} N_P(t_k)}, \tag{3.22}$$

where $\langle \rho_Z \rangle$ is the time-averaged density in the spherical cell-zone (see MOZ §3.5.1), $[N_j(r, t_k)]_i$ is the instantaneous number of j molecules located within the r^{th} bin for an arbitrary reference molecule i in P, and $V_{shell}(r) \approx 4\pi r^2 \Delta r$ is the shell volume of the bin. An accurate mean RDF is measured over $\mathcal{O}(10^6)$ MD time-steps ($\Delta t^*_{av} = 5000$).

The radial distribution function may also be measured in parallel, where the MD

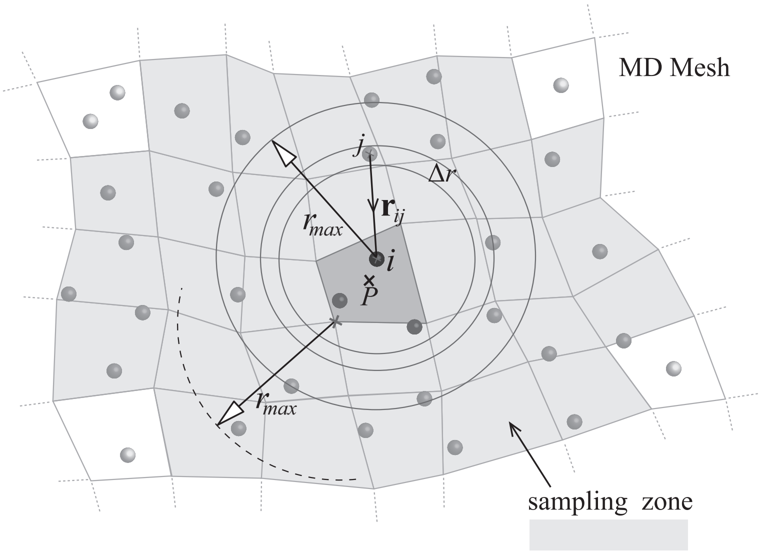


Figure 3.7: Schematic of the technique employed for measuring the radial distribution function from an arbitrary cell P.

mesh and the sampling zone are decomposed across processors. The procedure generally involves communicating all instantaneous positions of molecules located within the cell-zone of all processors to the single processor mesh which contains cell P. Examples of radial distributions sampled from a MD simulation are compared with literature correlations in Figure 4.20 (§4.5.5). The RDF may also be measured for mixtures.

3.7.2 Force, Potential Energy and Virial Distributions

Measurements of force, potential energy and virial distributions from fully-periodic MD simulations are presented. These distributions are required in non-periodic MD simulations, as described in §4.5.5. We consider two adjacent regions: a *sampling* zone (sz) and a *binning* zone (bz) (see Fig 3.8). For sampling a 1D distribution, both zones need to be cubic. The binning zone is divided into a series of bins, n_{bins}, and uses a technique identical to the 1D MOB. For each molecule, i in the binning zone, the contributions of force, potential energy and virial resulting from the other j molecules — solely occupying the sampling zone — are computed and appended to the bin in which i is located. The bin may be found in an identical manner as described in the MOB section.

The mean force, potential energy and virial in the r^{th} bin are computed using:

$$\langle f_{bi}(r)\rangle = \left[\frac{\displaystyle\sum_{k=1}^{S} \sum_{i=1}^{N_i^{bz}(r,t_k)} \sum_{j=1}^{N_j^{sz}} f(r_{ij}) \frac{\mathbf{r}_{ij}}{|\mathbf{r}_{ij}|}}{\displaystyle\sum_{k=1}^{S} N_i^{bz}} \right] \cdot \hat{\mathbf{n}}, \tag{3.23}$$

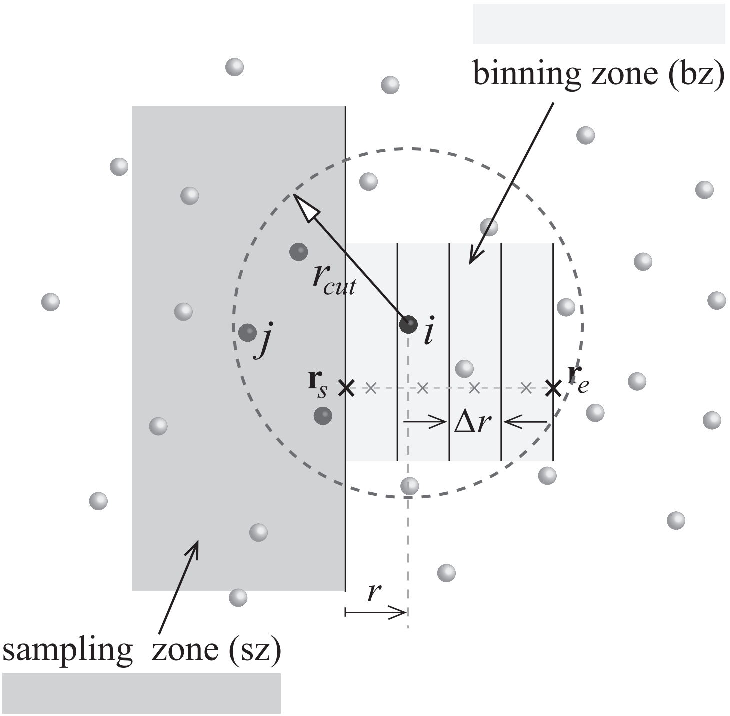


Figure 3.8: Schematic of the technique for sampling 1D force, potential energy and virial distributions for application at non-periodic boundaries of MD simulations. The same technique is used for sampling in higher dimensions.

$$\langle U_{bi}(r)\rangle = \frac{\displaystyle\sum_{k=1}^{S}\sum_{i=1}^{N_i^{bz}(r,t_k)}\sum_{j=1}^{N_j^{sz}}\frac{1}{2}U(r_{ij})}{\displaystyle\sum_{k=1}^{S}N_i^{bz}(r,t_k)}, \quad \langle w_{bi}(r)\rangle = \frac{\displaystyle\sum_{k=1}^{S}\sum_{i=1}^{N_i^{bz}(r,t_k)}\sum_{j=1}^{N_j^{sz}}\frac{1}{2}r_{ij}f(r_{ij})}{\displaystyle\sum_{k=1}^{S}N_i^{bz}(r,t_k)}, \tag{3.24}$$

where $N_i^{bz}(r,t_k)$ are the instantaneous molecules residing within the r^{th} bin, and N_j^{sz} are the j molecules located in the sampling zone, within r_{cut} from i. Note that $\hat{\mathbf{n}} = \mathbf{r}_e - \mathbf{r}_s/|\mathbf{r}_e - \mathbf{r}_s|$ is the normal vector, where $\mathbf{r}_s, \mathbf{r}_e$ are the start and end points of the distributions (see Fig 3.8). The mean distributions are averaged in a MD simulation run for $\mathcal{O}(10^6)$ time-steps ($\Delta t^*_{av} = 5000$). Examples of 1D distributions are shown in Figure 4.19 (§4.5.5).

The same method may be applied to measure mean distributions in higher dimensions, or for solid-liquid interactions. The latter requires that the sampling zone is occupied by solid-wall molecules of the desired lattice configuration, instead of a liquid. An example of solid-liquid distributions is given in Figure 5.6 (§5.1.2).

3.7.3 Potential Energy Landscapes

Measuring potential energy landscapes is sometimes required for verification purposes, for example, to study the complex topology that a particular fluid produces. In this research we are interested in computing energy landscapes to estimate the number of available sites for inserting additional molecules, for example, to match the target density within an arbitrary control cell P (§4.4.2). A target site is found in the molecular field that is equal to the average potential energy per molecule within the cell, $\langle U_P \rangle$. The site-searching algorithm uses a steepest-descent energy method, as described in Appendix B.1, until this criterion is met. Subsequently a molecule is inserted at the selected site.

A potential energy landscape is expensive to map and thus is not applied during an MD run. Rather, it is implemented as a post-processing utility, operating on molecules that have been written out to disk, and computing an *instantaneous* landscape for the required time-step. A potential energy landscape is measured in a sampling volume defined by the user and consisting of a sub-region of the total domain.

The algorithm first discretises the sampling zone uniformly into several 'nodes' (typically $\sim 30,000\sigma^{-3}$). At each node, i, a fictitious probe samples the potential energy $U_i = \frac{1}{2}\sum_{j=1}^{N_{mols}} U(r_{ij})$, where the sum is applied over all N_{mols} existing molecules within r_{cut} range. If at any point the node comes within a threshold radial-separation of a molecule ($\sim 0.01\sigma$), the potential energy need not be computed thus avoiding excessively large numbers due to the pair potential increasing to infinity at small values of radial separation r. In addition, we adopt a filtering scheme so as to view only the (quasi) iso-surfaces at the target potential energy $U_i = \langle U_P \rangle \pm U_{range}$, where $U_{range} \sim 0.05\epsilon$ is a small potential energy threshold. Examples of potential energy landscapes are shown in Figure 4.6 (§4.4.2).

3.7.4 Transport Coefficients

Transport coefficients may also be sampled from MD simulations, so that they can then be used in continuum hydrodynamic descriptions. For example, Delgado-Buscalioni and Coveney [5] show that their hybrid flux coupling scheme failed when the transport coefficients differed by more than 15% between MD and continuum formulations in the common coupling region. Okumura and Heyes [36] show that by inputting the MD-measured transport coefficients in the continuum governing equations, the same predictions are obtained when modelling the same unsteady-state cases using either MD or continuum hydrodynamics.

In the literature, two techniques for measuring transport properties in MD are commonly used. The first samples from an equilibrium (EMD) simulation in the microcanonical ensemble (constant number of molecules, volume and energy), that either makes use of: the Green-Kubo methods (that involve computing time-correlations from the system) or the Einstein methods (that involve computing displacements in properties over time). A different method applies perturbative non-equilibrium MD (NEMD) simulations, typically for steady-state systems, to compute the transport coefficients directly from the linear response of the system.

In this research we implement the Green-Kubo methods in OpenFOAM, since they are more widely used by researchers in the field. We present the appropriate Green-Kubo formulas for measuring the transport properties within an arbitrary sampling zone (Z) of the domain. Only the shear viscosity has been used in this thesis, as it is the property which is needed by our Navier-Stokes continuum formulation for coupling with MD in a hybrid simulation. The other transport coefficients have been implemented and described below to complete their description. The velocity autocorrelation function (VACF) is implemented to test our non-periodic boundary conditions and their effect on the bulk fluid dynamics of an equilibrium MD simulation (§5.1.1).

Instantaneous autocorrelation functions (ACFs) are computed over a time-segment $t \rightarrow t + \Delta t_{av}$, where $\Delta t^*_{av} \approx 2.0 - 4.0$. The ACFs are then calculated as a sum average of all instantaneous ACFs respectively. A large number of instantaneous ACFs ($\sim \mathcal{O}(10^4)$) are generally required to compute accurately the time-averaged ACFs as they suffer from a low signal-to-noise ratio.

Diffusion coefficient

The self-diffusion coefficient D_0 of a fluid at an equilibrium state point may be computed from

$$D_0 = \frac{1}{3} \int_{t=0}^{\infty} dt \langle \mathbf{v}_i(0) \cdot \mathbf{v}_i(t) \rangle, \tag{3.25}$$

where $\langle \mathbf{v}_i(0) \cdot \mathbf{v}_i(t) \rangle$ is the time-averaged VACF, which we describe next. In all time-correlation functions the integrals are evaluated numerically using Simpson's rule, and assuming the limits $t = [0, \Delta t_{av}]$. This is justifiable since the time-average correlation functions asymptotically approach zero very quickly.

Velocity Autocorrelation Function

The velocity autocorrelation function (VACF), $C_{vv}(t) = \langle \mathbf{v}_i(0) \cdot \mathbf{v}_i(t) \rangle$, is computed in a zone of an equilibrium MD system, and stored in a 1D histogram of size $\Delta t_{av}/\Delta t_s$, as described in the following procedure:

1. At the commencement of an averaging time period, which we call the relative time-origin, t_0, the current molecules (i) that occupy the zone (Z), and their instantaneous velocities ($\mathbf{v}_i(t_0)$), are stored in memory. The first entry in the histogram may be computed using

$$C_{vv}[0] = C_{vv}(t_0) = \frac{1}{N_Z(t_0)} \sum_{i=1}^{N_Z(t_0)} \mathbf{v}_i(t_0) \cdot \mathbf{v}_i(t_0). \tag{3.26}$$

2. At an arbitrary successive sampling time t_k, a new entry in the histogram is computed using:

$$C_{vv}[k] = C_{vv}(t_k) = \frac{1}{N_Z(t_k)} \sum_{i=1}^{N_Z(t_k)} \mathbf{v}_i(t_k) \cdot \mathbf{v}_i(t_0), \tag{3.27}$$

where k is the new bin index of the histogram $\{0 \leq k < (\Delta t_{av}/\Delta t_s)\}$. We loop over all existing molecules within the zone. Only if a molecule i is found in the list of stored molecules at $t = t_0$, is its scalar product $\mathbf{v}_i(t_k) \cdot \mathbf{v}_i(t_0)$ computed in the sum. Otherwise the molecule would have moved out of the measuring zone.

3. After all entries in the C_{vv} histogram are computed, that is at a time $t = t_0 + \Delta t_{av}$, the molecules and velocities that were stored at $t = t_0$ are discarded and the procedure starts afresh from step 1.

The procedure is repeated $\Delta t_w/\Delta t_{av}$ times, after which the average of all VACFs is determined and written out to disk or used to compute the diffusion coefficient (3.25). Computing the VACF in parallel is typically a daunting task, as molecules may travel across processors during the averaging time. Since a molecule i is completely discarded from the departing processor and a new molecule inserted on the receiving processor, the link between the old and the new molecule is destroyed for computing the scalar product.

Auto correlation functions for other variables that are required by the following transport coefficients use a similar procedure to the VACF.

Shear viscosity

The shear viscosity is calculated by:

$$\eta = \frac{V_Z}{k_B T_Z} \int_o^\infty dt \langle P_{\alpha\beta}(t) P_{\alpha\beta}(0) \rangle, \tag{3.28}$$

where V_Z is the volume of the zone, T_Z is the temperature measured in the zone, and $\langle P_{\alpha\beta}(t) P_{\alpha\beta}(0) \rangle$ is the correlation function for the off-diagonal part ($\alpha \neq \beta$) of the pressure tensor measured in the zone (subscript 'Z' is removed for simplicity). Example for $\alpha = x, \beta = y$:

$$P_{xy} = \frac{1}{V_Z} \sum_{i=1}^{N_Z} \left\{ m(v_x)_i (v_y)_i + \frac{1}{2} \sum_{j=1(\neq i)}^{N_{mols}} \frac{(r_x)_{ij}(r_y)_{ij}}{r_{ij}} f(r_{ij}) \right\}. \tag{3.29}$$

η is inherently fluctuating as a consequence of the noisy pressure measurement. It is therefore possible to improve the statistics by computing averages of $\langle P_{\alpha\beta}(t) P_{\alpha\beta}(0) \rangle$ over all off-diagonal parts.

To implement this correlation function, storing of all molecules' velocities at the time of origin t_0 is not needed, as was required in the VACF; only the pressure tensor of the measurement zone is necessary at t_0. Therefore, it is possible to run a simulation in parallel, when sampling η.

Bulk viscosity

The bulk viscosity is:

$$\zeta = \frac{V_Z}{k_B T_Z} \int_o^\infty dt \langle \delta P(t) \delta P(0) \rangle, \tag{3.30}$$

where

$$\delta P = -\langle P \rangle + \frac{1}{3V_Z} \sum_{i=1}^{N_Z} \left\{ m|\mathbf{v}_i|^2 + \frac{1}{2} \sum_{j=1(\neq i)}^{N_{mols}} r_{ij} f(r_{ij}) \right\}. \tag{3.31}$$

Longitudinal viscosity

The longitudinal viscosity is:

$$\frac{4}{3}\eta + \zeta = \frac{V_Z}{k_B T_Z} \int_o^\infty dt \langle \delta P_{\alpha\alpha}(t) \delta P_{\alpha\alpha}(0) \rangle, \tag{3.32}$$

where

$$\delta P_{xx} = -\langle P_{xx} \rangle + \frac{1}{V_Z} \sum_{i=1}^{N_Z} \left\{ m(v_x)_i^2 + \frac{1}{2} \sum_{j=1(\neq i)}^{N_{mols}} \frac{(r_x)_{ij}^2}{r_{ij}} f(r_{ij}) \right\}. \tag{3.33}$$

Thermal conductivity

The thermal conductivity is:

$$\kappa = \frac{V_Z}{k_B T_Z^2} \int_o^\infty dt \langle J_\alpha(t) J_\alpha(0) \rangle, \tag{3.34}$$

where

$$J_x = \frac{1}{2V_Z} \sum_{i=1}^{N_Z} \left\{ (v_x)_i \left(m_i |\mathbf{v}_i|^2 + \sum_{j=1(\neq i)}^{N_{mols}} U(r_{ij}) \right) + \sum_{j=1(\neq i)}^{N_{mols}} (\mathbf{v}_i \cdot \mathbf{r}_{ij}) \frac{(r_x)_{ij}}{r_{ij}} f(r_{ij}) \right\}. \tag{3.35}$$

3.8 Closure

We have presented a tool-box for carrying out measurements in MD simulations. In particular, we have described models (added to the architecture) for measuring hydrodynamic properties (density, velocity, temperature and pressure) in arbitrary cell-volumes, which may be sampled within zones or 1D, 2D and 3D fields of the MD domain. The 3D time-averaged fields are especially important for molecular-to-continuum coupling strategy of hybrid simulations in arbitrary geometries. In addition, we have described methods for sampling mass, momentum and energy fluxes at faces of the mesh. We have also presented other properties, such as correlation functions (RDF, VACF), effective distributions of forces, potential energy and virial, transport properties and potential energy landscapes. A general measurement time scheme has been implemented, because different properties require different sampling and averaging timings.

Interpreting measurements from MD should always be performed with strict vigilance. In our simulations, sampling is taken from the same system, rather than from different and uncorrelated realisations of the same system. Additionally, samples of a property are taken after short periods of time, for example, after every MD time-step, rather than over long uncorrelated time-intervals. The statistical error (or uncertainty) of a measured property will also depend on a number of factors, including the size of sampling space taken, number of samples, and the state-point of the fluid (ρ, T) [37, 38]. For example, fluxes at faces are inherently more fluctuating than volume measurements due to the sampling size, which is perhaps the reason why the latter have been widely preferred for coupling MD with continuum formulations. Simple test cases of small MD systems may be carried out to check the minimum number of samples (or time-averaging period) required to converge a time-averaged mean property with the desired variance. To further improve statistics in field measurements, we have implemented a very simple coarse-graining method: a measurement in a cell additionally includes sampling from neighbouring cells. This smoothing procedure is especially important in hybrid techniques, in which accurate data is required over the shortest possible time so that strong coupling is achieved.

The measurement tool-box has been designed so that researchers who use our code in the future may implement and test their own models. The process is very simple and fast, since other existing models serve as a template, and the code is generally reusable. As measuring properties is such an important aspect in MD, suggestions for future work include:

- Derive and implement new equations for measuring hydrodynamic properties (e.g., stress, and temperature) in highly non-equilibrium MD simulations (e.g., see [29, 39]).
- Extend current measurement models to polyatomic systems.
- Extend the current coarse-graining method for field properties to include weighting factors from neighbouring cells and from old-time states.
- Adopt a multi-mesh approach, in order to decouple problems with mesh dependency on characteristics such as resolution, accuracy and computational cost. For example, a mesh may be used solely by the AICA scheme, while another mesh is used solely for measurements. Cells on different meshes need to be linked to enable communication of data between meshes.
- Implement the thermodynamic field estimator [40, 41] for generating accurate and smooth fields using the notion of statistical inference.

Chapter 4

MD Boundary Conditions in Arbitrary Geometries

Are conditions at a boundary "boundary conditions"?
- Brenner and Ganesan [42].

4.1 Introduction

MD simulations on commonly accessible processing facilities, like personal computers, workstations or a cluster of connected processors, may only reasonably simulate systems of relatively small length-scales and short time-scales. The application of Periodic Boundary Conditions (PBCs) [21, 22] is therefore, an almost universal choice of MD boundary conditions, as it provides a method for simulating small but infinitely cyclic systems close to equilibrium. PBCs provide a conservative, efficient, and robust implementation for studies such as phase transitions, tabulation of equations of state and correlations of transport coefficients, to name a few. Despite these advantages, PBCs present some limitations:

1. *Isolated macroscopic molecules* — Simulation of large macromolecules (e.g., proteins, carbon nano-tubes and complex bio-molecules) embedded within a solvent of many molecules require that the domain of the system be relatively large so that the isolated molecule does not interact with the image of itself at any point during the run. This means large 'dead-spaces' at the extremities of the cubic domain, and a waste of computational cost is associated with solving the uninteresting solvent-solvent dynamics [43]. A spherical domain is usually preferred for these type of systems [44, 45], because it yields approximately 50% fewer molecules than in the cubic simulation box, and therefore takes only half the processing time.

2. *Complex geometries, non-equilibrium systems and hybrid methods* — PBCs are generally implemented as pairs of coupled boundaries of identical size and shape. Therefore, they may only be applied to simulate systems in simple (essentially symmetric)

geometries. Some examples include cubic, cuboid, parallelepiped and truncated octahedron domains. In other complex configurations, PBCs may not be applicable. Examples include MD systems with boundaries of different shapes and sizes; the application of inlet and outlet boundary conditions in non-zero flows through nano-channels (e.g., [2]); non-equilibrium systems exhibiting gradients of density and temperature across the domain (e.g., [46]); and hybrid methods that require boundary conditions prescribed from a continuum solution at an arbitrary coupling interface (e.g., [4, 8, 38, 47]).

In view of these limitations, we address in this chapter the development and implementation of non-periodic boundary conditions (NPBCs) in molecular dynamics simulations. In Section 4.2 we give a brief overview of the literature concerning boundary conditions in MD. In Section 4.3 we propose a method for applying general NPBCs in arbitrary geometries and in parallel. The set of boundary models that describe our NPBCs are presented in Sections 4.4 and 4.5.

4.2 Background

4.2.1 'Modified' Periodic Boundary Conditions

Several methods have been proposed in the literature to simulate equilibrium and non-equilibrium MD systems. The first set of techniques which we describe, are in fact not a boundary condition *per se*, but a form of 'internal condition', that generally applies perturbations on all domain molecules (modifying their equations of motion) in conjunction with conventional PBCs:

1. *Thermostats*, control temperature. Terms are added to the equations of motion of molecules, in order to include an extra degree of freedom that mimics the exchange of heat between real molecules and a fictitious thermal reservoir set at a requested temperature. Examples are the velocity-scaling, Nosé-Hoover, Langevin, Andersen and Berendsen thermostats [22].

2. *Barostats*, control pressure and density. In principle, changes in instantaneous pressure are performed by introducing an extra set of equations of motion for the simulation box, so as to allow contraction or expansion of the domain. Additional terms to the equations of motion of molecules in particular require the rescaling of molecular positions in order to keep molecules confined within the domain [22]. The issue with these types of barostats is that they may not be applied to localised regions of the computational domain, say at inlet or outlet boundaries, in order to apply changes in pressure across a nano-channel.

3. *Gravitational method* [48, 49], to drive Poiseuille-type flows through nano-channels. An additional constant external force of $\mathcal{O}(10^{12})$ magnitude greater than the gravi-

tational force of the earth is applied to the equations of motion of molecules to mimic the fluid falling under gravity. An issue with this technique is that the channel-centre velocity is an outcome of the simulation, rather than an input. Moreover, a tightly coupled thermostat must also be applied, so that the excess heat due to the force field is removed, consequently restricting the method to isothermal flows.

A second set of techniques proposed in the literature involves applying perturbations to molecules solely as they collide with a periodic boundary:

1. The *Lees and Edwards shear boundary conditions* [50] apply modifications to the periodic boundary conditions in one direction, to achieve a steady-state shear in a molecular system. Any molecule that crosses the modified PBC is wrapped around to the other side and translated (in the direction of shear) by a distance, the magnitude and direction of which are dependent on the target velocity at the boundary.

2. The *Reflective Particle Method* (RPM) [40] is a fictitious semi-permeable membrane applied to one periodic inlet of a channel, in which molecules that collide with it are reflected specularly by a probability p, or wrapped around the cyclic boundary (the outlet) by a probability of $1 - p$. With the number of molecules in the system kept constant, the result is a steady-state density and pressure gradient across the channel. Notably, the values of pressure and density at the boundaries are an outcome of the simulation rather than an input.

Although these techniques may be useful for extracting transport properties from systems far from equilibrium, they rely mostly on the application of periodic boundary conditions, hence their usefulness still seems limited.

4.2.2 Non-periodic boundary conditions

Various approaches have been considered in order to eliminate PBCs as part of the MD system. The simplest technique is to model the boundary as a specular wall, so that molecules remain confined within the domain. Despite its efficacy, the wall model introduces spurious (that is, unphysical) finite-size effects in the form of rapidly varying local density (and pressure) oscillations close to the boundary that decay to the bulk value a few molecular diameters away from the boundary. The oscillations increase in amplitude and spatial extent from the boundary for fluids at higher densities and lower temperatures [51]. The reasons for these inhomogeneities are partly due to disregarding interactions with 'missing' molecules beyond the boundary, and also due to the presence of the specular wall, which reverses instantaneously the molecules' normal momenta.

Ciccotti and Tenenbaum [52] propose a slight alteration to the specular wall model in order to simulate a stationary non-equilibrium temperature gradient across a partially-periodic MD system. Two stochastic thermal wall boundaries are applied in the non-periodic direction, each at different fixed temperatures. When an arbitrary molecule, i, collides with a boundary, its velocity is reset from a probability density proportional to $\hat{\mathbf{n}} \cdot \mathbf{p}_i f_T(\mathbf{p}_i)$, $\hat{\mathbf{n}} \cdot \mathbf{p}_i > 0$, where $\mathbf{p}_i$ is the molecule's momentum, $\hat{\mathbf{n}}$ is the boundary's unit vector (pointing into the domain), and $f_T(\mathbf{p}_i)$ is the Maxwellian distribution corresponding to the target wall temperature.

More advanced models have been proposed to circumvent the finite-size effects that occur next to reflective-type non-periodic boundaries. Berkowitz and McCammon [53] consider a spherical domain, comprising a many-body system, which is divided into three spherical non-overlapping regions: an inner reaction region modelling standard MD, an intermittent buffer region employing Langevin dynamics, and an outer reservoir region that keeps molecules frozen in space. The configuration of molecules within the outer reservoir is initially sampled from a fully-periodic larger MD or MC simulation. The fixed positions of molecules located externally are used explicitly to supply dynamic molecules within the reaction and buffer region with an estimate of the missing force field. The complete force on an arbitrary boundary molecule, i, is given by the sum of pair-interacting forces with *real* molecules, j, in addition to the pair-interacting forces with *fictitious* molecules[1], k:

$$\mathbf{f}_i \;=\; \mathbf{f}_i^{real} + \mathbf{f}_i^{fict} \;=\; \sum_{j=1(\neq i)}^{N_{mols}^{real}} f(r_{ij})\frac{\mathbf{r}_{ij}}{|\mathbf{r}_{ij}|} + \sum_{k=1(\neq i)}^{N_{mols}^{fict}} f(r_{ik})\frac{\mathbf{r}_{ik}}{|\mathbf{r}_{ik}|}, \tag{4.1}$$

where N_{mols}^{fict} are the number of fictitious molecules within interaction range (r_{cut}) of molecule i. The buffer region is then present to handle properly any thermal fluctuations at the boundary via the stochastic Langevin dynamics, which add stochastic and dissipative forces to the equations of motion. Furthermore, the entire configuration is continuously shifted to ensure that the main zone of interest, in which particular effects need to be analysed, is always centered in the middle of the domain. As a result of this shift, those molecules which subsequently occupy the reservoir region are kept fixed, while those molecules occupying the internal region are moved using MD.

A similar model, dubbed 'the deformable stochastic boundary conditions', was proposed by Brooks and Karplus [54]. However, in this boundary model the main difference is that the frozen molecules in the outer (reservoir) region are completely excluded from the simulation. Instead, an analytically-derived 'mean' force field is imposed on molecules within the buffer region in order to account for the external molecules being disregarded, as shown in Figure 4.1. The complete force on an arbitrary molecule, i, in proximity of the boundary is then the net force, $\mathbf{f}_i^{real}$, due to real molecules, in addition to a mean force component due to those missing molecules beyond the boundary, $\mathbf{f}_i^{miss}$:

$$\mathbf{f}_i = \mathbf{f}_i^{real} - \int_{V_i^{miss}} \rho g(|\mathbf{r}|, \rho, T)\nabla U(|\mathbf{r}|) dV, \tag{4.2}$$

---

[1] Periodic boundary conditions resemble this method substantially, with the exception that the interacting fictitious molecules are mirror-imaged from real molecules located at opposite ends of the MD domain.

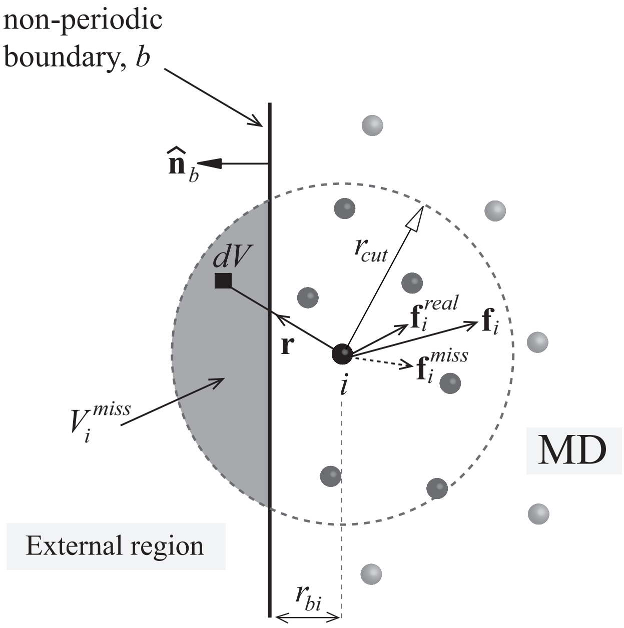


Figure 4.1: Schematic showing the total force, $\mathbf{f}_i$, on a boundary molecule, i, located a distance, $r_{bi} = \hat{\mathbf{n}}_b \cdot (\mathbf{r}_b - \mathbf{r}_i)$ from a non-periodic boundary, b. Superscripts '*miss*' represents the net force due to missing molecular interactions from the external volume V_i^{miss} and '*real*' represents the net force due to real molecules.

where $g(|\mathbf{r}|, \rho, T)$ is the radial distribution function, $U(|\mathbf{r}|)$ is the pair-wise intermolecular potential, and $\mathbf{r}$ is the vector between molecule i, located at $\mathbf{r}_i$ and an infinitesimally small element dV within the external missing volume of molecules V_i^{miss} (see Fig 4.1).

The main disadvantage of this method is that the integral in equation (4.2) may only be solved numerically, even for the simplest and most symmetric of geometries[2]. In the literature, 1D force distributions are derived normal to a boundary b, $f_{bi}(\mathbf{r}_{bi} \cdot \hat{\mathbf{n}}_b)$, where $\mathbf{r}_{bi} = \mathbf{r}_b - \mathbf{r}_i$ and $\hat{\mathbf{n}}_b$ is the normal unit vector of the boundary. For the planar boundary shown in Figure 4.1, the shape of the external volume $V_i^{miss}(r_{bi})$ is a 'spherical cap' for all $0 \leq \mathbf{r}_{bi} \leq r_{cut}$ positions. Consequently the force distribution is first integrated numerically, stored in a tabular form, and then imposed on a molecule i at every time-step of the simulation by the additive force $\mathbf{f}_i^{miss} = -f_{bi}(\mathbf{r}_{bi} \cdot \hat{\mathbf{n}}_b)\hat{\mathbf{n}}_b$, depending on its instantaneous distance from the boundary. However, this method is computationally cheaper than the boundary model of Berkowitz and McCammon [53], since only one extra force-computation is required per boundary molecule.

1D force distributions have been derived in Ref. [53] for a 2D and 3D spherical domain, and for a planar boundary, in which the other two directions are periodic boundary conditions. Force boundary models for the latter case were also derived by Werder *et al.* [38] and Attard [43]. Brünger *et al.* [55], extend the deformable stochastic force boundary

---

[2] Instead of numerical integration, the mean force may be measured directly from an equilibrium MD simulation at the target state-point, with $\mathcal{O}(10^6)$ MD time-steps of averaging (see §3.7.2).

model to simulate water molecules in a spherical domain. Other techniques have been proposed for complex solute-solute and solute-solvent interactions [56].

In hybrid MD-continuum methods for liquids, force boundary models have also been implemented at the continuum-molecular interface [4–6, 8, 47, 57]; however, these are prescribed using *ad hoc* diverging force distributions that still show density oscillations occurring next to the MD boundary, as reviewed in Ref. [38]. To rectify these fluctuations, Kotsalis *et al.* [51, 58] develop 1D controllers that impose modifiable forces in the direction of the spatial gradient of the density error.

4.2.3 Reservoirs and Control Volumes

A natural mechanism for constructing non-equilibrium ensembles in MD is through controlling (or constraining) thermodynamic and hydrodynamic properties within volumes or reservoirs at boundary regions of the MD domain. Sun and Ebner [2] consider a simple channel of finite length, and apply a source (reservoir) region on one side of the domain and a sink at the other. The reservoir region is maintained at a fixed density and temperature by means of a moveable piston that operates at regular intervals. When the density within the reservoir region decreases by a small amount, the piston compresses only those molecules within the reservoir region and resets to its original position, during which molecules in the bulk of the domain remain fixed in space. New molecules are inserted within the dilute region created by the compressive force of the piston in order to bring the density within the reservoir to the desired value. Inserted molecules are given a Maxwellian velocity at the desired temperature. On the other side of the channel, whenever a molecule enters the sink region it is instantaneously removed from the domain. In this method the authors do not account for the finite-size effects, nor do they control momentum.

Density may also be controlled in volumes of Grand-Canonical MD (GCMD) simulations using Monte-Carlo (MC) techniques. The method involves controlling the desired chemical potential and temperature within two independent control volumes [59], for example, to study colour diffusion (same species) and binary diffusion [46].

4.2.4 Continuum-to-MD hybrid coupling

Non-periodic boundary conditions in MD have also been the pinnacle of hybrid MD-continuum methods in the literature, in which proposed techniques impose state or flux coupling between continuum and molecular formulations. The development of one-direction continuum-*to*-MD boundary conditions in a hybrid formulation is also a central concern in pure MD simulations. These continuum-based boundary conditions are clearly very challenging as the problem consists of translating a few continuum quantities into molecular variables encompassing many degrees of freedom. We describe methods of this type in more detail in §6.2.

4.3 General implementation of MD Boundary Conditions at arbitrary boundaries

We introduce the following terminologies to best describe our notion of 'general' MD boundary conditions.

A "*boundary*" is a geometrical entity of finite area and infinitesimally small thickness, located at the extremities of the MD domain. In our method, the boundary is discretised by a set of unique faces on the domain mesh. The sum of the boundaries, therefore, define the total external boundary of the domain.

A "*boundary region*" may be considered as a region which is in proximity to its assigned boundary, that consists of a layer or 'halo' of cells internal to the domain. The thickness of this layer is arbitrary.

At the boundary and boundary-region a collection of "*boundary models*" are applied, using a superimposition technique, that together constitute a "*boundary condition*". In our proposed methodology, a boundary model may be classified as one of two types:

Type I Properties of molecules are altered upon instantaneous collision with the boundary.

Type II External perturbations are applied to molecules that occupy the boundary region at specific time-intervals.

The superimposition method which we propose ensures that: (1) a particular model is coded only once, and (2) the model may be used in many MD simulations with reasonable flexibility. The latter stipulates a 'building block' methodology, in which a user may combine different boundary models to collectively give one new MD boundary condition. Obviously, the choice of boundary models will depend on the type of case being simulated.

All our boundary models are implemented using a 'local' method, that is, in distinct cells, or at boundary faces, of the MD mesh. In this regard, we may access the well-established mesh characteristics distributed with the OpenFOAM software. For example, a mesh consists of a group of cells, and in turn each cell consists of a set of faces. Additionally, a 'cell-zone' is a group of cells on the mesh, while a 'face-zone' is a group of faces on the mesh. In Appendix C, we describe our well-developed set of versatile utilities for building zones on a mesh in a flexible and quick manner.

Important benefits of using control in individual cells or at individual boundary faces are described now, and compared with methods in the literature. In our method, control zones may be defined by 3D arbitrary geometries, since the MD domain mesh is generally constructed of polyhedral cells, of different shapes and sizes. In the literature, control in 3D arbitrary regions is however largely prohibited because of the way *global* Cartesian co-ordinates are used for applying *local* perturbations on molecules. For example, consider a molecule i located in a simple cubic MD domain of x_i co-ordinate. Control is achieved if the condition $x_{min} \leq x_i \leq x_{max}$ is satisfied [47]. Any attempt to extend this method to complex geometrical control regions not only represents a formidable task, but must be repeated for different control zone geometries. In addition, the literature method is non-optimal as the algorithm would probably loop over all molecules within the system to test the control-criteria.

Control in individual cells (or at faces) of the mesh makes our new technique more general, flexible and computationally efficient. The latter is achieved by directly applying control perturbations to molecules residing within an arbitrary control cell, thus avoiding expensive loops over molecules in the entire domain. This is facilitated by the 'cell-occupancy' data structure (§2.2.2), in which each cell on the MD mesh stores an inexpensive link to occupant molecules.

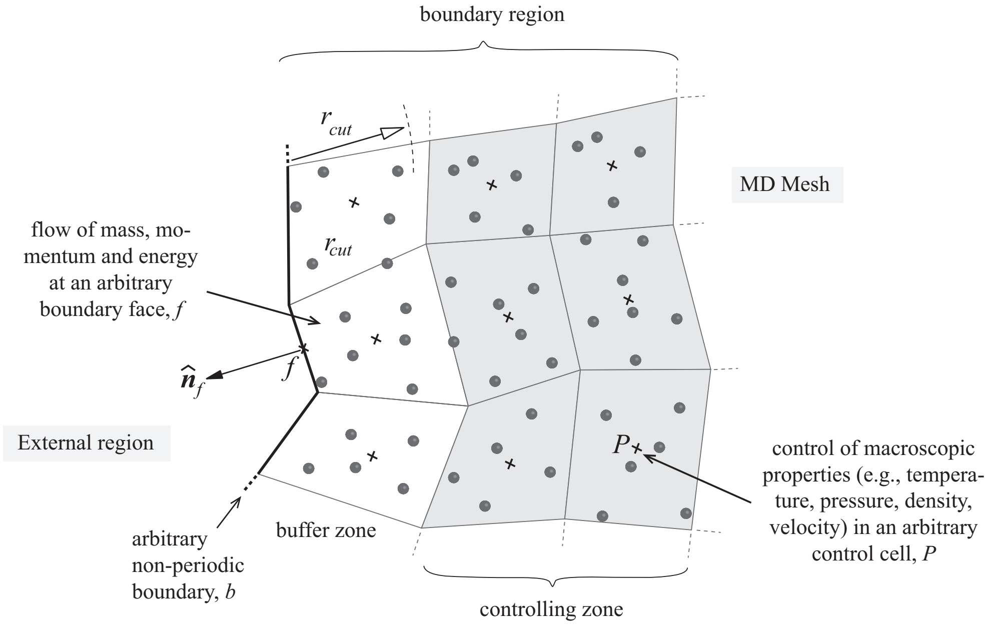


Figure 4.2: Schematic showing the implementation of our non-periodic boundary conditions, which impose flow of molecular properties in a buffer region next to an arbitrary boundary b, in addition to control of macroscopic (continuum) properties in the adjacent controlling zone.

A second benefit is that control of spatially-varying properties may be imposed, because each cell may be assigned a target value independent of other neighbouring cells in the controlling zone. Third, the method is tractable for parallel-processing. OpenFOAM decomposes the main mesh into sub-meshes on different processors [28]. Consequently, the zones too get divided uniquely on their respective sub-meshes, and control procedures are performed in tandem within smaller control zones. This is an important advantage of our technique. Sometimes it is unclear from published papers whether the proposed methods have been coded for parallel processing, an issue which is of utmost importance in computationally-intensive MD simulations. All of our models for use within the NPBCs of MD simulations have been parallelised. Finally, these NPBCs may also be applied to hybrid MD-continuum simulations in arbitrary geometries, entailing complex coupling regions with spatially varying properties, and in parallel (see Chapter 6).

The implementation of our non-periodic boundary conditions in arbitrary geometries consists of a *buffer* region of thickness $\sim r_{cut}$ adjacent to a non-periodic boundary b, in which flow of mass, momentum, and energy are prescribed, together with a neighbouring *controlling* region, in which thermodynamic and hydrodynamic continuum properties are constrained towards target values (see Fig 4.2). The control region typically consists of two or more layers of cells, so that spatial gradients in the normal direction of the boundary are also imposed. A property may only be controlled properly if it does not fluctuate with amplitude greater than the size of the control cell. The aim of the buffer is therefore to provide sufficient space for the boundary finite-size effects to settle completely.

In the rest of this chapter we describe the boundary models of Type I and Type II for multi-species monatomic Lennard-Jones fluids, which we use in our NPBC description. It suffices to describe models within an arbitrary control cell or at an arbitrary boundary face, since it is straightforward to extend the method to all cells and faces of the control zone. In Chapter 5 we demonstrate full-MD simulations using the NPBC description (e.g., at inlets and outlets of nano-channels), while in Chapter 6 we use the NPBC description as the entity which receives and sets continuum properties (by a continuum solver) in a direct hybrid scheme.

4.4 State Controllers

The ability to control fluid flows in nano-scale MD simulations is an essential requirement for many engineering applications. In this section, we present a new parallelised and general controller algorithm for steering an arbitrary geometric 3D region of a molecular dynamics simulation towards a desired thermodynamic and hydrodynamic state.

We describe the basic control methodology, in which a static MD mesh is used and a controlling region defined as groups of cells. The controlling zone may be applied solely at a boundary region, as shown in Figure 4.2, or generally anywhere within the internal part of the MD domain. Our method avoids using moving pistons or techniques involving expansion/contraction of the domain as typically used in the literature. Instead, the controller uses a feedback loop algorithm, in which a measured property within an arbitrary control cell is compared with the target value, and the error serves as the action for control on molecules.

A flexible time-scheme is proposed on a per-controller basis, so that an arbitrary continuum property may be controlled at frequencies appropriate to the time-variations of the target property dictated by the problem, and measured over longer time-scales to increase its statistical accuracy when supplied in the feedback loop. We describe specific control models for density, velocity, temperature and pressure. Test cases that apply control at boundaries of MD simulations are then investigated in Chapter 5.

4.4.1 Controller methodology

Fields of *target* continuum properties are prescribed to the state controller (see Fig 4.3). Each entry within a field corresponds to an MD cell in the controlling zone: an arbitrary control cell P therefore 'knows' the required cell-centred density ρ_P^{req}, velocity $\mathbf{u}_P^{\text{req}}$ and temperature T_P^{req}, at any time t of the simulation[3]. Throughout the MD run, the fields may either be uniform, or altered by an external function (temporally and/or spatially), or directly modified by the continuum solution in an overlapping region of a hybrid simulation. In the latter instance, the overlaying coupling region is identical to the MD controlling zone (but on a different, CFD mesh): an arbitrary finite-volume CFD cell P' corresponds to the MD control cell P in shape, size, and global co-ordinate.

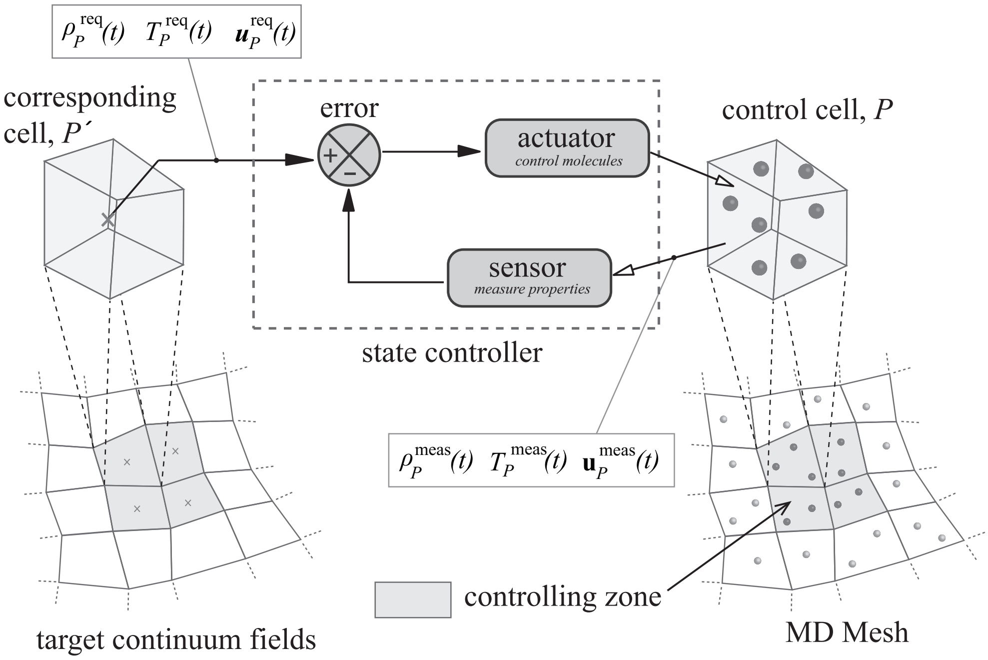


Figure 4.3: Schematic showing the operation of a generic controller using a cell-based approach.

Control functions

A simple closed-loop (negative-feedback) control system is implemented. The three functions of the control system are defined as follows (see Fig 4.3):

1. *Sensor* — measure the macroscopic property q_P^{meas} ($= \{\rho_P^{\text{meas}}, \mathbf{u}_P^{\text{meas}}, \text{or } T_P^{\text{meas}}\}$) from a zone cell P, using a bin-averaging technique.

2. *Error* — compute the difference between the target property $q_{P'}$ in cell P', which we write as q_P^{req} ($= \{\rho_P^{\text{req}}, \mathbf{u}_P^{\text{req}}, \text{or } T_P^{\text{req}}\}$) and the measured property in its corresponding

---

[3] In our method, pressure is controlled indirectly by a target density and/or target temperature, as detailed in §4.4.6.

cell P, generally given by:

$$\Delta q_P = K_P(q_P^{\text{req}} - q_P^{\text{meas}}), \tag{4.3}$$

where K_P is the proportional constant of the controller. Then Δq_P is converted into molecular properties, such as number of molecules extra/missing within the cell, forces, momentum, velocity or energy, so it is best suited for the actuation function that follows.

3. *Actuator* — impose the necessary control operations on molecular variables occupying P, based on the error signal.

Time scheme

We devise a flexible decoupled time-scheme for measurement and control of an arbitrary continuum property within its controller architecture. The scheme is decoupled because control and measurement of the macroscopic property may occur at independent frequencies and over different time-scales.

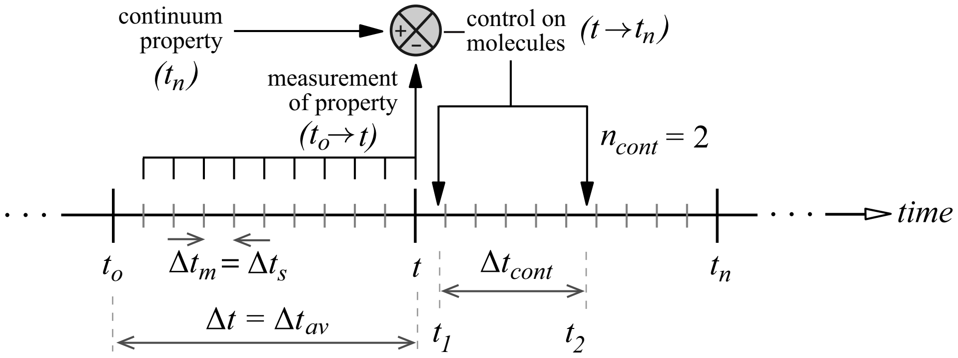


Figure 4.4: Schematic showing part of a time-line for an MD simulation with control enabled. Processes of measurement and control are decoupled, but linked by a common time-interval Δt.

Time-scales relating to *measurement* have been discussed in §3.3, and are used by the controller to time-average q_P^{meas} accurately.

Control is applied at well-defined time-intervals, Δt_{cont}. Between controlling steps, no perturbations are imposed within the control cells, so as to allow local equilibration of the fluid after being exposed to the controller's actions, and also to manage the computational cost. The frequency of control, and hence the strength of coupling the measured property with the target, is therefore dependent on the value chosen for Δt_{cont}.

Control and measurement processes are effectively *linked* by a common time interval $\Delta t = n_{av}\Delta t_{av} = n_{cont}\Delta t_{cont}$, where n_{av} and n_{cont} are the number of averages and number of control steps respectively that are performed within the time interval. Practically, $n_{av} = 1$ so that $\Delta t = \Delta t_{av}$ is the maximum possible averaging time-interval to reduce the statistical error of the measured property [37]. Therefore, a property measurement carried out during the previous time-interval $t - \Delta t \rightarrow t$, is used to compute the necessary controlling actions to take place over the next time period $t \rightarrow t + \Delta t$.

A key benefit of our proposed time scheme is that modification to Δt_{cont} is permitted in order to change the rate of control, without restricting Δt_{av} and hence affecting the accuracy of any measurement. We highlight this as an essential requirement in the controller methodology: poor sampling of q_P^{meas} due to a small Δt_{av} will feed a noisy error signal (Δq_P) to the actuator, that may result in an unstable diverging state. In contrast, too large a Δt_{av} limits any time-variations of target properties, $q_P^{\mathrm{req}}(t)$, within the cell.

The most challenging part of a decoupled time scheme is the formulation of a model that performs control operations on molecules over a series of n_{cont} control time-steps, as illustrated in Figure 4.4.

The controllersDict input file

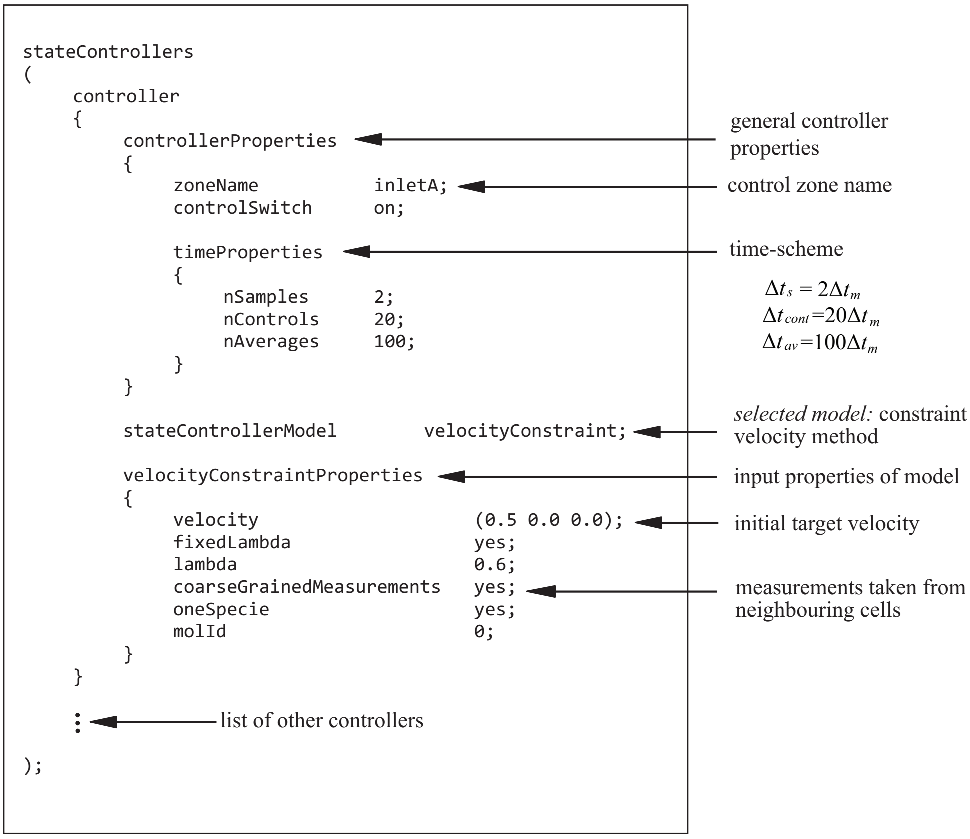
Any number of controllers may be applied to an MD simulation in the present OpenFOAM implementation by specifying them in a list format, within an input file called controllersDict (see Fig 4.5). A controller is specified by spatial and time properties, which are central to the general control architecture, and properties associated with the selected control model. Properties may be tweaked at any point in time during the simulation run, in order for example to switch *off/on* the controlling actions, or to attain the required time-response, accuracy or computational efficiency.

Models

In the following sections, we present models for controlling state-properties of density, velocity, temperature, and pressure for single or multi species mono-atomic MD fluids. Each controller model inherits the general spatial and temporal properties of the generic controller described previously. The current control models are extendable to polyatomic fluids.

4.4.2 Density control

A popular method for controlling 'global' density in the literature is a scheme similar to that for pressure control [22]. A change in the size of the simulation domain is applied and subsequent shifting of the molecules is performed so that they remain confined within the domain box (for further details see §4.4.6). A less common but localised method is the application of control to inlets of MD nano-channels using fictitious 'pistons' (forces applied to molecules) and 'reservoirs' [2] that are continually filled with molecules from a dilute region at the edge of the domain to match the density within the reservoir (see §4.2.3). The latter have also been implemented by most hybrid coupling strategies; however, mass is inserted/deleted to control mass flux at the boundary of the MD domain instead of



```
stateControllers
(
    controller
    {
        controllerProperties
        {
            zoneName            inletA;
            controlSwitch       on;

            timeProperties
            {
                nSamples        2;
                nControls       20;
                nAverages       100;
            }
        }

        stateControllerModel        velocityConstraint;

        velocityConstraintProperties
        {
            velocity                        (0.5 0.0 0.0);
            fixedLambda                     yes;
            lambda                          0.6;
            coarseGrainedMeasurements       yes;
            oneSpecie                       yes;
            molId                           0;
        }
    }

    ⋮

);
```

Figure 4.5: The controllerDict input file. We show an example of a velocity controller applied to an arbitrary inlet boundary of an MD simulation. The first part of the controller sub-dictionary represents the general properties, which are common to all controllers, e.g., the *on/off* control switch, the name of the controlling zone, and the general time-scheme for measurement and control. In the second part of the controller description the specific model is selected. In this case, we choose the model 'velocityConstraint'. Input properties (e.g., target velocity in this case) depend on the requirements of the model.

controlling the boundary density. Recently, a gradient-based controller was proposed for smoothing out density fluctuations that appear next to non-periodic boundaries using one-dimensional bins [51, 58].

We propose a new generalised density controller that works by introducing or deleting molecules in arbitrary (and individual) 3D control-cells of the mesh in order to match the cell target density. Consequently, the method like many constraint methods in the literature does not obey conservation principles after control steps. However, the formulation ensures that the cell- and time-averaged mass, momentum and energy are controlled to fluctuate about mean values. Parallelisation of our density controller is described in more detail in Appendix B.2. In Sections 4.4.6 and 4.5.4 we extend this method to control pressure and mass fluxes at boundaries.

Method

To control number density in an arbitrary control-cell P (cf Figures 4.3 and 4.4):

1. At a time t, compute the density error in P and translate this into a property representing whole "molecules" using the following formula:

$$\Delta N_P(t \to t_n) = \text{NINT}\left(K_P\left[\rho_P^{\text{req}}(t_n) - \rho_P^{\text{meas}}(t)\right] V_P\right), \tag{4.4}$$

   where $\rho_P^{\text{meas}}(t)$ is the measured cell density during the previous time period $(t_o \to t)$, $\rho_P^{\text{req}}(t_n)$ is the required cell density at a later time $(t_n = t + \Delta t)$, and $\Delta N_P(t \to t_n)$ is the total number of molecules to control from cell P during the next Δt. The NINT(x) function is required to apply the nearest integer, since only whole molecules may be introduced $(\Delta N_P > 0)$ or removed $(\Delta N_P < 0)$. $K_P(= 1)$ is the proportional gain that controls the response rate of the density controller. If $\rho_P^{\text{req}}(t_n)$ is not known at a time t_n, it may be extrapolated from old-time quantities: $\rho_P^{\text{req}}(t_n) = 2\rho_P^{\text{req}}(t) - \rho_P^{\text{req}}(t_o)$.

2. The total number of molecules to insert within P, that is, ΔN_P in equation (4.4) is divided equally across the subsequent n_{cont} control steps:

$$\Delta N_P(t_k) = \begin{cases} \text{ceil}\left(\Delta N_P/n_{cont}\right), & \text{if } \Delta N_P > 0, \\ \\ \text{floor}\left(\Delta N_P/n_{cont}\right), & \text{if } \Delta N_P < 0, \end{cases} \tag{4.5}$$

   where $t \leq t_k < t_n$, $\{k = 1, 2, \ldots, n_{cont}\}$, followed by a residual-correction to $\Delta N_P(t_k)$ at each control step so that the prescribed density is matched accurately during $(t \to t_n)$.

3. At an arbitrary control step, t_k, two types of mass residuals are corrected. The number of molecules that *fail* to be inserted or deleted in a previous control step, $\delta N_P^{fail}(t_{k-1})$, are added to $\Delta N_P(t_k)$. Then the cumulative error resulting from the

ceiling/flooring functions in equation (4.5) is checked:
If $|\Delta N_P(t_k) + \Delta N_P^{success}| > |\Delta N_P|$,
Then modify $\Delta N_P(t_k)$ to:

$$\Delta N_P(t_k) = \Delta N_P(t \rightarrow t_n) - \Delta N_P^{success}, \tag{4.6}$$

where $\Delta N_P^{success}$ is the cumulative number of molecules controlled within previous control steps. Note that both terms ΔN_P^{fail} and $\Delta N_P^{success}$ are positive or negative depending on whether $\Delta N_P > 0$ or $\Delta N_P < 0$, respectively. Furthermore, $\Delta N_P^{success}$ is reset to zero after the whole time-interval $(t \rightarrow t_n)$ is over while ΔN_P^{fail} is set to zero after every control step.

4. Insertion and deletion processes are performed immediately after the intermolecular force calculation step of the Leapfrog algorithm and are described separately below.

Inserting molecules

If $\Delta N_P > 0$, the actuation function inserts these whole molecules sequentially into the cell P. During this process, existing molecules that currently reside within the domain, remain fixed in space and time. The procedure for inserting one molecule, i, is described.

1. Search for a correct insertion site, $\mathbf{r}_i$, that ensures non-overlapping molecules and conserves the cell-average potential energy. We employ the USHER algorithm [60], which performs a steepest-descent iterative search in the potential energy landscape. Figure 4.6 shows examples of the potential energy landscapes for two different fluid states. The site-searching algorithm is described in more detail in Appendix B.1. A site to insert a molecule is accepted if its potential energy is equal to the average potential energy per molecule within the cell:

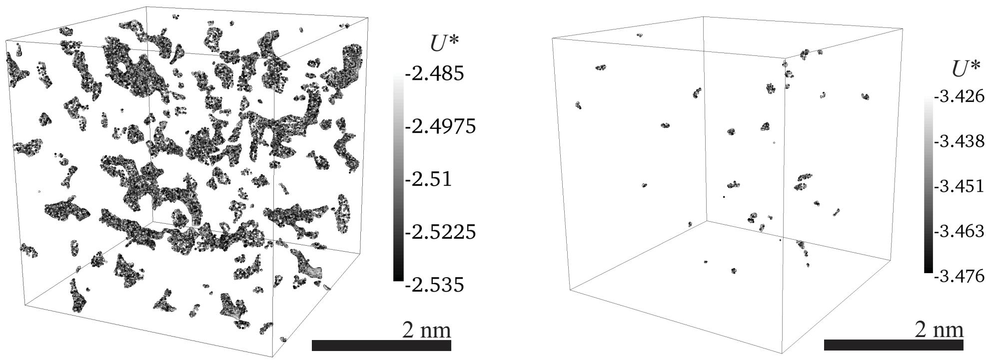
$$U_P^{\text{req}} = \langle U_i \rangle_P = \frac{\displaystyle\sum_{k=1}^{S} \sum_{i=1}^{N_P(t_k)} \frac{1}{2} \sum_{j=1(\neq i)}^{N_{mols}} U(r_{ij})}{\displaystyle\sum_{k=1}^{S} N_P(t_k)}, \tag{4.7}$$

where $N_P(t_k)$ is the number of molecules residing in P, at an instantaneous time t_k. Values for U_P^{req} for various liquid state points are plotted in Figure 4.7(a). We also show in Figure 4.7(b) the relationship between the number of available insertions sites per unit volume and the fluid state point.

2. Create a molecule at $\mathbf{r}_i$, and update the acceleration and potential energy of the surrounding molecules j, within interaction range:

$$\mathbf{a}_j^n = \mathbf{a}_j^o - \frac{f(r_{ij})\mathbf{r}_{ij}}{m_j|\mathbf{r}_{ij}|}, \qquad U_j^n = U_j^o + \frac{U(r_{ij})}{2}. \tag{4.8}$$

3. The initial velocity, $\mathbf{v}_i$, of the newly-inserted molecule is sampled randomly from a Maxwell-Boltzmann distribution at the required temperature, T_P^{req} and mean velocity, $\mathbf{u}_P^{\mathrm{req}}$. Conservation of momentum and kinetic energy within cell P is performed after insertion by rescaling velocities of all other molecules.



(a) Low Density

(b) High Density

Figure 4.6: Snapshots of potential energy landscapes for (a) a low density fluid at ($\rho^* = 0.6, T^* = 2.4, \langle U^* \rangle = -2.51$) and (b) a high density fluid at lower temperatures ($\rho^* = 0.8, T^* = 1.8, \langle U^* \rangle = -3.451$). The (quasi) iso-surfaces indicate locations in space (within a range of $\langle U^* \rangle \pm 0.025$) where new molecules may be inserted, for the instantaneous configurations shown. For lower density fluids, site-searching is more efficient and also more successful, due to the large number of target potential energy sites available (see also Fig 4.7). The method we employ for sampling potential energy landscapes is described in §3.7.3 .

Deleting molecules

If $\Delta N_P < 0$, molecules may be deleted sequentially using the reverse of the insertion process. Deletion is more computationally efficient and also more successful than insertion. Existing molecules within the cell are held fixed in time and space as candidate molecules are deleted. The process for deleting one molecule is described:

1. Initially select a candidate molecule i from a cell P, using a criterion that maintains the same potential energy in the cell. We use a scheme that loops over all molecules within cell P and chooses the molecule with its potential energy closest to U_P^{req}.

2. Update the accelerations and potential energies of surrounding molecules to account for the molecule being deleted, similar to Eq. (4.8):

$$\mathbf{a}_j^n = \mathbf{a}_j^o + \frac{f(r_{ij})\mathbf{r}_{ij}}{|m_j \mathbf{r}_{ij}|}, \qquad U_j^n = U_j^o - \frac{U(r_{ij})}{2}. \tag{4.9}$$

3. Delete molecule i from cell P.

Alternative criteria may be adopted for selecting candidate molecules to be deleted, for example, by choosing a molecule at random or choosing one that has a velocity closest to

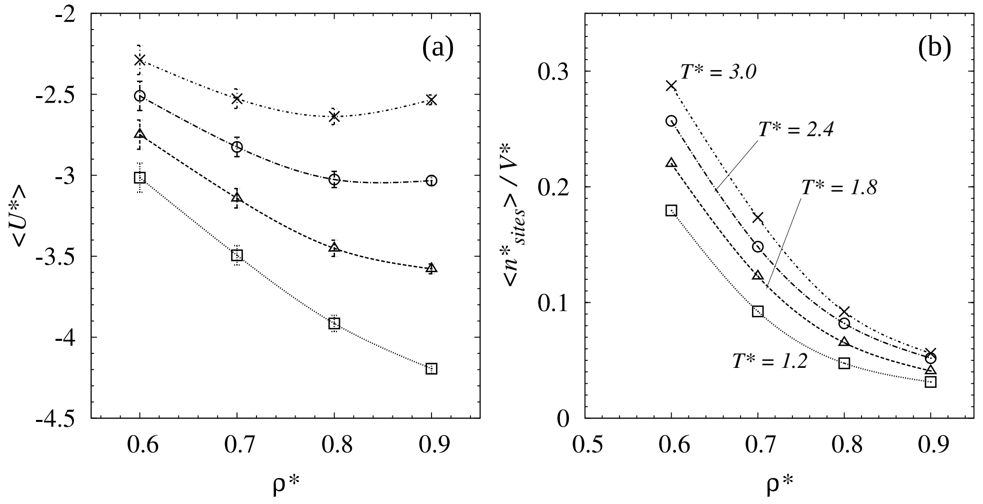


Figure 4.7: Graphs showing (a) the average potential energy per molecule, $\langle U^* \rangle$, within an MD system for various state-points; $\rho^* = \{0.6, 0.7, 0.8, 0.9\}$; $T^* = \{1.2, 1.8, 2.4, 3.0\}$, and (b) the average number of sites per unit volume, $\langle n^*_{sites} \rangle / V^*$, which satisfies the criteria $U^*_i = \langle U^* \rangle \pm 0.025$, that are available for inserting molecules. The latter have been sampled from MD simulations for the same range of state-points, using the potential energy landscapes. See also caption of Fig 4.6.

the cell-averaged velocity. Molecules deleted close to the cell-averaged potential energy is the preferred criterion, since it reduces the effect of temperature drifts within the local field.

Multi-species control

Our density controller is designed to operate on both single- and multi-species fluid systems. For multi-species fluids, the setup consists of applying a density controller per specie in a common controlling zone, and providing the controllers with the target partial-densities within the control cells.

Parallelisation

Our density controller has also been formulated to operate in parallel, where the control-cells are distributed on domain-decomposed processors. We describe this in more detail in Appendix B.2.

In brief, control cells that are internal to the mesh insert/delete molecules sequentially at first, since no parallel-communication is required. Insertions/deletions of molecules in control cells at the boundary then proceed in turn. Prior to insertion/deletion, processors communicate in order to check for ‘conflicts’. A conflict occurs if any two or more designated molecules are within r_{cut} range. The conflict is resolved by arbitrarily assigning a priority to each processor. The processor with the highest priority proceeds with the insertion/deletion of the molecule, while the other conflicting processors of lower priority terminate their tries, and re-attempt them immediately thereafter. After insertion/deletion, the molecule is 'copied'/'highlighted' to its set of corresponding referred cells, which for processor boundaries involves communication across processors. The step which updates properties of surrounding molecules due to the copied/highligthed molecule then proceeds in serial on all processors.

Guidelines for choosing the density controller parameters

The properties for site-searching may be selected as described in Appendix B.1 and in Ref. [60]. For fluids with high densities and low temperatures, fewer sites are available for inserting molecules, as shown in Figures 4.6 and 4.7. Solutions to this problem include choosing a small controlling interval $\Delta t^*_{cont} \sim 0.125$, or increasing the number of site-searching iterations (see Appendix B.1) in order to extend the probability of finding the sites, albeit at higher computational expense. Alternatively, one may widen the threshold range of the potential energy criterion to increase the number of potential sites, in addition to using a tightly coupled thermostat to remove the excess heat added to the cell. The density controller may not work as intended in the limit of the density approaching that of the solid phase.

The control-cell size, ΔX_P, and averaging time-interval, Δt_{av}, determine the statistical errors in density measurements and the resolution (spatial and temporal) of density control. This is alleviated by using coarse-graining of density measurements from neighbouring cells, as described in §3.5.2, so that $\Delta t^*_{av}(\sim 0.5)$ may be short and $\Delta X^*_P(\sim 4)$ is small.

A matching error between target and measured cell-density is introduced due to the nearest integer function in equation (4.4), given by $\mathrm{NINT}(x) = 0$, if $-0.5 < x < 0.5$:

$$\zeta_{error} = \frac{|\rho_P^{\mathrm{req}} - \rho_P^{\mathrm{meas}}|}{\rho_P^{\mathrm{req}}} \approx \frac{0.5}{\Delta X_P^3 \rho_P^{\mathrm{req}}}, \tag{4.10}$$

where ζ_{error} is the fractional error (see Fig 4.8).

In Figure 4.9 we show an example of an equilibrium MD simulation with periodic boundary conditions applied in all directions. Density is controlled distinctly in cells of the mesh.

4.4.3 Velocity control

In the literature, a few techniques have been proposed for controlling velocity (or momentum), e.g.,[4, 47]. Whereas most models apply velocity control to molecules every MD time-step, instead we propose allowing a time-period of equilibration between one control step and another. We generalise the control model for velocity so that the user may choose a light or tight coupling to continuum-velocity, depending on the case being simulated.

The convective velocity is controlled within an arbitrary control cell P using the fol-

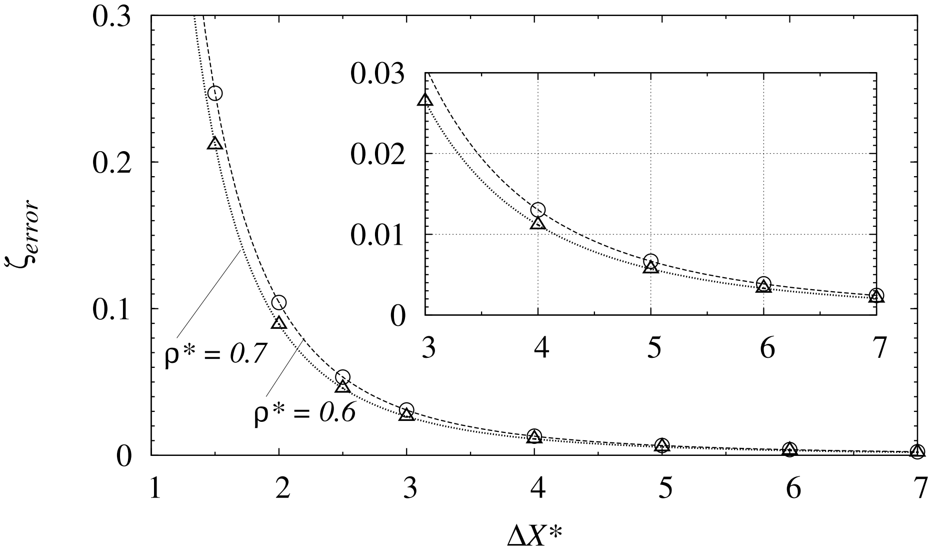


Figure 4.8: Estimate of the fractional error, ζ_{error}, between target and measured cell-density varying with cell-size, ΔX^* in reduced units, due to using the NINT(x) function in equation (4.4).

lowing procedure:

1. At a time t, compute the velocity error that is to be imposed per control step:

$$\Delta \mathbf{u}_P(t_k) = \frac{\Delta \mathbf{u}_P(t \to t_n)}{n_{cont}} = \frac{\lambda}{n_{cont}} \left(\mathbf{u}_P^{\mathrm{req}}(t_n) - \mathbf{u}_P^{\mathrm{meas}}(t) \right), \tag{4.11}$$

   where $\mathbf{u}_P^{\mathrm{meas}}(t)$ is the measured cell velocity at current time t, $\mathbf{u}_P^{\mathrm{req}}(t_n)$ is the required cell velocity at a later time ($t_n = t + \Delta t$), and λ / n_{cont} ($= K_P$) is a modifiable parameter that defines the rate at which the velocities of molecules occupying cell P are accelerated towards $\mathbf{u}_P^{\mathrm{req}}(t_n)$. If the term $\mathbf{u}_P^{\mathrm{req}}(t_n)$ is unknown at a time t_n it too may be extrapolated from known quantities: $\mathbf{u}_P^{\mathrm{req}}(t_n) = 2\mathbf{u}_P^{\mathrm{req}}(t) - \mathbf{u}_P^{\mathrm{req}}(t_o)$.

2. At each control step, add an external force to the equations of motion of *all* molecules occupying cell P, after the intermolecular force calculation step. For an arbitrary molecule i the force is:

$$\mathbf{f}_i^{ext} = \frac{\Delta \mathbf{u}_P(t_k)}{\Delta t_m} m_i. \tag{4.12}$$

Guidelines for choosing the velocity controller parameters

Similar to the density controller, the choice of ΔX_P and Δt_{av} introduces a trade-off between resolution (time and space), and statistical accuracy. The introduction of spatial coarse-graining for measuring velocity within a particular cell reduces the statistical scatter, so that a shorter Δt_{av} and smaller ΔX_P may be employed. For highly transient flows, $\Delta t_{av} \ll t_{flow}$, where t_{flow} is the shortest characteristic time of the target velocity $\mathbf{u}_P^{\mathrm{req}}(t)$.

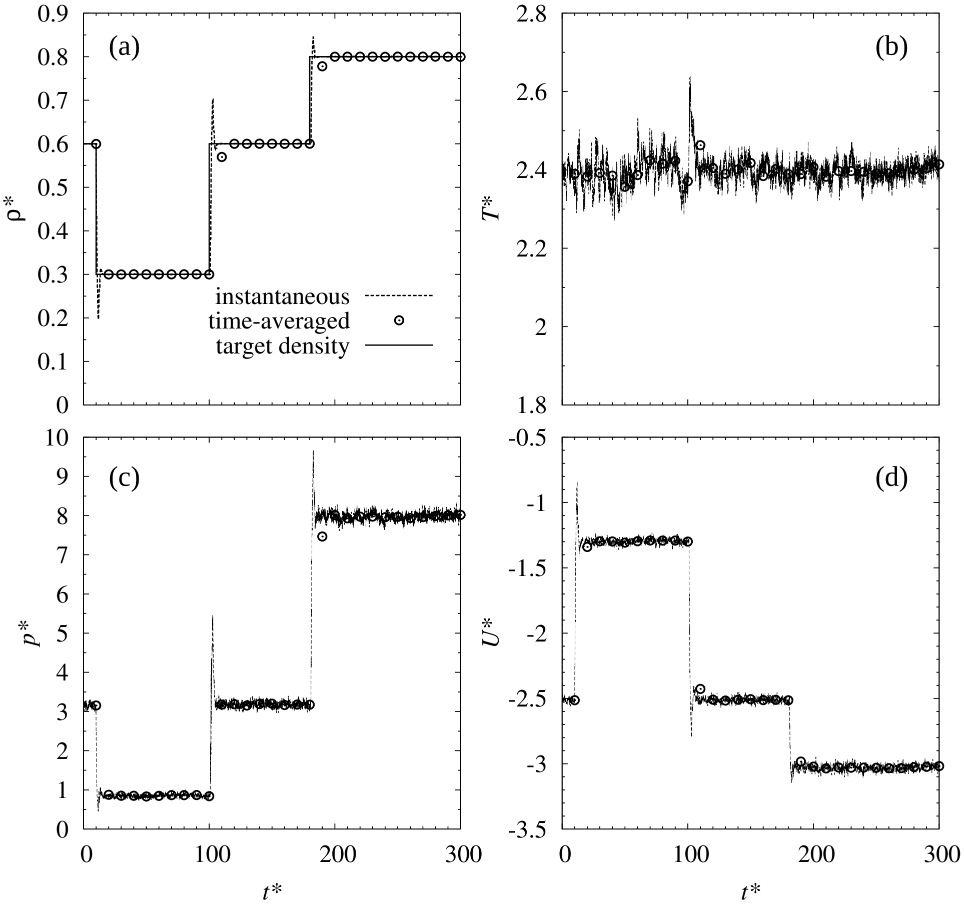


Figure 4.9: Density is controlled in individual cells (volume $\Delta V^* = 4^3$) of a MD system with periodic boundary conditions applied at all boundaries of the cubic domain. A step-input of density (—–) is arbitrarily applied at three time-instances of the simulation. Graphs show (a) density, (b) kinetic temperature, (c) pressure and (d) potential energy for instantaneous measured properties (- - -) and time-averaged properties over a period $\Delta t^*_{av} = 10$ (⊙). Spatial coarse-graining of cell-density measurements allows for short averaging and control time-intervals to be chosen in the density controller ($\Delta t^*_{av} = 1$, $\Delta t^*_{cont} = 0.25$) and therefore there is a rapid convergence of density. The initial density is $\rho^* = 0.6$ (4797 molecules), and temperature remains constant at $T^* = 2.4$ by applying the Andersen thermostat.

The response of the system is dependent on the proportional gain constant:

$$K_P = \frac{\lambda}{n_{cont}} = \frac{\lambda \Delta t_{cont}}{\Delta t_{av}}, \tag{4.13}$$

that is applied to all molecules every control step. A large value of K_P imposes a faster response on the system, but results show highly oscillating velocities and introduction of temperature drifts. A smaller value of K_P will converge the system to the target velocity over longer time-periods without imposing large oscillations of velocity. An overly small value of K_P will not allow the velocity within a cell to converge to the required value.

It is common practice to apply a temperature controller in the same region as the velocity controller so as to counteract the external work-done on the system. For thermostatting to be effective, it needs to control temperature over shorter control periods than velocity. An example of velocity control in individual cells of the same MD simulation is shown in Figure 4.10. We found that the values $\Delta X_P^* = 4$, $\Delta t_{av}^* = 0.5$, $\Delta t_{cont}^* = 0.1$ and $\lambda = 0.8$ (i.e. $K_P = 0.16$) give generally good results.

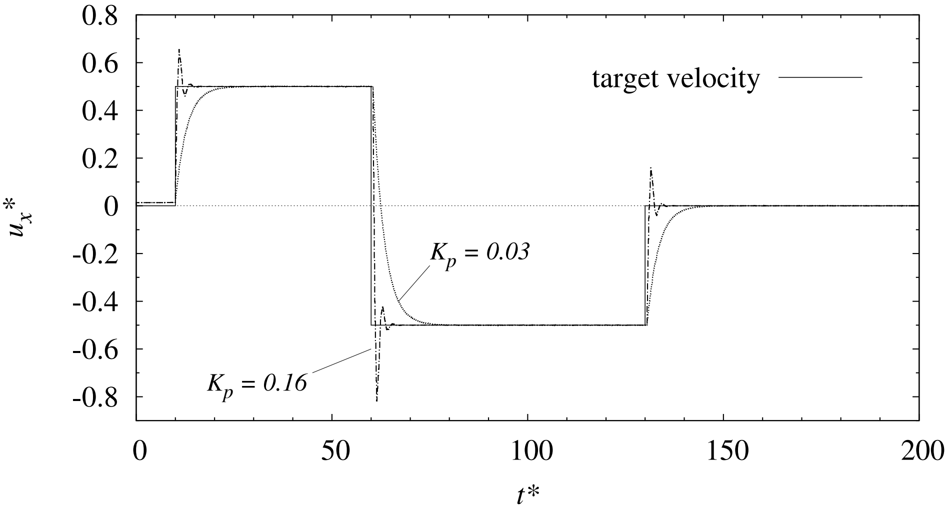


Figure 4.10: Velocity control applied to the same case as in Figure 4.9 within individual cells of the mesh. The averaging and control time-periods are chosen to be $\Delta t_{av}^* = 0.5, \Delta t_{cont}^* = 0.1$ and coarse-graining of velocity measurements within individual cells is applied. Dotted lines show two response characteristics of the MD system for $\lambda = 0.8$ ($K_P = 0.16$) and $\lambda = 0.15$ ($K_P = 0.03$). A Berendsen thermostat is applied to fix the temperature to $T^* = 2.4$.

Similar models in the literature for velocity imposition may be compared to our error-based feedback velocity controller method, in order to provide comparisons for the gain K_p in equation (4.13). We rearrange their equations to resemble our formula (4.11). The 1D momentum constraining technique within longitudinal bins proposed by O'Connell and Thompson [4] makes use of $K_p \approx 0.01$. The constraint Lagrangian technique proposed by

Nie *et al.* [47] provides a limit on the acceleration change applied to molecules:

$$K_P \approx 1 - \frac{\Delta t_m |\mathbf{a}_P^{\text{meas}}(t)|}{\left|\mathbf{u}_P^{\text{req}}(t) - \mathbf{u}_P^{\text{meas}(t)}\right|}, \tag{4.14}$$

where $\mathbf{a}_P^{\text{meas}}(t)$ is the instantaneous cell-averaged acceleration per molecule due to the intermolecular forces. Note that K_P is not kept fixed but varies depending on the velocity error and also the current acceleration of molecules. A value of $K_P \sim 0.2$ is a typical outcome of the simulation. The implicit Schwarz hybrid method used by Werder *et al.* [38] chooses the gain to be the inverse of the number of molecules within the boundary region. So $K_P = 0.03$, if one considers a cell of cubic dimensions $\Delta X_P^* = 4$ and number density $\rho^* = 0.6$.

Multi-species

The velocity controller may control single- and multi-species fluids. The latter is set up by applying a velocity controller per specie within a common controlling region, in which the partial target velocities need to be specified. Measurement of velocity for an arbitrary specie excludes molecules of other species, and control is distributed solely on molecules of the same specie.

Moving wall

Moving walls are required mainly for simulating Couette-type flows, e.g. for determining slip and contact-angles [61], or for the analysis of singularities and shear profiles in cavity-driven problems [57]. Other futuristic nano-devices may require finer methods of control, e.g., for simulating “nano-swimmers” [15].

In our simulations, molecular walls are moved using a new velocity controller model, similar to the model just described for liquids. If an arbitrary wall molecule, i, is tethered to a fixed point in space, $\mathbf{r}_i^{teth}$, via a fictitious spring, an equation of motion is also applied to the tether point. In the Leapfrog algorithm this is implemented as:

$$\mathbf{r}_i^{teth}(t + \Delta t) = \mathbf{r}_i^{teth}(t) + \Delta t_m \mathbf{u}_P^{\text{req}}(t). \tag{4.15}$$

4.4.4 Temperature control

Many thermostats have been proposed in the literature of MD [22]. In this work we implement the common methods, so that control of temperature is applied in (1) a global zone, or (2) exclusively in local-cells. The first is typically used in isothermal MD systems, in which the control zone is the entire domain. In our MD simulations, the zone can generally represent a subset of the MD domain. The temperature controller measures one global value of temperature from molecules that occupy all cells of the zone, and in turn cools or heats equally all residing molecules. The second method uses our cell-based controller methodology, described earlier, so that spatial variation of temperature may be imposed. In practice, we find that a measurement of cell-temperature requires data from neighbouring cells in order to increase its statistical accuracy, hence we use the method of coarse-graining described in §3.5.2. The controller then heats or cools the molecules only within that particular cell. This technique is more suitable for imposing heat flux, applying boundary conditions at different temperatures, and for linking MD cells with a continuum hybrid formulation.

Parallelisation

Parallel communication is required only in the measurement step, for both the zonal and local-cell (with coarse-graining) techniques. The zone method considers the transfer of scalar properties (such as kinetic energy) across all processors, so that the full temperature may be calculated on each processor mesh. The local method, however, requires transferring 'fields' of properties across processors. These are then reconstructed on each processor-mesh, so that cells at processor boundaries may compute the measured temperature from neighbouring cells that reside on other processors. Notably, the control step (heating or cooling of molecules) does not require parallel communication.

Current velocity-scaling thermostat

The velocity-scaling thermostat [22], is a common yet crude technique for controlling temperature. The thermostat involves scaling velocities of molecules by a factor:

$$\chi_Z(t) = \left(\frac{T_Z^{\mathrm{req}}(t)}{T_Z^{\mathrm{meas}}(t)} \right)^{1/2}, \tag{4.16}$$

where subscript Z denotes a zone quantity. The measured temperature $T_Z^{\mathrm{meas}}(t)$ is instantaneous over the entire zone.

In the Leapfrog Verlet algorithm the following steps are implemented after the second-step velocity update:

1. Compute the instantaneous kinetic temperature $T_Z^{\mathrm{meas}}(t) = 2k_B \sum_{i=1}^{N_Z(t_k)} m_i |\mathbf{v}_i - \mathbf{u}_Z|^2 / 2N_Z(t_k)/3$, where $\mathbf{u}_Z = \sum_{i=1}^{N_Z(t_k)} m_i \mathbf{v}_i / \sum_{i=1}^{N_Z(t_k)} m_i$ is the streaming velocity.

2. Compute the scaling ratio χ_Z from equation (4.16).

3. Loop through all molecules i occupying the zone-cells and re-scale their velocities:

$$\mathbf{v}_i(t + \Delta t_m) = \mathbf{v}_i(t + \Delta t_m)\chi_Z. \tag{4.17}$$

Berendsen thermostat

The Berendsen thermostat [62] is similar to the velocity scaling thermostat, but allows for a variable (commonly slower) coupling between molecules and a hypothetical heat bath, set at the target temperature. Molecules undergo scaling using the following modified factor:

$$\chi_Z = \left[1 + \frac{\Delta t_m}{\tau_T}\left(\frac{T_Z^{\mathrm{req}}}{T_Z^{\mathrm{meas}}} - 1\right)\right]^{1/2}, \tag{4.18}$$

where τ_T ($= \{0.2 - 2.0\}$ ps) is a time constant, that defines the coupling strength between the system and the heat bath. This thermostat is similar in implementation to the velocity scaling thermostat above. Indeed, when $\tau_T = \Delta t_m$ the velocity re-scaling thermostat is recovered.

This thermostat showed good results when applied to local-cells, and using coarse-grained instantaneous temperature measurements, as shown in Figure 4.11.

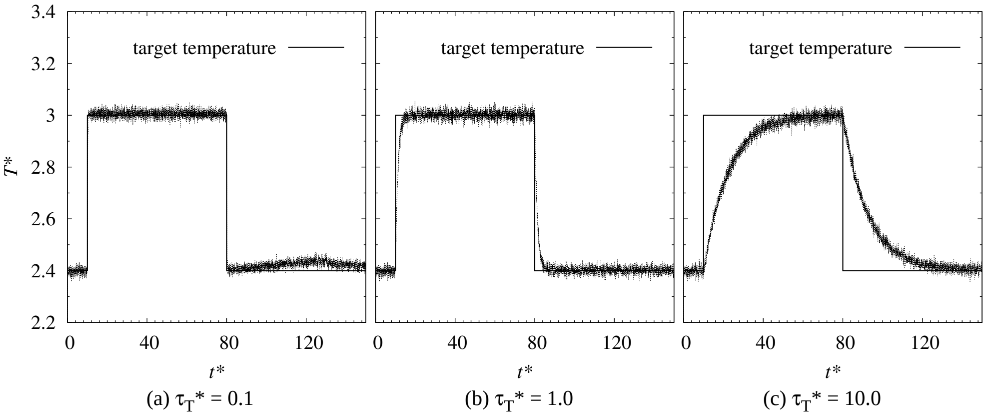


Figure 4.11: Berendsen thermostat using three different relaxation rates $\tau_T^* = \{0.1, 1.0, 10.0\}$, applied to cells of the mesh of the same MD simulation considered earlier. Coarse-grained instantaneous temperature measurements are invoked.

A modified velocity-scaling thermostat

We modify the velocity-scaling thermostat model described earlier so that it can be used in the local-cell technique of our controller methodology. Differences include measuring the cell temperature $T_P^{\mathrm{meas}}(t)$ over longer time periods ($t - \Delta t \rightarrow t$) followed by n_{cont} velocity-scaling steps distributed uniformly over the interval ($t \rightarrow t + \Delta t$), rather than instantaneously. A small equilibration period therefore occurs between one control step and the next. In addition, we introduce coarse-grained temperature measurements from neighbouring cells — only those cells that are within interaction range r_{cut} are chosen — to minimise the averaging time-interval. The procedure is described for an arbitrary control cell P as follows:

1. At a time t, compute the scaling ratio per control step within P:

$$\chi_P(t_k) = \left(\frac{T_P^{\mathrm{req}}(t_n)}{T_P^{\mathrm{meas}}(t)}\right)^{1/(2n_{cont})}, \tag{4.19}$$

where $t \leq t_k < t_n$, $\{k = 1, 2, \dots, n_{cont}\}$, i.e. t_k is an arbitrary control step at which velocity of molecules are scaled.

2. At an arbitrary control step t_k, scale velocities similar to the standard method using $\chi_P(t_k)$ instead of $\chi_Z(t)$.

An example of temperature control using this model is presented in Figure 4.12.

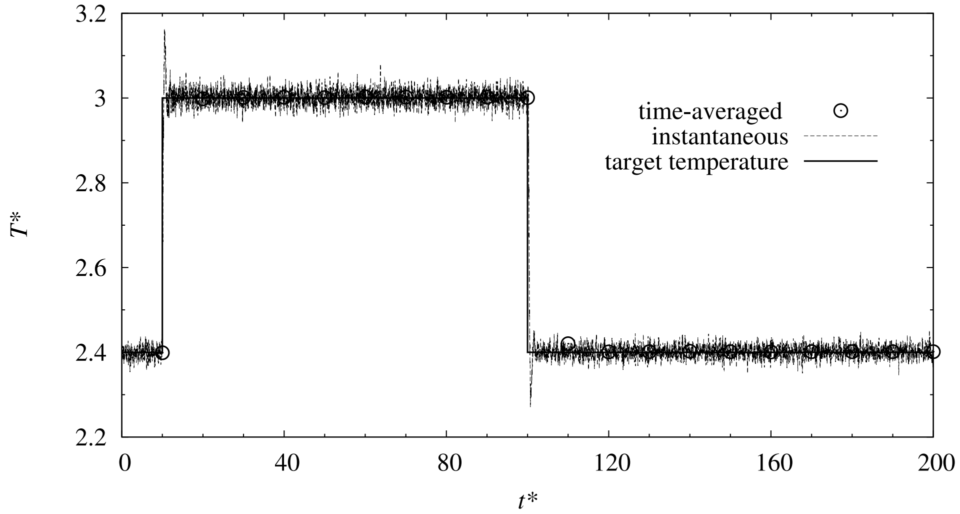


Figure 4.12: Our modified velocity-scaling temperature controller is applied exclusively to cells of the MD mesh, for the case presented in Figure 4.9. The graph shows the instantaneous temperature (- - -), the step input target temperature (——) and the time-averaged temperature (⊙) over $\Delta t^*_{av} = 10$. The cell-temperature is measured using the coarse-graining technique and over a time-averaging period $\Delta t^*_{av} = 0.25$, while the controlling time-interval is $\Delta t^*_{cont} = 0.05$.

4.4.5 A combined temperature and velocity control method

An interesting and effective method for controlling temperature and velocity simultaneously is by using a modified version of the Andersen thermostat [63], that was initially proposed by Hadjiconstantinou and Patera [64] — a “Maxwell Demon”. The standard Andersen thermostat is typically designed to pick molecules at random for Maxwellian thermalisation; this corresponds to a collision with a fictitious heat bath. The exchange of thermal energy occurs at a rate defined by the probability of collision, which is Poisson distributed:

$$p_{col}(t) = 1 - \exp\left(-\frac{\Delta t_m}{\tau_T}\right), \tag{4.20}$$

where τ_T is the relaxation time. Any molecule chosen for collision has its velocity reset from a Maxwell-Boltzmann distribution with zero mean velocity and target temperature $T_P^{\text{req}}(t)$, followed by a momentum conservation process that maintains the same centre-of-mass momentum of the system before and after collisions.

In our modified technique, an arbitrary molecule's velocity is also sampled from the Maxwell-Boltzmann distribution from a target velocity $\mathbf{u}_P^{\text{req}}(t)$ and temperature, but excludes the momentum conservation step at the end of the thermostatting process. Therefore, this resembles the collision of the molecule with a combined momentum and heat reservoir. While in [64], velocities of all boundary molecules are drawn from the Mawellian distribution every time-step, we use the probability of collision in equation (4.20) to allow for variable coupling.

Probably the major benefit of this new controller is that no feedback of temperature or velocity measurements is required. As a result, this technique is independent of the control cell-size or the averaging time-interval. Hence, highly spatial and temporal variations of target temperature and velocity may (in principle) be applied in arbitrary control regions. The coupling strength with the heat and momentum bath is controlled through the time-constant τ_T, as shown in Figure 4.13.

The drawbacks of this velocity-temperature constraint technique are that: it may only be applied in regions where the assumption of local thermodynamic equilibrium exists (typically at boundaries of MD simulations); it destroys momentum conservation of the system; and that the fluctuations in properties which are introduced by the scheme do not resemble any known MD ensemble.

All molecules within a control cell P are tested for collision, at the end of the Leapfrog algorithm. For an arbitrary molecule i;

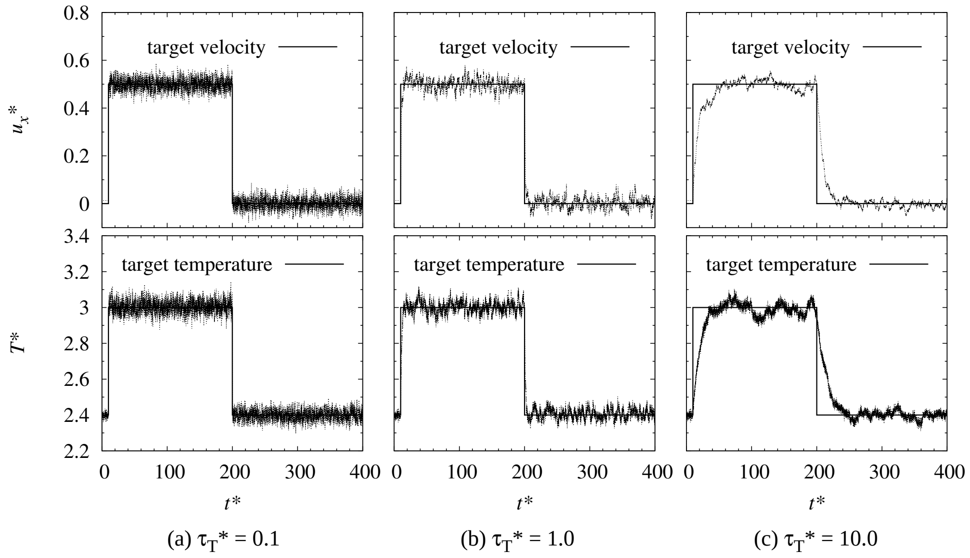
1. Select a random variate $\tilde{p}_i$ uniformly on [0,1].
2. **If** $\tilde{p}_i \leq p_{col}(t)$ from equation (4.20);
   Then collide molecule i with the heat and momentum bath:

   $$\mathbf{v}_i(t + \Delta t_m) = \tilde{\mathbf{v}}_{MB}, \tag{4.21}$$

   where $\tilde{\mathbf{v}}_{MB}$ is a random velocity sampled from the Maxwell-Boltzmann distribution at the cell-target temperature $T_P^{\text{req}}(t)$ and velocity $\mathbf{u}_P^{\text{req}}(t)$ (see Appendix A).

4.4.6 Pressure control

In the literature, pressure control mainly consists of varying the volume of the domain box, concomitant with scaling of molecular coordinates, in order to compress or expand a fluid [22]. This technique, however, can only be applied to achieve global pressure changes. Pressure control in engineering problems requires the simulation domain to remain spatially fixed, with a locally intrusive method for modifying pressure. Some examples are: applying different pressure constraints at inlet and outlet boundary regions of nano-channels, or applying local pressure changes that are set by overlaying CFD pressure solutions in a hybrid simulation.



(a) $\tau_T^* = 0.1$ (b) $\tau_T^* = 1.0$ (c) $\tau_T^* = 10.0$

Figure 4.13: Simulations showing an instantaneous step input of target velocity and temperature simultaneously applied to the same periodic MD case described in Figure 4.9. The modified Maxwell-Demon controller is applied to individual cells, for a range of $\tau_T^* = \{0.1\,1.0, 10.0\}$. Graphs show the response of the system to velocity (top) and temperature (bottom).

In view of this, we propose a new control method that differs from existing techniques by applying pressure changes in local cells of the mesh. The model which we employ for controlling pressure resembles that proposed by Berendsen [62]. However, there are some important differences between the two approaches. The Berendsen barostat drives the instantaneous system pressure to a target value by simultaneously scaling the volume of the simulation box by a factor χ and scaling all molecular positions from a reference point by a factor $\chi^{1/3}$ so that they remain confined within the domain box:

$$\chi(t) = 1 - \frac{\Delta t_m}{\tau_T}(p^{\text{req}}(t) - p^{\text{meas}}(t)), \tag{4.22}$$

where $p^{\text{req}}(t)$ is the required pressure, $p^{\text{meas}}(t)$ is the instantaneous measured pressure (see equation (3.10)) and $\tau_T^* = \mathcal{O}(0.01) - \mathcal{O}(0.1)$ is a time constant in reduced units that represents the strength of coupling. The number of molecules within the system do not change, and a thermostat is also applied to maintain constant temperature.

In our technique the domain-mesh is kept fixed, so that the cells remain of constant-volume throughout the simulation run. For pressure control to be applied independently within required cells of the mesh, we exclude completely the notion of “molecular co-ordinate scaling”. Instead, we base our scheme either on density scaling, temperature scaling or a ‘blend’ of both techniques. In the first technique, the measured *cell-density* is scaled by a factor χ, to give a target density closer to the target pressure. The required cell-density is controlled using the density controller described in §4.4.2. Molecules are inserted/deleted within an arbitrary control cell P at sites equal to the average potential energy per molecule. In this case, a thermostat at constant temperature may be applied to the cells, so that a change in density corresponds directly to a change in pressure. In the second technique, the measured *cell-temperature* is scaled by χ, and is controlled by the cell-based velocity-scaling thermostat described in §4.4.4. A density controller may be applied to maintain a fixed density, so that a change in temperature corresponds directly to a change in pressure. In the final technique, we employ both density and temperature controllers operating in one control region. The magnitude change in each property is weighted by a blending coefficient, α, $(0 \leq \alpha \leq 1)$: $\alpha = 1$, is purely density control, and $\alpha = 0$ is purely temperature control. Obviously, the value chosen for α depends on the problem.

From our MD simulations involving control of pressure using equation (4.22) we observed that the system converges faster to a positive change in pressure $\Delta p^{+} = p^{\mathrm{req}}(t) - p^{\mathrm{meas}}(t) > 0$ than a negative change $\Delta p^{-} = p^{\mathrm{req}}(t) - p^{\mathrm{meas}}(t) < 0$. In fact, for the case of $\alpha = 1$ and Δp^{+}, 'under-damped' pressure oscillations are observed, which in turn impose undesirably large target densities, which effectively slows down the MD simulation. We alleviate these issues by modifying the scaling factor formula for positive pressure changes and impose an upper and lower limit of target density ($\rho^{*}_{max} \sim 0.9$, $\rho^{*}_{min} \sim 0.01$):

1. At a time t, compute the scaling factor within P:

$$\chi_P(t) = \begin{cases} 1 - \frac{\Delta t_m}{\tau_T}\left(p_P^{\mathrm{req}}(t) - p_P^{\mathrm{meas}}(t)\right), & \text{if } p_P^{\mathrm{req}}(t) < p_P^{\mathrm{meas}}(t), \\ \\ P\left(2 - \frac{1}{1 - \frac{\Delta t_m}{\tau_T}(p_P^{\mathrm{meas}}(t)) - p_P^{\mathrm{req}}(t)}\right)^{-1}, & \text{if } p_P^{\mathrm{req}}(t) > p_P^{\mathrm{meas}}(t), \end{cases} \tag{4.23}$$

   where $p_P^{\mathrm{req}}(t_n)$ is the required cell-pressure at the time $t_n = t + \Delta t$ and $p_P^{\mathrm{meas}}(t)$ is the measured cell-pressure during the time-interval $t - \Delta t \rightarrow t$.

2. Compute the new target cell-density in P:

$$\rho_P^{\mathrm{req}}(t_n) = \frac{\rho_P^{\mathrm{meas}}(t)}{\chi_P(t)} + \rho_P^{\mathrm{meas}}(t)\left(1 - \frac{1}{\chi_P(t)}\right)(1 - \alpha), \tag{4.24}$$

   where $\rho_P^{\mathrm{meas}}(t)$ is the cell density measured during a previous time interval Δt.

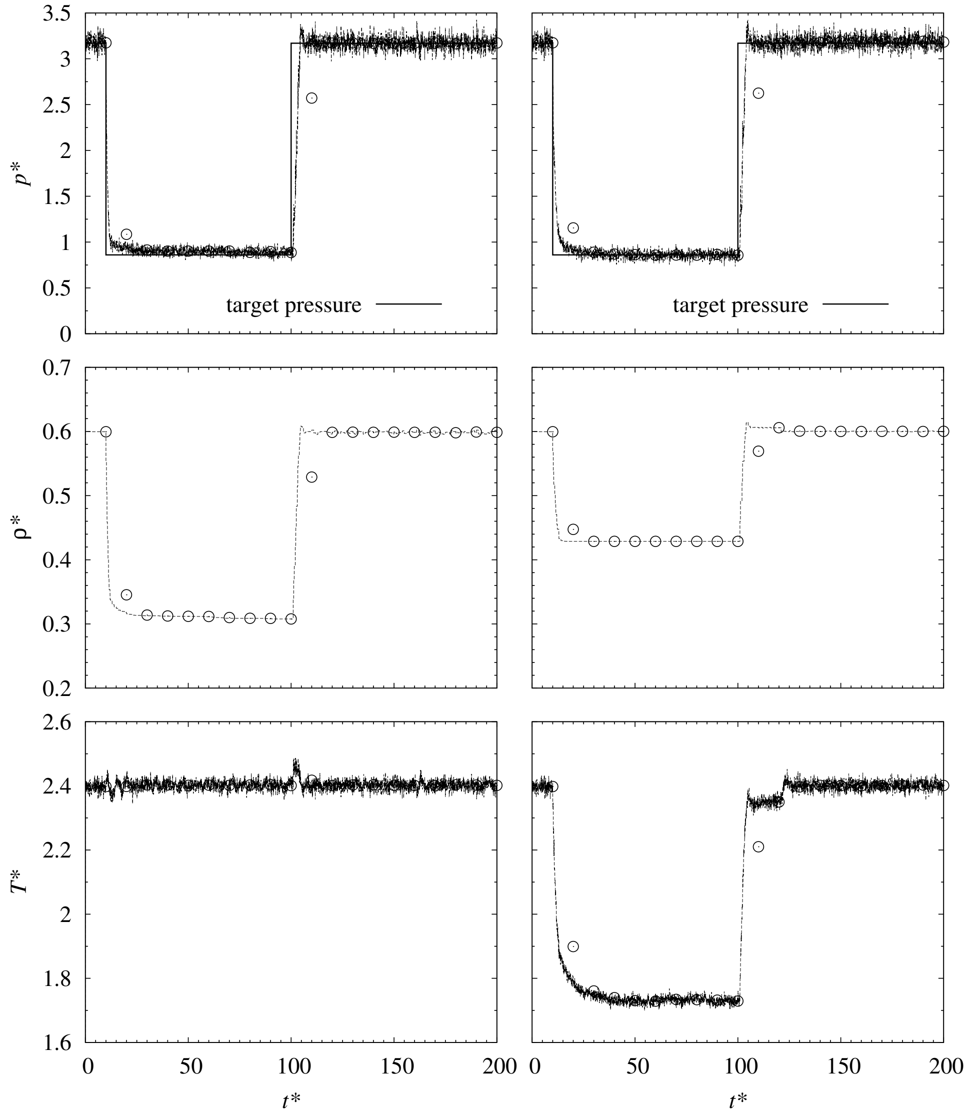
3. Compute the new target cell-temperature in P:

$$T_P^{\mathrm{req}}(t_n) = \frac{T_P^{\mathrm{meas}}(t)}{\chi_P(t)} + T_P^{\mathrm{meas}}(t)\left(1 - \frac{1}{\chi_P(t)}\right)\alpha, \tag{4.25}$$

   where $T_P^{\mathrm{meas}}(t)$ is the cell temperature measured during a previous time interval Δt.

4. Use the density controller described in §4.4.2 to drive the cell-density to the target $\rho_P^{\mathrm{req}}(t_n)$, and the cell-based thermostat in §4.4.4 to drive the cell temperature to the target $T_P^{\mathrm{req}}(t_n)$, for the time-interval $t \rightarrow t + \Delta t$.

Examples of time-changes in target cell pressures are shown in Figure 4.14.



(a) α = 1.0 (b) α = 0.5

Figure 4.14: MD simulations of the case described in Figure 4.9, in which an instantaneous step input of target pressure is applied in all cells of the mesh. Properties of the simulation run are: $\tau_T^* = 0.03$, $\Delta t_{av}^* = 1$ and $\Delta t_{cont}^* = 0.25$. Graphs show the system response to the step input of pressure for (a) varying density only ($\alpha = 1.0$), and (b) varying both density and temperature equally ($\alpha = 0.5$).

4.4.7 State-controllers in MD simulations

The state-controllers we have described may be used in three MD scenarios:

1. *Initial state for MD simulations* — Generally, in most of our MD simulations we use these controllers to set an accurate initial molecular state prior to the main MD simulation (see §2.4). First we use our pre-processing utility [27] for generating initial lattice structures of molecules in different zones of the domain mesh. Then, we apply an MD simulation with the controllers applied in all cells of the mesh, in order to converge the local fluid properties to the targets.
2. *Internal conditions* — Controllers may be applied in a subset of the MD domain, or in the entire domain, in order to constrain a particular property during the main MD simulation run (e.g., for isothermal cases). The resulting controlling procedure acts as an internal condition on the MD system. We reiterate that with our new controllers a property may be maintained uniform throughout the domain, or may even vary spatially and/or temporally, for example, to setup steady-state gradients.
3. *Boundary conditions* — The controllers may be applied solely in non-periodic boundary regions in order to constrain a particular property within individual cells. Apart from applying boundary conditions in pure MD simulations, these controllers have especially been formulated so that coupling with a continuum formulation in a hybrid scheme may be performed (a) in complex geometries, and (b) in parallel.

4.4.8 Future developments of new controller models

The underlying code of the controllers is generic and common to all models. It is therefore relatively easy to implement a new continuum-to-MD controller model based on this control system. The procedure generally requires two steps: a ‘copy-and-paste’ step of an existing model code, and the explicit definition of each of the new control-loop functions, as described in §4.4.1. Furthermore, once a new model is constructed, it is appended to the complete list of existing models automatically; the ‘run-time selection’ algorithm is a common property of any arbitrary controller. Therefore, any future users of the MD code may choose this new model from the model list.

Parallelisation is an issue which is solved internally to the controller model code. During parallel processing, the controller is ‘cloned’ on each processor-mesh. Functions of the controller therefore occur in tandem on all processors. If data needs to be sent across the same controller located on different processors, sending and receiving communication schemes have to be coded within the individual functions.

In summary, we have created a controlling framework that not only enhances flexibility during use, but also provides a simple and fast pathway towards the development and testing of new methods of control, continuum-to-MD boundary conditions, and hybrid MD-CFD coupling, in arbitrary geometries and in parallel.

4.5 Modelling the Domain Boundary

In this section we describe existing and new boundary models of Type I that have been implemented in our MD simulation code. Type I models consist of applying an 'action' on an arbitrary molecule, i, upon collision with an arbitrary boundary face, f. We also describe models of Type II that are applicable solely to the domain boundaries (see §4.5.5 and §4.5.4). In these models perturbations are applied to molecules when they come close to the boundary face.

4.5.1 Existing Boundary Models

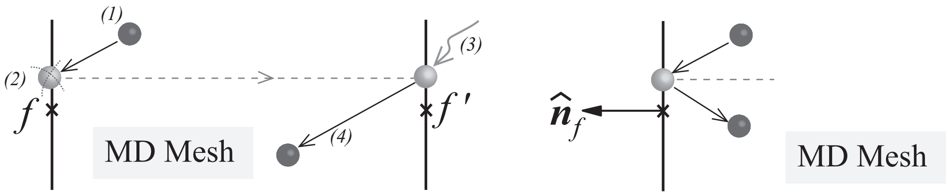
Some basic boundary models are already implemented in OpenFOAM's Lagrangian library for generic particles. We describe these in brief here, however the reader may refer to Ref. [20] for more information on how these have been implemented in the MD code.

Cyclic boundary

A cyclic boundary is a requirement for implementing periodic boundary conditions (PBCs). A molecule i is deleted as it hits a face f, and a new molecule i' inserted (wrapped around) at the coupled face f', with the same molecular properties as i except the global coordinate (see Fig 4.15(a)). The molecule's *tracking fraction*, $0 \leq \lambda_i \leq 1$ is also conserved for each MD time-step. Therefore, if $\lambda_i < 1$ when molecule i hits f, it may proceed with the remainder of the motion $(1 - \lambda_i)$ from its new position on f' (see Ref. [26]). PBCs are fully implemented by the addition of a process that refers images of molecules from the boundary region of one coupled boundary to the external region of the other, after every MD time-step. This is detailed in Ref. [20]. Therefore, we can say that a PBC is composed of two boundary models: a cyclic boundary of Type I that conserves mass, momentum and kinetic energy; and a force field boundary model of Type II that conserves potential energy.

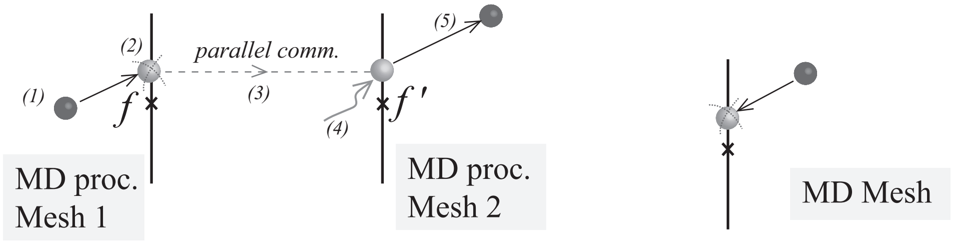
Processor boundary

A processor boundary model is required when the mesh is decomposed on distributed processors. In this setup, internal faces on the original mesh become boundary faces on the decomposed sub-meshes. The objective of a processor boundary condition is, therefore, to ensure the presence of the disconnected meshes does not affect the dynamics of molecules, but instead allows the system to develop in a manner identical to that of a serial processing scenario. A processor boundary model is very similar to the cyclic boundary model, in the sense that when a molecule i hits a processor boundary face f, the molecule is deleted from the current processor, and recreated at its coupled face f' located on another processor (see Fig 4.15(b)). Parallel communication between processors occurs first, and subsequently, the new molecule continues its motion according to the remaining tracking fraction. Similar to the cyclic boundary, all properties are conserved, including the global position. Conservation of potential energy is achieved using a similar molecule-referral scheme [20].



(a) Cyclic boundary

(c) Specular-reflective boundary



(b) Processor boundary

(d) Deletion boundary

Figure 4.15: The standard boundary models in OpenFOAM: (a) cyclic (coupled), (b) processor (coupled), (c) specular-reflective and (d) deletion boundaries.

Specular wall boundary

The function of a specular reflective wall boundary model is to invert the molecule's velocity component normal to the colliding face (see Fig 4.15(c)). Upon collision the new velocity of an arbitrary molecule i, is $\mathbf{v}_i' = \mathbf{v}_i - 2\hat{\mathbf{n}}_f(\hat{\mathbf{n}}_f \cdot \mathbf{v}_i)$ where $\hat{\mathbf{n}}_f$ is the normal vector of f pointing outwards of the domain. The molecule completes its remaining fraction of trajectory after colliding with the wall.

Deletion boundary

A deletion boundary model completely deletes molecules that collide with it (see Fig 4.15(d)). It may be used to model vacuum-like outlet conditions [2] in nano-channel simulations.

4.5.2 Development of New Boundary Models

We have developed and implemented additional boundary models in our MD solver, which we describe in the following sections. A data-structure similar to that used in the state-controllers section (§4.4) is employed, in which any new boundary model inherits its properties from a 'generic boundary'. The generic boundary will store properties such as links to the geometrical boundary face properties on the mesh (e.g., face-centres, normal face vectors, boundary cells etc.), and surface-fields of continuum-target properties (e.g., face interpolated density, velocity and temperature), if needed.

We develop two different types of generic boundaries, for (a) Type I and (b) Type II boundary models. When a molecule i collides with a boundary face f that is associated with a boundary model of Type I, it activates the model's 'function' which will perform the required actions on that molecule. Only one Type I model per boundary is permitted. Type II boundary models apply actions on molecules that are within a specific distance ($\sim r_{cut}$) of the boundary face f. Any number of Type II boundary models may be superimposed at the same boundary.

The aim of this data structure is to facilitate the rapid advancement and development of new boundary models. Implementing new boundary models requires a relatively simple 'copy-paste' procedure of an existing model, and then modifying the function that applies actions on molecules. Parallelisation issues are similarly solved internally to the model.

The boundariesDict input file

The user specifies the boundary models and their required input properties from the boundariesDict file. The layout is comparable to controllersDict, as shown in Figure 4.5.

4.5.3 Stochastic thermal wall boundary

We implement the stochastic thermal-wall boundary model of Ciccotti and Tenenbaum [52], and generalise its application to arbitrary boundaries using the local-face approach. Known quantities at an arbitrary boundary face, f, are the face interpolated temperature T_f, and velocity $\mathbf{u}_f$, and a probability p_f of molecules to undergo a collision with the 'heat bath'. The latter quantity is introduced to generalise the model between diffuse and specular: $1 - p_f$ is the probability for molecules to be specularly reflected. The following procedure is performed when a molecule i collides with f:

1. Determine a random variate $\tilde{r}$ from [0,1].
2. A stochastic collision occurs if $\tilde{r} < p_f$. The molecule is specularly reflected otherwise.
3. In a stochastic collision, the new velocity is given by $\mathbf{v}_i = \tilde{\mathbf{v}}_{MB}$, where $\tilde{\mathbf{v}}_{MB}$ is a random velocity vector sampled from a Maxwell-Boltzmann distribution at temperature T_f and mean velocity $\mathbf{u}_f$.
4. If the molecule's velocity points out of the domain, i.e. $\mathbf{v}_i \cdot \hat{\mathbf{n}}_f > 0$, the molecule is reflected back into domain.

4.5.4 Mass flux boundary models

A key technique needed for simulating incompressible or compressible fluid flows in non-periodic MD systems is to impose the correct mass flow rate at an arbitrary boundary.

Some examples include inlet and outlet boundary conditions in nano-channels, or for coupling with a continuum formulation in a hybrid scheme. In the literature, two types of models have been described:

1. An external force diverging distribution is first imposed on molecules, which varies from 0 in the bulk to ∞ at the boundary in order to gently create a steady-state dilute region at the termination edge of the MD domain. In this 'relaxation zone', the properties of density vary from zero (at the edge) to the bulk properties over a short distance (typically $2 - 15\sigma$ for liquids, and $5 - 70\sigma$ for gases [65]). Molecules may then be inserted or deleted at ease from the edge of the relaxation zone. See, for example, references [8, 47, 65].

2. Two identical boundaries are moved in synchronous motion using piston-type actuation [38] with magnitude and direction of the mean velocity of the system, while periodic boundary conditions are applied in the other two directions. Molecules may collide with the moving boundary walls during their motion. After the MD time-step, the walls are reset to their original positions, leaving some molecules outside the domain (on the outlet side). These molecules are then re-inserted at the inlet, in order to maintain a constant number of molecules in the system, and hence apply an incompressible flow condition.

We describe here the drawbacks of these techniques. In the first method, it is unphysical to impose large gradients of properties — they do not occur in the scenario which they are addressing — nor is it correct to apply boundary conditions to a gaseous phase that are derived from liquid properties. In the second method, the two boundaries are coupled together and the piston-like motion limits their application solely to two planar boundaries. For this reason, they may not be applied to complex boundaries nor to generalised single hybrid MD-continuum interfaces. The deletion boundary model, described earlier (§4.5.1), may only act as an outlet condition, while the mass flux is an uncontrollable parameter.

Because of these problems we implement a new mass flux boundary model (Type II) that differs from the literature models as follows. First the mesh is fixed, so that no moving piston-like boundary walls are applied. Second we introduce a buffer region close to the boundary instead of the relaxation zone (see Figure 4.2), and employ a force field boundary model (§4.5.5) so that the fluid properties remain constant right up to the edge of the MD system. In fact, this mass flux boundary model will also prevent density dipping unphysically to zero close to the boundary. The boundary itself is modelled by a Type I reflective or stochastic-diffuse wall model to confine molecules within the domain. Finally, we may choose a mass flux model from one of the schemes described below, so as to insert or delete molecules as close as possible from an arbitrary boundary face f, at a rate prescribed by a continuum mass flux, $\dot{m}_f$. Using this local-face technique, any arbitrarily shaped boundaries may be applied in MD or hybrid MD simulations. In the latter, spatial and time-variations of mass flux may be prescribed. Control is applied using a similar algorithm to the density controller (§4.4.2), hence parallel processing of this method is also achieved.

Mass flux schemes

The mass flux may be computed using one of the following techniques, and then imposed using a common method which we describe below:

1. *Uniform mass flux model* — Specify one spatially-uniform value of $\dot{m}_f$ from an input file, which is applied to all faces of the boundary.

2. *'Symmetric' mass flux model* — The mass flux at an arbitrary face f is computed from local measured properties:

   $$\dot{m}_f(t) = \rho_f^{\text{meas}}(t)\mathbf{u}_f^{\text{meas}}(t) \cdot \mathbf{A}_f, \tag{4.26}$$

   where $\mathbf{A}_f = A_f \hat{\mathbf{n}}_f$ is the surface area normal vector which points out from the domain, and $\rho_f^{\text{meas}}(t)$, $\mathbf{u}_f^{\text{meas}}(t)$ are the coarse-grained mass density and velocity measurements respectively taken close to face f. Measurements occur in neighbouring cells located a distance r_{cut} from face f.

3. *Constant density mass flux model* — The mass flux is selected to be proportional to the number density error specified within a cell-zone Z adjacent to the boundary. Mass flux may therefore be computed from:

   $$\dot{m}_f(t) = -\frac{m_i V_Z}{n_{faces}\Delta t}\left(\rho_Z^{\text{req}}(t) - \rho_Z^{\text{meas}}(t)\right), \tag{4.27}$$

   where n_{faces} is the number of faces on the boundary, m_i is the mass of one molecule, V_Z is the volume of the zone, Δt is the time-interval over which the mass flux is imposed, and $\rho_Z^{\text{meas}}(t)$ is the measured number density within the zone between $t - \Delta t \rightarrow t$. $\rho_Z^{\text{req}}(t)$ is a user-specified phenomenological parameter which determines the rate of mass flux emanating from the boundary.

4. *Coupled mass flux model* — The mass flux is obtained directly from a CFD finite-volume simulation. The MD boundary is linked with a corresponding face-zone on the CFD mesh. Each face on the continuum face-zone sets the prescribed mass flux to its corresponding face on the MD boundary:

   $$\dot{m}_f(t) = \rho_f^c(t)\mathbf{u}_f^c(t) \cdot \mathbf{A}_f, \tag{4.28}$$

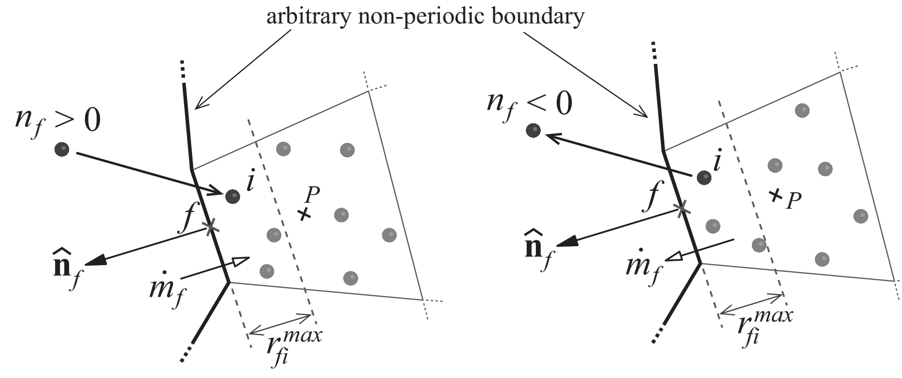
   where 'c' denotes fields obtained from the continuum solution.

Numerical implementation of mass flux imposition

The number of molecules to be inserted or deleted during a time interval $t \to t_n$ ($t_n = t + \Delta t$), at an arbitrary boundary face f is:

$$\Delta n_f(t \to t_n) = -\int_t^{t+\Delta t} \frac{\dot{m}_f(t)}{m_i}\, dt. \tag{4.29}$$

In equation (4.29) the negative sign is applied so as to keep the same convention as the density controller: molecules are inserted when $\Delta n_f > 0$ or deleted when $\Delta n_f < 0$ (see Fig 4.16).



(a) $\dot{m}_f < 0$ (b) $\dot{m}_f > 0$

Figure 4.16: Schematic of the mass flux boundary at an arbitrary boundary face, f, showing (a) a negative mass flux (insertion of molecules required), and (b) a positive mass flux (deletion of molecules required).

The numerical procedure for imposing this mass flux $\dot{m}_f$ at f is:

1. At a time t, compute the number of whole molecules to insert or delete at f during the following time interval Δt:

$$\Delta n_f(t \to t_n) = -\mathrm{NINT}\left(\frac{\Delta t}{2m_i}\left[\dot{m}_f(t_n) + \dot{m}_f(t)\right]\right) + \mathrm{ceil}(\delta n_f(t_o)), \tag{4.30}$$

   where $\dot{m}_f(t), \dot{m}_f(t_n)$ are the target mass fluxes at time t and $t + \Delta t$ respectively, and NINT(x) is the nearest integer function, since whole molecules only may be inserted/deleted. If $\dot{m}_f(t_n)$ is not known at the current time t it is either set to $\dot{m}_f(t)$ or extrapolated from old quantities, e.g., $\dot{m}_f(t_n) \approx 2\dot{m}_f(t) - \dot{m}_f(t_o)$, where $t_o = t - \Delta t$. The former condition is sufficient so long as $\Delta t < t_{flow}$, where t_{flow} is the time-characteristic of the flow velocity. Notably, in this scheme there is no feedback loop of measured mass flux. We assume that since the boundary is reflective, it imposes a zero net flux of molecules crossing the boundary. Thus, the mass flux boundary imposes the net flow of molecules at the MD boundary.

The term $\delta n_f(t_o)$ in equation (4.30) caters for conservation of mass flux, and represents the *cumulative* residual of molecules that arise either due to the NINT(x) function or from the net number of molecules that fail to be inserted/deleted in the previous controlling time-interval ($t_o \to t$). Whenever the term ceil$(\delta n_f(t_o))$ produces a whole number, it is added to equation (4.30).

2. The $\Delta n_f(t \to t_n)$ molecules are distributed uniformly over the span of Δt using $n_{cont} = \Delta t/\Delta t_{cont}$ number of control steps, where Δt_{cont} is the controlling time-interval. This scheme is similar to the density controller (see Steps 2–3 of the method in §4.4.2). However, the insertion/deletion procedures for imposing $\Delta n_f(t_k)$ molecules are different given that it is performed close to faces rather than uniformly within a cell. Note that t_k is an arbitrary control step, $t \leq t_k < t_n$, $\{k = 1, 2, \ldots, n_{cont}\}$.

Inserting molecules

If $n_f > 0$, these molecules are inserted as close as possible to the boundary face f, within its adjacent boundary cell P (see Fig 4.17). The algorithm we implement, also uses the USHER scheme for site-searching (see Appendix B.1), as was similarly applied by our density controller (§4.4.2), but with a few modifications described below.

For an insertion of molecule i at an arbitrary face f:

1. A new molecule i is initially placed randomly on the surface of f at:

$$\mathbf{r}_i^{(0)} = \mathbf{r}_f + (\mathbf{r}_{1\perp}\cos\tilde{\theta} + \mathbf{r}_{2\perp}\sin\tilde{\theta})\tilde{r}, \tag{4.31}$$

where $\mathbf{r}_f$ is the position-vector of the face-centre, $\tilde{\theta}$ is a random variate chosen uniformly on $[0,\ 2\pi]$ and $\tilde{r}$ is an independent random variate chosen between $[0,\ r_{vf}]$, where r_{vf} is the distance between the face-centre and the farthest vertex on the face (see Fig 4.17(a)). In equation (4.31), $\mathbf{r}_{1\perp}$ and $\mathbf{r}_{2\perp}$ are two orthogonal vectors to the face normal vector $\hat{\mathbf{n}}_f$, that are determined once at the beginning of the simulation:

   (a) Determine the tangent vector between a random vector $\tilde{\mathbf{r}}$ and $\hat{\mathbf{n}}_f$:

$$\mathbf{r}_{tan} = \tilde{\mathbf{r}} - (\tilde{\mathbf{r}} \cdot \hat{\mathbf{n}}_f)\hat{\mathbf{n}}_f. \tag{4.32}$$

   (b) If $|\mathbf{r}_{tan}|$ is zero, repeat the procedure by picking a new random vector. This is necessary because $\tilde{\mathbf{r}}$ could be chosen with the same unit vector as $\hat{\mathbf{n}}_f$.

   (c) The two orthogonal unit-vectors then are:

$$\mathbf{r}_{1\perp} = \mathbf{r}_{tan}/|\mathbf{r}_{tan}|, \qquad \mathbf{r}_{2\perp} = \hat{\mathbf{n}}_f \times \mathbf{r}_{1\perp}. \tag{4.33}$$

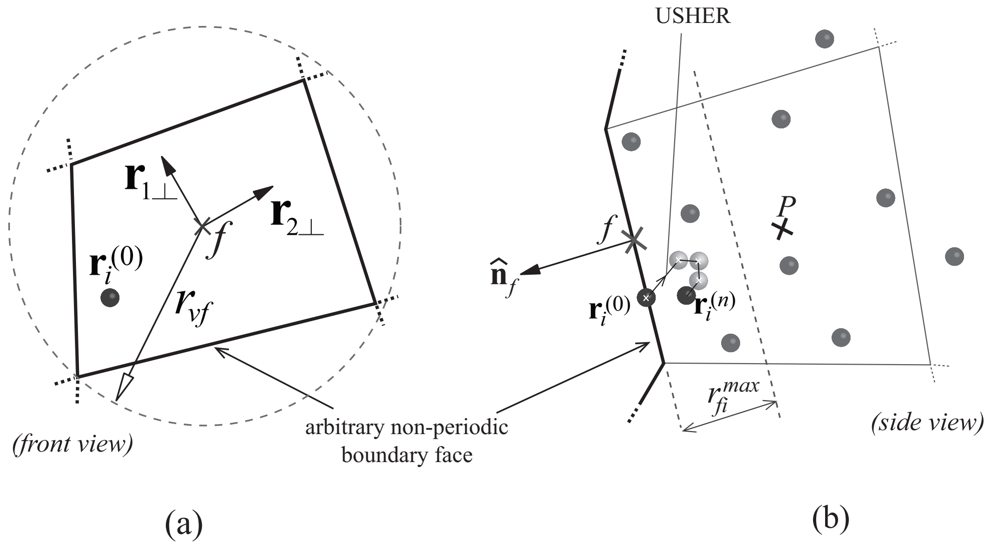


Figure 4.17: Figure showing the insertion of a molecule i within a maximum normal distance r_{fi}^{max} from an arbitrary non-periodic face f. Figure (a) shows the initial position $\mathbf{r}_i^{(0)}$ of the molecule i on f, while (b) shows the USHER scheme used to find an insertion site $\mathbf{r}_i^{(n)}$ within cell P.

2. If the distance between molecule i and the closest existing molecule j, $r_{ij} < r_{ij}^{min}$ (e.g., $\sim 0.1\sigma$), the molecules are too close to compute the force or potential energy that is required by the site-searching algorithm. The molecule is therefore deleted and the algorithm proceeds to Step 1, to attempt a new try.

3. The final step is to shift molecule i within the boundary cell using the site-searching USHER algorithm described in Appendix B.1, until a non-overlapping location in the cell P is found at the desired potential energy U_P^{req}. In view of the fact that the boundary is non-periodic, the potential energy of molecule i includes also the missing potential energy beyond the boundary $U_{bi}(r_{fi})$, where $r_{fi} = (\mathbf{r}_f - \mathbf{r}_i) \cdot \hat{\mathbf{n}}_f$ is the normal distance between i and f (see §4.5.5). Furthermore, if during the steepest-descent scheme the molecule i exceeds a normal displacement from f, $r_{fi} > r_{fi}^{max}$, it is then deleted and the procedure starts again from the beginning. The value chosen for r_{fi}^{max} ($\sim 1 - 2\sigma$ for cell-sizes $\Delta X \sim 4\sigma$) determines the rate of successful insertions and computational cost.

Deleting molecules

The model which we use for deleting molecules is similar to the insertion model. The algorithm searches through all existing molecules within the boundary cell, and chooses the candidate molecule with the shortest normal distance r_{fi} from the boundary face f. Furthermore, the molecule is chosen within the maximum allowable normal distance r_{fi}^{max}.

Multi-species

All mass flux boundary schemes we have proposed are applicable for single- and multi-species mass flux control. The setup of multi-species fluids requires a superimposition of one mass flux model per specie at the same boundary. In view of this, the boundary may insert/delete molecules of different species at different mass flux ratios.

4.5.5 Force field boundary models

An external force field needs to be applied to boundary molecules in order to account for the excluded (or missing) molecules beyond the non-periodic MD domain. These force fields, which are derived from the mean-structure of the liquid state, have been shown largely to rectify the finite-size effects that occur close to the boundary (e.g., density and pressure oscillations). In the literature, however, they have only been implemented in MD simulations for predominantly simple planar boundaries. In this section, we propose models for boundaries of arbitrary geometry. These are necessary for applying NPBCs in hybrid MD-continuum simulations encompassing complex 3D coupling regions.

Furthermore, we note that while the force field predicts the correct dynamical trajectories of boundary molecules, an additional update of potential energy and virial contributions due to the missing molecules are also required; the former to insert new molecules at the correct potential energy site by the mass flux boundary model, while the latter to measure properties such as pressure. The potential energy and virial of an arbitrary molecule i are given by:

$$U_i = \frac{1}{2}\sum_{j=1(\neq i)}^{N_{mols}} U(r_{ij}); \qquad w_i = \frac{1}{2}\sum_{j=1(\neq i)}^{N_{mols}} f(r_{ij})r_{ij}, \tag{4.34}$$

where $f(r_{ij})$ is the intermolecular force potential.

Simple 3D model

We describe a simple model that extends the 1D distribution method in the literature to 3D boundaries (see Fig 4.18). As a prerequisite step, the 1D force, potential energy and virial distributions i.e. $\mathbf{f}_{bi}(r_{bi})$, $U_{bi}(r_{bi})$, $w_{bi}(r_{bi})$, are obtained prior to the main MD simulation run, where $r_{bi} = (\mathbf{r}_b - \mathbf{r}_i)\cdot\hat{\mathbf{n}}_b$ is the normal distance between an arbitrary molecule i and boundary b.

These distributions may be determined by: (1) measuring them from a separate equilibrium MD simulation at the target state-point[4] (see §3.7.2), or (2) numerical integration (e.g. using our MFBM model described below). We show some examples of these distributions for a Lennard-Jones liquid in Figure 4.19. Unlike the fully-repulsive force distri-

---

[4] The choice of the cut-off (r_{cut}) in the intermolecular potential affects the accuracy of measured distributions. Depending on the state-point, errors of ~ 5 – 30% were observed between force distributions when measured from an MD simulation using in turn $r_{cut} = 2.5\sigma$ and $r_{cut} = 4.0\sigma$.

butions used in the common hybrid techniques, this force distribution (Figure 4.19(a)) resembles that in [38]. In particular, the boundary potential has an attractive part at separation distances equal to 1σ.

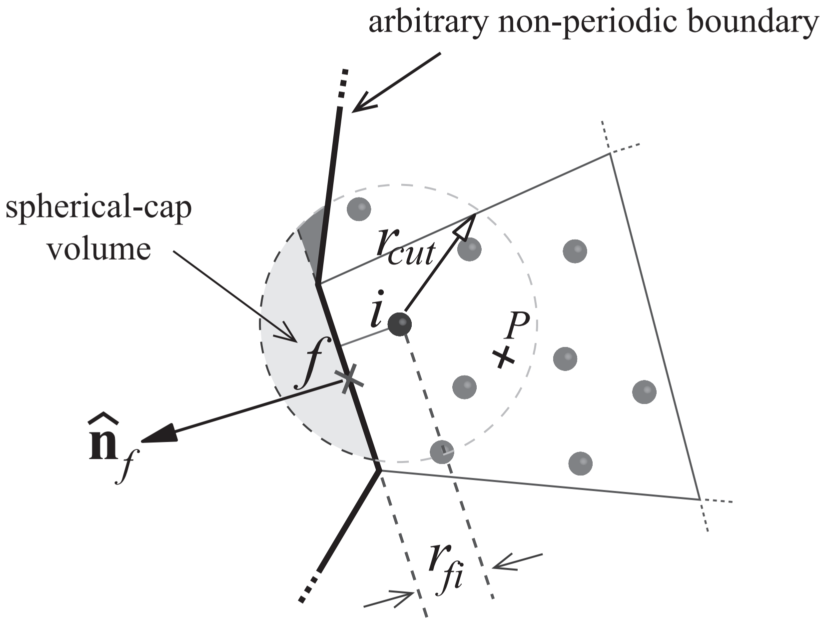


Figure 4.18: Schematic of the 3D simple force model. The imposition of a 1D external force distribution is applied to existing molecules within a non-periodic boundary cell P. The magnitude of the force is determined by computing the normal distance between the molecule's position and the closest face f on the boundary. The light shaded region defines the spherical cap over which the force distribution is integrated, while the dark shaded region shows the error in the model when applied to complex-shaped boundaries.

The 1D fields are then input to the code in a tabulated format, so that they are imposed during the MD run at a selected non-periodic boundary. In a sense, the force field acts as an MD potential between boundary-residing molecules and the boundary itself. At every MD time-step of the simulation, the following procedure is applied to all molecules within a boundary region of thickness r_{cut}. This procedure occurs immediately after the intermolecular force calculation step in the Leapfrog algorithm.

For a molecule i at $\mathbf{r}_i$ in a boundary cell P:

1. Find the closest boundary face f associated with cell P (see Fig 4.18). Note that each cell in the boundary region 'knows' its closest face on the boundary prior to the simulation run.
2. Compute the normal distance from the molecule to the face-centre, $r_{fi} = (\mathbf{r}_f - \mathbf{r}_i) \cdot \hat{\mathbf{n}}_f$.
3. Compute the missing force $\mathbf{f}_i^{miss} = -f_{bi}(r_{fi})\hat{\mathbf{n}}_f$, potential energy $U_i^{miss} = U_{bi}(r_{fi})$, and virial $w_i^{miss} = w_{bi}(r_{fi})$ contributions.
4. Update the molecule's acceleration $\mathbf{a}_i = \mathbf{a}_i - \mathbf{f}_i^{miss}/m_i$, potential energy $U_i = U_i + U_i^{miss}$ and virial $w_i = w_i + w_i^{miss}$.

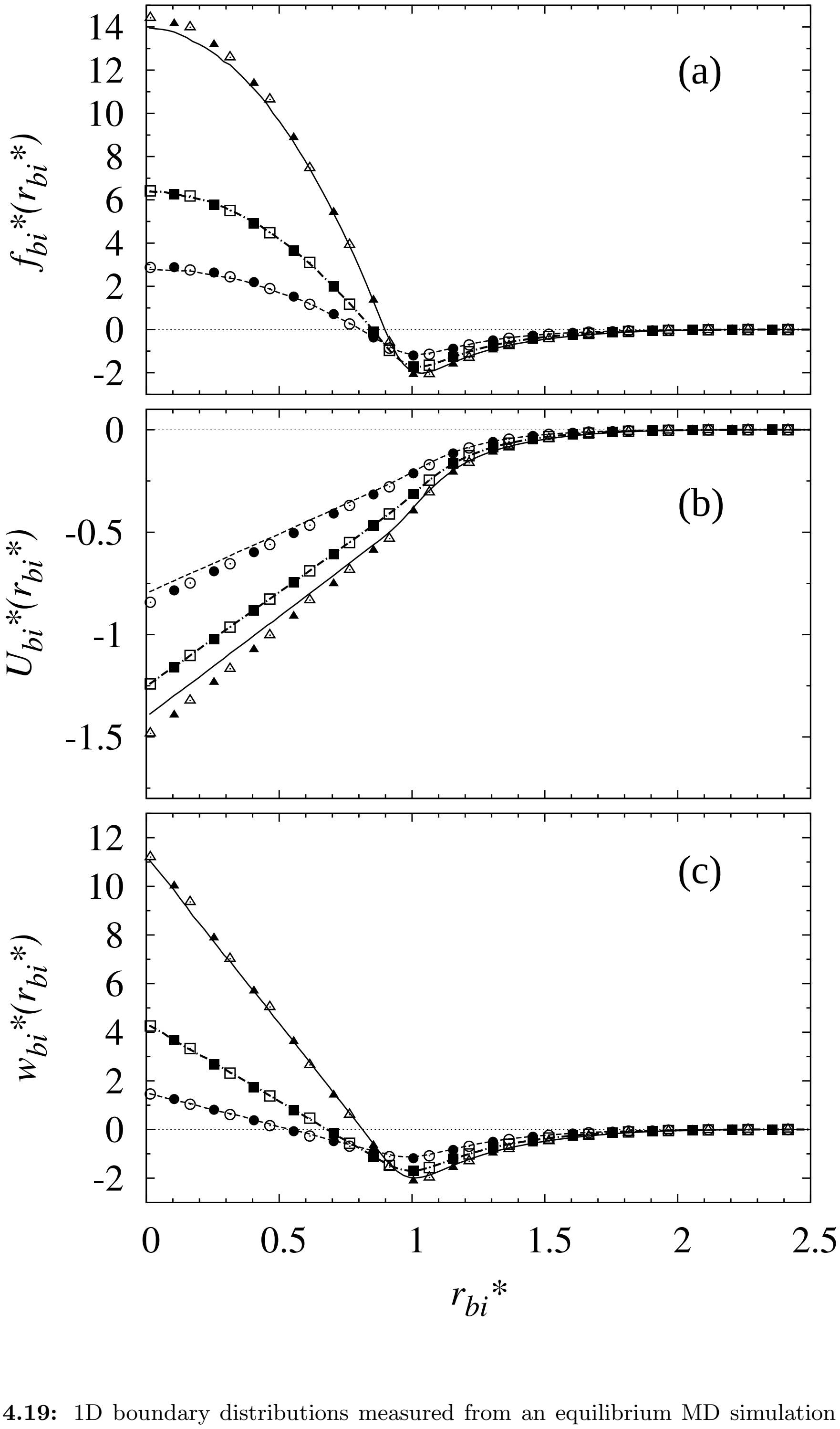


Figure 4.19: 1D boundary distributions measured from an equilibrium MD simulation at a fixed temperature, $T^* = 2.4$, for densities $\rho^* = 0.4$ (- - -), $\rho^* = 0.6$ (– · –) and $\rho^* = 0.8$ (——): (a) force, $\mathbf{f}_{bi}^*(r_{bi}^*)$, (b) potential energy, $U_{bi}^*(r_{bi}^*)$, and (c) virial, $w_{bi}^*(r_{bi}^*)$, missing contributions. The results obtained from our numerical models are superimposed on the graphs: open points ($\odot, \Box, \triangle$) for the mean-force boundary model (MFBM), and filled points (●, ■, ▲) for the mean-position boundary model (MPBM).

The advantages of this method is that it is relatively simple to implement, and computationally more efficient than the usual periodic boundary conditions, since it involves only one additional force computation per boundary molecule. However, the main drawback is that every time the state of the fluid at the boundary changes, a new MD simulation needs to be carried out in order to measure the new distributions. To our knowledge, no relationship or correlations so far exist between these distributions and the fluid state.

Significant computational savings may be achieved if this model is used for solid-fluid interactions; a solid wall defined by molecules may be excluded from a MD simulation run, and replaced by a force field that accounts for the structure of the missing solid wall. The setup is identical to the fluid-fluid implementation at a non-periodic boundary. To begin with, the 1D force, potential energy and virial distributions are measured normal to a planar molecular wall (of the desired lattice configuration), from a small MD system. Second, the wall-boundary is modelled by a specular-type boundary model (e.g., the stochastic boundary model), in addition to the boundary model described in this section. Since the method uses a local-face approach, the outer wall boundary of the channel may be of arbitrary geometry. In this scenario, a thermalisation process is also necessary at the boundary region to mimic heat transfer between the fictitious wall and the fluid. Several liquid-solid boundary potentials have been derived in the literature for various wall crystals (see for example Ref. [66] and references therein), which also may be input to this model.

Numerical mean-force boundary model (MFBM)

We propose a new numerical mean-force boundary model (MFBM) for arbitrary boundary geometries. The missing force on a boundary molecule i, reiterated from equation (4.2), is:

$$\mathbf{f}_i^{miss} = \int_{V_i^{miss}} \rho g(|\mathbf{r}|, \rho, T)\mathbf{f}(|\mathbf{r}|)dV, \tag{4.35}$$

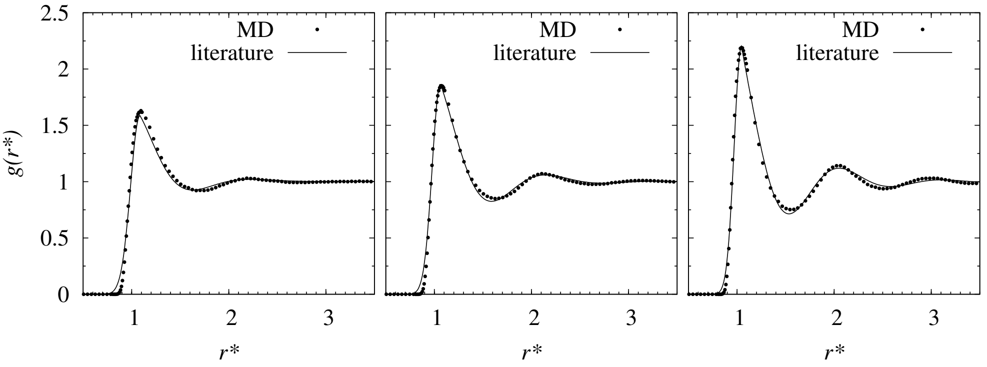
where $g(|\mathbf{r}|, \rho, T)$ is the radial distribution function that varies radially also as a function of temperature and density, and $\mathbf{f}(|r|) = f(r)\mathbf{r}/|\mathbf{r}|$ is the standard intermolecular LJ potential in vector form. The integrals for the missing potential energy and virial contributions are:

$$U_i^{miss} = \int_{V_i^{miss}} \frac{1}{2}\rho g(r, \rho, T)U(r)dV, \tag{4.36}$$

$$w_i^{miss} = \int_{V_i^{miss}} \frac{1}{2}\rho g(r, \rho, T)f(r)rdV. \tag{4.37}$$

Graphs for the radial distribution function $g(r, \rho, T)$ and the functions $s_f(r)$, $s_U(r)$, $s_w(r)$ in equations (4.35–4.37), are shown in Figures 4.20 and 4.21 respectively for different densities.

Our method consists of numerically integrating equation (4.35) using the Riemann



(a) ρ* = 0.4 (b) ρ* = 0.6 (c) ρ* = 0.8

Figure 4.20: Graphs for the radial distribution functions $g(r, \rho, T)$ at constant temperature $T^* = 2.4$, sampled from an MD simulation (•) using a cut off in the potential function of $r_{cut} = 2.5\sigma$. Comparisons are made with Ref. [1] (——).

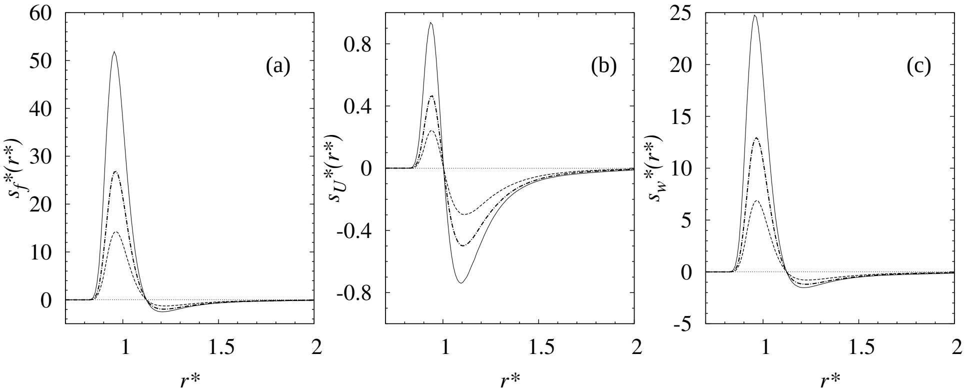


Figure 4.21: Graphs of the functions in the integrals (4.35), (4.36) and (4.37); (a) $s_f(r) = \rho g(r, \rho, T) f(r)$, (b) $s_U(r) = 1/2 \rho g(r, \rho, T) U(r)$ and (c) $s_w(r) = 1/2 \rho g(r, \rho, T) f(r) r$ respectively. Results are for the same densities and radial distribution functions $g(r, \rho, T)$ considered in Fig 4.20: $\rho^* = 0.4$ (- - -), $\rho^* = 0.6$ (–·–) and $\rho^* = 0.8$ (——).

sum technique[5] between an internal radius r_{min} and an external radius r_{cut} of a spherical shell, as seen in Figure 4.22(a). This is based on the fact that $s_f(r) = 0$ for $r < r_{min}$ due to $g(r, \rho, T) = 0$ (see Figure 4.20), and $s_f(r) = 0$ for $r > r_{cut}$ due to the shifted intermolecular force potential $f(r)$. The main spherical shell is discretised radially by a series of smaller spherical shells of thickness $\Delta r(r)$, and each shell is discretised uniformly into a series of nodes, $N_{nodes}(r)$. The partial force vector at an arbitrary node J located at a radial shell $r = r_{iJ}$ is,

$$\Delta \mathbf{f}_{iJ}(r_{iJ}) = \rho_P g(r_{iJ}, \rho_P, T_P) f(r_{iJ}) \Delta V_J(r_{iJ}), \tag{4.38}$$

where $\Delta V_J(r_{iJ}) = 4\pi r_{iJ}^2 \Delta r(r_{iJ})/N_{nodes}(r_{iJ})$ is the volume of node J, and ρ_P, T_P are the cell density and temperature. It is worthwhile mentioning here that the sum of all force contributions at each node should be zero, since $s_f(r)$ is radially symmetric. Each node also stores the potential energy and virial components:

$$\Delta U_{iJ}(r_{iJ}) = \frac{1}{2}\rho_P g(r_{iJ}, \rho_P, T_P) U(r_{iJ}) \Delta V_J(r_{iJ}), \tag{4.39}$$

$$\Delta w_{iJ}(r_{iJ}) = \frac{1}{2}\rho_P g(r_{iJ}, \rho_P, T_P) f(r_{iJ}) r_{iJ} \Delta V_J(r_{iJ}). \tag{4.40}$$

The computational sphere is constructed once, before the MD simulation starts. At each time-step of the simulation, the net missing force is computed on a boundary molecule i within a boundary cell P using the following procedure:

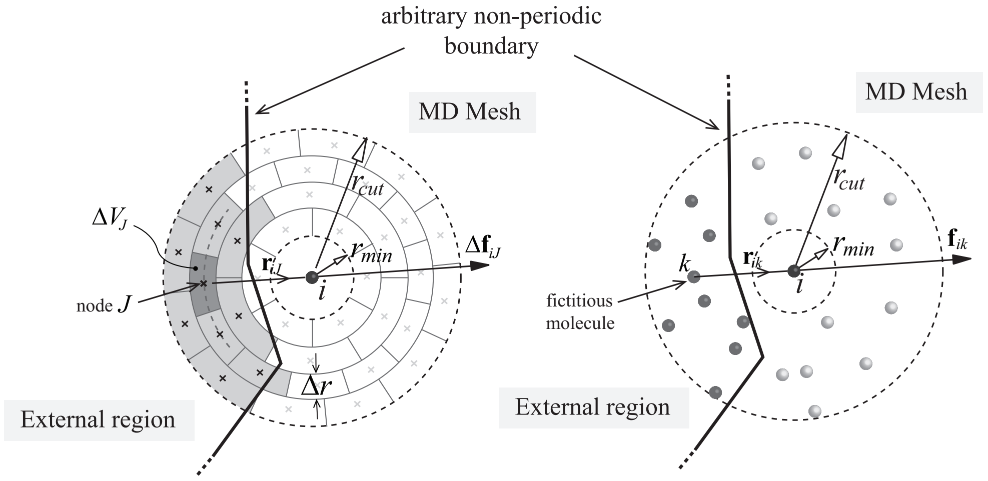
1. Make an inexpensive copy of the computational sphere and centre it on $\mathbf{r}_i$ (see Fig 4.22(a)).
2. Select only those nodes external to the domain N_{nodes}^{ext}.
3. Compute and update the net missing force, potential energy and virial over selected nodes using the sums:

$$\mathbf{f}_i^{miss} = \sum_{J=1}^{N_{nodes}^{ext}} \Delta \mathbf{f}_{iJ}(r_{iJ}), \quad U_i^{miss} = \sum_{J=1}^{N_{nodes}^{ext}} \Delta U_{iJ}(r_{iJ}), \quad w_i^{miss} = \sum_{J=1}^{N_{nodes}^{ext}} \Delta w_{iJ}(r_{iJ}). \tag{4.41}$$

The main disadvantage of this technique is its computational cost when applied directly to MD simulations. To achieve an accurate mean force distribution that resembles the distribution measured from an MD simulation over $\sim 400,000$ time-steps (see Fig 4.19), the computational sphere needs to be discretised into as many as ~ 4000 nodes. When using an optimum technique for distributing the nodes according to the topology of the $s_f(r)$ function, the number of nodes may be further reduced to ~ 400. Each boundary molecule

---

[5] The Riemann sum method for a volume integral: $\int_V F(x,y,z)dV = \lim_{n\to\infty} \sum_{k=1}^{n} F(x_k, y_k, z_k)\Delta V_k$, where $\mathbf{r}_k = (x_k, y_k, z_k)$ is a point within the bounds of a partial volume ΔV_k, and $V = \sum_{k=1}^{n} \Delta V_k$.



(a) Mean force boundary model (b) Molecular positions boundary model

Figure 4.22: Illustration of the proposed force field boundary models: (a) MFBM and (b) MPBM, applied to non-periodic boundaries of arbitrary geometries. In (b) only one real boundary molecule i is shown. The others are fictitious molecules and belong to the computational sphere of the model: the darker molecules are external to the MD domain, and interact with i, while the lighter molecules are internal and so do not interact with i.

therefore requires ~ 200 intermolecular force computations. Note that approximately half the computational nodes only lie external to the domain boundary. In comparison, a molecule in the bulk of a fluid of density $\rho^* = 0.6$ interacts with ~ 20 other molecules, using the minimum sum convention. The computational cost may be reduced if a mixed technique is adopted in which the MFBM is applied to rough boundary regions while the simple 3D model is applied to regions where the boundary is smooth.

We prefer using the simple 3D model at non-periodic boundaries because of its attractive computational cost, and also because it is impractical for so much processing power to be 'wasted' on a boundary condition over the entire MD simulation run. Nevertheless, this model may be used to compute numerically the 1D distributions $\mathbf{f}_{bi}(r_{bi})$, $U_{bi}(r_{bi})$, $w_{bi}(r_{bi})$, required by the simple 3D model, far cheaper than if they are measured from a prerequisite MD simulation. The radial distribution function $g(r, \rho, T)$, which is required by this model, may be determined from readily available correlations in the literature[6] (e.g., [1]). We reiterate that this model is applicable to arbitrary geometries, so it is possible to compute the force, potential energy and virial 3D fields next to complex boundaries.

The MFBM may be coupled with the 3D simple model in order to solve the issue of a variable state at the boundary. At regular time-intervals, the algorithm checks for changes of the state in a particular boundary cell. If the state exceeds a specific threshold, the algorithm computes the new 1D distributions using the MFBM numerical scheme

---

[6] The radial distribution function $g(r, \rho, T)$ obtained from literature sources must match closely that sampled from the MD simulation. Minute discrepancies in $g(r, \rho, T)$ as $g(r, \dots) \rightarrow 0$ get magnified due to the high repulsion and large gradient of the intermolecular function at small separation distances, resulting in mismatch errors of the force distributions up to 80%, closest to the edge of the boundary.

(typically taking a few seconds), and inputs these distributions to the 3D simple model.

Numerical molecular-positions boundary model (MPBM)

The molecular positions boundary model (MPBM) is proposed here as a computationally cheaper alternative to the MFBM method for arbitrary boundary geometries. It assimilates the periodic boundary conditions method, and the deformable boundary conditions proposed by Berkowitz and McCammon [53], by predicting fictitious molecules external to the MD domain to provide interactions with the real molecules residing in the boundary region.

The method is similar to our force boundary model, with the difference that the computational sphere is filled completely with fictitious LJ molecules using the radial distribution function and a random number generator. A sketch of this model is shown in Figure 4.22(b). The average number of molecules that are inserted within the sphere is:

$$\langle N_{mols} \rangle = 4\pi\rho_P \int_0^{r_{cut}} g(r, \rho_P, T_P) r^2 dr. \tag{4.42}$$

A sphere of randomly placed molecules is generated using the following technique:

1. Compute the total number of molecules to insert:

   $$N_{mols} = \mathrm{NINT}(\langle N_{mols} \rangle + \tilde{N}_{mols}), \tag{4.43}$$

   where $\mathrm{NINT}(x)$ is the nearest integer function, since only whole molecules may be inserted, and $\tilde{N}_{mols}$ is a fluctuating component of zero mean and variance $k_B T_P/m$.

2. Radially distribute molecules using the probability density function based on $g(r, \rho_P, T_P)$:

   $$p(r) = \rho_P g(r, \rho_P, T_P) V_{shell}(r), \tag{4.44}$$

   where $V_{shell}(r) = 4\pi r^2 \Delta r(r)$. Graphs of $p(r)$ for different state points are shown in Figure 4.23. This step is implemented using the well-known Von Neumann rejection sampling technique [67]:

   (a) Find the maximum point p_{max} on $p(r)$ between 0 and r_{cut} and normalise the curve so that its values lie between 0 and 1, i.e.

   $$h(r) = \frac{p(r)}{p_{max}}. \tag{4.45}$$

   (b) Pick a uniform random variable r_1 between r_{min} and r_{cut} and find $h(r_1)$. This is done by selecting first a random variate ζ on [0,1], then computing $r_1 = \zeta(r_{cut} - r_{min}) + r_{min}$.

   (c) Pick a second independent random variate r_2 on [0,1].

(d) **IF** $r_2 \leq h(r_1)$ then accept r_1 as r_{ik}: the radius for inserting the molecule. **ELSE** reject r_1 and go to Step 2b.

3. Distribute molecules evenly in the other two spherical co-ordinates (θ, ϕ).

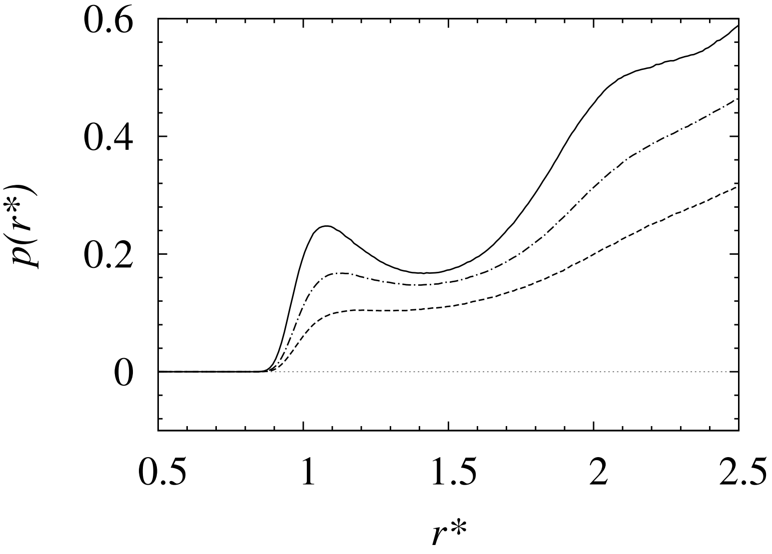


Figure 4.23: The probability distribution function $p(r)$ in equation (4.44), used to distributed molecules radially, for densities: $\rho^* = 0.4$ (- - -), $\rho^* = 0.6$ (- · -) and $\rho^* = 0.8$ (——). Temperature $T^* = 2.4$.

During the simulation a similar scheme to the MFBM is performed at each time-step; for each boundary molecule i:

1. Generate a sphere of fictitious molecules within a radius of sphere r_{cut} using the method above, and centre it at $\mathbf{r}_i$ (see Fig 4.22(b)).
2. Select only those N^{ext}_{mols} fictitious molecules external to the domain.
3. Compute and update the net missing force $\mathbf{f}^{miss}_i$ using the sum of the pair-intermolecular forces due to the selected missing molecules, as given in equation (4.1). The potential energy and virial are updated similarly to the periodic boundary conditions.

The main advantage of this model is that it has a similar computational cost to the common periodic boundary conditions, when applied in a real-time MD simulation with non-periodic boundaries. The major concern with this model is that there is no correlation between instantaneous configurations of existing molecules at the boundary, and the randomly generated set of fictitious molecules. Nor is there a correlation between one configuration and the next, at subsequent MD time-steps. Therefore, the forces applied to molecules, and hence the overall pressure transmitted at the boundary, are found to be over-compensated. Only in the limit of a large number of generated spheres is the averaged force distribution (and radial distribution function) identical to those sampled from an MD simulation (see Fig 4.19). The computational effort required to reach an accurate force distribution, although cheaper than the MD simulation itself, is still heavier than using the MFBM model.

4.6 Discussion

In this Chapter we have described the development of non-periodic boundary conditions (NPBCs) for MD simulations (see Chapter 5), and for coupled MD-continuum simulations (see Chapter 6). The models are applicable to arbitrary boundary geometries, monatomic LJ fluids (single and multi-species), and for parallel-processing. The boundary models have been divided into models of Type I and Type II — the former accounts for molecular collisions with the boundary wall, while the latter applies perturbations to molecules close to the boundary.

The state-controllers (Type II) constrain macroscopic properties within distinct cells of the mesh, including density, velocity, temperature and pressure, which may be used to initialise accurately the local properties of an MD case prior to the main simulation run. However, our main objective is to apply them solely at boundary regions to act as a Dirichlet-type (state-based) boundary condition on the MD domain. The target properties of the boundary may also be set from a continuum formulation in a hybrid MD-CFD simulation.

The existence of thermal fluctuations in MD simulations implies that numerical techniques for measuring properties in the feedback loop of the proposed controller algorithm may source a statistical uncertainty. The target (mean) hydrodynamic quantity is accurately converged within a control cell if averaging of the property has a low scatter about the statistical mean value. Effectively, this limits the size of the control cell, in which measurements are taken, as well as the time-averaging period, and consequently the time-characteristic of the target property. A coarse-graining method (§3.5.2) which measures properties from neighbouring cells has shown to meliorate these issues, except in specific cases, for example, where the spatial variation of the measured or target property is smaller than the size of the cell. In these cases, smaller control cells are required to smoothen out the fluctuations or impose the target properties accurately, and the coarse-graining method must be ignored. It is apparent that the problem of fluctuations and resolution (time and space) are strongly coupled, and further work is still required.

Computationally efficiency also play a role in these control procedures. In general, the intermolecular force step has shown to be more demanding than any of the controllers, save the density and pressure controllers when the target density is high. In cases where the noise is high, such as in low-speed flows, longer averaging times are required by the velocity controller to reduce the statistical scatter of measured velocity, which result in longer MD simulations.

Other boundary models have been developed mainly to ensure correct flow of molecular properties, and also to facilitate the imposition of the state-controllers. Some of the models we have considered are:

- mass flux boundary model (Type II) — to introduce molecules at the boundary of the domain at a prescribed rate;
- stochastic wall model (Type I) — to confine molecules within the domain and collide molecules randomly, selecting a velocity proportional to the target temperature;
- force-field model (Type II) — to account for the excluded molecules outside the domain.

A significant point about our work described in this Chapter is that we have implemented a general framework so that future scientists can contribute to the MD code with relative ease. A non-exhaustive list of future work includes:

1. The extension of the NPBC description (i.e., all boundary models) to polyatomic systems.
2. The development of more complex control-systems, e.g. the Proportional Integral Derivative (PID) control system [68], in our state-controller models. In addition to a term proportional to the difference between the target and measured property, the PID includes also an integral term that corrects for the accumulated offset of the error during the simulation run, and a derivative term that accounts for the rate of change of error.
3. The development of new strategies that solve the issues due to the interplay between statistical uncertainty, resolution and computational efficiency.
4. The implementation of more complex flux-based boundary models.

Chapter 5

Results

> *Insanity: doing the same thing over*
> *and over again and expecting different results.*
> - Albert Einstein.

5.1 Verification studies

In this section we demonstrate and verify the application of our non-periodic boundary conditions (NPBCs) in MD simulations of relatively simple geometries and systems.

5.1.1 Test of NPBCs

We replace periodic boundary conditions in one direction of a cubic domain (see Fig 5.1) of dimensions $(x, y, z) = (50\sigma, 20\sigma, 20\sigma)$ by two independent NPBCs (A, B) at $x_A = 0$ and $x_B = 50\sigma$. Results from our non-periodic MD simulations are compared with the corresponding fully-periodic case. We consider three types of MD simulations: a fluid in equilibrium; a fluid undergoing 1D uniform flow; and a fluid under steady shear.

The NPBC consists of an arbitrary boundary region which is divided uniquely into a buffer and controlling region. In the controlling region, temperature, density and velocity control is applied using our proposed models (§4.4). In the buffer and at the boundary, the following models are applied: (a) the simple 3D force model (§4.5.5); (b) the uniform mass flux model (§4.5.4); (c) the stochastic thermal wall (§4.5.3); and (d) the stochastic Maxwell-Demon method (§4.4.5).

Equilibrium simulation

We first consider the equilibrium MD simulation, in which target properties of temperature, density and velocity at both NPBCs are set to $T^* = 2.4$, $\rho^* = 0.6$, and $\mathbf{u}^* = (0, 0, 0)$. In this particular case, the mass flux is zero. We compare density profiles of properties in the x-direction for three separate cases using: fully-periodic boundary conditions, semi-periodic boundary conditions (only specular-reflective walls are applied) and the

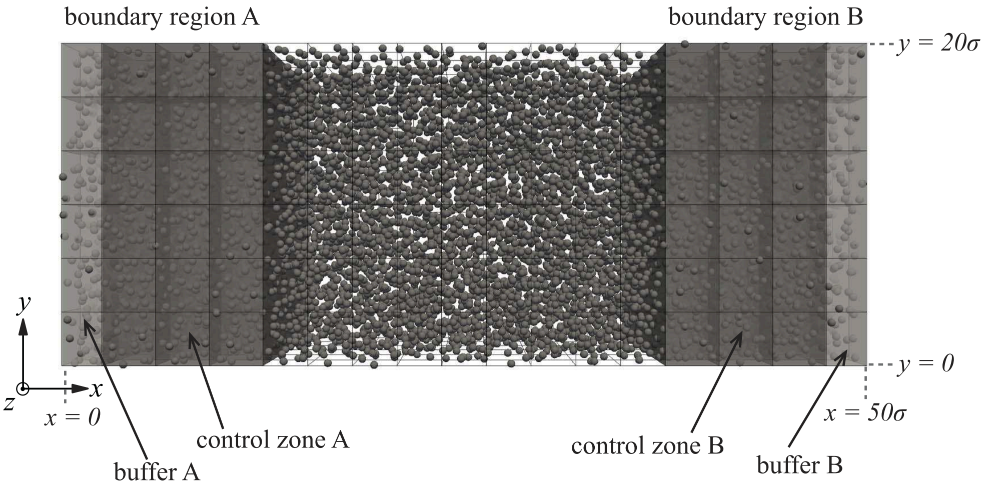


Figure 5.1: Test case of a simple cubic domain in which periodic boundary conditions are applied in the $y-$ and $z-$directions, while our non-periodic boundary conditions (A, B) are applied in the $x-$direction. The case setup is used for simulating equilibrium, 1D uniform flows and 1D shear flows.

non-periodic boundary conditions described above. The results in Figure 5.2 show that conformity is achieved in the control region and central region of the domain between the fully-periodic case and the NPBCs that employ the controllers. In addition, the density and pressure oscillations that occur due to the finite-size effects of the specular wall boundary are mostly rectified, and anyway are confined solely within the buffer region. Furthermore, we see in Figure 5.3 that external perturbations imposed within the boundary region do not influence the liquid structure or its dynamics in the central part of the computational domain: radial distribution and velocity autocorrelation functions sampled from both the fully-periodic and non-periodic cases are compared.

1D uniform flow

In the uniform flow case, a mean velocity of $\mathbf{u}^* = (-0.5, 0, 0)$ is applied at both boundaries, while other target properties of the equilibrium case remain unaltered. The mass flux imposed at each boundary of the domain is approximately 56 molecules per pico-second. At the outlet boundary ($x_A = 0$) the mass flux per boundary face is $\dot{m}_f^* = 0.6(-0.5, 0, 0) \cdot (-11.11, 0, 0) = 3.33$, while at the inlet boundary ($x_B = 50\sigma$), the mass flux is $\dot{m}_f = -3.33(\sqrt{\epsilon/m\sigma^2})$. Fluid properties in the streamwise direction are averaged over a time-period of $\Delta t^* = 2000$ and are presented in Figure 5.4. We see that our NPBCs effectively control the state at the boundary, even if the imposed flow rate is large.

1D shear flow

A 1D shear flow case is simulated using the same setup as in Figure 5.1. Two sets of shear rates are applied by controlling velocities in the boundary regions (A, B): (1) $\mathbf{u}_A^* = (0, 0.5, 0)$, $\mathbf{u}_B^* = (0, -0.5, 0)$; and (2) $\mathbf{u}_A^* = (0, 1.0, 0)$, $\mathbf{u}_B^* = (0, -1.0, 0)$. Results

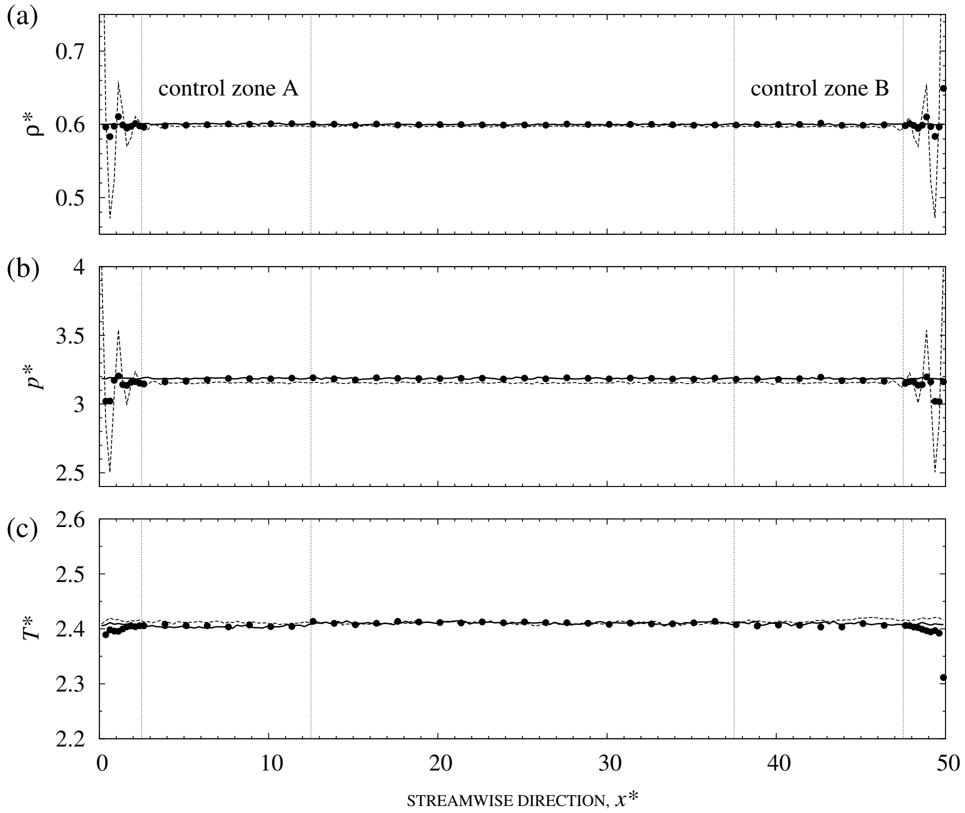


Figure 5.2: Equilibrium MD simulations showing profiles for (a) density, (b) pressure, and (c) temperature. Comparisons are made for periodic boundary conditions (——), specular-reflection boundary conditions (- - -) and our proposed non-periodic boundary conditions (•). Large density and pressure oscillations that occur next to the specular-wall are minimised with the incorporation of the mean force field boundary model, the stochastic boundary model and the Maxwell-Demon technique. Furthermore, the controllers maintain the correct state close to the boundaries, which act as an accurate Dirichlet-type boundary condition on the rest of the system. Consequently, the results between PBCs and our NPBCs agree well in the internal part of the domain.

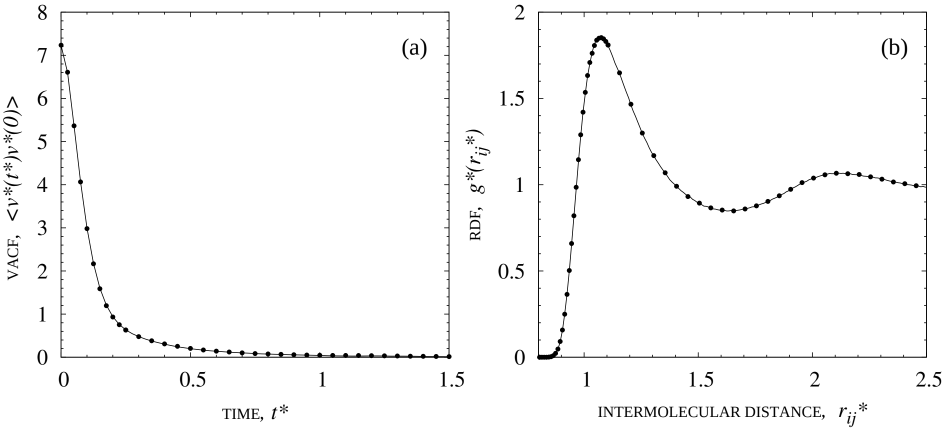


Figure 5.3: Comparisons between the fully-periodic (—) and our non-periodic (●) cases for an equilibrium MD simulation: (a) the velocity autocorrelation function (VACF) and (b) the radial distribution function (RDF).

of the steady-state velocity profiles are shown in Figure 5.5(a). We compute the shear viscosity, η using the linear relationship, $\eta = -P_{xy}/\dot{\gamma}$, where $\dot{\gamma} = du_y/dx$ is the strain-rate, and P_{xy} is the shear stress component of the stress tensor, equation (3.29), see Figure 5.5(b). Both quantities $(P_{xy}, \dot{\gamma})$ are sampled from this NEMD simulation over 200,000 time-steps. The shear viscosities calculated using this technique are reported in Table 5.1, and are compared with the value obtained from an equilibrium MD simulation[1] (2,000,000 MD time-steps) using the Green-Kubo relation, equation (3.28), and Ref. [7].

Table 5.1: Comparisons of shear viscosity calculated from non-equilibrium MD (NEMD) simulations (Figure 5.5) and measured from equilibrium MD (EMD) simulations (equation (3.28)), at density $\rho^* = 0.6$ and temperature $T^* = 2.4$. In the EMD simulations a cut-off of $r_{cut} = 4\sigma$ is used, in order to increase the accuracy of the measured viscosity.

Strain rate $\dot{\gamma}$ $(\sqrt{\epsilon/m\sigma^2})$	Shear stress P_{xy} (ϵ/σ^3)	Shear viscosity η $(\sqrt{m\epsilon}/\sigma^2)$		
		NEMD	EMD	Ref. [7]
0.036851	0.0328462	0.8913	0.977	0.919
0.0734834	0.0662877	0.902		

5.1.2 Eliminating a solid molecular-wall: liquid-wall NPBCs

In relatively large and complex MD nano-channel cases containing flows of simple liquids, it is more practical to completely eliminate the wall molecules in order to gain on the computational efficiency of the simulation. The effects of the wall on the liquid may be

---

[1] The equilibrium viscosity may also be obtained by extrapolating to zero shear the viscosities obtained from the non-equilibrium simulations.

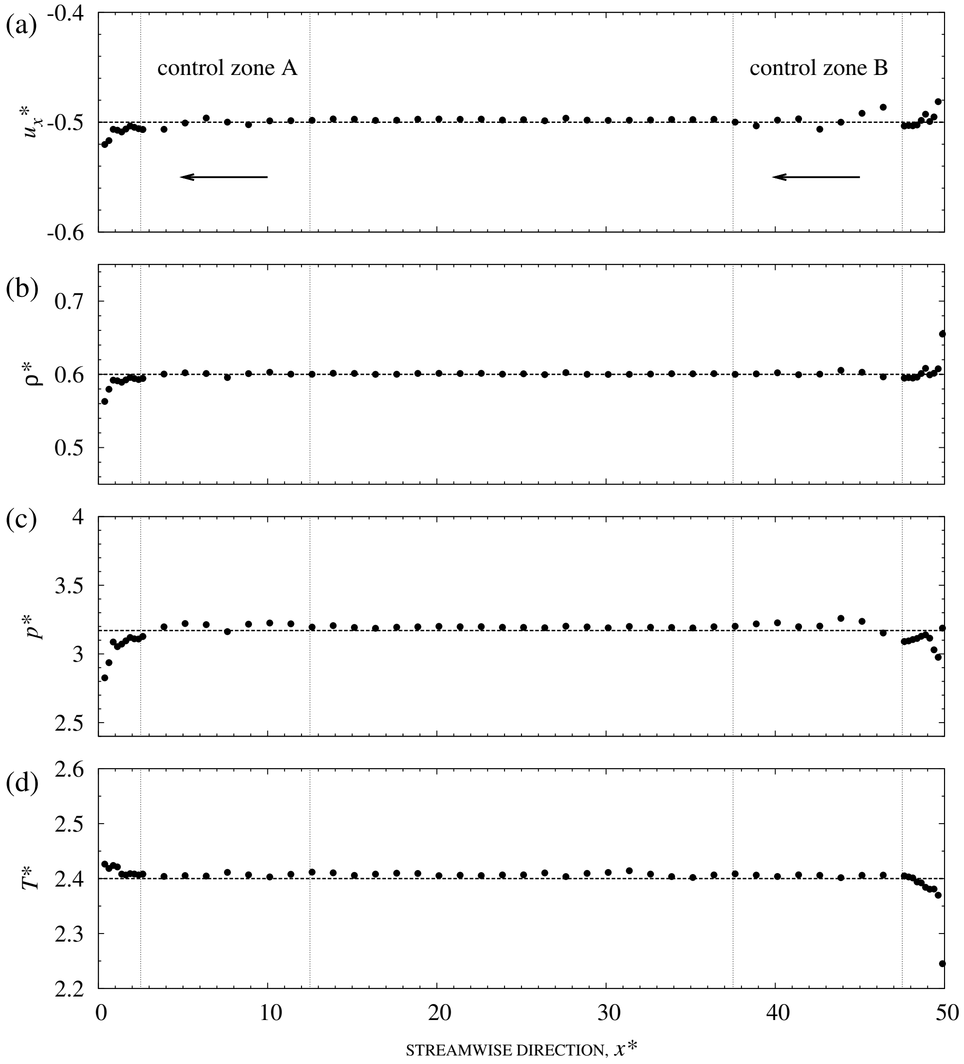


Figure 5.4: Results showing the profiles of (a) density; (b) velocity; (c) pressure; and (d) temperature (•) using our non-periodic boundary conditions for the uniform flow case; the flow is from right to left in the figures. The controllers and flux model, defined within the NPBC description, produce a non-fluctuating velocity and density profile in the central part of the domain, as observed in fully-periodic simulations. Slight fluctuations at the inlet and outlet buffer regions occur due to the numerical approximation of the mass flux model, such that existing molecules are diffusely reflected back into the domain when colliding with the wall, in addition to molecules inserted/deleted at regular well-defined intervals, instead of stochastically. These discrepancies are however minimised in the control zones, adjacent to the terminating buffer regions.

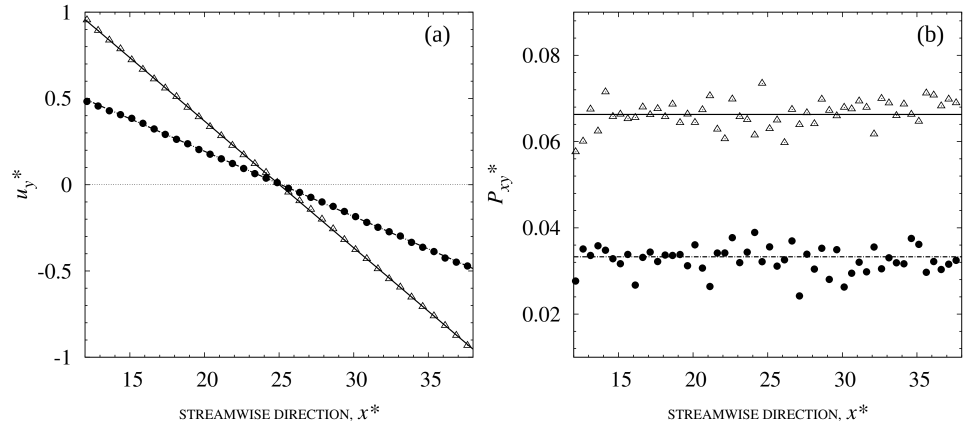


Figure 5.5: Profiles for (a) velocity (u_y^*) and (b) $x - y$ plane shear-stress (P_{xy}^*) in the streamwise direction (x^*) of the domain. For two shear rates, $\dot{\gamma}^* = 0.036851$ (●), and $\dot{\gamma}^* = 0.0734834$ (△). Linear fits are applied to all graphs.

replaced by an effective force field boundary model (§4.5.5) and stochastically reflective thermal wall model (§4.5.3).

To verify this, we first measure the 1D mean force, potential energy and virial distributions close to a planar molecular wall from a fully-periodic MD simulation. Figure 5.6 shows distributions taken at an FCC wall ($\rho^* = 0.8, T^* = 2.4$) for two cases of different wall-wall and wall-liquid interactions, *viz.* (a) Lennard-Jones (LJ) and (b) Weeks-Chandler-Anderson[2] (WCA) potentials. Subsequently, the wall molecules are removed, and replaced by NPBCs at the ends of the domain. The NPBC consists of: (a) the 3D simple force field boundary model (§4.5.5) utilising the measured distributions in Figure 5.6, (b) the stochastic wall boundary model at zero wall velocity, and (c) the Maxwell-Demon model in the buffer region. We also include a temperature controller in a small controlling region adjacent to the boundary surface, to mimic constant temperature walls. Control of mass flux, density or velocity properties are not included in this description of liquid-wall NPBCs.

Results comparing profiles between an explicit MD wall and our NPBC model, are shown in Figure 5.7. We verify that the application of the implicit NPBC wall boundary conditions reconstructs the average density and pressure variations occurring in the vicinity of the walls, in addition to good agreement in the bulk of the domain.

Although results are presented for simple planar boundaries, this wall boundary model may additionally be applied to arbitrary boundary geometries as we show in §5.2.2. In small nano-pores ($< 8.0\sigma$) [69], modelling the structure of the wall using molecules becomes almost essential as the two walls are within interaction distance of each other [70]. Somers and Davis [69] show that as the channel width decreases, different magnitudes of peaks and number of molecular layers in the density distribution of the liquid result between

---

[2] The WCA potential is the repulsive part of the LJ potential with cut-off $r_{cut} = 2^{1/6}\sigma$.

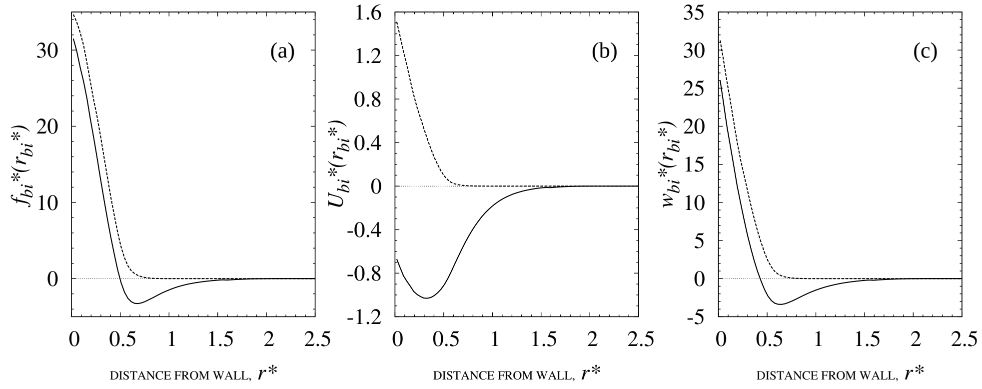


Figure 5.6: Results showing the 1D distributions for (a) force, (b) potential energy, and (c) virial, next to an FCC wall ($\rho^* = 0.8, T^* = 2.4$) in a fully-periodic equilibrium MD simulation. Wall molecules are tethered in space and a harmonic spring potential is applied between a tether point and its corresponding wall molecule, $U_h = 1/2K_s(\mathbf{r}_i - \mathbf{r}_i^{teth})^2$, where $K_s = 150(\epsilon/\sigma^2)$ is the spring constant. We show results for LJ (——) and WCA (- - -) potentials that have been chosen for the solid-solid and solid-liquid interactions.

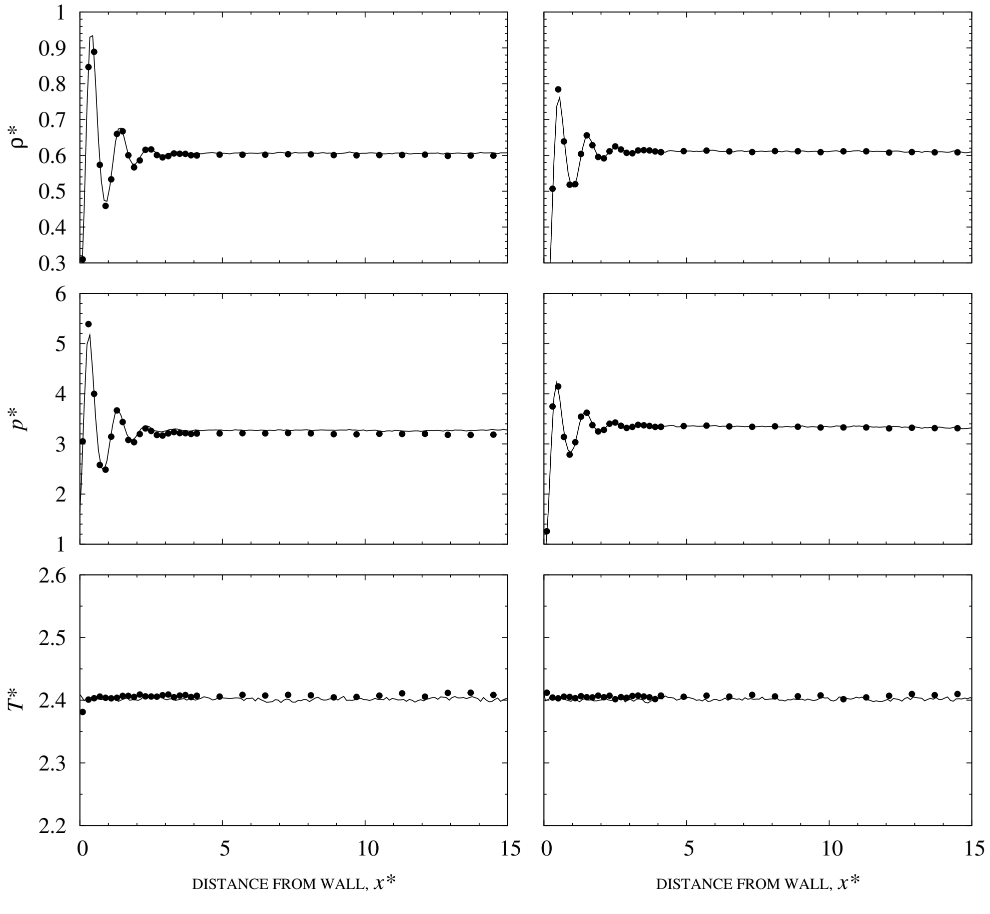
channel walls.

5.2 Liquid flows through nano-channels

In this section we show the capability of our NPBCs at inlets and outlets of nano-channels, in simple and complex MD geometries and in parallel. The method we propose, applies to general nano-channel configurations, and we demonstrate that it will serve as a useful tool in the engineering design of future nano-scale applications.

5.2.1 2D Poiseuille flow

Molecular dynamics' methods for simulating Poiseuille-type flows in simple 2D nano-channels have long been established (see Fig 5.8). Some examples of methods found in the literature were briefly described in §4.2. The most common technique is by far the gravitational method [48], in which periodic boundary conditions (PBCs) are applied in the flow direction and a constant gravitational force imposed on all fluid molecules. The moving wall method [49] operates by moving both top and bottom walls in the same direction at a steady-state velocity. Both methods maintain the same number of MD molecules, so that the fluid remains homogeneous in the longitudinal direction of the channel, that is, there are no pressure or density gradients across the domain [71]. The first method also imposes a tightly coupled thermostat, so as to overcome the heating introduced by the large gravitational forces. Consequently, it takes the system a very long time (∼ 4 ns) to reach steady-state. Furthermore, it requires a tedious trial and error procedure to determine the correlation between the imposed gravity force and a reference velocity, for example in the centre-line of the channel. The second method is clearly unphysical in its representation,



(a) LJ

(b) WCA

Figure 5.7: Comparisons of profiles for density ρ^*, pressure p^*, and temperature T^* measured near an explicit molecular wall (——) and the proposed implicit NPBC for solid-liquid wall boundary conditions (●). Results are presented using wall-liquid interactions, as seen in Figure 5.6, for (a) LJ and (b) WCA potentials.

even though it gives the same results. A technique for applying pressure gradients is the ‘semi-permeable reflective particle membrane’ [40], which constrains one of the PBCs to reflect molecules based on a specific probability. In this method, the pressure gradient is setup by fine-tuning the probability of reflection until the desired pressure gradient is achieved. A drawback of this method is that the pressure values at the inlet and outlet are uncontrollable, since they are an outcome of the simulation. These Poiseuille flow techniques, albeit computationally efficient and simple to implement, rely on the notion of periodicity, thus prohibiting simulations of more complex nano-channel systems.

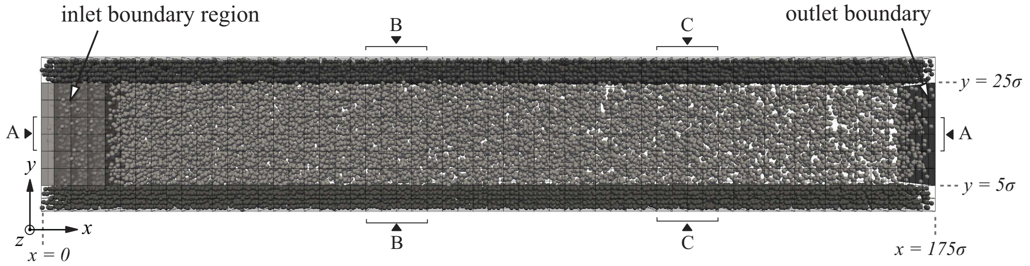


Figure 5.8: Snapshot of the MD case used for simulating Poiseuille liquid flow between two parallel walls (dark molecules). Regions (A, B, C) indicate the measurement zones.

The 2D compressible technique proposed by Sun and Ebner [2] (see §4.2.3) applies a relatively large reservoir (a quarter of the domain size) at constant density and temperature at the inlet by using a moveable piston to insert and compress molecules into the domain. The outlet boundary is described by an instantly-deleting model. Perhaps the main issues of this technique are that it may not resemble a specific MD ensemble, and it is based on the assumption that the outlet is connected to a vacuum. Consequently, the streamwise density gradient is set up because molecules are deleted at a faster rate at the outlet than they are inserted at the inlet, the effect of which propagates centrally into the channel until steady-state is achieved. The flux of molecules inserted in the source reservoir is therefore equal to the rate at which molecules exit from it. No temperature control is applied in the channel and therefore a temperature decline in the streamwise direction is also recorded. We also note that no control of velocity is applied at the inlet, that is, molecules are inserted at a zero mean velocity, which is not what the mean flow in the channel should be. Obviously, as molecules move within the bulk of the domain their speed increases [2], and therefore the channel must be made relatively long for the inlet effects to reach the bulk flow behaviour. The main differences between our work and that of Sun and Ebner are outlined below.

In our new method we apply Poiseuille flow with the values of the gradients and values at the inlets and outlets controllable. We simulate flow through the simple 2D long channel shown in Figure 5.8, which consists of a cuboid of dimensions $(x, y, z) = (175\sigma, 30\sigma, 20\sigma)$ with two parallel FCC walls of 5σ thickness and set at $(\rho^* = 0.7, T^* = 2.4)$. Solid-solid and solid-liquid molecular interactions are treated using the WCA potential, while the liquid-

liquid interactions are modelled by the LJ potential. The inlet non-periodic boundary conditions consist of a controlling region that sets the velocity ($\mathbf{u}^* = (0.5, 0, 0)$), density ($\rho^* = 0.6$) and temperature ($T^* = 2.4$) to fixed values at individual control cells. In addition, the boundary is modelled by a uniform mass flux model, with molecules inserted at a constant rate $\dot{m}_f = -3.33(\sqrt{\epsilon/m\sigma^2})$ at individual boundary face-areas, $A_f = 11.11\sigma^2$. The buffer region employs the Maxwell-Demon at the same target velocity and temperature with time-constant $\tau_T = 0.05(\sqrt{m\sigma^2/\epsilon})$. Finally, the simple 3D boundary model is applied to account for the truncation of the MD domain. These distributions are sampled from an MD simulation, as was outlined in §4.5.5: we choose the state point at ($T^* = 2.4$,$\rho^* = 0.6$) for the inlet, while at the outlet the state-point is roughly ($T^* = 2.4$,$\rho^* = 0.4$). In more practical simulations, the inlet boundary conditions may be imposed from fully-developed continuum profiles (velocity and mass flux), directly in a real-time hybrid simulation for unsteady flows, or indirectly from a steady-state solution.

The outlet boundary is modelled using the constant density mass flux model described in §4.5.4, instead of a fully-deleting 'vacuum' boundary condition. Molecules at the outlet boundary are deleted at a rate based on an adaptive criteria that ensures the zone density (here chosen as the entire domain) matches a prescribed phenomenological value, which in this case we choose to be $\rho^*_Z = 0.5$ (see equation (4.27)). Since ρ^*_Z is lower than the target density at the inlet, a density and pressure gradient is set up along the channel length. In this case, we assume isothermal internal conditions by applying the Berendsen thermostat in the entire computational domain with time-constant $\tau^*_T = 1.0$, at $T^* = 2.4$.

Results in the *streamwise* direction of the channel are shown in Figure 5.9. We carry out two different simulations to compare results with Ref. [2]. While both simulations apply our inlet non-periodic boundary conditions, we compare the results using the two different outlet boundary conditions. In particular, our adaptive model produces a linear steady-state gradient for pressure and density across the domain, which is expected for small pressure changes $\Delta p = p_{out} - p_{in}$. Conversely, the vacuum boundary condition produces a non-linear pressure and density gradient, which is probably due to the large dip in density it creates at the outlet boundary. A benefit of our technique is apparent from these simulations: both p_{in} and the pressure gradient ($p_{out} - p_{in}/L$), where L is the length of the channel, are controllable.

In Figure 5.9 we also compare the solutions obtained from the MD simulation using our NPBCs to those using a pure Navier-Stokes (N-S) CFD solver applied to the same case. The N-S solver (§6.4) uses a fixed viscosity ($\eta^* = 0.7438$) throughout the mesh, which is obtained from Ref. [7] using the state ($\rho^* = 0.52$, $T^* = 2.4$) in the central part of the MD domain. Additionally, we use an equation of state fitted from an equilibrium MD simulation at the same temperature (§6.4). Fixed-value pressure boundary conditions ($p^* = 1.5$) are applied at the outlet, while fixed-value velocity boundary conditions ($\mathbf{u}^* = (0.5, 0, 0)$) are enforced at the inlet in order to drive the flow. These values are both taken from the MD case. At the walls, no-slip velocity boundary conditions are applied.

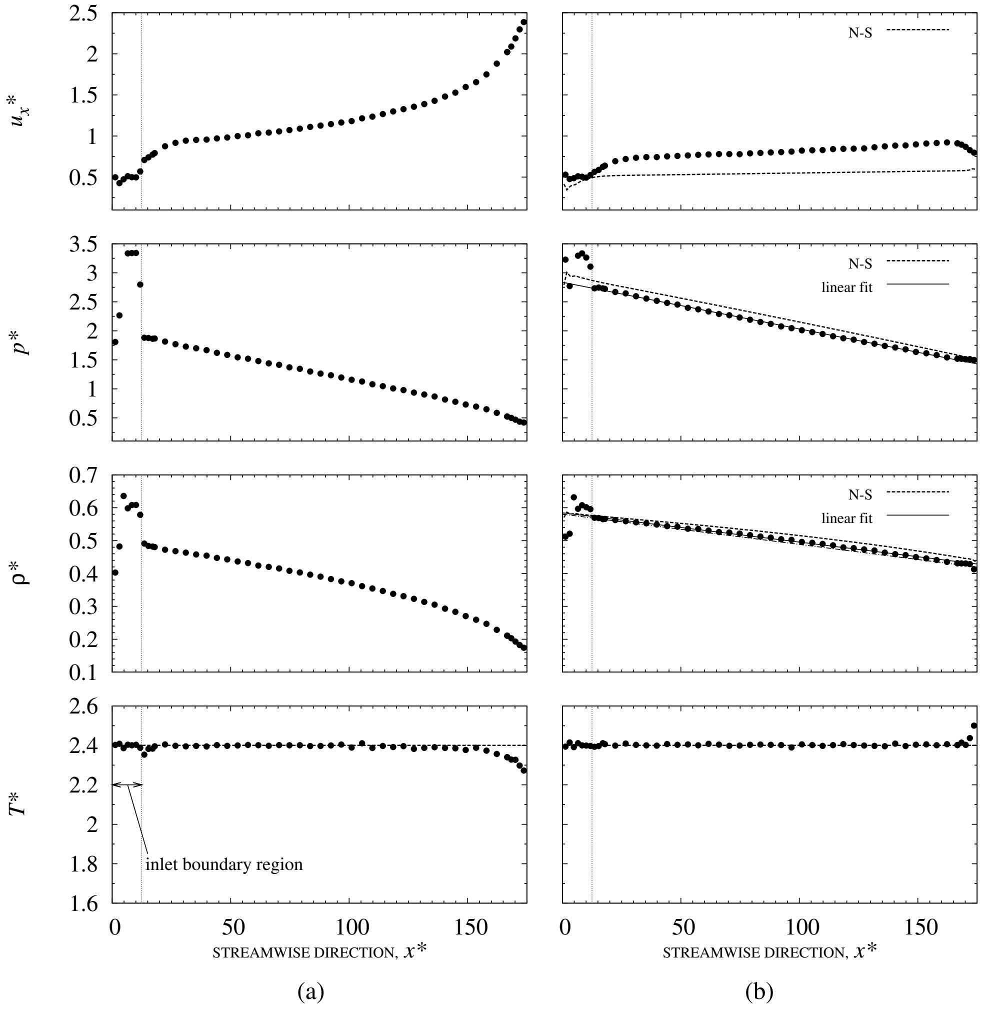


Figure 5.9: Centre-line profiles of velocity in the x-direction u_x^*, pressure p^*, density ρ^* and temperature T^*, taken from the sampling region A (see Figure 5.8): (a) the deletion vacuum boundary model used by Sun and Ebner [2] and (b) our constant-density mass flux model. In the latter case, results show that the density and pressure profiles are generally linear (—) and agree with the Navier-Stokes (N-S) numerical solution (- - -). Disagreement in the velocity profile is due to the no-slip boundary condition employed in the N-S solution. Superimposed we show an estimate of the density profile (– · –) based on the assumption in our mass flux model that the average density in the system should be $\rho_Z^* = 0.5$, hence $\rho^*(x^*) = \nabla\rho^* x^* + C^*$, where $\nabla\rho^* = -9.2 \times 10^{-4}$ is the density gradient computed between the centre of the channel and the inlet, and $C^* = 0.5805$.

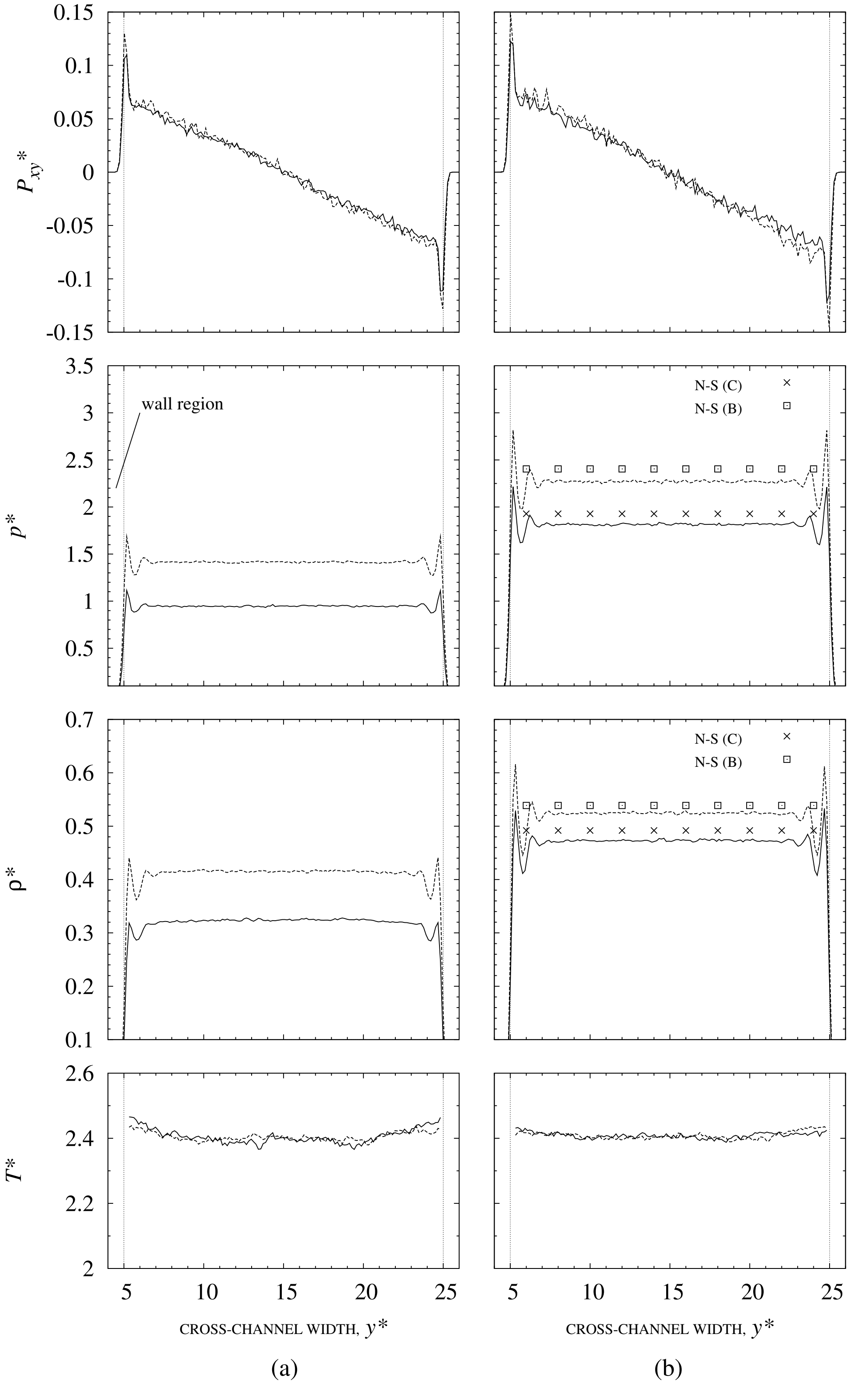


Figure 5.10: Cross-channel distributions (in the y-direction) for the shear-stress component in the $x-y$ plane P_{xy}^*, normal pressure p^*, density ρ^* and temperature T^*, taken from the sampling region B (- - -) and C (—), (see Figure 5.8). Figures (a) are for the deletion boundary model and Figures (b) are for our constant-density mass flux model.

Density and pressure profiles obtained from our MD simulation generally agree with the Navier-Stokes (N-S) numerical solution. However, there is a large disagreement between velocity profiles in the streamwise direction, which clearly indicates that the no-slip boundary condition in the N-S solution is not valid.

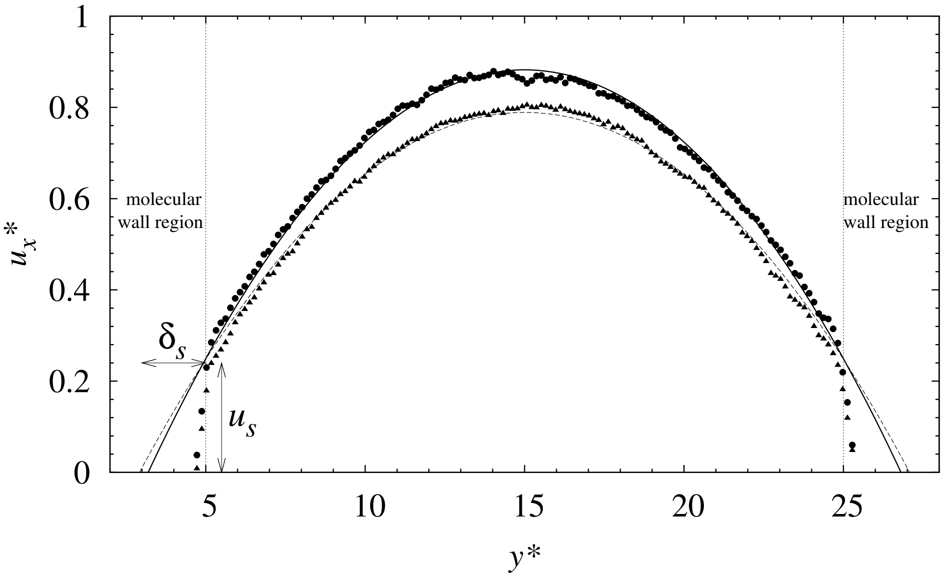


Figure 5.11: Cross-channel velocity profiles measured at streamwise locations B (▲) and C (●) of the nano-channel in Figure 5.8. We also show parabolic fits, using equation (5.1). The distance from the wall where the parabolic fits reach $u_x^* = 0$ defines the slip length δ_s, while the velocity at which the fits intersect the first wall layer (y_w^1, y_w^2), defines the slip velocity u_s.

Cross-channel profiles are also presented in Figures 5.10 and 5.11. The former figure shows that the discrepancies are minimal for pressure and density in the bulk. However, the continuum model does not capture the molecular layering phenomena occurring next to the channel walls. Figure 5.11 shows the velocity profiles at two streamwise locations (B,C) of the nano-channel. We superimpose the incompressible Navier-Stokes quadratic solution to the Poiseuille flow case, with microscopic slip, given by:

$$u_x(y) = -\frac{\Delta p}{\Delta x}\frac{1}{2\eta}(y - y_w^1)(y_w^2 - y) + u_s, \tag{5.1}$$

where $\Delta p/\Delta x = (p_{out} - p_{in})/\Delta x$ is the pressure gradient, η is the shear viscosity (which we take as varying only in the streamwise direction), (y_w^1, y_w^2) are the positions of the two walls, and u_s is the slip velocity. Equation (5.1) is used to compute the slip velocity u_s and slip length δ_s at different sections of the Poiseuille flow domain. These properties are defined in Figure 5.11 and recorded in Table 5.2.

In this section we have considered Poiseuille flow in a nano-channel with non-periodic boundary conditions applied at the inlet and outlet of the channel. Although a simple two-dimensional flow case is considered, our technique is generally applicable to arbitrary geometries as well to any number of outlets/inlets that the channel comprises (as we show

Table 5.2: Velocity slip, u_s and slip length, δ_s at two streamwise locations (B,C) of the nano-channel domain. The middle-channel densities are measured at (B,C), while the shear-viscosities are interpolated from tables in Ref. [7]. Quadratic fits using equation (5.1), plotted in Figure 5.11, give approximate values for u_s and δ_s, where the pressure gradient $\Delta P^*/\Delta x^* = -8.0478 \times 10^{-3}$ is obtained from the linear fit in Figure 5.9.

	Density ρ (σ^{-3})	Viscosity η ($\sqrt{m\epsilon}/\sigma^2$)	Slip velocity u_s ($\sqrt{\epsilon/m}$)	Slip length δ_s (σ)	Peak velocity u_{max} ($\sqrt{\epsilon/m}$)
B	0.52	0.7438	0.248	2.075	0.8
C	0.47	0.6355	0.249	1.803	0.87

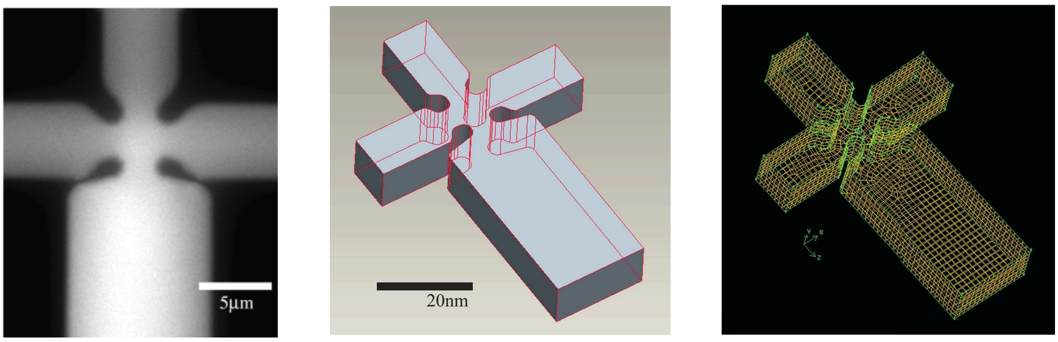
in the next section). It is to be noted that the techniques we propose may also be used for indirect hybrid methods in which MD derived properties (e.g., the slip velocities in Table 5.2) may be used as input to more computationally-efficient CFD codes. In this way, one can simulate larger nano- or micro-channels while preserving some level of microscopic detail next to the walls. Despite this, there is a lower-limit on the widths of the nano-channels in which continuum solutions may be reasonably applied. Travis and Gubbins [11] show that in smaller width pores ($\sim 5\sigma$), the velocity profiles deviate substantially from the quadratic N-S equation (5.1). Such profiles, which can have more than one maxima and minima, cannot be captured using solely continuum formulations.

5.2.2 3D complex-geometry mixing channel

We now consider a micro-scale mixing channel of three-inlet one-outlet design, taken from Hertzog *et al.* [3] (Fig 5.12(a)) and reduce its scale to nanometers so that reasonable mixing time-scales may be simulated using molecular dynamics. Using continuum numerical simulations and dye-quenching experiments the authors showed that the optimised mixer geometry is better than an older design [72] since it decreases the mixing time and hence improves its performance for protein folding.

Our modified nano-mixer geometry is constructed using the following route, which we generally adopt for other complex geometry cases. Initially the part is drawn in Pro/ENGINEER®, a commercial computer-aided design (CAD) drawing application (see Fig 5.12(b)), and exported in STEP format (.stp) to GAMBIT® (a meshing utility normally used for FLUENT® CFD). The geometry is meshed using hexahedral cells (see Fig 5.12(c)), and subsequently exported in mesh format (.msh) into OpenFOAM, where it is filled with molecules of two species, Fluid I and Fluid II, as shown in Figure 5.13.

The two fluids are essentially isotopes of argon; the Lennard-Jones potential is used for all I-I, II-I and II-II fluid interactions. Both fluids have identical properties but different identification (id) number, so that mixing can be observed and measured. The non-periodic boundary conditions at the inlet A (Fig 5.13), supply molecules of fluid I at a constant rate, $\mathbf{u}^*_A = (0, 0, 0.25)$, while inlets B1, B2 supply molecules of fluid II at the same rate: $\mathbf{u}^*_{B1} = (0.07, 0, 0)$, $\mathbf{u}^*_{B2} = (-0.07, 0, 0)$. These velocities were selected based



(a) (b) (c)

Figure 5.12: (a) An image of the original fabricated microscale mixer taken from Ref. [3], (b) part drawn in Pro/ENGINEER® at the nanoscale (using reduced units) and (c) part meshed in GAMBIT®.

on the parameter sensitivity guidelines for flow rate ratios described in [3]. The NPBCs at the inlets are of the same description as in the previous case for Poiseuille flow, with target density and temperature set to $\rho^* = 0.6$ and $T^* = 2.4$ respectively. At the outlet C, we similarly apply the constant-density mass flux boundary model that removes molecules of fluid I and II equally, based on the total zonal density $\rho^*_Z = 0.55$. A temperature Berendsen thermostat is applied in the whole domain to keep the local temperature close to $T^* = 2.4$.

The channel wall boundary is three-dimensional — no periodic boundary conditions are applied at all during this simulation. Instead of applying wall molecules, we use the boundary wall model outlined in §4.5.5 in order to reduce the number of molecules within the system. Distributions verified in §5.1.2 (see Figure 5.6), are applied at all faces of the wall boundary in addition to the implicit stochastic reflecting boundary model.

The mixing-channel case is decomposed and solved on eight processors for a duration of $t^* = 2500$. Transition from the initial configuration to an almost fully mixed fluid in the outlet of the channel occurs after approximately $t^* = 500$ of the simulation run. We see in Figure 5.14 that steady-state mixing is more complete in the central part of the outlet channel than at the sides close to the walls, where discrepancies of $\Delta\rho^* \approx 0.03$ relative to the mean (target) partial density are observed. Complete mixing may be achieved by increasing the length of the outlet channel substantially, in order to allow for diffusion to occur across the width. The geometry of the system may also be modified to improve mixing, for example by introducing one or more constrictions in the outlet channel and forcing molecules to mix again.

Mixing is generally dependent on the boundary conditions that are applied to the MD system, as was similarly noted by Hertzog *et al.* [3] for continuum simulations. Here we perform a trial-and-error procedure until the desired level of mixing is achieved. In general, the velocity imposed at inlet A should be significantly higher than the flow at inlets (B1, B2).

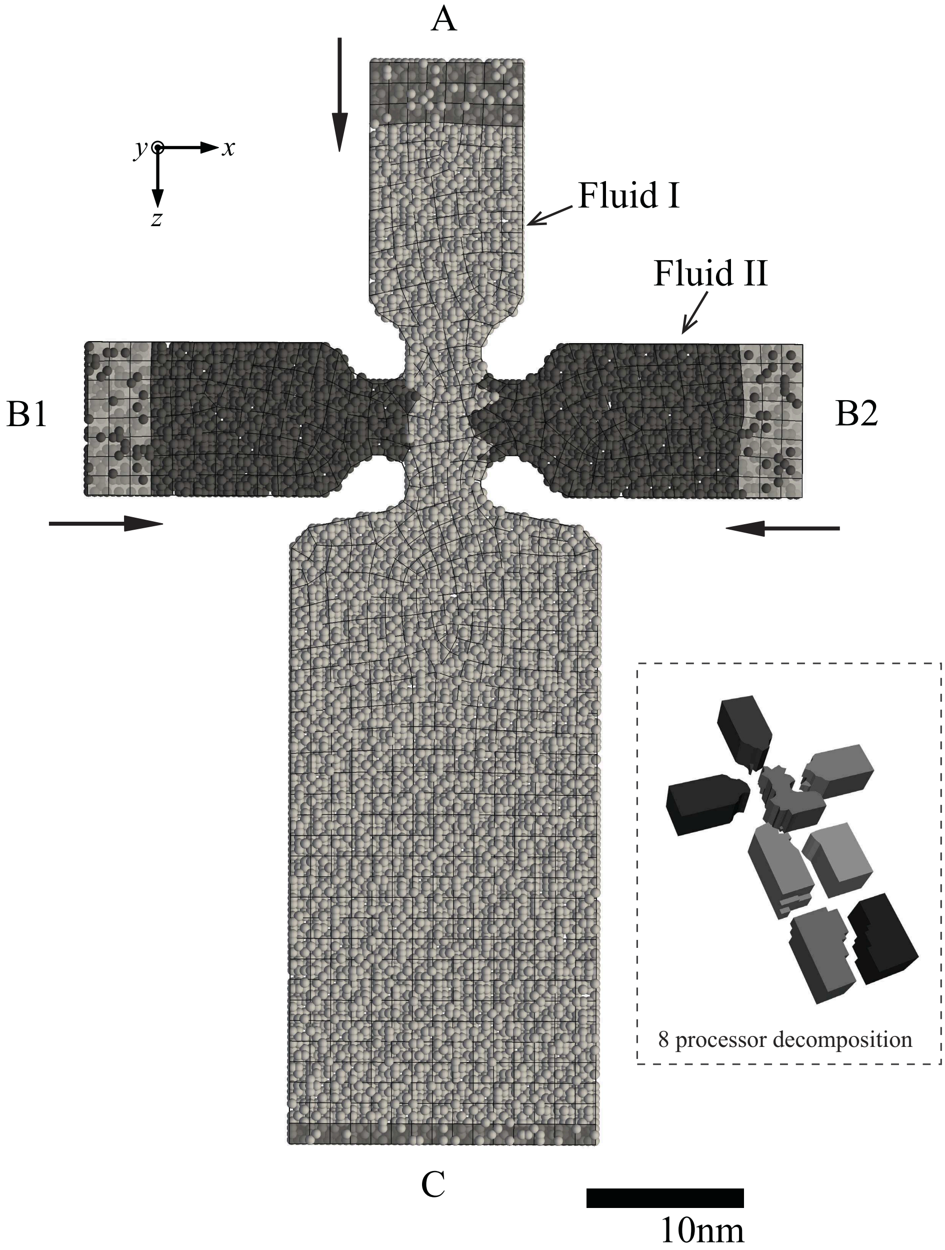


Figure 5.13: The nano-mixer MD case shown in its initial state. NPBCs are applied at all three inlets (A,B1,B2) and the outlet (C), depicted by the shaded regions. Note: only 30,000 random molecules are shown out of the ∼200,000 molecules that occupy the simulation domain. Inset: showing the domain-decomposition of the mesh on 8 processors.

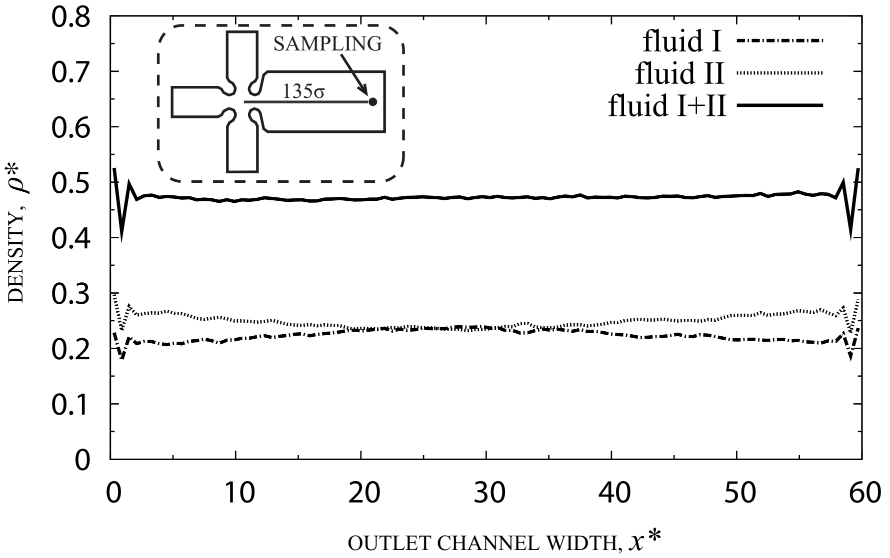


Figure 5.14: The partial-density cross-channel distribution for both fluids, taken in a sampling region at the outlet of the mixer.

The mixing-channel case is then repeated to model two fluids with different miscibilities. This is performed by modifying the interaction potential between fluid I and fluid II, which in this case was chosen to be the WCA potential. The WCA potential was also used for II-II interactions, while the LJ potential remains the same for I-I interactions. The boundary conditions of the previous case then had to be modified as blocking effects at the throat appeared. Interestingly, the WCA fluid passed preferentially through the throat if it came in at the two inlets, and similarly if the inlets of fluids I and II are switched. Consequently, only fluid II passed through the outlet channel and mixing was not established. For satisfactory mixing, the velocity of fluid I at inlet A had to be increased by four times, i.e. to $\mathbf{u}_A^* = (0, 0, 1.0)$, and the pressure gradient increased, using the property $\rho_Z^* = 0.5$ in the outlet mass flux model.

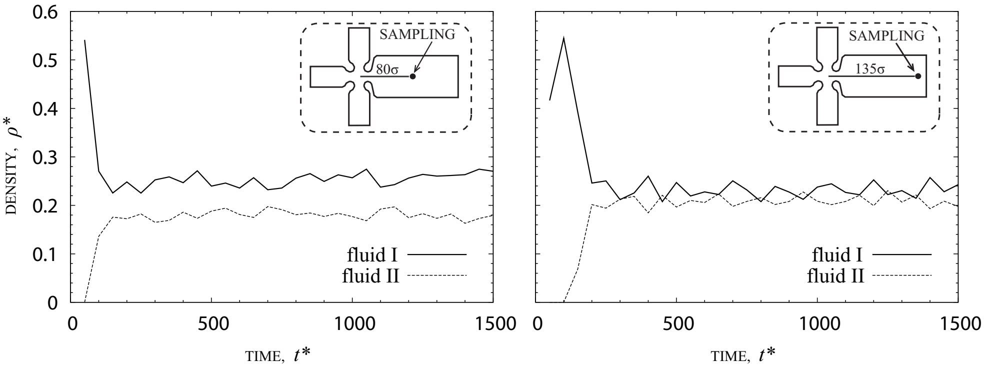
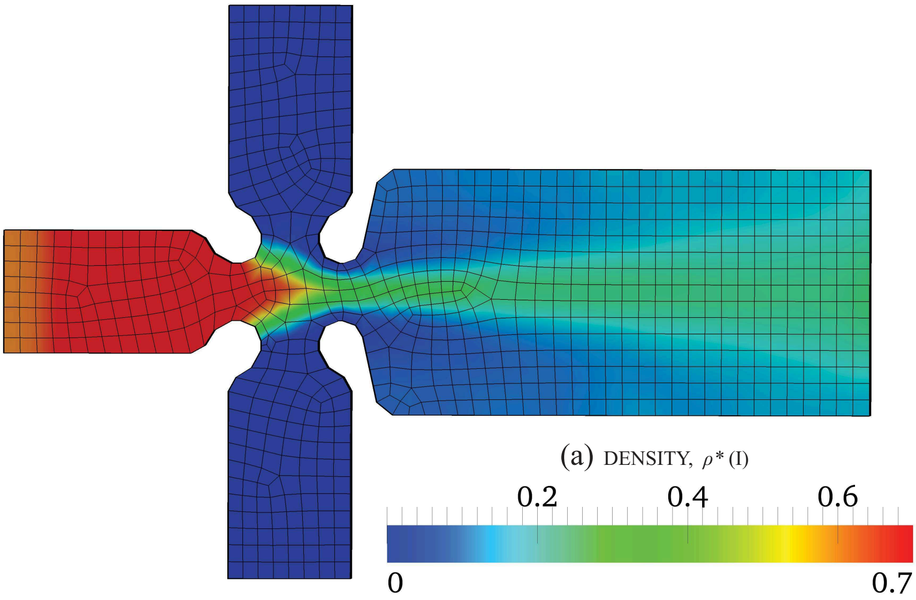
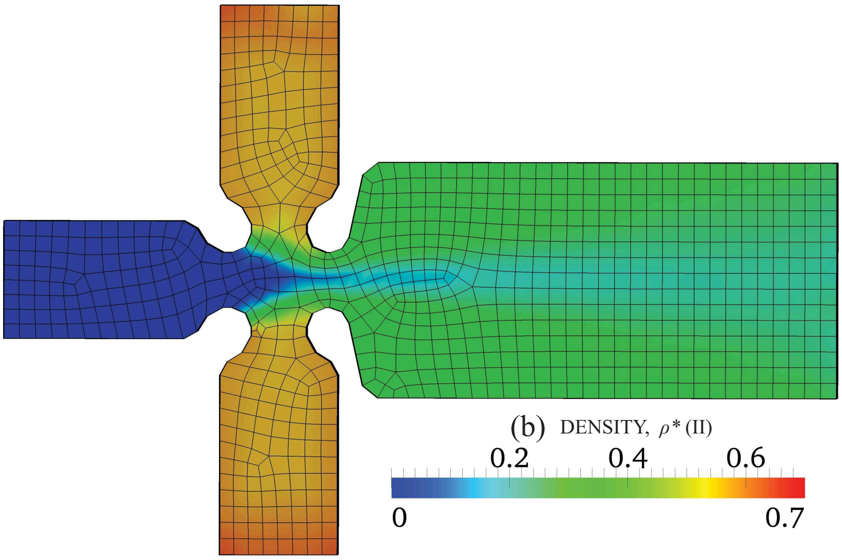


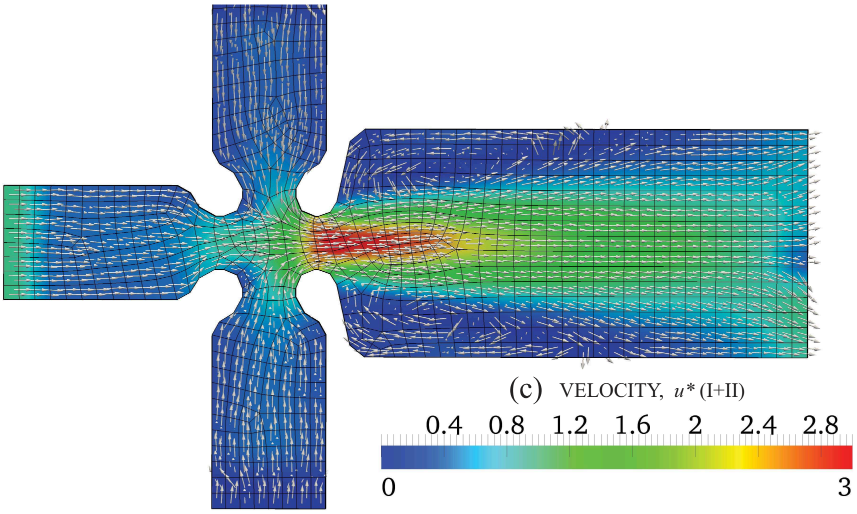
Figure 5.15: The time over which mixing occurs between the two fluids for the second case, taken from two sampling regions at the outlet channel of the mixer.



(a) DENSITY, ρ^* (I)



(b) DENSITY, ρ^* (II)



(c) VELOCITY, u^* (I+II)

Figure 5.16: Mid-channel-plane fields for the second case, showing (a) partial density for fluid I, (b) partial density for fluid II and (c) total velocity.

The transient mixing characteristics resemble that of the previous case study, as seen in Figure 5.15. Results of steady-state mixing are averaged over the preceding time period $\Delta t^* = 500$ of the full simulation run. Fields for density and velocity in a cross-section through the middle part of the channel are presented in Figure 5.16. In these figures we show that the throat geometry of the channel plays an important role in the mixing procedure. As is expected from this type of channel configuration, the flow velocity at the throat increases substantially, which in turn forces fluid molecules to mix locally. Furthermore, the shape and size of the outlet channel is key to cross-channel and longitudinal mixing. The partial densities of both fluids are sampled in a region close to the outlet, and shown in Figure 5.17.

MD simulations of these case studies were extremely computationally intensive, requiring approximately 2 weeks of processing time for each case when executed in parallel on 8 processors. Future investigations of mixing may involve carrying out sensitivity experiments to observe the effect of changing the geometry (e.g., throat width, outlet channel width and length) and boundary conditions (e.g., pressure gradients and inlet velocities) on mixing between two different fluids.

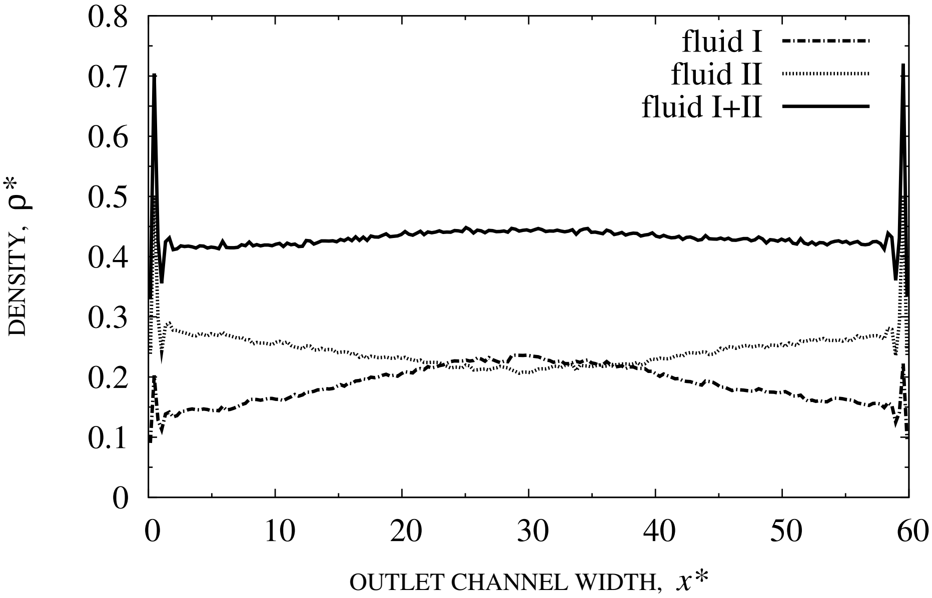


Figure 5.17: The partial-density distribution of both fluids for the second mixing case, taken from a sampling region across the outlet of the mixer.

5.2.3 Nano-droplet deformation in extensional flows

In this section we demonstrate the application of our non-periodic boundary conditions in order to investigate emulsion droplet deformation and breakup within extensional-type flow geometries. Applications of such studies are important in nanofluidic and microfluidic technologies such as heat-exchange devices, lab-on-a-chip components and ink-jet printing, in which the dynamic (Newtonian or non-Newtonian) behaviour of droplets needs understanding at a molecular scale and within arbitrary flow domain geometries.

The domain consists of a cross-shaped nano-channel as shown in Figure 5.18; one pair of opposing channels are set as inlet boundary conditions and the other pair of opposing channels are set as outlets. The premise of the simulation is to create an extensional shear flow at the throat of the channel in order to elongate, deform and breakup the three-dimensional viscous droplets that pass through the channel.

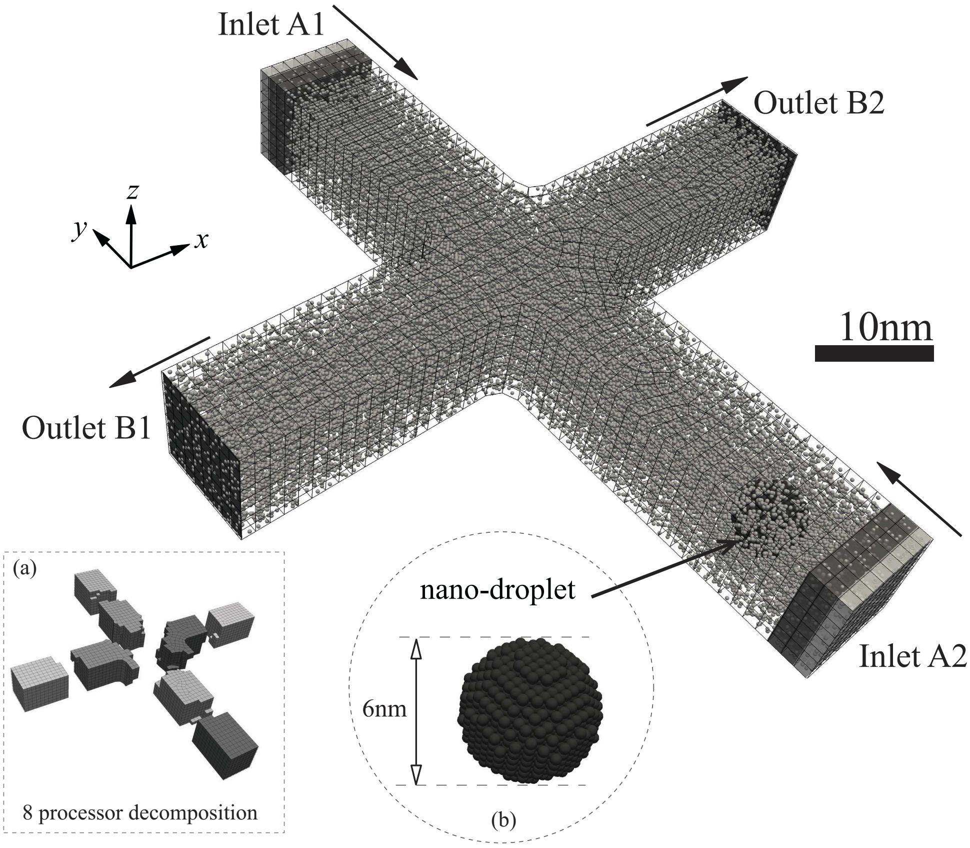


Figure 5.18: The cross-shaped nano-channel flow geometry. Extensional flow is induced at the throat of the channel by the application of inlet (A1, A2) and outlet (B1,B2) boundary conditions. Inset: (a) the 8 processor decomposition of the geometry, and (b) the nano-droplet (diameter $\sim$ 6nm) inserted at the boundary region of the inlet channel A2.

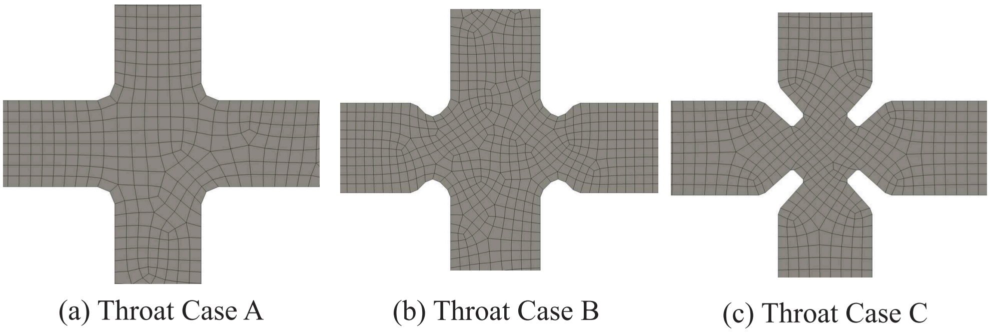
The interaction potentials are purposely chosen to prevent the molecules in the droplet (d) from diffusing into the solvent (s). A simple method to implement this is to use the repulsive WCA potential for the droplet-solvent pair-interactions with the following modified characteristics $\sigma^*_{sd} = 1.4$ and $\epsilon^*_{sd} = 0.8$. The droplet-droplet and solvent-solvent interactions are chosen to be the common shifted LJ potential with standard characteristics $\sigma^*_{dd} = \sigma^*_{ss} = 1.0$ and $\epsilon^*_{dd} = \epsilon^*_{ss} = 1.0$.

The geometry of the device is created through a similar route to the mixing channel above. Non-periodic boundary conditions at the inlets (A1,A2) consist of controlling the density at $\rho^* = 0.6$, temperature at $T^* = 2.4$, and velocity is constrained to point inwards to the domain: $\mathbf{u}^*_{A1} = (0, -0.25, 0)$ and $\mathbf{u}^*_{A2} = (0, 0.25, 0)$. At the outlets (B1,B2), two independent constant-density mass flux boundary models are applied at a value of density $\rho^*_Z = 0.57$, where Z is their respective outlet channel zone.

Droplets are inserted close to the inlet boundary A1 at a rate of 2.314×10^9 s^{-1}. The time between insertions is specifically chosen to prevent multiple droplets from interacting during their motion through the channel. A simple and effective method for inserting droplets is employed, and implemented using our controller framework: when the droplet is to be inserted, a reference point $\mathbf{r}_p$ is located at the inlet region, bounded by a user-defined cell-zone. All solvent molecules in the cell-zone are then tested and 'transformed' into droplet molecules only if $|\mathbf{r}_i - \mathbf{r}_p| < r_{drop}$ is satisfied, where $\mathbf{r}_i$ is the position of an arbitrary molecule i, and $r_{drop} = 9\sigma$ is the target droplet radius. The solvent-droplet transformation is implemented in the code by switching its 'id' number, thus assigning different interaction potentials to all droplet molecules. The process occurs instantaneously and the droplet then adjusts to the potential over a few time-integration steps.

Individual droplets travel through the inlet channel in a generally spherical shape, until they reach the throat of the channel. Droplets then deform, elongate and breakup as they exit the throat, mainly due to extensional forces in the throat. Breakup of droplets occurs when the local pulling forces acting on the droplet exceed the surface tension between droplet and solvent. As the droplets travel along the outlet channels, the forces on the fluid are no longer of an extensional type, and surface tension (due the highly repulsive droplet-solvent interactions) acts predominantly to return the droplet to a spherical shape. Deformation and breakup characteristics are generally dependent on a number of factors, such as droplet size, boundary conditions and geometry of the throat. In this thesis, we have only investigated droplet deformation caused by the geometry of the throats, and examined the three cases in Figure 5.19.

Simulations are carried out for a time of $t^* = 2000$ and solved on 8 processors. Each simulation took approximately 2 weeks of processing time. Results of the droplet deformations are shown in Figures 5.20 – 5.23 for the three cases. In the first case, we observed a random variety of droplet deformation characteristics, which can be described as follows. As the droplet approaches the throat, it deforms from a spherical shape to an elongated shape, roughly in the middle of the throat, with its length pointing in the direction of the outlets. The droplet either extends to a point where it then typically breaks up into two, three or four smaller droplets, or else is pulled entirely to one side of the channel, pulling with it the tail part of the droplet and eventually forming a spherical droplet again. The

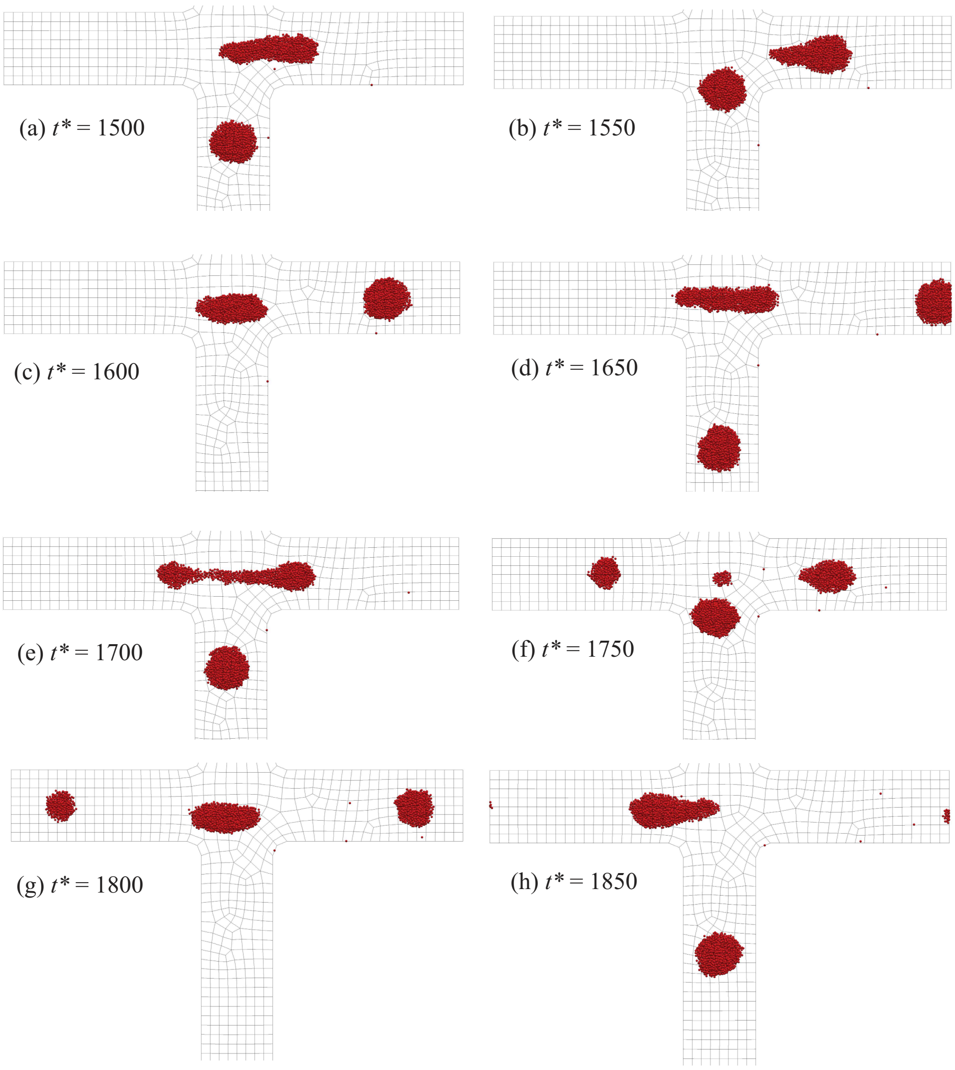


(a) Throat Case A (b) Throat Case B (c) Throat Case C

Figure 5.19: The three trial throat geometries used for the extensional droplet flow cases.

local stochastic velocity/force field is what determines which side the droplet chooses to travel. In the second geometry, the throat is narrowed within the side channels so as to constrict the flow, and obtain a more pronounced breakup. Although similar deformations are observed, more breakups of droplets occur than in the previous case. However, due to the presence of the constriction, some of the droplets tend to stick to the wall. In the near-wall region the velocity is low so molecules travel slower, generally reducing the speed of the droplet as it exits the channel. There is also the attractive part of the liquid-wall potential which prevents the droplet from being set free from the wall. In the final geometry, we reduce the width of the throat further and observe that droplets always break up into two smaller droplets. The drawback of this geometry however is that all the droplets now stick to the wall as they travel from the throat to the channel exit, an effect which is generated by the tight throat constriction.

The purpose of these simulations has been to demonstrate the potential applications of our non-periodic boundary conditions in arbitrary domain geometries. Extensive work may now be carried out using our methods in order to analyse droplet deformation with different fluid properties (e.g., to analyse the viscoelastic properties of polymeric droplets), and different domain geometries. Another useful study would be to optimise the design geometry of throat case C in order to get droplet breakup without the problem of sticking. In this regard it would be a better modelling assumption to replace the wall boundary potential with a more realistic molecular wall representation. Finally, results may be compared with a Brownian Dynamics simulation approach, in which the solvent molecules are excluded from the simulation, while droplet molecule motion is solved using random and dissipative forces, saving a significant amount of computational time.



(a) $t^* = 1500$

(b) $t^* = 1550$

(c) $t^* = 1600$

(d) $t^* = 1650$

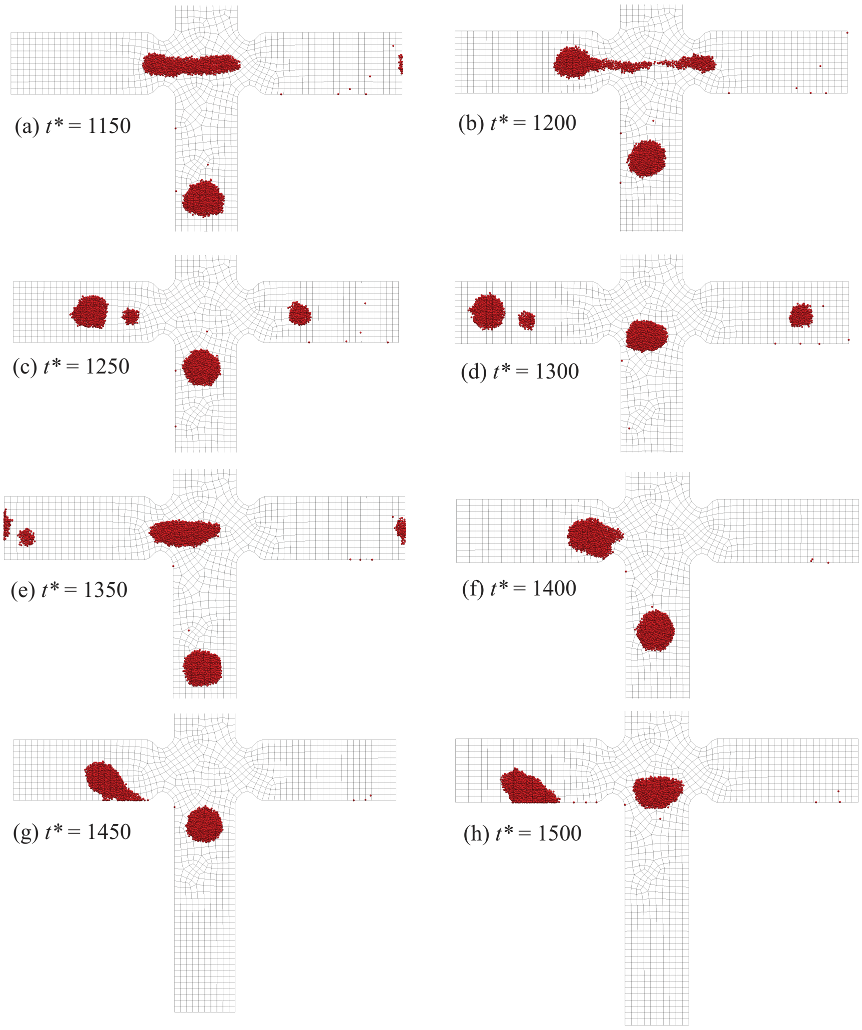
(e) $t^* = 1700$

(f) $t^* = 1750$

(g) $t^* = 1800$

(h) $t^* = 1850$

Figure 5.20: Droplet deformation snapshots in the extensional flow problem, for throat case A. Times range from $t^* = 1500 - 1850$.



(a) $t^* = 1150$

(b) $t^* = 1200$

(c) $t^* = 1250$

(d) $t^* = 1300$

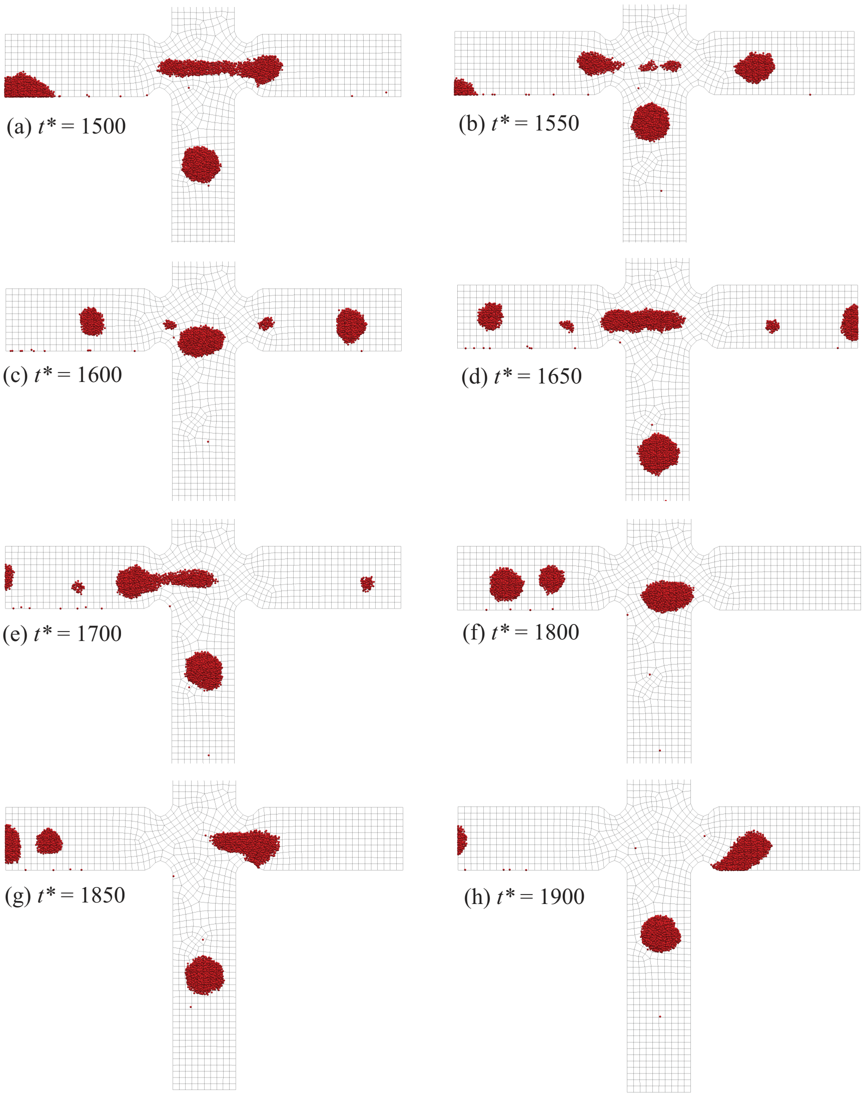
(e) $t^* = 1350$

(f) $t^* = 1400$

(g) $t^* = 1450$

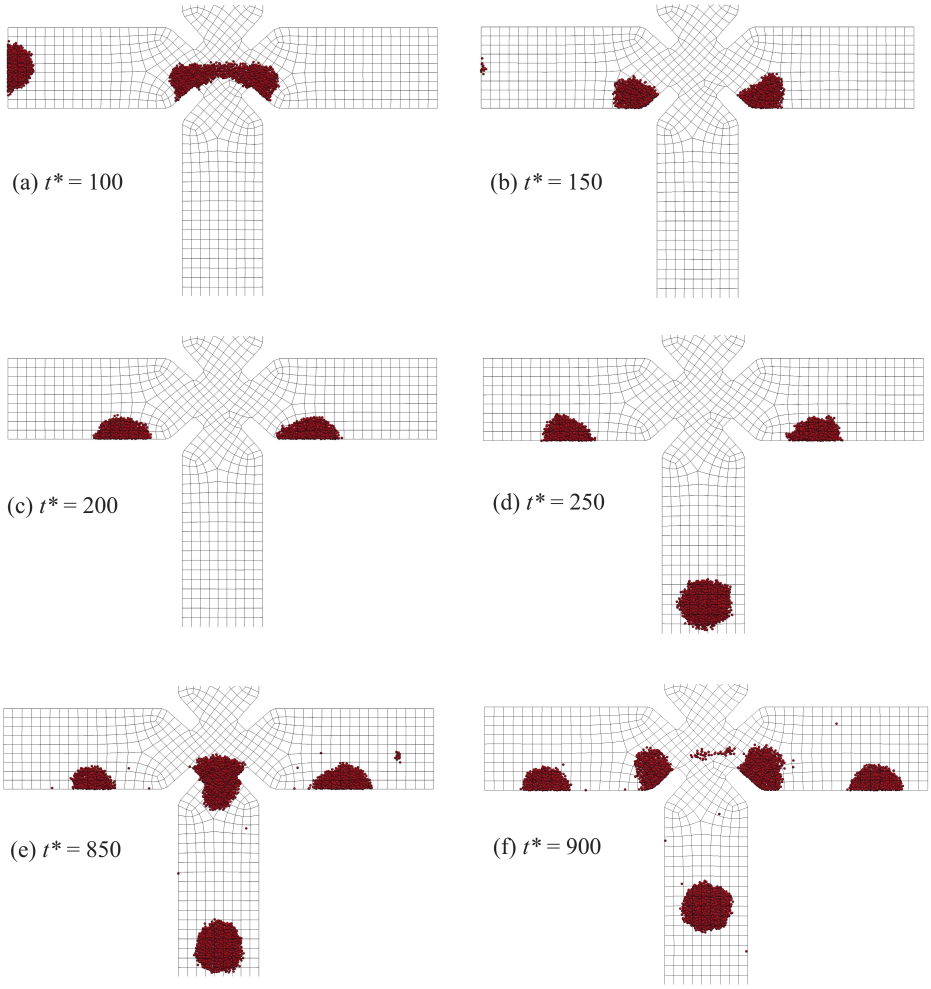
(h) $t^* = 1500$

Figure 5.21: Droplet deformation snapshots in the extensional flow problem, for throat case B. Times range from $t^* = 1150 - 1500$.



(a) $t* = 1500$
(b) $t* = 1550$
(c) $t* = 1600$
(d) $t* = 1650$
(e) $t* = 1700$
(f) $t* = 1800$
(g) $t* = 1850$
(h) $t* = 1900$

Figure 5.22: Droplet deformation snapshots in the extensional flow problem, for throat case B. Times range from $t^* = 1500 - 1900$.


(a) $t^* = 100$

(b) $t^* = 150$

(c) $t^* = 200$

(d) $t^* = 250$

(e) $t^* = 850$

(f) $t^* = 900$

Figure 5.23: Droplet deformation snapshots in the extensional flow problem, for throat case C. Times range from $t^* = 100 - 900$.

Chapter 6

Hybridising MD and Continuum Fluid Solvers

To achieve the best of both worlds.
- Anonymous

6.1 Introduction

'Hybrid' technologies have been steadily permeating the market in recent years — a concept which was originally inspired by genetics. The main objective of a hybrid system is to combine two or more constituents, each having its own desirable characteristics, that operate in conjunction with each other in a more efficient and productive manner than each constituent can do on its own. Essentially, the main aim is "to achieve the best of both worlds".

A hybrid methodology is of interest to us in this study because discrete, molecular solvers for studying fluid mechanics — that is, molecular dynamics — are burdened by an excessive processing requirement. Although it would be ideal to apply detailed Molecular Dynamics to extremely large macro-scale systems and long simulation runs it is: (1) prohibitively computationally expensive to do so, (2) inpractical, as most of the time only small parts of a flow system need the detail of a molecular approach, and (3) involves an excessive time of active processors, which apart from being a financial burden is also a cost to the environment. In view of these formal limitations, a hybrid approach may be adopted in which a computationally cheap solver — a continuum-based formulation numerically implemented in a Computational Fluid Dynamics (CFD) package — is contiguously coupled with a molecular solver, for example, in a domain-decomposition type configuration. Through this approach, MD is applied solely in regions of the computational domain where it is required to resolve the details and new phenomena of a particular problem, while CFD is applied to the rest of the flow domain to alleviate the computational cost. As a result, longer time-scales and length-scales may be simulated. Combining the two disparate formulations is the subject of much current research.

Indeed, purely-continuum hydrodynamics has had a monumental impact in most fluid mechanics problems, ranging from flows around aircraft wings to flows around miniscule objects. Quite notably in these circumstances, the hydrodynamic formulation does not require any form of molecular detail, for example, in the boundary conditions or constitutive relations, with the exception of the transport coefficients such as viscosity and thermal conductivity. However, at extremely small-scales, the molecular detail becomes ever more important, and a purely-continuum approach may not be adequate. In these circumstances, a ‘hybrid’ approach may be exploited that differs from domain-decomposition methods. The continuum model may still be applied in the entire domain, however, molecular simulations are applied to provide the missing data. Fluid-mechanics’ problems of this type may then be classified into two [6]: (1) unknown boundary conditions or (2) unknown fluid-properties (or constitutive relations). The former are especially dominant in nano-scale channel flow problems, where the study of heat transfer and stress at channel walls is vital, and also phenomena such as velocity and temperature slip, adsorbtion/wetting and chemical reactions are important. A typical problem is the contact of two immiscible fluid interfaces on a solid wall; the continuum formulation breaks-down close to the contact line [73, 74]. Unknown fluid-properties, the second type of problem mentioned above, are needed for modelling complex fluid systems such as polymeric fluids, in which the constitutive relationship is non-Newtonian, or else where the relationships between the transport coefficients and the fluid-state are unknown.

In general, “multi-scaling” is the term coined to refer to processes and problems found in many disciplinary fields that span a large range of length-scales and/or time-scales; the hybrid methodology is a computational tool which bridges the gap between these disparate scales by applying different formulations, where appropriate.

The main issues that are encountered in the development of hybrid MD-continuum methods for liquids are outlined below. This list has been compiled from literature sources and also from communications with researchers in this field, including Petros Koumoutsakos, Rafael Delgado-Buscalioni and Nicolas Hadjiconstantinou.

1. Implementation and programming issues:
    (a) Hybrid solutions in arbitrary geometries — Most MD solvers used in the literature are restricted to cuboid domains, an issue which obviously is inherited by hybrid MD-continuum case setups.
    (b) Parallelisation issues — Parallelisation of the entire hybrid code is usually a top priority due to the MD component, and is not a trivial task [75].
    (c) Code-coupling issues — It is probably always necessary to implement a coupling framework (e.g., [76]), which handles all the necessary requirements of code coupling, including deployment and execution of solvers, and explicit transfer and imposition of information from contiguous solvers at well-defined times of

the simulation run. If for example, the MD and CFD solvers are written in different programming languages, they need to communicate with each other using a common language which is understood by both codes. In such cases, a 'translator' is always necessary during input/output communications. Perhaps the most critical issue in code-coupling is dealing with codes which are not open-source, in which it becomes difficult to modify underlying code to fit the coupling framework.

2. Coupling-specific modelling issues:
    (a) Applying continuum-derived boundary conditions on an MD simulation is a challenging task, as the number of degrees of freedom from the continuum solution is usually much smaller than that of its molecular counterpart. Both state- and flux-based methods have been proposed in the literature, and are typically used for steady-state and transient cases respectively.
    (b) Measuring coarse-grained properties from a molecular simulation run and applying them onto a continuum mesh is typically straightforward in CFD but problems still persist. Smoothing of fluctuations in the coarse-grained quantities is important for maintaining a stable continuum solution. If fluctuations are physical, it is also possible to handle them in the continuum sub-domain by including stochastic and dissipative terms in the governing equations [77]. Furthermore, there is still a balance of competing aims between spatial- and time- resolution of coarse grained properties and their statistical accuracy.
3. Continuum breakdown — Identifying regions of the computational domain where the continuum formulation breaks-down and where a molecular approach is required to provide the necessary data.

In Section 6.2 we detail the general aspects of hybrid methods and document the published methods for dense fluids that couple MD and continuum solvers. We explain the key ingredients for creating robust, flexible and parallelised hybrid formulations in Section 6.3, while in Section 6.5 we develop and explain our generalised coupling framework. The continuum solver, which is used in this hybrid work is briefly described in Section 6.4. We give an overview of how to run a typical hybrid simulation in Section 6.6, and present results in Section 6.7.

6.2 Background

Hybrid methods can be classified as *indirect* or *direct*. The indirect hybrid method refers to the well-established "molecular-*then*-continuum" approach, in which the continuum-*unknowns* are measured from *a priori* molecular simulations. Some examples of these unknowns include velocity-slip and temperature-slip boundary conditions; transport properties [36]; or contact angles in immiscible fluids next to wall boundaries [78]. The direct methods, which are of most interest to us, adopt a concurrent "molecular-*with*-continuum" integrative framework [64], that involves direct-coupling of information between different representations in real time simulation. In general, it is the continuum-to-molecular (C→M) coupling direction which causes so much concern, since a few continuum variables need to be translated into many degrees of freedom: a "boundary condition" on the molecular sub-domain. In a sense, the exact C→M transformation is formally intractable. Therefore, the main challenge of most hybrid models is to develop a sufficiently-tractable C→M coupling model, which in most circumstances depends on the physics of the flow problem. The M→C coupling direction is less problematic, as the continuum formulation will always represent average properties over a large number of molecules.

Hybrid molecular-continuum methods have been developed for all materials: solids, liquids and gases. In solid mechanics problems, hybrid techniques [79–84] are less daunting, since coupling is generally aided by the local-confinement of molecules. C→M coupling is mainly based on displacements. In these techniques, the molecular formulation is limited to small subsets of the computational domain, where the continuum approximation of the finite-element method is equivocal, or the formulation is unable to capture the underlying phenomena. Some examples include the transfer of bulk-forces to defects or cracks within the material, in order to observe the transition from local deformation, to crack-propagation and ultimately to failure.

In fluid mechanics, the molecules are free to move and collide, and the C→M coupling scheme is generally not straightforward. Gases are however more tractable than liquids as they may make use of the well-established kinetic theory of gases, in particular by using the statistical-based Direct Simulation Monte Carlo (DSMC) [85] formulation instead of MD. DSMC particles represent groups of molecules and interact only through stochastic collisions. Therefore, the C→M boundary conditions are generally more manageable. The method typically resets all DSMC particles at the coupling region after every time-step with velocities sampled from a distribution function (e.g., the Chapman-Enskog) at the desired temperature and mean velocity. Hybrid models for rarefied gases [86–90] have been used in many applications, including diffusion, shock waves and flow past a sphere.

Hybrid techniques for dense fluids that incorporate MD as the molecular solver are generally more complex. The high densities and short-range interactions prohibit the same implementation of boundary conditions used in hybrid methods for solids or gases. To a certain extent, C→M type of boundary conditions also lack in their development in pure-MD. Major contributions to hybrid methods for liquids are by O'Connell and Thompson [4], and Nie and co-workers [47] for the constrained Lagrangian techniques, Hadjiconstantinou and Patera [64], and Werder and Koumotsakos [38] for their work in Schwarz alternating methods, E and Ren [6] for their novel Heterogeneous Multiscale Method (HMM), and Flekkoy, Delgado-Buscalioni, Coveney [5, 8] for their flux-based coupling methods. As these techniques are suitable for our purposes, we describe them in brief, highlighting in particular their advantages and drawbacks.

6.2.1 Hybrid MD-Continuum methods for liquids in the literature

O'Connell and Thompson [4] pioneered the first hybrid method for dense fluids using a domain-decomposition methodology. They considered a simple cuboid domain for simulating a startup Couette flow problem, in which MD is applied in one half of the domain and a finite-difference CFD solver is applied in the other half (see Fig 6.1). A 'Hybrid Solution Interface' (HSI) is placed at the intersection of both flow domains, which consists of a coupling region of finite thickness that overlaps both sub-domains. The coupling region provides for continuity of mass and momentum across the continuum-molecular interface.

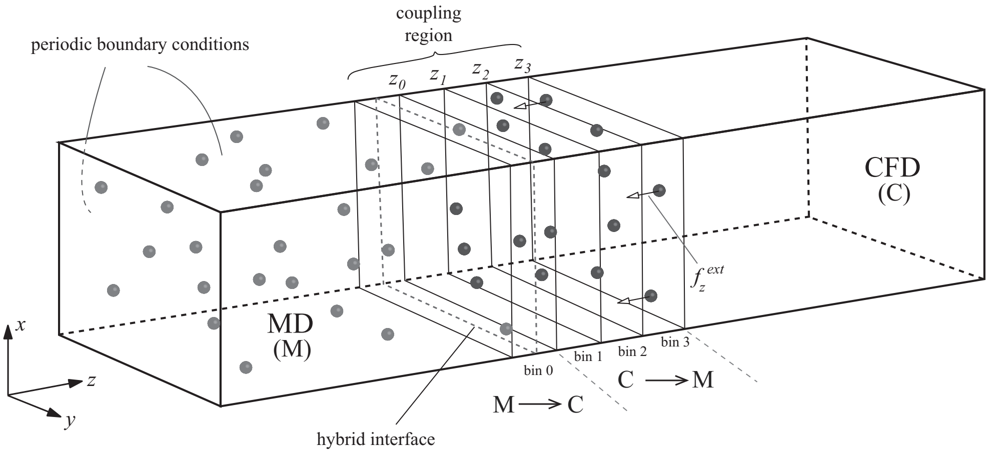


Figure 6.1: Schematic of the domain-decomposition of CFD and MD solvers in the hybrid methodology proposed by O'Connell and Thompson [4]. This schematic is generally also applicable to many other domain-decomposition techniques described in the literature.

The coupling region is intentionally divided into a series of bins. M→C coupling is achieved by averaging the velocity of molecules located inside the zeroth bin. This is imposed as a boundary condition to the CFD solution, replacing values at nodes distributed on the HSI at each time-step of the time-evolution of both solvers. O'Connell and Thompson invoke the same time-step for both CFD and MD formulations, $\Delta t_c = \Delta t_m$. In the C→M direction, coupling is determined by applying a constrained Lagrangian technique to molecules that reside in the remaining bins of the C→M coupling region. The equations of motion for these molecules are modified to include the non-holonomic constraint,

$$\sum_{i=1}^{N_J} (p_i)_x - M^{(J)} u_x^{(J)} = 0, \tag{6.1}$$

where N_J is the total number of molecules in the J^{th} bin and $(p_i)_x = m(v_i)_x$ is the momentum of the i^{th} molecule in the $x-$direction. $M^{(J)}$ is the continuum mass of the fluid element of the J^{th} bin and $u_x^{(J)}$ is its continuum velocity in the $x-$direction. The equations of motion for a molecule i within the J^{th} bin are derived by recasting equation (6.1) into a holonomic constraint,

$$\dot{x}_i = \frac{(p_i)_x}{m} + \xi \left[\frac{M^{(J)}}{mN_j} u_x^{(J)} - \frac{1}{N_J} \sum_{i=1}^{N_J} \frac{(p_i)_x}{m} \right], \tag{6.2}$$

$$(\dot{p}_i)_x = - \sum_j \frac{\partial}{\partial x} U_{LJ}, \tag{6.3}$$

where U_{LJ} is the Lennard-Jones intermolecular potential and $\xi \sim 0.01$ is the strength that controls the rate at which molecules' momenta relax to the local continuum velocity. In addition, a linear external force $f_z^{ext} = -\alpha P \rho^{-2/3}$ is applied to molecules in the last bin so as to set the correct density and pressure, and a weakly coupled Langevin thermostat is applied on all molecules in the system to set a constant temperature.

The hybrid method proposed by Nie *et al.* [47] follows similarly the method of O'Connell and Thompson [4]. The non-holonomic constraint for C→M velocity-coupling which they propose is for three-dimensional cells:

$$\frac{1}{N_J} \sum_{i=1}^{N_J} \mathbf{v}_i = \mathbf{u}_J(t). \tag{6.4}$$

The equations of motion are derived from the latter constraint:

$$\ddot{\mathbf{x}}_i = \frac{\mathbf{f}_i}{m} + \frac{D\mathbf{u}_J(t)}{Dt} - \frac{1}{N_J} \sum_{i=1}^{N_J} \frac{\mathbf{f}_i}{m}, \tag{6.5}$$

where $\mathbf{f}_i$ is sum of pair intermolecular forces and $D\mathbf{u}_J(t)/Dt$ is the Lagrangian derivative of the continuum velocity, which is discretised to:

$$\frac{D\mathbf{u}_J(t)}{Dt} = \frac{1}{\Delta t_m} \left(\mathbf{u}_J(t + \Delta t_m) - \frac{1}{N_J} \sum_{i=1}^{N_J} \dot{\mathbf{x}}_i(t) \right). \tag{6.6}$$

A different method of discretisation of the Lagrangian derivative is given in [91].

Additionally, Nie and co-workers invoke a continuum time-step, Δt_c, 50 times larger than the MD time-step, Δt_m. This time-step difference technique may generally be more practical, but the computation saved is perhaps still negligible, since one MD time-step is several times more demanding than one CFD time-step. Flux of mass is introduced at the terminating edge of the domain, according to which molecules are inserted/deleted following from $n' = -A\rho u_z \Delta t_c / m$, where A is the area perpendicular to the cell and u_z is the perpendicular component of velocity. In the same bin, a diverging external force is additionally applied to the residing molecules:

$$f_z^{ext} = -\alpha p_0 \sigma \frac{z - z_2}{1 - \frac{(z-z_2)}{(z_3-z_2)}}, \tag{6.7}$$

where p_0 is the average pressure in the MD region, α is a constant of $\mathcal{O}(1)$, and z_k, $k = \{1, 2, 3\}$ are the end positions of the bins as shown in Figure 6.1. The force component points in the direction of the interface so that particles do not exit the MD sub-domain, while introducing a dilute region for new molecules to be inserted straightforwardly. The z-component of the particle velocities inside $z_2 < z < z_3$ are coupled to a heat bath to prevent overheating by the action of the external force f_z^{ext}.

Certainly, the major drawback of this hybrid method is that the MD code used is limited only to cuboid-shaped computational domains. It could also be the case that not even an internal mesh is used to define the domain space. Therefore, the method in which they implement the coupling schemes requires 'hard-coding' to link small volumes of the coupling region with the nodes on the staggered mesh used for the Navier-Stokes CFD solver. As a result, these hybrid techniques prohibit complex-shaped geometries. Another general drawback of constrained Lagrangian techniques is that the accuracy and resolution of velocity is coupled. Since adequate statistics are required for accurate sampling of MD-derived properties, two major concerns are the size of the coupling bins and the averaging period during which properties are sampled. Other work using this constrained Lagrangian hybrid technique is reported in [57, 91–93].

A similar approach to domain-decomposition was proposed by Hadjiconstantinou and Patera [64]. However, there are several noticeable differences between the two methods. Most prominent is the application of a 'Schwarz alternating method' — a domain-decomposition methodology originally published by Schwarz [94] in 1870. This method has recently resurfaced for its natural adaptation to parallel processing, sub-structuring methods and multi-scale problems. The Schwarz method is implicit (i.e., iterative) in determining the correct boundary conditions at the interface of two coupled formulations. On each sub-domain, solutions proceed sequentially, in which the *latest* solution from the other sub-domain is always taken as the new boundary condition. The iterative method is repeated a number of times (typically $\mathcal{O}(20)$) until both 'macroscopic' solutions converge within the common coupling region. Hadjiconstantinou and Patera [64] couple the CFD and MD formulations solely through velocity.

A 'relaxation technique' is used within Schwarz-type methods in order to guarantee convergence of solutions within each respective sub-domain, and even accelerate the rate of convergence [95], e.g.,

$$(\mathbf{u}_J^{(n)})_m = \theta(\mathbf{u}_J^{(n-1)})_m + (1 - \theta)(\mathbf{u}_J^{(n-1)})_c, \tag{6.8}$$

where θ $(0 \leq \theta < 1)$ is the relaxation parameter, m, c denote molecular and continuum solutions, and n is the current Schwarz iteration. Hadjiconstantinou and Patera [64] use no relaxation, $\theta = 0$, because the systems they considered were relatively small. Larger hybrid systems without relaxation enabled are, however, susceptible to diverging solutions.

Another difference to the constraint Lagrangian technique is that a simpler and faster 'Maxwell-Demon' method is preferred in the C→M coupling region. The description of this velocity-control technique has been detailed in Section 4.4.5. In brief, all molecules residing within a C→M coupling bin (Figure 6.1) are reset every time-step from a Maxwell-Boltzmann distribution at a target temperature and the continuum velocity $(\mathbf{u}_J^{(n)})_m$ (see Eqn (6.8)) which is supplied from the CFD solution of the previous Schwarz iteration. Periodic boundary conditions are applied at the boundaries of the MD domain in order to keep constant the number of molecules in the system. A relatively large 'reservoir' is placed between the C→M region and the periodic boundaries in order to remove the effects introduced by the domain's periodicity. Therefore, there are no external forces applied to molecules at the boundary, except of course those forces resulting from the explicit positions of imaged molecules from the cyclic boundaries.

Researchers who work with this hybrid technique argue that the Schwarz alternating method for hybrid MD and CFD formulations decouples also the time-scales as well as length-scales, since the hybrid simulation may stop, upon achieving steady-state. However, we observe that the Schwarz alternating technique and the constrained Lagrangian techniques mentioned earlier are very similar to each other. They both use the same exact domain-decomposition, and also use the same time-step differences and time-coupling intervals. However, rather than employing a sequential time-marching scheme, the Schwarz technique uses a sequential iterative scheme. The benefit of this is the inclusion of a criterion within the hybrid algorithm for stopping the simulation run, once matching of fields within the coupling region is detected. Perhaps a better technique is to ensure that velocity solutions in the whole flow field do not change in successive iterations.

Hadjiconstantinou and co-workers test their hybrid model on several applications, including 2D flow around an obstruction located in a nano-channel [64], and a moving contact line between two immiscible fluids in Poiseuille-type flow configurations [96]. In these examples it is perhaps more clear that their MD code allows only relatively simple cuboid-shaped domains, albeit their proposed Maxwell-Demon coupling method in theory may be applied in 3D coupling regions. The application of periodic boundary conditions is also the cause for limiting the complexity of their hybrid domain geometries.

Werder, Walther and Koumoutsakos [38] also use the Schwarz alternating method for studying steady-state flows around a fixed carbon nano-tube (CNT). The CNT and a cuboid MD domain of solvent molecules surrounding it is modelled using MD that is coupled to a larger CFD simulation. In their hybrid technique, they do not employ the reservoir or Maxwell-Demon techniques for applying C→M boundary conditions. Instead, they adopt the constrained Lagrangian technique for imposing continuum velocity, and introduce movable pistons for imposing mass flux in one direction (the other directions remain periodic), applying a physical external force distribution — based on the structure of the fluid — to maintain constant density profiles close to the termination edge of the MD domain. Kotsalis *et al.* [51, 97] improve upon this method by further controlling one-dimensional density gradients next to non-periodic planar boundaries.

The hybrid methods discussed so far impose continuum-molecular coupling through macroscopic state (i.e. volume) properties. The first *flux*-based hybrid method for liquids was first explored by Flekkøy, Wagner and Feder [8]. This method employs the same domain-decomposition considered earlier, see Figure 6.2. However, coupling involves the mutual exchange and conservation of 'flux-densities' — mass, momentum, and subsequently also energy [98]. Similar work was also published by Delgado-Buscalioni and Coveney [5]. The continuum formulation may generally be written as:

$$\frac{\partial \phi}{\partial t} = -\nabla \cdot \mathbf{J}_\phi, \tag{6.9}$$

where $\phi = \{\rho, \rho\mathbf{u}, \rho e\}$ are the densities, while $\mathbf{J}_\phi = \{\rho\mathbf{u}, \rho\mathbf{u} + \mathbf{\Pi}, \rho\mathbf{u}e + \mathbf{\Pi} \cdot \mathbf{u} + \mathbf{q}\}$ are the flux-densities; $\mathbf{\Pi} = p\mathbf{1} + \tau$ is the stress tensor, including the normal pressure and the viscous shear-stress, $\mathbf{q}$ is the heat flux vector, and e is the total energy.

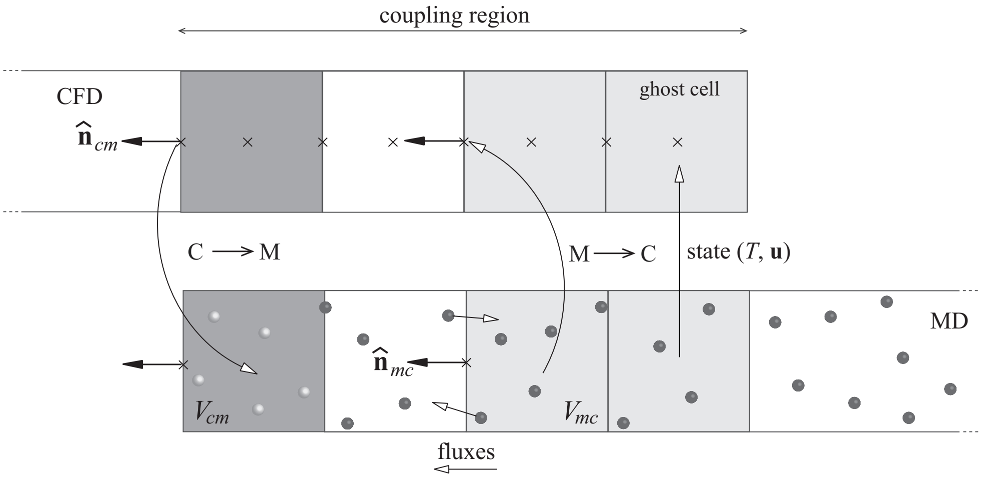


Figure 6.2: Schematic showing the exchange of flux-densities between CFD (top) and MD (bottom) domains within the coupling region as proposed in Ref. [5]. State-coupling was introduced to rectify the discontinuities between solutions in the coupling region.

For the M→C coupling direction, the computed molecular flux-densities replace the terms $\mathbf{J}_\phi$ in the overlaying continuum formulation, using a von-Neumann type of boundary condition: see lower part of Table 6.1. For the C→M coupling direction, the continuum fluxes $\mathbf{J}_\phi$ are transformed and imposed into molecular fluxes in the C→M bin: see upper part of Table 6.1. Molecules are inserted with velocities sampled from a Maxwell-Boltzmann distribution at the target temperature and velocity, and deleted by choosing molecules with velocities close to the continuum velocity, hence in general satisfying the condition $\langle \mathbf{v}' \rangle = \mathbf{u}$. External forces are applied to molecules to satisfy momentum flux due to continuum stress. Flekkøy *et al.* [8, 99] use a diverging distribution satisfying $g_i(z_3) = \infty$, $g_i(z_2) = 0$, such that $\mathbf{f}_i^{ext} = -A\mathbf{\Pi}\cdot\hat{\mathbf{n}}_{cm}(g_i(z)/\sum_i^{N_{cm}} g_i(z))$. Delgado-Buscalioni and Coveney [5] prefer a simpler uniform distribution $\mathbf{f}_i^{ext} = -A\mathbf{\Pi}\cdot\hat{\mathbf{n}}_{cm}/N_{cm}$. To impose the heat flux condition, $\langle\mathbf{J}_Q^{ext}\rangle\cdot\hat{\mathbf{n}}_{cm} = -A\mathbf{q}\cdot\hat{\mathbf{n}}_{cm}$, a set of (two or three) Nosè-Hoover thermostats are applied next to each other within the C→M cell, with each set at a different temperature.

Fluxes	C→M
Mass	$ms = -A\rho\mathbf{u}\cdot\hat{\mathbf{n}}_{cm}$
Momentum	$ms\langle\mathbf{v}'\rangle + \left\langle\sum_{i=1}^{N_{cm}}\mathbf{f}_i^{ext}\right\rangle = -A(\rho\mathbf{uu}+\mathbf{\Pi})\cdot\hat{\mathbf{n}}_{cm}$
Energy	$ms\langle\epsilon'\rangle + \left\langle\sum_{i=1}^{N_{cm}}\mathbf{f}_i^{ext}\cdot\mathbf{v}_i\right\rangle - \langle\mathbf{J}_Q^{ext}\rangle\cdot\hat{\mathbf{n}}_{cm} = -A(\rho\mathbf{u}e+\mathbf{\Pi}\cdot\mathbf{u}+\mathbf{q})\cdot\hat{\mathbf{n}}_{cm}$
	M→C
Mass	$\rho\mathbf{u}\cdot\hat{\mathbf{n}}_{mc} = \frac{1}{V_{mc}}\left\langle\sum_{i=1}^{N_{mc}} m\mathbf{v}_i\right\rangle\cdot\hat{\mathbf{n}}_{mc}$
Momentum	$\mathbf{\Pi}\cdot\hat{\mathbf{n}}_{mc} = \frac{1}{V_{mc}}\left\langle\left(\sum_{i=1}^{N_{mc}} m\mathbf{v}_i\mathbf{v}_i - \frac{1}{2}\sum_{i,j}^{N_{mc}}\mathbf{r}_{ij}\mathbf{f}_{ij}\right)\right\rangle\cdot\hat{\mathbf{n}}_{mc}$
Energy	$\mathbf{q}\cdot\hat{\mathbf{n}}_{mc} = \frac{1}{V_{mc}}\left\langle\sum_{i=1}^{N_{mc}} m\epsilon_i\mathbf{v}_i - \frac{1}{2}\sum_{i,j}^{N_{mc}}\mathbf{r}_{ij}\mathbf{v}_i\mathbf{f}_{ij}\right\rangle\cdot\hat{\mathbf{n}}_{mc}$

Table 6.1: Coupling of flux-densities from C→M (top) M→C (bottom) [5, 8]. See also Figure 6.2. In the table, s denotes the number of molecules which are inserted/deleted in the C→M bin.

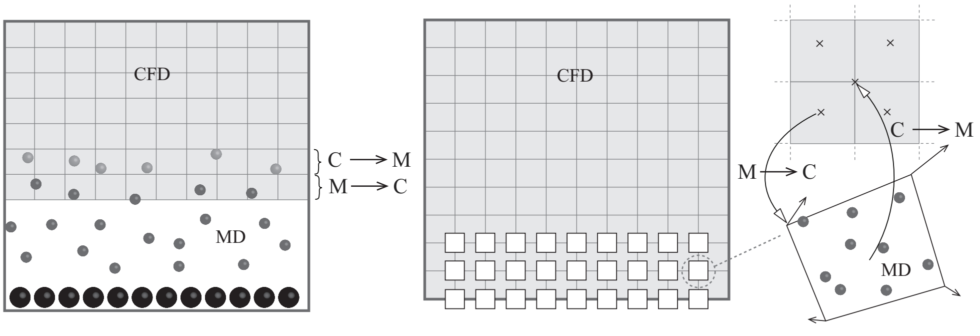
In the M→C coupling region, we note that it is not the molecular fluxes which are imposed on the continuum formulation, but merely 'state' properties sampled from a cell volume V_{mc}, which are converted into a 'flux' at a face by applying the dot product of the state-property (defined at the cell-centre) with the normal-face vector. Evidently, this does not represent a 'conservative' flux-coupling technique, which should constitute the explicit crossing of molecules through faces, as was performed in flux-based methods for gases [86, 87]. Indeed, the drawback of measuring fluxes in molecular systems is that there are high statistical fluctuations [37], which is probably the reason the authors preferred cell-based measurements. A more accurate representation of a flux property would be to measure the state-properties within two adjacent cells first, then perform a linear interpolation to determine the property at the communal face, and finally apply the dot-product.

Similar remarks apply to the C→M coupling technique. While we presume that the continuum fluxes in the upper part of Table 6.1 are obtained from face-interpolation, these are, however, imposed 'equally' within the adjacent cell *volume*, rather than in the 'neighbourhood' of the particular *face*. Obviously, the latter represents a formidable task.

In addition to these points, researchers showed in subsequent papers [100–102] that a mismatch in velocity profiles exists between the continuum and molecular formulations in the overlapping region, (e.g., in the case of a hybrid oscillatory shear-flow [100]). To fix this, they introduce 'hybrid gradients' in the M→C coupling region, in which state-based velocity and temperature measurements are set as 'ghost' cells on the boundary of the continuum formulation. Similar to other hybrid methods, only simple cuboid-shaped MD domains have been considered in this work.

E and Ren [6] describe a new hybrid framework, which they dub the 'Heterogeneous Multiscale Method' (HMM) [103]. The objectives they set were to decouple length-scales and time-scales, significantly more so than other hybrid methods in the literature. The premise behind the HMM is that the continuum model is applied to the entire computational domain, hence there is no domain-decomposition of MD and continuum formulations (see Fig 6.3). The MD simulation only runs in disconnected parts of the computational domain, and over a discontinuous time-line, in order to provide unknown information to the continuum formulation in real-time simulation:

1. *Boundary conditions* — MD regions are distributed on the boundary (and adjacent to it), to provide wall boundary conditions for the continuum simulation.
2. *Constitutive relations* — MD regions are distributed in the internal part of the domain to provide an interpolated field of properties such as the stress tensor and heat flux vector. Calculating transport coefficients in a direct simulation, such as viscosity and thermal conductivity, may not be a viable option, since the low signal-to-noise ratios of these properties demand extremely long averaging time periods.



(a) Domain-decomposition

(b) Heterogeneous Multiscale Method (HMM)

Figure 6.3: Schematics comparing hybrid methods: (a) domain-decomposition and (b) the heterogeneous framework proposed in Ref. [6]. Inset to Figure (b) the C→M coupling is first achieved by setting the velocities at nodes of the MD domain box from continuum values, while PBCs are applied everywhere. The domain is allowed to shear during the MD simulation run. M→C coupling is then performed by measuring the stress tensor from the MD simulation and passing it to the continuum formulation where it is unknown.

The C→M boundary conditions for boundary related problems are similar to the previous methods, i.e. constrained Lagrangian methods are applied. However, for problems associated with the unknown constitutive relations, the authors use a shearing domain approach as found in standard MD literature [22]. In essence, velocities of the MD domain boundary nodes move with velocities set from the overlaying continuum solution. The M→C coupling then involves the sampling of measurable properties such as stress, and setting them to the respective continuum nodes.

In a recent paper E. *et al.* [104] describe a seamless multiscale method for ‘unknown constitutive relations’ type of problems, which differs from the HMM by a modified time-coupling framework. Both continuum and molecular simulations use the same implicit time-step (although using different integration time-steps) and boundary conditions are passed every time-step. The method is not applicable for the ‘unknown boundary conditions’ type of problems, because of the short time available for sampling and subsequently controlling properties within statistically accurate limits.

It seems that the HMM is the most practical and efficient hybrid method, but lacks generality due to the complexities involved: a larger amount of programming effort is required for arbitrary geometries (e.g., the process for distributing large number of MD sub-domains within a complex domain, and re-initialisation of the domains when the shearing MD box elongates too much); there is a computational limitation on the number of cells used by the continuum domain mesh, since one MD sub-domain (typically $\mathcal{O}(10^4)$ molecules per sub-domain) needs to be applied to each cell node; wall-fluid boundary and fluid-bulk methods are treated very differently; temperature coupling in the moving MD box method is very challenging, in order to measure the heat flux vector.

6.3 Overview of a ‘general’ hybrid method

The three key “ingredients” of a hybrid solver are:

1. The *components* – referring to two or more solver components that need to be coupled. (e.g., in our work we use a two-component hybrid system: a Molecular Dynamics solver and the Navier-Stokes continuum formulation, numerically implemented as a CFD solver).

2. The *global coupling framework* – how space and time are defined across all components, including, in particular, coupling interfaces and coupling times.

3. The *coupling model* – the actual models used by each component that exchange and transfer information, and impose the necessary actions in the form of boundary conditions.

We use OpenFOAM as our numerical code framework. Our MD component can be defined in a computational domain of arbitrary geometries; the same applies to hybrid cases. Both the MD and CFD codes which we use are coupled in a single hybrid code, since both are written in the same programming language (C++), and use the same infrastructure (mesh, time-scheme, property fields etc.). Therefore, there are no issues of input-output data-manipulation or data-transfer between diverse programming languages. The hybrid code is parallelised using a domain-decomposition technique based on the cells that comprise the MD and CFD meshes.

A general coupling framework is developed to integrate the two solvers in time and space, which we describe in more detail in a later section. The spatial part of the coupling framework is defined around a domain-decomposition methodology, in which the continuum and molecular formulations are applied in unique and non-overlapping parts of the computational domain. An overlap region is only allowed where the continuum and molecular formulations meet in order to pass boundary conditions. In our implementation, any number of meshes may be created within a hybrid case as long as each mesh belongs to either the continuum sub-domain (CFD solver) or the molecular sub-domain (MD solver). The overlapping regions at the continuum-molecular interfaces, which can be of any arbitrary 3D shape, need to be identical on the corresponding pair of continuum and molecular meshes; a process which we achieve efficiently using ‘sub-meshing’.

The temporal part of the framework distinguishes between the different time-schemes required by the CFD and MD meshes, in particular their time-integration steps. The framework also handles the coupling times, that is, the times at which communication of field properties (e.g., velocity and density) within the overlapping region take place between formulations. Fields measured from an overlapping region of one mesh are set as boundary conditions on its corresponding coupled mesh. The linking of cells and faces between continuum and molecular meshes is an implementation requirement of the coupling framework, and is performed for each overlapping region within the domain. In this research, we have adopted a method similar to the Schwarz Alternating Method. The only difference is that our method is presumed to be time-marching rather than iterative. We make use of an alternating sequential time coupling framework and incorporate a relaxation method for passing boundary conditions. All continuum-molecular interfaces remain stationary throughout the duration of the simulation: an adaptive domain-decomposition formulation has not yet been developed. Our coupling framework may also be used to couple other continuum-molecular types of solvers, (e.g., DSMC with continuum) or for implementing the hybrid HMM framework [6].

6.4 Continuum Solver

The continuum component of the hybrid model which we use in this thesis is the compressible, viscous, isothermal Navier-Stokes governing equations [105, 106], which are numerically implemented in a finite-volume solver called sonicLiquidFlow [107] already available in OpenFOAM [19] version 1.6. We alter the CFD code so that it uses the same reduced units for all space, time and field properties as used by our MD solver (see Section 2.3), in order to establish proper transfer of properties between MD and CFD solvers. The continuity and momentum equations are expressed by:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{6.10}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla p + \nabla \cdot \mu \nabla \mathbf{u}, \tag{6.11}$$

where μ is the dynamic viscosity, which is measured from equilibrium MD simulations (see §3.7.4), or from tables [7]. Moreover, the closure of the formulation is subject to an equation of state which is measured from an MD simulation at constant temperature. For example, a Lennard-Jones fluid at temperature $T^* = 2.4$ (see Fig 6.4) gives a fitted expression:

$$\rho^*(p^*) = \begin{cases} 0.221569 \ln(p^*) + 0.340401, & p^* \geq 0.9, \\ 0.368607 p^*, & 0 < p^* < 0.9, \end{cases} \tag{6.12}$$

or inversely:

$$p^*(\rho^*) = \begin{cases} \exp(\frac{\rho^* - 0.340401}{0.221569}), & \rho^* \geq 0.33, \\ 2.7129 \rho^*, & 0 < \rho^* < 0.33. \end{cases} \tag{6.13}$$

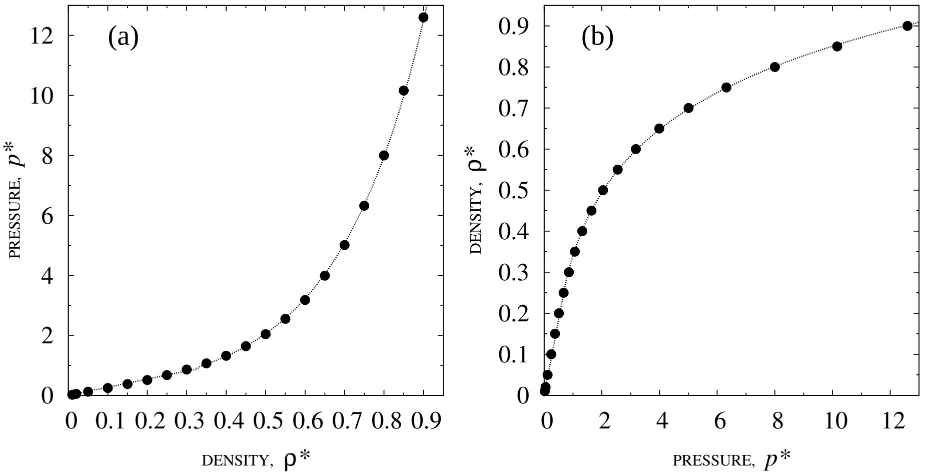


Figure 6.4: Equations of state for a Lennard-Jones liquid at constant temperature $T^* = 2.4$, (a) $p^*(\rho^*)$ and (b) $\rho^*(p^*)$. Data points are obtained from an equilibrium MD simulation with a thermostat applied. A density controller is used to change the density of the system at uniform time-intervals.

Limitations of solver

The current CFD code which we use in our hybrid code has many limitations, which we note for future consideration:

- An equation for temperature or energy is required in the continuum formulation in order to model heat or energy flow across the MD-continuum interfaces.

- The second term on the r.h.s. of equation (6.11), does not represent the full Newtonian shear stress relationship, that is $\nabla \cdot \mu \left[\nabla \mathbf{u} + (\nabla \mathbf{u})^{\mathrm{T}}\right] - \frac{2}{3}\nabla \mu \nabla \cdot \mathbf{u}$.
- The dynamic viscosity μ is assumed to be equal to the shear viscosity η sampled from an MD simulation. Also the dynamic viscosity is taken to be constant across the continuum domain, which is not the case for flows with varying state.
- The governing equations do not include hydrodynamic fluctuating and dissipative stress terms. These may be included to transfer and handle the fluctuations inherent within the molecular domain.

6.5 Coupling Framework

In this section we develop a general coupling framework for a two-component hybrid model in which molecular and continuum solvers are coherently coupled in time and space. The framework is generalised so that it may also be used for other hybrid schemes, e.g., for high altitude rarefied gas flows, using compressible Navier-Stokes and DSMC solvers as the continuum and molecular solver, with state- and/or flux-based coupling.

6.5.1 Spatial Coupling

We consider a general (multi-mesh) domain-decomposition scenario which is defined uniquely and comprehensively by n_{Ω}^{C} disconnected *continuum meshes*, in addition to n_{Ω}^{M} disconnected *molecular meshes*. Indeed, $n_{\Omega}^{C} \neq n_{\Omega}^{M}$ is possible (see Fig 6.5). In this work we consider a two-component hybrid method (not a multi-component), so it is reasonable to presume one molecular mesh overlapping any number of disconnected continuum meshes, but not any of the other molecular meshes. If two molecular meshes overlap they should be combined to form one molecular mesh (and vice versa for the continuum meshes). Therefore, the total number of *coupling regions* (or continuum-molecular interfaces) is $n_{\Gamma} = n_{\Omega}^{C} + n_{\Omega}^{M} - 1$. An arbitrary continuum mesh domain is denoted by Ω_{i}^{C}, where $i = 1, 2, \ldots, n_{\Omega}^{C}$, an arbitrary molecular mesh is denoted by Ω_{j}^{M}, where $j = 1, 2, \ldots, n_{\Omega}^{M}$ and their connected coupling region is denoted by $\Gamma_{i,j}$. In Figure 6.5 we show a multi-mesh domain-decomposition into continuum and molecular meshes.

We now describe our spatial coupling methodology within an arbitrary coupling region $\Gamma_{i,j}$ (that may consist of any geometric shape) contiguous to a continuum ($C = \Omega_{i}^{C}$) and molecular ($M = \Omega_{j}^{M}$) mesh. The example we consider is a complex-shaped overlap region resulting from the presence of a protruding molecular wall, as shown in Figure 6.6(a). In the coupling region, cells on the continuum mesh correspond exactly in size, shape and global coordinates to cells on the molecular mesh (see Fig 6.6(b)). We highlight this as key to fulfilling a robust spatial coupling scheme, realised by exchange of ‘fields’ of properties: a field is a list of values (scalar, vectors or tensors) for a given property (e.g., density, velocity and stress), in which one entry in the field corresponds to one cell on

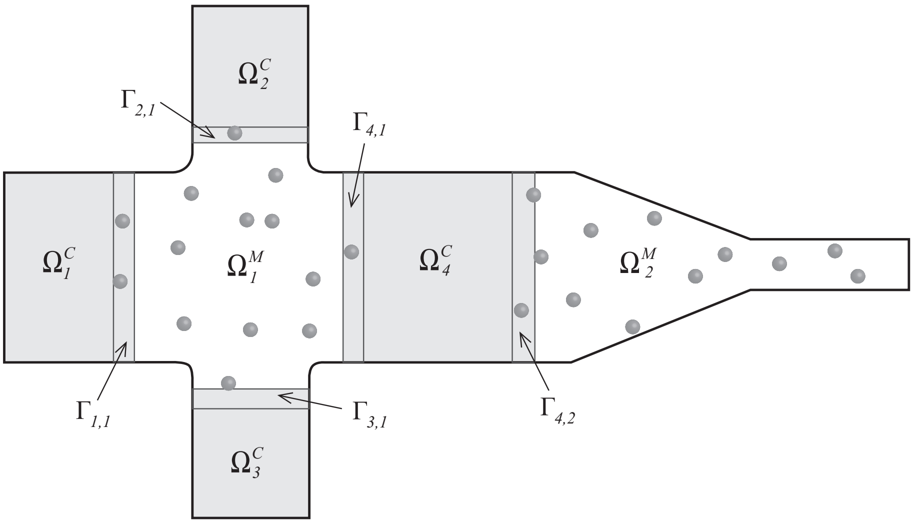


Figure 6.5: Schematic illustrating our multi-mesh domain-decomposition technique applied to an arbitrary nano-channel. The domain is decomposed into 4 continuum and 2 molecular meshes arbitrarily: Ω_i^C; $i = 1, \ldots, 4$, and Ω_j^M; $j = 1, 2$, respectively, with $n_\Gamma = 5$ molecular-continuum interfaces/coupling regions. Note that the schematic is for explanation purposes only, by no means does it represent an optimal domain decomposition; in practice MD would be applied next to wall interfaces, or in regions where compressibility effects might occur. Additionally, the coupling regions, which have been illustrated using rectangular shapes, may consist of any other geometry.

the particular mesh — properties are defined at the centre of the cell. In our case, a common overlapping region is obtained by sub-meshing molecular and continuum subdomain meshes from the global hybrid mesh. This is described in more detail in Section 6.6.

The coupling region consists of two main sub-regions where transfer of information occurs throughout the hybrid simulation: (1) at the boundary region of the continuum mesh; boundary conditions are imposed from the molecular to the continuum sub-domains (M→C), and (2) at the boundary region of the molecular mesh; boundary conditions are imposed from the continuum to the molecular mesh (C→M). The M→C sub-region is typically a layer of one or two cell widths thick; it may be made larger to include more layers or even made equal to the entire MD domain if the entire MD solution needs to be transferred to the continuum mesh. The C→M sub-region is at least two cell widths thick $\sim \mathcal{O}(10)\sigma$, and incorporates the non-periodic boundary conditions (NPBCs) which we developed in Chapter 4. The coupling region may include also a small buffer region, positioned between the M→C and C→M coupling zones. The aim of the buffer region is to allow spatial and temporal relaxation between boundary condition steps.

The M→C and C→M boundary conditions are based on quantities which can be understood by both continuum and molecular solvers. In this work we employ the continuum hydrodynamic state properties, velocity, density and temperature, that are numerically computed within cells of their respective discretised mesh and transferred between solvers at regular coupling intervals. For the M→C coupling direction an arbitrary cell-

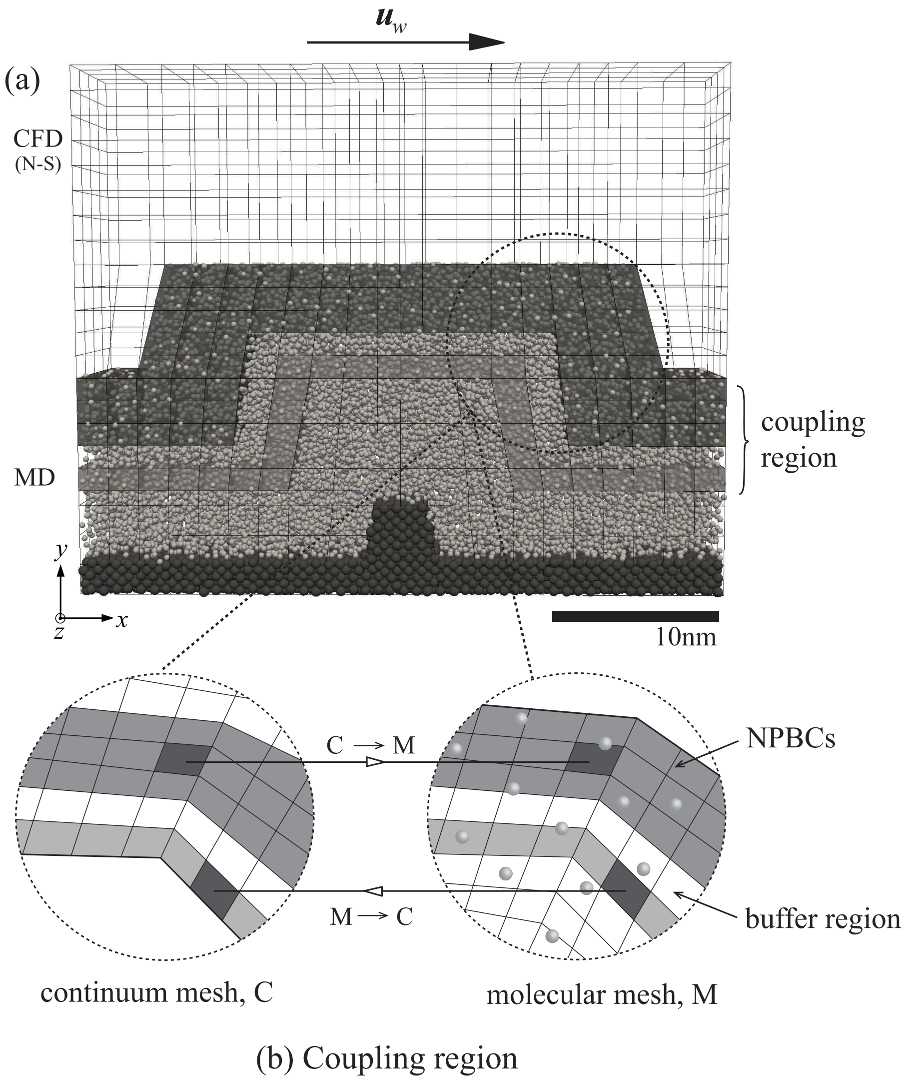


Figure 6.6: (a) Hybrid domain decomposition of a shear flow with a complex fixed-wall topology. (b) Schematic of the coupling region between continuum and molecular sub-domain meshes. M→C and C→M properties are transferred between pairs of coupled cells, as shown in the two highlighted regions. C→M coupling is achieved via the controllers from continuum-derived properties, while M→C coupling is achieved by coarse-graining of properties from cell-occupant molecules.

property measured numerically from the molecular mesh replaces the continuum value on the boundary cell of the continuum mesh. For the C→M coupling direction an arbitrary cell-property from the continuum mesh is transformed into molecular control procedures via the controllers and other boundary models (e.g. mass flux coupling at the boundary). As a consequence of coupling through hydrodynamic properties instead of conservative properties such as fluxes of mass, momentum and energy, this state-based hybrid strategy may not indirectly satisfy conservation laws. For example the molecular system may never retain the same total energy, total momentum or total mass. However we note that even if conservation laws were obeyed (for example in Ref. [100]), the hydrodynamic properties are not necessarily continuous at the coupling interface across different descriptions, thereby requiring also state-coupling to meliorate this error. The Schwarz method which is adopted in this thesis, commensurate with the implementation of multi-layer coupling to control first order gradients of properties, ensures that the hydrodynamic properties are continuous across the two formulations, and that the system converges to a steady-state, despite the fact that the method of boundary conditions is unconventional and non-conservative.

Spatially Linked Cells

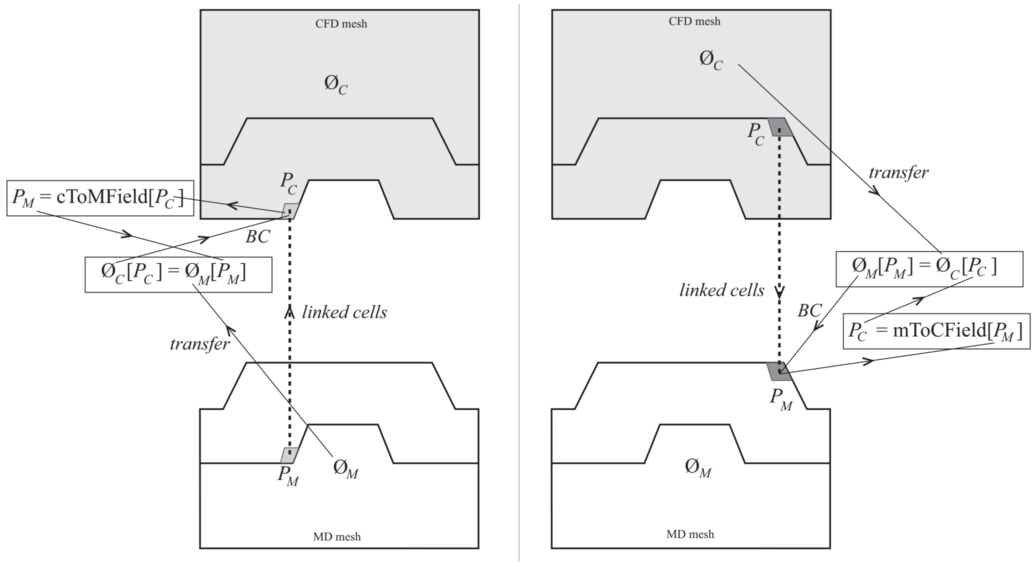
A prerequisite to implementing M→C and C→M state-based boundary conditions, is to link cells within the coupling region on the continuum mesh with the corresponding cells on the molecular mesh (see Fig 6.6(b)). We implement this in OpenFOAM by constructing two separate fields. The first field we define as the "continuum-to-molecular cell-index addressing", and refer to it as '`cToMfield`', in C++ notation. The `cToMfield` essentially represents all cells of the *continuum* mesh, in which each entry stores its corresponding (linked) cell-index of the *molecular* mesh; a value of -1 is stored if no link exists. Note that the indices of the field represent the indices of cells on the continuum mesh ($i.e., 0, 1, \ldots, n^C_{cells} - 1$). The second field represents the opposite data structure: the "molecular-to-continuum cell-index addressing", and denoted by '`mToCfield`' instead. Similarly, the indices of the latter run over the number of cells on the molecular mesh ($0, 1, \ldots, n^M_{cells} - 1$).

M→C BCs

We describe the M→C boundary conditions (BCs) implementation on cell-based properties. Consider an arbitrary property field ϕ_M (say velocity), sampled from the molecular mesh, and subsequently 'transferred' to the continuum mesh. On the continuum mesh, a loop is performed over those cells within the M→C coupling region. For each arbitrary cell P_C the following steps are carried out (see Fig 6.7(a)):

1. Find its corresponding linked cell index on the molecular mesh:

$$P_M = \mathsf{cToMfield}[P_C].$$



(a) M⟶C Boundary Condition (b) C⟶M Boundary Condition

Figure 6.7: Serial case showing two arbitrary links for applying boundary conditions in arbitrary geometries: (a) M→C and (b) C→M coupling. P_M, P_C denote two linked cells, which are identical in size, shape and location in global space, but are located on different meshes. ϕ_C, ϕ_M denote a property field from continuum and molecular meshes respectively.

2. Find the molecular coarse-grained velocity that corresponds to cell P_C, and replace its value within its corresponding field entry:

$$\phi_C[P_C] = \phi_M[P_M].$$

In general, the latter is written to include the relaxation method, equation (6.8):

$$\phi_C^{(n)}[P_C] = (1-\theta)\phi_M^{(n)}[P_M] + \theta\phi_C^{(n-1)}[P_C],$$

where $\phi_C^{(n-1)}$ is the previous continuum property field, and θ is the relaxation coefficient.

C→M BCs

The C→M boundary conditions (BCs) are implemented using a similar approach, in this case using our non-periodic boundary conditions (NPBCs). The method involves a transfer of a continuum field ϕ_C to the molecular mesh, followed by a looping procedure through all MD cells belonging to the C→M coupling region. For each arbitrary cell P_M, the following steps are carried out (see Fig 6.7(b)):

1. Find the corresponding linked cell of P_M:

$$P_C = \mathsf{mToCfield}[P_M].$$

2. Find the continuum velocity that corresponds to cell P_M, and replace its value within its corresponding target property in the NPBC on the MD mesh:

$$\phi_M[P_M] = (1-\theta)\phi_C[P_C] + \theta\phi_M^{(n-1)}[P_M].$$

Parallel processing

If the hybrid method requires to be simulated in parallel, the molecular and continuum meshes will be, as a pre-processing step, decomposed on different processors, as illustrated in Figure 6.8. Consequently, the coupling region of the continuum mesh may be located on a different processor than the corresponding coupling region of the molecular mesh. Cells of continuum and molecular meshes must also therefore be generally and properly linked across processors. We describe our modified technique for applying C→M boundary conditions in parallel. The same procedure may be applied to the M→C boundary conditions.

1. At the exact point in time when C→M boundary conditions are imposed, reconstruct the property fields ϕ_C on all processors $(0, 1, \ldots, n_{proc})$ into one list ϕ_C^{rec}. This is the only step that involves communication across different processors (see Fig 6.8).

2. For an arbitrary molecular cell P_M, in the C→M coupling region (on the current processor), first find the processor number on which its corresponding continuum cell resides on:

   $$proc = \mathsf{whichProc}[P_M],$$

   where $\mathsf{whichProc}$ is a field establishing the processor links between corresponding cells.

3. Then find the corresponding cell, P_C:

   $$P_C = \mathsf{mToCfieldOnProcs}[proc][P_M],$$

   where $\mathsf{mToCfieldOnProcs}$ is the new molecular-continuum cell-index addressing (list of fields) for handling multi-processors.

4. Finally determine the target property for cell P_M:

   $$\phi_M[P_M] = (1-\theta)\phi_C^{rec}[proc][P_C] + \theta\phi_M^{(n-1)}[P_M].$$

Flux BCs — Spatially Linked Faces

To implement flux coupling, faces on the *boundary* of the continuum mesh are linked to *internal* faces on the molecular mesh, and vice versa, using a similar data structure to that for cell-linking. Two fields are set up: (1) continuum-to-molecular face-index addressing

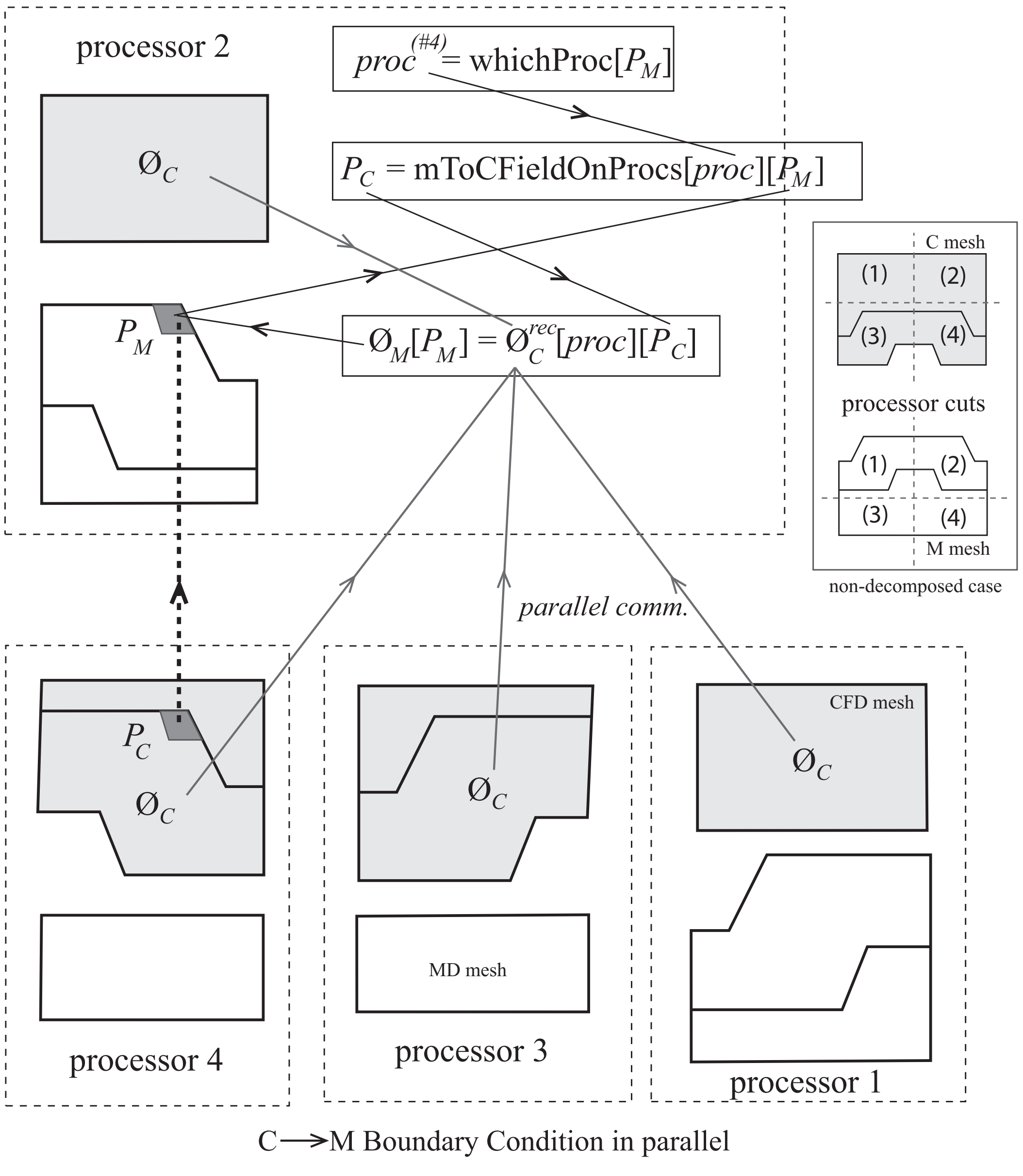


Figure 6.8: Schematic showing an arbitrary link between continuum and molecular cells for applying C→M boundary conditions in arbitrary geometries and in parallel. ϕ_C^{rec} is the target continuum field, which is first reconstructed on each processor. Inset (right): a non-decomposed schematic of the case, showing clearly the processor cuts, and assignment of meshes to processors.

for the continuum mesh (i.e. $f_M = \text{cToMfacefield}[f_C]$), and (2) molecular-to-continuum face-index addressing for the molecular mesh (i.e. $f_C = \text{mToCfacefield}[f_M]$). In addition to this, when imposing fluxes across different meshes, the face vector directions of two linked faces may not be the same, and an account of this is necessary in order not to reverse the direction of the flux when imposing it as a boundary condition. Accordingly, we introduce two other linked-fields (δ_M, δ_C) that give the 'sign' of the dot product of the pair of normal face vectors, e.g., $\delta_M[f_M] = \delta_C[f_C] = sgn(\hat{\mathbf{n}}_f^M \cdot \hat{\mathbf{n}}_f^C)$. Both face normal vectors point in the same direction if $sgn(x) = 1$; and in opposite directions if $sgn(x) = -1$. For example, if $\nabla \phi_C$ is a field of continuum fluxes $\{0, 1, \dots n^C_{faces} - 1\}$ prescribed to the molecular mesh, then for an arbitrary face f_M on the molecular mesh:

1. first find its linked face on the continuum mesh:

$$f_C = \text{mToCfacefield}[f_M],$$

2. then prescribe the target flux, multiplying by the sign direction δ_M:

$$\nabla \phi_M[f_M] = \delta_M[f_M] \nabla \phi_C[f_C].$$

6.5.2 Time Coupling

A time-coupling framework is required within a hybrid model in order to combine the two different molecular and continuum time-scales in a coherent manner. This defines the periods over which continuum and molecular meshes are solved, and the times at which C→M and M→C spatial boundary conditions are executed. We implement this part of the framework as a separate 'entity' so that various time-coupling schemes may be implemented.

Sequential Technique

The common sequential time-scheme consists of advancing molecular and continuum meshes in succession, for a duration of a common coupling interval $\Delta t_{coupling}$, after which boundary conditions are passed between solvers. The time coupling interval is expressed as:

$$\Delta t_{coupling} = \tau_C \Delta t_c = \tau_M \Delta t_m, \tag{6.14}$$

where $\Delta t_c, \Delta t_m$ are the continuum and molecular integration time-steps, respectively, and τ_C, τ_M are positive integers that represent the number of time-steps within the coupling interval. Note that the continuum-molecular time-step disparity ratio is given by $\Delta t_c / \Delta t_m = \tau_M / \tau_C$. The coupling time interval, and hence the frequency of BC imposition, is a user-defined property.

Possibly the main drawback of this technique is that as one mesh advances its solution for the duration of $\Delta t_{coupling}$, all other meshes must wait their turn in the sequence.

Exchange of boundary conditions is of the *alternating* or the *synchronous* type. We opt to use the former in our work and describe it as follows (see Fig 6.9(a)):

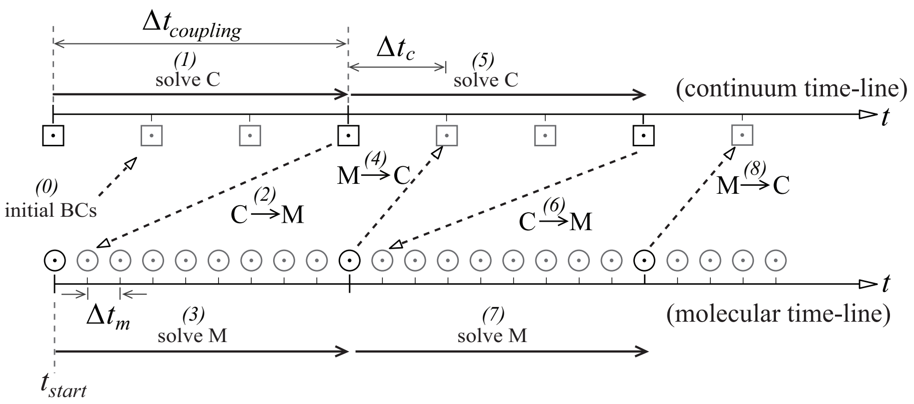
1. At the start of the simulation, apply initial boundary conditions on the continuum meshes.
2. Advance all continuum meshes (sequentially) from t to $t + \Delta t_{coupling}$, by τ_C time-steps. Molecular meshes wait.
3. Apply C→M spatial boundary conditions for all coupling regions.
4. Advance molecular meshes (sequentially) from t to $t + \Delta t_{coupling}$, by τ_M time-steps. Continuum meshes wait.
5. Apply M→C spatial boundary conditions for all coupling regions.
6. Repeat the procedure from step 2 until the end of the simulation.

A disadvantage of this technique is the presence of a time-delay when passing M→C boundary conditions, which can be observed in Figure 6.9(a). The delay escalates with increasing $\Delta t_{coupling}$. One may alleviate this by an extrapolation of target properties from stored old-time values, or else select a small $\Delta t_{coupling}$ to reduce its effects. It is especially worth noting that this time-coupling scheme is suitable also for the Schwarz alternating method, and the above then represents one Schwarz iteration. In the Schwarz scheme, the hybrid simulation proceeds until convergence is attained, or when a maximum number of iterations is reached.

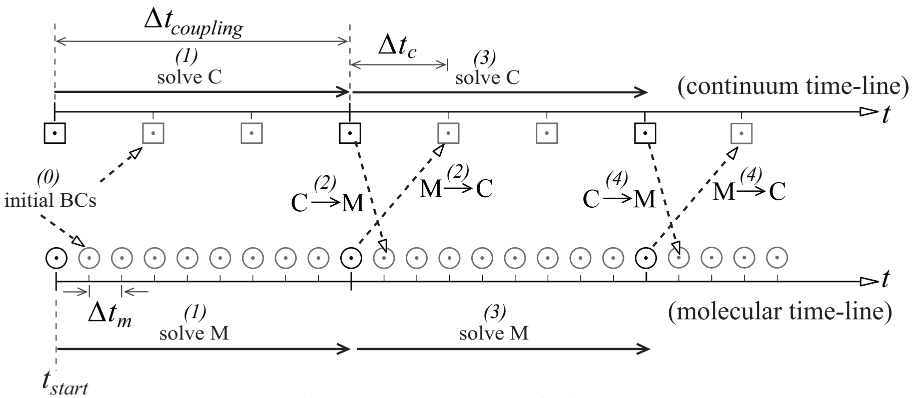
The *synchronous* scheme exchanges boundary conditions between solvers simultaneously, rather than at alternating times. This technique is described as follows (see Fig 6.9(b)):

1. Advance continuum meshes (sequentially) from t to $t + \Delta t_{coupling}$, by τ_C time-steps. Molecular meshes wait.
2. Advance molecular meshes (sequentially) from t to $t + \Delta t_{coupling}$, by τ_M time-steps. Continuum meshes wait.
3. Apply C→M and M→C spatial boundary conditions for all coupling regions.
4. Repeat from step 1.

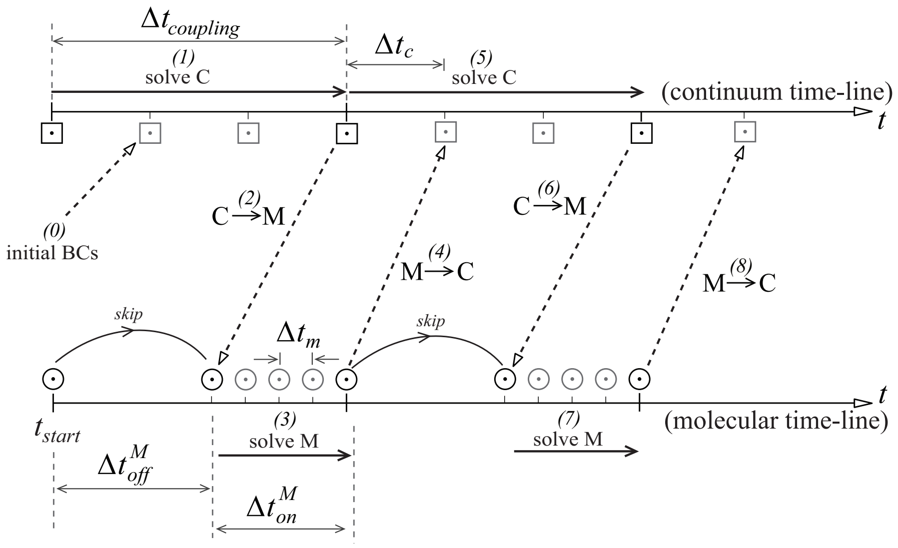
The drawback of this technique is that a time-delay of $\Delta t_{coupling}$ is now present for both M→C and C→M boundary conditions. A processor-weighting scheme may be introduced, so that molecular meshes are decomposed onto more processors than those assigned to the continuum meshes. In particular, this means that steps 1 and 2, are effectively run in parallel, so that waiting times may be reduced.



(a) Alternating technique



(b) Synchronous technique



(c) Time-decoupled technique

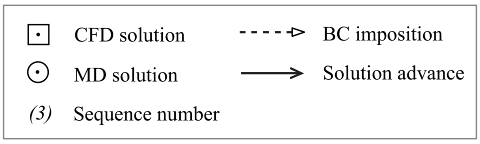


Figure 6.9: Figure showing the different time-schemes that may be used for hybridising MD and CFD meshes: (a) alternating, (b) synchronous, and (c) time-decoupled techniques.

Time-decoupled Sequential Technique

Computational-time costs may further be saved in a time-decoupled scheme, which allows the continuum formulation to proceed for the entire duration of the simulation, while the molecular solutions proceed for shorter disconnected time-intervals, thereby taking up only a fraction of the total run-time. Exchange of data still occurs at uniform coupling intervals, $\Delta t_{coupling} = \Delta t_{on}^{M} + \Delta t_{off}^{M}$, however the molecular meshes are left idle for a duration of Δt_{off}^{M} and switched on for Δt_{on}^{M} so to provide the necessary data to the continuum formulation. This time-decoupled scheme is illustrated in Figure 6.9(c), and is described as follows:

1. At the start of the simulation, apply initial boundary conditions on the continuum meshes.
2. Advance continuum meshes from t to $t + \Delta t_{coupling}$.
3. Molecular meshes are switched off from t to $t + \Delta t_{off}^{M}$.
4. Apply C→M spatial boundary conditions for all coupling regions.
5. Advance molecular meshes from $t + \Delta t_{off}^{M}$ to $t + \Delta t_{off}^{M} + \Delta t_{on}^{M}$. Continuum meshes wait.
6. Apply M→C spatial boundary conditions.
7. Repeat the time-decoupled scheme until the end of the simulation.

6.6 Setting up a hybrid simulation

We describe the procedure for setting up a general hybrid simulation:

1. *Hybrid Geometry* — The geometry of the hybrid domain is defined first. Complex systems are drawn in Pro/ENGINEER® and exported in STEP (.stp) format to GAMBIT®. The system-geometry is meshed, and imported to OpenFOAM. For simple cuboid domains, we prefer constructing and meshing of the computational domain using the `blockMesh` utility available with OpenFOAM.
2. *Sub-domains and sub-meshing* — The next step is for the domain geometry mesh to be divided into its respective set of MD and CFD sub-meshes. This step could have been carried out in the previous step, since different sub-domain geometries could have been defined and meshed separately. However, by using a sub-meshing utility in OpenFOAM, the coupling (or overlapping) regions (i.e., the cells) match across pairs of MD and CFD meshes.

3. *Coupling regions* — The coupling regions are built using methods which we describe in Appendix C.2. Building zones on the meshes is key to the flexibility of our technique and the application of coupling in arbitrary geometries [108].

4. *Initial conditions* — MD meshes are then filled with molecules using the utility described in Appendix C.1, and equilibrated for a short period of time ($t^* = 100$). The controllers are typically used to steer the system towards the target values.

5. *Coupling framework information* — Time and space coupling properties are defined within the couplingDict input file, per coupling region.

6. *Field measurements* — Measurement properties from MD meshes are defined in the fieldPropertiesDict input file.

7. *Run simulation* — The case is run in serial or decomposed into n_{proc} processor meshes, and run in parallel.

6.7 Test Cases and Results

We demonstrate the capability of this hybrid framework for a start-up two-wall shear flow problem, an oscillating wall shear flow problem, and a shear flow with a complex shaped coupling region.

6.7.1 Start-up Couette flow problem

The case geometry consists of two moving walls at the extremities of the domain, separated by a distance $L^* = 188$, that shear liquid argon residing between the walls. The domain decomposition applies MD at the two wall regions, and the compressible Navier-Stokes equations in the central part of the domain (see Fig 6.10). Two independent coupling regions are therefore required at the continuum-molecular interfaces, while periodic boundary conditions are applied in the $y-$ and $z-$directions. The shear viscosity in the entire continuum field is chosen to match closely that in the coupling region of the MD domain, i.e., $\eta = 0.899(\sqrt{\epsilon m}/\sigma^2)$, where the bulk state of the fluid is $\rho^* = 0.6$, and $T^* = 2.4$. The walls (A, B) are accelerated from rest using a ramp velocity function over a time-period $\Delta t^* = 20$, until a maximum velocity $\mathbf{u}_A^* = (0.0, -2.5, 0.0)$ and $\mathbf{u}_B^* = (0.0, 2.5, 0.0)$ respectively are reached. The walls are then maintained at uniform velocity.

A sequential time-coupling scheme is used at which C→M and M→C state-coupling is performed through velocity and density field properties at alternating intervals of $\Delta t^*_{coupling} = 10$. The time-steps for the continuum and molecular formulations are $\Delta t^*_c = 0.1$ and $\Delta t^*_m = 0.005$ respectively.

The simulation reaches a steady-state after $t^* = 4000$. In Figure 6.11 we show transient velocity profiles, while in Figures 6.12 and 6.13 we present density and pressure profiles at

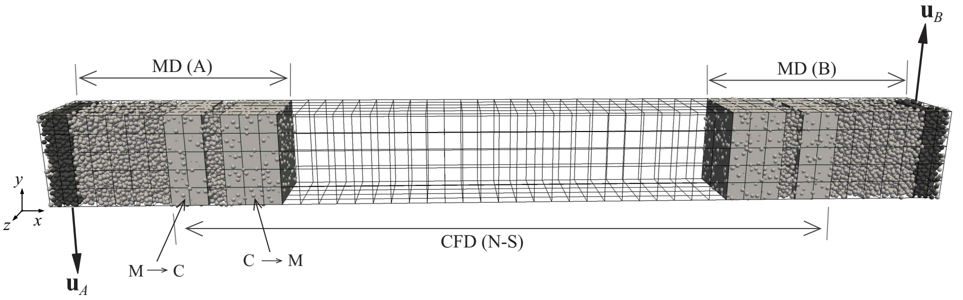


Figure 6.10: Hybrid domain decomposition for a shear flow case with two walls moving in opposite directions. Dark grey molecules represent wall molecules, while light grey molecules identify liquid argon molecules. Because the case is symmetric, profiles from the right hand side of the domain are only presented in subsequent figures.

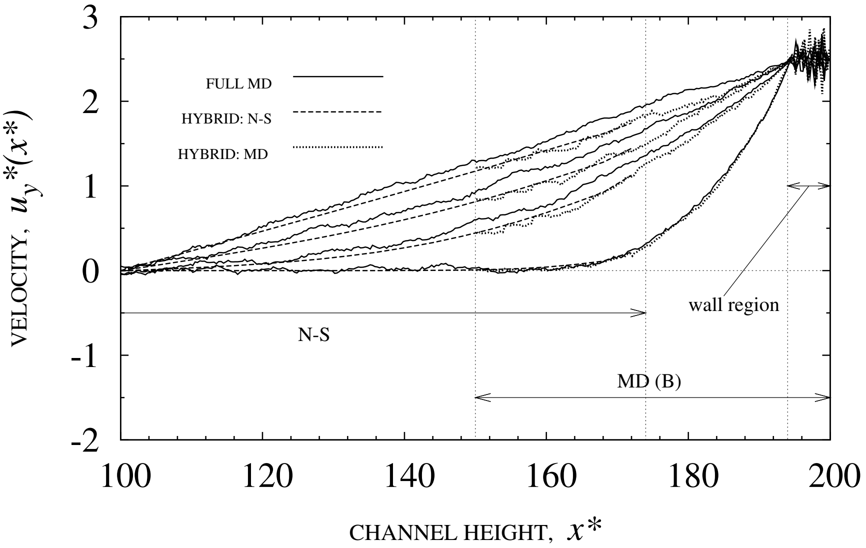


Figure 6.11: Transient velocity profiles for the start-up Couette flow problem, with a relaxation value of $\theta = 0.6$. Profiles are shown for times: $t^* = \{100, 500, 1000, 4000.\}$. Hybrid solutions are compared with the full molecular solution. The velocity measured within the wall region is non-zero because of trapped liquid molecules in a small gap located between the lattice wall and the periodic boundary. The velocity oscillations are therefore the result of low statistical sampling.

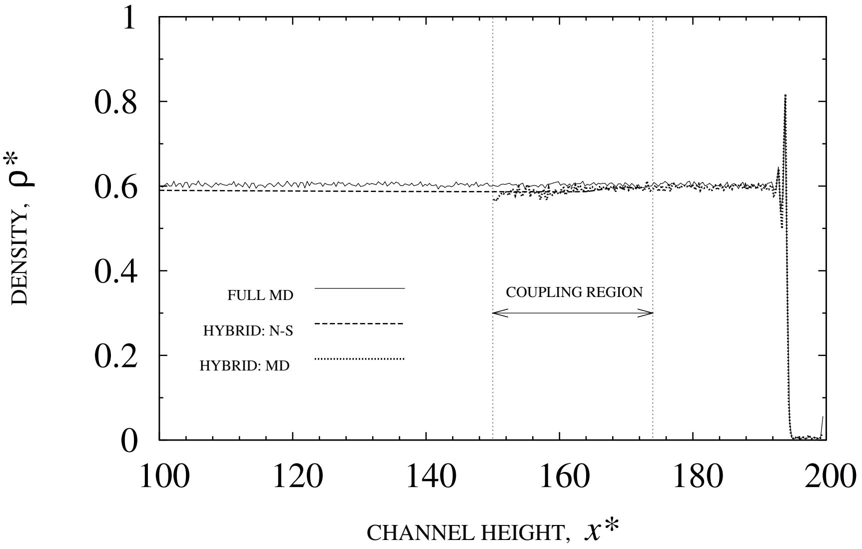


Figure 6.12: Steady-state density profiles for the Couette flow problem, with a relaxation value of $\theta = 0.6$. The hybrid solution is compared with the full molecular solution.

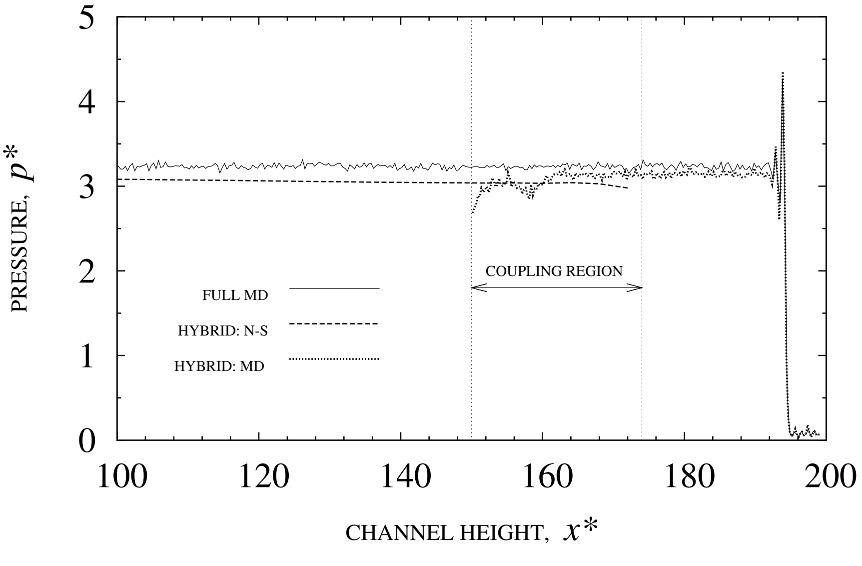


Figure 6.13: Steady-state pressure profiles for the Couette flow problem, with a relaxation value of $\theta = 0.6$. Note that no pressure coupling is employed in this study.

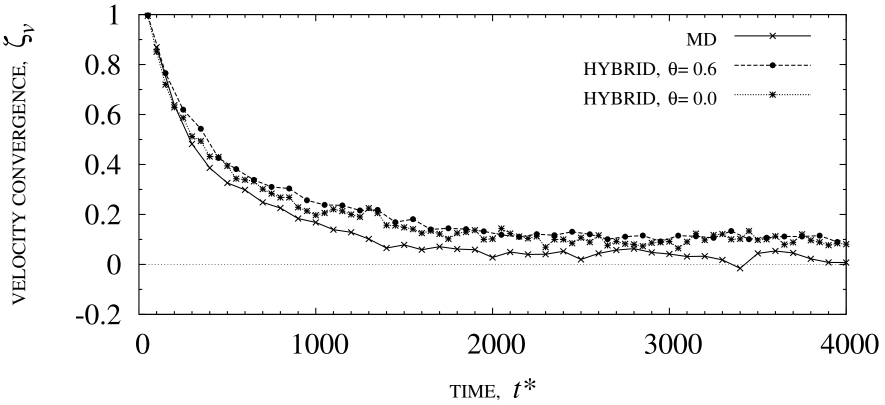


Figure 6.14: The convergence of the velocity solution at $x^* = 172$, given by $\zeta_v = |(u^o_y - u^*_y)/u^o_y|$, where $u^o_y = 1.915(\sqrt{\epsilon/m})$ is the linear interpolated value.

the end of the simulation. Hybrid results consist of a combination between MD and CFD solutions. We see that the coupling method is implemented correctly since the values and gradients of properties within the overlapping region match each other accurately. Hybrid results are also compared with full MD simulations of the same case. A relaxation value of $\theta = 0.6$ was used in the hybrid solutions. Without relaxation ($\theta = 0$), results showed tighter coupling, which is especially noticeable in the initial transient part of the simulation, see Figure 6.14. While the relaxation rate may introduce time-delays in the transient solution, slight discrepancies between full MD and the hybrid solutions are still observed in the non-relaxed case. To explain this, recall that the coupling region is present solely to transmit flow of properties mutually between the sub-domains through a method of boundary conditions imposition. Consequently, the fluctuations that are inherent in the MD sub-domain are usually enough to alter the solution in the continuum sub-domain. This changes the solution in the continuum sub-domain, which in turn propagates back into the molecular sub-domain. Solution oscillation in the coupling region may be reduced by increasing θ (to reduce the rate of change of property imposition) or/and increasing $\Delta t_{coupling}, \Delta X$ (to increase the accuracy of the measured property, thus reducing the amplitude of fluctuations).

Slight mismatch between the shear viscosity used in the continuum part of the formulation (obtained from Ref. [7]), as compared with the actual viscosity in the MD part of the formulation, also explains the time-delay in the hybrid solution. Coupling of shear-viscosity in the M→C direction would be useful to miminise these time-delay errors, albeit measurement of it is computationally intensive.

The simulations were carried out in serial and in parallel; two processors are sufficient since the case we consider is relatively small (the processor decomposition of the shear flow case can be seen in Figure 6.15). Excellent agreement between the transient velocities captured from both setups may be observed in Figure 6.16. Comparisons of the timings of the simulations are listed in Table 6.2, while percentage computational costs of the

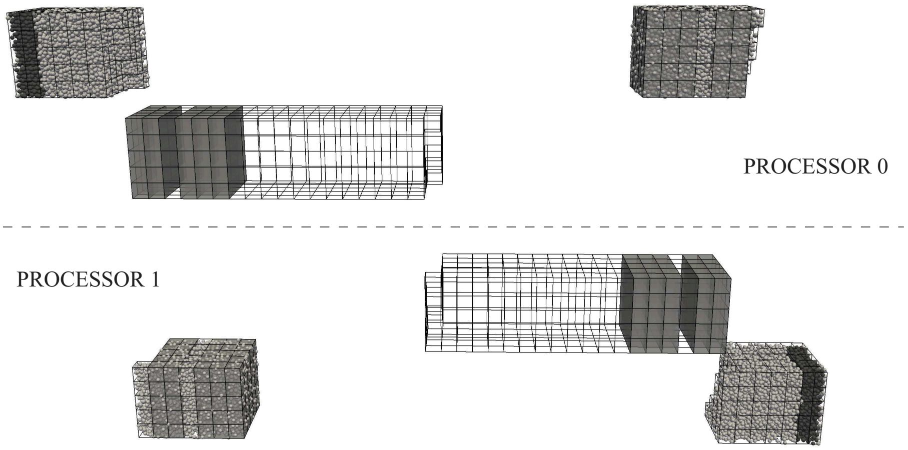


Figure 6.15: Processor-mesh decomposition and sub-meshing of MD and CFD sub-domains, for a two-processor Couette flow job, cf Figure 6.10. Note that the overlapping regions reside on different processors, thus processor communication is necessary for coupling of information.

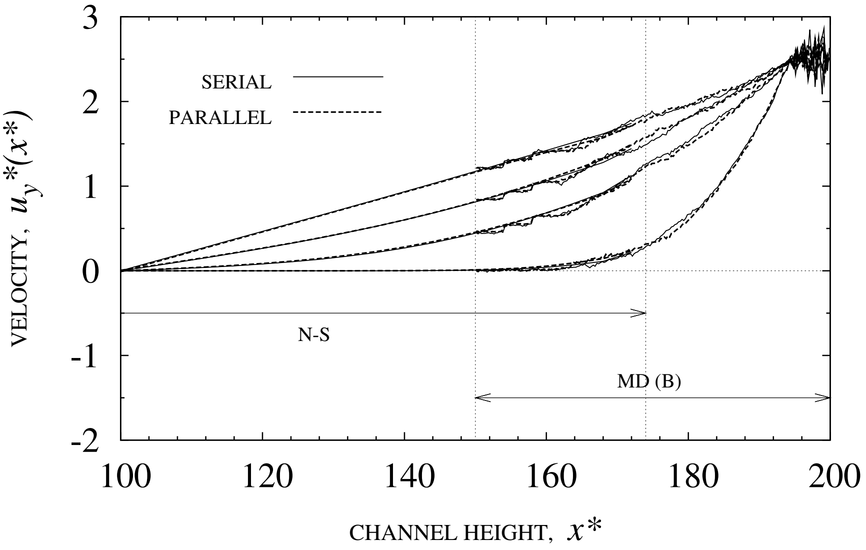


Figure 6.16: Transient velocity profiles for the start-up Couette flow problem. Comparison is made between serial and parallel processing results.

separate components in our hybrid simulation are reported in Table 6.3. From these tables it is clear that a hybrid approach shows promising processing savings, however it is still completely burdened by the cost of the MD component.

Table 6.2: Timings for the three shear flow cases, for a common problem time of $t^* = 4000$. Comparisons of the computational times need to take into account the number of molecules (that is, the extent of hybrid savings) and the number of processors.

Case	No. of processors	No. of molecules	Computational time (s)
Full-MD	8	$\sim 48,000$	393,512
Hybrid – serial	1	$\sim 24,000$	555,207
Hybrid – parallel	2	$\sim 24,000$	487,262

Table 6.3: Percentage computational cost of the individual components in the hybrid simulation. The cost of transferring boundary conditions between solvers is negligible.

	Computational cost
MD solver	99.89 %
CFD solver	0.11 %
Boundary conditions	–

The shear flow case is repeated for different domain lengths, in order to show that convergence to steady-state of a transient system is dependent on its size (Fig 6.17). The molecular sub-domain size is kept the same as in the previous case and we increase only the continuum sub-domain length, while keeping the strain rate the same ($\dot{\gamma} = 0.025(\sqrt{\epsilon/m\sigma^2})$). This behaviour shows that larger systems — in particular continuum sub-domains spanning into the micro-scales — would be still too expensive to model using a hybrid MD-continuum simulation, since the molecular sub-domain would need to be solved over very long time-scales. This is a significant drawback of the alternating time-coupled framework which we used in these case studies. Perhaps a more suitable approach for larger cases is the HMM, which uses a decoupled time- and space- framework. Furthermore, for such large cases one cannot compare solutions with a full MD solution, because of the computational expense.

6.7.2 Simple oscillating shear flow

A hybrid simulation of an oscillating shear flow problem is carried out to demonstrate that the hybrid algorithm may be applied to unsteady cases. The hybrid domain decomposition consists of an MD sub-domain, which encompasses the oscillating molecular wall and adjacent layer of molecular liquid, coupled to a Navier-Stokes continuum solver with no-slip boundary conditions applied at the other end of the domain (i.e. at $x = 100\sigma$). The molecular wall moves with an oscillating motion $\mathbf{u}_{wall}(t) = \mathbf{u}_{max}\sin(2\pi t)/\tau_T$, where $\mathbf{u}^*_{max} = (0, 10.0, 0)$ is the maximum velocity, and $\tau^*_T = 200$ is the time-period of the oscillatory wave.

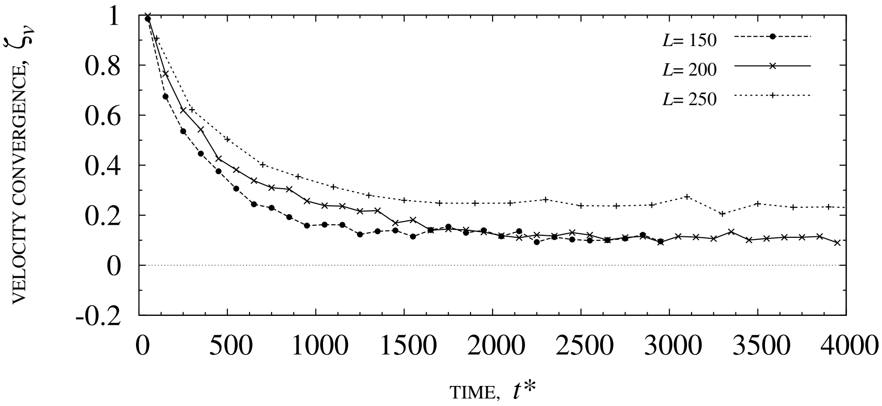


Figure 6.17: Velocity convergence for the same shear flow problem, using different lengths for the domain $L^* = \{150, 200, 250\}$ but equivalent shear rates. Note that the MD sub-domains have remained the same size for all cases, so changes in length were directly applied to the continuum sub-domain only.

In Figures 6.18 and 6.19 we show velocity profiles at different times in one oscillatory cycle, and see that good agreement is achieved between the hybrid and full-MD case.

6.7.3 Hybrid shear-flow with a complex coupling region

We can further demonstrate the capabilities of our hybrid model by constructing coupling regions in complex geometries. The case we choose is an isothermal shear-flow (Couette-type) that has a complex protrusion in the stationary wall (cf Fig 6.6). The wall and adjacent layer of liquid is modelled by MD, while the rest of the domain and moving wall is simulated by CFD. A moving wall velocity of $\mathbf{u}_w^* = (0.5, 0, 0)$, with a no-slip boundary condition, is applied to the top boundary of the continuum domain, and cyclic boundary conditions are applied in the other two directions. At the molecular-continuum interface, a 3D overlap region enables coupling between CFD and MD formulations at regular time-intervals of the simulation.

Both the hybrid and full-MD simulations are solved for a duration of $t^* = 2000$, and in parallel on two processors so that timings can be compared. The hybrid simulation is approximately two times faster than the full-MD simulation. Speed-up may also be estimated from the term $N^{\mathrm{FM}}/N^{\mathrm{H}}$, where $N^{\mathrm{H}} = 34,394$ and $N^{\mathrm{FM}} = 68,944$ are the average number of molecules in the hybrid MD sub-domain and the full-MD domain respectively. Steady-state results are shown in Figure 6.20, where general qualitative agreement of the velocity field vector solution is observed between the full-MD and hybrid MD-CFD simulations. Although we show in Figure 6.21 that the coupling procedure is implemented accurately, there are discrepancies of approximately 17% between the full MD and hybrid steady-state solutions when compared quantitatively. The sources of these discrepancies are the same as noted in §6.7.1.

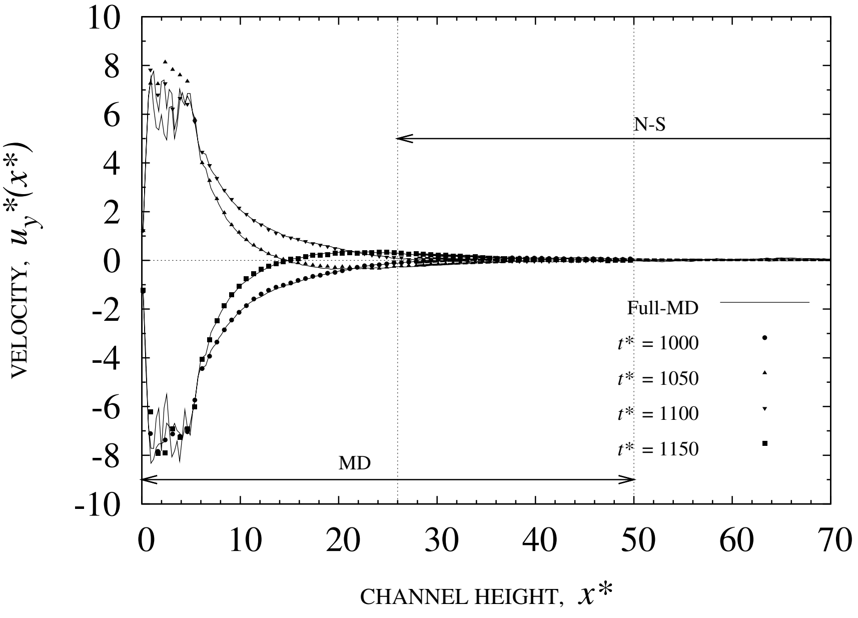


Figure 6.18: Velocity profiles within one cycle ($\tau_T^* = 200$) of the simple oscillatory-wall shear case. Comparisons are made with a full MD simulation; cf Figure 6.19.

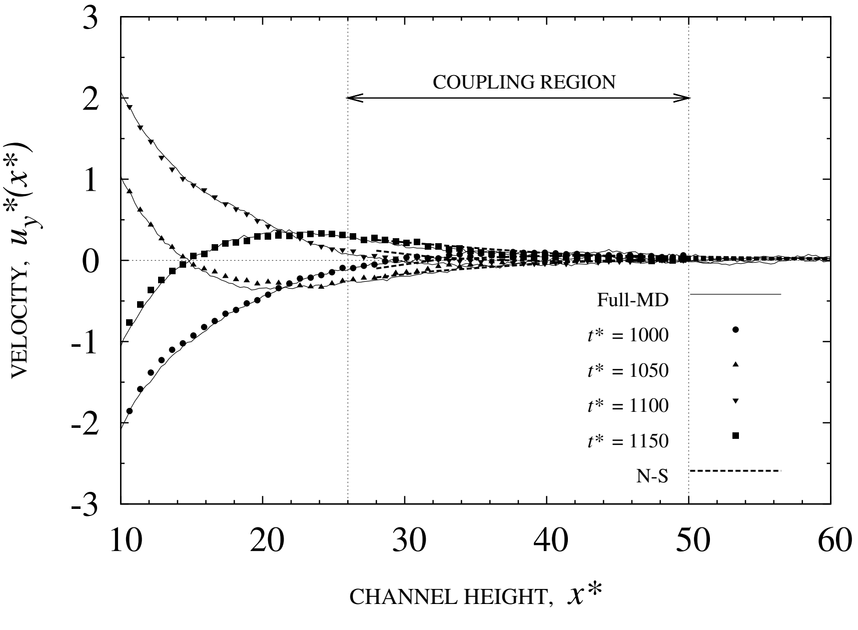
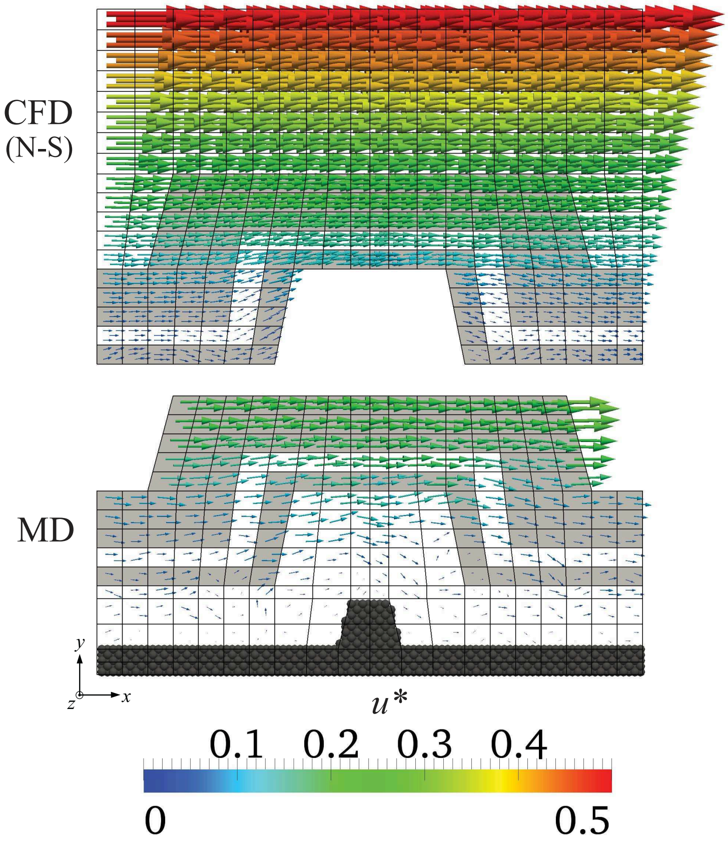
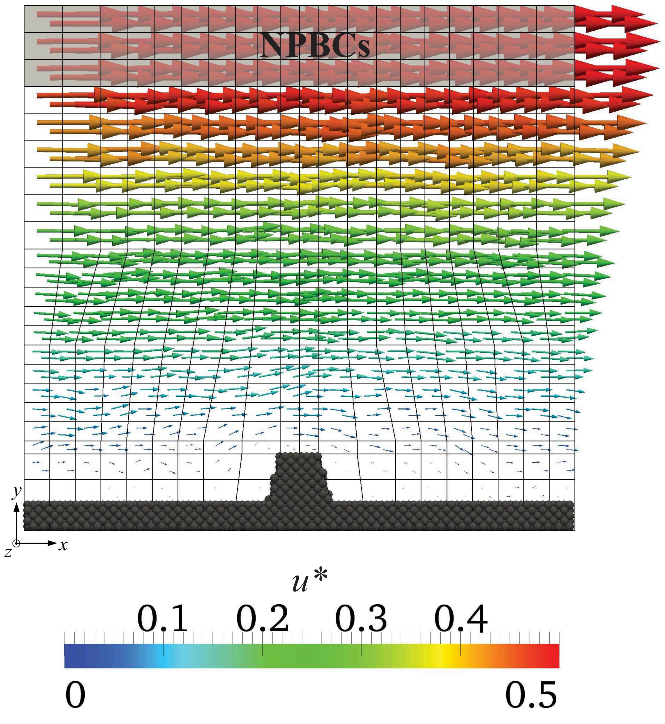


Figure 6.19: Velocity profiles in the coupling region, for the simple oscillating shear flow case.



(a) HYBRID SOLUTION



(b) FULL MD SOLUTION

Figure 6.20: Velocity vector fields: (a) hybrid (top – N-S solution; bottom – MD solution) and (b) full-MD simulations of the same complex case.

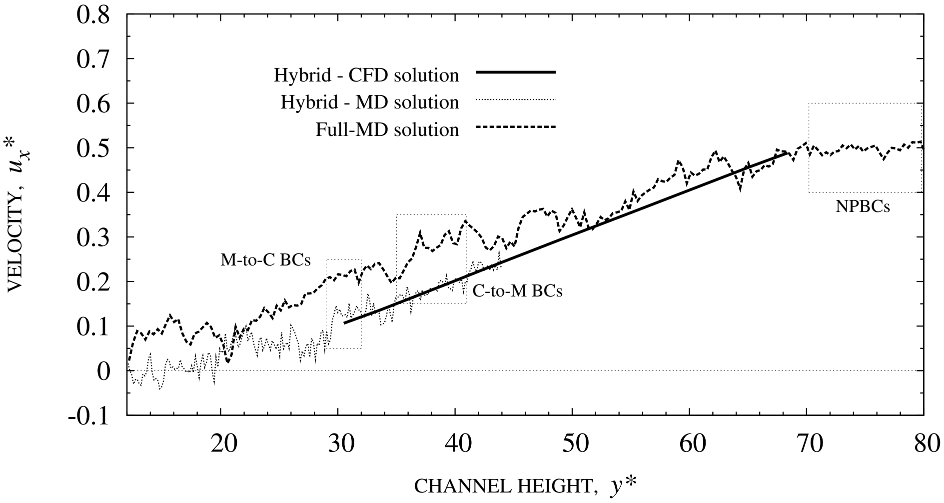


Figure 6.21: Comparisons of the velocity profile in the y-direction starting from the wall protrusion seen in Figure 6.20. Although accurate coupling between continuum and molecular sub-domains are observed within the coupling region, discrepancies between the full-MD and hybrid solution are evident throughout the computational domain. Hybrid simulations with varying densities give similar results, but convergence is slower for fewer number of molecules within the system due to the low viscosity, however still computationally less costly.

Chapter 7

Conclusions

This is not the end.
It is not even the beginning of the end.
But it is, perhaps, the end of the beginning.
- Winston Churchill

This thesis has presented the development of non-periodic boundary conditions within an existing non-equilibrium molecular dynamics code in the OpenFOAM software, and shown their application in a new hybrid continuum-MD code. The objectives were to understand and develop MD methods for arbitrary three-dimensional geometries, and to run simulations in parallel on distributed processors. We have shown in this research that pure-MD or hybrid simulations of liquids can be carried out in more realistic engineering geometries, rather than just in cuboid-shaped domains. The process of setting up a simulation case generally involves the exporting of a part-geometry drawn in CAD software. This is followed by a meshing procedure for which any unstructured polyhedral mesh is defined. The mesh is subsequently filled with molecules using appropriate pre-processing tools, segmented into MD and CFD meshes for a hybrid simulation, and decomposed into smaller sub-domains for parallel deployment.

Imposing boundary conditions on the molecular dynamics domain from the continuum solution is still an open and daunting problem. In lieu of this, we implemented a series of methods for imposing locally- and temporally-varying continuum properties at the boundary of a molecular dynamics mesh. We described controllers that converge state properties, such as velocity, density, temperature and pressure, within individual three-dimensional cells of a mesh by using a feedback control loop. One key problem which we encountered using this approach is the dependency on the statistical accuracy of the sampled (feedback) measurement of the cell-size and averaging-time period. To alleviate this issue and its effect on the resolution of control, we implemented a method of coarse-grained measurement, whereby samples are also taken from the neighbourhood of an arbitrary control cell. This led to a reduction in the time-averaging period and hence a tighter control procedure.

Small signal-to-noise ratios occur more pronouncedly in low speed flows. For instance, to obtain an accurate velocity measurement, one needs to allow a longer averaging time interval, which limits the temporal variation of control and also increases the computational cost of the simulation. It is for this reason that larger (typically unrealistic) Reynolds number flows are preferred for control of MD or similarly for coupling with continuum formulations. Undesired spatial variations (or oscillations) of hydrodynamic properties may be rectified using our controller method as long as the size of the cell is smaller than the amplitude of any spatial variation. The problem of pressure and density control in local regions has been tackled using insertion/deletion of molecules in order to match the target cell values. This technique has proved to be very effective, albeit computationally intensive since it needs to identify sites within the existing molecular field in accordance with the potential energy landscape. The processing intensity for this controller was higher when properties such as density increased, temperature decreased, and the number of control cells increased (that is, the number of molecules to control). As a result, this limited the size of the boundary region per processor mesh.

In this research we also implemented other continuum-type models at the boundary, including a series of mass flux models, stochastic models and new force field models. These models were implemented to provide flow of mass, momentum and energy from the edge of the MD domain, and produce more realistic, non-fluctuating hydrodynamic profiles next to the controlling regions. All our methods were designed for applying non-periodic boundary conditions (NPBC) at arbitrary shaped boundaries and for running in parallel. The importance of choosing appropriate boundary models and their operating parameters, vis-à-vis the underlying physics of the case in question, cannot be overstated. For example, different continuum properties need to be measured and controlled at different frequencies. Temperature needs to be controlled over shorter time-scales than velocity, in order to remove temperature drifts that arise due to accelerated molecules. Density needs to be controlled over time-scales shorter than the characteristic time of the flow velocity, so that molecules do not deplete the cell. Inlets and outlets of nano-channels require different NPBCs; inlets are kept at constant state properties, while outlets are modelled completely by a flux method.

We applied NPBCs to MD simulations and verified them by comparing hydrodynamic profiles with fully-periodic simulations for equilibrium, uniform flow and 1D shear flow. Furthermore, a new method was presented for applying pressure gradients in arbitrary-shaped nano-channels in MD simulations. Notably, the main advantage of our method is that the properties at the inlets and outlets, as well as the pressure difference across the channel, are controllable. The method was tested on a three-dimensional complex mixing channel with three inlets and one outlet. Fluids of different species flowed separately at the inlets and were forced to mix at the throat of the channel. Different LJ and WCA potentials were investigated, and the velocity boundaries were varied until reasonable mixing was observed at the outlet. The method was also tested on an extensional shear flow problem that consisted of a cross-shaped channel with two inlets and two outlets. Droplets travelled from one inlet of the channel and deformed in the complex velocity field at the throat. Deformation consisted of elongation of the droplet, and in most cases these broke up into smaller droplets. We showed that variation of the throat geometry gave different deformation and breakup characteristics. Droplets broke up consistently into two smaller droplets when the throat constriction was reduced substantially, but this resulted in the droplets sticking to the device surface when exiting the throat and outlet channel.

In this thesis, we have also presented a new hybrid method coupling MD with a Navier-Stokes continuum solver. The hybrid coupling framework consists of a multi-mesh domain-decomposition. Molecular and continuum meshes overlap at the interfaces, within which continuum-to-molecular and molecular-to-continuum time-relaxed boundary conditions are imposed at regular and alternate coupling times of the simulation run. In the continuum-to-molecular direction, continuum cell values from the CFD component replace target values in the NPBC description. In the molecular-to-continuum direction, coarse-grained field measurements derived from MD cells replace state boundary values on the continuum finite-volume mesh. The hybrid method was tested for transient shear flow cases, including shear of two molecular walls moving in different directions, and an oscillating wall. We also demonstrated a hybrid simulation of a shear flow case that required the coupling region to be of a complex (non-cuboid) shape. Our results showed that accurate coupling between MD and CFD solutions (both values and gradients) was achieved within the overlapping region. This was facilitated by employing two or more layers of cells for each coupling sub-region. Comparisons of the hybrid solutions with those from a full MD simulation have shown time-delays in the transient profiles. We explained how the errors were caused by the relaxation method employed when passing boundary conditions, together with any mismatch between the shear viscosity used by the CFD solution and the viscosity effects in the MD simulation. Furthermore, fluctuations in coarse-grained MD measurements were passed asymmetrically into the CFD sub-domain. These inaccuracies propagated back and forth between MD and CFD sub-domains because of the mutual coupling method, thus causing instabilities in the solution in the overlap region.

In transient cases, the time required to reach steady-state scales with the lengthscale of the flow domain. For a time-coupled hybrid method, the MD component therefore needs to be integrated over the entire simulation run. Although the hybrid methodology looks very promising, often reducing the processing time by more than half, more development work is now needed to make it feasible for large scale systems that bridge into the micro regime.

Our code has been carefully designed so that future development of new boundary models and new field measurements for passing boundary conditions in arbitrary geometries and in parallel, may be incorporated within the code and tested in a straightforward and flexible way. Our new models were coded once, in a low-level data structure. The hybrid solver is coded in a top-level structure and so new coupling strategies may be devised very quickly. This work was greatly facilitated by the versatile OpenFOAM toolbox and the C++ programming environment.

Future work

The following represents a discussion of future work:

1. *Decoupling resolution and statistical accuracy of measurements in cell-volumes* — Measurement of properties from the molecules within small cell-volumes, with low statistical scatter and sampled from short MD time-scales, is key for accurately converging continuum properties that involve feedback-loop control procedures. Additionally, such measurements are also required to impose low-noise tight-coupling on the continuum sub-domain in hybrid simulations. Although the problem of numerical errors inherent in measurements of molecular simulations has been alleviated in our research by using a method of spatial coarse-graining, it still requires more suitable solutions (e.g., advanced variance reduction techniques, wavelet analysis and other techniques from signal processing).

2. *Measurement framework* — The range of properties that have been implemented within our new measurement framework are limited to the requirements of the MD and hybrid simulations carried out in this thesis. Other measurement properties need to be incorporated within the framework e.g., correlation functions, estimation of errors in properties, and also measurements of new derived properties. A measurement post-processing utility also needs to be developed in OpenFOAM to compute measurable properties from Lagrangian fields that have been written out to hard disk after the simulation run is completed. This may reduce the computational time of the MD simulation and avoid the need for repeated simulations for sampling new properties that are forgotten the first time around.

3. *Advancements in non-periodic boundary conditions in MD*

   (a) *Density control* — Control of spatial density is a critical requirement in non-periodic MD (and hybrid) simulations. As future work, we suggest performance improvements for the current method when the MD simulation is executed in serial or in parallel. For example, when run in parallel, the search for conflicts and retrial steps, albeit essential to create a robust system, invoke many parallel communications thereby slowing down the simulation. We also suggest as future work the development of new and more efficient techniques with similar effect to our current method. An example is the one dimensional method proposed by Kotsalis *et al.* [51, 97] that seems to be more efficient than the insertion/deletion method which we use, because it applies forces to control the density error-gradient: molecules are forced out of the cell if the density error gradient is positive and forced into the cell if the gradient is negative. There are still many limitations for applying the method in arbitrary 3D geometries and in parallel. The main limitation is the relatively small cell-sizes which are required to reduce spatial variations of density, and the increase of the time-averaging period over which the cell density is accurately measured. This results in a lower frequency control strategy, which may be ineffective in the long run.

   (b) *New control schemes* — Various different control systems may be deployed within the continuum-to-MD controllers described in this thesis.

   (c) *Polyatomic fluids and electrostatics* — In extending the current boundary models to polyatomic systems, velocity and temperature control are typically straightforward tasks. However, density and pressure control (using the method described in this thesis) are more challenging because they require insertions of polyatomic molecules (translation and rotation) within the existing Lagrangian field of molecules [109]. New boundary models are also required to model long range electrostatics at non-periodic boundaries. These might be similar to the method we use to compute and impose the mean force field due to missing molecules beyond the boundary (§4.5.5).

4. *New applications and simulations* — To investigate many more engineering systems and applications, using our NPBC method in MD simulations. Two examples are the construction of nano-channel systems that simulate flows through carbon nanotube membranes for desalination and filtration applications, and the investigation of new mechanical methods for generating fluid pumping power in nano devices (e.g., methods similar to cilia in human lungs).

5. *Hybrid methods*

   (a) *Time-decoupled systems* — It is still too computationally intensive to apply the current hybrid method to relatively large unsteady systems, since the MD portion of the computational framework needs to be integrated over the entire time of the simulation run. Our methods, which we have implemented in this research, may however be applied in the Heterogeneous Multiscale Method (HMM) proposed by E and Ren [6], or for developing similar time-decoupled hybrid methods.

   (b) *Correctness of boundary conditions in coupled CFD-MD systems* — For systems in which the continuum sub-domain is embedded fully within the MD sub-domain, Dirichlet boundary conditions of the M→C variety were applied at opposite ends of the sub-domain. A mixture of flux- and state-coupling at each boundary, however, represents a better approach for compressible viscous flows. In view of this, new flux-based boundary conditions for both M→C and C→M directions need to be developed in the OpenFOAM code, so that they are incorporated within the hybrid simulations and results may then be compared with the current state-based method. In particular there is a need to analyse and optimise the mix of boundary conditions that need to be applied for different problems. The drawbacks of flux-based methods proposed in the literature have been discussed in §6.2.1, and it appears that more work still needs to be done, including solving the problems of noise, flux conservation and variable coupling region sizes.

   (c) *Continuum solver* — The limitations of the Navier-Stokes continuum solver used in this research, and for potential future work, are described in detail in §6.4.

   (d) *Hybrid parallel decomposition* — Progress is still required in optimising the parallel decomposition we employed for our multi-mesh hybrid approach. It is more suitable to devise a weighted scheme that assigns more processors to the MD meshes.

   (e) *Adaptive interface and continuum breakdown* — The problem of identifying continuum breakdown regions, or the opposite (i.e. regions where MD can be replaced by a continuum approach), is still not well-characterised. The location of the molecular-continuum interface for liquids is generally based on distance from interfaces, e.g., 20σ away from a molecular wall. In future work, the hybrid code could be redesigned to handle automatic adaptive motion of the interfaces, based on continuum-breakdown parameters and switching criteria. Examples where this kind of algorithm is required include moving contact lines, mixing, diffusion of fluid interfaces, and modelling the motion of macromolecules through nanochannels. Implementation and programming are major issues in this future development.

   (f) *Multi-solver coupling framework* — To generalise the current two-component hybrid framework in order to couple any arbitrary number of components. Different component solvers may include: Monte Carlo, polyatomic MD with electrostatics, Coarse-Grained MD, Quantum Mechanics and Finite Element Analysis.

6. *Indirect hybrid methods* — To develop more efficient and accurate engineering models by using data obtained from *a priori* MD simulations. An example is to use the Poiseuille flow method described in this thesis to extract phenomenological parameters that describe the phenomena occurring next to molecular walls, which are in turn used as boundary conditions for a CFD solver. The latter method may require a boundary ‘layered’ approach.

References

[1] E. Matteoli and G. Ali Mansoori. A simple expression for radial distribution functions of pure fluids and mixtures. *Journal of Chemical Physics*, 103(11): 4672–4677, 1995.

[2] M. Sun and C. Ebner. Molecular-dynamics simulation of compressible fluid flow in two-dimensional channels. *Physical Review A*, 46(8): 4813–4818, 1992.

[3] D. E. Hertzog, B. I., B. Mohammadi, O. Bakajin and J. G. Santiago. Optimization of a microfluidic mixer for studying protein folding kinetics. *Journal of Analytical Chemistry*, 78(13): 4299–4306, 2006.

[4] S. T. O'Connell and P. A. Thompson. Molecular dynamics–continuum hybrid computations: A tool for studying complex fluid flows. *Physical Review E*, 52: R5792–R5795, 1995.

[5] R. Delgado-Buscalioni and P. V. Coveney. Continuum-particle hybrid coupling for mass, momentum, and energy transfers in unsteady fluid flow. *Physical Review E*, 67(4): 046704, 2003.

[6] W. Ren and W. E. Heterogeneous multiscale method for the modeling of complex fluids and micro-fluidics. *Journal of Computational Physics*, 204(1): 1–26, 2005.

[7] R. L. Rowley and M. M. Painter. Diffusion and viscosity equations of state for a Lennard-Jones fluid obtained from molecular dynamics simulations. *International Journal of Thermophysics*, 18(5): 1109–1121, 1997.

[8] E. G. Flekkøy, G. Wagner and J. Feder. Hybrid model for combined particle and continuum dynamics. *Europhysics Letters*, 52(3): 271–276, 2000.

[9] R. P. Feynmann. There's plenty of room at the bottom. Available online: www.zyvex.com/nanotech/feynman.html.

[10] S. U.-S. Choi. Nanofluid technology: current status and future research. *Korea-U.S. Technical Conference on Strategic Technologies*. 1998.

[11] K. P. Travis and K. E. Gubbins. Poiseuille flow of Lennard-Jones fluids in narrow slit pores. *Journal of Chemical Physics*, 112(4): 1984–1994, 2000.

[12] M. C. Roco. Reviews of national research programs in nanoparticle and nanotechnology research in the U.S.A. *Journal of Aerosol Science*, 29(5–6): 749–760, 1998.

[13] M. Whitby and N. Quirke. Fluid flow in carbon nanotubes and nanopipes. *Nature Nanotechnology*, 2: 87 – 94, 2007.

[14] D. Mattia and Y. Gogotsi. Review: static and dynamic behavior of liquids inside carbon nanotubes. *Microfluidics and Nanofluidics*, 5(3), 2008.

[15] D. C. Rapaport. Microscale swimming: The molecular dynamics approach. *Physical Review Letters*, 99(23): 238101, 2007.

[16] M. Gad-el-Hak. Liquids: The holy grail of microfluidic modeling. *Physics of Fluids*, 17(10): 100612, 2005.

[17] W. Loose and S. Hess. Rheology of dense model fluids via nonequilibrium molecular dynamics: Shear thinning and ordering transition. *Rheologica Acta*, 28(2): 91–101, 1989.

[18] K. V. Sharp, R. J. Adrian, J. G. Santiago and J. I. Molho. Liquid flows in microchannels, in *MEMS: Introduction and Fundamentals*, edited by M. Gad-el-hak (Taylor and Francis, 2nd edition). pages 10–1, 2006.

[19] OpenFOAM: The open source CFD toolbox. Available online: www.openfoam.org.

[20] G. Macpherson. *Molecular Dynamics Simulation in Arbitrary Geometries for Nanoscale Fluid Mechanics.* Ph.D. thesis, University of Strathclyde, Glasgow, 2008.

[21] D. C. Rapaport. *The Art of Molecular Dynamics Simulation.* Cambridge University Press, 2nd edition, 2004.

[22] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids.* Oxford University Press, 1987.

[23] M. Gad-el-Hak. The fluid mechanics of microdevices — The Freeman Scholar lecture. *Journal of Fluids Engineering*, 121: 5–33, 1999.

[24] G. B. Macpherson and J. M. Reese. Molecular dynamics in arbitrary geometries: Parallel evaluation of pair forces. *Molecular Simulation*, 34(1): 97–115, 2008.

[25] Open mpi: Open source high performance computing. Available online: www.open-mpi.org.

[26] G. B. Macpherson, N. Nordin and H. G. Weller. Particle tracking in unstructured, arbitrary polyhedral meshes for use in CFD and molecular dynamics. *Communications in Numerical Methods in Engineering*, 25(3): 263–273, 2009.

[27] G. B. Macpherson, M. K. Borg and J. M. Reese. Generation of initial molecular dynamics configurations in arbitrary geometries and in parallel. *Molecular Simulation*, 33(15): 1199–1212, 2007.

[28] Software package and libraries for sequential and parallel graph partitioning, static mapping, and sparse matrix block ordering, and sequential mesh and hypergraph partitioning. Available online: www.labri.fr/perso/pelegrin/scotch/.

[29] M. W. Tysanner and A. L. Garcia. Measurement bias of fluid velocity in molecular simulations. *Journal of Computational Physics*, 196: 173–183, 2004.

[30] B. D. Todd and D. J. Evans. Temperature profile for Poiseuille flow. *Physical Review E*, 55(3): 2800–2807, 1997.

[31] J. H. Irving and J. G. Kirkwood. The statistical mechanical theory of transport processes. iv. the equations of hydrodynamics. *Journal of Chemical Physics*, 18(6): 817–829, 1950.

[32] P. J. Daivis, K. P. Travis and B. D. Todd. A technique for the calculation of mass, energy, and momentum densities at planes in molecular dynamics simulations. *Journal of Chemical Physics*, 104(23): 9651–9653, 1996.

[33] B. D. Todd, D. J. Evans and P. J. Daivis. Pressure tensor for inhomogeneous fluids. *Physical Review E*, 52: 1627–1638, 1995.

[34] B. D. Todd, P. J. Daivis and D. J. Evans. Heat flux vector in highly inhomogeneous nonequilibrium fluids. *Physical Review E*, 51: 4362–4368, 1995.

[35] B. D. Todd and D. J. Evans. The heat flux vector for highly inhomogeneous nonequilibrium fluids in very narrow pores. *Journal of Chemical Physics*, 103: 9804–9809, 1995.

[36] H. Okumura and D. M. Heyes. Comparisons between molecular dynamics and hydrodynamics treatment of nonstationary thermal processes in a liquid. *Physical Review E*, 70(6): 061206, 2004.

[37] N. Hadjiconstantinou, A. Garcia, M. Bazant and G. He. Statistical error in particle simulations of hydrodynamic phenomena. *Journal of Computational Physics*, 187(1): 274–297, 2003.

[38] T. Werder, J. H. Walther and P. Koumoutsakos. Hybrid atomistic-continuum method for the simulation of dense fluid flows. *Journal of Computational Physics*, 205(1): 373–390, 2005.

[39] J. Powles, G. Rickayzen and D. Heyes. Temperatures: old, new and middle aged. *Molecular Physics*, 103(10): 1361–1373(13), 2005.

[40] J. Li, D. Liao and S. Yip. Coupling continuum to molecular-dynamics simulation: Reflecting particle method and the field estimator. *Physical Review E*, 57: 7259–7267, 1998.

[41] J. Eapen, J. Li and S. Yip. Statistical field estimators for multiscale simulations. *Physical Review E*, 72(5): 056712–+, 2005.

[42] H. Brenner and V. Ganesan. Molecular wall effects: Are conditions at a boundary "boundary conditions"? *Physical Review E*, 61(6): 6879–6897, 2000.

[43] P. Attard. Non-periodic boundary conditions for molecular simulations of condensed matter. *Molecular Physics*, 104: 1951–1960, 2006.

[44] C. L. Brooks III, M. Karplus and B. M. Pettitt. *Advances in Chemical Physics, Volume 71, Proteins: A Theoretical Perspective of Dynamics, Structure, and Thermodynamics*. Wiley-Interscience, 1988.

[45] C. L. Brooks III, A. Brünger and M. Karplus. Active site dynamics in protein molecules: A stochastic boundary molecular-dynamics approach. *Biopolymers*, 24(5): 843–865, 1985.

[46] A. P. Thompson and G. S. Heffelfinger. Direct molecular simulation of gradient-driven diffusion of large molecules using constant pressure. *Journal of Chemical Physics*, 110(22): 10693, 1999.

[47] X. B. Nie, S. Y. Chen, W. E and M. O. Robbins. A continuum and molecular dynamics hybrid method for micro- and nano-fluid flow. *Journal of Fluid Mechanics*, 500: 55–64, 2004.

[48] J. Koplik, J. R. Banavar and J. F. Willemsen. Molecular dynamics of Poiseuille flow and moving contact lines. *Physical Review Letters*, 60: 1282–1285, 1988.

[49] J. Koplik, J. R. Banavar and J. F. Willemsen. Molecular dynamics of fluid flow at solid surfaces. *Physics of Fluids A*, 1(5): 781–794, 1989.

[50] A. W. Lees and S. F. Edwards. The computer study of transport processes under extreme conditions. *Journal of Physics C (Solid State Physics)*, 5: 1921–1928, 1972.

[51] E. M. Kotsalis, J. H. Walther and P. Koumoutsakos. Control of density fluctuations in atomistic-continuum simulations of dense liquids. *Physical Review E*, 76: 016709, 2007.

[52] G. Ciccotti and A. Tenenbaum. Canonical ensemble and nonequilibrium states by molecular dynamics. *Journal of Statistical Physics*, 23(6): 767–772, 1980.

[53] M. Berkowitz and J. A. McCammon. Molecular dynamics with stochastic boundary conditions. *Chemical Physics Letters*, 90(215), 1982.

[54] C. L. Brooks III and M. Karplus. Deformable stochastic boundary conditions in molecular dynamics. *Journal of Chemical Physics*, 79(12): 6312–6325, 1983.

[55] A. Brunünger, C. L. Brooks III and M. Karplus. Stochastic boundary conditions for molecular dynamics of ST2 water. *Chemical Physics Letters*, 105(5): 495–500, 1984.

[56] R. Eva-Stina, M. J. Manuel and K. Lars. An evaluation of non-periodic boundary condition models in molecular dynamics simulations using prion octapeptides as probes. *Journal of molecular structure. Theochem*, 760: 91–98, 2006.

[57] X. Nie, S. Chen and M. O. Robbins. Hybrid continuum-atomistic simulation of singular corner flow. *Physics of Fluids*, 16: 3579–3591, 2004.

[58] E. M. Kotsalis and P. Koumoutsakos. A control algorithm for multiscale simulations of liquid water. *ICCS '08: Proceedings of the 8th international conference on Computational Science, Part II*, pages 234–241. Springer-Verlag, Berlin, Heiderlberg, 2008.

[59] G. S. Heffelfinger and F. van Swol. Diffusion in Lennard-Jones fluids using dual control volume grand canonical molecular dynamics simulation (DVC-GCMD). *Journal of Chemical Physics*, 100(10): 7548–7552, 1994.

[60] R. Delgado-Buscalioni and P. V. Coveney. USHER: An algorithm for particle insertion in dense fluids. *Journal of Chemical Physics*, 119(2): 978–987, 2003.

[61] D. M. Huang, C. Sendner, D. Horinek, R. R. Netz and L. Bocquet. Water slippage versus contact angle: A quasiuniversal relationship. *Physical Review Letters*, 101(22): 226101, 2008.

[62] H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. Dinola and J. R. Haak. Molecular dynamics with coupling to an external bath. *Journal of Chemical Physics*, 81: 3684–3690, 1984.

[63] H. C. Andersen. Molecular dynamics simulations at constant pressure and/or temperature. *Journal of Chemical Physics*, 72(4): 2384–2393, 1980.

[64] N. Hadjiconstantinou and A. Patera. Heterogeneous atomistic-continuum methods for dense fluid systems. *International Journal of Modern Physics C*, 8(4): 967–976, 1997.

[65] M. Kalweit and D. Drikakis. Coupling strategies for hybrid molecular-continuum simulation methods. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 222(5): 797–806, 2008.

[66] P. Spijker. *Exploring the Boundaries of Molecular Modeling: A Study of Nanochannels and Transmembrane Proteins.* Ph.D. thesis, Eindhoven University of Technology, 2009.

[67] W. H. Press, W. T. Vetterling, S. A. Teukolsky and B. P. Flannery. *Numerical Recipes in C++: The Art of Scientific Computing.* Cambridge University Press, 4th edition, 2002.

[68] K. Ogata. *Modern Control Engineering.* Prentice Hall, 4th edition, 2002.

[69] S. A. Somers and H. T. Davis. Microscopic dynamics of fluids confined between smooth and atomically structured solid surfaces. *Journal of Chemical Physics*, 96(7): 5389–5407, 1992.

[70] J. Koplik and J. R. Banavar. Continuum deductions from molecular hydrodynamics. *Annual Review of Fluid Mechanics*, 27: 257–292, 1995.

[71] K. P. Travis, B. D. Todd and D. J. Evans. Poiseuille flow of molecular fluids. *Physica A: Statistical and Theoretical Physics*, 240(1-2): 315 – 327, 1997. Proceedings of the Euroconference on the microscopic approach to complexity in non-equilibrium molecular simulations.

[72] D. E. Hertzog, X. Michalet, M. Jäger, X. Kong, J. G. Santiago, S. Weiss and O. Bakajin. Femtomole mixer for microsecond kinetic studies of protein folding. *Journal of Analytical Chemistry*, 76(24): 7169–7178, 2004.

[73] E. B. Dussan. On the spreading of liquids on solid surfaces: Static and dynamic contact angles. *Annual Review of Fluid Mechanics*, 11: 371–400, 1979.

[74] P. A. Thompson and M. O. Robbins. Simulations of contact-line motion: Slip and the dynamic contact angle. *Physical Review Letters*, 63: 766–769, 1989.

[75] R. Delgado-Buscalioni, P. Coveney, G. Riley and R. Ford. Hybrid molecular-continuum fluid models: implementation within a general coupling framework. *Philosophical Transactions of the Royal Society London*, 363(1833): 1975–1985, 2005.

[76] P. V. Coveney, G. D. Fabritiis, M. J. Harvey, S. M. Pickles and A. R. Porter. Coupled applications on distributed resources. *Computer Physics Communications*, 175: 389–396, 2006.

[77] G. D. Fabritiis, M. Serrano, R. Delgado-Buscalioni and P. V. Coveney. Fluctuating hydrodynamic modeling of fluids at the nanoscale. *Physical Review E*, 75(2): 026307, 2007.

[78] N. G. Hadjiconstantinou. Combining atomistic and continuum simulations of contact-line motion. *Physical Review E*, 59(2): 2475–2478, 1999.

[79] S. Kohlhoff, S. Schmauder and P. Gumbsch. Coupled atomistic-continuum calculations of near interface cracking in metal/ceramic composites. M. Rühle, A. G. Evans, M. F. Ashby and J. P. Hirth (editors), *Bonding, Structure and Mechanical Properties of Metal-Ceramic Interfaces, Acta/Scripta Metallurgica Proceedings*, volume 4, pages 63–70. Pergamon Press, 1990.

[80] F. F. Abraham, J. Q. Broughton, N. Bernstein and E. Kaxiras. Spanning the continuum to quantum length scales in a dynamic simulation of brittle fracture. *Europhysics Letters*, 44(6): 783–787, 1998.

[81] V. B. Shenoy, R. Miller, E. B. Tadmor, D. Rodney, R. Phillips and M. Ortiz. An adaptive finite element approach to atomic-scale mechanics — the quasicontinuum method. *Journal of the Mechanics and Physics of Solids*, 47(3): 611–642(32), 1999.

[82] R. E. Rudd and J. Q. Broughton. Concurrent coupling of length scales in solid state systems. *Physica Status Solidi (b)*, 217: 251–291, 2000.

[83] D. D. Vvedensky. Multiscale modelling of nanostructures. *Journal of Physics-Condensed Matter*, 16(50): R1537–R1576, 2004.

[84] J. Fish. Bridging the scales in nano engineering and science. *Journal of Nanoparticle Research*, 8(5): 577–594, 2006.

[85] G. A. Bird. *Molecular Gas Dynamics and the Direct Simulation of Gas Flows.* Oxford University Press, 1994.

[86] A. L. Garcia, J. B. Bell, W. Y. Crutchfield and B. J. Alder. Adaptive mesh and algorithm refinement using Direct Simulation Monte Carlo. *Journal of Computational Physics*, 154(1): 134–155, 1999.

[87] H. Wijesinghe, R. Hornung, A. Garcia and N. Hadjiconstantinou. Three-dimensional hybrid continuum-atomistic simulations for multiscale hydrodynamics. *Journal of Fluids Engineering, Transactions of the ASME*, 126(5): 768–777, 2004.

[88] V. Kolobov, R. Arslanbekov, V. Aristov, A. Frolova and S. Zabelok. Unified solver for rarefied and continuum flows with adaptive mesh and algorithm refinement. *Journal of Computational Physics*, 223: 589–608, 2007.

[89] T. E. Schwartzentruber and I. D. Boyd. A hybrid particle-continuum method applied to shock waves. *J. Comput. Phys.*, 215(2): 402–416, 2006.

[90] Q. Sun, I. D. Boyd and G. V. Candler. A hybrid continuum/particle approach for modeling subsonic, rarefied gas flow. *Journal of Computational Physics*, 194: 256–277, 2004.

[91] T. Yen, C. Soong and P. Tzeng. Hybrid molecular dynamics-continuum simulation for nano/mesoscale channel flows. *Microfluidics and Nanofluidics*, 3(6): 665–675, 2007.

[92] X. Nie, M. O. Robbins and S. Chen. Resolving singular forces in cavity flow: multiscale modeling from atomic to millimeter scales. *Physical Review Letters*, 96(13): 134501–+, 2006.

[93] J. Liu, S. Chen, X. Nie and M. O. Robbins. A continuum atomistic simulation of heat transfer in micro- and nano-flows. *Journal of Computational Physics*, 227: 279–291, 2007.

[94] H. A. Schwarz. Über einen grenzübergang durch alternierendes verfahren. *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich*, 1870.

[95] R. Henderson and G. E. Karniadakis. Hybrid spectral-element-low-order methods for incompressible flows. *Journal of Scientific Computing*, 6(2): 79–100, 1991.

[96] N. G. Hadjiconstantinou. Hybrid atomistic-continuum formulations and the moving contact-line problem. *Journal of Computational Physics*, 154(2): 245–265, 1999.

[97] E. M. Kotsalis, J. H. Walther, E. Kaxiras and P. Koumoutsakos. A control algorithm for multiscale flow simulations of water. *Physical Review E*, 79(4): 045701–+, 2009.

[98] G. Wagner, E. Flekkøy, J. Feder and T. Jøssang. Coupling molecular dynamics and continuum dynamics. *Journal of Computer Physics Communications*, 147: 670–673, 2002.

[99] G. Wagner and E. G. Flekkøy. Hybrid computations with flux exchange. *Philosophical Transactions of the Royal Society London*, 362(1821): 1655–1665, 2004.

[100] R. Delgado-Buscalioni, E. G. Flekkøy and P. V. Coveney. Oscillatory shear flow in nanochannels via hybrid particle-continuum scheme. *Condensed Matter, abstract cond-mat/0401575 - xxx.lanl.gov/abs/cond-mat/0401575*, 2004.

[101] R. Delgado-Buscalioni and P. V. Coveney. Hybrid molecular-continuum fluid dynamics. *Philosophical Transactions of the Royal Society London A*, 362(1821): 1639–1654, 2004.

[102] R. Delgado-Buscalioni, E. G. Flekkøy and P. V. Coveney. Fluctuations and continuity in particle-continuum hybrid simulations of unsteady flows based on flux-exchange. *Europhysics Letters*, 69(6): 959–965, 2005.

[103] E. Weinan, E. Bjorn and H. Zhongyi. Heterogeneous multiscale method: A general methodology for multiscale modeling. *Physical Review B*, 67: 092101, 2003.

[104] E. Weinan, R. Weiqing and E. Vanden-Eijnden. A general strategy for designing seamless multiscale methods. *Journal of Computational Physics*, 228(15): 5437–5453, 2009.

[105] H. K. Versteeg and W. Malalasekera. *An Introduction to Computational Fluid Dynamics: The Finite Volume Method.* Prentice Hall, 2nd edition, 2007.

[106] J. H. Ferziger and M. Perić. *Computational Methods for Fluid Dynamics.* Springer, 3rd edition, 2007.

[107] OpenCFD. OpenFOAM: The open source CFD toolbox – programmer's guide, 2009, version 1.6. Available online: www.opencfd.co.uk/openfoam/index.html.

[108] M. K. Borg and J. M. Reese. A hybrid particle-continuum framework. *6th International Conference on Nanochannels, Microchannels and Minichannels.* 2008. Darmstadt, Germany.

[109] G. De Fabritiis, R. Delgado-Buscalioni and P. V. Coveney. Energy controlled insertion of polar molecules in dense fluids. *Journal of Chemical Physics*, 121(24): 12139–12142, 2004.

[110] J. M. Goodfellow, M. Knaggs, M. A. Williams and J. M. Thornton. Modelling protein unfolding: a solvent insertion protocol. *Faraday Discussions*, 103: 339–347, 1996.

[111] J. Ji, T. Çagin and B. Montgomery Pettitt. Dynamic simulations of water at constant chemical potential. *Journal of Chemical Physics*, 96: 1333–1342, 1992.

Appendix A

Sampling from a Maxwell-Boltzmann distribution

In many aspects of an MD simulation it is sometimes required to sample a velocity vector from the equilibrium Maxwell-Boltzmann probability distribution at a target temperature T_o and velocity $\mathbf{u}_o$:

$$p(\mathbf{v}) = \left(\frac{m}{2\pi k_B T_o}\right)^{3/2} \exp\left(-\frac{m(\mathbf{v}-\mathbf{u}_o)^2}{2k_B T_o}\right), \tag{A.1}$$

where m is the mass of a molecule, and k_B is the Boltzmann constant.

In practice, this is implemented by sampling individual components of a velocity vector v_k, $\{k = x, y, z\}$ from a Gaussian distribution, with variance $\sigma^2 = T_o k_B/m$ and mean $(u_k)_o$:

$$p(v_k) = \left(\frac{m}{2\pi k_B T_o}\right)^{1/2} \exp\left(-\frac{m(v_k-(u_k)_o)^2}{2k_B T_o}\right). \tag{A.2}$$

If $\zeta_{GAUSS}(k)$ is a random variable sampled from a standard normal distribution (zero mean, and unit variance), which is supplied with common random number generators, then the random component velocities may be computed using the following expression:

$$v_k = \sigma\ \zeta_{GAUSS}(k) + (u_k)_o. \tag{A.3}$$

In our simulations, T_o and $\mathbf{u}_o$ are determined from the cell-target values, T_P^{req} and $\mathbf{u}_P^{\mathrm{req}}$ in which a molecule i resides.

Appendix B

Insertions and deletions of molecules in MD

B.1 Site-Searching Algorithm

We adopt a modified version of the USHER scheme [60] for the search of non-overlapping sites in a molecular dynamics region, in which new molecules can be inserted without blowing-up the simulation. USHER is preferred to other insertion schemes published in the literature, such as cavity-biased schemes [59, 110], or grid-based methods [111], mainly because it is faster in finding a suitable site for the new molecule and so has reduced processing requirements. USHER performs a steepest-descent iteration scheme, based on the criteria that the target site where a molecule is to be inserted should have (or be close to) the cell-averaged potential energy, so that negligible energy is introduced within the system. In principle, computationally-expensive long thermalization steps to remove the heat generated after each molecular insertion are not required. The scheme is computationally efficient, especially in low density fluids, since it only requires approximately $\mathcal{O}(10)$–$\mathcal{O}(100)$ intermolecular force computations (i.e. steepest-descent iterations) before a site is found [60]. It is also very suitable for systems with densities that need to be altered over particularly short time-scales.

Perhaps the greatest drawback of this technique, and, for that matter, all other insertion techniques, is that it may not work as accurately as intended if insertions are required in a fluid close to the solid state point. In this scenario, fewer sites may be available, hence the algorithm might need to produce many iterations to find them. Consequently, the scheme may consume a lot of processing time.

The USHER [60] site-searching algorithm is described here in procedural format. We include our modified element in step 1 of the procedure.

The objective of the algorithm is that, in a few steepest-descent iterations (n) within the energy landscape of an arbitrary control cell P, it finds a site, $\mathbf{r}_i$, at the desired potential energy value, U_P^{req}, at which to insert a molecule i. U_P^{req} is obtained in our

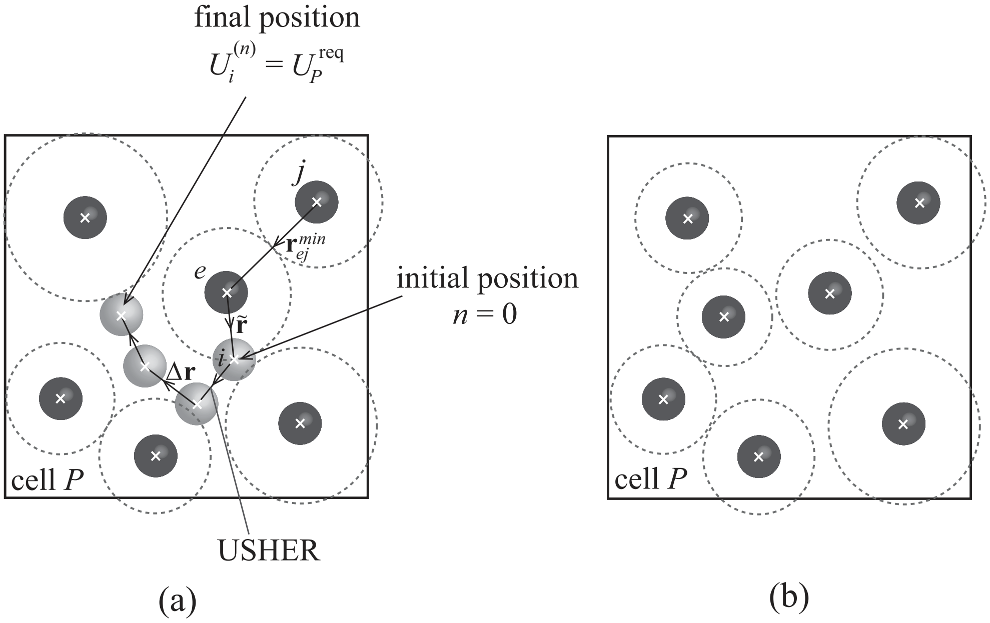


Figure B.1: Illustration showing (a) the search algorithm for a non-overlapping site within a cell P at the target potential energy, U_P^{req} and (b) inserting the molecule at the site. r_{ej}^{min} is the minimum radial separation between a molecule e and the closest neighbour molecule j.

implementation by sampling the average potential energy per molecule within P using equation (4.7). The procedure for finding $\mathbf{r}_i$ is described as follows:

1. *Find the initial starting point:* Randomly select an existing molecule, e, from the set of molecules currently occupying P and place a new fictitious molecule, i, at an initial position:

$$\mathbf{r}_i^{(n=0)} = \mathbf{r}_e + \left(\frac{r_{ej}^{min}}{2} \right) \tilde{\mathbf{d}}, \tag{B.1}$$

   where $\mathbf{r}_e$ is the position of the existing molecule, r_{ej}^{min} is the shortest separation between e and any other molecule j, $(e \neq j \neq i)$, and $\tilde{\mathbf{d}}$ is a random unit vector, with components selected independently from a Gaussian distribution. r_{ej}^{min} (see Fig B.1(a)) is computed for all molecules in the system during the latest intermolecular force calculation step. In the original USHER version, Delgado-Buscalioni and Coveney [60] choose at random the initial location of the new molecule within the domain space. However, we observed a high probability that it would overlap with other existing molecules ($r_{ie} \to 0$), and therefore blow up the simulation (since in this limit $U(r_{ie}) \to \infty$). When this occurs, a cut-off could be applied to the potential energy function, alternatively a re-insertion of the trial molecule may be performed. Our modification, however, alleviates this issue directly since the initial trial position of molecule i is always placed a distance away from other molecules and, to begin with, is also more likely to be at a lower potential energy site (see Fig B.2).

Note that if cell P is not initially occupied by molecules, the initial location $\mathbf{r}_i^{(n=0)}$ is selected to be the centre of the cell.

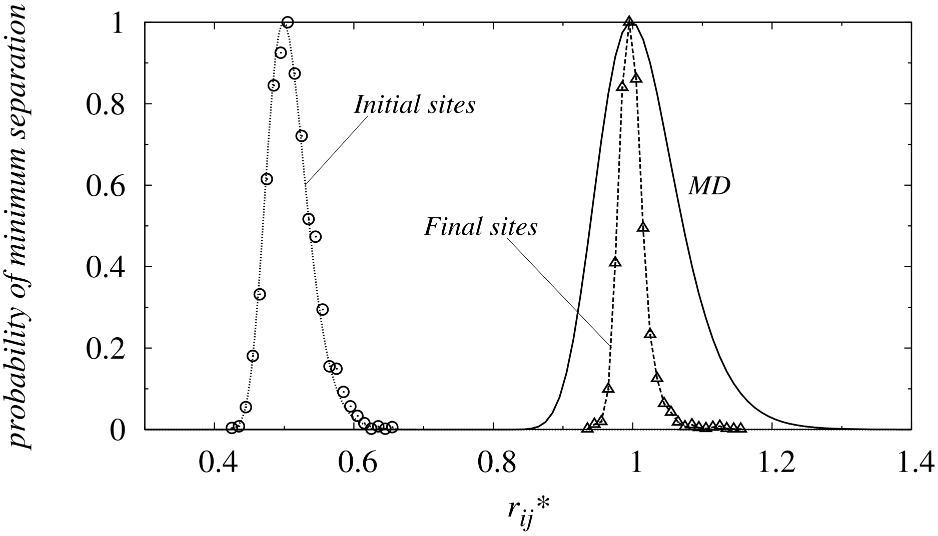


Figure B.2: We show that the initial sites of newly inserted molecules are always picked with their radial-separation from existing molecules (r_{ie}) greater than zero and close to $r_{char}/2$, where r_{char} is the smallest separation between any pair of molecules (i, j) in the entirety of the MD simulation run. In this case $\rho^* = 0.6, T^* = 2.4$, and $r^*_{char} = 0.84$. Molecules are then "ushered" to the final sites, and the figure shows that this distribution resembles the minimum radial distribution sampled from molecules within the same MD simulation.

2. *Confine the search within the cell:* If the fictitious molecule i lies outside cell P, it is deleted and the algorithm performs another "try", i.e. by starting again from Step 1. The maximum number of molecular tries, $N_{tries} \sim \mathcal{O}(10)$ is a user-defined parameter.

3. *Compute the new force and potential energy:* Calculate the net force $\mathbf{f}_i^{(n)}$ and the total potential energy $U_i^{(n)}$ on molecule i, caused by the surrounding j molecules, using:

$$\mathbf{f}_i^{(n)} = \sum_{j=1(\neq i)}^{N_{mols}} \mathbf{f}_{ij}(\mathbf{r}_{ij}), \qquad U_i^{(n)} = \frac{1}{2} \sum_{j=1(\neq i)}^{N_{mols}} U(r_{ij}), \tag{B.2}$$

where n is the iterate number. Initially $n = 0$.

4. *Test the potential energy criterion:*
IF

$$\left(U_P^{\text{req}} - \frac{U_{\text{range}}}{2}\right) \leq U_i^{(n)} \leq \left(U_P^{\text{req}} + \frac{U_{\text{range}}}{2}\right), \tag{B.3}$$

where $U_{\text{range}}(\sim 0.001)$ is the potential energy criterion range, then the insertion site $\mathbf{r}_i^{(n)}$ is found; proceed to Step 6.

ELSE Proceed to Step 5.

5. *Failed criterion — steepest-descent iteration:* Use the steepest-descent scheme proposed in the USHER protocol [60] to shift the molecule i downhill of the potential energy gradient to a new position $\mathbf{r}_i^{(n+1)}$, by a displacement $\Delta r^{(n+1)}$. This step is explained in more detail below. Proceed to Step 2, and iterate until the criteria in equation (B.3) is satisfied or until a maximum number of molecular displacements, $N_{disp}(\sim \mathcal{O}(10))$ is exceeded. The latter condition is set in place to avoid infinite looping, and also to limit the computational cost of site searching in an MD simulation. If the number of maximum displacements is exceeded, molecule i is removed and the algorithm returns to Step 1.

6. *Accepted criterion — stop search, insert molecule:* The fictitious molecule is transformed into a 'real' molecule (see Fig B.1(b)) by appending it to the list of existing molecules, initialising its velocity, and updating the forces and potential energies of all other molecules within interaction range, due to its immediate habitation.

The steepest-descent scheme [60] required in Step 5, is outlined here in two steps:

5. (a) *Compute the new displacement:*

$$\Delta r^{(n+1)} = \begin{cases} \Delta r_{ovlp}, & \text{if } U_i^{(n)} > U_{ovlp}, \\ \min\left(\Delta s, \frac{U_i^{(n)} - U_P^{\text{req}}}{|\mathbf{f}_i^{(n)}|}\right), & \text{if } U_i^{(n)} \leq U_{ovlp}. \end{cases} \tag{B.4}$$

where U_{ovlp} is an arbitrary large potential energy value representing a pair-molecular overlap ($\sim 10^4$) and $\Delta r_{ovlp} = r_{char} - (4/U_i^{(n)})^{1/12}$ is the displacement that frees the molecule from the high overlapping potential energy in one step. $r_{char} \sim 0.9\sigma$ is the characteristic distance which is typically the maximum radius at which $g(r) = 0$ in the radial distribution function. After the molecule is outside the overlapping region, the second term of equation (B.4) is used to 'usher' the molecule towards the target potential energy U_P^{req}, depending on the topology of the potential energy landscape [60]. In [60], the relationship $\Delta s^* \approx 0.1(\rho^*)^{-1.5}$ is used as the maximum displacement that ensures the optimum search for the required potential energy.

5. (b) *Displace molecule:* Move the molecule to the new position in the direction of $\mathbf{f}_i^{(n)}$ with magnitude $\Delta r^{(n+1)}$:

$$\mathbf{r}_i^{(n+1)} = \mathbf{r}_i^{(n)} + \Delta r^{(n+1)} \frac{\mathbf{f}_i^{(n)}}{|\mathbf{f}_i^{(n)}|}. \tag{B.5}$$

The Lagrangian tracking algorithm [26] is employed to track the fictitious molecule i during the 'usher' process within cell P. If i is ushered outside P, it is deleted and the search procedure starts over again. Note also that if the molecule has moved $N_{uphill} \sim \mathcal{O}(1)$ number of uphill moves in the potential energy landscape, i.e. $U_i^{(n+1)} > U_i^{(n)}$, the fictitious molecule is discarded and the procedure starts again from Step 1, since an energy trap will most likely be encountered and the algorithm may never find the target potential energy value.

B.2 Insertions and deletions in parallel

The density controller (see Section 4.4.2) and mass flux boundary (see Section 4.5.4) both require processes for insertion and deletion of molecules. Whereas these may be relatively trivial implementations for serial processing, they are daunting operations for simulations running in parallel. In fact, the authors of the USHER algorithm [60] describe only serial processing test cases for insertion of molecules. In this section we describe the issues of parallelisation, and propose a robust technique for running these processes in parallel.

We consider an arbitrary control cell P, that requires ΔN_P molecules inserted *or* deleted at a particular time in the simulation run. We use a domain-decomposition for parallel processing, so that P may be located solely on one processor. Upon insertion or deletion of a molecule, it is essential that properties (i.e. partial forces and potential energies) of other existing molecules within interaction range r_{cut} are updated prior to any other tries within P, or for that matter other tries which are carried out in any of the neighbouring control-cells of P — even those located across periodic boundaries. In many parallel-decompositions, it is reasonably common for a control cell to be located right at the edge of an inter-processor boundary. Parallel-communication is therefore necessary to update properties of existing molecules residing also on other processors. Furthermore, whenever molecules are deleted or inserted close to a periodic or processor boundary, their 'referred' images must be deleted or else new molecules referred for interaction.

Perhaps the most crucial issue arises for multiple insertions/deletions of molecules occurring in parallel (i.e. on different processors). The USHER scheme, which we use for insertion, requires that the potential energy landscape within range (r_{cut}) of cell P remains static so that the correct insertion site at the target potential energy is found (see Section B.1). This does not apply if, on a neighbouring processor (within range of P), another molecule is inserted/deleted at the same instance. Both molecules are inserted at incorrect potential energies, since the presence of the other new molecule on the alternate processor, modifies completely the local potential energy landscape. For deletion processes, the candidate molecules is chosen based on its closest potential energy to the target, and therefore poses a similar issue. If these conflicts are not accounted for properly, overheated regions, overlapping molecules and simulation blow-up may result.

B.2.1 Implementation

For the purposes of explanation, we present a simple schematic of a 2D domain mesh (with cubic cells) as illustrated in Figure B.3, and control is applied in all cells of the mesh. Our implementation is, however, generally applicable to arbitrary 3D unstructured meshes, and control zones applied as subsets of the computational domain.

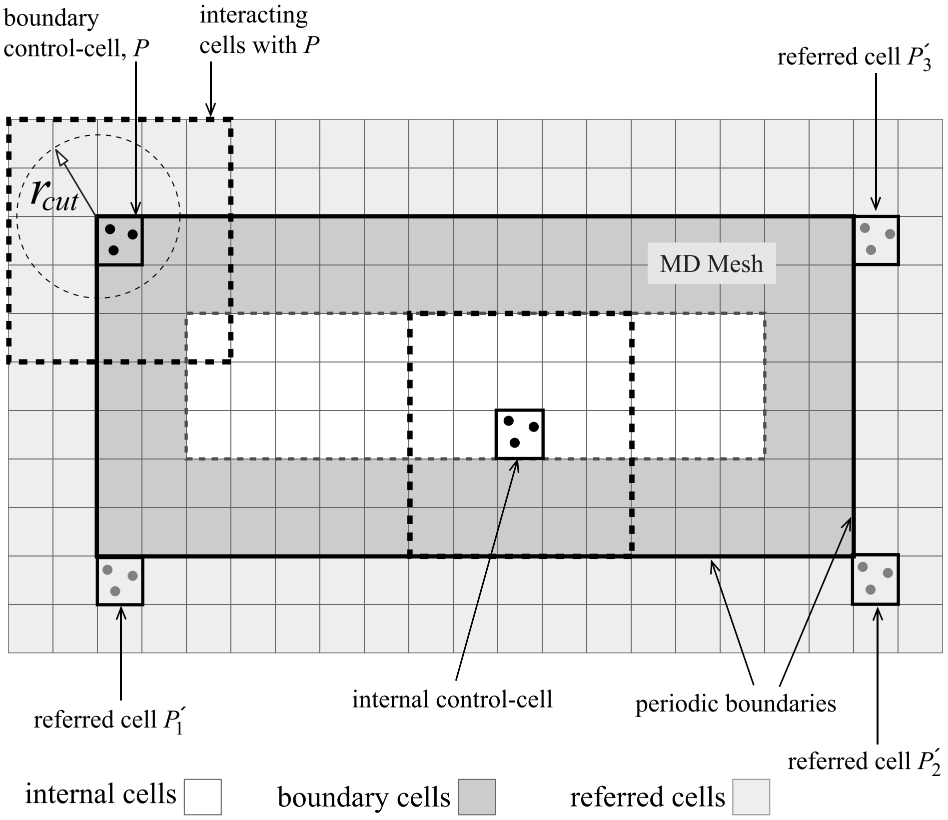


Figure B.3: Schematic of a 2D MD mesh with periodic boundary conditions applied in all directions. 'Real' cells on the mesh are either internal cells (located r_{cut} away from any periodic boundaries) or boundary cells. The 'referred' cells are copies of real boundary cells. E.g., molecules located in a boundary control-cell P are imaged to referred cells P_k', $k = \{1, 2, 3\}$.

Prerequisites

In the sections that follow, we adopt the same methodology as the Arbitrary Interaction Cells Algorithm (AICA) scheme [20, 24]. AICA is a generalisation of the linked-cells algorithm for computing intermolecular forces in complex meshes. For each '*real*' cell on the mesh, AICA creates a sphere of '*interacting*' cells within a radius r_{cut}. Any molecule that occupies an arbitrary real cell computes the pair intermolecular force with molecules residing within its own cell in addition to molecules within the interacting cells. Cells on the mesh may be either 'internal' or 'boundary' cells. Internal cells are located at a distance greater than r_{cut} from any periodic or processor boundary. In this case, the interacting cells will comprise only real cells (see Fig B.3). A boundary cell is located within r_{cut} of processor and periodic boundaries, and the complete sphere of interacting cells consists of a partial list of real cells in addition to a partial list of '*referred*' cells. If the system is said to be fully periodic, as we initially assume for the case in Figure B.3, a set of referred cells are applied in the form of a "skin" around the entire domain boundary, of thickness r_{cut}. These referred cells will store images of molecules the positions of which are transformed from real cells across the coupled boundary. If the case shown in Figure B.3 is decomposed for parallel processing (see Fig B.6), an arbitrary processor-mesh will still have a skin of referred cells all around the partial domain. However, those referred cells which are due to parallelisation require imaging of molecules from the corresponding processor meshes, and hence parallel communication is necessary.

Insertion/deletion procedure

The insertion and deletion processes of one molecule (see Section 4.4.2) may be each divided into three steps specifically for alleviating the issues of parallelisation;

- for insertion,
  (1) search for a candidate site,
  (2) insert a molecule at this chosen site and
  (3) update properties of surrounding molecules;
- for deletion,
  (1) search and highlight a candidate existing molecule,
  (2) update properties of surrounding molecules from the highlighted molecule and
  (3) delete the molecule.

Serial-processing

Control cells of the zone are looped over in succession, and the molecules are inserted/deleted in turn within an arbitrary control cell, P. Each of the aforementioned functions '(1)','(2)' and '(3)' are therefore performed sequentially; no two functions are run simultaneously. Figure B.4 shows these steps for an arbitrary *internal* cell P. The update scheme (step '(3)' for insertion or step '(2)' for deletion) consists of updating partial forces (or accelerations) and potential energies using equations (4.8) or (4.9) for molecules within r_{cut} range. For this implementation, the cell uses its interaction list of real cells described in the AICA scheme. Other updates include also R_e (required for further insertions) and the virial tensor.

In the case of insertion of a molecule i in an arbitrary *boundary* control-cell, P, which for serial-processing may only be of 'periodic' type, the update step is segmented as follows (see Fig B.5):

1. Update the properties of existing molecules in the partial list of (real) interacting cells.
2. Refer the image of the newly inserted molecule i to cell P's set of corresponding referred cells P'_k, $k = \{1, 2, \dots\}$.

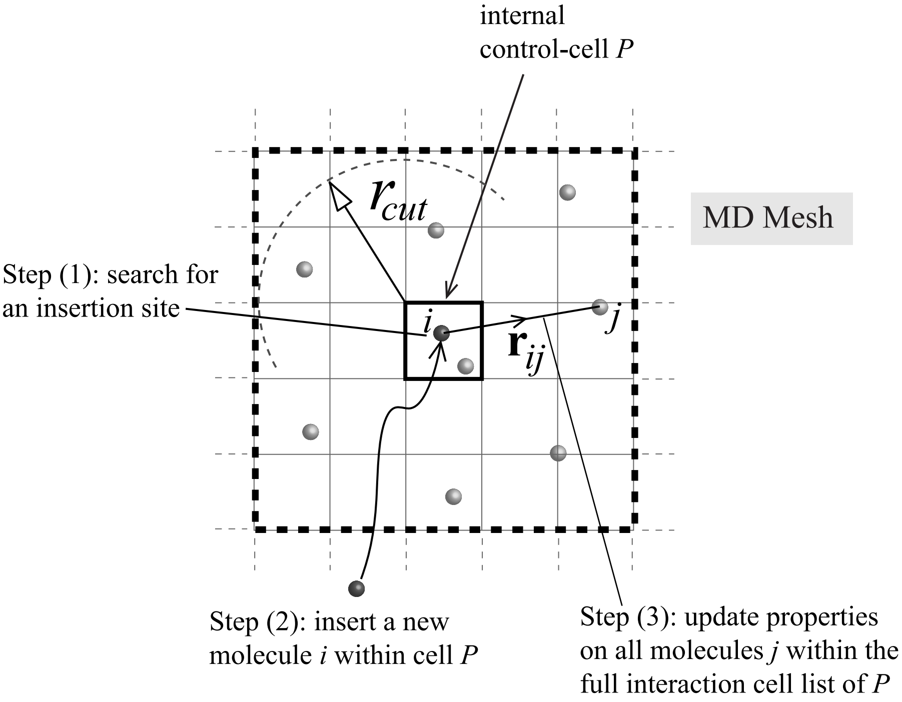


Figure B.4: Schematic showing the process of inserting a molecule i within an *internal* control-cell P. The thick dotted line contains the interaction cells for cell P, located within a radius of r_{cut} from the boundaries of P. The process is similar for deletion purposes.

3. For each referred cell, P'_k, update the real molecules within its interacting cell list.

Note that in the case of deletion, first the candidate referred molecule is highlighted, then the surrounding real molecules are updated, and finally the referred molecule is deleted.

Parallel-processing

During parallel-processing, the control zone is divided across processors, hence reducing the computational effort required per processor (see Fig B.6). Each processor starts off first with insertions/deletions within its *internal* control-cells. Since in these cases no processor communication is necessary, speed-up is achieved. Then the processors simultaneously proceed with insertions/deletions within the *boundary* control-cells, during which communications are introduced. In these cases, one molecule per processor is inserted/deleted in tandem. Each processor therefore starts off by selecting one arbitrary boundary control cell P, and performs step '(1)' in serial. Subsequently, processors communicate to check for *conflicts.* A conflict occurs if any two or more designated molecules are within r_{cut} range. Inter-processor conflicts are resolved by arbitrarily assigning a priority to each processor. The processor with the highest priority proceeds with the insertion/deletion of the molecule, while processors of lower priority — that detect a conflict — terminate their tries and reattempt them in the next try. Those processors that do not detect a conflict, proceed to the update step: '(2)' for deletion and '(3)' for insertion. This is performed by

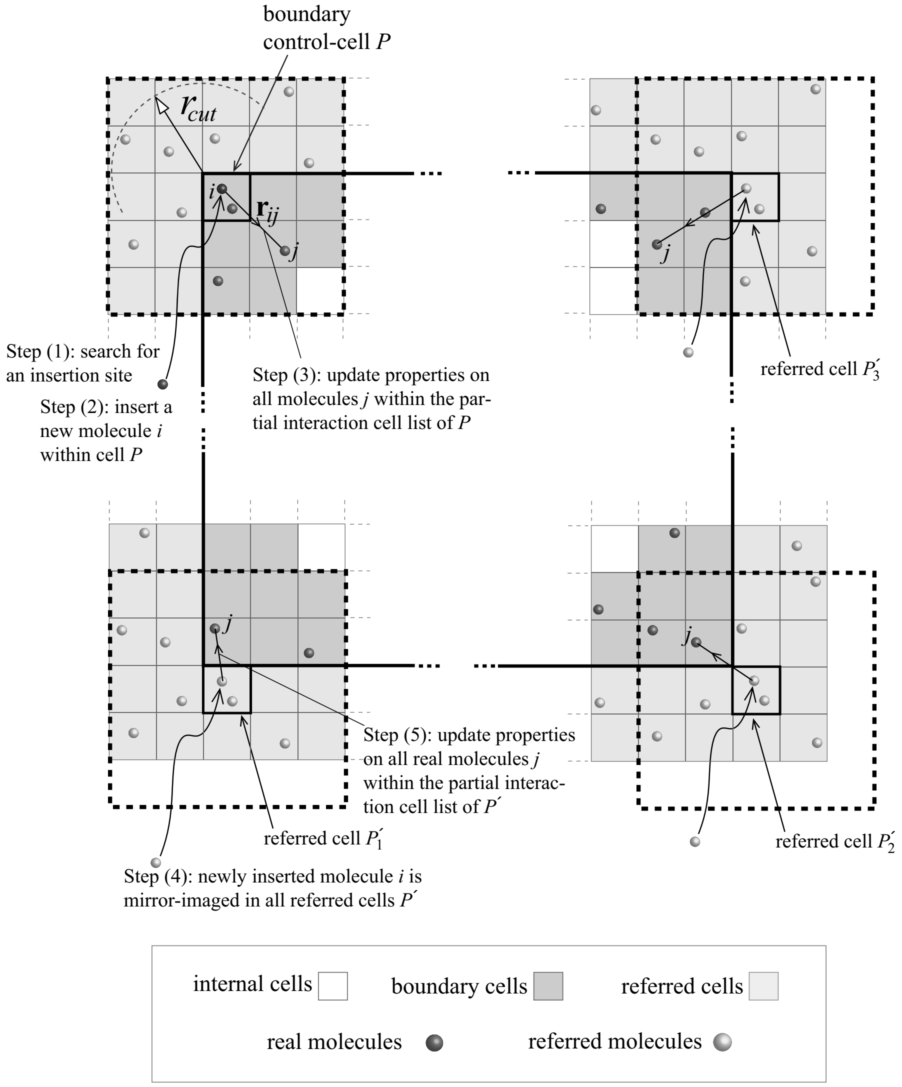


Figure B.5: Schematic showing the process of inserting a molecule i within a *boundary* control-cell P in serial. The case consists only of cells referred from periodic boundaries.

employing the referred cells in a similar manner to the serial-processing case. The only difference is that the images of referred molecules need to be disseminated across processors prior to continuation of the update. The MD simulation proceeds until all the molecules have been controlled within each cell of the control zone, and across all processors.

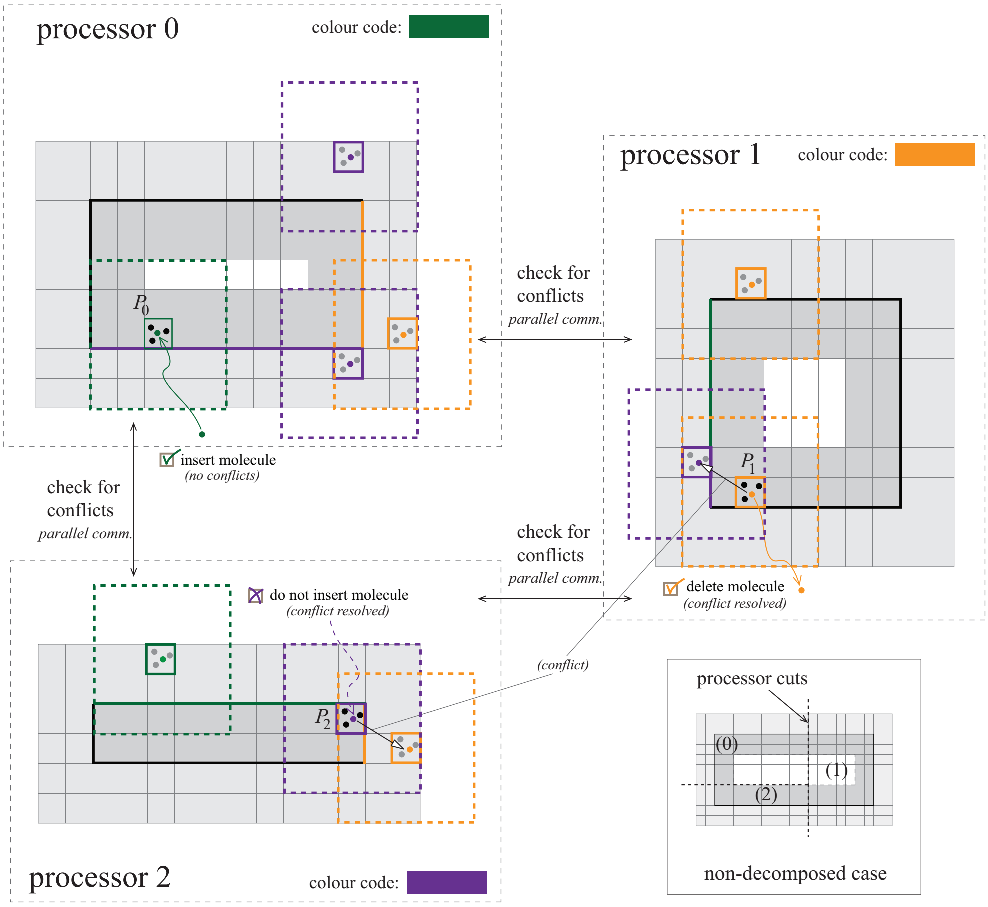


Figure B.6: Processor decomposition of the simple mesh shown in Figure B.3. In this figure we illustrate how conflicts are resolved across processors. One molecule per processor is attempted first, followed by parallel communication to test for conflicts.

Appendix C

MD Pre-processing Utilities

We have devised a set of numerical pre-processing utilities in order to introduce desired features in MD and hybrid simulations:

- *Lagrangian*-type utilities are directly related to processes that manipulate the positions of molecules within the mesh;
- *Zone*-type utilities generate localised zones on the MD mesh.

A cell-zone is a group of cells, while face-zones are a group of faces on the mesh. Zones are mostly required by the algorithms described in this thesis, so that perturbations are applied locally in cells (e.g., state controllers, see Section 4.4), as molecules cross faces (e.g., boundary models, see Section 4.5) or for localised measurement of properties (e.g., measurement techniques, see Section 3.5). Furthermore, zones are also required in hybrid simulations to define the coupling regions, and for sub-meshing. The zone utilities which are described below therefore provide a vast number of options for building arbitrary zones in a flexible and fast manner on the mesh.

C.1 Lagrangian

1. *Initial molecular lattice-configurations* — This pre-processing utility [20, 27] is considered key to MD simulations as it creates molecules for solid, liquid or gas matter in a lattice configuration. The mesh is uniquely divided into zones by the user and each zone is filled with a particular lattice structure, using input properties which include number density, temperature, velocity-distribution, bulk velocity, identification number for assigning its set of potentials, mass, and lattice structure (e.g. FCC, SC, BCC). The filling algorithm consists of an expanding unit cube of molecules that terminates once the zone in question is completely filled.
2. *Mapping of Lagrangian fields* — A utility that maps a ‘cloud’ of existing molecules from an old mesh to a new one has been constructed. Molecules may be translated by a spatial vector $\mathbf{r}$ within the new mesh. This is a very useful tool for creating an initial

molecular lattice configuration for a particular object of complex three-dimensional topology. A fine mesh defining the object is first constructed, filled appropriately with molecules, then translated to a coarser mesh for subsequent simulations. It may also be utilised to create copies of a cloud of molecules within a larger system or for distributing identical macromolecules within the same system.

3. *Sub-set mapping of molecules* — Only those molecules of a chosen identification number that are located within a user-defined cell-zone are mapped to the new mesh. All other molecules are deleted. This utility provides a process that 'cuts-out' molecules from a zone within an equilibrated fluid, rather than 'fills-up' the zone with a lattice fluid that still needs to relax, prior to the MD run. This utility has also been used here for visualising one specie (e.g. droplets) within a multi-specie MD system.

4. *Initial molecular configuration for solute-solvent systems* — After constructing a series of macromolecules (e.g., droplets, nanotubes, complex-shaped walls) this utility fills in the remaining empty volume with a lattice configuration that defines the solvent. In this case we use the same algorithm described in the first utility above. Overlapping molecules are avoided by providing a minimum distance r_{min} between the original molecules and the solvent that is created.

C.2 Zones

1. *Create cuboid zones* — This relatively simple and useful utility creates any arbitrary number of cuboid-shaped zones within the computational domain. Each cuboid zone may be of any size: the origin and end point of the cuboid are specified as an input, that correspond to vertices on the mesh. Cells on the mesh are accepted as part of the cuboid-zone if its cell-centre is within bounds of the start and end points.

2. *Combine regions* — A utility that combines together any arbitrary number of zones.

3. *Create layers* — Starting from an arbitrary boundary (or multiple boundaries), or a starting surface, this utility creates a series of user-defined "layers" — each layer may have a different thickness. This is mainly used to create zones either for setting up the spatial framework for non-periodic boundary conditions in MD simulations or for creating the coupling regions in hybrid methods. The algorithm [108] is designed to speed up the process of creating the sub-regions of a series of coupling regions in arbitrary mesh geometries.

4. *Create spherical zone* — A utility that starts from a cell and builds a spherical zone of a user-defined radius.

5. *Create surfaces* — A utility that builds face zones, by specifying two adjacent zones. The algorithm then selects common faces on the zone-zone interface.