School of Aerospace, Mechanical & Mechatronic Engineering

Faculty of Engineering and Information Technologies

The University of Sydney, Australia, 2016

Lattice Boltzmann methods for direct numerical simulation of turbulent fluid flows

Author:

Vanja Zecevic

Supervisor:

Assoc. Prof. Michael Kirkpatrick

Associate Supervisor:

Prof. Steven Armfield

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

Abstract

We study the use of lattice Boltzmann (LB) methods for simulation of turbulent fluid flows motivated by their high computational throughput and amenability to highly parallel platforms such as graphics processing units (GPUs). Several algorithmic improvements are unearthed including work on non-unit Courant numbers, the force operator, use of alternative topologies based on face and body centered cubic lattices and a new formulation using a generalized eigendecomposition that allows a new freedom in tuning the eigenvectors of the linearised collision operator. Applications include a variable bulk viscosity and the use of a stretched grid, our implementation of which reduces errors present in previous efforts. We present details for numerous lattices including all required matrices, their moments the procedures and programs used to generate these and perform linear stability analysis.

Small Mach number flows where density variations are negligible except in the buoyancy force term allow the use of a highly accurate finite volume solver to simulate the evolution of the buoyancy field which is coupled to the LB simulation as an external force. We use a multidimensional flux limited third order flux integral based advection scheme. The simplified algorithm we have devised is easier to implement, has higher performance and does not sacrifice any accuracy compared to the leading alternative. Our algorithm is particularly suited to an outflow based implementation which furthers the stated benefits. We present numerical experiments confirming the third order accuracy of our scheme when applied to multidimensional advection.

The coupled solver is implemented in a new code that runs in parallel across multiple machines using GPUs. Our code achieves high computational throughput and accuracy and is used to simulate a range of turbulent flows. Details regarding turbulent channel flow and sheared convective boundary layer simulations are presented including some new insight into the scaling properties of the latter flow.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Associate Professor Michael Kirkpatrick. He has been a constant figure of guidance, support, inspiration and wisdom. I would also like to mention his patience in reading my writing and providing feedback. I would also like to thank Professor Steven Armfield who has also been a continued source of guidance, support, inspiration and wisdom. I would also like to express my gratitude for the financial support of the Australian Postgraduate Award and for the additional finiancial support provided by my supervisor as well as the school of Aerospace, Mechanical and Mechatronic Engineering for their support of this thesis and for providing part time employment during this time. Finally I would like to thank my partner Jessica Cheok for her support and understanding during this time.

Contents

List of Tables

List of Figures

List of Symbols

Symbol	description	unit
ρ	density	kg m^{-3}
$c_{\boldsymbol{i}\alpha}$	particle velocity	m s^{-1}
$c'_{\boldsymbol{i}\alpha}$	particle thermal velocity	m s^{-1}
u_α	stream velocity	m s^{-1}
$\Pi_{\alpha\beta}$	momentum flux tensor	
$\sigma_{\alpha\beta}$	stress tensor	
$\sigma'_{\alpha\beta}$	viscous stress tensor	
$\sigma^s_{\alpha\beta}$	viscous shear stress tensor	
$\sigma^v_{\alpha\beta}$	viscous volumetric stress tensor	
$\epsilon_{\alpha\beta}$	rate of strain tensor	
P	thermodynamic pressure	kPa
P'	pressure tensor	kPa
q_α	heat flux tensor	
M	molar mass	g mol^{-1}
Pr	Prandtl number	
Re	Reynolds number	
Cr	Courant number	
Kn	Knudsen number	

Symbol	description	unit
$\hat{n}$	molar density	$\mathrm{mol\,m^{-3}}$
n	number density	$\mathrm{m^{-3}}$
$\mathcal{C}$	collision operator	

List of Constants

Constant	description	unit
R_u	universal gas constant	$8.3144621\,\mathrm{kJ\,kmol^{-1}\,K^{-1}}$
P_ATM	standard pressure	$101.325\,\mathrm{kPa}$
T_0	standard temperature	$0\,°\mathrm{C} = 273.15\,\mathrm{K}$
k_B	Boltzmann's constant	$1.3806488 \times 10^{-23}\,\mathrm{J\,K^{-1}}$
N_A	Avogadro's constant	$6.02214129 \times 10^{23}\,\mathrm{mol^{-1}}$
n_0	standard number density	$2.686781 \times 10^{25}\,\mathrm{m^{-3}}$

Chapter 1

Introduction

This thesis concentrates on three distinct but closely interwoven aspects of the lattice Boltzmann (LB) method, theoretical, computational and practical. We have established a thorough framework analysing these methods while also developing a more general treatment of the collision term allowing individual selection of eigenvectors and adjustment of eigenvalues for selected lattices. Our methodology exposes new techniques to deal with non-isotropic lattices, to increase accuracy and to independently vary parameters such as bulk viscosity and thermal diffusivity. We also present an accurate treatment of the external force term within this framework. The process of linear stability analysis is detailed and is used to establish the stability characteristics of various schemes. Chapter 3 is dedicated to this analysis as well as the literature review of LB methods.

The LB method has been increasing in popularity as a means of simulating fluid flow. The explicit and local nature of calculations has been attractive to numericists, allowing highly efficient parallel implementations. The method has been successfully adapted to massively parallel computing architectures such as graphics processing units [262, 258, 244], also achieving high performance on traditional parallel machines [189, 173, 164]. The kinetic nature of the LB method opens the door to implementation of additional physics at the particle level and the method has enjoyed success in simulating flows involving multiple phases or components, reacting species and complex geometries such as porous media as well as particle laden flows, thermal flows and microfluidics. Thorough reviews of the method and its application to the range of problems mentioned, as well as incompressible and compressible fluid dynamics are presented in Annual Reviews [38, 2] and books [255, 234]. The LB method is based on the Boltzmann equation which describes the underlying particle kinetics responsible for dilute gas flow at a more fundamental level than the Navier-Stokes equations, giving researchers optimism that the method may be applied to a larger range of parameters including high Knudsen number, highly non-equilibrium flows such as rarefied gases, microfluidics and shocks. Some of these applications are discussed in Chapter 3 however we will note at this stage that the single relaxation time method as it is commonly encountered, is only able to simulate the athermal Navier-Stokes flows at a small Mach number. Chapter 2 gives the theoretical background for the lattice Boltzmann equation.

The properties of LB methods are closely tied to the collision operator and to the ability of the underlying lattice to correctly represent higher order moments of the particle population $f_{\boldsymbol{i}}$. Chapter 3 covers in detail the implementation of lattice Boltzmann methods. The quantity $f_{\boldsymbol{i}}$ is a discrete probability distribution function (PDF) representing the probability of finding a particle with a particular momentum at a particular place and time. Moments of a PDF are integrals or sums over the distribution weighted by powers of the independent variable, in this case velocity. The zeroth order moment is the density,

$$m \sum_{\boldsymbol{i}=0}^{q} f_{\boldsymbol{i}} = \rho.$$

The first order moments are the fluid momenta,

$$m \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} f_{\boldsymbol{i}} = \rho u_\alpha.$$

Since the discrete PDF $f_{\boldsymbol{i}}(x_\alpha, t)$ is an approximation of the continuous PDF $f(c_\alpha, x_\alpha, t)$, a considerable source of difficulty comes from selecting an appropriate set of discrete velocities $c_{i\alpha}$ that can reproduce all of the required higher order moments. A discrete set of q velocities can only support at most q independent moments, the map between particle populations and moments can be specified by a matrix multiplication,

$$m_i = \sum_{j=0}^{q} M_{ij} f_j.$$

Many commonly encountered LB methods use a lattice that only correctly recovers the equilibrium moments up to the second order. Third order moments are partially recovered but a cubic velocity deviation, which is assumed to be negligible when the Mach number is sufficiently small remains. The fourth order moments which appear in the energy equation are not recovered correctly and thus the methods are only able to recover the athermal Navier-Stokes equations. The resulting scheme is a compressible solver that is only accurate in the incompressible limit. The constant temperature means that acoustic perturbations are not physically accurate. These methods are nevertheless an attractive means of simulating incompressible flow due to their simplicity, lack of computational effort and parallel efficiency.

Incompressible fluid dynamics is the small Mach number limit of compressible fluid dynamics. As the speed of sound increases, the density fluctuations required to produce a given pressure get smaller. In the limiting case, as the speed of sound approaches infinity, the density fluctuations become infinitesimally small and the density can be considered constant everywhere except in the pressure term where it is multiplied by a term proportional to the speed of sound squared. When the speed of sound is much larger than the fluid velocity, these propagations take a negligible amount of time compared to the characteristic time scales of the flow. It is as though the pressure adjusts itself instantly to maintain a divergence free velocity field. Additional work is required in order to simulate the propagation of these acoustic perturbations over the whole domain when using a compressible scheme to approximate an incompressible fluid. Schemes must have a sufficiently small time step to capture all features of the flow. The Courant number,

$$\mathrm{Cr} = \frac{u\Delta t}{\Delta x}$$

must typically be kept below some limit. As the characteristic velocity increases, the required time step becomes smaller. For a compressible solver at a small Mach number, the relevant velocity is the speed of sound (c_s) which is related to the characteristic stream velocity within the flow (u_0) by the Mach number,

$$\mathrm{Ma} = \frac{u_0}{c_s}.$$

The time step is inversely proportional to the Mach number, a ten fold reduction in Mach number requires ten times as many time steps. In practice, an infinitely small Mach number cannot be achieved using this method so a sufficiently small value must be chosen so that no compressible effects are introduced. Simulations using the LB method typically use Mach numbers of order 10^{-2} to 10^{-1}. Another issue with using a very small Mach number is that the acoustic velocity scales become much smaller than the mean flow velocity and the density perturbations likewise become much smaller than the mean density. Care must to be taken to ensure that these scales can be sufficiently captured by the floating point representation used in the code.

A typical Courant number required when using LB methods can be illustrated by making the common assumption that Δx, Δt and the speed of particles $c_{\boldsymbol{i}\alpha}$ are one. Any other values that allow particles to stream directly from site to site in one time step give the same result. The speed of sound for most lattices on a cubic grid contained in this thesis is taken to be,

$$c_s = \frac{1}{\sqrt{3}} \sim 0.5774$$

If we use a common value of Ma $= 0.1$ this means that the maximum velocity is around,

$$u_{\max} \sim 0.05774.$$

Since Δx and Δt are one, this also gives a Courant number of 0.05774 which compares reasonably with some explicit methods but implicit methods used in incompressible simulations can do much better. If the Mach number needs to be further reduced, the comparison becomes even less favourable for LB and other artificially compressible methods. We further discuss performance results in Section 5.1.

Incompressible schemes are not completely immune to this additional effort, they typically need to solve a Poisson equation at each time step in order to generate a pressure field.

$$\partial_\alpha \partial_\alpha P = -\partial_\alpha \rho u_\beta \partial_\beta u_\alpha$$

There are many different algorithms, some of which are described in Section 4.3 however they all require the solution of a similar equation at least once per time step if not more. Comparing these schemes to artificially compressible LB methods, they represent a trade-off allowing a larger time step but more numerical operations per time step. We have found that this works out favourably for the artificially compressible LB method only when parallel efficiency is taken into account, the LB methods tend to scale better to parallel architectures due to the local nature of calculations. Another caveat is that performance results are highly dependent on which schemes are selected for comparison, performance tuning and code optimization.

The order of accuracy of LB methods depends on the accuracy of the underlying model, as well as the accuracy of the velocity discretization and of the numerical scheme used to solve the advection of particles. Many commonly used LB methods including all in this thesis are second order accurate in space and Mach number. Higher order methods are an area of ongoing research. Other issues include oscillations and stability and the reliance on a uniform grid. There has been considerable research into LB schemes on non-uniform grids including our own work on the use of a rectangular lattice however there appears to be no simple solution.

In order to simulate buoyancy driven flows using the Boussinesq approximation, we use a highly accurate finite volume scheme to solve the advection-diffusion equation for temperature as a scalar. The numerical scheme we use is significantly more adept at simulating scalar advection than the LB equation for a scalar. This scheme is a third order accurate multi-dimensional flux integral based scheme which I have derived named NEWTOPIA. It is based extensively on Leonard's [144, 143] UTOPIA scheme. The scheme is presented as as an outflow implementation which is easier to program and offers higher performance than traditional implementations. This advection scheme, coupled with Thuburn's [242] multi-dimensional flux limiter, results in a highly accurate and stable algorithm. Details are covered further in Chapter 4.

The superiority of finite volume method for the advection of a scalar motivates the search for a similar scheme for the Navier-Stokes equations. Unfortunately, the compressible NS equations require the solution of a Riemann problem at control volume boundaries significantly complicating the construction of difference schemes compared to the advection-diffusion equations. The formulation of a non-oscillatory difference scheme in this context is much more complicated. In order to make a comparison, we have developed an athermal compressible code based on the numerical scheme of Kurganov and Tadmor [129]. Our code does not solve the energy equation because it is meant to somewhat match the simulation capabilities of the low Mach number athermal LB schemes considered in this thesis. This finite volume scheme is significantly more complicated and roughly seven times slower than the LB method although it is more stable and able to be used at any Mach number. The finite volume scheme is also able to be extended to solve the full Navier-Stokes equations without too much additional effort.

I have developed a lattice Boltzmann code as well as a number of utility programs designed to aid in the theoretical study of LB methods as well as in their implementation. The simulation code runs on conventional multi-core processors using a shared memory parallel processing model, as well as on massively parallel graphics processing units (GPU's) which use a single instruction multiple device (SIMD) processing model. The code also uses domain decomposition in order to run on a cluster of such machines. It is written in the C programming language, switching to CUDA for GPU programming and using MPI for domain decomposition. The utility programs `lb_moment` and `lb_pert` are covered in Chapter 3 which is dedicated to LB method theory. The `lb_moment` program aids in the analysis of lattices and the creation of the matrices required to implement our method. The `lb_pert` program performs linear stability analysis on a lattice over selected parameters such as mean background velocity and relaxation rates.

Finally, the code is put to practical use in Chapter 5. Test cases including laminar and turbulent channel flow, Taylor-Green vortex flow and the sheared convective boundary layer (SCBL) which is a commonly occuring flow in the atmospheric boundary layer are simulated. We have uncovered some new insight into the scaling behaviour of the SCBL. The high computational throughput of our schemes is also discussed in this section.

Chapter 2

The equations of fluid motion

The motion of fluids, including gases and liquids, at the macroscopic and larger scales results from the microscopic motion of a large system of molecules. In this chapter, we will cover the equations governing fluid motion and various methods of simulating fluid flow ranging from the direct simulation of the motion of individual particles to the continuous Navier-Stokes equations. We do this in order to frame where our lattice Boltzmann (LB) methods fit into this landscape, in order to make clear the assumptions made during theoretical work and in order to allow comparison to competing methods. We start with a probabilistic particle description and show how the conservation of mass, momentum and energy on a particle level give rise to a set of conservation equations for the hydrodynamic variables, ρ, u_α and e. The equations derived are not fully specified in terms of the hydrodynamic variables, the stress tensor and heat flux tensor are defined in terms of moments of the particle distribution and need a series of assumptions about the behaviour of the fluid to arrive at a closed form. If the system is close to equilibrium, the thermal energy is equally partitioned between all available degrees of freedom ζ and described by one temperature. This allows an equation of state of the form $P(\rho, T)$ to link the energy and momentum equations. A linear relationship between stress and strain and a linear relationship between temperature gradient and heat flux then completely specify the unknown tensors in the hydrodynamic conservation equations leading to the Navier-Stokes equations.

Particle methods such as molecular dynamics, Boltzmann solvers and direct simulation Monte Carlo methods aim to solve the underlying kinetics directly without invoking the hydrodynamic variables. Molecular dynamics is the most straightforward, the exact motion of each particle in the system is simulated directly. This approach can capture all characteristics of the fluid flow however since the number density of an ideal gas at standard conditions is $2.7 \times 10^{25}\ \mathrm{m}^{-3}$, the computational difficulties are considerable. The direct simulation Monte-Carlo method (DSMC) attempts to improve computational efficiency by replacing a large number of particles with a single simulation particle. Deterministic collisions between real particles are replaced with probabilistic collisions between any particles in the same region of space, usually defined by a grid of cells. The Boltzmann equation describes the particles in terms of a probability distribution function which is subject to a differential transport equation. The effect of collisions among particles is captured via a collision integral. The simulated particles of the DSMC method can be thought of as basis functions representing the probability distribution function of the Boltzmann equation. The probabilistic collisions of the two methods are also similar and both are also restricted to dilute gases.

Terminology and notation is established in this chapter and will be referred to without explanation subsequently. With this background in place, we will then cover the lattice Boltzmann method specifically in Chapter 3 including our unique solutions to several problems that are encountered. We will also pay some additional attention to finite volume methods used to solve the continuous equations in Chapter 4.

We discuss the regions of applicability of the Boltzmann and Navier-Stokes formulations as well as approximations that allow their application to other situations. An example of this is our use of the lattice Boltzmann methods to simulate incompressible fluids that are far from being dilute gases.

2.1 Tensor notation

We will use a simplified variant of Einstein summation notation to denote tensors. Greek subscripts (α, β, γ, etc.) will be used to denote spatial dimensions and summation over repeated indices is implied. These subscripts are different to the Roman bold subscripts that will represent particle population indices, those do not have implied summation. Tensor notation allows for the compact representation of differential equations and also simplifies the description the moments of the equilibrium distribution. On the other hand, tensor notation is not well suited to neatly expressing the discrete sums of various factors multiplied by the discrete particle distribution function. The correct value for the third order moments of the equilibrium distribution function used by single relaxation time, athermal discrete Boltzmann methods is shown below to illustrate both of these features,

$$m \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f_{\boldsymbol{i}}^{eq} = RT \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma} \right) \rho u_\delta + \rho u_\alpha u_\beta u_\gamma .$$

The left hand side is a sum over all q particle populations, each term in the sum is a product of the equilibrium probability distribution function $f_{\boldsymbol{i}}^{eq}$ for the $\boldsymbol{i}^{\mathrm{th}}$ particle and three of that particle's velocity components, for example $m c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} c_{\boldsymbol{i}y} f_{\boldsymbol{i}}^{eq}$. If we consider all possible combinations of different components of velocity, the result is a rank three tensor with DIM^3 distinct combinations. The correct value for this tensor required for the LB method to simulate the Navier-Stokes equations is given on the right hand side demonstrating the implied summation over the repeated index δ resulting in a rank three tensor in terms of α, β and γ. The Kronecker delta has a value of one only if the indices match, it is equivalent to the identity matrix,

$$\delta_{\alpha\beta} = \begin{cases} 1 & \text{if } \alpha = \beta \\ 0 & \text{otherwise.} \end{cases}$$

Multiplication of the Kronecker delta by another tensor amounts to simply replacing the repeated index with the other one in the delta,

$$\delta_{\alpha\beta}u_{\alpha} = u_{\beta},$$
$$\delta_{\alpha\beta}A_{\beta\gamma} = A_{\alpha\gamma}.$$

The trace of a matrix can be written,

$$\delta_{\alpha\beta}A_{\alpha\beta}$$

Vector cross products are written in terms of the Levi-Civita symbol [7]. The rank and dimension of these symbols are equal and the value is one when the indices are even permutations of $1, 2, 3, \ldots n$, negative one for odd permutations and zero otherwise. The cross product makes use of the Levi-Civita symbol of rank and dimension three. In this case the even and odd permutations also happen to be cyclic and anti-cyclic permutations,

$$\varepsilon_{\alpha\beta\gamma} = \begin{cases} +1 & \text{for } \{\alpha, \beta, \gamma\} = \{1, 2, 3\} \text{ or } \{2, 3, 1\} \text{ or } \{3, 1, 2\} \\ -1 & \text{for } \{\alpha, \beta, \gamma\} = \{3, 2, 1\} \text{ or } \{2, 1, 3\} \text{ or } \{1, 3, 2\} \\ 0 & \text{otherwise} \end{cases}$$

Then,

$$\mathbf{v} \times \mathbf{w} = \varepsilon_{\alpha\beta\gamma}v_{\beta}w_{\gamma}.$$

The cross product of two dimensional vectors is found by treating them as three dimensional vectors with one component equal to zero, the resulting vector is normal to the two dimensional plane shared by the original vectors.

2.2 Molecular dynamics

The most physically accurate results can be obtained by directly simulating the motion of each particle in the system and its interaction with other particles. These are referred to as molecular dynamics (MD) simulations and they are in principle able to capture all possible characteristics of the problem with the drawback being that the calculations are extremely onerous. In this section we briefly discuss MD techniques as an aside in order to later contrast them to lattice Boltzmann methods which are obtained by a series of simplifications.

The particles form a discrete system where each can be described by a position and velocity, six degrees of freedom in three dimensions. The particles may also possess internal degrees of freedom such as angular momentum or vibrational energy. Assuming the intermolecular forces are short range compared to the spacing between molecules, the motion of particles can be separated into a streaming phase, where the particle travels through space, possibly under the influence of some external force such as electromagnetic fields or gravity until it collides with some other particle. This is in contrast to an n-body problem where each particle in the system constantly interacts with every other one. During the subsequent collision phase, the particles briefly interact, exchanging momentum and perhaps energy or even undergoing a chemical reaction before continuing along their path beginning another streaming phase. In general there may be more than one particle involved during a collision especially for more dense matter. When considering gas flows, the collisions are generally short range, elastic interactions unless there are electromagnetic effects which can cause long range interactions between many particles. Even electromagnetic effects may be approximated by considering the combined effect of all particles as a force field that can be discretized over volumes and acts during the streaming step.

The number density of an ideal gas at standard conditions is roughly $2.7 \times 10^{25}\,\mathrm{m}^{-3}$. To get an idea about just how many resources are required to perform these simulations, assume double precision floating point values taking up 8 B are used to store each of the six degrees of freedom, then in order to model a problem at the engineering scale, say a cubic meter of ideal gas would take 1.3×10^{18} GB of memory. According to the TOP 500 supercomputer list, as of 2014 the world's fastest supercomputer is the Tianhe-2 [1] which has 1.024×10^{6} GB of memory, a shortfall of the order of 10^{12}. Even if Moore's law continues it will take hundreds of years for the worlds largest supercomputers to reach this size and with transistor sizes fast approaching the limiting size of single silicon atoms it is not clear if the miniaturization trend can even continue. Without getting into questions about the theoretical limits of computation [146, 136, 107], it seems quite unlikely that silicone based digital computers will ever be able to come close to simulating something like a cubic meter of ideal gas using molecular dynamics.

The MD method can include chemical reactions between molecules and has found applications in chemistry and biology. For example, MD simulations can be used to simulate the interaction between various proteins or drugs and DNA [176]. In these situations, there are a much smaller number of molecules of interest that interact in much more complicated ways than in typical gases. A large statistical sample may not be required. Despite the fact that it is not feasible to simulate engineering scale fluids problems using molecular dynamics, the method is used in order to predict properties that are used in higher level models. Some simple examples of such properties are the dependence of viscosity or thermal coefficient on temperature and density which can be deduced analytically for some simplified models such as hard spheres but must be calculated or measured for more complicated gases. Tokumasu and Matsumoto [245] use MD simulations to determine several properties of non-polar diatomic molecules at a moderate temperature including the collision cross section and the probability density functions of translational and rotational energy after collision. These parameters can then be used in DSMC models.

Molecular dynamics is itself only an approximation to the more complicated underlying quantum mechanics, the numerical simulation of which is generally considered to be intractable [78]. Each quantum particle is described by a three dimensional probability distribution rather than a point location. The problem becomes exponentially more difficult because the probability of a particle existing at a point depends on the state of each other particle in the system. The number of variables required to simply store the probability distributions of N particles at P discrete locations is P^N. Since the number of discrete locations must be at least N but likely some large constant factor greater, the complexity is $\mathcal{O}(N^N)$. Each subatomic particle must also be simulated so for example, one Helium atom is actually 6 particles. Storage for the wave function of only two Helium atoms on a $10 \times 10 \times 10$ three dimensional grid (probably far too small) would require around 1×10^{15} GB, already unfeasible. Increasing the number of Helium atoms to four and the number of grid points to 2000 requires 1.6^{43} GB, 25 orders of magnitude greater than the cubic meter of ideal gas was already established as likely impossible. There is some optimism that quantum simulators will lead to advances in this field [145, 167, 53].

Leaving aside quantum considerations which are not used in practice, since MD simulations are computationally expensive, means of simplification are sought. One such technique is to replace the gas with an equivalent gas that has a lower number density. This procedure requires a detailed knowledge about the relationship between the fluid properties and the molecular model in order to fine tune any free parameters while keeping the desired fluid properties constant if possible. Chapman-Enskog [36] theory predicts that for a small Knudsen number, the viscosity a dilute gas of hard spheres is,

$$\mu = \frac{5}{16}\sqrt{\frac{R_{\mathrm{u}}T}{M\pi}}\frac{m}{d^2} = \frac{5}{16}\sqrt{\frac{mk_{\mathrm{B}}T}{\pi}}\frac{1}{d^2}.$$

If the number density n is multiplied by α, the mass must also be multiplied by α^{-1} in order to to keep density constant. In order to maintain viscosity, d must be multiplied by $\alpha^{\frac{-1}{4}}$. The molecular spacing is $n^{\frac{-1}{3}}$ so the ratio of molecular spacing to diameter will scale like,

$$\frac{\delta}{d} \sim \alpha^{\frac{1}{4}-\frac{1}{3}} = \alpha^{\frac{-1}{12}}.$$

The gas will become more dilute and the Knudsen number will increase as the number density is decreased. The Chapman-Enskog procedure used to predict the viscosity becomes invalid at higher Knudsen numbers placing a limit on this sort of replacement.

This process of replacing a fluid by an equivalent one that is easier to simulate may not always be possible particularly if a simple equation governing the relevant properties such as the one above is not available. Rather than replacing a number of smaller molecules by a larger molecule, the Boltzmann equation and DSMC methods replace a collection of molecules by a basis function representing their statistical properties.

2.3 Continuous conservation equations

When the smallest length scales of the problem are large enough and contain a sufficient number of molecules, the macroscopic behaviour becomes independent from the underlying particle behavior and can be described in terms of continuous density, velocity and energy fields. Just as the mass, momentum and energy of individual particles are conserved, so too are the corresponding fields which are defined over small control volumes,

$$\begin{aligned} \rho &= \frac{1}{\Delta V} \sum_{\boldsymbol{i} \in V} m_{\boldsymbol{i}}, \\ \rho u_\alpha &= \frac{1}{\Delta V} \sum_{\boldsymbol{i} \in V} m_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha}. \end{aligned}$$

These control volumes are made arbitrarily small in the formulation of differential equations using the continuous approach. Despite this, they must also be considered large enough to contain a sufficient number of molecules to neglect statistical fluctuations. Density represents the mass per unit volume and the stream velocity u_α is the average particle velocity. The total energy per unit mass of the system of particles e [J kg^{-1}] can be broken down into a thermal component e_{th} and a hydrodynamic component e_{hyd} which is a function of the stream velocity. The thermal component can be further broken down into a translational component e_{tr} which is responsible for fluid pressure and an internal component e_{int} which represents energy stored in the internal degrees of freedom such as vibration and rotation.

$$\begin{aligned} e &= e_{\text{hyd}} + e_{\text{th}}, \\ e_{\text{th}} &= e_{\text{tr}} + e_{\text{int}}. \end{aligned}$$

In this thesis we will not consider any cases where the particles have internal degrees of freedom, in these cases e_{int} is zero. The translational thermal energy is a function of the thermal velocity (also referred to as the peculiar velocity) $c'_{\boldsymbol{i}\alpha}$ which represents the particle's deviation from the stream velocity due to thermal fluctuations.

$$\begin{aligned} c'_{\boldsymbol{i}\alpha} &= c_{\boldsymbol{i}\alpha} - u_\alpha, \\ \rho e_{\text{tr}} &= \frac{1}{2\Delta V} \sum_{\boldsymbol{i}\in V} m_{\boldsymbol{i}} c'_{\boldsymbol{i}\alpha} c'_{\boldsymbol{i}\alpha}, \\ \rho e_{\text{hyd}} &= \frac{1}{2} \rho u_\alpha u_\alpha. \end{aligned}$$

The total energy due to particle velocity e_c is the sum of the translational thermal energy and the hydrodynamic energy,

$$\begin{aligned} \rho e_c &= \rho e_{\text{tr}} + \rho e_{\text{hyd}}, \\ &= \frac{1}{2\Delta V} \sum_{\boldsymbol{i}\in V} m_{\boldsymbol{i}} \left(c_{\boldsymbol{i}\alpha} - u_\alpha\right)\left(c_{\boldsymbol{i}\alpha} - u_\alpha\right) + \frac{1}{2}\rho u_\alpha u_\alpha, \\ &= \frac{1}{2\Delta V} \sum_{\boldsymbol{i}\in V} m_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\alpha} - u_\alpha \frac{1}{\Delta V} \sum_{\boldsymbol{i}\in V} m_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} + u_\alpha u_\alpha \frac{1}{2\Delta V} \sum_{\boldsymbol{i}\in V} m_{\boldsymbol{i}} + \frac{1}{2}\rho u_\alpha u_\alpha, \\ &= \frac{1}{2\Delta V} \sum_{\boldsymbol{i}\in V} m_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\alpha}. \end{aligned}$$

The thermal energy of an ideal gas is related to temperature. According to the equipartition theorem, at equilibrium, the thermal energy is equally distributed amongst each of the ζ degrees of freedom [16],

$$
\begin{aligned}
e_{\text{th}} &= \frac{\zeta}{2} RT, \\
e_{\text{tr}} &= \frac{3}{2} RT.
\end{aligned}
$$

Since each degree of freedom contains the same amount of energy, one temperature can fully describe the fluid. For an ideal gas, this temperature links the pressure and density via the ideal gas equation of state,

$$
P = \rho RT = \frac{2}{3} \rho e_{\text{tr}}.
$$

The conservation equations for a system of particles will now be introduced. We will assume a system of homogeneous particles with mass m. The probability distribution function (PDF) for particles $f(c_\alpha, x_\alpha, t)$ gives the expected number of particles with velocity between c_α and $c_\alpha + dc_\alpha$ and with position between x_α and $x_\alpha + dx_\alpha$. The hydrodynamic variables, previously defined over control volumes can now be defined as,

$$
\begin{aligned}
\rho &= m \int f \; dc_\alpha, \\
\rho u_\alpha &= m \int c_\alpha f \; dc_\alpha, \\
\rho e_c &= m \frac{1}{2} \int c_\alpha c_\alpha f \; dc_\alpha.
\end{aligned}
$$

The transport equation for the PDF is,

$$
\frac{\partial f}{\partial t} + c_\alpha \frac{\partial f}{\partial x_\alpha} + \frac{F_\alpha}{\rho} \frac{\partial f}{\partial c_\alpha} = \mathcal{C}(f). \tag{2.1}
$$

The collision operator $\mathcal{C}(f)$ represents a gain and loss due to collisions with other molecules. The collision operator conserves mass, momentum and energy. The second and third terms represent advection of particles and the effects of a body force F_α. These terms can be understood by considering the total differential of f in phase space,

$$df = \frac{\partial f}{\partial t}dt + \frac{\partial f}{\partial x_\alpha}dx_\alpha + \frac{\partial f}{\partial c_\alpha}dc_\alpha, \tag{2.2}$$

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + c_\alpha\frac{\partial f}{\partial x_\alpha} + \frac{F_\alpha}{\rho}\frac{\partial f}{\partial c_\alpha}. \tag{2.3}$$

This transport equation is a form of Liouville's theorem.

The PDF can be discretized over a set of distinct velocities, the hydrodynamic variables are then expressed as sums over the discrete PDF values and discrete velocities,

$$\rho = m\sum_{\boldsymbol{i}=0}^{q} f_{\boldsymbol{i}},$$

$$\rho u_\alpha = m\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} f_{\boldsymbol{i}},$$

$$\rho e_c = m\frac{1}{2}\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\alpha} f_{\boldsymbol{i}}.$$

The corresponding transport equation is,

$$\partial t f_{\boldsymbol{i}} + c_{\boldsymbol{i}\alpha}\partial_\alpha f_{\boldsymbol{i}} = \mathcal{C}_{\boldsymbol{i}} + \mathcal{F}_{\boldsymbol{i}} \tag{2.4}$$

The force term is now written as a discrete operator since the basis particles cannot be accelerated, their velocity is constant. By considering the integral of the force term in the continuous transport equations, the desired moments of the discrete operator may be set equal to the moments of the continuous force term,

$$\int q(c_\alpha)\frac{F_\alpha}{\rho}\frac{\partial f}{\partial c_\alpha}dc_\alpha = \sum_{\boldsymbol{i}=0}^{q} q_{\boldsymbol{i}}\mathcal{F}_{\boldsymbol{i}}.$$

Integration by parts is used and the PDF is assumed to approaches zero as velocity approaches ∞,

$$\begin{aligned}\int q\frac{F_\alpha}{\rho}\frac{\partial f}{\partial c_\alpha}dc_\alpha &= \left[q\frac{F_\alpha}{\rho}f\right]_{-\infty}^{+\infty} - \frac{F_\alpha}{\rho}\int f\frac{\partial q}{\partial c_\alpha}dc_\alpha \\ &= \frac{F_\alpha}{\rho}\int f\frac{\partial q}{\partial c_\alpha}dc_\alpha.\end{aligned}$$

By setting $q_{\boldsymbol{i}}$ to be m, $mc_{\boldsymbol{i}\alpha}$ and $\frac{1}{2}mc_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}$ and using

$$\partial_\alpha c_\beta = \delta_{\alpha\beta} \tag{2.5}$$

$$\partial_\alpha c_\beta c_\gamma = \delta_{\alpha\beta}c_\gamma + \delta_{\alpha\gamma}c_\beta \tag{2.6}$$

$$\partial_\alpha c_\beta c_\gamma c_\delta = \delta_{\alpha\beta}c_\gamma c_\delta + \delta_{\alpha\gamma}c_\beta c_\delta + \delta_{\alpha\delta}c_\beta c_\gamma \tag{2.7}$$

we arrive at the correct moments for the force operator,

$$\begin{aligned}m\sum_{\boldsymbol{i}=0}^{q}\mathcal{F}_{\boldsymbol{i}} &= 0 \\ m\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}\mathcal{F}_{\boldsymbol{i}} &= F_\alpha \\ m\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}\mathcal{F}_{\boldsymbol{i}} &= u_\alpha F_\beta + u_\beta F_\alpha \\ m\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\gamma}\mathcal{F}_{\boldsymbol{i}} &= \delta_{\alpha\delta}(RT\delta_{\beta\gamma} + u_\beta u_\gamma)F_\delta + \delta_{\beta\delta}(RT\delta_{\alpha\gamma} + u_\alpha u_\gamma)F_\delta \\ &\quad + \delta_{\gamma\delta}(RT\delta_{\alpha\beta} + u_\alpha u_\beta)F_\delta \\ m\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}\mathcal{F}_{\boldsymbol{i}} &= 5RTF_\alpha + u_\beta u_\beta F_\alpha + 2u_\alpha u_\beta F_\beta \\ &= 2eF_\alpha + 2RTF_\alpha + 2u_\alpha u_\beta F_\beta\end{aligned}$$

If internal energy is included, then

$$m\sum_{\boldsymbol{i}=0}^{q}e_{\mathrm{int},\boldsymbol{i}}\mathcal{F}_{\boldsymbol{i}} = 0$$

since the force term is independent of internal energy.

Without making any further assumptions about the behaviour of the particles other than that they can be described in terms of a continuous PDF and considering the conservation of mass, momentum and energy during collisions, it is possible to arrive at a set of continuous conservation equations that describe the fluid flow. Summing Eqn. 2.4 over all velocities weighted by mass gives the continuous conservation of mass equation,

$$\partial_t \rho + \partial_\alpha \rho u_\alpha = 0.$$

Since the velocity is independent of position, we can move the velocity inside the derivative,

$$m \sum_{i=0}^{q} c_{i\alpha} \partial_\alpha f_i = m \partial_\alpha \sum_{i=0}^{q} c_{i\alpha} f_i$$

The continuous analogue makes this clearer, velocity is another phase variable like position. It is also the integration variable,

$$\int c_\alpha \partial_\alpha f dc_\alpha = \partial_\alpha \int c_\alpha f dc_\alpha.$$

Summing the same equation weighted by $mc_{i\alpha}$ gives the equation for continuous conservation of momentum,

$$\partial_t \rho u_\alpha + \partial_\beta \Pi_{\alpha\beta} = F_\alpha.$$

The momentum flux tensor is defined as,

$$\Pi_{\alpha\beta} = m \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} f_i$$

Substituting the definition for peculiar velocity and using the knowledge that,

$$\sum_{i=0}^{q} c'_{i\alpha} = 0$$

leads to,

$$\partial_t \rho u_\alpha + \partial_\beta \rho u_\alpha u_\beta - \partial_\beta \sigma_{\alpha\beta} = 0.$$

The stress tensor,

$$\sigma_{\alpha\beta} = -m \sum_{\boldsymbol{i}=0}^{q} c'_{\boldsymbol{i}\alpha} c'_{\boldsymbol{i}\beta} f_{\boldsymbol{i}},$$

can be further decomposed into an isotropic component P' and a traceless component σ^s,

$$P' = m\frac{1}{3} \sum_{\boldsymbol{i}=0}^{q} c'_{\boldsymbol{i}\alpha} c'_{\boldsymbol{i}\alpha} f_{\boldsymbol{i}} = \frac{2}{3}\rho e_{\mathrm{tr}}.$$

$$\sigma_{\alpha\beta} = \sigma^s_{\alpha\beta} - P'.$$

This fluid pressure P', defined in terms of the translational velocity is only equal to the thermodynamic pressure at equilibrium when the equipartition of energy theorem is valid. One important case when this is not true is for polyatomic gases where the bulk viscosity is not zero. For these gases, the temperature corresponding to internal degrees of freedom may not have time to reach equilibrium with the translational energy. In this case, the pressure tensor while still defined in terms of the translational energy now must contain a correction proportional to the volumetric strain and the bulk viscosity.

$$\begin{aligned} P' &= P - \sigma^v, \\ &= \rho RT - \mu^v \partial_\alpha u_\alpha \end{aligned}$$

Except for simple cases such as monatomic gases, the bulk viscosity is difficult to predict theoretically and to measure experimentally (Section 2.6).

Weighting the sum by the total particle energy, $\frac{1}{2} m c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\beta} + m e_{\mathrm{int},\boldsymbol{i}}$ gives,

$$\partial_t \rho e + \partial_\alpha \frac{1}{2} m \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\alpha} f_{\boldsymbol{i}} + \partial_\alpha m \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} e_{\mathrm{int},\boldsymbol{i}} f_{\boldsymbol{i}} = u_\alpha F_\alpha.$$

Substituting the peculiar velocity,

$$\partial_t \rho e + \partial_\alpha \frac{1}{2} m \sum_{\boldsymbol{i}=0}^{q} (c'_{\boldsymbol{i}\beta} c'_{\boldsymbol{i}\beta} + 2u_\beta c'_{\boldsymbol{i}\beta} + u_\beta u_\beta)(c'_{\boldsymbol{i}\alpha} + u_\alpha) f_{\boldsymbol{i}} \\ + \partial_\alpha u_\alpha \rho e_{\text{int}} + \partial_\alpha m \sum_{\boldsymbol{i}=0}^{q} c'_{\boldsymbol{i}\alpha} e_{\text{int},\boldsymbol{i}} f_{\boldsymbol{i}} = u_\alpha F_\alpha .$$

Introducing the heat flux tensor,

$$q_\alpha = m \sum_{\boldsymbol{i}=0}^{q} c'_{\boldsymbol{i}\alpha} \left(\frac{1}{2} c'_{\boldsymbol{i}\beta} c'_{\boldsymbol{i}\beta} + e_{\text{int},\boldsymbol{i}} \right) f_{\boldsymbol{i}}$$

and using previous definition of the stress tensor leads to,

$$\partial_t \rho e + \partial_\alpha u_\alpha \rho e - \partial_\alpha u_\beta \sigma_{\alpha\beta} + \partial_\alpha q_\alpha = u_\alpha F_\alpha .$$

Splitting the stress tensor as before and summarising all three equations,

$$\boxed{\begin{gathered} \partial_t \rho + \underbrace{\partial_\alpha \rho u_\alpha}_{\text{divergence}} = 0 \\ \partial_t \rho u_\alpha + \underbrace{\partial_\beta \rho u_\alpha u_\beta}_{\substack{\text{convective}\\ \text{acceleration}}} + \underbrace{\partial_\alpha P'}_{\text{pressure}} - \underbrace{\partial_\beta \sigma^s_{\alpha\beta}}_{\substack{\text{shear}\\ \text{stress}}} = F_\alpha \\ \partial_t \rho e + \underbrace{\partial_\alpha u_\alpha \rho e}_{\text{advection}} + \underbrace{\partial_\alpha u_\alpha P'}_{\substack{\text{flow}\\ \text{work}}} - \underbrace{\partial_\alpha u_\beta \sigma^s_{\alpha\beta}}_{\substack{\text{viscous}\\ \text{dissipation}}} + \underbrace{\partial_\alpha q_\alpha}_{\substack{\text{heat}\\ \text{flux}}} = u_\alpha F_\alpha \end{gathered}} \qquad \begin{gathered} (2.8) \\ (2.9) \\ (2.10) \end{gathered}$$

The first two terms in the energy equation give an advection equation for energy, the third term gives the change in energy due to flow work. The fourth term is the viscous dissipation due to the stress tensor, the change in momentum created by the stress tensor in the momentum equations also causes a change in energy, transferring kinetic energy from the bulk fluid motion to thermal energy. The final term, the heat flux at this stage is a general flux term, represents the diffusion of thermal energy.

The conservation equations for a system of particles are remarkably close to the Navier-Stokes equations, all that is required is a description of the stress tensor $\sigma_{\alpha\beta}$ and the heat flux tensor q_α in terms of the hydrodynamic variables ρ, u_α and e.

2.4 The Boltzmann equation

The Boltzmann equation is the combination of the previously described transport equation with Boltzmann's famous collision integral,

$$\frac{\partial f}{\partial t} + c_\alpha \frac{\partial f}{\partial x_\alpha} + \frac{F_\alpha}{\rho}\frac{\partial f}{\partial c_\alpha} = \iiint_{v_1} \iint_{\Omega} (\underbrace{f' f_1'}_{\text{gain}} - \underbrace{f f_1}_{\text{loss}}) v_{\text{rel}} \frac{d\sigma}{d\Omega} d\Omega dv_1. \tag{2.11}$$

The loss component is the integral over all other velocities v_1 of the two particle probability ff_1 representing the likelihood of finding two particles with the specified velocities. The two particle probability is assumed to be the product of two single particle PDFs $f_{12} = f_1 f_2$ in what is known as the molecular chaos or Stosszahlansatz assumption. This means that particle populations are uncorrelated, the PDF of one velocity is completely unrelated to another. The loss term is then multiplied by the relative velocity, particles with a higher relative velocity are more likely to collide. The final factor is the differential cross section, since each other term is independent of the solid angle the integral over all solid angles can be replaced by the total collision cross section σ_T,

$$\sigma_T = \int_0^{4\pi} \frac{d\sigma}{d\Omega} d\Omega$$

The gain component is similar however it integrates over all possible post collision velocities, these vary with solid angle and with the other pre-collision velocity v_1. If the collisions are isotropic, scattering molecules into all solid angles with equal probability, then the collision integral can be simplified,

$$\mathcal{C}(f) = \iiint_{v_1} (f' f_1' - f f_1) v_{\text{rel}} \sigma_T dv_1.$$

If the momenta and positions of the particles were known precisely enough, then in principle the exact post collision velocities could be calculated as in molecular dynamics simulations. This is usually described in terms of an impact parameter b and a scattering angle θ as shown in Figure 2.2. Small uncertainties in the impact parameter can lead to large uncertainties in scattering angle. The Boltzmann equation describes the collision process in terms of an average over a large number of particles. By definition, the regions over which the equation is defined must be significantly larger than the size of a single molecule making it impossible to predict the impact parameter and hence all values must be considered. As an aside, due to Heisenberg's uncertainty principle, it is not possible to know a molecule's position and momentum precisely so it is useful that the statistical mechanics approach does not depend on the exact positions.

2.4.1 Hard sphere gas collisions

The simplest possible method used to model collisions between gas molecules assumes that they are hard spheres. There is no force between molecules until they make contact at which point they bounce off each other. The velocity of the center of mass remains constant as does the magnitude of the relative velocity,

$$\begin{gathered} c_{\mathrm{cm},\alpha} = \frac{1}{2}(c_{1,\alpha} + c_{2,\alpha}), \\ c_{\mathrm{rel},\alpha} = c_{1,\alpha} - c_{2,\alpha}. \end{gathered}$$

All that changes during collision is the angle of the relative velocity so the post collision relative velocities are,

$$\begin{aligned} c'_{\mathrm{rel},x} &= |c_{\mathrm{rel}}| \sin\zeta \cos\phi, \\ c'_{\mathrm{rel},y} &= |c_{\mathrm{rel}}| \sin\zeta \sin\phi, \\ c'_{\mathrm{rel},z} &= |c_{\mathrm{rel}}| \cos\zeta. \end{aligned} \tag{2.12}$$

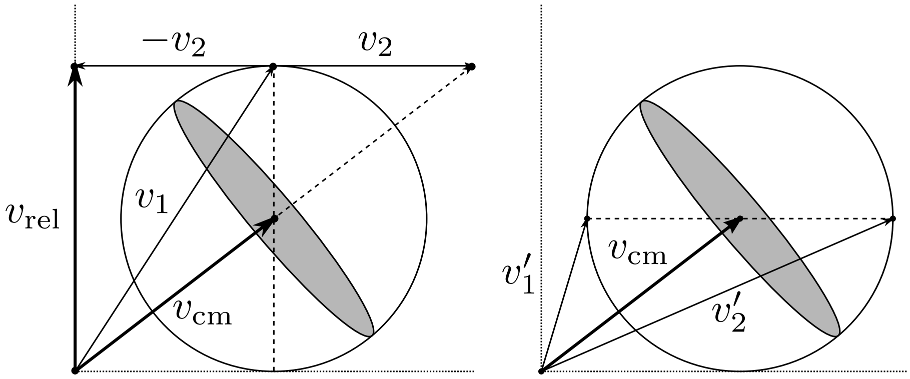


Figure 2.1: Collision sphere representing all possible post collision velocities.

The post collision velocities are,

$$
\begin{aligned}
c_{1,\alpha} &= c_{\mathrm{cm},\alpha} + \frac{1}{2}c'_{\mathrm{rel},\alpha}, \\
c_{2,\alpha} &= c_{\mathrm{cm},\alpha} - \frac{1}{2}c'_{\mathrm{rel},\alpha}.
\end{aligned}
$$

This is illustrated by the collision sphere shown in Figure 2.1. The diameter of the sphere is the relative velocity and it is centered around the center of mass so the range of post collision velocities v_1' and v_2' can be found by taking any two diametrically opposite points on that sphere.

The angle between the incoming and outgoing velocities is referred to as the scattering angle θ and will lie in the collision plane. It is important to predict the functional relationship between the impact parameter b and the scattering angle. Figure 2.2 shows a typical collision in the center of mass frame so that the average velocity appears to be zero. The relationship between impact parameter and scattering angle is,

$$
\begin{aligned}
\frac{b}{2r} &= \sin(\alpha), \\
\frac{b}{2r} &= \sin\left(90 - \frac{\theta}{2}\right), \\
b &= 2r\cos\left(\frac{\theta}{2}\right).
\end{aligned}
$$

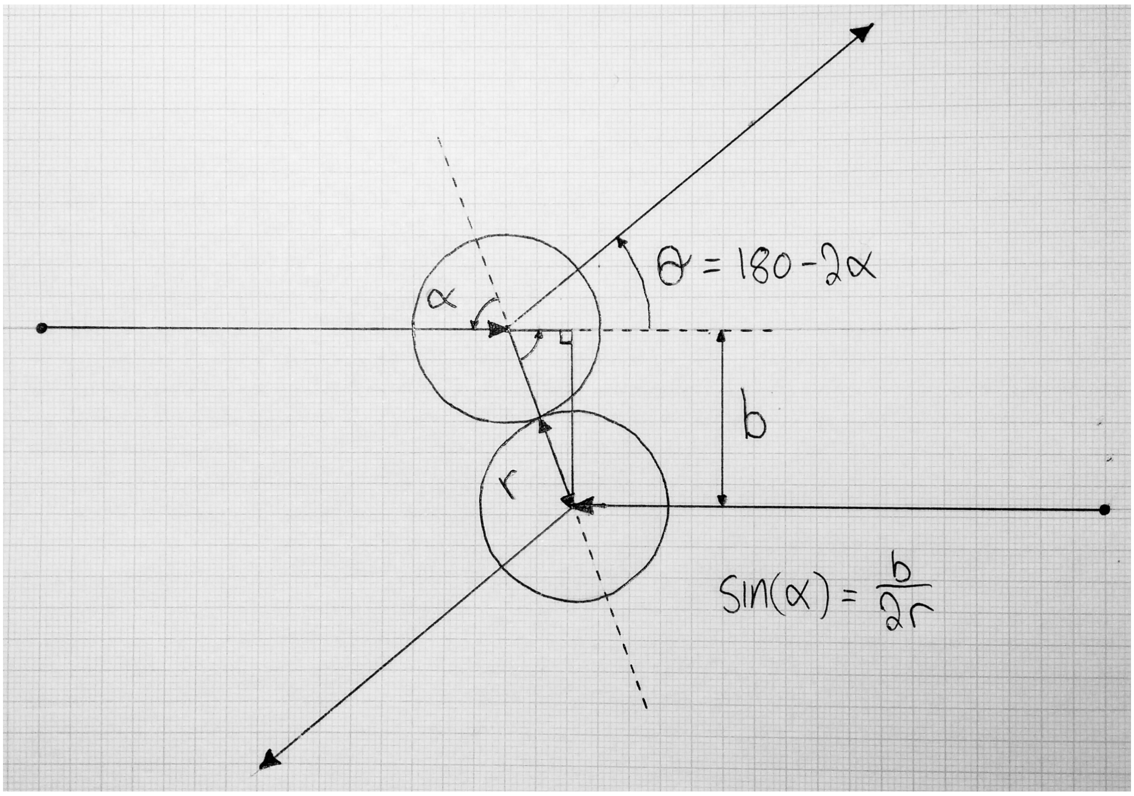


Figure 2.2: Collision between two hard shell molecules in the center of mass frame.

This relationship is used to calculate the differential cross section which is used in the Boltzmann equation to represent the probability of two particles scattering into a particular solid angle.

The differential cross section is calculated using the formula (see Chapter 14.5 in Taylor [240]),

$$\frac{d\sigma}{d\Omega} = \frac{b}{\sin(\theta)} \left| \frac{db}{d\theta} \right| .$$

For hard spheres, this reduces to,

$$\frac{d\sigma}{d\Omega} = r^2 \frac{2\cos\left(\frac{\theta}{2}\right)\sin\left(\frac{\theta}{2}\right)}{\sin(\theta)} = r^2.$$

The differential cross section does not depend on the angle so the scattering is isotropic. The total collision cross section is,

$$\sigma_T = \int_0^{4\pi} \frac{d\sigma}{d\Omega} d\Omega = 4\pi r^2.$$

Similar calculations may be performed for other models such as the inverse power law.

2.4.2 Chapman-Enskog expansion

The Boltzmann equation has been studied extensively using both theoretical and numerical techniques. Efforts have focused on a variety questions ranging from statistical mechanics and thermodynamics through to fluid dynamics with considerable overlap. Some properties of the Boltzmann equation can be solved analytically in order to derive the gas properties such as viscosity. It can be shown that an ideal gas at thermal equilibrium will have a Maxwellian velocity distribution,

$$f^{\mathrm{eq}}(c'_\alpha) = \rho \left(\frac{1}{2\pi RT} \right)^{\frac{3}{2}} \exp\left(-\frac{|c'_\alpha|^2}{2RT} \right)$$

The Chapman-Enskog (CE) procedure expresses the particle population in terms of its departure from the equilibrium distribution as a power series in some small parameter ϵ that can be taken to be the Knudsen number,

$$f = f^{\mathrm{eq}} + \epsilon f^{(1)} + \epsilon^2 f^{(2)} + \dots$$

Assuming that the fluid is close to equilibrium allows one to discard all terms of higher order than ϵ^2 making it then possible to calculate the stress tensor, heat flux tensor and transport coefficients for a variety of collision models including the simple hard sphere model [6, 95]. Some other examples of the CE procedure are covered in textbooks [16, 195, 36, 34] and early research papers [33, 71]. Muntz [161] also gives a review of several methods of solution for the Boltzmann equation.

2.4.3 Grad's method

Grad's method [89, 88, 36] is another technique developed to analyse the Boltzmann equation. Grad continues the process of taking moments of the Boltzmann equation to include the second order, third order and potentially even higher order moments. The resulting equations express the rate of change of each moment in terms of a flux term and in the case of non-conserved moments, also a source term due to collisions. This process can be continued up to higher and higher rank tensors, each balance equation depending on successively higher rank tensors. Reinecke and Kremer [196] give a generalization of the method that goes up to an arbitrary order. Closure is achieved by expressing the equilibrium distribution in terms of a finite order Hermite polynomial. Due to the orthogonality of the Hermite polynomials, it works out that the Hermite coefficients at a particular order are related to the moments of the same order. Thus by limiting the representation to a particular order, all higher order moments can be expressed in terms of lower order moments. Grad's method usually refers to a 13 moment system with third order Hermite polynomials where the third order coefficients are proportional to the rank three heat flux tensor. Reinecke and Kremer generalize the method to $13 + 9N$ moments and $3 + N$ order Hermite polynomials.

Assuming that the pressure tensor and heat flux tensor are negligible unless divided by the relaxation time, the Chapman-Enskog values are obtained. The two methods give the same results.

Since the Grad moment equations may be formulated using an arbitrary number of moments and with the Navier-Stokes equations and the Burnett equations being obtained by lower order approximations, it is expected that as the order of approximation is increased, the Grad moment equations can more closely approach the true solution to the Boltzmann equation than lower order approximations. One of the ambitions of this method is to represent hydrodynamics far from equilibrium such as shock waves, more accurrately than the Navier-Stokes equations. Continuous solutions for a finite shock thickness are sought. Grad showed early on that his 13 moment method does not admit a continuous solution to shock waves above a critical Mach number [90] and it has more recently been shown that adding moments does not improve this situation [203].

The second order Chapman-Enskog solution possesses a so called H-theorem, a property H which represents entropy can be shown to increase with time as the system approaches equilibrium [36]. Neither the higher order Chapman-Enskog solutions nor the higher order Grad methods have been shown to have an H-theorem and hence may not increase entropy. Bobylev [17] has shown that the linearised Burnett equations do possess a H-theorem and Struchtrup et al. [233] have shown the same for linearised Grad equations.

2.4.4 Computational techniques

There have been a large variety of computational methods used to approximately solve the Boltzmann equation. The main sources of difficulty are the highly multidimensional collision integral and the discretization of phase space. Attempts at solution can be very roughly split into two categories, attempts to simulate the Boltzmann equation as accurately as possible and attempts to simplify the simulation as much as possible while retaining some desired quality. Accurate simulations are extremely computationally intense and severely limit the resolution of the domain. The discrete velocity Boltzmann simulations of Morris et al. [159] discretize phase space into as many as 100 velocities in each direction, a total of 10,000 velocities. Despite their method being accelerated by using a Monte Carlo approach to evaluate the collision integral, they are still limited to a small physical domain of 200 cells in one dimension. Meanwhile, the simplest lattice Boltzmann methods represent phase space using only 19 velocities and replace the collision operator with a single relaxation towards equilibrium. These simulations only afford a very rough approximation of the Boltzmann equation, approaching the athermal Navier-Stokes equations in the low Mach number limit. They do not correctly model the energy equation let alone non-equilibrium effects. The ease of calculation allows domains comprising tens or hundreds of millions of cells and the direct numerical simulation of turbulent flows [262].

The most direct approach is to discretize the entire phase space, solve the differential transport equations which is easy and then solve the collision integral which is extremely difficult. This is referred to as the discrete velocity Boltzmann method. Palczewski et al. [168] have proven the consistency of these schemes. A common way to accelerate the evaluation of the collision integral is to use a Monte Carlo sampling where a random subset of collisions are calculated. An early example of discrete velocity Boltzmann scheme with Monte Carlo collisions was by Yen [259] in 1971. Inamuro and Sturtevant [112] solved the collision term directly. Velocity space was discretized over as many as 13^3 discrete velocities and shock wave simulations were performed over 60 cells in one dimension. Platkowski and Walus [184] detail an efficient technique used to calculate the collision term and have simulated a one dimensional shock tube with 10^3 discrete velocities and 50 cells. They have also verified that their scheme correctly reproduced the analytic solution for the time evolution of the spatially homogeneous Boltzmann equation with a particular initial condition. Kowalczyk et al. [128] compare discrete velocity Boltzmann schemes against molecular dynamics and direct simulation Monte Carlo simulations showing general agreement between the methods. Morris et al. [159] use Monte Carlo collisions for their discrete velocity method and present an interpolation technique to solve the problem of post collision velocities that do not lie on the grid. They also simulate a Mach 10 shock which requires a larger range of particle velocities for the high temperature gas and a fine resolution of the velocity space for the low temperature upstream gas. Their simulation used 100^3 velocity points and 200 cells and found good agreement with accepted DSMC results. Discrete velocity simulations take longer than a corresponding DSMC simulation but provide a smooth unsteady solution in contrast to the noisy solution of DSMC methods.

The collision operator can be greatly simplified by replacing it with a relaxation towards local equilibrium.

$$\mathcal{C}(f) = \omega \left(f^{\mathrm{eq}} - f\right)$$

This is also referred to as the Bhatnager Gross Krook collision (BGK) operator [13] or the single relaxation time SRT collision operator. This assumption naturally limits the region of applicability by requiring that the fluid remain close to equilibrium however the requirements for the discretization of phase space can be significantly relaxed because now all that is required is the representation of the moments that appear in the CE expansion. The SRT collision operator is a first order approximation to the Boltzmann collision integral. Higher order approximations [179, 95] are not used much in practice even though they offer improved accuracy.

If the velocity set is chosen so that particles stream exactly from one lattice site to the next, then the transport equation is simplified resulting in lattice Boltzmann methods [12]. If the discrete velocities are chosen freely and do not land on other lattice sites, these are referred to as off-lattice Boltzmann methods. Surmas et al. [237] detail a number of lattices of each type. Lattice Boltzmann methods will be discussed in more detail in Chapter 3.

Lattice gas cellular automata (LGCA) are only mentioned in passing for their historical significance. Discrete particles travel along a grid and are allowed to collide if more than one particle meet at a site. Collisions are calculated based on the conservation of mass, momentum and energy. The fluid behaviour naturally emerges if the lattice possesses sufficient isotropy [65, 64, 67, 82, 66]. There is a large amount of statistical scatter. It is remarkable that a simple two dimensional model with only 6 possible velocities can produce a turbulent energy cascade similar to real hydrodynamic behaviour [70]. One interesting feature is that while a 6 velocity hexagonal lattice can be used in two dimensions, there is no suitable lattice in three dimensions. It is possible to use a four dimensional face centered hypercube with 24 velocities for three dimensional simulations [66]. These methods predate lattice Boltzmann methods and have also been entirely superseded by them.

2.5 Direct simulation Monte Carlo (DSMC) method

Direct simulation Monte Carlo (DSMC) methods were developed by Graeme Bird [16] as a means of simulating gas dynamics at a particle level, with a greater computational efficiency than molecular dynamics. The summary below is sourced largely from Bird's book and a succinct paper summarising the method by Alexander and Garcia [4]. The DSMC method tracks the motion of virtual particles that each represent a collection of F_N real particles as they undergo a series of collision and streaming steps. The streaming step occurs over a discrete time step during which particles freely stream according to their velocity and any external force, the time step must be small enough that the particles are unlikely to have collided during that time. The domain is discretized into cells in which collisions are calculated in a probabilistic fashion. Since each virtual particle represents many real particles, the collisions are calculated on the basis that the position of each particle in the cell is not known exactly. The probability of any two particles in the cell colliding is given by,

$$P_{\text{col}} = F_N \sigma_T c_r \frac{\Delta t}{V_c}$$

where σ_T is the total collision cross section, c_r is the magnitude of the relative velocity and V_c is the volume of the cell.

Rather than testing each pair of particles, there are several ways to improve computational efficiency. The NTC method calculates an expected maximum number of collisions,

$$\frac{1}{2} N \bar{N} F_N (\sigma_T c_r)_{\text{max}} \frac{\Delta t}{V_c}.$$

where N is the number of particles in the cell and $\bar{N}$ is the average number of particles in the cell. This number of pairs are selected at random as collision candidates. The probability of a collision occurring between candidates is then,

$$\frac{\sigma_T c_r}{(\sigma_T c_r)_{\text{max}}}$$

The post collision velocities are calculated based on the molecular model, for example, we have already described hard spheres in Section 2.4.1.

For molecules with no internal energy, the conservation of momentum and energy lead to post collision velocities that lie on the collision sphere with the formula given in Equations 2.12. For a deterministic collision, the scattering angle depends on the impact parameter however since the exact locations are not known, these angles are chosen from a random variable to give the expected probability distribution. The scattering for hard spheres is isotropic so ϕ is uniformly distributed between 0 and 2π and ζ is a random variable with the probability distribution,

$$P(\zeta) = \frac{1}{2} \sin \zeta$$

The desired distribution for ζ can be achieved by creating a random variable q distributed between 0 and 1 and calculating,

$$\begin{aligned} \cos \zeta &= q, \\ \sin \zeta &= \sqrt{1 - q^2}. \end{aligned}$$

The virtual particles continue to stream and collide in this way and the hydrodynamic variables are calculated by summing over control volumes as set out in Section 2.3.

The hydrodynamic fields will have some statistical noise inversely proportional to the number of virtual particles present in the cell. For typical simulations with around 20 particles, the scatter is significant. A variety of techniques can be used to smooth the results if a steady state solution is sought. The noise may be removed by taking a large temporal average. For a deterministic unsteady flow, several simulations may be performed to arrive at an ensemble average. Another option is to average over some time interval although this may introduce smearing. Cave et al. [30] have an interesting technique with the acronym DREAM where an ensemble average is constructed about a desired point in time by repeatedly restarting the flow from some previous time.

Although the DSMC algorithm makes no mention of the Boltzmann equation, it is derived under similar dilute gas assumptions of two particle collisions and molecular chaos and uses similar probabilistic collisions. The Boltzmann equation makes the molecular chaos assumption explicit by replacing the two particle distribution function with the product of the corresponding single particle distribution functions$f_{12} = f_1 f_2$ while the DSMC method makes the assumption implicitly since the only factor affecting the likelihood of a collision between two particles of a particular velocity is the number of those particles in a cell. The virtual particles of the DSMC method can be considered as basis functions representing the particle PDF of the Boltzmann equation. It has been shown that solutions obtained using the DSMC method approach solutions to the Boltzmann equation as the number of particles increases [251, 15].

2.6 Continuum equations

Recalling the continuous conservation equations,

$$\boxed{\begin{gathered}
\partial_t \rho + \underbrace{\partial_\alpha \rho u_\alpha}_{\text{divergence}} = 0 \\
\partial_t \rho u_\alpha + \underbrace{\partial_\beta \rho u_\alpha u_\beta}_{\substack{\text{convective}\\ \text{acceleration}}} + \underbrace{\partial_\alpha P'}_{\text{pressure}} - \underbrace{\partial_\beta \sigma^s_{\alpha\beta}}_{\substack{\text{shear}\\ \text{stress}}} = F_\alpha \\
\partial_t \rho e + \underbrace{\partial_\alpha u_\alpha \rho e}_{\text{advection}} + \underbrace{\partial_\alpha u_\alpha P'}_{\substack{\text{flow}\\ \text{work}}} - \underbrace{\partial_\alpha u_\beta \sigma^s_{\alpha\beta}}_{\substack{\text{viscous}\\ \text{dissipation}}} + \underbrace{\partial_\alpha q_\alpha}_{\substack{\text{heat}\\ \text{flux}}} = u_\alpha F_\alpha
\end{gathered}}$$

The stress tensor $\sigma_{\alpha\beta}$ was previously decomposed into the pressure tensor P' and the traceless shear stress $\sigma^s_{\alpha\beta}$,

$$\sigma_{\alpha\beta} = \sigma^s_{\alpha\beta} - P'\delta_{\alpha\beta}.$$

Instead, we will express the stress tensor as a sum of the thermodynamic pressure P and the viscous stress σ' which has a traceless component $\sigma^s_{\alpha\beta}$ and an isotropic component $\sigma^v_{\alpha\beta}$,

$$\begin{aligned}\sigma_{\alpha\beta} &= \sigma'_{\alpha\beta} - P\delta_{\alpha\beta},\\ &= \sigma^s_{\alpha\beta} + \sigma^v_{\alpha\beta} - P\delta_{\alpha\beta}.\end{aligned}$$

The pressure tensor is the sum of the thermodynamic pressure and the volumetric viscous stress,

$$P' = P - \sigma^v_{\alpha\beta}.$$

The volumetric shear stress is zero for gases with no internal degrees of freedom and is sometimes neglected. We will further discuss this term later in this section.

The viscous stress tensor for a Newtonian fluid is linearly proportional to the rate of strain tensor $\epsilon_{\alpha\beta}$,

$$\begin{aligned}\sigma'_{\alpha\beta} &= \mu_{\alpha\beta\gamma\delta}\epsilon_{\gamma\delta}\\ \epsilon_{\alpha\beta} &= \frac{1}{2}\left(\partial_\alpha u_\beta + \partial_\beta u_\alpha\right)\end{aligned}$$

The rate of strain tensor can be expressed as a rate of volumetric expansion tensor $\epsilon^v_{\alpha\beta}$ and a traceless rate of shear tensor $\epsilon^s_{\alpha\beta}$ simply by subtracting the trace multiplied by the identity matrix $\delta_{\alpha\beta}$,

$$\begin{aligned}\epsilon_{\alpha\beta} &= \epsilon^v_{\alpha\beta} + \epsilon^s_{\alpha\beta},\\ \epsilon^v_{\alpha\beta} &= \frac{1}{3}\delta_{\alpha\beta}\partial_\gamma u_\gamma,\\ \epsilon^s_{\alpha\beta} &= \frac{1}{2}\left(\partial_\alpha u_\beta + \partial_\beta u_\alpha\right) - \frac{1}{3}\delta_{\alpha\beta}\partial_\gamma u_\gamma.\end{aligned}$$

Assuming an isotropic fluid, the viscosity must be an isotropic tensor, the general form for an isotropic tensor of rank four is,

$$\mu_{\alpha\beta\gamma\delta} = \mu^0 \delta_{\alpha\beta}\delta_{\gamma\delta} + \mu^1 \delta_{\alpha\gamma}\delta_{\beta\delta} + \mu^2 \delta_{\alpha\delta}\delta_{\beta\gamma}$$

This isotropy requirement significantly reduces the number of degrees of freedom required to specify viscosity from 3^4 to two, the shear viscosity,

$$\mu^s = \frac{1}{2}\left(\mu^1 + \mu^2\right)$$

and the second viscosity μ^0,

$$\begin{aligned}
\sigma'_{\alpha\beta} &= \mu^0 \delta_{\alpha\beta}\partial_\gamma u_\gamma + \frac{1}{2}\left(\mu^1 + \mu^2\right) \cdot \left(\partial_\alpha u_\beta + \partial_\beta u_\alpha\right) \\
&= \mu^0 \delta_{\alpha\beta}\partial_\gamma u_\gamma + \mu^s \left(\partial_\alpha u_\beta + \partial_\beta u_\alpha\right).
\end{aligned}$$

The viscous stress tensor can likewise be split into a volumetric viscous stress component $\sigma^v_{\alpha\beta}$ and a traceless viscous shear stress component $\sigma^s_{\alpha\beta}$,

$$\begin{aligned}
\sigma'_{\alpha\beta} &= \sigma^v_{\alpha\beta} + \sigma^s_{\alpha\beta}, \\
\sigma^v_{\alpha\beta} &= \mu^v \epsilon^v = \delta_{\alpha\beta}\left(\mu^0 + \frac{2}{3}\mu^s\right)\partial_\gamma u_\gamma, \\
\sigma^s_{\alpha\beta} &= \mu^s \epsilon^s = \mu^s \left(\partial_\alpha u_\beta + \partial_\beta u_\alpha - \frac{2}{3}\delta_{\alpha\beta}\partial_\gamma u_\gamma\right).
\end{aligned}$$

We will use μ^v for the bulk viscosity, a distinct term which is related to the second viscosity by,

$$\mu^v = \mu^0 + \frac{2}{3}\mu^s. \tag{2.13}$$

The volume viscosity coefficient represents the viscous resistance of the fluid to pure volumetric expansion or contraction. The gradient of the stress tensor is then,

$$\begin{aligned}
\partial_\beta \sigma_{\alpha\beta} &= -\partial_\alpha P + \left(\mu^0 + \mu^s\right)\partial_\alpha\partial_\gamma u_\gamma + \mu^s \partial_\beta\partial_\beta u_\alpha, \\
&= -\partial_\alpha P + \left(\mu^v + \frac{1}{3}\mu^s\right)\partial_\alpha\partial_\gamma u_\gamma + \mu^s \partial_\beta\partial_\beta u_\alpha.
\end{aligned}$$

The latter form is more frequently encountered although the bulk viscosity term is also frequently dropped.

The role of the volume viscosity coefficient μ^v, introduced in Eqn. 2.13, has been the subject of significant research and is not as well understood as shear viscosity. Stokes' original formula for sound attenuation of plane waves ignored bulk viscosity although he had some reservations about this assumption [243]. We have already shown that $\mu^v = 0$ for monatomic gases. For these, there is only one temperature, it is proportional to the translational energy and gives rise to the pressure. For gases with internal degrees of freedom, there is also one temperature at equilibrium when the equipartition theorem is true. However in situations away from equilibrium, such as at very short timescales, the temperature corresponding to internal energy may not match the temperature corresponding to translational energy, this manifests as the bulk viscosity. We note that even neglecting bulk viscosity, plane waves propagating in a multidimensional medium will experience attenuation since the deformation has both a volume expansion component and a shear component. For such a flow, the rate of volumetric expansion will be,

$$\epsilon^v_{\alpha\beta} = \frac{1}{3}\begin{pmatrix} \partial_x u_x & 0 & 0 \\ 0 & \partial_x u_x & 0 \\ 0 & 0 & \partial_x u_x \end{pmatrix}$$

and the rate of shear matrix will be,

$$\epsilon^s_{\alpha\beta} = \frac{1}{3}\begin{pmatrix} 2 \cdot \partial_x u_x & 0 & 0 \\ 0 & -\partial_x u_x & 0 \\ 0 & 0 & -\partial_x u_x \end{pmatrix}$$

so even in the absence of a bulk viscosity, the shear viscosity will create some dissipation.

The ratios of bulk to shear viscosity range from zero to the order of one for air to several thousand for CO_2 with reference values appearing in papers [243, 120, 60].

Early experiments involving a supersonic shock shows good experimental agreement using a value of $\mu^v = 0$ for Argon and $\mu^v = 2/3\mu^s$ for air [216]. The volume viscosity coefficient in general depends on temperature and density. For high frequency acoustic perturbations, the assumption of local thermodynamic equilibrium (LTE) may no longer be valid. As the time scales involved approach those of the underlying relaxation to thermodynamic equilibrium, the coefficient begins to depend on frequency [91, 243].

Measurement of bulk viscosity has been difficult due to experimental uncertainty in the attribution of measured dissipation to various mechanisms including scattering [243] and heat conduction [120, 91]. Results may also be sensitive to impurities, experiments with air are particularly sensitive to humidity [120, 91]. Graves and Argrow [91] also mention alternative methods proposed to measure bulk viscosity by various authors. Cramer [60] has recently estimated bulk viscosities for a range of fluids from a theoretical basis. Single relaxation time (BGK) solutions to the Boltzmann equation give a value of $\mu^0 = 0$ or $\mu^v = 2/3\mu^s$ which is not generally correct although it does come close for some gases such as air.

The pressure term is related to density and temperature by an equation of state,

$$P(\rho, t)$$

For an ideal gas the familiar equation of state is,

$$P = \rho R T$$

where R is the specific gas constant calculated using the molar mass M,

$$R = \frac{R_\mathrm{u}}{M}.$$

The temperature is then in turn a function of the thermal energy,

$$\begin{aligned} T &= \frac{2e^{\text{th}}}{\zeta R}, \\ e^{\text{th}} &= e - \frac{1}{2}u_\alpha u_\alpha \end{aligned}$$

Where ζ is the number of degrees of freedom of each molecule.

The final piece of the puzzle required to arrive at the Navier-Stokes equations is the heat flux tensor which is proportional to the conductivity and the temperature gradient.

$$q_\alpha = -\kappa \partial_\alpha T$$

Here κ is the thermal conductivity.

The full Navier-Stokes equations for an ideal gas are then,

$$\boxed{\begin{gathered} \partial_t \rho + \partial_\alpha \rho u_\alpha = 0 \\ \partial_t \rho u_\alpha + \partial_\beta \rho u_\alpha u_\beta = \partial_\beta \sigma_{\alpha\beta} + F_\alpha \\ \partial_t \rho e + \partial_\alpha u_\alpha \rho e = \kappa \partial_\alpha \partial_\alpha T + \partial_\alpha u_\beta \sigma_{\alpha\beta} + u_\alpha F_\alpha \\ \sigma_{\alpha\beta} = -\delta_{\alpha\beta} P + \left(\mu^v - \frac{2}{3}\mu^s\right) \delta_{\alpha\beta} \partial_\gamma u_\gamma + \mu^s \left(\partial_\alpha u_\beta + \partial_\beta u_\alpha\right) \\ P = \rho R T \\ T = \frac{2e - u_\alpha u_\alpha}{\zeta R} \end{gathered}} \tag{2.14}$$

The momentum equation may be written as,

$$\partial_t \rho u_\alpha + \partial_\beta \rho u_\alpha u_\beta = -\partial_\alpha P + \left(\mu^0 + \mu^s\right) \partial_\alpha \partial_\beta u_\beta + \mu^s \partial_\beta \partial_\beta u_\alpha + F_\alpha$$

Other equations of state can be used and several simplifications exist such as incompressible flow which will be discussed in a subsequent section.

2.7 Isothermal flow

If temperature variations are considered negligible, the density and momentum equations can be solved independently of the energy equation. The pressure is then purely a function of density. This is a highly non-physical assumption. Even when density variations are small, the local rise in temperature due to adiabatic heating causes a significant increase in pressure. In fact, Newton famously mis-predicted the speed of sound by assuming that the compression waves were athermal rather than adiabatic. Despite the assumption being invalid, it is used by many of the most common lattice Boltzmann schemes, including many of those that we will use. The errors introduced by the athermal assumption become negligible as the effects of density perturbations become negligible in the incompressible limit.

The speed of sound [31] is given by,

$$c_s^2 = \left(\frac{\partial P}{\partial \rho}\right)_s$$

The derivative must be taken adiabatically because the time scales of acoustic perturbations are not long enough for any heat transfer to occur. Using the cyclic chain rule,

$$\left(\frac{\partial x}{\partial y}\right)_z \left(\frac{\partial y}{\partial z}\right)_x \left(\frac{\partial z}{\partial x}\right)_y = -1,$$

we can express the adiabatic derivative in terms of the athermal derivative,

$$c_s^2 = \left(\frac{\partial P}{\partial \rho}\right)_s = -\frac{\left(\frac{\partial s}{\partial \rho}\right)_P}{\left(\frac{\partial s}{\partial P}\right)_\rho} = -\frac{\left(\frac{\partial s}{\partial T}\right)_P \left(\frac{\partial T}{\partial \rho}\right)_P}{\left(\frac{\partial s}{\partial T}\right)_\rho \left(\frac{\partial T}{\partial P}\right)_\rho} = \frac{c_p}{c_v}\left(\frac{\partial P}{\partial \rho}\right)_T.$$

The ratio of specific heats that appears here is also called the adiabatic index,

$$\gamma = \frac{c_p}{c_v}.$$

Using the ideal gas equation of state, the speed of sound is,

$$c_s^2 = \gamma RT.$$

Even for small perturbations, the coupling between energy and momentum is crucial.

If the flow is forced to be athermal, then the adiabatic index disappears from the speed of sound and the pressure may be written as,

$$P = c_s^2 \rho.$$

Many common lattice Boltzmann methods and artificial compressibility methods simulate this form of the Navier-Stokes equations. They cannot accurately simulate flows where compressible effects are important but the errors become insignificant in the incompressible limit so they can be used as a means of simulating incompressible flows.

2.8 Incompressible flow

If density variations in the fluid are small enough, density can be considered as a constant whenever it appears in the Navier-Stokes equations other than in the equation for pressure. When the speed of sound is large, small variations in density can cause a large change in pressure. For these flows, the pressure can be treated as another variable and the density as a constant. Assuming that the following derivatives commute,

$$\partial_\alpha \partial_t u_\alpha = \partial_t \partial_\alpha u_\alpha = 0$$
$$\partial_\alpha \partial_\beta \partial_\beta u_\alpha = \partial_\beta \partial_\beta \partial_\alpha u_\alpha = 0.$$

Taking the divergence of the momentum equation leads to the pressure Poisson equation (PPE).

$$\partial_\alpha \partial_\beta \rho u_\alpha u_\beta + \partial_\alpha \partial_\alpha P = \partial_\alpha F_\alpha. \tag{2.15}$$

The pressure is no longer a thermodynamic variable related to density and temperature, its value is determined entirely by the velocity field and the boundary conditions. In fact, if we define a projection operator that returns the divergence free component of a vector field, we can express the incompressible Navier-Stokes equations without referring to pressure at all.

Any vector field g_α can be decomposed into a divergence free, incompressible component which is the curl of some other vector field $\varepsilon_{\alpha\beta\gamma}\partial_\beta h_\gamma$ and a curl free, irrotational component which is the gradient of a scalar $\partial_\alpha \phi$. This is called the Helmholtz decomposition or the Hodge decomposition. There may also be a third component, called the Laplacian field that can also be written as the gradient of a scalar $\partial_\alpha \psi$ and is both divergence and curl free.

$$g_\alpha = \varepsilon_{\alpha\beta\gamma}\partial_\beta h_\gamma + \partial_\alpha \phi + \partial_\alpha \psi.$$

The Laplacian component can be calculated by setting its divergence to zero and solving the resulting Laplace equation,

$$\partial_\alpha \partial_\alpha \psi = 0.$$

Hence it does not depend on the original vector field at all, only on the boundary conditions. When the boundary conditions are zero, the Laplacian component is also zero so it is sometimes neglected in discussions with these sort of boundaries. The curl free component can be determined by taking the divergence of the vector field and solving the resulting Poisson equation for ϕ.

$$\partial_\alpha \partial_\alpha \phi = \partial_\alpha g_\alpha.$$

The divergence free component may then be calculated by subtraction from the original field. We can define a projection operator $\mathcal{P}(g_\alpha)$ which returns the incompressible component of a vector field,

$$\mathcal{P}(g_\alpha) = \varepsilon_{\alpha\beta\gamma}\partial_\beta h_\gamma = g_\alpha - \partial_\alpha\phi + \partial_\alpha\psi.$$

Assuming a divergence free velocity field, the momentum equations can be seen as a Helmholtz decomposition of this kind. The incompressible component is $\partial_t\rho u_\alpha$ and the irrotational component and Laplacian component combined are the pressure gradient $\partial_\alpha P$:

$$\partial_t\rho u_\alpha + \partial_\alpha P = \rho\nu\partial_\beta\partial_\beta u_\alpha - \rho u_\beta\partial_\beta u_\alpha.$$

In terms of the projection operator, the NS equations are now,

$$\partial_t\rho u_\alpha = \mathcal{P}(\rho\nu\partial_\beta\partial_\beta u_\alpha - \rho u_\beta\partial_\beta u_\alpha).$$

Pressure does not appear at all.

In order to solve the PPE, it is necessary to supply boundary condition information for pressure. This may appear to conflict with the notion that the pressure at the boundary is an unknown quantity to be calculated, however a Neumann type boundary condition can be arrived at by taking the normal component of the momentum equations at a boundary.

$$\hat{n}_\alpha\partial_\alpha P = -\partial_t\rho\hat{n}_\alpha u_\alpha + \rho\nu\partial_\beta\partial_\beta\hat{n}_\alpha u_\alpha - \rho u_\beta\partial_\beta\hat{n}_\alpha u_\alpha.$$

At a no-slip wall where velocity is zero, this gives,

$$\hat{n}_\alpha\partial_\alpha P = \rho\nu\partial_\beta\partial_\beta\hat{n}_\alpha u_\alpha.$$

In other words, for a wall normal to the x direction,

$$\partial_x P = \rho\nu\partial_x\partial_x u_x.$$

The solution of Laplace's equation with this boundary condition can be thought of as the Laplacian component of the pressure, while the solution of the PPE with a zero gradient boundary condition gives the irrotational component. Neglecting the Laplacian component of pressure is a common approximation that give acceptable results under some circumstances.

Incompressible fluid dynamics only depends on the gradient of the pressure, the addition of any constant does not produce any effect on the flow field. This gives us confidence that a Neumann boundary condition, where the normal gradient at the wall is specified, is appropriate since any constant added to give a Dirichlet boundary condition would be inconsequential. Gresho and Sani [94] provide a thorough discussion about pressure boundary conditions and also show that if the pressure boundary condition is derived from the momentum equation as shown, then the solvability of the Poisson equation is guaranteed.

2.9 Regions of applicability

Dense gases and liquids, those where the mean space between molecules is of the same order as the molecular diameter can be treated by a continuum approach as soon as the characteristic scales exceed the molecular length scale by a sufficient amount. Bird [16] gives the standard deviation of density for a system of n particles as $\sqrt{n^{-1}}$ which gives around 3% for a system of 1000 particles or 0.1% for a system of 1×10^6 particles which he selects to represent the cutoff between statistical fluctuations being significant or not.

The molar density depends only on temperature and pressure,

$$\hat{n} = \frac{P}{R_{\mathrm{u}}T} \quad \mathrm{mol\,m}^3.$$

At atmospheric pressure and a temperature of zero degrees Celsius, this gives us the standard number density for an ideal gas,

$$\hat{n}_0 = 44.61504\,\mathrm{mol\,m}^{-3} \quad \text{or} \quad n_0 = 2.686781 \times 10^{25}\ \mathrm{m}^{-3}. \tag{2.16}$$

One million ideal gas molecules, enough to neglect statistical fluctuations will be contained in a cube with edge length 3.3×10^{-7} m.

Dilute gas flows are characterised by a large average distance between molecules δ compared to the molecular diameter d. Bird [16] uses a value of $\delta/d = 7$ as a rough boundary between dense and dilute gases. Collisions in dilute gases are mostly binary, due to the large separation between molecules and the short range of interaction. It is unlikely to find three or more molecules in the same place at the same time. Another assumption that can be made is that of molecular chaos where the one particle distribution functions are uncorrelated. In this case the two particle distribution function is simply the product of two one particle distribution functions. Dilute gases under these assumptions can be approximated by the Boltzmann equation.

Dilute gases exhibit a unique feature, the mean free path is significantly larger than the molecular diameter giving rise to a regime where the characteristic scales are large enough to remove statistical fluctuations but too small in relation to the mean free path to allow the gas to reach thermal equilibrium. It is often said that the continuum hypothesis breaks down for these flows however this is a bit of a misnomer since the fluid still obeys the continuous hydrodynamic conservation equations we have described. In these situations the equilibrium assumption breaks down. The continuum hypothesis may still be valid in a sense that the length scales of interest are large enough that the problem may be described in terms of continuous variables as indeed they are in the Boltzmann equation. The assumptions that breakdown are the equation of state, stress-strain relationship and the heat flux vector that allow the closed form of the Navier-Stokes equations to be derived from the conservation equations. The Burnett and hyper-Burnett equations may be a better approximation for such flows.

The Knudsen number,

$$\mathrm{Kn} = \frac{\lambda}{L}.$$

gives the relationship between the mean free path λ and the characteristic size of flow L. The Boltzmann equation can be analysed using Chapman-Enskog (CE) theory which assumes that the gas is close to equilibrium and that particle populations can be expressed as a small perturbation from this equilibrium. The size of the perturbation in this analysis is proportional to the Knudsen number. Based on CE theory, it is possible to predict the equation of state, the stress tensor and the heat flux vector including the transport coefficients, the viscosity and thermal conductivity. This link between the Boltzmann equation and the Navier-Stokes equations breaks down as the Knudsen number increases. The Knudsen number is thus a good indicator of the breakdown of the equilibrium assumption. Bird [16] gives a value of $\mathrm{Kn} < 0.1$ for the continuum assumption to be valid. High Knudsen number flows may be better simulated by including more terms in the CE expansion leading to the Burnett and hyper-Burnett hydrodynamic equations [85].

This CE breakdown is not the only sort that can occur. Even for Euler type flows where the viscosity is not important, at high enough Knudsen numbers, the pressure tensor becomes anisotropic. The amount of thermal energy in each direction varies and the form of the stress tensor breaks down. Bird describes the breakdown of equilibrium for high Mach number expansions [14] based on results obtained using DSMC simulations. The radial temperature of the expanding flow diverges from the average temperature at a particular Knudsen number. A summary of other methods used to predict the breakdown of the Navier-Stokes equations is given by Macrossan [149].

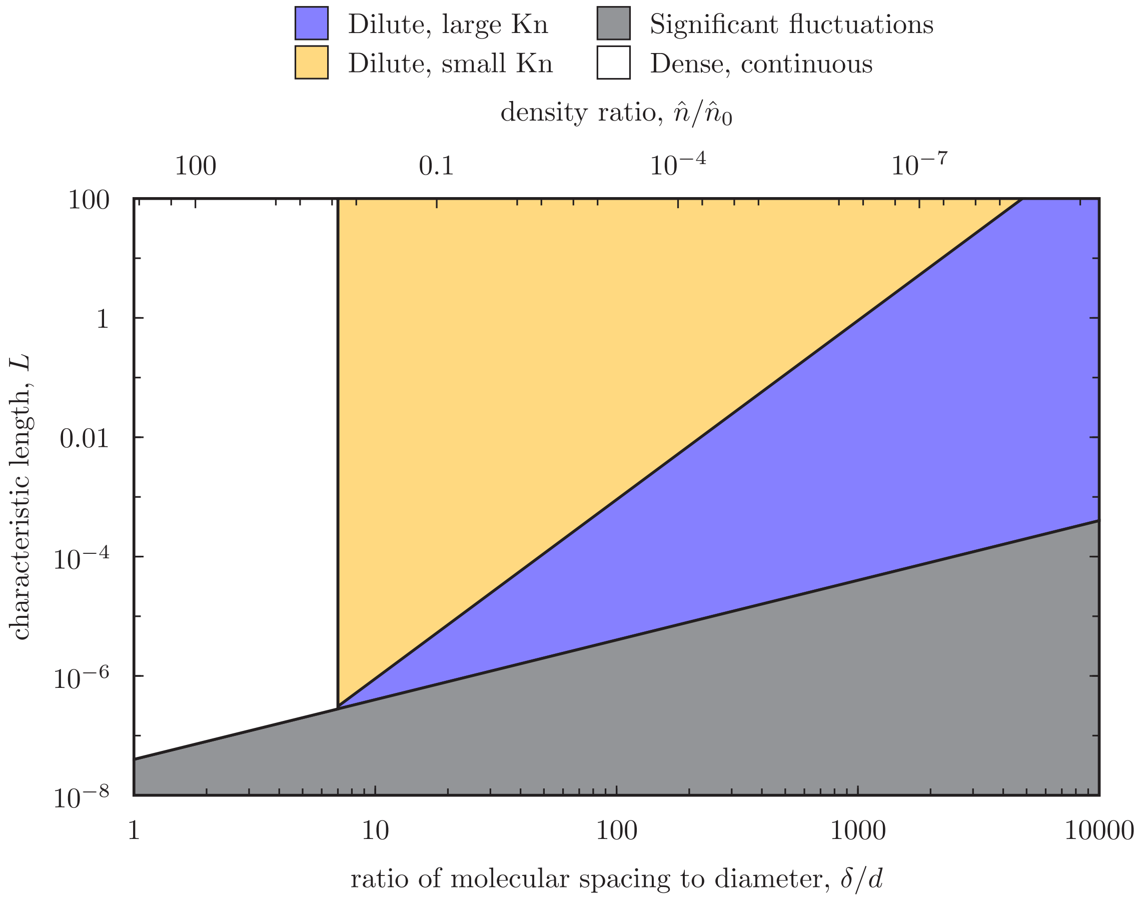


Figure 2.3: Regions of validity for various approximations, based on Bird [16].

Higher Knudsen numbers can be encountered in rarefied gas flows where the mean free path is larger or in micro fluid flows where the characteristic lengths are smaller. Another situation where high Knudsen number flows may be encountered is in heat pipes [27]. At some stages of operation the vapor pressure of the gas and hence the density is very low, combined with the small scale of these devices the Knudsen number can be quite high. Deviations from thermal equilibrium can also occur in other situations. As mentioned, bulk viscosity is a non-equilibrium phenomenon. Rapid compressions, expansions and shocks are other examples of flows that can deviate from equilibrium.

These various regions are plotted in Figure 2.3. The Navier-Stokes approach can be used in the dense region as well as the dilute region at a small Knudsen number. The Boltzmann approach may be used in the dilute region at a high or low Knudsen number although as we have seen, the single relaxation time approximations of the Boltzmann collision operator will limit it to small Kn only. Even though the Boltzmann method is not valid for dense gases or liquids, for a sufficiently small Knudsen number the scheme approaches the Navier-Stokes equations so in practice it may be used despite the breakdown of its assumptions. In that case the transport parameters cannot be predicted based on kinetic theory but an equivalent gas with matching parameters can be substituted. In the region where statistical fluctuations are significant, neither the Boltzmann or Navier-Stokes approaches are valid, molecular dynamics simulations are one option that may be suitable in this region.

In Chapter 5 of this thesis, I use the athermal lattice Boltzmann method to simulate incompressible fluid dynamics such as those encountered in liquids, quite far away from the types of flows intended to be modelled by the Boltzmann equation. This hydrodynamic solver is coupled to a finite volume scheme that solves an advection diffusion equation for a scalar. Under the assumptions that flow work and heating due to viscous dissipation are negligible and that the kinetic energy of the flow is also negligible, the energy equation reduces to a simple advection diffusion equation for a scalar. Under the Boussinesq approximation, the changes in density due to temperature only affect the fluid via the force due to buoyancy. The scalar buoyancy field can also be used to represent buoyant effects due to changes in concentration of some mixture such as saline water. Details about the lattice Boltzmann and finite volume schemes developed for this thesis are presented in the chapters that follow.

Chapter 3

Lattice Boltzmann methods

The lattice Boltzmann (LB) method is based on the Boltzmann equation 2.11 which captures a wide range of dilute gas dynamics flows including compressible and rarefied gases and can also be used to approximate incompressible fluid dynamics for other continuous substances such as liquids. A series of assumptions is made in order to reduce the Boltzmann equation (BE) to a more tractable form reducing the range of applicability in the process. The majority of LB methods use a linearised collision operator that replaces the collision integral by a relaxation towards local equilibrium. This limits the simulations to a small Knudsen number but allows a radical reduction in the number of velocities required to represent phase space. Methods that seek to solve the full collision integral are referred to as discrete velocity Boltzmann equation methods. These need to be able to represent all pre and post collision particle velocities in phase space using tens or hundreds of velocity samples in each direction, the resulting storage requirement of thousands or millions of samples per grid point severely limits the feasibility for three dimensional simulations. Instead of accurately representing the entire velocity phase space, the linearised collision operator only depends upon the first four moments of the equilibrium distribution so it is sufficient to ensure that these can be independently represented by the velocity basis chosen. Many of the commonly used lattices take the simplification one step further and neglect the fourth order moments which only appear in the energy equation, opting only to simulate the athermal Navier-Stokes equations, a non-physical approximation which is mostly used as a stepping stone allowing the simulation of incompressible flows in the small Mach number limit.

The final simplification that results in the lattice Boltzmann method is made by only selecting velocities that are integer multiples of the lattice size so that the advection of particles can be replaced by simple streaming from one lattice site to another. The collision term is replaced by a first order upwind approximation, the influence of the collision operator is calculated at the start of the time step then the post collision PDFs stream to the next nodes in their path. This is sometimes referred to as a collide and stream process and greatly simplifies the numerical implementation.

$$f_{\boldsymbol{i}}^{x+,t+} = f_{\boldsymbol{i}}^{x,t} + A_{\boldsymbol{ij}} \left(f_{\boldsymbol{j}}^{eq} - f_{\boldsymbol{j}}^{x,t} \right)$$

Numerical errors introduced by first order upwind advection are reduced to zero in the exact Lagrangian streaming case with a Courant number of one. In this Chapter we also analyse the case of non exact streaming, when the Courant number is less than one we find that the accuracy remains second order since diffusive errors can be absorbed into a redefined viscosity. Likewise the first order errors introduced by the upwind collision operator are also absorbed by a redefined viscosity as shown in this Chapter.

Calculation of LB methods can be further simplified by replacing the collision matrix with a single relaxation time τ multiplied by the identity matrix,

$$f_{\boldsymbol{i}}^{x+,t+} = f_{\boldsymbol{i}}^{x,t} + \frac{1}{\tau} \left(f_{\boldsymbol{i}}^{eq} - f_{\boldsymbol{i}}^{x,t} \right).$$

This is often referred to as the single relaxation time (SRT) method or the Bhatnager Gross Krook (BGK) [13] method.

The hydrodynamic properties of these lattice Boltzmann methods depend heavily on the choice of velocity basis vectors. The most common nomenclature for these velocity sets is DQ notation where D is the number of physical dimensions and Q is the number of velocities, for example the two dimensional set of 9 velocities is referred to as D2Q9. In Section 3.5 we will cover a more specific extension to this naming system that we have devised.

3.1 The discrete velocity Boltzmann equation with linearized collision operator

The discrete velocity Boltzmann equation only considers a discrete set of velocities $f_i(x_\alpha, t)$ rather than a continuous phase space significantly reducing the dimensionality of the equations by exchanging three velocity dimensions for a set of q discrete velocities,

$$\frac{\partial f_i}{\partial t} + c_{i\alpha}\frac{\partial f_i}{\partial x_\alpha} = \mathcal{C}_i + \mathcal{F}_i.$$

The advection term is similar to the continuous BE, however the force term must now be represented as a discrete operator $\mathcal{F}_i$ since it is no longer possible to accelerate particles in response to the applied force. We have seen (Chap. 2, Sec. 2.3) that the moments of the force operator in discrete space can be calculated by assuming that they are equal to the moments of the force term in continuous space,

$$\int q(c_\alpha)\frac{F_\alpha}{\rho}\frac{\partial f}{\partial c_\alpha}dc_\alpha = \sum_{i=0}^{q} q_i\mathcal{F}_i,$$

resulting in,

$$m \sum_{\boldsymbol{i}=0}^{q} \mathcal{F}_{\boldsymbol{i}} = 0, \tag{3.1}$$

$$m \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} \mathcal{F}_{\boldsymbol{i}} = F_{\alpha}, \tag{3.2}$$

$$m \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} \mathcal{F}_{\boldsymbol{i}} = u_{\alpha} F_{\beta} + u_{\beta} F_{\alpha}, \tag{3.3}$$

$$m \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\beta} \mathcal{F}_{\boldsymbol{i}} = 2eF_{\alpha} + 2RTF_{\alpha} + 2u_{\alpha} u_{\beta} F_{\beta}. \tag{3.4}$$

Another way to arrive at the same correct values for these moments is to observe the terms appearing in the conservation equations obtained by multi-scale expansion and make the moments equal to values which reproduce the correct hydrodynamic behaviour. Guo et al. have used the latter approach in their paper [98, 41]. There has been a wealth of research published about the force term in the lattice Boltzmann equation. Most of these papers have correctly identified the above moments of the force operator however few have realised additional errors are introduced by the discretization or the simple solution which we will discuss in section 3.4.3.

This is the first appearance of a common pattern, that LB methods can be analysed either from a top down perspective where the Boltzmann equation is assumed to be correct and the implementation is designed to approach it, or from a bottom up perspective where a multi-scale expansion is applied to the scheme and various parameters or additional terms are tweaked to give the desired behaviour in the hydrodynamic limit. The former approach naturally appeals because it is more rigorous however the latter approach can quickly yield practical results despite the shortcoming. For example, the equilibrium distribution can be modified so that Poisson or advection-diffusion equations are approximated at a particular order in the multi-scale expansion (3.9), this may involve some hand waving in order to justify the use of the multi-scale expansion.

A method for calculating a discrete approximation to the equilibrium distribution is required. One necessary condition on the discrete distribution is that its moments must match the moments of the continuous distribution however it is not immediately clear weather this is a sufficient condition to guarantee the convergence of the scheme towards a solution of the Boltzmann equation.

3.2 Equilibrium distribution

The steady state Boltzmann equation may be solved in order to determine the velocity distribution of a dilute gas at equilibrium, this is the famous Maxwell distribution,

$$f^{\text{eq}}(c'_\alpha) = \rho \left(\frac{1}{2\pi RT} \right)^{\frac{3}{2}} \exp\left(-\frac{|c'_\alpha|^2}{2RT} \right).$$

Moments of the Maxwell distribution may be calculated by first considering some standard Gaussian integrals. The one dimensional case may be calculated as the square root of the two dimensional Gaussian integral using a transformation to polar co-ordinates to calculate the latter,

$$\begin{aligned} \mathrm{I} &= \int_{-\infty}^{\infty} e^{-\lambda x^2} \, dx, \\ \mathrm{I}^2 &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-\lambda\left(x^2+y^2\right)} \, dxdy, \\ &= \int_0^{2\pi} \int_0^{\infty} e^{-\lambda r^2} r \, drd\theta. \end{aligned}$$

Making the substitution,

$$u = \lambda r^2, \quad du = 2\lambda r \, dr$$

leads to the result,

$$\begin{aligned} \mathrm{I}^2 &= \frac{1}{2\lambda} \int_0^{2\pi} \int_0^{\infty} e^{-u} \, dud\theta, \\ &= \frac{\pi}{\lambda} \int_0^{\infty} e^{-u} \, du, \\ &= \frac{-\pi}{\lambda} \left[e^{-u} \right]_0^{\infty} = \frac{\pi}{\lambda}. \end{aligned}$$

Thus the multi-dimensional Gaussian integrals are,

$$\begin{aligned}
\mathrm{I}^1 &= \int_{\mathbb{R}} e^{-\lambda x^2}\,dx = \left(\frac{\pi}{\lambda}\right)^{\frac{1}{2}}, \\
\mathrm{I}^2 &= \int_{\mathbb{R}^2} e^{-\lambda\left(x^2+y^2\right)}\,dxdy = \left(\frac{\pi}{\lambda}\right), \\
\mathrm{I}^3 &= \int_{\mathbb{R}^3} e^{-\lambda\left(x^2+y^2+z^2\right)}\,dxdydz = \left(\frac{\pi}{\lambda}\right)^{\frac{3}{2}}.
\end{aligned}$$

Odd moments of the Gaussian are zero since the product of an odd function and an even function is odd and a symmetric integral of an odd function is zero. The even moments may be calculated using integration by parts,

$$\begin{aligned}
\mathrm{H}_n = \int_{-\infty}^{\infty} x^{2n} e^{-\lambda x^2}\,dx &= \left[\frac{x^{2n+1}}{2n+1} e^{-\lambda x^2}\right]_{-\infty}^{\infty} - \int_{-\infty}^{\infty} \frac{x^{2n+1}}{2n+1}(-2\lambda x) e^{-\lambda x^2}\,dx \\
&= \frac{2\lambda}{2n+1} \int_{-\infty}^{\infty} x^{2(n+1)} e^{-\lambda x^2}\,dx = \frac{\lambda}{n+\frac{1}{2}} \mathrm{H}_{n+1}.
\end{aligned}$$

Using the fact that $\mathrm{H}_0 = \mathrm{I}^1$,

$$\begin{aligned}
\mathrm{H}_n &= \left(\frac{\pi}{\lambda}\right)^{\frac{1}{2}} \frac{1}{\lambda^n} \left(n - \frac{1}{2}\right)\left(n - \frac{3}{2}\right) \dots \left(\frac{3}{2}\right)\left(\frac{1}{2}\right), \\
&= \frac{\pi^{\frac{1}{2}}}{\lambda^{n+\frac{1}{2}} 2^n} (2n-1)(2n-3)\dots(3)(1), \\
&= \frac{\pi^{\frac{1}{2}}}{\lambda^{n+\frac{1}{2}} 2^n} \cdot \frac{(2n)(2n-1)(2n-2)\dots(2)(1)}{(2n)(2n-2)\dots(4)(2)}, \\
&= \frac{\pi^{\frac{1}{2}}}{\lambda^{n+\frac{1}{2}} 4^n} \cdot \frac{(2n)(2n-1)(2n-2)\dots(2)(1)}{(n)(n-1)\dots(2)(1)}, \\
&= \frac{\pi^{\frac{1}{2}}}{\lambda^{n+\frac{1}{2}} 4^n} \cdot \frac{(2n)!}{n!}.
\end{aligned}$$

The first three values are,

$$\begin{aligned}
\mathrm{H}_0 &= \left(\frac{\pi}{\lambda}\right)^{\frac{1}{2}}, \\
\mathrm{H}_1 &= \frac{1}{2\lambda}\left(\frac{\pi}{\lambda}\right)^{\frac{1}{2}}, \\
\mathrm{H}_2 &= \frac{3}{4\lambda^2}\left(\frac{\pi}{\lambda}\right)^{\frac{1}{2}}.
\end{aligned}$$

These can be used to calculate the fluctuating moments of the Maxwellian distribution which are required in the analysis of Boltzmann methods.

$$f^{\mathrm{eq}}(c'_\alpha) = \rho \left(\frac{\lambda}{\pi}\right)^{\frac{3}{2}} \exp\left(-\lambda |c'_\alpha|^2\right)$$

Using $\lambda = \frac{1}{2RT}$,

$$\begin{aligned}
\int_{\mathbb{R}^3} f^{\mathrm{eq}}(c'_\alpha)\, dc'_\alpha &= \rho \left(\frac{\lambda}{\pi}\right)^{\frac{3}{2}} \mathrm{H}_0^3 = \rho,\\
\int_{\mathbb{R}^3} c'_\alpha f^{\mathrm{eq}}(c'_\beta)\, dc'_\beta &= 0,\\
\int_{\mathbb{R}^3} c'_\alpha c'_\beta f^{\mathrm{eq}}(c'_\gamma)\, dc'_\gamma &= \rho \left(\frac{\lambda}{\pi}\right)^{\frac{3}{2}} \mathrm{H}_1(\mathrm{H}_0)^2 \delta_{\alpha\beta} = RT\rho\delta_{\alpha\beta},\\
\int_{\mathbb{R}^3} c'_\alpha c'_\beta c'_\gamma f^{\mathrm{eq}}(c'_\delta)\, dc'_\delta &= 0,\\
\int_{\mathbb{R}^3} c'_\alpha c'_\beta c'_\gamma c'_\delta f^{\mathrm{eq}}(c'_\zeta)\, dc'_\zeta &= \rho \left(\frac{\lambda}{\pi}\right)^{\frac{3}{2}} (\mathrm{H}_1)^2 \mathrm{H}_0 (\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma})\\
&= (RT)^2 \rho (\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}).
\end{aligned}$$

For example, if $\alpha = \beta = x$ and $\gamma = \delta = y$ then,

$$\begin{aligned}
&\int_{\mathbb{R}^3} c'^2_x c'^2_y f^{\mathrm{eq}}(c'_\zeta)\, dc'_\zeta\\
&\quad = \rho \left(\frac{\lambda}{\pi}\right)^{\frac{3}{2}} \int_{\mathbb{R}} c'^2_x \exp\left(-\lambda c'^2_x\right) dc'_x \int_{\mathbb{R}} c'^2_y \exp\left(-\lambda c'^2_y\right) dc'_y \int_{\mathbb{R}} \exp\left(-\lambda c'^2_z\right) dc'_z\\
&\quad = (RT)^2 \rho.
\end{aligned}$$

Since the Maxwell distribution only depends on the fluctuating component of the particle velocity $c'_\alpha = c_\alpha - u_\alpha$ and noting that odd moments of the fluctuating velocity are equal to zero allows the velocity moments of the equilibrium distribution to be calculated. The discrete moments should correspond directly,

$$
\begin{aligned}
\sum_{\boldsymbol{i}=0}^{q} f_{\boldsymbol{i}}^{eq} &= \int_{\mathbb{R}^3} f^{\mathrm{eq}}(c_\alpha')\, dc_\alpha = \rho, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} f_{\boldsymbol{i}}^{eq} &= \int_{\mathbb{R}^3} c_\alpha f^{\mathrm{eq}}(c_\beta')\, dc_\beta = \int_{\mathbb{R}^3} (c_\alpha' + u_\alpha) f^{\mathrm{eq}}(c_\beta')\, dc_\beta' = \rho u_\alpha, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} &= \int_{\mathbb{R}^3} c_\alpha c_\beta f^{\mathrm{eq}}(c_\gamma')\, dc_\gamma = \int_{\mathbb{R}^3} (c_\alpha' + u_\alpha)(c_\beta' + u_\beta) f^{\mathrm{eq}}(c_\gamma')\, dc_\gamma', \\
&= \int_{\mathbb{R}^3} (c_\alpha' c_\beta' + u_\alpha u_\beta) f^{\mathrm{eq}}(c_\gamma')\, dc_\gamma, \\
&= RT\rho\delta_{\alpha\beta} + \rho u_\alpha u_\beta, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f_{\boldsymbol{i}}^{eq} &= \int_{\mathbb{R}^3} c_\alpha c_\beta c_\gamma f^{\mathrm{eq}}(c_\delta')\, dc_\delta, \\
&= \int_{\mathbb{R}^3} (c_\alpha' + u_\alpha)(c_\beta' + u_\beta)(c_\gamma' + u_\gamma) f^{\mathrm{eq}}(c_\delta')\, dc_\delta', \\
&= \int_{\mathbb{R}^3} (c_\alpha' c_\beta' + c_\alpha' u_\beta + c_\beta' u_\alpha + u_\alpha u_\beta)(c_\gamma' + u_\gamma) f^{\mathrm{eq}}(c_\delta')\, dc_\delta', \\
&= (\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}) RT\rho u_\delta + \rho u_\alpha u_\beta u_\gamma, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} c_{\boldsymbol{i}\delta} f_{\boldsymbol{i}}^{eq} &= \int_{\mathbb{R}^3} c_\alpha c_\beta c_\gamma c_\delta f^{\mathrm{eq}}(c_\zeta')\, dc_\zeta, \\
&= \int_{\mathbb{R}^3} (c_\alpha' + u_\alpha)(c_\beta' + u_\beta)(c_\gamma' + u_\gamma)(c_\delta' + u_\delta) f^{\mathrm{eq}}(c_\zeta')\, dc_\zeta', \\
&= \int_{\mathbb{R}^3} (u_\alpha u_\beta + u_\beta c_\alpha' + u_\alpha c_\beta' + c_\alpha' c_\beta') \\
&\qquad (u_\gamma u_\delta + u_\delta c_\gamma' + u_\gamma c_\delta' + c_\gamma' c_\delta') f^{\mathrm{eq}}(c_\zeta')\, dc_\zeta', \\
&= \int_{\mathbb{R}^3} (u_\alpha u_\beta u_\gamma u_\delta + u_\alpha u_\beta c_\gamma' c_\delta' + u_\beta u_\delta c_\alpha' c_\gamma' + u_\beta u_\gamma c_\alpha' c_\delta' \\
&\qquad + u_\alpha u_\delta c_\beta' c_\gamma' + u_\alpha u_\gamma c_\beta' c_\delta' + u_\gamma u_\delta c_\alpha' c_\beta' + c_\alpha' c_\beta' c_\gamma' c_\delta') f^{\mathrm{eq}}(c_\zeta')\, dc_\zeta', \\
&= \rho u_\alpha u_\beta u_\gamma u_\delta + \rho RT(u_\alpha u_\beta \delta_{\gamma\delta} + u_\beta u_\delta \delta_{\alpha\gamma} + u_\beta u_\gamma \delta_{\alpha\delta} + u_\alpha u_\delta \delta_{\beta\gamma} \\
&\qquad + u_\alpha u_\gamma \delta_{\beta\delta} + u_\gamma u_\delta \delta_{\alpha\beta}) + (RT)^2(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma})\rho.
\end{aligned}
$$

The full fourth order moment does not appear in the multi-scale expansion, only the contracted fourth order moment is required,

$$\begin{aligned}\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\gamma}f_{\boldsymbol{i}}^{eq} &= \int_{\mathbb{R}^3} c_\alpha c_\beta c_\beta c_\gamma f^{\mathrm{eq}}(c'_\delta)\, dc_\delta, \\ &= \rho u_\alpha u_\beta u_\beta u_\gamma + \rho RT(u_\alpha u_\gamma + u_\beta u_\beta \delta_{\alpha\gamma} + u_\alpha u_\gamma + u_\alpha u_\gamma \\ &\quad + 3u_\alpha u_\gamma + u_\alpha u_\gamma) + (RT)^2(\delta_{\alpha\gamma} + 3\delta_{\alpha\gamma} + \delta_{\alpha\gamma})\rho, \\ &= \rho u_\alpha u_\beta u_\beta u_\gamma + \rho RT(7u_\alpha u_\gamma + u_\beta u_\beta \delta_{\alpha\gamma}) + 5(RT)^2\delta_{\alpha\gamma}\rho, \\ &= 2\rho e u_\alpha u_\gamma + 4RT\rho u_\alpha u_\gamma + 2RT\delta_{\alpha\gamma}\rho e + 2(RT)^2\delta_{\alpha\gamma}\rho.\end{aligned}$$

Siebert et al. [220] have tested LB schemes which are only able to correctly represent the reduced fourth order moment and found no worse results compared to schemes which are able to capture the full fourth order moment.

Noting these constraints, it is possible to reformulate the method so that only the moments of $f_{\boldsymbol{i}}^{eq}$ are referred to. This approach is called the multiple relaxation time (MRT) method or the generalised lattice Boltzmann method. No further work is required to implement such a scheme.

The more common approach is to formulate an explicit expression for the equilibrium distribution, the equilibrium moments of which should match those above. It is common to make use of 'lattice tensors' in the process. These are formed by assigning weights $W_{\boldsymbol{i}}$ to the discrete velocities, velocities with the same magnitude are typically assigned the same weight. Any freedom in selecting weights is used to ensure that the lattice tensors are equal to the isotropic tensors below,

$$
\begin{aligned}
\sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} &= 1, \\
\sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} &= 0, \\
\sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} &= RT\delta_{\alpha\beta}, \\
\sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} &= 0, \\
\sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} c_{\boldsymbol{i}\delta} &= (RT)^2 \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}\right).
\end{aligned}
\tag{3.5}
$$

In general, a lattice with higher order isotropic lattice tensors is able to represent higher order equilibrium moments.

A common explicit form drops third order and higher velocity terms and uses a constant temperature. We use c_s^2 rather than RT in order to convey that these athermal methods use a constant speed of sound and temperature.

$$
f_{\boldsymbol{i}}^{eq} = \rho W_{\boldsymbol{i}} \left[1 + \frac{c_{\boldsymbol{i}\alpha} u_\alpha}{c_s^2} + \frac{\left(c_{\boldsymbol{i}\alpha} u_\alpha\right)^2}{2c_s^4} - \frac{u_\alpha u_\alpha}{2c_s^2} \right]
$$

Assuming that weights can be found to construct the required lattice tensors, then we can quickly verify that the equilibrium moments up to rank three are,

$$
\begin{aligned}
\sum_{\boldsymbol{i}=0}^{q} f_{\boldsymbol{i}}^{eq} &= \rho\left(1 + c_s^2\frac{u_\alpha u_\alpha}{2c_s^4} - \frac{u_\alpha u_\alpha}{2c_s^2}\right) = \rho, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} f_{\boldsymbol{i}}^{eq} &= \rho u_\alpha, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} &= \rho\left(c_s^2\delta_{\alpha\beta} + c_s^4\left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}\right)\cdot\frac{u_\gamma u_\delta}{2c_s^4}\right. \\
&\qquad \left. -c_s^2\delta_{\alpha\beta}\cdot\frac{u_\gamma u_\gamma}{2c_s^2}\right), \\
&= \rho\left(c_s^2\delta_{\alpha\beta} + \frac{1}{2}\delta_{\alpha\beta}u_\gamma u_\gamma + \frac{1}{2}u_\alpha u_\beta + \frac{1}{2}u_\alpha u_\beta - \frac{1}{2}\delta_{\alpha\beta}u_\gamma u_\gamma\right), \\
&= \rho\left(c_s^2\delta_{\alpha\beta} + u_\alpha u_\beta\right), \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f_{\boldsymbol{i}}^{eq} &= \rho c_s^2\left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}\right)u_\delta.
\end{aligned}
$$

Note that we expect errors proportional to the velocity cubed due to the incomplete representation of the third order moment. We have not considered the fourth order equilibrium moment since energy is not conserved using these athermal methods. Thermal methods require the rank 6 tensor to be isotropic in order to correctly recover the fourth equilibrium moment.

It is straight forward to see how the terms in the equilibrium distribution are chosen for the example above, the construction of more complicated distributions that recover higher order moments correctly requires a more generalized procedure. There is also the question of how well the discrete distribution approaches the continuous one. He and Luo [103] and Shan and He [212] have shown a general methodology for deriving equilibrium distributions including the one above. They use the isotropic lattice tensors (Eqn. 3.5) to form a set of discrete Hermite polynomial tensors which are orthogonal under the weights $W_{\boldsymbol{i}}$. The continuous equilibrium distribution is expressed as a series of Hermite polynomials, the coefficients are then calculated using a form of Gauss-Hermite quadrature which takes advantage of the orthogonality of the Hermite tensors.

Philippi et al. [178] go into considerably more detail on the topic. They consider the discrete Hilbert space in some detail proving that norm preservation under this space guarantees the orthogonality of the Hermite polynomials for Bravais lattices. Hence the ability to construct an equilibrium distribution is also guaranteed. The Gauss-Hermite quadrature already mentioned is not accurate when the temperature varies, Philippi et al. propose a method of quadrature based on prescribed abscissas in order to resolve this problem. A similar approach is also presented by Shan et al. [213].

This Hermite tensor series is expected to approach the continuous distribution as the order increases. The expression of the equilibrium distribution as a sum of Hermite polynomial tensors bears some similarity to the Grad method discussed in Section 2.4.3. The orthogonality of the Hermite polynomials is also a vital component of Grad's method. It is speculated that the properties of the discrete Hilbert space outlined by Philippi et al. might prove important to accuracy however we note the absence of tests demonstrating any benefit. We also note the existence of several correctly functioning schemes that do not preserve the orthogonality of the Hilbert space.

Hermite polynomial based methods only consider equilibrium distributions which can be expressed in terms of isotropic tensors. It is possible to find counter examples, lattices whose equilibrium distribution is not expressed in terms of isotropic tensors but whose equilibrium moments are correct, see Section 3.5. More generally, assuming that the lattice vectors corresponding to equilibrium moments up to a certain order are linearly independent, one is always able to construct a transformation matrix between the discrete equilibrium distribution and its moments which can be used to convert between one and the other. Such lattices may not necessarily possess an isotropic lattice tensor of the same order.

$$m_{\boldsymbol{i}}^{eq} = M_{\boldsymbol{ij}} f_{\boldsymbol{j}}^{eq}, \quad f_{\boldsymbol{j}}^{eq} = M_{\boldsymbol{ji}}^{-1} m_{\boldsymbol{i}}^{eq}$$

A concrete example is the D3Q13 lattice which does not have a symmetric lattice tensor of rank four but is nevertheless able to recover the second order equilibrium moment. D'Humieres, Bouzidi and Lallemand [62] introduced this lattice using a moment based method in 2001 however we are not aware of anyone who has shown an explicit formula for the equilibrium distribution as we have calculated in Section 3.A.3. In this case the rank four lattice tensor includes non-isotropic terms proportional to the rank four delta function $\delta_{\alpha\beta\gamma\delta}$. This delta function is one only if all four indices are equal and although it prevents an equilibrium distribution expressed purely in terms of Hermite tensors, it can be resolved by using a corresponding rank four delta term in the equilibrium distribution that will be presented below in Section 3.A.3. In other cases such as when a rectangular lattice is used, we have used the collision operator to correct the same sort of anisotropic errors. Another counter example is the D2Q13 lattice which Philippi et al. [178] claim cannot be used for third order methods. We will demonstrate a set of orthogonal vectors which prove the existence of an equilibrium distribution for this lattice 3.A.2. The rest of the work presented in this thesis centers around the MRT method which does not require the use of explicit equilibrium distributions.

3.3 Chapman-Enskog expansion

We consider the Chapman-Enskog expansion of a discrete velocity Boltzmann equation with SRT collision operator.

$$\partial_t f_i + c_{i\alpha}\partial_\alpha f_i = \frac{1}{\tau}\left(f_i^{eq} - f_i\right) + \mathcal{F}_i$$

The results from this section will be used as a foundation from which other results will be derived by considering additional terms in subsequent sections. This simple example contains fewer calculations and quickly illustrated key points. Summing the discrete BE over all basis particle directions gives,

$$\partial_t \rho + \partial_\alpha \rho u_\alpha = 0. \tag{3.6}$$

Thus the conservation of mass equation is easily recovered. Multiplying each term by $c_{i\alpha}$ and summing gives,

$$\partial_t \rho u_\alpha + \partial_\beta \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} f_i = \sum_{i=0}^{q} c_{i\alpha} \mathcal{F}_i = F_\alpha.$$

The second order moments are equal to the momentum flux tensor Π,

$$\Pi_{\alpha\beta} = \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} f_i.$$

These conservation equations do not depend on the collision operator as long as it conserves mass and momentum. In order to get an expression for the momentum flux tensor, using the methods of Chapman and Enskog, a multi-scale expansion in terms of a small parameter ϵ is performed. The small parameter in the Chapman-Enskog method is the Knudsen number,

$$f_i = f_i^{eq} + \epsilon f_i^{(1)} + O(\epsilon^2). \tag{3.7}$$

Derivatives are also redefined as,

$$\partial_t = \epsilon \partial_{t_1} + \epsilon^2 \partial_{t_2} \quad \text{and} \quad \partial_\alpha = \epsilon \partial_{\alpha_1}. \tag{3.8}$$

Now gathering $\mathcal{O}(\epsilon)$ terms in the BE gives an expression for the first order non-equilibrium term,

$$f_i^{(1)} = \tau \left(F_i - \partial_{t_1} f_i^{eq} - c_{i\alpha} \partial_{\alpha_1} f_i^{eq} \right). \tag{3.9}$$

The momentum equation at $\mathcal{O}(\epsilon)$ is,

$$\partial_{t_1}\rho u_\alpha + \partial_\beta \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} = F_\alpha. \tag{3.10}$$

Substituting the Maxwellian equilibrium distribution gives the equilibrium momentum flux tensor,

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} = \rho RT\delta_{\alpha\beta} + \rho u_\alpha u_\beta.$$

The momentum equation at $\mathcal{O}(\epsilon)$ recovers the non-linear advection and pressure terms in the NS equations,

$$\partial_{t_1}\rho u_\alpha + \partial_\beta \rho u_\alpha u_\beta = -RT\partial_\alpha \rho + F_\alpha.$$

This is also referred to as the acoustic time scale.

In order to determine the remainder of the stress tensor and the heat flux tensor, terms of order $\mathcal{O}(\epsilon^2)$ must be considered,

$$\partial_{t_2}\rho u_\alpha + \partial_\beta \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{(1)} = 0. \tag{3.11}$$

Substituting in the expression for first order non-equilibrium terms (Eqn. 3.9) gives,

$$\partial_{t_2}\rho u_\alpha = \tau\partial_\beta \left\{ \partial_\gamma \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f_{\boldsymbol{i}}^{eq} + \partial_{t_1} \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} - \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} F_{\boldsymbol{i}} \right\}. \tag{3.12}$$

The following identities are used,

$$\begin{aligned}
\partial_t abc &= bc\partial_t a + ac\partial_t b + ab\partial_t c,\\
&= bc\partial_t a + c\partial_t ab - bc\partial_t a + b\partial_t ac - bc\partial_t a,\\
&= -bc\partial_t a + c\partial_t ab + b\partial_t ac.\\
\partial_t abcd &= bc\partial_t ad + ad\partial_t bc,\\
&= -bc\partial_t ad + c\partial_t abd + b\partial_t acd.
\end{aligned}$$

Time derivatives are recast as space derivatives,

$$\begin{aligned}
\partial_t \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} &= \partial_t \left(RT\rho\delta_{\alpha\beta} + \rho u_\alpha u_\beta\right),\\
&= -RT\delta_{\alpha\beta}\partial_\gamma \rho u_\gamma + u_\alpha u_\beta \partial_\gamma \rho u_\gamma + u_\alpha \partial_t \rho u_\beta + u_\beta \partial_t \rho u_\alpha,\\
&= -RT\delta_{\alpha\beta}\partial_\gamma \rho u_\gamma + u_\alpha u_\beta \partial_\gamma \rho u_\gamma + u_\alpha \left(-RT\partial_\beta \rho - \partial_\gamma \rho u_\beta u_\gamma + F_\beta\right)\\
&\quad + u_\beta \left(-RT\partial_\alpha \rho - \partial_\gamma \rho u_\alpha u_\gamma + F_\alpha\right),\\
&= -RT\left(\delta_{\alpha\beta}\partial_\gamma \rho u_\gamma + u_\alpha \partial_\beta \rho + u_\beta \partial_\alpha \rho\right) - \partial_\gamma \rho u_\alpha u_\beta u_\gamma\\
&\quad + u_\alpha F_\beta + u_\beta F_\alpha.
\end{aligned}$$

The second moment of the force operator should take its previously calculated value,

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} F_{\boldsymbol{i}} = u_\alpha F_\beta + u_\beta F_\alpha.$$

The third order moments should match those of the Maxwellian distributions,

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f_{\boldsymbol{i}}^{eq} = RT\left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}\right)\rho u_\delta + \rho u_\alpha u_\beta u_\gamma.$$

These allow us to simplify the expression,

$$\begin{aligned}\partial_\beta\left\{\partial_\gamma\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\gamma}f_{\boldsymbol{i}}^{eq}+\partial_{t_1}\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}f_{\boldsymbol{i}}^{eq}-\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}F_{\boldsymbol{i}}\right\}&\\ =\partial_\beta\left\{RT\partial_\gamma\left(\delta_{\alpha\gamma}\delta_{\beta\delta}+\delta_{\alpha\delta}\delta_{\beta\gamma}\right)\rho u_\delta-RT\left(u_\alpha\partial_\beta\rho+u_\beta\partial_\alpha\rho\right)\right\},&\\ =RT\partial_\beta\left\{\partial_\alpha\rho u_\beta+\partial_\beta\rho u_\alpha-\left(u_\alpha\partial_\beta\rho+u_\beta\partial_\alpha\rho\right)\right\},&\\ =RT\partial_\beta\left(\rho\partial_\alpha u_\beta+\rho\partial_\beta u_\alpha\right).&\quad(3.13)\end{aligned}$$

Thus, the second order in ϵ terms recover the viscous terms in the Navier-Stokes equations,

$$\partial_{t_2}\rho u_\alpha=\tau RT\left(\partial_\beta\rho\partial_\alpha u_\beta+\partial_\beta\rho\partial_\beta u_\alpha\right),$$

with shear and bulk viscosities of

$$\nu^s=RT\tau\quad\text{and}\quad\nu^v=\frac{2}{3}\nu^s.$$

These terms are referred to as the diffusive time scale with the sum of both time scales resulting in the Navier-Stokes equations.

We assume that there is no internal energy and multiply by the total kinetic energy, $\frac{1}{2}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\alpha}$,

$$\partial_t\rho e+\frac{1}{2}\partial_\alpha\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}f_{\boldsymbol{i}}=\frac{1}{2}\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\alpha}\mathcal{F}_{\boldsymbol{i}}=u_\alpha F_\alpha.$$

At order ϵ we have,

$$\begin{gathered}\partial_{t_1}\rho e+\frac{1}{2}\partial_\alpha\sum_{\boldsymbol{i}=0}^{q}c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}f_{\boldsymbol{i}}^{eq}=u_\alpha F_\alpha,\\ \partial_{t_1}\rho e+\frac{1}{2}\partial_\alpha\delta_{\beta\gamma}RT\left(\delta_{\alpha\beta}\delta_{\gamma\delta}+\delta_{\alpha\gamma}\delta_{\beta\delta}+\delta_{\alpha\delta}\delta_{\beta\gamma}\right)\rho u_\delta+\frac{1}{2}\partial_\alpha\rho u_\alpha u_\beta u_\beta=u_\alpha F_\alpha,\\ \partial_{t_1}\rho e+\frac{5}{2}RT\partial_\alpha\rho u_\alpha+\frac{1}{2}\partial_\alpha\rho u_\alpha u_\beta u_\beta=u_\alpha F_\alpha.\end{gathered}$$

Since $e_{th}=\frac{3}{2}RT$,

$$\partial_{t_1}\rho e+\partial_\alpha u_\alpha\left(\rho e+P\right)=u_\alpha F_\alpha.$$

These terms represent the advection of energy by the flow and the flow work. At order ϵ^2,

$$\begin{aligned}\partial_{t_2}\rho e &= -\frac{1}{2}\partial_\alpha \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}f_{\boldsymbol{i}}^{(1)}, \qquad (3.14)\\ &= \tau\frac{1}{2}\partial_\alpha\left(\partial_{t_1}\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}f^{eq} + \partial_\gamma\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\gamma}f^{eq} - \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}F_{\boldsymbol{i}}\right).\end{aligned}$$

The mixed time and space derivative needs to be recast in terms of space derivatives again,

$$\begin{aligned}&\frac{1}{2}\partial_\alpha\partial_{t_1}\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\beta}f^{eq}\\ &\quad= \frac{1}{2}\partial_\alpha\partial_{t_1}\left(5RT\rho u_\alpha + \rho u_\alpha u_\beta u_\beta\right),\\ &\quad= \partial_\alpha\partial_{t_1}\rho u_\alpha e + RT\partial_\alpha\partial_{t_1}\rho u_\alpha,\\ &\quad= \partial_\alpha\left(-u_\alpha e\partial_{t_1}\rho + u_\alpha\partial_{t_1}\rho e + e\partial_{t_1}\rho u_\alpha - (RT)^2\partial_\alpha\rho + RTF_\alpha - RT\partial_\beta\rho u_\alpha u_\beta\right),\\ &\quad= \partial_\alpha(u_\alpha e\partial_\beta\rho u_\beta + u_\alpha u_\beta F_\beta - u_\alpha\partial_\beta u_\beta\rho e - RTu_\alpha\partial_\beta\rho u_\beta\\ &\qquad - eRT\partial_\alpha\rho + eF_\alpha - e\partial_\beta\rho u_\alpha u_\beta - (RT)^2\partial_\alpha\rho + RTF_\alpha - RT\partial_\beta\rho u_\alpha u_\beta),\\ &\quad= \partial_\alpha(-\partial_\beta\rho e u_\alpha u_\beta - RTu_\alpha\partial_\beta\rho u_\beta - RT\partial_\beta\rho u_\alpha u_\beta\\ &\qquad - eRT\partial_\alpha\rho - (RT)^2\partial_\alpha\rho + eF_\alpha + RTF_\alpha + u_\alpha u_\beta F_\beta).\end{aligned} \tag{3.15}$$

The fourth order Maxwellian provides a nice cancellation of terms in Eqn. 3.15 as does the expected form of the forcing term Eqn. 3.1. The remaining terms at $\mathcal{O}(\epsilon^2)$ are,

$$\partial_{t_2}\rho e = RT\tau(2\partial_\alpha\partial_\beta\rho u_\alpha u_\beta - \partial_\alpha u_\alpha\partial_\beta\rho u_\beta - \partial_\alpha\partial_\beta\rho u_\alpha u_\beta + \partial_\alpha\rho\partial_\alpha e).$$

Using the identity,

$$\frac{1}{2}\partial_\alpha\partial_\alpha u_\beta u_\beta = \partial_\alpha u_\beta\partial_\alpha u_\beta,$$

we arrive at,

$$\partial_{t_2}\rho e = RT\tau(-\partial_\alpha \rho u_\beta \partial_\beta u_\alpha - \partial_\alpha \rho u_\beta \partial_\alpha u_\beta - \rho \partial_\alpha \partial_\alpha e_{th}).$$

Combining first and second order in ϵ terms results in,

$$\partial_t \rho e + \partial_\alpha u_\alpha(\rho e + P) = RT\tau(-\partial_\alpha \rho u_\beta \partial_\beta u_\alpha - \partial_\alpha \rho u_\beta \partial_\alpha u_\beta - \rho \partial_\alpha \partial_\alpha e_{th}) + u_\alpha F_\alpha.$$

These are the continuum hydrodynamic equations for a monatomic gas with no internal energy and the following parameters,

$$\begin{aligned} \nu^s &= RT\tau, \\ \nu^v &= \frac{2}{3}\nu^s, \\ \kappa &= \frac{3}{2}(RT)^2\rho\tau, \\ T &= \frac{2}{3R}e_{th}. \end{aligned}$$

The specific heat for a monatomic gas gives a relationship for Prandtl number,

$$\begin{aligned} c_P &= \frac{5}{2}RT, \\ \alpha &= \frac{\kappa}{\rho c_P} = \frac{3}{5}RT\tau, \\ Pr &= \frac{\alpha}{\nu} = \frac{3}{5} = 0.6. \end{aligned}$$

3.3.1 Thermal methods

Lattice Boltzmann methods offer an attractive means for simulating thermal hydrodynamics due to their roots in the Boltzmann equation. Within the theoretical framework presented thus far, the main requirement for simulating thermal hydrodynamics is the ability to represent the equilibrium moments up to the contracted rank four tensor. Moments up to a third order are sufficient to simulate athermal hydrodynamics and as noted, some commonly used lattices lack the freedom to correctly set even the third order moments, making do with a partial representation of the third order moments. Higher order equilibrium distributions can also be useful in eliminating third order errors which cause problems when simulating multi-phase flows [180, 152].

We have shown in Section 3.10 that is is possible to correct for errors introduced by an incorrect equilibrium distribution by selecting suitable eigenvectors and eigenvalues for the collision operator. Another approach, used by Prasianakis and Karlin [186, 187], is to introduce corrections to the evolution equation based on the gradients of the hydrodynamic fields in order to correct errors introduced in the equilibrium moments.

A rigorous framework has been established for thermal lattice Boltzmann methods by Shan et al. [213] and Philippi et al. [178]. Within this framework, the equilibrium distribution must be expressed in terms of Hermite polynomial tensors, a condition that is more strict than simply being able to represent the equilibrium moments but which may not be necessary. A higher order polynomial basis is expected to improve the approximation of the continuous distribution. When these higher order terms take their Maxwellian values, the terms introduced approach some higher order hydrodynamics such as the Burnett equations or higher order Grad equations. These additional terms may still be considered errors compared to the Navier-Stokes (NS) equations and may be a source of instability. Alternatively, these moments may be set to minimise deviations from the NS equations or maximise stability. In any case, the ability to adjust higher order moments is advantageous.

We will list several lattices which have been reported over the years for use with thermal lattice Boltzmann methods. Most velocity sets will be referred to using the previously mentioned DQ notation. This notation is ambiguous with regards to exactly which velocities are used or the topology, there are infinitely many ways to pick a given number of directions in space, to avoid this ambiguity we will introduce a more precise terminology in Section 3.5. We have not converted all lattices mentioned in this section to our more precise terminology. Chen, Ohashi and Akiyama [40] reported in 1994 that a D2Q13 lattice on a square grid was sufficient to model the full thermal equations. Unfortunately the fourth order moments are not correctly represented by this lattice. Vahala, Vahala, Pavlo and Martys [246, 174] found that a D2Q13 model on a hexagonal grid, rather than a square grid was more isotropic but still could not fully recover the Maxwellian moments. They found that a D2Q17 grid using an octagonal basis was required. This lattice does not align with grid points and hence is one of the first so called "off lattice" schemes. A simple streaming operation from site to site is no longer possible and the advection of particles must be solved using a second order or higher scheme in order to prevent the introduction of spurious terms into the CE expansion. The advantage in using a velocity basis that does not tile space is that there is more freedom in choosing velocities. Some authors such as Shim and Gatignol [218] have used the D2Q13 lattice on a hexagonal grid as recently as 2011. Their shock tube simulations exhibit some oscillations.

McNamara et al. [153, 154] have shown that increasing the velocities to two units along the co-ordinate axes can allow the D3Q27 lattice to capture the reduced fourth order Maxwellian moments. We have discovered, as detailed in Section 3.A.6, that also extending the $(1, 1, 0)$ family of velocities to $(2, 2, 0)$ results in a lattice which tiles space using a truncated octahedral unit cell and also allows the reduced fourth order Maxwellian to be simulated.

Watari and Tsutahara [253] have determined a fourth order accurate D2Q29 and D3Q77 lattice. Shan et al. [213] present several lattices that recover the Maxwellian moments up to the third order for both on lattice D2Q17 and D3Q39 and off lattice D2Q12 and D3Q27 schemes. Philippi et al. [178] found on lattice two dimensional D2Q25 and D2Q37 lattices that recover the reduced and full fourth order Maxwellian moments using Hermite tensors. The D2Q37 lattice from this paper is frequently used in subsequent literature and is believed to be the minimal Hermite tensors based fourth order lattice. Philippi et al. [179] also present two dimensional lattices that support fifth (D2Q53) and sixth (D2Q81) order Maxwellian moments again using Hermite tensors. Their work uses a second order collision model in order to improve the accuracy of the solution requiring the Maxwellian moments up to sixth order in order to simulate thermal flows.

Siebert et al. [220] also look at the D2Q17, D2Q25 and D2Q37 lattices, testing over a range of temperatures and noting that the reduced fourth order lattice is no worse than the full fourth order one. Rubinstein and Luo [202] also present a derivation of higher order lattices including a D3Q51 lattice capable of recovering up to fourth order moments of the equilibrium distribution. Surmas et al. [237] use a similar technique to Philippe et al., they give an overview of many of the aforementioned lattices as well as introducing new reduced fourth order velocity sets, the on lattice D3Q59 set and the off lattice D3Q33 set. They also expose new full fourth order velocity sets, the on lattice D3Q107 set and the off lattice D3Q52 and D3Q53 sets. This illustrates the substantial reduction in complexity offered by the off lattice scheme. They perform numerical tests using a shock tube with temperature ratio of 4:3 and density ratio of 8:1. Gan et al. [84] use a highly accurate WENO finite difference scheme and a D2Q19 scheme which uses three velocity groups in 6 directions. Although their scheme is formulated as an off lattice finite difference Boltzmann scheme, it is not clear why since their 19 chosen velocities do align with a hexagonal grid. They also present a successful shock tube simulation with a temperature ration of 5:4 and a density ratio of 8:1.

The stability of lattice Boltzmann schemes with additional velocities can be compromised. Work by Chikatamarla, Karlin and Asinari [42, 121, 43, 44] evaluates the stability of potential lattices based on an entropy principle to arrive at stable lattices that are unfortunately not fully fourth order accurate. In one dimension, they use a D1Q5 lattice which uses $\{0, 1, 3\}$ velocities rather than the usual $\{0, 1, 2\}$. The one dimensional case is then generalised to a D2Q25 and D3Q125 lattice, the latter can be pruned down to D3Q41. They note that the D1Q7 lattice would be the minimum required to capture the fourth order moments and hence fully simulate compressible flows however they do not present details about this hypothetical lattice. Shim [217] reviews these methods as well as those of Philippi et al. [178, 179] and several off-lattice schemes mentioned. Shim also notes that the methods of Karlin are only third order accurate while Philippi are fourth order.

Works by Sbragaglia et al. [206] and Scagliarini et al. [207] concentrate on the discrete lattice effects and the forcing term. They also use the fourth order D2Q37 Hermite lattice of Philippi et al [178].

The BGK collision operator will result in a fixed Prandtl number, this can be remidied by using a multiple relaxation time (MRT) collision operator [133, 134, 266]. Zheng et al [266] present a MRT method based on the D2Q17 lattice and test using using low speed Couette flow, Poiseuille flow and a lid driven cavity. The simulation of a shock tube or other strongly compressible flow is a notable omission. Shan and Chen [211] develop a more general multiple relaxation time collision operator for the Boltzmann equation based on a Hermite expansion where the eigenvalues are independent of the velocity basis. They test their method on a shear wave confirming the ability to set the desired Prandtl number. Although claimed to be independent of the velocity basis, their method relies on the orthogonality of the Hermite polynomials which indirectly sets constraints on the lattice used.

Prasianakis and Karlin [186, 187] present a method which uses a low number of velocities (D2Q9) and adds correction terms based on finite difference approximations to the deviation terms in order to simulate low speed thermal flows, the stability of their method is also verified using a shock tube simulation. Although their analysis appears to fully recover the Navier-Stokes equations, they state a caveat that "In two dimensions, the nine populations are found to be adequate for a successful simulation of subsonic compressible flows" hence it is not clear whether the method works well for high speed compressible flows. Prasianakis et al. [188] also apply this method of corrections based on finite difference approximations in order to restore Galilean invariance for athermal methods.

All of the thermal methods mentioned feature a linear or constant dependence of viscosity (ν) and thermal conductivity on temperature (T). This corresponds to a Maxwellian model [36],

$$\nu = C_1 \cdot T.$$

The thermal conductivity (κ) follows the same form resulting in a constant Prandtl number. There are several more accurate models, one that is encountered in textbooks such as Cengel and Turner [32] is the Sutherland model,

$$\nu = \frac{C_1 \cdot T^{3/2}}{T + C_2}.$$

Other models may be created based on the nature of particle interactions and the Boltzmann equation include the Lenard-Jones model or the Lorenz model discussed in Chapter 10 of Chapman and Cowling [36]. Over a small enough temperature range or if the effect of the variation in properties with temperature is not considered the most important feature of the flow under consideration, the linear approximation may be sufficient.

If compressibility effects are not important, then the problem can be greatly simplified by decoupling the energy equation from the momentum and density equations. These are referred to as "hybrid thermal" simulations. When it is assumed that the energy equation does not affect momentum or density in any way, this is referred to as the passive scalar approach. The Boussinesq approximation assumes that the only significant effects of changing energy occur via the associated density changes and buoyancy effects which are treated as an external force. The energy equation is often simplified to an advection-diffusion equation dropping compression work and viscous heating terms although these can in principle be included also.

He, Chen and Doolen [101] use a novel method that includes compression work terms and viscous heating by performing a coupled LB simulation for internal energy. Shan [210] uses an LB simulation for the temperature field treating it more simply as an advection-diffusion equation. Guo, Shi and Zheng [97] employ a similar approach. Others have performed LB simulations of the advection-diffusion equation alone [110, 265, 157].

A finite-difference approach may also be used to solve the advection-diffusion equation [80, 134, 122, 164]. Lallemand and Luo [134] present MRT methods using a D2Q33 lattice which are able to simulate thermal hydrodynamics however they note that hybrid thermal LB methods offer increased stability.

Most of the lattices I have implemented in the code are isothermal and use the Boussinesq approximation to approximate the effects of buoyancy. The scalar buoyancy field is solved by a highly accurate finite volume scheme detailed in Chapter 4. As we will explain, our scalar solver has a higher order of accuracy and greatly improved stability compared to LB methods for advection and diffusion. One lattice that has been implemented is able to simulate full thermal hydrodynamics, that is the `bcc_d3q27_abc` lattice which is fully detailed in Section 3.A.6.

3.4 Discretizing the space and time

In this section we will analyse additional effects which are introduced by the discretization of space. Here we present a new scheme I have devised in order to eliminate errors that arise when non-Lagrangian streaming is used. We will also cover the effect of lattice discretization on the force operator.

3.4.1 Lagrangian streaming

The conventional space and time discretization used results from letting the discrete velocities be integer multiples of the lattice so that the advection term may be replaced by a simple shift in space. The effect of the collision term over the course of the time step is approximated in an upwind fashion by its value at the start of the time step.

$$f_{\boldsymbol{i}}^{x+,t+} = f_{\boldsymbol{i}}^{x,t} + \Omega_{\boldsymbol{i}}^{x,t} + \frac{1}{2}(F_{\boldsymbol{i}}^{x,t} + F_{\boldsymbol{i}}^{x+,t+})$$

We have included what we call the averaged force operator which combines upwind and downwind terms in the correct ratio. This force operator is not often encountered although it is required for an accurate representation as will be shown. Taking the first two terms of the Taylor expansion of this discrete equation gives,

$$\partial_t f_i + \partial_\alpha c_{i\alpha} f_i + \underline{\frac{1}{2}\partial_t\partial_t f_i + \frac{1}{2}\partial_\beta\partial_\beta c_{i\alpha} c_{i\beta} f_i + \partial_t\partial_\alpha c_{i\alpha} f_i} \\ = \Omega_i + F_i + \underline{\frac{1}{2}\partial_\alpha c_{i\alpha} F_i + \frac{1}{2}\partial_t F_i} + \mathcal{O}(\epsilon^3). \quad (3.16)$$

The underlined terms represent second order errors introduced to the Boltzmann equation by the discretization. Additional second order terms added to the RHS of the conservation of mass equation are,

$$\partial_{t_2}\rho = \cdots - \frac{1}{2}\partial_{t_1}\partial_{t_1}\rho - \frac{1}{2}\partial_\alpha\partial_\beta \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} f_i^{eq} - \partial_{t_1}\partial_\alpha \rho u_\alpha + \frac{1}{2}\partial_\alpha F_\alpha.$$

Using the equation for conservation of mass (Eqn. 3.6) we can rearrange,

$$\partial_{t_2}\rho = \cdots - \frac{1}{2}\partial_\alpha \left(\partial_{t_1}\rho u_\alpha + \partial_\beta \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} f_i^{eq} - F_\alpha \right).$$

The bracketed terms sum to zero at the same order in ϵ. It is apparent from this equation that the averaged force operator was required in order to allow complete cancellation of terms proportional to $\partial_\alpha F_\alpha$.

Additional second order terms added to the RHS of the conservation of momentum equation are,

$$\partial_{t_2}\rho u_\alpha = \cdots - \frac{1}{2}\partial_{t_1}\partial_{t_1}\rho u_\alpha - \frac{1}{2}\partial_\beta\partial_\gamma \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} c_{i\gamma} f_i^{eq} - \partial_{t_1}\partial_\beta \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} f_i^{eq} \\ + \frac{1}{2}\partial_\beta \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} F_i + \frac{1}{2}\partial_{t_1} F_\alpha.$$

Using the first order equation for the conservation of momentum (Eqn. 3.10) we can similarly rearrange,

$$\partial_{t_2} \rho u_\alpha = \cdots - \frac{1}{2}\partial_\beta \left(\partial_\gamma \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} c_{i\gamma} f_i^{eq} + \partial_{t_1} \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} f_i^{eq} - \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} F_i \right).$$

The averaged force operator is again required to cancel $\partial_t F_\alpha$ terms. The identity from Eqn. 3.13 is applied to find that the second order errors amount to a negative viscosity contribution,

$$\partial_{t_2} \rho u_\alpha = \cdots - \frac{1}{2} RT \left(\partial_\beta \rho \partial_\alpha u_\beta + \partial_\beta \rho \partial_\beta u_\alpha \right).$$

The total viscosity is now,

$$\nu^s = RT \left(\tau - \frac{1}{2} \right).$$

3.4.2 A non-unit Courant number – Our corrected scheme

Lattice Boltzmann methods typically have a Courant number of exactly one so that particles stream directly from one site to the next. We extended the analysis of the previous section to a more generalized advection scheme. We discretize the advection term using a second order central difference and then add a diffusion term proportional to α. This scheme generalizes to the first order upwind scheme for $\alpha = \frac{\Delta t}{2\,\mathrm{Cr}}$ and the Lax-Wendroff scheme for $\alpha = \frac{\Delta t}{2}$. We have also added an inertia like term which turns out to be necessary in order to restore the accuracy of the first order upwind scheme when the Courant number is not unity. This correction term is initially written with an unknown factor β which is then shown to be necessarily equal to $\beta = \alpha - \frac{\Delta t}{2}$. Once the negative viscosity terms introduced by the discretization error are accounted for, the Lagrangian scheme, the Lax-Wendroff scheme and our corrected first order upwind scheme all have second order accuracy in space and time. We find that the use of a non-unit Courant number can suppress some numerical instabilities that come about as a result of artificially conserved quantities. In this section we also introduce a more general force term which uses first order approximations to correct new errors introduced by our scheme. Again, we begin with variables α_F and β_F and finally express these in terms of α. The force term reduces to the form from the previous section for the Lagrangian case.

We will use the Courant number,

$$\mathrm{Cr} = \frac{||c_{i\alpha}||\Delta t}{||\Delta x_\alpha||}, \tag{3.17}$$

to simplify equations whenever the grid spacing Δx_α appears. The central advection scheme with added diffusion,

$$\begin{aligned}
&\frac{\partial f_i}{\partial t} + c_{i\alpha}\frac{\partial f_i}{\partial x_\alpha} \\
&\quad = \frac{f_i^{x,t+} - f_i^{x,t}}{\Delta t} + ||c_{i\alpha}||\frac{f_i^{x+,t} - f_i^{x-,t}}{2||\Delta x_{i\alpha}||} - \alpha||c_{i\alpha}||^2\frac{f_i^{x+,t} + f_i^{x-,t} - 2f_i^{x,t}}{||\Delta x_{i\alpha}||^2} \\
&\quad = \frac{1}{\Delta t}\left[f_i^{x,t+} - f_i^{x,t} + \frac{\mathrm{Cr}}{2}\left(f_i^{x+,t} - f_i^{x-,t}\right) - \frac{\alpha\,\mathrm{Cr}^2}{\Delta t}\left(f_i^{x+,t} + f_i^{x-,t} - 2f_i^{x,t}\right)\right],
\end{aligned}$$

becomes the Lax-Wendroff scheme when $\alpha = \frac{\Delta t}{2}$,

$$\begin{aligned}
&\frac{\partial f_i}{\partial t} + c_{i\alpha}\frac{\partial f_i}{\partial x_\alpha} \\
&\quad = \frac{1}{\Delta t}\left[f_i^{x,t+} - f_i^{x,t} + \frac{\mathrm{Cr}}{2}\left(f_i^{x+,t} - f_i^{x-,t}\right) - \frac{\mathrm{Cr}^2}{2}\left(f_i^{x+,t} + f_i^{x-,t} - 2f_i^{x,t}\right)\right].
\end{aligned}$$

The same scheme reduces to the first order upwind scheme when $\alpha = \frac{\Delta t}{2\,\mathrm{Cr}}$,

$$\frac{\partial f_i}{\partial t} + c_{i\alpha}\frac{\partial f_i}{\partial x_\alpha} = \frac{1}{\Delta t}\left[f_i^{x,t+} - f_i^{x,t} + \mathrm{Cr}\left(f_i^{x,t} - f_i^{x-,t}\right)\right].$$

Our scheme including the inertia like term and force operator is written,

$$\begin{aligned}
&\frac{f_i^{x,t+} - f_i^{x,t}}{\Delta t} + ||c_{i\alpha}||\frac{f_i^{x+,t} - f_i^{x-,t}}{2||\Delta x_{i\alpha}||} - \alpha||c_{i\alpha}||^2\frac{f_i^{x+,t} - 2f_i^{x,t} + f_i^{x-,t}}{||\Delta x_{i\alpha}||^2} \\
&\qquad + \beta\frac{f_i^{x,t+} - 2f_i^{x,t} + f_i^{x,t-}}{\Delta t^2} = \Omega_i^{x-,t} + F_i^{x,t} \\
&\qquad\qquad - \alpha_F||c_{i\alpha}||\frac{F_i^{x,t} - F_i^{x-,t}}{||\Delta x_{i\alpha}||} + \beta_F\frac{F_i^{x,t+} - F_i^{x,t}}{\Delta t}.
\end{aligned}$$

The Courant number is substituted where convenient and the co-ordinate axes are shifted by one grid unit in order to avoid needing to Taylor expand the collision term. In effect the Taylor expansion is done about the collision center.

$$\begin{aligned}\frac{1}{\Delta t}\left(f_{\boldsymbol{i}}^{x+,t+}-f_{\boldsymbol{i}}^{x+,t}\right)+\frac{\mathrm{Cr}}{2\Delta t}\left(f_{\boldsymbol{i}}^{x++,t}-f_{\boldsymbol{i}}^{x,t}\right)-\frac{\alpha\,\mathrm{Cr}^2}{\Delta t^2}\left(f_{\boldsymbol{i}}^{x++,t}-2f_{\boldsymbol{i}}^{x+,t}+f_{\boldsymbol{i}}^{x,t}\right)&\\+\frac{\beta}{\Delta t^2}\left(f_{\boldsymbol{i}}^{x+,t+}-2f_{\boldsymbol{i}}^{x+,t}+f_{\boldsymbol{i}}^{x+,t-}\right)=\Omega_{\boldsymbol{i}}^{x,t}+F_{\boldsymbol{i}}^{x+,t}&\\-\frac{\alpha_F\,\mathrm{Cr}}{\Delta t}\left(F_{\boldsymbol{i}}^{x+,t}-F_{\boldsymbol{i}}^{x,t}\right)+\frac{\beta_F}{\Delta t}\left(F_{\boldsymbol{i}}^{x+,t+}-F_{\boldsymbol{i}}^{x+,t}\right)&.\end{aligned}$$

Collecting like terms,

$$\begin{aligned}f_{\boldsymbol{i}}^{x+,t+}\left(\frac{1}{\Delta t}+\frac{\beta}{\Delta t^2}\right)+f_{\boldsymbol{i}}^{x+,t}\left(\frac{-1}{\Delta t}+\frac{2\alpha\,\mathrm{Cr}^2}{\Delta t^2}-\frac{2\beta}{\Delta t^2}\right)+f_{\boldsymbol{i}}^{x+,t-}\left(\frac{\beta}{\Delta t^2}\right)&\\+f_{\boldsymbol{i}}^{x++,t}\left(\frac{\mathrm{Cr}}{2\Delta t}-\frac{\alpha\,\mathrm{Cr}^2}{\Delta t^2}\right)+f_{\boldsymbol{i}}^{x,t}\left(\frac{-\,\mathrm{Cr}}{2\Delta t}-\frac{\alpha\,\mathrm{Cr}^2}{\Delta t^2}\right)=\Omega_{\boldsymbol{i}}^{x,t}&\\+F_{\boldsymbol{i}}^{x+,t}\left(1-\frac{\alpha_F\,\mathrm{Cr}}{\Delta t}-\frac{\beta_F}{\Delta t}\right)+F_{\boldsymbol{i}}^{x,t}\left(\frac{\alpha_F\,\mathrm{Cr}}{\Delta t}\right)+F_{\boldsymbol{i}}^{x+,t+}\left(\frac{\beta_F}{\Delta t}\right)&.\qquad(3.18)\end{aligned}$$

Substituting in the Taylor expansions and again only using the first order terms when multiplied by the force operator,

$$
\begin{aligned}
\left(\frac{1}{\Delta t}+\frac{\beta}{\Delta t^2}\right)\Bigg[f_i+\Delta t\partial_t f_i+\frac{\Delta t}{\mathrm{Cr}}\partial_\alpha c_{i\alpha}f_i+\frac{\Delta t^2}{\mathrm{Cr}}\partial_t\partial_\alpha c_{i\alpha}f_i&\\
+\frac{\Delta t^2}{2}\partial_t\partial_t f_i+\frac{\Delta t^2}{2\,\mathrm{Cr}^2}\partial_\alpha\partial_\beta c_{i\alpha}c_{i\beta}f_i\Bigg]&\\
+\left(\frac{-1}{\Delta t}+\frac{2\alpha\,\mathrm{Cr}^2}{\Delta t^2}-\frac{2\beta}{\Delta t^2}\right)\left[f_i+\frac{\Delta t}{\mathrm{Cr}}\partial_\alpha c_{i\alpha}f_i+\frac{\Delta t^2}{2\,\mathrm{Cr}^2}\partial_\alpha\partial_\beta c_{i\alpha}c_{i\beta}f_i\right]&\\
+\left(\frac{\beta}{\Delta t^2}\right)\Bigg[f_i-\Delta t\partial_t f_i+\frac{\Delta t}{\mathrm{Cr}}\partial_\alpha c_{i\alpha}f_i-\frac{\Delta t^2}{\mathrm{Cr}}\partial_t\partial_\alpha c_{i\alpha}f_i&\\
+\frac{\Delta t^2}{2}\partial_t\partial_t f_i+\frac{\Delta t^2}{2\,\mathrm{Cr}^2}\partial_\alpha\partial_\beta c_{i\alpha}c_{i\beta}f_i\Bigg]&\\
+\left(\frac{\mathrm{Cr}}{2\Delta t}-\frac{\alpha\,\mathrm{Cr}^2}{\Delta t^2}\right)\left[f_i+\frac{2\Delta t}{\mathrm{Cr}}\partial_\alpha c_{i\alpha}f_i+\frac{2\Delta t^2}{\mathrm{Cr}^2}\partial_\alpha\partial_\beta c_{i\alpha}c_{i\beta}f_i\right]&\\
+\left(\frac{-\,\mathrm{Cr}}{2\Delta t}-\frac{\alpha\,\mathrm{Cr}^2}{\Delta t^2}\right)f_i=\Omega_i&\\
+\left(1-\frac{\alpha_F\,\mathrm{Cr}}{\Delta t}-\frac{\beta_F}{\Delta t}\right)\left[F_i+\frac{\Delta t}{Cr}\partial_\alpha c_{i\alpha}F_i\right]&\\
+\left(\frac{\alpha_F\,\mathrm{Cr}}{\Delta t}\right)F_i+\left(\frac{\beta_F}{\Delta t}\right)\left[F_i+\frac{\Delta t}{Cr}\partial_\alpha c_{i\alpha}F_i+\Delta t\partial_t F_i\right]&.
\end{aligned}
$$

Collecting terms,

$$
\begin{aligned}
\partial_t f_i+\partial_\alpha c_{i\alpha}f_i+\underline{\frac{\Delta t}{\mathrm{Cr}}\partial_t\partial_\alpha c_{i\alpha}f_i+\left(\frac{\Delta t}{2}+\beta\right)\partial_t\partial_t f_i+\left(\frac{\Delta t}{\mathrm{Cr}}-\alpha\right)\partial_\alpha\partial_\beta c_{i\alpha}c_{i\beta}f_i}&\\
=\Omega_i+F_i+\underline{\left(\frac{\Delta t}{\mathrm{Cr}}-\alpha_F\right)\partial_\alpha c_{i\alpha}F_i+\beta_F\partial_t F_i}&.
\end{aligned}
$$

As before, the terms introduced by discretization error are underlined. The space and time derivatives of the force term can be individually adjusted by the constants α_F and β_F respectively.

Additional second order terms added to the RHS of the density equation are,

$$
\begin{aligned}
\partial_{t_2}\rho=\cdots-\frac{\Delta t}{\mathrm{Cr}}\partial_t\partial_\alpha\rho u_\alpha-\left(\frac{\Delta t}{2}+\beta\right)\partial_t\partial_t\rho&\\
-\left(\frac{\Delta t}{\mathrm{Cr}}-\alpha\right)\partial_\alpha\partial_\beta\sum_{i=0}^{q}c_{i\alpha}c_{i\beta}f_i^{eq}+\left(\frac{\Delta t}{\mathrm{Cr}}-\alpha_F\right)\partial_\alpha F_\alpha&.
\end{aligned}
$$

Using the equation for conservation of mass (Eqn. 3.6) we can rearrange,

$$\partial_{t_2}\rho = \dots - \left(\frac{\Delta t}{\mathrm{Cr}} - \frac{\Delta t}{2} - \beta\right)\partial_t\partial_\alpha\rho u_\alpha$$
$$- \left(\frac{\Delta t}{\mathrm{Cr}} - \alpha\right)\partial_\alpha\partial_\beta\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}f_{\boldsymbol{i}}^{eq} + \left(\frac{\Delta t}{\mathrm{Cr}} - \alpha_F\right)\partial_\alpha F_\alpha.$$

It is clear that in order to cancel the additional terms we need,

$$\frac{\Delta t}{2} + \beta = \alpha \quad \text{and} \quad \alpha_F = \alpha.$$

This results in,

$$\partial_{t_2}\rho = \dots - \left(\frac{\Delta t}{\mathrm{Cr}} - \alpha\right)\partial_\alpha\left(\partial_t\rho u_\alpha + \partial_\beta\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}f_{\boldsymbol{i}}^{eq} - F_\alpha\right),$$

which again sums to zero.

Additional second order terms added to the RHS of the conservation of momentum equation are,

$$\partial_{t_2}\rho u_\alpha = \dots - \frac{\Delta t}{\mathrm{Cr}}\partial_t\partial_\beta\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}f_{\boldsymbol{i}}^{eq} - \alpha\partial_t\partial_t\rho u_\alpha$$
$$- \left(\frac{\Delta t}{\mathrm{Cr}} - \alpha\right)\partial_\beta\partial_\gamma\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\gamma}f_{\boldsymbol{i}}^{eq} + \left(\frac{\Delta t}{\mathrm{Cr}} - \alpha\right)\partial_\beta\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}F_{\boldsymbol{i}} + \beta_F\partial_t F_\alpha.$$

By using Eqn. 3.10 again and setting $\beta_F = \alpha$ we arrive at,

$$\partial_{t_2}\rho u_\alpha = \dots$$
$$+ \left(\alpha - \frac{\Delta t}{\mathrm{Cr}}\right)\partial_\beta\left\{\partial_t\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}f_{\boldsymbol{i}}^{eq} + \partial_\gamma\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}c_{\boldsymbol{i}\gamma}f_{\boldsymbol{i}}^{eq} - \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}c_{\boldsymbol{i}\beta}F_{\boldsymbol{i}}\right\}. \quad (3.19)$$

Eqn. 3.13 is again applied to find that the second order errors amount to a negative viscosity contribution,

$$\partial_{t_2}\rho u_\alpha = \cdots + \left(\alpha - \frac{\Delta t}{\mathrm{Cr}}\right) RT\left(\partial_\beta\rho\partial_\alpha u_\beta + \partial_\beta\rho\partial_\beta u_\alpha\right).$$

The total viscosity is now,

$$\nu^s = RT\left(\tau + \alpha - \frac{\Delta t}{\mathrm{Cr}}\right).$$

With the additional variables all expressed in terms of α, our scheme is written as,

$$\begin{aligned} f_{\boldsymbol{i}}^{x+,t+}\left(\frac{1}{2\Delta t} + \frac{\alpha}{\Delta t^2}\right) + f_{\boldsymbol{i}}^{x+,t}\left(\frac{2\alpha\,\mathrm{Cr}^2}{\Delta t^2} - \frac{2\alpha}{\Delta t^2}\right) + f_{\boldsymbol{i}}^{x+,t-}\left(\frac{\alpha}{\Delta t^2} - \frac{1}{2\Delta t}\right) \\ + f_{\boldsymbol{i}}^{x++,t}\left(\frac{\mathrm{Cr}}{2\Delta t} - \frac{\alpha\,\mathrm{Cr}^2}{\Delta t^2}\right) + f_{\boldsymbol{i}}^{x,t}\left(\frac{-\mathrm{Cr}}{2\Delta t} - \frac{\alpha\,\mathrm{Cr}^2}{\Delta t^2}\right) = \Omega_{\boldsymbol{i}}^{x,t} \\ + F_{\boldsymbol{i}}^{x+,t}\left(1 - \frac{\alpha}{\Delta t}(\mathrm{Cr}+1)\right) + F_{\boldsymbol{i}}^{x,t}\left(\frac{\alpha\,\mathrm{Cr}}{\Delta t}\right) + F_{\boldsymbol{i}}^{x+,t+}\left(\frac{\alpha}{\Delta t}\right). \end{aligned} \tag{3.20}$$

When $\alpha = \frac{\Delta t}{2}$ the Lax-Wendroff based scheme is,

$$\begin{aligned} f_{\boldsymbol{i}}^{x+,t+}\left(\frac{1}{\Delta t}\right) + f_{\boldsymbol{i}}^{x+,t}\left(\frac{\mathrm{Cr}^2}{\Delta t} - \frac{1}{\Delta t}\right) \\ + f_{\boldsymbol{i}}^{x++,t}\left(\frac{\mathrm{Cr}}{2\Delta t} - \frac{\mathrm{Cr}^2}{2\Delta t}\right) + f_{\boldsymbol{i}}^{x,t}\left(\frac{-\mathrm{Cr}}{2\Delta t} - \frac{\mathrm{Cr}^2}{2\Delta t}\right) = \Omega_{\boldsymbol{i}}^{x,t} \\ + F_{\boldsymbol{i}}^{x+,t}\left(\frac{1}{2} - \frac{\mathrm{Cr}}{2}\right) + F_{\boldsymbol{i}}^{x,t}\left(\frac{\mathrm{Cr}}{2}\right) + F_{\boldsymbol{i}}^{x+,t+}\left(\frac{1}{2}\right). \end{aligned} \tag{3.21}$$

When $\alpha = \frac{\Delta t}{2\,\mathrm{Cr}}$ the modified upwind scheme is,

$$\begin{aligned} f_{\boldsymbol{i}}^{x+,t+}\left(\frac{1}{2\Delta t} + \frac{1}{2\,\mathrm{Cr}\,\Delta t}\right) + f_{\boldsymbol{i}}^{x+,t}\left(\frac{\mathrm{Cr}}{\Delta t} - \frac{1}{\mathrm{Cr}\,\Delta t}\right) + f_{\boldsymbol{i}}^{x+,t-}\left(\frac{1}{2\,\mathrm{Cr}\,\Delta t} - \frac{1}{2\Delta t}\right) \\ + f_{\boldsymbol{i}}^{x,t}\left(-\frac{\mathrm{Cr}}{\Delta t}\right) = \Omega_{\boldsymbol{i}}^{x,t} \\ + F_{\boldsymbol{i}}^{x+,t}\left(\frac{1}{2} - \frac{1}{2\,\mathrm{Cr}}\right) + F_{\boldsymbol{i}}^{x,t}\left(\frac{1}{2}\right) + F_{\boldsymbol{i}}^{x+,t+}\left(\frac{1}{2\,\mathrm{Cr}}\right). \end{aligned} \tag{3.22}$$

Both of these reduce to the Lagrangian case when Cr = 1.

3.4.3 Force term

As we have demonstrated in the previous sections, the moments of the force operator must be correctly set up to second order in order to avoid the introduction of errors into the multi-scale expansion of the Boltzmann equation. Martys et al. [150] showed quite early on that the correct form for the force operator can be deduced using Hermite polynomials.

The upwind collision operator and Lagrangian streaming typically used in lattice Boltzmann methods introduces a discretization error which contains terms proportional to the force operator. As we have shown in the previous section, averaging the force operator over two points in space and time is sufficient to cancel these errors. The earliest reference we have found that correctly identifies this requirement is the textbook by Wolf and Gladrow [255], hence we will refer to this scheme as the WG force term. Despite its straightforward nature and early publication in a textbook, many papers that have since been published have missed this technique. In fact, Cheng and Li [41] published this exact scheme in 2008 seemingly unaware of previous work. Du and Shi [69] sample at 3 locations to arrive at a similar scheme in 2006. Around the same time as Wolf and Gladrow, Buick and Greated [22] presented a review of several methods, none of which ultimately eliminate all errors.

Ladd and Verberg [132] identify another means of cancelling discretization errors which involves redefining the momentum which is used in calculation of the equilibrium distribution,

$$\rho u_\alpha^\star = \sum_{i=0}^{q} c_{i\alpha} f_i + \frac{\Delta t}{2} F_\alpha.$$

Guo, Zheng and Shi [98] compare many forcing terms and also use this technique in their scheme. Similar conclusions are reached by Mohamad and Kuzmin [158], Kuzmin et al. [130], Silva and Semiao [221] and Silva and Semiao [222]. Son et al. [227] implement an immersed boundary condition treating the force operator using this redefined momentum technique. None of these authors mention the WG scheme.

3.5 Lattice topologies and order

The velocity set chosen to discretize phase space plays a crucial role in lattice Boltzmann methods. The velocities are typically chosen so that in one time step particles stream an integer number of grid units and hence just move from point to point in a Lagrangian fashion. The lattice sites must therefore tile space, in this thesis we will consider square and hexagonal lattice topologies in two dimensions and cubic, body-centered cubic (BCC) and face-centered cubic (FCC) topologies in three dimensions. The unit cells for hexagonal, square and cubic lattices are the same shapes. The unit cells for the body-centered cubic lattice are truncated octahedra and for the face-centered cubic lattice are rhombic-dodecahedra. The non-unit Courant number schemes discussed in Section 3.4.2 are also subject to the same lattice restrictions since the analysis assumes no mixing between particle populations during streaming and a uniform Courant number for all particles.

Table 3.1 compares the different lattice topologies, of particular interest is the ratio of the major sphere to minor sphere diameter which represents how uniformly that lattice tiles space. The Cartesian square and cubic lattices are the worst in this regard with a large difference in resolution between the grid aligned direction and diagonal. Thus the hexagonal and `bcc` lattices have the most isotropic resolution in two and three dimensions respectively. The BCC lattice may be decomposed into two cubic sub-lattices, each offset by half a grid unit in each direction from the other. Likewise, the FCC lattice can be split into four sub-lattices, each is offset from the others by half a grid unit in two directions.

Using a cubic lattice and a structured grid allows the location co-ordinates of a point to be determined based on three indices `ix`, `iy` and `iz`. The neighbours are determined by considering offsets in each direction as demonstrated in Figure 3.1. Using the BCC lattice, an index `il` is introduced in order to represent the sub lattice a site belongs to in order to efficiently store and address data. The co-ordinates and neighbours can then be found as demonstrated in Figure 3.2. Neighbours in the

2D	SQUARE	HEXAGONAL	
Unit cell	Square	Hexagon	
Number of edges	4	6	
Minor circle (d_{min})	1	1	
Major circle (d_{maj})	$\sqrt{2}$	$\frac{2}{\sqrt{3}}$	
$d_{\text{maj}} : d_{\text{min}}$	$\sqrt{2} \sim 1.41$	$\frac{2}{\sqrt{3}} \sim 1.15$	
3D	CUBIC	BCC	FCC
Unit cell	Cube	Truncated octahedron	Rhombic dodecahedron
Number of faces	6	14	12
Minor sphere (d_{min})	1	$\sqrt{3}$	1
Major sphere (d_{maj})	$\sqrt{3}$	$\sqrt{5}$	$\sqrt{2}$
$d_{\text{maj}} : d_{\text{min}}$	$\sqrt{3} \sim 1.73$	$\sqrt{\frac{5}{3}} \sim 1.29$	$\sqrt{2} \sim 1.41$

Table 3.1: Lattice topologies.

same sub-lattice are simply offset in the Cartesian directions however neighbours in the opposite lattice are offset along diagonals. We introduce a new set of indices `jx`, `jy` and `jz` which represent locations in the opposite lattice, calculated in a way that places them offset in the positive direction along all axes. This calculation assumes that the "one" lattice is offset in the positive direction from the "zero" lattice and that one and zero integers are used to represent the sub-lattice index. The opposite lattice index `jl` can simply be found using the not operator.

If the alignment restriction is relaxed, then we arrive at the previously mentioned

```
self = *(f + ix + iy*nx + iz*nx*ny);

x = dx*(float)ix;
y = dy*(float)iy;
z = dz*(float)iz;

sy = nx;
sz = nx*ny;

nbr_0 = *(f + (ix+1) +  iy    *sy +  iz    *sz); // (+dx,  0,  0)
nbr_1 = *(f + (ix-1) +  iy    *sy +  iz    *sz); // (-dx,  0,  0)
nbr_2 = *(f +  ix    + (iy+1)*sy +  iz    *sz); // (  0,+dy,  0)
nbr_3 = *(f +  ix    + (iy-1)*sy +  iz    *sz); // (  0,-dy,  0)
nbr_4 = *(f +  ix    +  iy    *sy + (iz+1)*sz); // (  0,  0,+dz)
nbr_5 = *(f +  ix    +  iy    *sy + (iz-1)*sz); // (  0,  0,-dz)
```

Figure 3.1: Pseudocode to determine location and neighbours for cubic structured grid.

```
self  = *(f + ix + iy*nx + iz*nx*ny + il*nx*ny*nz);

x = dx*((float)ix*2. + (float)il);
y = dy*((float)iy*2. + (float)il);
z = dz*((float)iz*2. + (float)il);

sy  = nx;
sz  = nx*ny;
sl  = nx*ny*nz;

nbr_0  = *(f + (ix+1) +  iy    *sy +  iz    *sz + il*sl); // (+2dx,   0,   0)
nbr_1  = *(f + (ix-1) +  iy    *sy +  iz    *sz + il*sl); // (-2dx,   0,   0)
nbr_2  = *(f +  ix    + (iy+1)*sy +  iz    *sz + il*sl); // (   0,+2dy,   0)
nbr_3  = *(f +  ix    + (iy-1)*sy +  iz    *sz + il*sl); // (   0,-2dy,   0)
nbr_4  = *(f +  ix    +  iy    *sy + (iz+1)*sz + il*sl); // (   0,   0,+2dz)
nbr_5  = *(f +  ix    +  iy    *sy + (iz-1)*sz + il*sl); // (   0,   0,-2dz)

jx  = ix + il;
jy  = iy + il;
jz  = iz + il;
jl  = !il;

nbr_6  = *(f +  jx    +  jy    *sy +  jz    *sz + jl*sl); // (+dx,+dx,+dx)
nbr_7  = *(f +  jx    +  jy    *sy + (jz-1)*sz + jl*sl); // (+dx,+dx,-dx)
nbr_8  = *(f +  jx    + (jy-1)*sy +  jz    *sz + jl*sl); // (+dx,-dx,+dx)
nbr_9  = *(f +  jx    + (jy-1)*sy + (jz-1)*sz + jl*sl); // (+dx,-dx,-dx)
nbr_a  = *(f + (jx-1) +  jy    *sy +  jz    *sz + jl*sl); // (-dx,+dx,+dx)
nbr_b  = *(f + (jx-1) +  jy    *sy + (jz-1)*sz + jl*sl); // (-dx,+dx,-dx)
nbr_c  = *(f + (jx-1) + (jy-1)*sy +  jz    *sz + jl*sl); // (-dx,-dx,+dx)
nbr_d  = *(f + (jx-1) + (jy-1)*sy + (jz-1)*sz + jl*sl); // (-dx,-dx,-dx)
```

Figure 3.2: Pseudocode to determine location and neighbours for BCC structured grid.

off-lattice schemes. The D3Q33 and D3Q53 lattices of Surmas et al. [237] and the D3Q27 lattice of Shan et al. [213] maintain velocity directions which are aligned with a Cartesian lattice sites but with magnitudes that are not integer multiples of grid spacing and hence do not allow exact Lagrangian streaming. Other off-lattice schemes use unit cells that do not even tile space such as the octagonal D2Q17 scheme introduced by Vahala, Pavlo et al. [246, 174] or the icosahedral D3Q13 scheme of Shan et al. [213] or Tamura et al. [239].

The number of velocities in the set is also equal to the number of degrees of freedom in terms of moments of the particle distribution that can be specified independently. As we have seen in the preceding sections, the Chapman-Enskog (CE) procedure places requirements on the moments of the equilibrium distribution. Equilibrium moments up to the second order always need to be set. The third order moments in principle also need to take their correct Maxwellian values however a common simplification is to allow a linear dependence on the first order moments (momentum) which introduces errors that are proportional to $\mathcal{O}(u^3)$. These athermal, small Mach number hydrodynamics simulations thus only require the second order moments to be linearly independent. The third order errors may be eliminated by correctly specifying the third order equilibrium moments however this is of limited use unless the energy equation is also correctly solved since thermal effects will also become important at larger Mach numbers. The full thermal hydrodynamic equations are modelled by also specifying the reduced fourth order moments $\sum f_{\boldsymbol{i}}^{eq} c_{\boldsymbol{i}\gamma} c_{\boldsymbol{i}\gamma} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta}$. The full fourth order moments are not strictly required although they may affect higher order errors and stability. The results of Siebert et al. [220] "did not show any meaningful difference" between full and reduced fourth order Maxwellian lattices. Higher order moments are required in some situations such as multi-phase flows or higher order CE expansions approaching the Burnett and hyper-Burnett equations. The Hermite tensor approach and Grad method predict improved accuracy as increasingly higher order moments are correctly represented. The higher order collision operator of Philippi et al. [179] requires equilibrium moments up to sixth order.

The number of unique moments of order s with dimension DIM is equal to the number of unique combinations of s lattice vectors and can be calculated using the multiset coefficient,

$$\left(\!\!\binom{\text{DIM}}{s}\!\!\right).$$

The binomial coefficient gives the number of ways of selecting k elements from n possibilities without repetition,

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{n_k}{k!}.$$

The multiset coefficient gives the number of ways of selecting k elements from n possibilities allowing repetition,

$$\left(\!\!\binom{n}{k}\!\!\right) = \binom{n+k-1}{k} = \frac{(n+k-1)!}{k!(n-1)!} = \frac{n^{(k)}}{k!}.$$

	Each		Total	
Lattice vectors	2D	3D	2D	3D
c_0	1	1	1	1
c_α	2	3	3	4
$c_\alpha c_\beta$	3	6	6	10
$c_\alpha c_\beta c_\gamma$	4	10	10	20
$c_\gamma c_\gamma c_\alpha c_\beta$	3	6	13	26
$c_\alpha c_\beta c_\gamma c_\delta$	5	15	15	35
$c_\alpha c_\beta c_\gamma c_\delta c_\epsilon$	6	21	21	56
$c_\alpha c_\beta c_\gamma c_\delta c_\epsilon c_\zeta$	7	28	28	84

Table 3.2: Number of moments at each order and cumulative total.

The rising and falling factorials are,

$$n^{(k)} = n \cdot (n+1) \cdot (n+2) \cdots (n+k-1),$$
$$n_k = n \cdot (n-1) \cdot (n-2) \cdots (n-k+1).$$

The number of unique moments at each order and the sum of all moments required up to that order are listed in Table 3.2. This gives a lower bound on the number of lattice velocities required however simply having enough velocities is not a sufficient condition. Each lattice vector up to the desired order must be tested to make sure it it linearly independent to all others up to that order. Hence most velocity sets have a larger number of velocities than these minima would suggest. Also, the Hermite tensor approach [213, 178] places a more stringent restriction on the choice of lattice, that is the ability to construct Hermite polynomial tensors at the chosen order.

The procedure for testing linear independence of lattice vectors for a given velocity set is straightforward, we simply make sure each of the lattice vectors in turn are linearly independent using a program I have developed called `lb_moment`. Although it is possible using hand calculations, large velocity sets become onerous and mistakes may be difficult to detect. The code has been developed using the free software symbolic algebra programming language Maxima.

Lattices are often identified using a DQ terminology where D is the number of physical dimensions and Q is the number of velocities. This information alone is insufficient to identify a lattice. We use a more precise terminology, first the lattice neighbours are assigned letters based on which symmetric group they belong to. These groups are detailed for square (Table 3.3), hexagonal (Table 3.4), cubic (Table 3.5 and Table 3.6), BCC (Table 3.7) and FCC (Table 3.8) topologies. We also illustrate these families in Figure 3.4 to Figure 3.14. A single letter subscript indicates that the closest neighbour from that group is used, additional neighbours from the same group may be created by using integer multiples, in this case a bit field is appended with a one signifying that neighbour is used, the right most bit represents the nearest neighbour. We also prefix the lattice with a 3 character identifier, `sqr`, `hex`, `cub`, `bcc` and `fcc` in order to identify the topology. Using this terminology the commonly used cubic lattices are `cub_d3q15_ac`, `cub_d3q19_ab` and `cub_d3q27_abc`.

Figure 3.3 illustrates what is typically referred to as the `d2q9` lattice. Using our notation this is the `sqr_d2q9_ab` lattice. Figure 3.4 shows a large number of lattice neighbours for the square two dimensional topology. By taking the 'a' and 'b' neighbours, the `sqr_d2q9_ab` lattice is obtained. This figure and correspondingly Figure 3.5 which shows the hexagonal topology can be used to visualise all other lattice combinations. The figures that follow also illustrate three dimensional lattices.

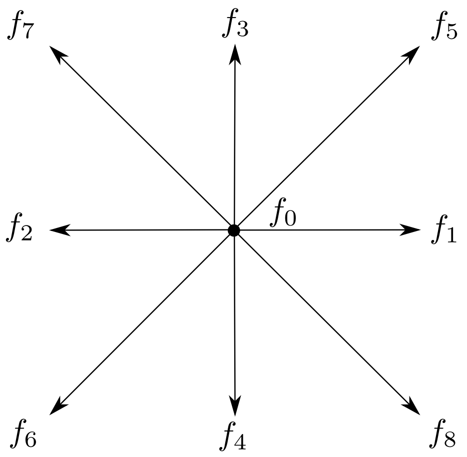


Figure 3.3: Illustration of typical `d2q9` lattice.

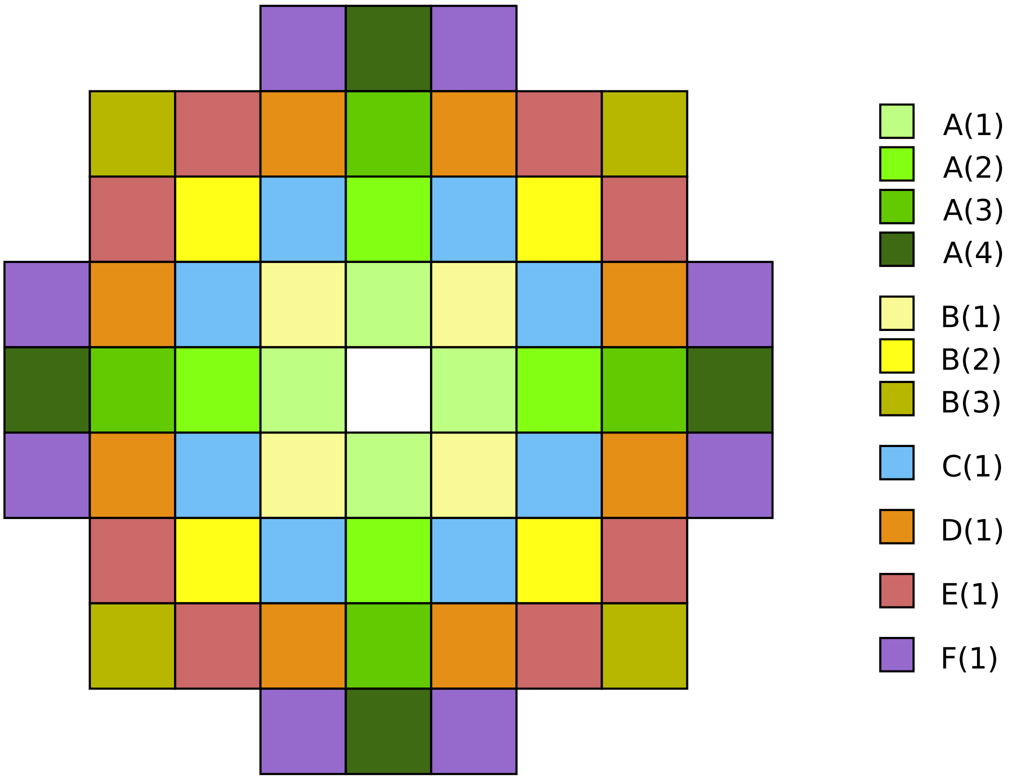


Figure 3.4: Illustration of `sqr_d2q61_a1111b111cdef` lattice.

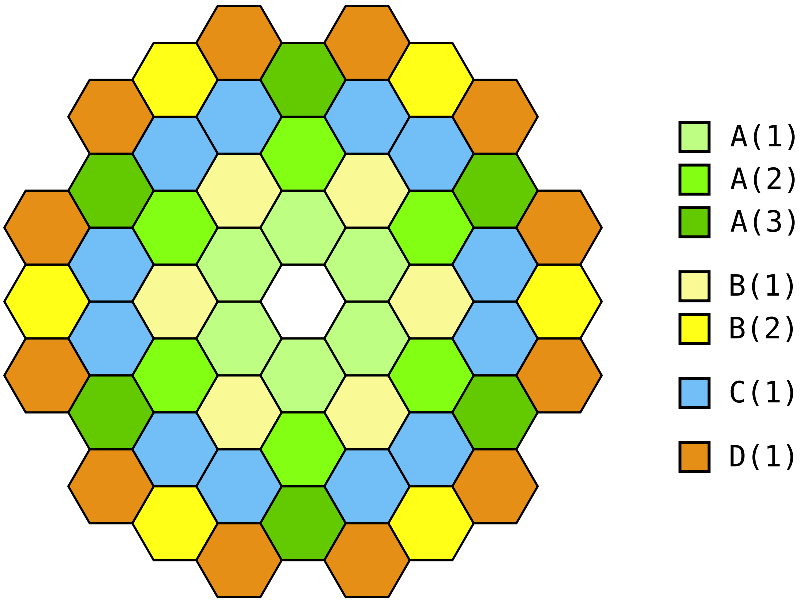


Figure 3.5: Illustration of `hex_d2q55_a111b11cd` lattice.

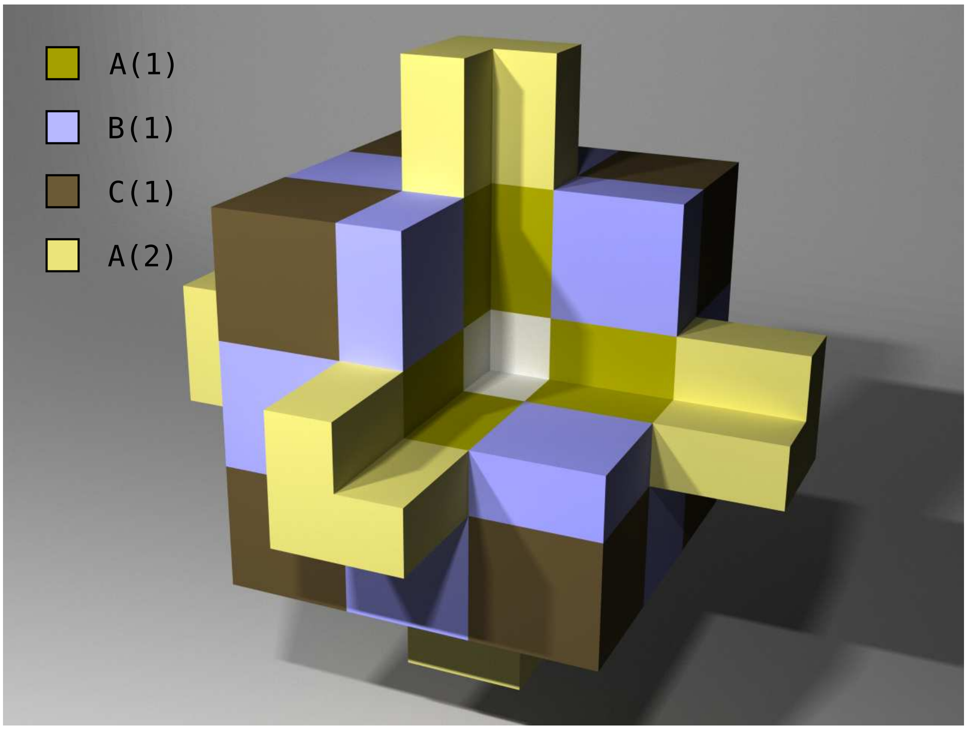


Figure 3.6: Illustration of `cub_d3q33_a11bc` lattice.

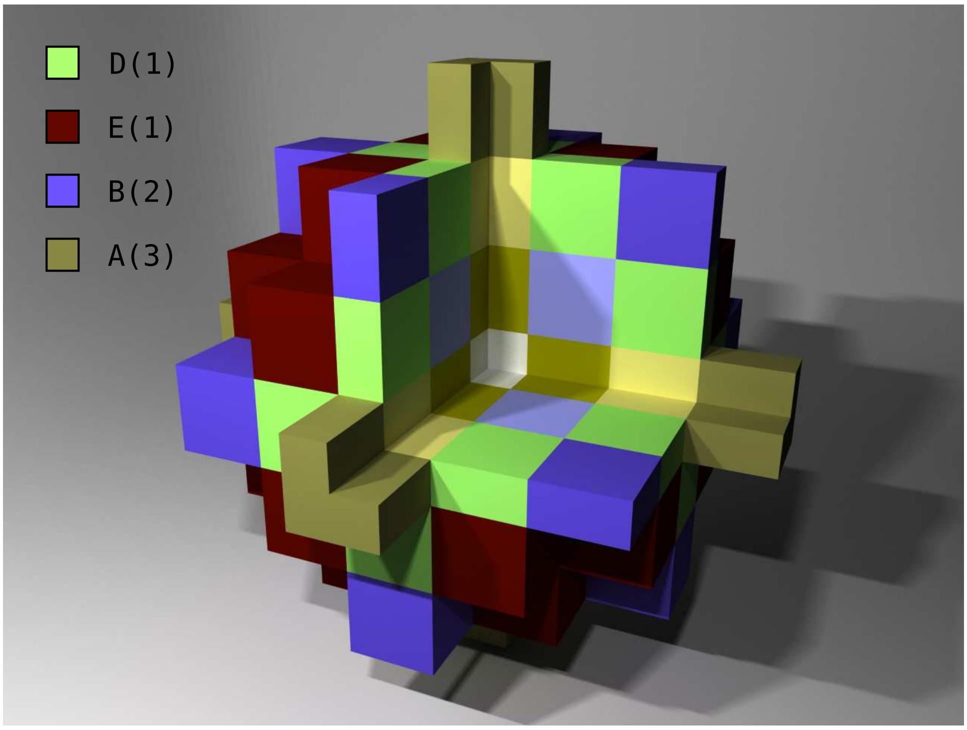


Figure 3.7: Illustration of `cub_d3q99_a111b11cde` lattice.

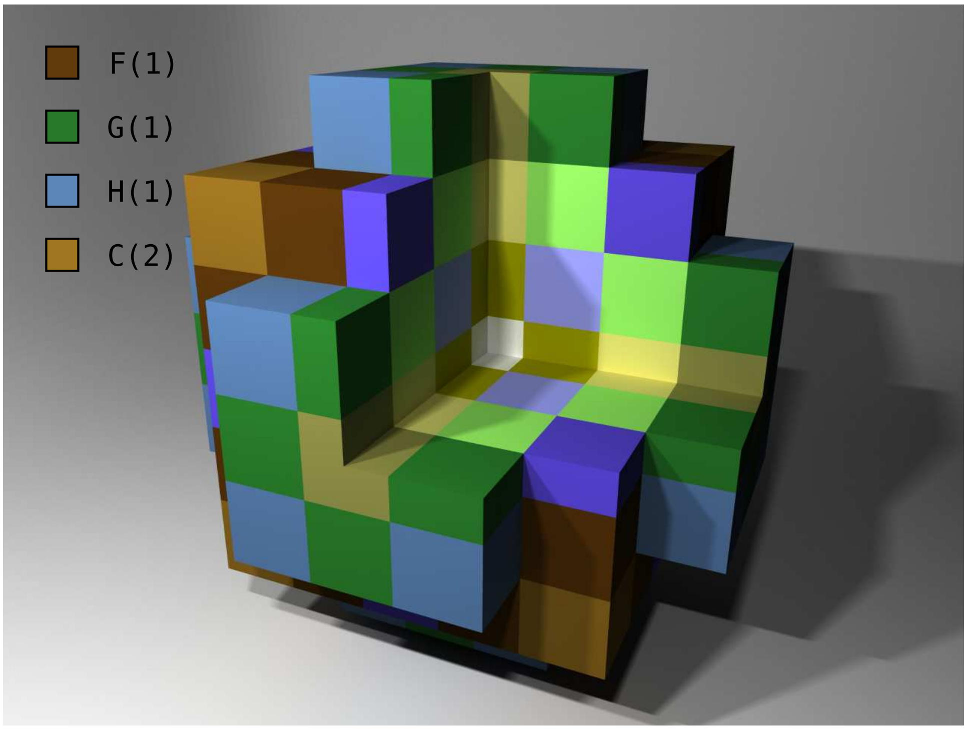


Figure 3.8: Illustration of `cub_d3q179_a111b11c11defgh` lattice.

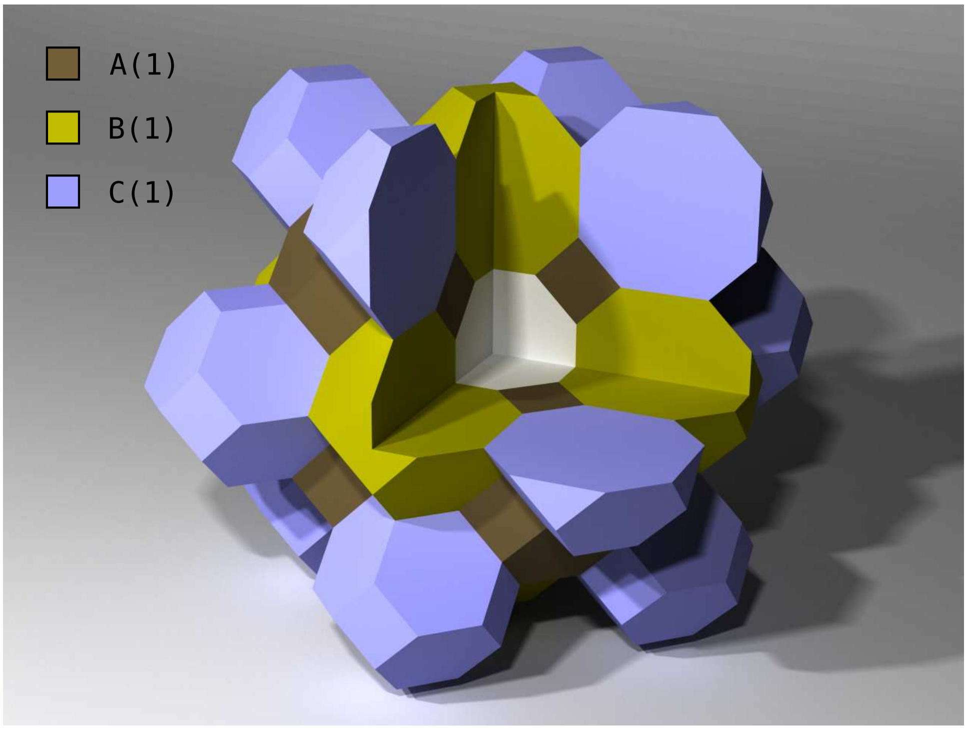


Figure 3.9: Illustration of `bcc_d3q27_abc` lattice.

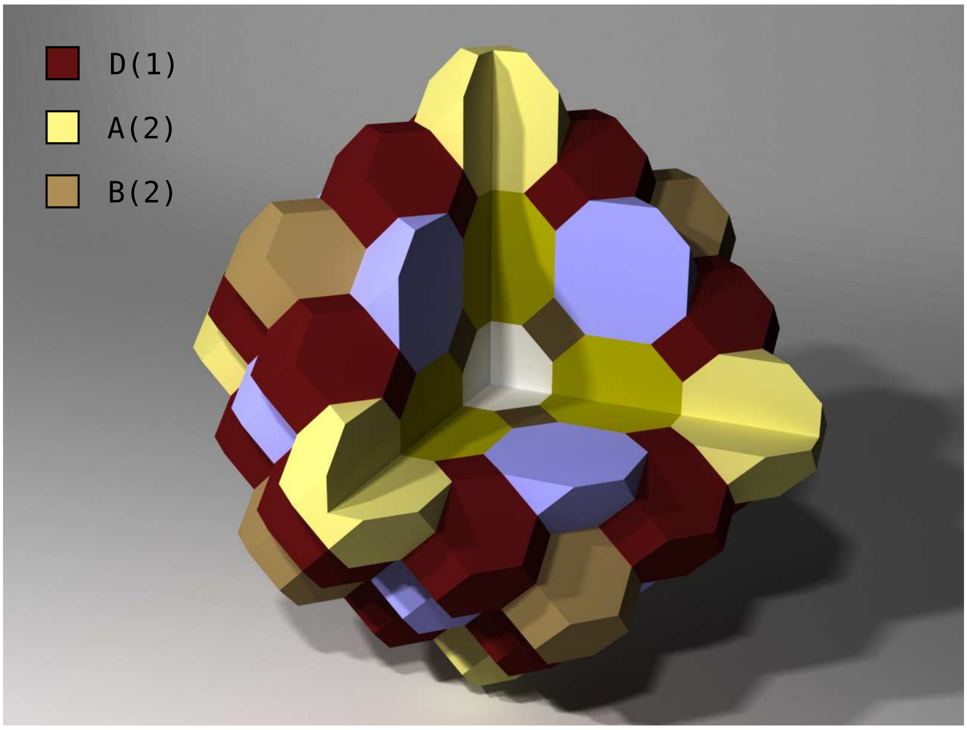


Figure 3.10: Illustration of `bcc_d3q65_a11b11cd` lattice.

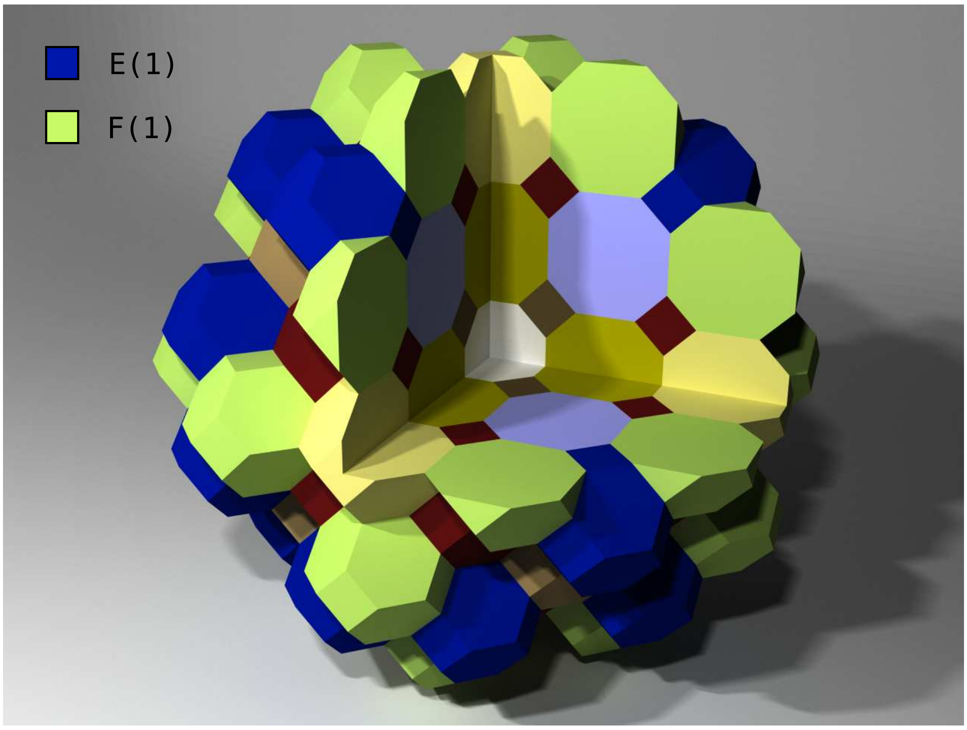


Figure 3.11: Illustration of `bcc_d3q113_a11b11cdef` lattice.

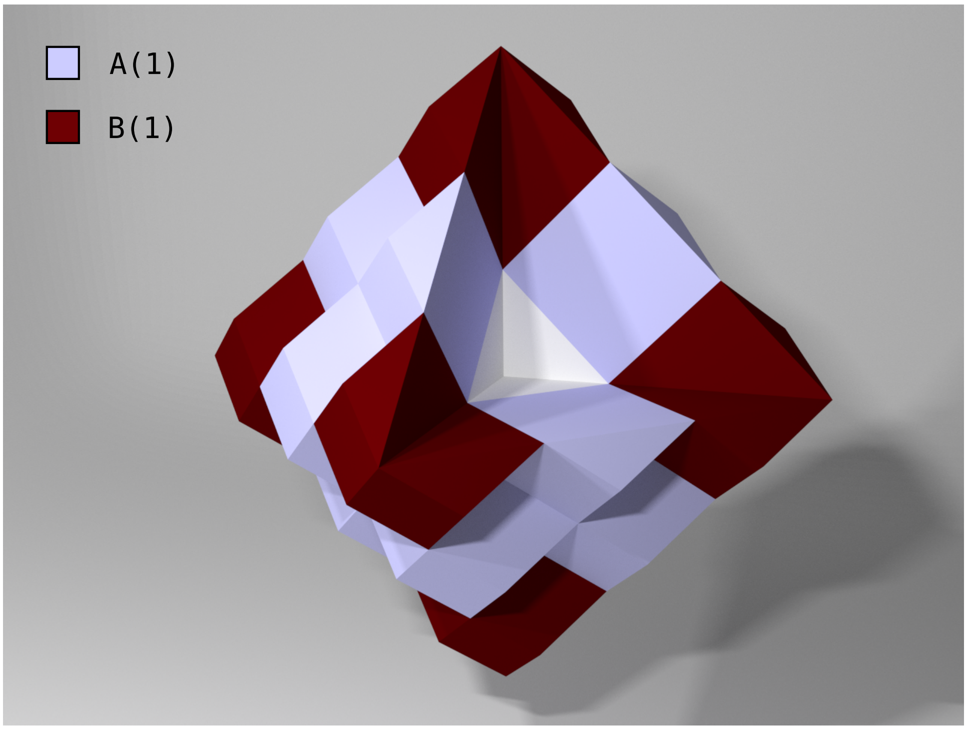


Figure 3.12: Illustration of `fcc_d3q19_ab` lattice.

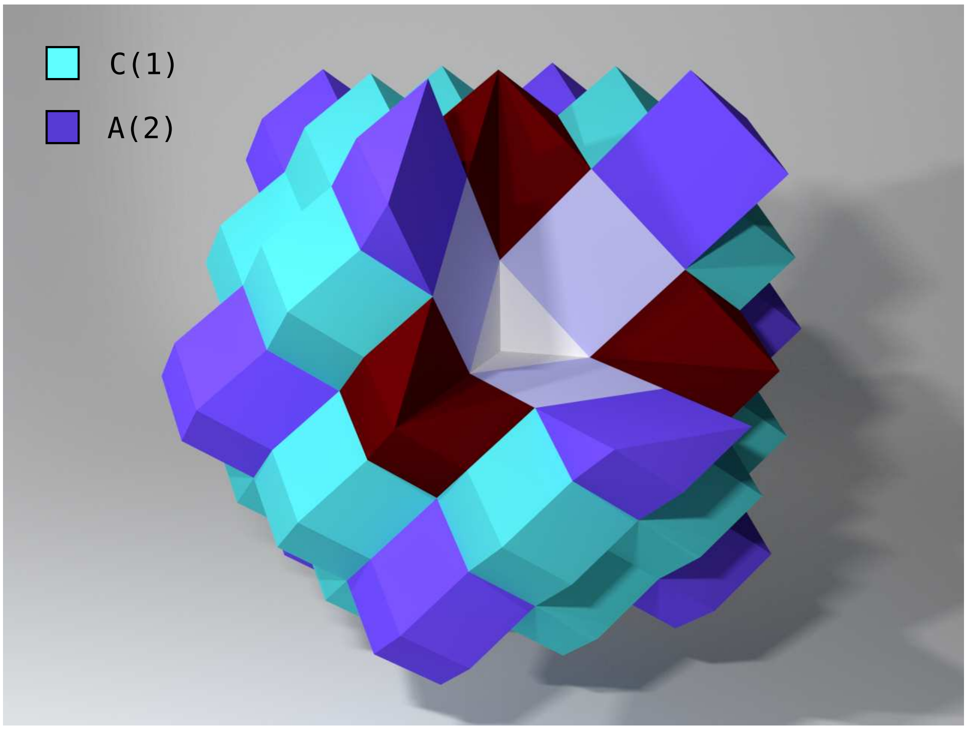


Figure 3.13: Illustration of `fcc_d3q55_a11bc` lattice.

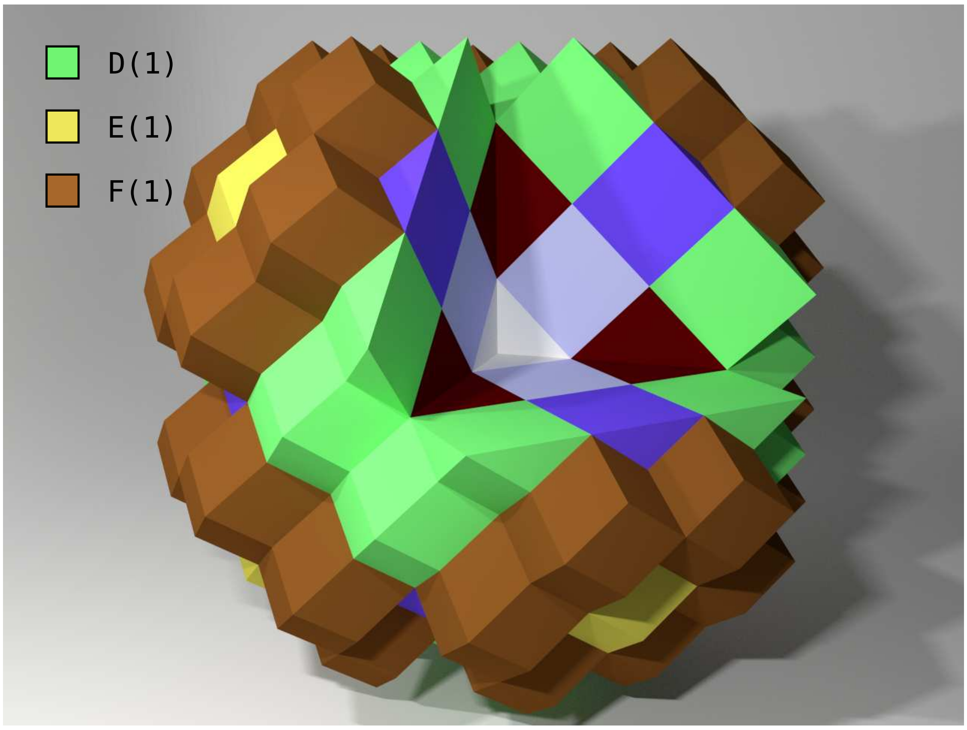


Figure 3.14: Illustration of `fcc_d3q135_a11bcdef` lattice.

neighbour	count	co-ordinates	distance

A	4	`( 1, 0) ( 0, 1)`
		`(-1, 0) ( 0,-1)`
			1

B	4	`( 1, 1) (-1, 1)`
		`( 1,-1) (-1,-1)`
			$\sqrt{2} \sim 1.41$

C	8	`( 2, 1) (-2, 1) ( 1, 2) ( 1,-2)`
		`( 2,-1) (-2,-1) (-1, 2) (-1,-2)`
			$\sqrt{5} \sim 2.24$

D	8	`( 3, 1) (-3, 1) ( 1, 3) ( 1,-3)`
		`( 3,-1) (-3,-1) (-1, 3) (-1,-3)`
			$\sqrt{10} \sim 3.16$

E	8	`( 3, 2) (-3, 2) ( 2, 3) ( 2,-3)`
		`( 3,-2) (-3,-2) (-2, 3) (-2,-3)`
			$\sqrt{13} \sim 3.61$

F	8	`( 4, 1) (-4, 1) ( 1, 4) ( 1,-4)`
		`( 4,-1) (-4,-1) (-1, 4) (-1,-4)`
			$\sqrt{17} \sim 4.12$

Table 3.3: Lattice neighbour families – square lattice.

We also list a sequence of nearest neighbours for the same topologies in Table 3.9 to Table 3.13. A selection of lattices up to sixth order and including some of the previously mentioned Hermite lattices are detailed in Table 3.14 to Table 3.18, these tables also detail the number of unique directions and how many unique velocity magnitudes are present in each set. The lattices listed in these tables are far from an exhaustive list, they were mostly developed by trial and error in an effort to maximise the order of moments. We have not listed all lattices that were tested. The data in these tables suggests that in order to effectively increase the order, there is a balance between adding new directions and new magnitudes. Adding a new family of neighbours adds several new directions and one new magnitude while adding a new magnitude for an existing family does not add any new directions. For example, even though the `cub_d3q107_a1011b11c11dh` lattice has 107 velocities and 9 magnitudes, it only has 75 directions which results in fifth order accuracy. Meanwhile the `cub_d3q99_a100bcdfg` lattice has 99 velocities and 99 directions and is sixth order accurate despite only having 5 velocity magnitudes since the `a100` and `f` neighbours both have a magnitude of 3.

Searching for velocity sets that are able to capture hydrodynamics with a minimum number of velocities, the `fcc_d3q13_a` lattice stands out as having exactly the min-

neighbour	count	co-ordinates	distance

A	6	`(   0,  2) ( 3a,  1) (-3a,  1)`
		`(   0, -2) ( 3a, -1) (-3a, -1)`
			2

B	6	`(  3a,  3) (-3a,  3) ( 6a,  0)`
		`(  3a, -3) (-3a, -3) (-6a,  0)`
			$\sqrt{12} \sim 3.46$

C	12	`(  9a,  1) ( 6a,  4) ( 3a,  5)`
		`(  9a, -1) ( 6a, -4) ( 3a, -5)`
		`( -9a,  1) (-6a,  4) (-3a,  5)`
		`( -9a, -1) (-6a, -4) (-3a, -5)`
			$\sqrt{28} \sim 5.29$

D	12	`( 12a,  2) ( 9a,  5) ( 3a,  7)`
		`( 12a, -2) ( 9a, -5) ( 3a, -7)`
		`(-12a,  2) (-9a,  5) (-3a,  7)`
		`(-12a, -2) (-9a, -5) (-3a, -7)`
			$\sqrt{52} \sim 7.21$

Table 3.4: Lattice neighbour families – hexagonal lattice.

imum number of velocities. A drawback of this lattice is that it is not a Hermite lattice and hence the usual explicit equilibrium distribution cannot be used. It is possible to use a modified distribution we have developed in Section 3.A.3 or to use the MRT method. If the same basis velocities are used on a cubic lattice, it will result in 4 decoupled `fcc` simulations. The `bcc_d3q15_ab` and `cub_d3q15_ac` second order lattices are also fairly computationally efficient however the cubic version possesses a staggered invariant that creates stability issues [261, 191, 119]. The `bcc` version, first seen in [5] does not suffer from the same problems. The `cub`, `bcc` and `fcc` lattices all have 27 speed reduced fourth order lattices, the cubic version being the less commonly encountered `cub_d3q27_a10bc` lattice of McNamara et al. [153, 154] which has the velocities along the Cartesian axes increased to two. These lattices compare very well with the minimum of 26 velocities required, the minimal reduced fourth order Hermite lattices `cub_d3q59_a111b11c11` from Surmas et al. [237] requires more than twice the number of velocities. The minimal third order Hermite lattice `cub_d3q39_a111b10c` from the same paper is actually able to represent the reduced fourth order moments but does not possess the required isotropy for use as a reduced fourth order Hermite lattice. Similarly the full fourth order `cub_d3q107_a1011b11c11dh` Hermite lattice from the same paper is actually able to represent the fifth order moments.

neighbour	count	co-ordinates	distance
A	6	(1, 0, 0) (0, 1, 0) (0, 0, 1)	1
		(-1, 0, 0) (0,-1, 0) (0, 0,-1)	
B	12	(1, 1, 0) (0, 1, 1) (1, 0, 1)	$\sqrt{2} \sim 1.41$
		(1,-1, 0) (0, 1,-1) (1, 0,-1)	
		(-1, 1, 0) (0,-1, 1) (-1, 0, 1)	
		(-1,-1, 0) (0,-1,-1) (-1, 0,-1)	
C	8	(1, 1, 1) (-1, 1, 1)	$\sqrt{3} \sim 1.73$
		(1, 1,-1) (-1, 1,-1)	
		(1,-1, 1) (-1,-1, 1)	
		(1,-1,-1) (-1,-1,-1)	
D	24	(2, 1, 0) (1, 2, 0) (1, 0, 2)	$\sqrt{5} \sim 2.24$
		(2,-1, 0) (-1, 2, 0) (-1, 0, 2)	
		(2, 0, 1) (0, 2, 1) (0, 1, 2)	
		(2, 0,-1) (0, 2,-1) (0,-1, 2)	
		(-2, 1, 0) (1,-2, 0) (1, 0,-2)	
		(-2,-1, 0) (-1,-2, 0) (-1, 0,-2)	
		(-2, 0, 1) (0,-2, 1) (0, 1,-2)	
		(-2, 0,-1) (0,-2,-1) (0,-1,-2)	
E	24	(2, 1, 1) (1, 2, 1) (1, 1, 2)	$\sqrt{6} \sim 2.45$
		(2, 1,-1) (1, 2,-1) (1,-1, 2)	
		(2,-1, 1) (-1, 2, 1) (-1, 1, 2)	
		(2,-1,-1) (-1, 2,-1) (-1,-1, 2)	
		(-2, 1, 1) (1,-2, 1) (1, 1,-2)	
		(-2, 1,-1) (1,-2,-1) (1,-1,-2)	
		(-2,-1, 1) (-1,-2, 1) (-1, 1,-2)	
		(-2,-1,-1) (-1,-2,-1) (-1,-1,-2)	
F	24	(2, 2, 1) (1, 2, 2) (2, 1, 2)	3
		(2, 2,-1) (-1, 2, 2) (2,-1, 2)	
		(2,-2, 1) (1, 2,-2) (-2, 1, 2)	
		(2,-2,-1) (-1, 2,-2) (-2,-1, 2)	
		(-2, 2, 1) (1,-2, 2) (2, 1,-2)	
		(-2, 2,-1) (-1,-2, 2) (2,-1,-2)	
		(-2,-2, 1) (1,-2,-2) (-2, 1,-2)	
		(-2,-2,-1) (-1,-2,-2) (-2,-1,-2)	

Table 3.5: Lattice neighbour families – cubic lattice.

neighbour	count	co-ordinates			distance
G	24	(3, 1, 0)	(1, 3, 0)	(1, 0, 3)	$\sqrt{10} \sim 3.16$
		(3,-1, 0)	(-1, 3, 0)	(-1, 0, 3)	
		(3, 0, 1)	(0, 3, 1)	(0, 1, 3)	
		(3, 0,-1)	(0, 3,-1)	(0,-1, 3)	
		(-3, 1, 0)	(1,-3, 0)	(1, 0,-3)	
		(-3,-1, 0)	(-1,-3, 0)	(-1, 0,-3)	
		(-3, 0, 1)	(0,-3, 1)	(0, 1,-3)	
		(-3, 0,-1)	(0,-3,-1)	(0,-1,-3)	
H	24	(3, 1, 1)	(1, 3, 1)	(1, 1, 3)	$\sqrt{11} \sim 3.32$
		(3, 1,-1)	(1, 3,-1)	(1,-1, 3)	
		(3,-1, 1)	(-1, 3, 1)	(-1, 1, 3)	
		(3,-1,-1)	(-1, 3,-1)	(-1,-1, 3)	
		(-3, 1, 1)	(1,-3, 1)	(1, 1,-3)	
		(-3, 1,-1)	(1,-3,-1)	(1,-1,-3)	
		(-3,-1, 1)	(-1,-3, 1)	(-1, 1,-3)	
		(-3,-1,-1)	(-1,-3,-1)	(-1,-1,-3)	

Table 3.6: Lattice neighbour families – cubic lattice (continued).

If we use the MRT method, then an explicit formula for the equilibrium distribution is not required, the method may be implemented using the moments alone. It is always possible to figure out an explicit equilibrium distribution by inverting the moment matrix although a simple form involving symmetric tensors like that obtained using the Hermite approach may not be forthcoming. As we have shown in Section 3.A.3, it is also possible to construct an equilibrium distribution for some lattices that do not possess the isotropic tensors usually required by the Hermite tensor approach.

neighbour	count	co-ordinates	distance
A	8	`( 1, 1, 1) (-1, 1, 1)`	$\sqrt{3} \sim 1.73$
		`( 1, 1,-1) (-1, 1,-1)`	
		`( 1,-1, 1) (-1,-1, 1)`	
		`( 1,-1,-1) (-1,-1,-1)`	
B	6	`( 2, 0, 0) ( 0, 2, 0) ( 0, 0, 2)`	2
		`(-2, 0, 0) ( 0,-2, 0) ( 0, 0,-2)`	
C	12	`( 2, 2, 0) ( 0, 2, 2) ( 2, 0, 2)`	$\sqrt{8} \sim 2.83$
		`( 2,-2, 0) ( 0, 2,-2) ( 2, 0,-2)`	
		`(-2, 2, 0) ( 0,-2, 2) (-2, 0, 2)`	
		`(-2,-2, 0) ( 0,-2,-2) (-2, 0,-2)`	
D	24	`( 3, 1, 1) ( 1, 3, 1) ( 1, 1, 3)`	$\sqrt{11} \sim 3.32$
		`( 3, 1,-1) ( 1, 3,-1) ( 1,-1, 3)`	
		`( 3,-1, 1) (-1, 3, 1) (-1, 1, 3)`	
		`( 3,-1,-1) (-1, 3,-1) (-1,-1, 3)`	
		`(-3, 1, 1) ( 1,-3, 1) ( 1, 1,-3)`	
		`(-3, 1,-1) ( 1,-3,-1) ( 1,-1,-3)`	
		`(-3,-1, 1) (-1,-3, 1) (-1, 1,-3)`	
		`(-3,-1,-1) (-1,-3,-1) (-1,-1,-3)`	
E	24	`( 3, 3, 1) ( 1, 3, 3) ( 3, 1, 3)`	$\sqrt{19} \sim 4.36$
		`( 3, 3,-1) (-1, 3, 3) ( 3,-1, 3)`	
		`( 3,-3, 1) ( 1, 3,-3) (-3, 1, 3)`	
		`( 3,-3,-1) (-1, 3,-3) (-3,-1, 3)`	
		`(-3, 3, 1) ( 1,-3, 3) ( 3, 1,-3)`	
		`(-3, 3,-1) (-1,-3, 3) ( 3,-1,-3)`	
		`(-3,-3, 1) ( 1,-3,-3) (-3, 1,-3)`	
		`(-3,-3,-1) (-1,-3,-3) (-3,-1,-3)`	
F	24	`( 4, 2, 0) ( 2, 4, 0) ( 2, 0, 4)`	$\sqrt{20} \sim 4.47$
		`( 4,-2, 0) (-2, 4, 0) (-2, 0, 4)`	
		`( 4, 0, 2) ( 0, 4, 2) ( 0, 2, 4)`	
		`( 4, 0,-2) ( 0, 4,-2) ( 0,-2, 4)`	
		`(-4, 2, 0) ( 2,-4, 0) ( 2, 0,-4)`	
		`(-4,-2, 0) (-2,-4, 0) (-2, 0,-4)`	
		`(-4, 0, 2) ( 0,-4, 2) ( 0, 2,-4)`	
		`(-4, 0,-2) ( 0,-4,-2) ( 0,-2,-4)`	

Table 3.7: Lattice neighbour families – body centered cubic lattice.

fam	cnt	co-ordinates	distance
A	12	(1, 1, 0) (0, 1, 1) (1, 0, 1)	$\sqrt{2} \sim 1.41$
		(1,-1, 0) (0, 1,-1) (1, 0,-1)	
		(-1, 1, 0) (0,-1, 1) (-1, 0, 1)	
		(-1,-1, 0) (0,-1,-1) (-1, 0,-1)	
B	6	(2, 0, 0) (0, 2, 0) (0, 0, 2)	2
		(-2, 0, 0) (0,-2, 0) (0, 0,-2)	
C	24	(2, 1, 1) (1, 2, 1) (1, 1, 2)	$\sqrt{6} \sim 2.45$
		(2, 1,-1) (1, 2,-1) (1,-1, 2)	
		(2,-1, 1) (-1, 2, 1) (-1, 1, 2)	
		(2,-1,-1) (-1, 2,-1) (-1,-1, 2)	
		(-2, 1, 1) (1,-2, 1) (1, 1,-2)	
		(-2, 1,-1) (1,-2,-1) (1,-1,-2)	
		(-2,-1, 1) (-1,-2, 1) (-1, 1,-2)	
		(-2,-1,-1) (-1,-2,-1) (-1,-1,-2)	
D	24	(3, 1, 0) (1, 3, 0) (1, 0, 3)	$\sqrt{11} \sim 3.32$
		(3,-1, 0) (-1, 3, 0) (-1, 0, 3)	
		(3, 0, 1) (0, 3, 1) (0, 1, 3)	
		(3, 0,-1) (0, 3,-1) (0,-1, 3)	
		(-3, 1, 0) (1,-3, 0) (1, 0,-3)	
		(-3,-1, 0) (-1,-3, 0) (-1, 0,-3)	
		(-3, 0, 1) (0,-3, 1) (0, 1,-3)	
		(-3, 0,-1) (0,-3,-1) (0,-1,-3)	
E	8	(2, 2, 2) (2,-2, 2) (-2, 2, 2) (-2,-2, 2)	$\sqrt{12} \sim 3.46$
		(2, 2,-2) (2,-2,-2) (-2, 2,-2) (-2,-2,-2)	
F	48	(3, 2, 1) (3,-2, 1) (3, 2,-1) (3,-2,-1)	$\sqrt{14} \sim 3.74$
		(3, 1, 2) (3,-1, 2) (3, 1,-2) (3,-1,-2)	
		(2, 1, 3) (2,-1, 3) (2, 1,-3) (2,-1,-3)	
		(2, 3, 1) (2,-3, 1) (2, 3,-1) (2,-3,-1)	
		(1, 2, 3) (1,-2, 3) (1, 2,-3) (1,-2,-3)	
		(1, 3, 2) (1,-3, 2) (1, 3,-2) (1,-3,-2)	
		(-3, 2, 1) (-3,-2, 1) (-3, 2,-1) (-3,-2,-1)	
		(-3, 1, 2) (-3,-1, 2) (-3, 1,-2) (-3,-1,-2)	
		(-2, 1, 3) (-2,-1, 3) (-2, 1,-3) (-2,-1,-3)	
		(-2, 3, 1) (-2,-3, 1) (-2, 3,-1) (-2,-3,-1)	
		(-1, 2, 3) (-1,-2, 3) (-1, 2,-3) (-1,-2,-3)	
		(-1, 3, 2) (-1,-3, 2) (-1, 3,-2) (-1,-3,-2)	

Table 3.8: Lattice neighbour families – face centered cubic lattice.

index	neighbour	distance	total
1	A_1	1.00	5
2	B_1	1.41	9
3	A_2	2.00	13
4	C_1	2.24	21
5	B_2	2.83	25
6	A_3	3.00	29
7	D_1	3.16	37
8	E_1	3.61	45
9	A_4	4.00	49
10	F_1	4.12	57
11	B_3	4.24	61

Table 3.9: Ordered lattice neighbours - square lattice.

index	neighbour	distance	total
1	A_1	2.00	7
2	B_1	3.46	13
3	A_2	4.00	19
4	C_1	5.29	31
5	A_3	6.00	37
6	B_2	6.93	43
7	D_1	7.21	55

Table 3.10: Ordered lattice neighbours – hexagonal lattice.

index	neighbour	distance	total
1	A_1	1.00	7
2	B_1	1.41	19
3	C_1	1.73	27
4	A_2	2.00	33
5	D_1	2.24	57
6	E_1	2.45	81
7	B_2	2.83	93
8	A_3	3.00	99
9	F_1	3.00	123
10	G_1	3.16	147
11	H_1	3.32	171
12	C_2	3.46	179

Table 3.11: Ordered lattice neighbours – cubic lattice.

index	neighbour	distance	total
1	A_1	1.73	9
2	B_1	2.00	15
3	C_1	2.83	27
4	D_1	3.32	51
5	A_2	3.46	59
6	B_2	4.00	65
7	E_1	4.36	89
8	F_1	4.47	113

Table 3.12: Ordered lattice neighbours – body centered cubic lattice.

index	neighbour	distance	total
1	A_1	1.41	13
2	B_1	2.00	19
3	C_1	2.45	43
4	A_2	2.83	55
5	D_1	3.32	79
6	E_1	3.46	87
7	F_1	3.74	135

Table 3.13: Ordered lattice neighbours – face centered cubic lattice.

Lattice	Order	Directions	Magnitudes
`sqr_d2q9_ab`	2^{nd}	9	2
`sqr_d2q13_a11b`	3^{rd}	9	3
`sqr_d2q17_a11b11`	4^{th} reduced	9	4
`sqr_d2q17_a101b11`	4^{th} reduced	9	4
`sqr_d2q17_abc`	4^{th}	17	3
`sqr_d2q25_abcd`	4^{th}	25	4
`sqr_d2q25_a10bcd`	5^{th}	25	4
`sqr_d2q33_a10bcde`	6^{th}	33	5
`sqr_d2q53_a10111b111cdf`	7^{th}	33	10

Table 3.14: Various lattices and their order – square.

Lattice	Order	Directions	Magnitudes
`hex_d2q7_a`	2^{nd}	7	1
`hex_d2q13_ab`	4^{th} reduced	13	2
`hex_d2q25_abc`	5^{th}	25	3
`hex_d2q37_abcd`	6^{th}	37	4

Table 3.15: Various lattices and their order – hexagonal.

Lattice	Order	Directions	Magnitudes
`cub_d3q15_ac`	2^{nd}	15	2
`cub_d3q19_ab`	2^{nd}	19	2
`cub_d3q21_a11c`	2^{nd}	15	3
`cub_d3q21_bc`	2^{nd}	21	2
`cub_d3q27_abc`	2^{nd}	27	3
`cub_d3q27_a10bc`	4^{th} reduced	27	3
`cub_d3q39_a111bc`	4^{th} reduced	27	5
`cub_d3q39_a111b10c`	4^{th} reduced	27	5
`cub_d3q45_a11cd`	4^{th} reduced	39	4
`cub_d3q59_a111b11c11`	4^{th} reduced	27	7
`cub_d3q45_a11ce`	4^{th}	39	4
`cub_d3q51_abcd`	4^{th}	51	4
`cub_d3q75_abcde`	4^{th}	75	5
`cub_d3q75_a100bcde`	5^{th}	75	5
`cub_d3q99_a100bcdef`	5^{th}	99	5
`cub_d3q107_a1011b11c11dh`	5^{th}	75	9
`cub_d3q99_a100bcdfg`	6^{th}	99	5

Table 3.16: Various lattices and their order – cubic.

Lattice	Order	Directions	Magnitudes
`bcc_d3q15_ab`	2^{nd}	15	2
`bcc_d3q21_ac`	2^{nd}	21	2
`bcc_d3q27_abc`	4^{th} reduced	27	3
`bcc_d3q47_a11bd`	4^{th} reduced	39	4
`bcc_d3q53_a11cd`	4^{th} reduced	45	4
`bcc_d3q43_bcd`	4^{th}	43	3
`bcc_d3q45_ab11d`	4^{th}	39	4
`bcc_d3q49_b11cd`	4^{th}	43	4
`bcc_d3q53_a11b11d`	4^{th}	39	5
`bcc_d3q57_a10b11cd`	4^{th}	51	5
`bcc_d3q73_b11cdf`	4^{th}	67	5
`bcc_d3q69_ab11df`	5^{th}	63	5
`bcc_d3q93_ab11def`	5^{th}	87	6
`bcc_d3q99_abcdef`	6^{th}	99	6

Table 3.17: Various lattices and their order – body centered cubic.

Lattice	Order	Directions	Magnitudes
`fcc_d3q13_a`	2^{nd}	13	1
`fcc_d3q19_ab`	2^{nd}	19	2
`fcc_d3q27_abe`	4^{th} reduced	27	3
`fcc_d3q43_abc`	4^{th}	43	3
`fcc_d3q67_abcd`	4^{th}	67	4
`fcc_d3q75_abcde`	5^{th}	75	5
`fcc_d3q87_a101bcde`	5^{th}	75	6
`fcc_d3q93_a101b11cde`	5^{th}	75	7
`fcc_d3q99_a101b111cde`	5^{th}	75	8
`fcc_d3q99_a111bcde`	5^{th}	75	7
`fcc_d3q99_abcef`	5^{th}	99	5
`fcc_d3q99_abdef`	5^{th}	99	5
`fcc_d3q111_bcdef`	5^{th}	111	5
`fcc_d3q115_abcdf`	5^{th}	115	5
`fcc_d3q117_acdef`	5^{th}	117	5
`fcc_d3q123_abcdef`	6^{th}	123	6

Table 3.18: Various lattices and their order – face centered cubic.

3.6 The collision matrix

The analysis from the preceding sections assumed the single relaxation time collision operator to simplify the exposition;

$$\Omega_{\boldsymbol{i}}\left(x_\alpha,t\right)=\frac{1}{\tau}\big(f_{\boldsymbol{i}}\left(x_\alpha,t\right)-f_{\boldsymbol{i}}^{eq}\left(x_\alpha,t\right)\big). \tag{3.23}$$

The more general linearized collision operator;

$$\Omega_{\boldsymbol{i}}\left(x_\alpha,t\right)=\sum_{\boldsymbol{j}=0}^{q}A_{\boldsymbol{ij}}\big(f_{\boldsymbol{j}}\left(x_\alpha,t\right)-f_{\boldsymbol{j}}^{eq}\left(x_\alpha,t\right)\big), \tag{3.24}$$

may be used in its place so long as the following conditions are met: firstly, the zeroth order and first order lattice vectors should be left eigenvectors of the collision matrix.

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}}^{0} A_{\boldsymbol{ij}} = \lambda_0 c_{\boldsymbol{j}}^{0},$$
$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} A_{\boldsymbol{ij}} = \lambda_1 c_{\boldsymbol{j}\alpha}.$$

This will ensure conservation of mass and momentum.

$$\begin{aligned}\sum_{\boldsymbol{i}=0}^{q} \Omega_{\boldsymbol{i}} &= \sum_{\boldsymbol{i}=0}^{q}\sum_{\boldsymbol{j}=0}^{q} A_{\boldsymbol{ij}}\big(f_{\boldsymbol{j}}\left(x_\alpha, t\right) - f_{\boldsymbol{j}}^{eq}\left(x_\alpha, t\right)\big),\\ &= \lambda_0 \sum_{\boldsymbol{j}=0}^{q} \big(f_{\boldsymbol{j}}\left(x_\alpha, t\right) - f_{\boldsymbol{j}}^{eq}\left(x_\alpha, t\right)\big) = 0,\\ \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha}\Omega_{\boldsymbol{i}} &= \sum_{\boldsymbol{i}=0}^{q}\sum_{\boldsymbol{j}=0}^{q} c_{\boldsymbol{i}\alpha} A_{\boldsymbol{ij}}\big(f_{\boldsymbol{j}}\left(x_\alpha, t\right) - f_{\boldsymbol{j}}^{eq}\left(x_\alpha, t\right)\big),\\ &= \lambda_1 \sum_{\boldsymbol{j}=0}^{q} c_{\boldsymbol{j}\alpha}\big(f_{\boldsymbol{j}}\left(x_\alpha, t\right) - f_{\boldsymbol{j}}^{eq}\left(x_\alpha, t\right)\big) = 0.\end{aligned}$$

The first two sets of eigenvalues are not important. The second order lattice vectors should also be left eigenvectors of the collision matrix, they should all have the same eigenvalue and its value will determine the viscosity,

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} A_{\boldsymbol{ij}}^{-1} = \frac{1}{\lambda_2} c_{\boldsymbol{j}\alpha} c_{\boldsymbol{j}\beta}.$$

The first off-equilibrium term (Eqn. 3.9) now becomes,

$$\sum_{\boldsymbol{j}=0}^{q} A_{\boldsymbol{ij}} f_{\boldsymbol{j}}^{(1)} = \left(F_{\boldsymbol{i}} - \partial_{t_1} f_{\boldsymbol{i}}^{eq} - c_{\boldsymbol{i}\alpha}\partial_{\alpha_1} f_{\boldsymbol{i}}^{eq}\right),$$
$$f_{\boldsymbol{i}}^{(1)} = \sum_{\boldsymbol{j}=0}^{q} A_{\boldsymbol{ij}}^{-1}\left(F_{\boldsymbol{j}} - \partial_{t_1} f_{\boldsymbol{j}}^{eq} - c_{\boldsymbol{j}\alpha}\partial_{\alpha_1} f_{\boldsymbol{j}}^{eq}\right).$$

Substituting into Eqn. 3.11,

$$\partial_{t_2}\rho u_\alpha + \partial_\beta \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{(1)} = 0,$$

leads to,

$$\partial_{t_2}\rho u_\alpha + \partial_\beta \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} \sum_{j=0}^{q} A_{ij}^{-1} \left(F_j - \partial_{t_1} f_j^{eq} - c_{j\alpha}\partial_{\alpha_1} f_j^{eq}\right) = 0.$$

Substituting the previously mentioned eigenvalues results in a form identical to Eqn.3.12 however with $\frac{1}{\lambda_2}$ instead of τ,

$$\partial_{t_2}\rho u_\alpha = \frac{1}{\lambda_2}\partial_\beta \left\{ \partial_\gamma \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} c_{i\gamma} f_i^{eq} + \partial_{t_1} \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} f_i^{eq} - \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} F_i \right\}.$$

Thus the shear and bulk viscosities are,

$$\nu^s = \frac{RT}{\lambda_2} \quad \text{and} \quad \nu^v = \frac{2}{3}\nu^s.$$

Similarly, the eigenvalues corresponding to the rank 3 lattice vectors λ_3 affect the thermal diffusivity in Eqn. 3.14,

$$\partial_{t_2}\rho e + \frac{1}{2}\partial_\alpha \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} c_{i\beta} f_i^{(1)} = 0,$$

$$\partial_{t_2}\rho e + \frac{1}{2}\partial_\alpha \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} c_{i\beta} \sum_{j=0}^{q} A_{ij}^{-1} \left(F_j - \partial_{t_1} f_j^{eq} - c_{j\alpha}\partial_{\alpha_1} f_j^{eq}\right) = 0,$$

$$\partial_{t_2}\rho e = \frac{1}{2\lambda_3}\partial_\alpha \left\{ \partial_{t_1} \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} c_{i\beta} f^{eq} + \partial_\gamma \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} c_{i\beta} c_{i\gamma} f^{eq} - \sum_{i=0}^{q} c_{i\alpha} c_{i\beta} c_{i\beta} F_i \right\}.$$

As well as controlling the thermal diffusivity, λ_3 also controls the coefficient for viscous heating. This introduces a source of error if λ_3 is set differently to λ_2 in an effort to change the Prandtl number as proposed by some earlier works [133, 134, 211]. Zheng et al. [266] have remedied the problem by combining the modified equilibrium moments of Guo et al. [96] with multiple relaxation times in order to achieve a variable Prandtl number scheme. Machado [148] goes into considerable additional detail regarding equilibrium distributions for a variable Prandtl number.

An interesting feature of the collision matrix is that the coefficients may only depend on the angle between velocities if the solution is to be isotropic, for example, Ginzbuorg and Adler [86] show that there can only be 5 unknowns in the collision matrix, corresponding to collisions between particles separated by 60, 90, 120 and 180 degrees. Some of the early works on lattice Boltzmann methods [109, 108] also utilized this constraint.

3.6.1 Eigendecomposition

In the preceding section we assumed that the eigenvalues of the inverse of the collision matrix could be set to prescribed values. The purpose of this section is to demonstrate the method we have developed in order to achieve this. We will briefly summarize some concepts in order to establish nomenclature.

The motivation for this is that the collision operator used in the MRT method [133, 62, 63] is typically written as,

$$\begin{aligned}\Omega_i &= A_{ij}\left(f_j - f_j^{eq}\right), \\ &= \left(M^{-1}SM\right)_{ij}\left(f_j - f_j^{eq}\right), \\ &= \left(M^T T^{-1} S\right)_{ij}\left(m_j - m_j^{eq}\right).\end{aligned}$$

Here the moments m_i of the particle probability distribution function f_i are obtained by multiplication by the M_{ij} matrix,

$$m_i = M_{ij} f_j.$$

As mentioned in section 3.2, the equilibrium particle distribution f^{eq} does not need to be calculated, only its moments do and using the moments directly also avoids one matrix multiplication. As we will show the S matrix is a diagonal matrix of eigenvalues. It is important to emphasize that the method in the following section is different to the usual MRT procedure even though it is based upon it. Our method is more generalized and allows completely arbitrary adjustment of eigenvalues and eigenvectors. Our method also offers significant performance improvements due to the sparse decomposition we introduce.

An orthogonal matrix is an invertible square matrix made using an orthonormal set of vectors. For such a matrix, the inverse and transpose are equal and both are also orthonormal matrices. If the rows of a matrix form an orthonormal set so do the column vectors,

$$B^T B = I, \quad \Rightarrow \quad B^T = B^{-1}.$$

In our case, we make use of a related type of matrix whose rows form an orthogonal set which has not been normalized,

$$MM^T = T, \quad \Rightarrow \quad M^T = M^{-1}T.$$

In this case, T is a diagonal matrix. The inverse of this type of matrix is also easy to find based on the transpose and is typically used in the implementation of the MRT method.

The Gram-Schmidt procedure is used to create an orthogonal set from the linearly independent set. It takes each vector $v_{\boldsymbol{i}}$ in a set and makes a corresponding vector $u_{\boldsymbol{i}}$ that is orthogonal to all previous vectors,

$$u_{\boldsymbol{i}\alpha} = v_{\boldsymbol{i}\alpha} - \sum_{\boldsymbol{j}=1}^{i-1} \text{proj}_{u_{\boldsymbol{j}\alpha}}(v_{\boldsymbol{i}\alpha}).$$

Here the projection operator gives a vector whose magnitude is a dot product between v_α and the unit vector $\hat{u}_\alpha$ and whose direction is also $\hat{u}_\alpha$,

$$\text{proj}_{u_\alpha}(v_\alpha) = v_\beta \hat{u}_\beta \hat{u}_\alpha = \frac{u_\beta v_\beta}{u_\gamma u_\gamma} u_\alpha, \quad \text{where} \quad \hat{u}_\alpha = \frac{u_\alpha}{\sqrt{u_\beta u_\beta}}.$$

I have developed a program called `lb_moment` which takes an input set of vectors, grouped into a matrix N and performs the Gram-Schmidt procedure finding a triangular transformation matrix Q which when multiplied by the original matrix forms an orthogonal set, grouped into the matrix M.

$$M = Q \cdot N$$

During program execution, if any resulting basis vector $v_{i\alpha}$ is zero, then this means that its original vector u_i was not linearly independent relative to the previous vectors, useful for checking the maximum order supported by a lattice as we discussed in previous sections.

A square matrix A with dimension q that has q distinct left eigenvectors $\mathbf{n}_i$ with eigenvalues s_i can be written in a form called the eigendecomposition,

$$A = N^{-1} S N.$$

Here N is a matrix whose rows are left eigenvectors $\mathbf{n}_i$,

$$N = \begin{pmatrix} \mathbf{n}_1 \\ \mathbf{n}_2 \\ \vdots \\ \mathbf{n}_q \end{pmatrix}$$

and S is a diagonal matrix of left eigenvalues s_i,

$$S = \begin{pmatrix} s_1 & 0 & \dots & 0 \\ 0 & s_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & s_q \end{pmatrix}.$$

This can be verified by looking at the rows,

$$NA = \begin{pmatrix} \mathbf{n}_1 A \\ \mathbf{n}_2 A \\ \vdots \\ \mathbf{n}_q A \end{pmatrix} = \begin{pmatrix} s_1 \mathbf{n}_1 \\ s_2 \mathbf{n}_2 \\ \vdots \\ s_q \mathbf{n}_q \end{pmatrix} = SN.$$

Similarly for right eigenvectors,

$$A = \mathring{N}\mathring{S}\mathring{N}^{-1}.$$

Where $\mathring{N}$ is a matrix whose columns are right eigenvectors $\mathring{\mathbf{n}}_{\boldsymbol{i}}$,

$$\mathring{N} = \left(\mathring{\mathbf{n}}_1 \middle| \mathring{\mathbf{n}}_2 \middle| \dots \middle| \mathring{\mathbf{n}}_q\right).$$

The diagonal matrix $\mathring{S}$ consists of right eigenvalues $\mathring{s}_{\boldsymbol{i}}$ as before,

$$A\mathring{N} = \left(A\mathring{\mathbf{n}}_1 \middle| A\mathring{\mathbf{n}}_2 \middle| \dots \middle| A\mathring{\mathbf{n}}_q\right) = \left(s_1\mathring{\mathbf{n}}_1 \middle| s_2\mathring{\mathbf{n}}_2 \middle| \dots \middle| s_q\mathring{\mathbf{n}}_q\right) = \mathring{N}\mathring{S}.$$

The lattice Boltzmann method requires setting left eigenvalues and eigenvectors of the inverse collision matrix,

$$A^{-1} = N^{-1}S^{-1}N,$$
$$A = N^{-1}SN.$$

If the eigenvectors are also orthogonal with $N \cdot N^T = T$ then,

$$A^{-1} = N^T T^{-1} S^{-1} N,$$
$$A = N^T T^{-1} S N.$$

This is the form typically used in order to implement MRT methods. Eigenvalues corresponding to each of the orthogonal eigenvectors may be set individually in contrast to the single relaxation time method where all eigenvalues are equal and

$$A = \omega I.$$

A disadvantage of this method is the lack of freedom in choosing eigenvectors imposed by the orthogonal requirement. I have made a simple yet novel and extremely useful modification that not only allows the choice of any eigenvectors without any additional complexity, it also allows the action of the collision operator to take on a more general form where lattice vectors can be transformed into any linear combination of other lattice vectors.

Consider the orthogonal matrix M obtained using the Gram-Schmidt procedure. Usually MRT methods use these as the eigenvectors of the collision matrix. Instead, we perform an eigendecomposition based on the input vectors N and use the orthogonal matrix and the transformation matrix in order to arrive at a convenient expression for the inverse of N.

$$\begin{aligned} M &= QN, \\ N &= Q^{-1}M, \\ N^{-1} &= M^{-1}Q = M^T T^{-1} Q = N^T Q^T T^{-1} Q. \end{aligned}$$

We can substitute this definition for N^{-1} into the collision matrix,

$$A = N^T Q^T T^{-1} QSN.$$

Although there are additional matrix multiplication present, in practice there is a performance improvement if these matrix multiplications are expressed in a sparse form. We have gone one step further and decomposed N into two sparser matrices,

$$N = N_0 N_q.$$

Even for conventional MRT methods we have demonstrated performance improve-

Lattice	Dense	Orthogonal	Factorized
`sqr_d2q9_ab`	162	77	64
`fcc_d3q13_a`	338	123	78
`cub_d3q15_ac`	450		
`bcc_d3q15_ab`	450	191	121
`cub_d3q19_ab`	722		
`cub_d3q27_abc`	1,458	705	274
`bcc_d3q27_abc`	1,458	737	331

Table 3.19: Number of operations required to calculate the collision matrix for lattices considered in this thesis.

ments using a sparse factorization with the savings in floating point operations shown in Table 3.19 which lists the number of operations required in order to implement a dense matrix, the orthogonal matrix and our factorization for each of the lattices that appear in this thesis. A naive implementation would calculate the inverse A matrix directly taking q^2 multiplications and the same number of additions. Implementation details are presented for a range of lattices in Section 3.A.

For example, using the `cub_d3q27_abc` lattice, the total floating point calculations can be reduced from $1,458$ for a dense matrix multiplication down to 274 with our factorization. Apart from the increase in computational efficiency, the lower number of factors may also result in lower rounding errors accumulating per time step which is an ongoing concern in the implementation of LB methods where large and small numbers orders of magnitude apart need to be summed.

A further optimization along these lines is hard coding the factors into the program at compile time rather than reading the matrix from memory. Although the matrix will most likely be cached well by modern CPUs, hard coding the operations can offer advantages for GPUs and eliminates branching and indexing of the matrix.

In some cases it is not desirable to express the action of the collision matrix in terms of eigenvectors. Instead, a more general form transforms a set of vectors into another linear combination of the same vectors. The most general such matrix is,

$$\begin{pmatrix} B_{0,0} & B_{0,1} & \dots & B_{0,q} \\ B_{1,0} & B_{1,1} & \dots & B_{1,q} \\ \vdots & \vdots & \ddots & \vdots \\ B_{q,0} & B_{q,1} & \dots & B_{q,q} \end{pmatrix} \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_q \end{pmatrix} = \begin{pmatrix} B_{0,0}\mathbf{c}_0 + B_{0,1}\mathbf{c}_1 + \dots + B_{0,q}\mathbf{c}_q \\ B_{1,0}\mathbf{c}_0 + B_{1,1}\mathbf{c}_1 + \dots + B_{1,q}\mathbf{c}_q \\ \vdots \\ B_{q,0}\mathbf{c}_0 + B_{q,1}\mathbf{c}_1 + \dots + B_{q,q}\mathbf{c}_q \end{pmatrix}.$$

3.6.2 Bulk viscosity

One application where the individual adjustment of eigenvalues and vectors is useful is when the bulk viscosity needs to be adjusted. Although the bulk viscosity is not expected to significantly influence the solution of low Mach number simulations as presented in the results of this thesis, it may affect the stability of these simulations and is important in compressible simulations which are an intended application of the lattices and multiple relaxation time methods presented in this Chapter. Using the eigendecomposition we have developed it is possible to tune the bulk viscosity in a way that is not possible using other multiple relaxation time formulations. In this case the action of the collision matrix on the rank two lattice vectors needs to take the following form,

$$c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} A^{-1}_{\boldsymbol{ij}} = \tau_2 c_{\boldsymbol{j}\alpha} c_{\boldsymbol{j}\beta} + \tau'_2 \delta_{\alpha\beta} \delta_{\gamma\delta} c_{\boldsymbol{j}\gamma} c_{\boldsymbol{j}\delta}. \tag{3.25}$$

Here we use the notation that τ is the inverse of the eigenvalue λ. The off-equilibrium momentum equations then become,

$$\partial_{t_2} \rho u_\alpha = \tau_2 RT \partial_\beta \left(\rho \partial_\alpha u_\beta + \rho \partial_\beta u_\alpha \right) + \tau'_2 RT \partial_\alpha \left(\rho \partial_\gamma u_\delta + \rho \partial_\delta u_\gamma \right),$$
$$\partial_{t_2} \rho u_\alpha = \tau_2 RT \partial_\beta \left(\rho \partial_\alpha u_\beta + \rho \partial_\beta u_\alpha \right) + 2\tau'_2 RT \partial_\alpha \rho \partial_\beta u_\beta.$$

The additional term proportional to τ_2' is the same form as the second viscosity, a term which is often zero (Sec. 2.6). The second viscosity is,

$$\nu^0 = 2RT\tau_2'.$$

Bulk viscosity is related to the second viscosity,

$$\begin{aligned}\nu^v &= \nu^0 + \frac{2}{3}\nu^s, \\ &= 2RT\left(\tau_2' + \frac{\tau_2}{3}\right).\end{aligned}$$

This method does not produce a corresponding change in the viscous heating term of the energy equation.

In matrix form, Eqn. 3.25 can be written as,

$$\begin{pmatrix} \vdots \\ c_xc_x \\ c_yc_y \\ c_zc_z \\ c_xc_y \\ c_xc_z \\ c_yc_z \\ \vdots \end{pmatrix} \cdot A^{-1} = \begin{pmatrix} \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \iddots \\ \dots & (\tau_2+\tau_2') & \tau_2' & \tau_2' & 0 & 0 & 0 & \dots \\ \dots & \tau_2' & (\tau_2+\tau_2') & \tau_2' & 0 & 0 & 0 & \dots \\ \dots & \tau_2' & \tau_2' & (\tau_2+\tau_2') & 0 & 0 & 0 & \dots \\ \dots & 0 & 0 & 0 & \tau_2 & 0 & 0 & \dots \\ \dots & 0 & 0 & 0 & 0 & \tau_2 & 0 & \dots \\ \dots & 0 & 0 & 0 & 0 & 0 & \tau_3 & \dots \\ \iddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \vdots \\ c_xc_x \\ c_yc_y \\ c_zc_z \\ c_xc_y \\ c_xc_z \\ c_yc_z \\ \vdots \end{pmatrix}.$$

This relationship already has a similar form to the previously described eigendecomposition,

$$\begin{aligned}CA^{-1} &= BC, \\ A^{-1} &= C^{-1}BC, \\ A &= C^{-1}B^{-1}C.\end{aligned}$$

The C matrix and its inverse are dealt with in the usual way using an orthogonal matrix transformation. The only difference in this case is that instead of a diagonal matrix, the matrix B is a block diagonal matrix,

$$B = \begin{pmatrix} B_1 & 0 & \dots & 0 \\ 0 & B_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & B_q \end{pmatrix}, \quad B^{-1} = \begin{pmatrix} B_1^{-1} & 0 & \dots & 0 \\ 0 & B_2^{-1} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & B_q^{-1} \end{pmatrix}.$$

Almost all of the blocks are diagonal and hence trivial to invert except for B_2 which is inverted using Gauss-Jordan elimination,

$$B_2 = \begin{pmatrix} (\tau_0 + \tau_1) & \tau_1 & \tau_1 \\ \tau_1 & (\tau_0 + \tau_1) & \tau_1 \\ \tau_1 & \tau_1 & (\tau_0 + \tau_1) \end{pmatrix},$$

$$B_2^{-1} = \frac{1}{\tau_0 (\tau_0 + 3\tau_1)} \begin{pmatrix} (\tau_0 + 2\tau_1) & -\tau_1 & -\tau_1 \\ -\tau_1 & (\tau_0 + 2\tau_1) & -\tau_1 \\ -\tau_1 & -\tau_1 & (\tau_0 + 2\tau_1) \end{pmatrix}.$$

3.7 Linear stability analysis

The stability of lattice Boltzmann methods has been the subject of considerable research. Sterling and Chen [231] describe a method of linear analysis to determine stability. Lallemand and Luo [133] use the technique to analyse the accuracy of their schemes as well as stability. Banda, Yong and Klar [10] use a similar method to develop a stability notion which they then show is satisfied for a range of commonly used schemes. Their stability structure is further developed by Rheinlaender [197]. Entropy concerns are also critical to the stability of lattice Boltzmann methods, they are used by Chikatamarla and Karlin [42] to determine stable lattices noting their lack of reliance on Gauss-Hermite quadrature. Brownlee, Gorban and Levesley [20] present a mathematically rigorous treatment of the stability of lattice Boltzmann schemes including entropic concerns among other things in their analysis. They identify several mechanisms for instability and provide methods of stabilization. We use basic linear analysis because it has accurately predicted stability issues that we have encountered.

We begin with the first order upwind time advancement with variable Courant number.

$$f_{\boldsymbol{i}}(x_\alpha + \Delta x_\alpha, t + \Delta t) = (1 - \mathrm{Cr}) f_{\boldsymbol{i}}(x_\alpha + \Delta x_\alpha, t) + \mathrm{Cr}\, f_{\boldsymbol{i}}\left(x_\alpha, t\right) + \Delta t \Omega_{\boldsymbol{i}}\left(x_\alpha, t\right).$$

Ω is the linearised collision operator,

$$\Omega_{\boldsymbol{i}}\left(x_\alpha, t\right) = \sum_{\boldsymbol{j}=0}^{q} A_{\boldsymbol{ij}} \left(f_{\boldsymbol{j}}\left(x_\alpha, t\right) - f_{\boldsymbol{j}}^{eq}\left(x_\alpha, t\right)\right).$$

The overall aim of the analysis is to form a linear recurrence relation describing the evolution of the flow field subject to some disturbance to the mean uniform flow. The method is then assumed to be stable if the recurrence relation remains bounded as time advances. We have not analysed our modified time stepping scheme since the inertia like correction term introduces a second time level into the recurrence relation complicating analysis. We seek to determine the stability boundary as the mean background velocity is increased.

The linearised collision operator Ω represents the effects of collisions between particles as a linear combination of other particles ($f_{\boldsymbol{i}}$) and their equilibrium values ($f_{\boldsymbol{i}}^{eq}$). This operator is still not suitable for the formation of a linear recurrence relation since the equilibrium values themselves ($f_{\boldsymbol{i}}^{eq}$) are non-linear in $f_{\boldsymbol{i}}$.

In order to proceed we need an expression for Ω that is linear in $f_{\boldsymbol{i}}$ so we split the particle distribution functions into a mean ($\bar{f}_{\boldsymbol{i}}$) and perturbation ($f'_{\boldsymbol{i}}$) component,

$$f_{\boldsymbol{i}} = \bar{f}_{\boldsymbol{i}} + f'_{\boldsymbol{i}},$$

then perform a Taylor expansion about the mean flow,

$$\Omega_{\boldsymbol{i}}\left(f_{\boldsymbol{j}}\right)=\Omega_{\boldsymbol{i}}\left(\bar{f}_{\boldsymbol{j}}\right)+\sum_{\boldsymbol{k}=0}^{q} f_{\boldsymbol{k}}^{\prime}\cdot\frac{\partial\Omega_{\boldsymbol{i}}}{\partial f_{\boldsymbol{k}}}\left(\bar{f}_{\boldsymbol{j}}\right)+\mathcal{O}\left({f_{\boldsymbol{k}}^{\prime}}^{2}\right).$$

Now since $\bar{f}_{\boldsymbol{i}}$ is a constant, so is the Jacobian matrix J,

$$J_{\boldsymbol{ik}}=\frac{\partial\Omega_{\boldsymbol{i}}}{\partial f_{\boldsymbol{k}}}\left(\bar{f}_{\boldsymbol{j}}\right),\quad \Omega_{\boldsymbol{i}}\left(f_{\boldsymbol{j}}\right)\sim f_{\boldsymbol{k}}^{\prime}\cdot J_{\boldsymbol{ik}}.$$

We transform to Fourier space using the notation that $\hat{f}(\xi)=\mathcal{F}\{f(x)\}$ and use the translation property,

$$h(x_\alpha)=f(x_\alpha-r_\alpha)\quad\Rightarrow\quad\hat{h}(\xi_\alpha)=e^{-2\pi i r_\alpha\xi_\alpha}\hat{f}(\xi_\alpha).$$

We introduce a new translation matrix T,

$$T_{\boldsymbol{ij}}(\xi_\alpha)=\begin{cases}e^{-2\pi i\Delta x_{\boldsymbol{i}\alpha}\xi_\alpha} & \text{if } \boldsymbol{i}=\boldsymbol{j}\\ 0 & \text{otherwise}\end{cases}$$

so that

$$\mathcal{F}\left(f_{\boldsymbol{i}}\left(x_\alpha+\Delta x_\alpha,t+\Delta t\right)\right)=\hat{f}_{\boldsymbol{i}}\left(\xi_\alpha,t+\Delta t\right)\cdot T_{\boldsymbol{ij}}\left(\xi_\alpha\right).$$

The evolution equation reduces to,

$$\begin{aligned}&\hat{f}_{\boldsymbol{i}}\left(\xi_\alpha,t+\Delta t\right)\cdot T_{\boldsymbol{ij}}\left(\xi_\alpha\right)\\ &\qquad=\left(1-\mathrm{Cr}\right)\hat{f}_{\boldsymbol{i}}\left(\xi_\alpha,t\right)\cdot T_{\boldsymbol{ij}}\left(\xi_\alpha\right)+\mathrm{Cr}\,\hat{f}_{\boldsymbol{i}}\left(\xi_\alpha,t\right)+\Delta t\hat{f}_{\boldsymbol{j}}^{\prime}\left(\xi_\alpha,t\right)\cdot J_{\boldsymbol{ij}}.\end{aligned}$$

Splitting $\hat{f}_{\boldsymbol{i}}$ into its flow and perturbation components, all of the flow components cancel out and we are left with,

$$\hat{f}_{\boldsymbol{i}}^{\prime}\left(\xi_\alpha,t+\Delta t\right)=\hat{f}_{\boldsymbol{i}}^{\prime}\left(\xi_\alpha,t\right)\cdot\left[\left(1-\mathrm{Cr}\right)I_{\boldsymbol{ij}}+\mathrm{Cr}\,T_{\boldsymbol{ij}}^{-1}\left(\xi_\alpha\right)+\Delta tJ_{\boldsymbol{ik}}T_{\boldsymbol{kj}}^{-1}\left(\xi_\alpha\right)\right].$$

This is now a matrix recurrence relation of the type,

$$x_{\boldsymbol{i}}^{n+1} = A_{\boldsymbol{ij}} \cdot x_{\boldsymbol{i}}^{n} \quad \text{with solution} \quad x_{\boldsymbol{i}}^{n} = A_{\boldsymbol{ij}}^{n} \cdot x_{\boldsymbol{i}}^{0}$$

We will use the criteria that $A_{\boldsymbol{ij}}^{n}$ remains bounded as n grows as a measure of stability. Expressing $A_{\boldsymbol{ij}}^{n}$ in terms of its eigendecomposition,

$$A^{n} = PD^{n}P^{-1}.$$

An equivalent condition is that D^{n} remains bounded. Since D is a diagonal matrix comprised of the eigenvalues $z_{\boldsymbol{i}}$ of A, we require that each eigenvalue remains bounded when raised to an arbitrary power.

$$\begin{aligned} &\lim_{n\to\infty} z_{\boldsymbol{i}}^{n} = \lim_{n\to\infty} \exp\left(n \log\left(z_{\boldsymbol{i}}\right)\right) < \infty \\ &\Rightarrow \operatorname{Re}\left(\log\left(z_{\boldsymbol{i}}\right)\right) < 0 \end{aligned}$$

This means that the real component the logarithm of each of the eigenvalues must be less than zero.

All that remains is to calculate the Jacobian matrix for the LB scheme under consideration. When using the generalised LB method, it is easier to work in the moment space,

$$m_{\boldsymbol{i}} = \sum_{\boldsymbol{j}=0}^{q} M_{\boldsymbol{ij}} f_{\boldsymbol{j}}.$$

Derivatives are redefined,

$$\begin{aligned} \frac{\partial}{\partial f_{\boldsymbol{i}}} &= \sum_{\boldsymbol{j}=0}^{q} \frac{\partial}{\partial m_{\boldsymbol{j}}} \frac{\partial m_{\boldsymbol{j}}}{\partial f_{\boldsymbol{i}}}, \\ &= \sum_{\boldsymbol{j}=0}^{q} \frac{\partial}{\partial m_{\boldsymbol{j}}} M_{\boldsymbol{ji}}. \end{aligned}$$

Thus the Jacobian matrix is then calculated,

$$J_{\boldsymbol{ij}} = \frac{\partial \Omega_{\boldsymbol{i}}}{\partial m_{\boldsymbol{k}}} \left(\bar{f}_{\boldsymbol{l}}\right) M_{\boldsymbol{kj}},$$

With the collision operator in the following form,

$$\begin{aligned} \Omega_{\boldsymbol{i}} \left(f_{\boldsymbol{o}}\right) &= M_{\boldsymbol{il}}^{-1} S_{\boldsymbol{lm}} M_{\boldsymbol{mn}}^{-1} \left[f_{\boldsymbol{n}} - f_{\boldsymbol{n}}^{eq} \left(f_{\boldsymbol{o}}\right)\right], \\ \Omega_{\boldsymbol{i}} \left(m_{\boldsymbol{n}}\right) &= M_{\boldsymbol{il}}^{-1} S_{\boldsymbol{lm}} \left[m_{\boldsymbol{m}} - m_{\boldsymbol{m}}^{eq} \left(m_{\boldsymbol{n}}\right)\right], \end{aligned}$$

the Jacobian matrix is,

$$\begin{aligned} J_{\boldsymbol{ij}} &= M_{\boldsymbol{il}}^{-1} S_{\boldsymbol{lm}} \frac{\partial}{\partial m_{\boldsymbol{k}}} \left[m_{\boldsymbol{m}} - m_{\boldsymbol{m}}^{eq}\right] \left(\bar{m}_{\boldsymbol{n}}\right) M_{\boldsymbol{kj}}, \\ &= M_{\boldsymbol{il}}^{-1} S_{\boldsymbol{lm}} \left[I_{\boldsymbol{mk}} - \frac{\partial m_{\boldsymbol{m}}^{eq}}{\partial m_{\boldsymbol{k}}} \left(\bar{m}_{\boldsymbol{n}}\right)\right] M_{\boldsymbol{kj}}. \end{aligned}$$

We confine the derivatives to the matrix,

$$C_{\boldsymbol{mk}} \left(\bar{m}_{\boldsymbol{n}}\right) = \frac{\partial m_{\boldsymbol{m}}^{eq}}{\partial m_{\boldsymbol{k}}} \left(\bar{m}_{\boldsymbol{n}}\right).$$

Now the matrix A from the recurrence relation is,

$$(1 - \mathrm{Cr})\, I_{\boldsymbol{ij}} + \mathrm{Cr}\, T_{\boldsymbol{ij}}^{-1} \left(\xi_\alpha\right) + \Delta t M_{\boldsymbol{il}}^{-1} S_{\boldsymbol{lm}} \left[I_{\boldsymbol{mn}} - C_{\boldsymbol{mn}} \left(m_{\boldsymbol{o}}^{flow}\right)\right] M_{\boldsymbol{nk}} T_{\boldsymbol{kj}}^{-1} \left(\xi_\alpha\right).$$

We need to make sure that the real part of the logarithm of each of the eigenvalues of this matrix is less than zero to guarantee stability. The matrix depends on the wavenumber of the perturbation ξ_α (the inverse of the wavelength) and the moments of the mean flow $\bar{m}_{\boldsymbol{i}}$.

The eigenvalues depend strongly on the wavenumber of the perturbation and on the mean background velocity. Shorter wavelengths and larger background velocities being less stable. These results agree qualitatively with our observations during simulations which show the development of instabilities at a particular dominant frequency in regions with a large mean flow velocity. We search a three dimensional

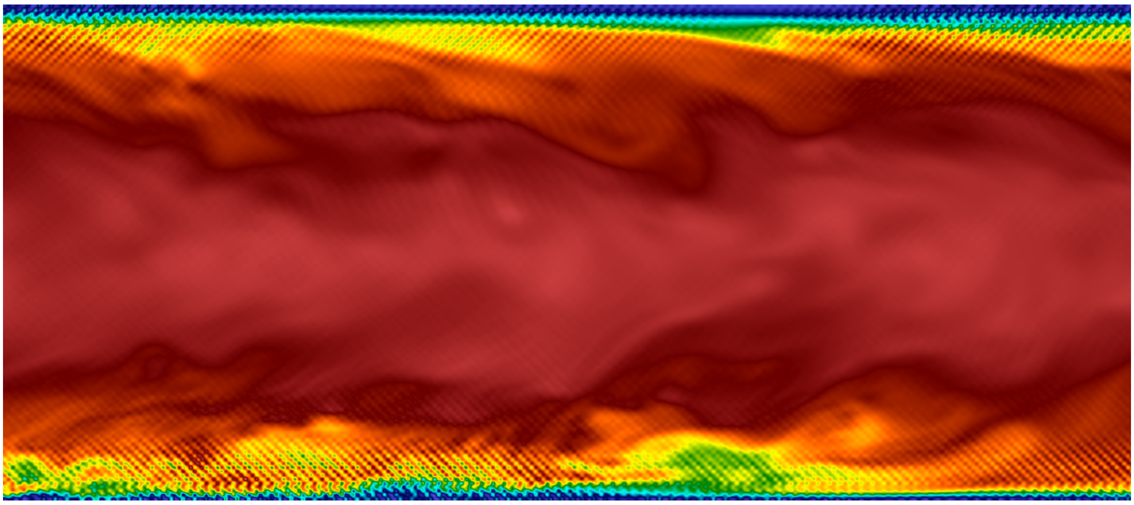


Figure 3.15: Channel flow instability using `bcc_d3q27_abc` lattice, $\omega = 1.9964$ and $u_{\max} = 0.058$ prior to blowup.

parameter space for the least stable wavenumber during each test. While earlier work by Lallemand and Luo [133] restricted their search to directions aligned with the flow, our results indicate that the least stable perturbation is rarely aligned with the flow direction. This corresponds well with observations of unstable simulations, with an example shown in Figure 3.15 which show patterns at an angle to the mean flow field. The simulation shown in this figure used the single relaxation time method, it was initially expected that the MRT method could be used to improve the stability properties of simulations however we cannot report any success in this area at this stage. We anticipate that further work exploring the parameter space in a more rigorous manner may yield improved stability.

I have developed a FORTRAN program called `lb_pert` in order to perform linear stability analysis on the lattices presented in this thesis. The program has several modes of operation, the results obtained in this section are the result of sweeping over a range of relaxation factors ω and determining the maximum stable background velocity at each data point. We use a bracketing approach to determine the maximum velocity. Each iteration begins with an interval that consists of one stable and one unstable velocity, the boundary lies somewhere in that interval. The midpoint of the interval is then tested and depending on the result a new interval of

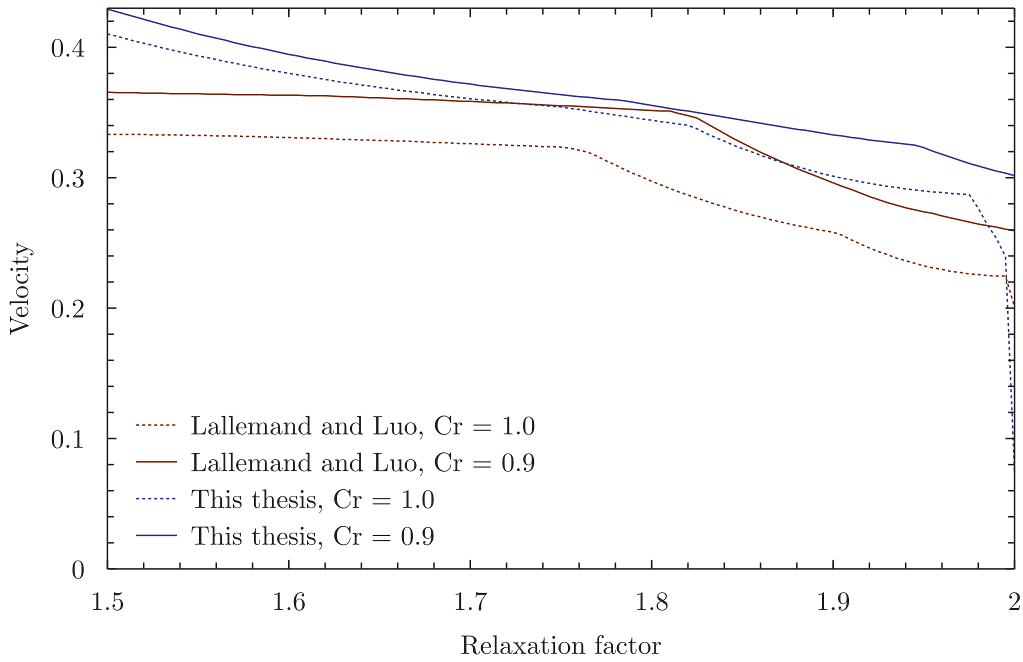


Figure 3.16: Stability boundary for `sqr_d2q9_ab` lattice.

half the size is constructed. The results below use 10 such bracketing steps in order to achieve a sufficient resolution of the boundary. We have run our tests searching integer wavelengths in the range $2, 3, 4, \ldots 20$, typical wavelengths are between two and five. The largest wavelength observed in any test has been 16 giving us confidence in our choice to restrict the search to wavelengths below 20. Increasing the number of wavelengths to search requires an $\mathcal{O}\left(n^3\right)$ increase in computational time.

Figure 3.16 shows the stability boundary for the `sqr_d2q9_ab` lattice, in agreement with practical experience, the maximum stable velocity reduces to zero as the relaxation factor approaches two, the value corresponding to zero viscosity. The figure shows the increased stability of our moment system compared to the more common system found in the original paper by Lallemand and Luo [133]. Both moments systems use the same lattice and result in the same moments up to third order, the differences lie in the fourth order and higher moments. This illustrates the important effect of higher order moments on the stability of the lattice. These higher order effects are not explained by the second order multi-scale expansion. The figure shows results using a Courant number of one, the usual Lagrangian streaming

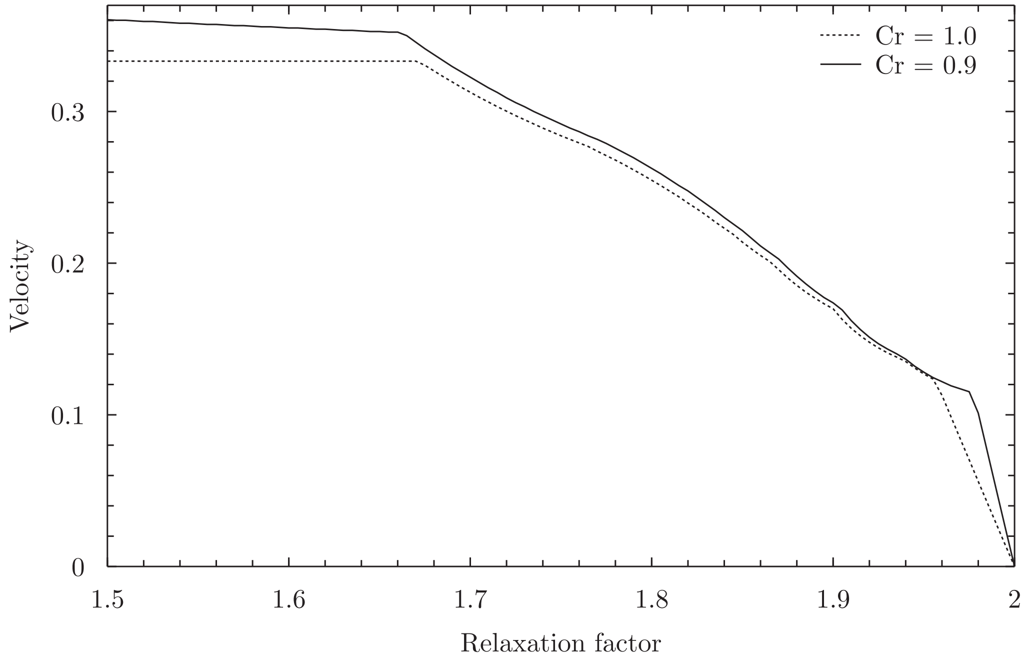


Figure 3.17: Stability boundary for `fcc_d3q13_a` lattice.

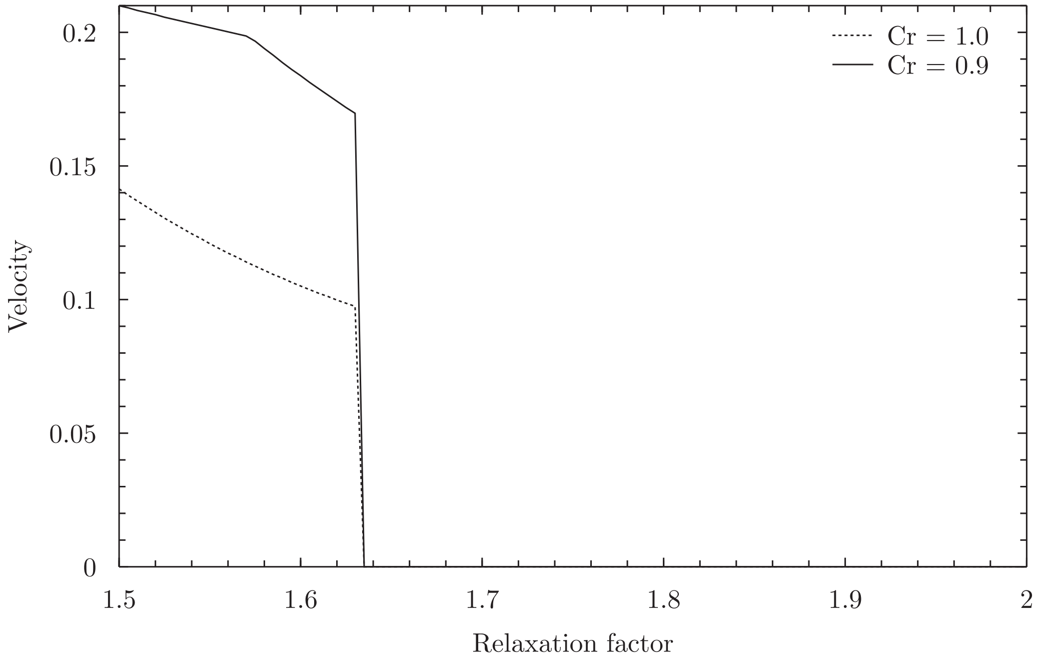


Figure 3.18: Stability boundary for `cub_d3q15_ac` lattice.

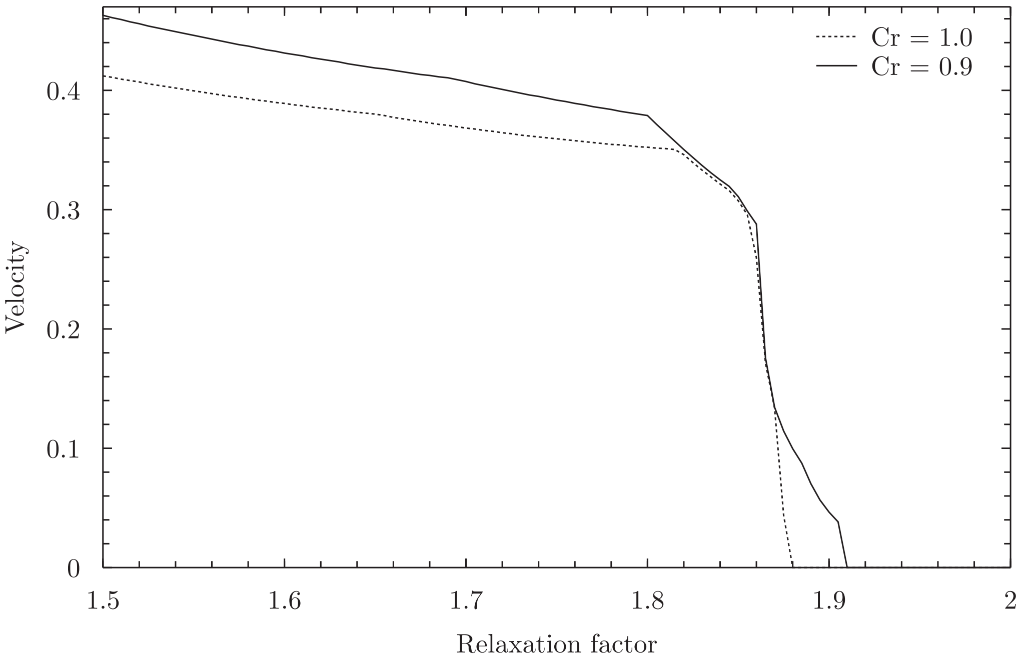


Figure 3.19: Stability boundary for `bcc_d3q15_ab` lattice.

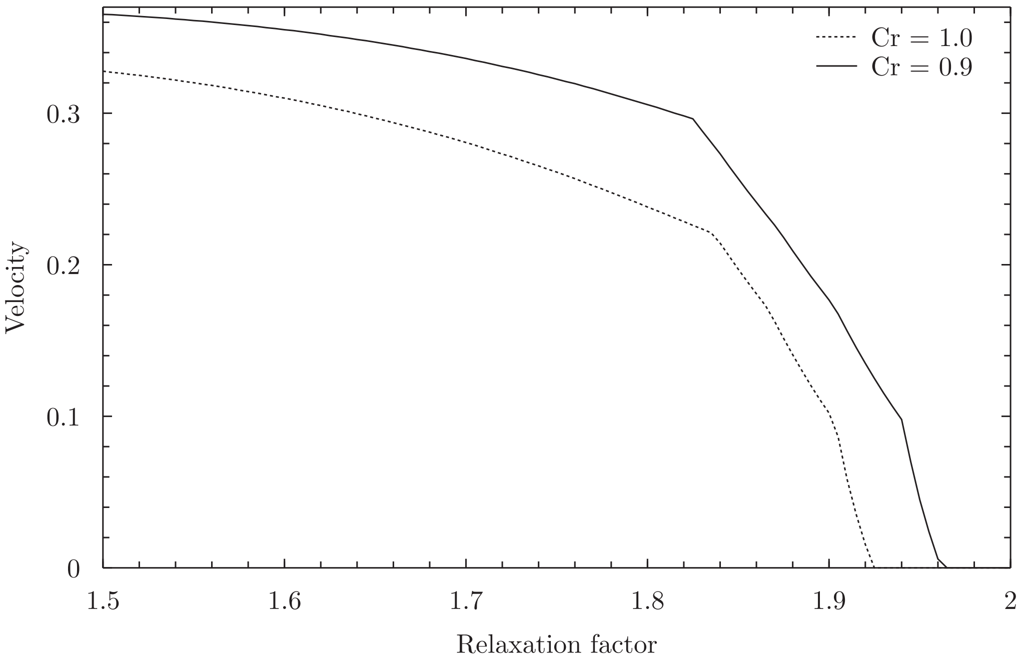


Figure 3.20: Stability boundary for `cub_d3q19_ab` lattice.

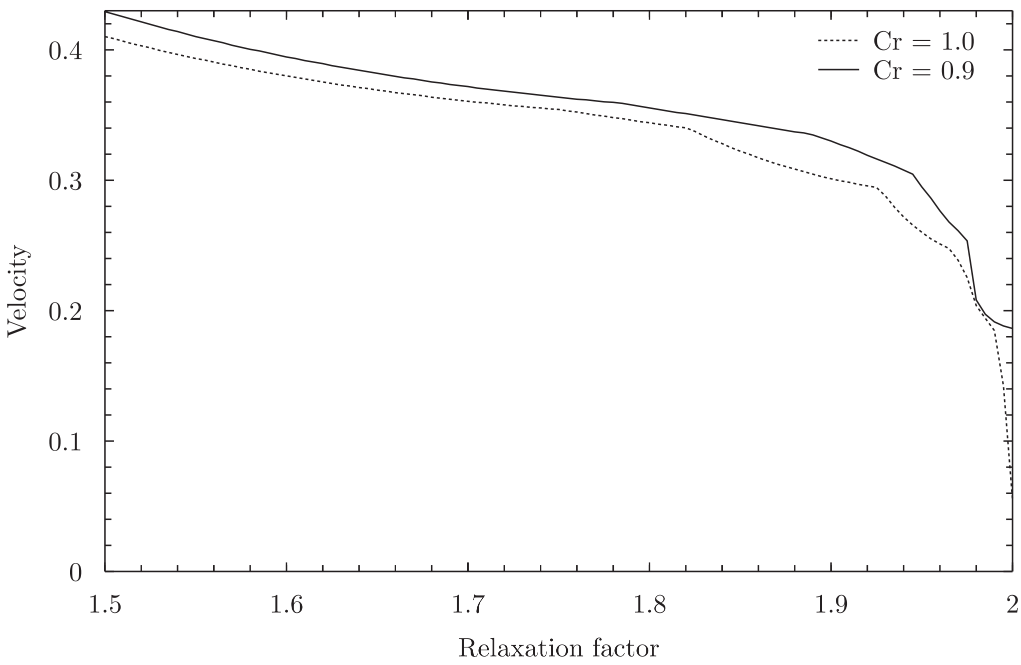


Figure 3.21: Stability boundary for `cub_d3q27_abc` lattice.

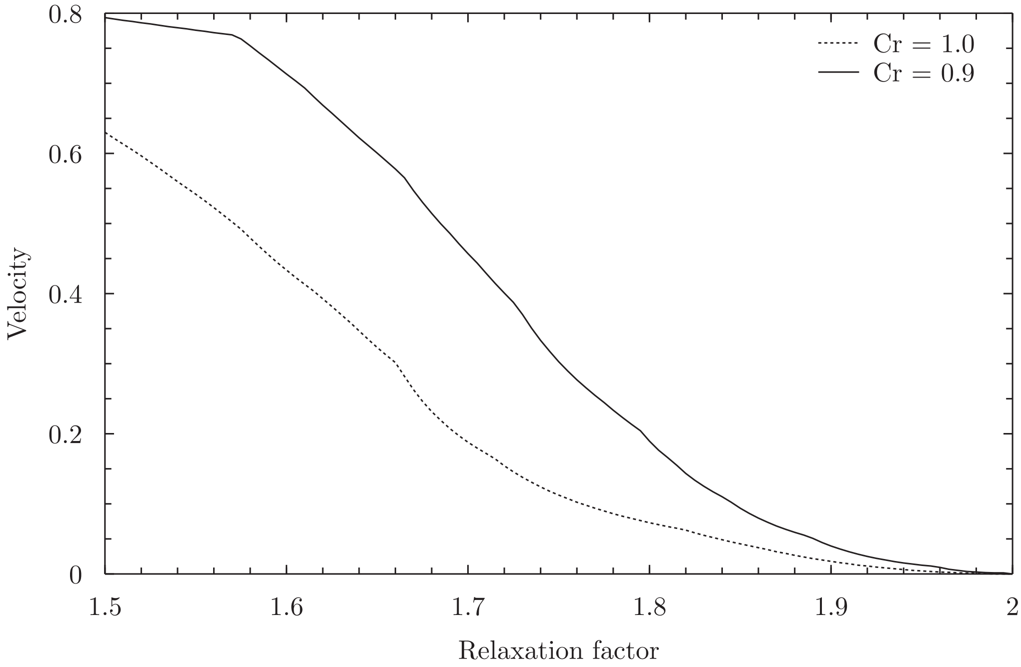


Figure 3.22: Stability boundary for `bcc_d3q27_abc` lattice.

used for LB methods as well as a Courant number of 0.9 obtained using the first order upwind difference scheme from Section 3.4.2. In this case and in all cases that follow a reduction of Courant number improves stability as shown.

Figure 3.17 shows the reasonably good stability of the `fcc_d3q13_a` lattice using the moment system of D'Humieres et al. [62]. In comparison, Figure 3.18 and Figure 3.20 show poor stability of the `cub_d3q15_ac` and `cub_d3q19_ab` schemes using the moment set from D'Humieres et al. [63]. Our experience with these lattices using the BGK collision operator indicate good stability which leads us to believe that the poor stability shown in these results is due to the moment basis that was chosen. Figure 3.19 shows that our `bcc_d3q15_ab` moment system has reasonably good stability until a relaxation factor of 1.85, our experiments have confirmed this. Figure 3.22 shows that our `bcc_d3q27_abc` moment system has a relatively poor stability, this has also been verified experimentally.

3.8 Boundary conditions

Our work has used periodic, slip and no-slip boundary conditions aligned with the grid. Periodic boundary conditions are simple to implement, the particles simply stream to the next node in their path taking into account the topology of the domain. The boundary may lie on a node as shown on the left hand side of Figure 3.23 resulting in a domain size of $n - 1$ or on a link as on the right hand side with a domain size of n. My code uses link type periodic boundaries.

The literature is filled with hundreds of papers detailing different implementations of the non-slip boundary. A search for "lattice Boltzmann boundary condition" on the Web of Science brings up roughly 1,300 results. The simplest way to implement a no-slip boundary condition is by a bounce-back rule at the wall. This rule is a leftover from LGCA where bouncing particles back from walls would enforce zero momentum at all times. Another simple boundary condition is to maintain the boundary nodes at the Maxwellian distribution at the correct temperature, this is

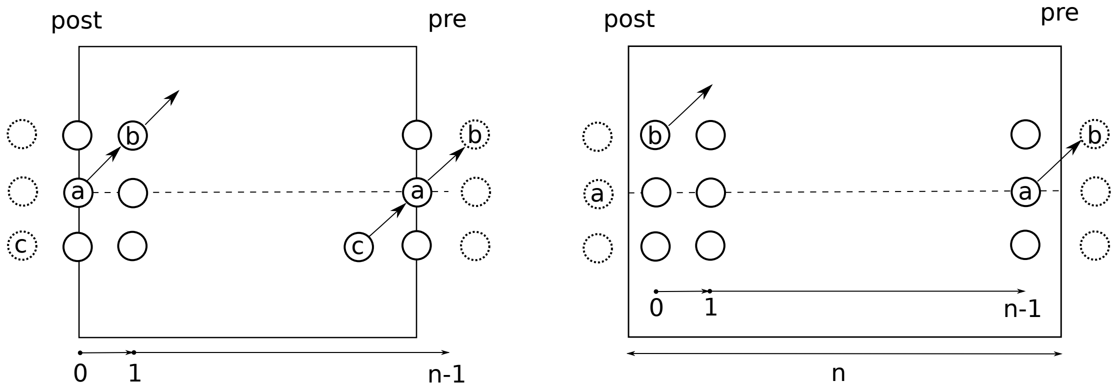


Figure 3.23: Periodic boundary conditions, node and link type.

often referred to as a “diffuse boundary” and is commonly used in other gas dynamic schemes, it does not result in a no-slip condition but rather has some slip velocity.

It was early recognized that the bounce back boundary condition does not accurately model a no-slip wall [86]. The location of zero normal velocity is close to the node itself but the location of zero tangential velocity is offset by close to half a lattice unit. Second order effects introduce errors that further change the wall location as a function of the derivatives of the flow field. One of the earliest efforts to formulate an accurate boundary condition was in 1991 by Cornubert, D’Humeieres and Levermore [59]. They applied a blend of specular reflection and bounce back rules in order to ensure a constant slip velocity, resulting in a rule that could also be applied to LGCA in a stochastic fashion by deciding if a particle should be bounced back or reflected according to a random variable.

Some of the earlier efforts concerning bounce back boundary conditions also referred to node-type and link-type bounce back boundary conditions [59]. Figure 3.24 shows both types, for the case of a node-type boundary, the particle streams to a fictitious node in one time step, then its direction is changed and it streams back to the original node taking two streaming operations in total. The link type boundary reverses the direction of the bounced back particle in one time step, as if it bounced from a wall half a lattice unit away. Link type bounce back conditions are the most common however it has been suggested that there are stability differences between

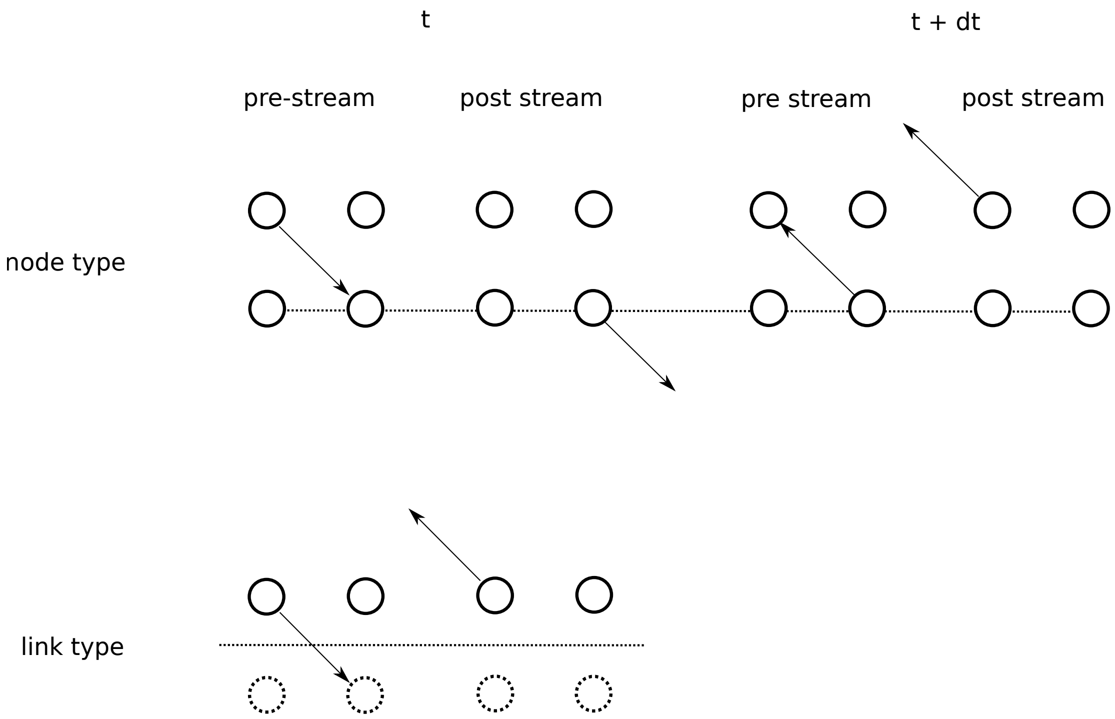


Figure 3.24: Node and link type bounce back boundary conditions.

the two [59]. Our code uses link-type bounce back boundary conditions.

Skodros [223] experimented in 1993 with augmenting the collision operator using derivatives of the hydrodynamic fields in order to restore higher order accuracy at the boundaries which are set to a Maxwellian distribution. The derivatives can be calculated by finite differences or found using the exact solution if that is known for the flow field. They found reduced numerical stability when using second order finite differences. Skodros also used his enhanced collisions to initialise the flow field which usually lacks non-equilibrium information if initialized using purely an equilibrium distribution. As could be expected, they found the best accuracy when using the exact values of derivatives and found that using a pure equilibrium distribution at the wall was less accurate as the relaxation rate diverged from one and non-equilibrium effects become more important. Augmenting the LB method with finite differences has also been proposed by Prasianakis et al. [186, 188] as a way to restore Galilean invariance and gain higher order accuracy using otherwise unsuitable velocity sets. Latt [135] also use a finite difference approximation for the non-equilibrium terms in their BC.

One of the earlier efforts to correct the slip velocity present in the bounce back boundary condition was by Ziegler [269]. The slip component is eliminated by setting the two opposite tangential velocities to be equal to the average of the other velocities and then lets the boundary undergo collisions. The modified method shows a slightly improved error using a limited test of channel flow although it is not clear that second order errors are eliminated.

Ginzbourg and Adler [86] analysed the behaviour of the bounce back BC in response to theoretical Poiseuille and stagnation flows and confirm that the location of zero tangential velocity is halfway along the link to within a second order for a specific value of the third order eigenvalue of the collision matrix. On the other hand, the location of zero normal velocity is generally not at the same point.

In 1995 Noble et al. [163] devised a simple boundary condition for the hexagonal D2Q7 lattice by noting that only two velocities are unknown at the boundary node. If the density at the node is also unknown, then there are exactly three equations in three unknowns required to specify the unknown velocities and density. Some tests using Poiseuille flow are presented. This BC cannot be applied in principle to other lattices and it is not clear that second order terms have been dealt with. Ginzbourg and D'Humieres [87] have noted that Noble's scheme suffers from the same second order errors as the bounce back rule.

Inamuro et al. [113] calculate only unknown particles using a Maxwellian distribution, the velocity used in the calculation has an added counter slip component so that the total velocity at the node is as desired. This boundary condition can be generalized to other lattices including three dimensional ones although some attempts to do so [135] have not been able to find an explicit formulation for the required counter slip velocity. Latt also found that there are still second order errors present in this scheme however that they should not affect simulations in practice [135].

Chen, Martinez and Mei [39] propose an extrapolation condition. A layer of ghost nodes outside of the fluid region are maintained by extrapolation,

$$f_i^{\text{ghost}} = 2f_i^{\text{bound}} - f_i^{\text{fluid}}.$$

The idea assumes that if the velocity gradient is linear, there will be a zero velocity at the boundary. It is not clear that second order errors are accounted for and our experiments indicate stability problems with this scheme.

Ginzbourg and D'Humieres [87] have used a precise theoretical analysis to arrive at what they call the "local second order boundary" (LSOB). Using the same multi-scale expansion as in the derivation of the LB method itself, the desired values of higher order moments (the momentum flux tensor) are calculated based on expected values for derivatives at the wall. Values at the wall, which is offset by some distance from the boundary node are converted to values at the node using Taylor expansion. The resulting linear system of equations for the unknown populations at the wall can then generally be solved. We have also performed calculations, shown later in this section that show the somewhat unexpected result that the non-zero components of the momentum flux tensor can be calculated using only the known particle populations for the D2Q9, D3Q15, D3Q19 and D3Q27 lattices.

The LSOB works for arbitrary positions and inclinations of the wall and for the most simple case of a non-slip wall, the coefficients used in the linear system remain constant and form an explicit method. Although it is based on a more rigorous framework than other methods, the LSOB does have one drawback, that mass is not conserved at the boundary. It is not clear that the linear system is solvable for other lattices not discussed in the paper.

Zou and He [270] bounce back the non-equilibrium distribution. This has become one of the more popular second order accurate boundary conditions. They note that the bounce back BC works well for relaxation rates close to one.

Guo, Zheng and Shi [100, 99] (GZS) superimpose the non-equilibrium moments from the nearest neighbour on to the desired equilibrium distribution at the wall. This BC was intended to be used for curved boundaries, in that case, the prescribed velocity needs to be calculated by an extrapolation technique discussed in the paper and the non-equilibrium moments are also extrapolated. The order of accuracy of this scheme is unclear however it does offer improved accuracy compared to the bounce back BC [235]. Unfortunately this boundary condition does not conserve mass at the boundary.

The GZS BC, like many of the more accurate boundary conditions including the LSOB aims to be applicable for general curved boundaries however we are only concerned with the case of a flat wall aligned with the grid. One possible modification to the GZS boundary to enforce mass conservation might be to adjusting the density at the wall to balance the populations streaming out of the domain.

Filippova and Hannel [79] propose an immersed boundary type scheme where nodes outside the flow are maintained at a velocity set so that the interpolated velocity at the boundary is zero. Their scheme deals with second order terms by a form of non-equilibrium extrapolation. Mei, Luo and Shyy [156] review the Filippova and Hannel boundary condition exploring the stability concerns introduced by the extrapolation. They improve the stability of this boundary condition with an improved choice of parameters.

Latt and Chopard [135] give an overview covering schemes by Inamuro et al., Zhou and He, their own "regularized BC" and their own finite difference based schemes which are based on earlier work by Skodros [223]. In their regularized scheme, they calculate the non-equilibrium momentum flux tensor based on the bounce back off non-equilibrium distributions in a manner similar to Zou and He. The populations at the wall node are then set to a superposition of the equilibrium distribution with a term that recovers the first order non-equilibrium momentum flux tensor.

In order to match the correct hydrodynamics at the wall up to second order, the

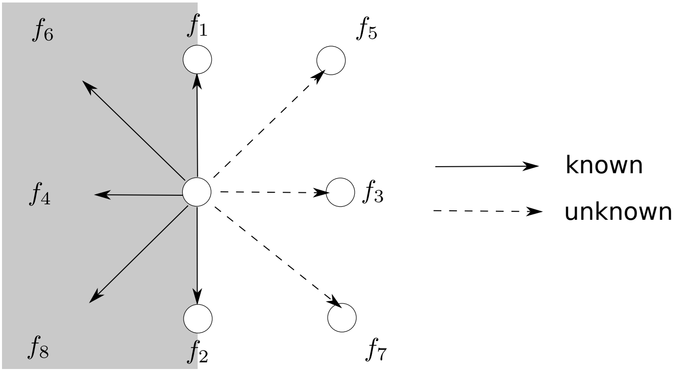


Figure 3.25: Node on a boundary using the D2Q9 lattice.

density, velocity and momentum flux tensor,

$$\begin{aligned}
\Pi_{\alpha\beta} &= \Pi_{\alpha\beta}^{eq} + \Pi_{\alpha\beta}^{(1)}, \\
\Pi_{\alpha\beta}^{eq} &= \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} = c_s^2 \delta_{\alpha\beta} \rho + \rho u_\alpha u_\beta, \\
\Pi_{\alpha\beta}^{(1)} &= \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{(1)} = c_s^2 \rho \left(\partial_\alpha u_\beta + \partial_\beta u_\alpha \right).
\end{aligned}$$

must all be correctly specified. At a no-slip wall, all velocities u_α are zero and all derivatives tangential to the boundary are zero. For example, in two dimensions for a boundary in the y direction,

$$\begin{aligned}
\Pi_{xx} &= c_s^2 \rho + 2 c_s^2 \rho \partial_x u_x, \\
\Pi_{yy} &= c_s^2 \rho, \\
\Pi_{xy} &= \rho \partial_x u_y.
\end{aligned}$$

Using the common low Mach number Cartesian lattices, the momentum flux tensor turns out to be fully constrained. Consider the D2Q9 lattice with a node positioned on the wall as in Figure 3.25. After collision and streaming as usual, the dashed particle populations (f_3, f_5 and f_7) are unknown since they would have streamed from within a solid wall.

Population	c_x	c_y
f_0	0	0
f_1	0	1
f_2	0	-1
f_3	1	0
f_4	-1	0
f_5	1	1
f_6	-1	1
f_7	1	-1
f_8	-1	-1

Table 3.20: Velocities for the D2Q9 lattice.

The specification of x momentum gives,

$$\begin{aligned} \rho u_x = 0 &= f_3 - f_4 + f_5 - f_6 + f_7 - f_8, \\ \Rightarrow f_3 + f_5 + f_7 &= f_4 + f_6 + f_8. \end{aligned}$$

This constrains the density to,

$$\rho = f_0 + f_1 + \cdots + f_8 = f_1 + f_2 + 2\left(f_4 + f_6 + f_8\right).$$

The specification of y momentum adds the constraint,

$$\begin{aligned} \rho u_y = 0 &= f_1 - f_2 + f_5 + f_6 - f_7 - f_8, \\ \Rightarrow f_5 - f_7 &= -f_1 + f_2 - f_6 + f_8. \end{aligned}$$

The unknown terms in the momentum flux tensor are thus constrained,

$$\begin{aligned} \Pi_{xx} &= c_s^2\rho + 2c_s^2\rho\partial_x u_x = f_3 + f_4 + f_5 + f_6 + f_7 + f_8 = 2\left(f_4 + f_6 + f_8\right), \\ \Pi_{xy} &= \rho\partial_x u_y = f_5 - f_6 - f_7 + f_8 = -f_1 + f_2 - 2\left(f_6 + f_8\right). \end{aligned}$$

Population	c_x	c_y	c_z	Population	c_x	c_y	c_z
f_0	0	0	0	f_{19}	1	1	1
f_1	0	0	1	f_{20}	1	-1	1
f_2	0	0	-1	f_{21}	1	1	-1
f_3	0	1	0	f_{22}	1	-1	-1
f_4	0	-1	0	f_{23}	-1	1	1
f_5	1	0	0	f_{24}	-1	-1	1
f_6	-1	0	0	f_{25}	-1	1	-1
f_7	0	1	1	f_{26}	-1	-1	-1
f_8	0	-1	1				
f_9	0	1	-1				
f_{10}	0	-1	-1				
f_{11}	1	1	0				
f_{12}	-1	1	0				
f_{13}	1	-1	0				
f_{14}	-1	-1	0				
f_{15}	1	0	1				
f_{16}	-1	0	1				
f_{17}	1	0	-1				
f_{18}	-1	0	-1				

Table 3.21: Velocities for the `cub_d3q27_abc` lattice.

The known component of the momentum flux tensor constrains f_3,

$$\begin{aligned}\Pi_{yy} = c_s^2\rho = f_1 + f_2 + f_5 + f_6 + f_7 + f_8 &= f_1 + f_2 + f_4 + 2\left(f_6 + f_8\right) - f_3, \\ \Rightarrow f_3 = f_1 + f_2 + f_4 + 2\left(f_6 + f_8\right) - c_s^2\rho &= \rho\left(1 - c_s^2\right) - f_4.\end{aligned}$$

The remaining distributions f_5 and f_7 can then be determined. These constraints are not met by any of the boundary conditions we are aware of however the Π_{xx} and Π_{xy} constraints are met by the bounce back and Zou-He conditions.

In three dimensions the values of the momentum flux tensor at a wall aligned with the yz plane are,

$$
\begin{aligned}
\Pi_{xx} &= c_s^2\rho + 2c_s^2\rho\partial_x u_x, \\
\Pi_{yy} &= c_s^2\rho, \\
\Pi_{zz} &= c_s^2\rho, \\
\Pi_{xy} &= \rho\partial_x u_y, \\
\Pi_{xz} &= \rho\partial_x u_z, \\
\Pi_{yz} &= 0.
\end{aligned}
$$

Using the `cub_d3q27_abc` lattice with velocities from Table 3.21, the unknown distributions are now f_5, f_{11}, f_{13}, f_{15}, f_{17}, f_{19}, f_{20}, f_{21} and f_{22}. It is clear that similar constraints can be placed on the unknown momentum flux terms.

$$
\begin{aligned}
\Pi_{xx} &= 2(f_6 + f_{12} + f_{14} + f_{16} + f_{18} + f_{23} + f_{24} + f_{25} + f_{26}, \\
\Pi_{xy} &= 2(-f_{12} + f_{14} - f_{23} + f_{24} - f_{25} + f_{26}), \\
\Pi_{xz} &= 2(-f_{16} + f_{18} - f_{23} - f_{24} + f_{25} + f_{26}).
\end{aligned}
$$

Thus all the non-equilibrium momentum flux terms can be calculated without knowledge of the missing particle populations. In this case, the 6 remaining equations are not sufficient to fully constrain the 9 unknown velocities.

3.9 Initialization

The most simple way to initialize a lattice Boltzmann simulation is to use equilibrium values. The non-equilibrium component of the momentum-flux tensor will not be correct as it will not contain the required gradient information. Skodros [223] uses an enhanced collision operator to add gradient information which is obtained using finite differences of the flow field. Another approach is to “thermalize” the flow after initializing with equilibrium values, the procedure is described below.

By treating the fluid velocities u_α as constant and leaving only density as a conserved moment, the evolution equation will satisfy a Poisson equation for pressure [26, 155],

$$\partial_t P = \chi \left(\partial_\alpha \partial_\alpha P + \rho \partial_\alpha \partial_\beta u_\alpha u_\beta \right) - c_s^2 \rho \partial_\alpha u_\alpha.$$

In the steady state limit $\partial_t P = 0$ and in the incompressible limit $\partial_\alpha u_\alpha = 0$ leaving,

$$\partial_\alpha \partial_\alpha P + \rho \partial_\alpha \partial_\beta u_\alpha u_\beta = 0.$$

The steady-state momentum flux tensor will approach the correct hydrodynamic value.

This scheme could also be used as a means of solving the pressure Poisson equation for the incompressible Navier-Stokes equations. A variation may also be used to solve the advection diffusion equation. The equilibrium moments are set to the following values,

$$\sum_{\boldsymbol{i}=0}^{q} g_{\boldsymbol{i}}^{\mathrm{eq}} = \phi,$$
$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} g_{\boldsymbol{i}}^{\mathrm{eq}} = \phi u_\alpha,$$
$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} g_{\boldsymbol{i}}^{\mathrm{eq}} \sim \phi.$$

Chapman-Enskog expansion shows that the scalar ϕ solves the advection diffusion equation [97, 110, 265, 157],

$$\partial_t \phi + \partial_\alpha u_\alpha \phi = \alpha \partial_\alpha \partial_\alpha \phi.$$

The diffusivity coefficient α depends upon the collision operator and the second order equilibrium tensor. As we shall cover in Chapter 4, lattice Boltzmann methods are significantly less adept both in terms of accuracy and stability than state of the art finite volume schemes at simulating the advection diffusion equation.

3.10 Stretched grid

We have been motivated to investigate means of implementing a stretched grid by our channel flow simulations which require significantly finer resolution near the wall. The grid we use is stretched, with a different grid spacing in each Cartesian direction however it is still uniform – that is the grid size does not vary. Using the techniques we have developed, we have successfully simulated laminar channel flow (Section 5.2), turbulent channel flow (Section 5.7) and Taylor-Green vortex flow (Section 5.4).

The interpolation supplemented LB method [102, 104, 68, 219, 131] uses particle locations that no longer coincide with grid points after advection so interpolation is used to reconstruct the populations at grid sites. Second order or higher interpolation must be used in order to maintain the second order accuracy of the LB method and results can be improved by using a least squares interpolation [219, 131]. These methods have had success at higher Reynolds numbers however they require significant additional computational effort and reduce stability. Another approach is to use local grid refinement, using smaller cube elements [79, 232, 189]. Grid refinement does not allow the use of non–rectangular elements and has been shown to successfully simulate fully turbulent three dimensional flows [200, 189]. There have also been previous efforts to use a rectangular lattice [19, 267] however our approach is the most accurate so far.

We consider the use of a stretched grid with the common second order low Mach number `d2q9`, `d3q15`, `d3q19` and `d3q27` lattices. The third order moments for these lattices are not correct for a rectangular grid since it is inevitable that

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}x}c_{\boldsymbol{i}x}c_{\boldsymbol{i}x}f_{\boldsymbol{i}} = \Delta x^2 u_x.$$

Our solution is to introduce a correction term via the eigenvalues of the collision operator. Higher order lattices with a larger number of velocities that allow individual setting of third order moments do not suffer from this problem. Zhou [267] has published a paper on the rectangular lattice Boltzmann method using the D2Q9 lattice however has neglected these errors completely as pointed out in the reply by Chikatamarla and Karlin [45]. Zhou has produced a follow-up paper [268] justifying the use of their scheme despite the additional errors that are introduced by benchmarking a selection of flows. Bouzidi et al. [19] have a similar scheme to ours however the moment basis they have used results in the introduction of additional error terms. Our method is more accurate than those of Zhou or Bouzidi et al. in theory as well as in practice as shown by our numerical experiments in section 5.4. The main improvement over the scheme of Bouzidi et al. is in the choice of moment basis. Bouzidi et al. use the linear combinations of lattice vectors $c_x c_x + c_y c_y$ and $c_x c_x - c_y c_y$ as eigenvectors which does not allow sufficient freedom to independently adjust the relaxation rate of the second order moments as required.

Consider a matrix A with two eigenvectors,

$$\begin{aligned} \psi_1 &= \frac{1}{2}\left(c_{xx} + c_{yy}\right), & \qquad c_{xx} &= \psi_1 + \psi_2, \\ \psi_2 &= \frac{1}{2}\left(c_{xx} - c_{yy}\right), & \qquad c_{yy} &= \psi_1 - \psi_2. \end{aligned}$$

and corresponding eigenvalues λ_1 and λ_2. The effect of left multiplication by this matrix on the lattice vectors is now,

$$\begin{aligned} A c_x c_x &= \lambda_1 \psi_1 + \lambda_2 \psi_2 = \frac{\lambda_1 + \lambda_2}{2} c_x c_x + \frac{\lambda_1 - \lambda_2}{2} c_y c_y, \\ A c_y c_y &= \lambda_1 \psi_1 - \lambda_2 \psi_2 = \frac{\lambda_1 - \lambda_2}{2} c_x c_x + \frac{\lambda_1 + \lambda_2}{2} c_y c_y. \end{aligned}$$

There is no way to individually adjust the coefficients of the second order lattice vectors. Bouzidi et al. do not provide a multi-scale expansion for their scheme instead showing their analysis of the dispersion relation which is claimed to be sufficient proof of its accuracy.

We introduce the following rank two tensors in order to allow analysis of these new sources of error;

$$d^{sq}_{\alpha\beta} = \begin{pmatrix} \Delta x^2 & 0 & 0 \\ 0 & \Delta y^2 & 0 \\ 0 & 0 & \Delta z^2 \end{pmatrix}, \ \lambda^{-1}_{\alpha\beta} = \begin{pmatrix} \lambda^{-1}_{xx} & 0 & 0 \\ 0 & \lambda^{-1}_{yy} & 0 \\ 0 & 0 & \lambda^{-1}_{zz} \end{pmatrix}.$$

We also generalise these to diagonal rank four tensors,

$$\left.\begin{aligned} d^{sq}_{\alpha\beta\gamma\delta} &= d^{sq}_{\alpha\beta} \\ \lambda^{-1}_{\alpha\beta\gamma\delta} &= \lambda^{-1}_{\alpha\beta} \end{aligned}\right\} \quad \text{if} \quad \alpha = \beta = \gamma = \delta.$$

The third order moments are,

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f^{eq}_{\boldsymbol{i}} = \left[c_s^2(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}) - 3c_s^2\delta_{\alpha\beta\gamma\delta} + d^{sq}_{\alpha\beta\gamma\delta}\right] \rho u_\delta.$$

An anisotropic term which will cancel errors is introduced by setting eigenvalues λ_{xx}, λ_{yy} and λ_{zz}, corresponding to eigenvectors $c_{\boldsymbol{i}x}c_{\boldsymbol{i}x}$, $c_{\boldsymbol{i}y}c_{\boldsymbol{i}y}$ and $c_{\boldsymbol{i}z}c_{\boldsymbol{i}z}$ differently to λ_2,

$$\sum_{\boldsymbol{j}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} A^{-1}_{\boldsymbol{ij}} = \lambda_2^{-1} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} + \left(\lambda^{-1}_{\alpha\beta\epsilon\zeta} - \lambda_2^{-1}\delta_{\alpha\beta\epsilon\zeta}\right) c_{\boldsymbol{i}\epsilon} c_{\boldsymbol{i}\zeta}.$$

Substituting into Eqn. 3.11 and adding additional terms due to discretization (Eqn. 3.19),

$$\begin{aligned} &\partial_{t_2} \rho u_\alpha = \\ &\quad \left(\lambda_2^{-1} + \alpha - \frac{\Delta t}{\mathrm{Cr}}\right) \partial_\beta \left\{ \partial_\gamma \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f^{eq}_{\boldsymbol{i}} + \partial_{t_1} \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f^{eq}_{\boldsymbol{i}} - \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} F_{\boldsymbol{i}} \right\} \\ &\quad + \left(\lambda^{-1}_{\alpha\beta\epsilon\zeta} - \lambda_2^{-1}\delta_{\alpha\beta\epsilon\zeta}\right) \partial_\beta \left\{ \partial_\gamma \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\epsilon} c_{\boldsymbol{i}\zeta} c_{\boldsymbol{i}\gamma} f^{eq}_{\boldsymbol{i}} + \partial_{t_1} \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\epsilon} c_{\boldsymbol{i}\zeta} f^{eq}_{\boldsymbol{i}} - \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\epsilon} c_{\boldsymbol{i}\zeta} F_{\boldsymbol{i}} \right\}. \end{aligned}$$

The terms inside curly brackets can be simplified using Eqn. 3.13,

$$
\begin{aligned}
\partial_\beta \left\{ \partial_\gamma \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f_{\boldsymbol{i}}^{eq} + \partial_{t_1} \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} - \sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} F_{\boldsymbol{i}} \right\} \\
= c_s^2 \partial_\beta \left(\partial_\alpha \rho u_\beta + \partial_\beta \rho u_\alpha \right) - c_s^2 \left(u_\alpha \partial_\beta \rho + u_\beta \partial_\alpha \rho \right) \\
+ \partial_\beta \partial_\gamma \left(d^{sq}_{\alpha\beta\gamma\delta} - 3 c_s^2 \delta_{\alpha\beta\gamma\delta} \right) \rho u_\delta + \mathcal{O}\left(u^3\right).
\end{aligned}
$$

Expanding leads to,

$$
\begin{aligned}
\partial_{t_2} \rho u_\alpha = c_s^2 \left(\lambda_2^{-1} + \alpha - \frac{\Delta t}{\mathrm{Cr}} \right) \partial_\beta \left(\rho \partial_\alpha u_\beta + \rho \partial_\beta u_\alpha \right) \\
+ \left(\lambda_2^{-1} + \alpha - \frac{\Delta t}{\mathrm{Cr}} \right) \partial_\beta \partial_\gamma \left(d^{sq}_{\alpha\beta\gamma\delta} - 3 c_s^2 \delta_{\alpha\beta\gamma\delta} \right) \rho u_\delta \\
+ c_s^2 \left(\lambda^{-1}_{\alpha\beta\epsilon\zeta} - \lambda_2^{-1} \delta_{\alpha\beta\epsilon\zeta} \right) \partial_\beta \left(\partial_\epsilon \rho u_\zeta + \partial_\zeta \rho u_\epsilon \right) \\
- c_s^2 \left(\lambda^{-1}_{\alpha\beta\epsilon\zeta} - \lambda_2^{-1} \delta_{\alpha\beta\epsilon\zeta} \right) \partial_\beta \left(u_\epsilon \partial_\zeta \rho + u_\zeta \partial_\epsilon \rho \right) \\
+ \left(\lambda^{-1}_{\alpha\beta\epsilon\zeta} - \lambda_2^{-1} \delta_{\alpha\beta\epsilon\zeta} \right) \partial_\beta \partial_\gamma \left(d^{sq}_{\epsilon\zeta\gamma\delta} - 3 c_s^2 \delta_{\epsilon\zeta\gamma\delta} \right) \rho u_\delta + \mathcal{O}\left(u^3\right).
\end{aligned}
$$

The contribution from the anisotropic rank four terms can mostly be set to zero by setting each of the eigenvalues λ_{xx}, λ_{yy} and λ_{zz} to,

$$
\lambda_{xx} = \frac{\Delta x^2 - c_s^2}{2 \frac{c_s^2}{\lambda_2} + \left(\alpha - \frac{\Delta t}{Cr} \right) \left(3 c_s^2 - \Delta x^2 \right)}. \tag{3.26}
$$

These values solve the equation,

$$
\left(\alpha - \frac{\Delta t}{Cr} + \lambda_{xx}^{-1} \right) \left(\Delta x^2 - 3 c_s^2 \right) + 2 c_s^2 \left(\lambda_{xx}^{-1} - \lambda_2^{-1} \right) = 0.
$$

The shear viscosity is unchanged,

$$
\nu^s = c_s^2 \left(\frac{1}{\lambda_2} + \alpha - \frac{\Delta t}{\mathrm{Cr}} \right).
$$

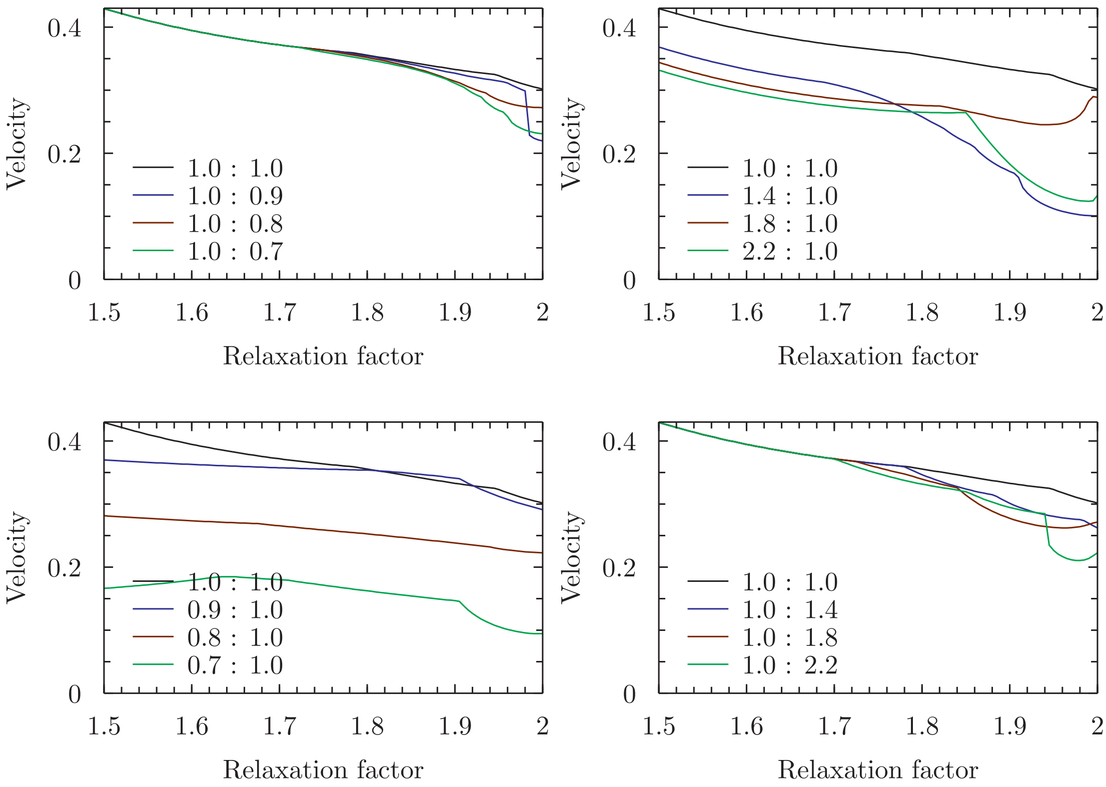


Figure 3.26: Stability boundaries for stretched grid using `sqr_d2q9_ab` lattice and various aspect ratios ($\Delta x : \Delta y$).

An anisotropic error term,

$$c_s^2 \left(\lambda^{-1}_{\alpha\beta\epsilon\zeta} - \lambda_2^{-1}\delta_{\alpha\beta\epsilon\zeta}\right) \partial_\beta \left(u_\epsilon \partial_\zeta \rho + u_\zeta \partial_\epsilon \rho\right)$$

remains unresolved but is expected to be small when density fluctuations are small and for moderate grid aspect ratios. We have successfully simulated laminar channel flow (Section 5.2), turbulent channel flow (Section 5.7) and Taylor-Green vortex flow (Section 5.4) in order to confirm the stability and accuracy of the scheme and confirm the improved accuracy compared to the scheme of Bouzidi et al. We suspect that these anisotropic errors are responsible for the reduction in stability experienced at larger grid aspect ratios.

We have performed linear stability analysis on our stretched grid schemes. Figure 3.26 shows stability results obtained when a mean velocity in the x direction was considered, these results use a Courant number of 0.9 in all cases. In general, stability is reduced as the grid is stretched. The stability is more sensitive to changes of grid in the x direction which is aligned with the mean velocity. As grid size is reduced, there appears to be a hard lower limit near a value of half beyond which the scheme is completely unstable with a maximum velocity of zero for all relaxation factors. There does not appear to be a corresponding upper limit. While maximum velocity does decrease for larger grid sizes, even at an aspect ratio of 10 the maximum velocity is still not zero. Not shown, results using our three dimensional `cub_d3q27_abc` lattice and a stretched grid are almost identical.

3.A Appendix - List of lattices

3.A.1 `sqr_d2q9_ab`

We have developed the following data for the `sqr_d2q9_ab` lattice. The basis vectors are;

$$
\begin{aligned}
n_0 &= c^0 \\
n_1 &= c_x \\
n_2 &= c_y \\
n_3 &= c_x^2 \\
n_4 &= c_y^2 \\
n_5 &= c_x c_y \\
n_6 &= c_x c_y^2 \\
n_7 &= c_y c_x^2 \\
n_8 &= c_x^2 c_y^2
\end{aligned}
$$

The T matrix is,

$$T = \mathrm{diag}(9, 6, 6, 18, 18, 4, 12, 12, 36).$$

```
  1  1  1  1  1  1  1  1  1
  .  1 -1  .  .  1 -1 -1  1
  .  .  .  1 -1  1 -1  1 -1
  .  1  1  .  .  1  1  1  1
  .  .  .  1  1  1  1  1  1
  .  .  .  .  .  1  1 -1 -1
  .  .  .  .  .  1 -1 -1  1
  .  .  .  .  .  1 -1  1 -1
  .  .  .  .  .  1  1  1  1
49 add operations
```

Table 3.22: Lattice vector matrix N for the `sqr_d2q9_ab` lattice.

```
  1  .  .  .  .  .  .  .  .
  .  1  .  .  .  .  1  .  .
  .  .  1  .  .  .  .  1  .
  .  .  .  1  .  .  .  .  1
  .  .  .  .  1  .  .  .  1
  .  .  .  .  .  1  .  .  .
  .  .  .  .  .  .  1  .  .
  .  .  .  .  .  .  .  1  .
  .  .  .  .  .  .  .  .  1
4 add operations
```

Table 3.23: Lattice vector matrix N^0 for the `sqr_d2q9_ab` lattice.

```
  1  1  1  1  1  1  1  1  1
  .  1 -1  .  .  .  .  .  .
  .  .  .  1 -1  .  .  .  .
  .  1  1  .  .  .  .  .  .
  .  .  .  1  1  .  .  .  .
  .  .  .  .  .  1  1 -1 -1
  .  .  .  .  .  1 -1 -1  1
  .  .  .  .  .  1 -1  1 -1
  .  .  .  .  .  1  1  1  1
41 add operations
```

Table 3.24: Lattice vector matrix N^1 for the `sqr_d2q9_ab` lattice.

```
  1  .  .  .  .  .  .  .  .
  .  1  .  .  .  .  .  .  .
  .  .  1  .  .  .  .  .  .
 -2  .  .  3  .  .  .  .  .
 -2  .  .  .  3  .  .  .  .
  .  .  .  .  .  1  .  .  .
  . -2  .  .  .  .  3  .  .
  . -2  .  .  .  .  .  3  .
  4  .  . -6 -6  .  .  .  9
7  add operations
12 mul operations
```

Table 3.25: Transformation matrix Q for the `sqr_d2q9_ab` lattice.

```
 1  1  1  1  1  1  1  1  1
 .  1 -1  .  .  1 -1 -1  1
 .  .  .  1 -1  1 -1  1 -1
-2  1  1 -2 -2  1  1  1  1
-2 -2 -2  1  1  1  1  1  1
 .  .  .  .  .  1  1 -1 -1
 . -2  2  .  .  1 -1 -1  1
 .  .  . -2  2  1 -1  1 -1
 4 -2 -2 -2 -2  1  1  1  1
64 add operations
15 mul operations
```

Table 3.26: Orthogonal basis matrix M for the `sqr_d2q9_ab` lattice.

The orthogonal vectors for the `sqr_d2q9_ab` lattice are;

$$\begin{aligned}
\psi_0 &= c^0 \\
\psi_1 &= c_x \\
\psi_2 &= c_y \\
\psi_3 &= 3c_x^2 - 2\Delta x^2 c^0 \\
\psi_4 &= 3c_y^2 - 2\Delta y^2 c^0 \\
\psi_5 &= c_x c_y \\
\psi_6 &= 3c_x c_y^2 - 2\Delta y^2 c_x \\
\psi_7 &= 3c_y c_x^2 - 2\Delta x^2 c_y \\
\psi_8 &= 9c_x^2 c_y^2 - 6\Delta x^2 c_y^2 - 6\Delta y^2 c_x^2 + 4\Delta x^2 \Delta y^2
\end{aligned}$$

The resulting equilibrium moments for the `sqr_d2q9_ab` lattice are;

$$\begin{aligned}
m_0^{eq} &= \rho \\
m_1^{eq} &= \rho u_x \\
m_2^{eq} &= \rho u_y \\
m_3^{eq} &= \rho(3c_s^2 - 2\Delta x^2) + 3\rho u_x u_x \\
m_4^{eq} &= \rho(3c_s^2 - 2\Delta y^2) + 3\rho u_y u_y \\
m_5^{eq} &= \rho u_x u_y \\
m_6^{eq} &= \rho u_x(3\gamma_1 - 2\Delta y^2) \\
m_7^{eq} &= \rho u_y(3\gamma_1 - 2\Delta x^2) \\
m_8^{eq} &= \rho(9\gamma_2 - 6c_s^2(\Delta x^2 + \Delta y^2) + 4\Delta x^2 \Delta y^2) \\
&\qquad + \rho u_x u_x(9\gamma_3 - 6\Delta y^2) + \rho u_y u_y(9\gamma_3 - 6\Delta x^2)
\end{aligned}$$

The orthogonal matrix required 64 add operations and 15 mul operations to calculate, a total of 77 operations. The factorized matrix $N^0 N^1 Q$ requires $4+41+7 = 52$ add operations and 12 mul operations to calculate, a total of 64 operations.

The temperature and speed of sound are constant for this lattice with values,

$$RT = c_s^2 = \frac{1}{3}.$$

1	1	1	1	1	1	1	1	1	1	1	1	1
.	1	-1	.	.	2	-2	.	.	1	-1	-1	1
.	.	.	1	-1	.	.	2	-2	1	-1	1	-1
.	1	1	1	1	4	4	4	4	2	2	2	2
.	1	1	-1	-1	4	4	-4	-4	.	.	.	.
.	.	.	.	.	.	.	.	.	1	1	-1	-1
.	1	-1	.	.	8	-8	.	.	1	-1	-1	1
.	.	.	1	-1	.	.	8	-8	1	-1	1	-1
.	.	.	.	.	.	.	.	.	1	-1	1	-1
.	.	.	.	.	.	.	.	.	1	-1	-1	1
.	1	1	1	1	16	16	16	16	2	2	2	2
.	1	1	-1	-1	16	16	-16	-16	.	.	.	.
.	.	.	.	.	.	.	.	.	1	1	1	1

Table 3.27: Lattice vector matrix N for the `sqr_d2q13_a11b` lattice.

3.A.2 `sqr_d2q13_a11b`

The data presented in this section is incomplete, it is included to show existence of the equilibrium distribution. The basis vectors for the `sqr_d2q13_a11b` lattice are;

$$n_0 = c^0$$

$$n_1 = c_x$$

$$n_2 = c_y$$

$$n_3 = c_\alpha^2$$

$$n_4 = c_x^2 - c_y^2$$

$$n_5 = c_x c_y$$

$$n_6 = c_x^3$$

$$n_7 = c_y^3$$

$$n_8 = c_y c_x^2$$

$$n_9 = c_x c_y^2$$

$$n_{10} = c_x^4 + c_y^4$$

$$n_{11} = c_x^4 - c_y^4$$

$$n_{12} = c_x^2 c_y^2$$

```
  1   .   .    .   .   .   .   .   .   .   .   .   . (   1)
  .   1   .    .   .   .   .   .   .   .   .   .   . (   1)
  .   .   1    .   .   .   .   .   .   .   .   .   . (   1)
-28   .   .    1   .   .   .   .   .   .   .   .   . (1/13)
  .   .   .    .   1   .   .   .   .   .   .   .   . (   1)
  .   .   .    .   .   1   .   .   .   .   .   .   . (   1)
  . -19   .    .   .   .   1   .   .   .   .   .   . ( 1/7)
  .   . -19    .   .   .   .   1   .   .   .   .   . ( 1/7)
  .   .  -8    .   .   .   .   2   1   .   .   .   . ( 1/9)
  .  -8   .    .   .   .   2   .   .   1   .   .   . ( 1/9)
336   .   . -365   .   .   .   .   .   .   1   .   . (1/77)
  .   .   .    . -65   .   .   .   .   .   .   1   . (1/17)
 12   .   .  -23   .   .   .   .   .   .   5   .   1 (1/27)
```

Table 3.28: Transformation matrix Q for the `sqr_d2q13_a11b` lattice.

```
  1   1   1   1   1   1   1   1   1    1    1    1    1 (   1)
  .   1  -1   .   .   2  -2   .   .    1   -1   -1    1 (   1)
  .   .   .   1  -1   .   .   2  -2    1   -1    1   -1 (   1)
-28 -15 -15 -15 -15  24  24  24  24   -2   -2   -2   -2 (1/13)
  .   1   1  -1  -1   4   4  -4  -4    .    .    .    . (   1)
  .   .   .   .   .   .   .   .   .    1    1   -1   -1 (   1)
  . -12  12   .   .  18 -18   .   .  -12   12   12  -12 ( 1/7)
  .   .   . -12  12   .   .  18 -18  -12   12  -12   12 ( 1/7)
  .   .   .  -2   2   .   .   .   .    1   -1    1   -1 ( 1/3)
  .  -2   2   .   .   .   .   .   .    1   -1   -1    1 ( 1/3)
336  48  48  48  48 108 108 108 108 -240 -240 -240 -240 (1/77)
  . -48 -48  48  48  12  12 -12 -12    .    .    .    . (1/17)
  4  -2  -2  -2  -2   .   .   .   .    1    1    1    1 ( 1/9)
```

Table 3.29: Orthogonal basis matrix M for the `sqr_d2q13_a11b` lattice.

The T matrix is,

$$T = \mathrm{diag}(13, 14, 14, \frac{308}{13}, 68, 4, \frac{216}{7}, \frac{216}{7}, \frac{4}{3}, \frac{4}{3}, \frac{5184}{77}, \frac{576}{17}, \frac{4}{9}).$$

3.A.3 `fcc_d3q13_a`

The `fcc_d3q13_a` lattice was first introduced as a moment based method in 2001 by D'Humieres et al. [62]. We have developed the sparse decomposition for this lattice. This lattice uses the minimum possible number of velocities required to recover low Mach number athermal hydrodynamics and as such has enjoyed application in performance critical applications such as GPU computing [244]. Karlin and Asinari [121] have developed a novel 'quasi-equilibrium' approach to allow the use of this lattice without the need for a moment based method. Petkov et al. [177] have attempted to use this lattice with a conventional equilibrium distribution without correcting for the anisotropic errors. The basis vectors for the `fcc_d3q13_a` lattice

```
 1  1  1  1  1  1  1  1  1  1  1  1  1
 .  1  1 -1 -1  .  .  .  .  1  1 -1 -1
 .  1 -1  1 -1  1  1 -1 -1  .  .  .  .
 .  .  .  .  .  1 -1  1 -1  1 -1  1 -1
 .  1  1  1  1  .  .  .  .  1  1  1  1
 .  1  1  1  1  1  1  1  1  .  .  .  .
 .  .  .  .  .  1  1  1  1  1  1  1  1
 .  1 -1 -1  1  .  .  .  .  .  .  .  .
 .  .  .  .  .  .  .  .  .  1 -1 -1  1
 .  .  .  .  .  1 -1 -1  1  .  .  .  .
 .  1  1 -1 -1  .  .  .  . -1 -1  1  1
 . -1  1 -1  1  1  1 -1 -1  .  .  .  .
 .  .  .  .  . -1  1 -1  1  1 -1  1 -1
97 add operations
```

Table 3.30: Lattice vector matrix N for the `fcc_d3q13_a` lattice.

are;

$$n_0 = c^0$$

$$n_1 = c_x$$

$$n_2 = c_y$$

$$n_3 = c_z$$

$$n_4 = c_x^2$$

$$n_5 = c_y^2$$

$$n_6 = c_z^2$$

$$n_7 = c_x c_y$$

$$n_8 = c_x c_z$$

$$n_9 = c_y c_z$$

$$n_{10} = c_x(c_y^2 - c_z^2)$$

$$n_{11} = c_y(c_z^2 - c_x^2)$$

$$n_{12} = c_z(c_x^2 - c_y^2)$$

The T matrix is,

$$T = \mathrm{diag}(13, 8, 8, 8, 624, 24, 8, 4, 4, 4, 8, 8, 8).$$

```
  1  .  .  .  .  .  .  .  .  .  .  .  .
  .  1  .  .  .  .  .  .  .  .  1  .  .
  .  .  1  .  .  .  .  .  .  .  .  1  .
  .  .  .  1  .  .  .  .  .  .  .  .  1
  .  .  .  .  1  .  1  .  .  .  .  .  .
  .  .  .  .  1  1  .  .  .  .  .  .  .
  .  .  .  .  1  .  1  .  .  .  .  .  .
  .  .  .  .  .  .  .  1  .  .  .  .  .
  .  .  .  .  .  .  .  .  1  .  .  .  .
  .  .  .  .  .  .  .  .  .  1  .  .  .
  . -1  .  .  .  .  .  .  .  .  1  .  .
  .  . -1  .  .  .  .  .  .  .  .  1  .
  .  .  . -1  .  .  .  .  .  .  .  .  1
9 add operations
```

Table 3.31: Lattice vector matrix N^0 for the `fcc_d3q13_a` lattice.

```
  1  1  1  1  1  1  1  1  1  1  1  1  1
  .  .  .  .  .  .  .  .  .  1  1 -1 -1
  .  1 -1  1 -1  .  .  .  .  .  .  .  .
  .  .  .  .  .  1 -1  1 -1  .  .  .  .
  .  1  1  1  1  .  .  .  .  .  .  .  .
  .  .  .  .  .  1  1  1  1  .  .  .  .
  .  .  .  .  .  .  .  .  .  1  1  1  1
  .  1 -1 -1  1  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  1 -1 -1  1
  .  .  .  .  .  1 -1 -1  1  .  .  .  .
  .  1  1 -1 -1  .  .  .  .  .  .  .  .
  .  .  .  .  .  1  1 -1 -1  .  .  .  .
  .  .  .  .  .  .  .  .  .  1 -1  1 -1
61 add operations
```

Table 3.32: Lattice vector matrix N^1 for the `fcc_d3q13_a` lattice.

```
  1  .  .  .  .  .  .  .  .  .  .  .  .
  .  1  .  .  .  .  .  .  .  .  .  .  .
  .  .  1  .  .  .  .  .  .  .  .  .  .
  .  .  .  1  .  .  .  .  .  .  .  .  .
  .  .  .  .  1  1  1  .  .  .  .  .  .
  .  .  .  .  .  1  .  .  .  .  .  .  .
  .  .  .  . -1  .  1  .  .  .  .  .  .
  .  .  .  .  .  .  .  1  .  .  .  .  .
  .  .  .  .  .  .  .  .  1  .  .  .  .
  .  .  .  .  .  .  .  .  .  1  .  .  .
  .  .  .  .  .  .  .  .  .  .  1  .  .
  .  .  .  .  .  .  .  .  .  .  .  1  .
  .  .  .  .  .  .  .  .  .  .  .  .  1
3 add
```

Table 3.33: Transformation matrix Q^0 for the `fcc_d3q13_a` lattice.

```
  1  .  .  .  .  .  .  .  .  .  .  .  .
  .  1  .  .  .  .  .  .  .  .  .  .  .
  .  .  1  .  .  .  .  .  .  .  .  .  .
  .  .  .  1  .  .  .  .  .  .  .  .  .
-24  .  .  . 13  .  .  .  .  .  .  .  .
  .  .  .  . -1  3  .  .  .  .  .  .  .
  .  .  .  .  .  .  1  .  .  .  .  .  .
  .  .  .  .  .  .  .  1  .  .  .  .  .
  .  .  .  .  .  .  .  .  1  .  .  .  .
  .  .  .  .  .  .  .  .  .  1  .  .  .
  .  .  .  .  .  .  .  .  .  .  1  .  .
  .  .  .  .  .  .  .  .  .  .  .  1  .
  .  .  .  .  .  .  .  .  .  .  .  .  1
2 add
3 mul
```

Table 3.34: Transformation matrix Q^1 for the fcc_d3q13_a lattice.

```
  1  1  1  1  1  1  1  1  1  1  1  1  1
  .  1  1 -1 -1  .  .  .  .  1  1 -1 -1
  .  1 -1  1 -1  1  1 -1 -1  .  .  .  .
  .  .  .  .  .  1 -1  1 -1  1 -1  1 -1
-24  2  2  2  2  2  2  2  2  2  2  2  2
  .  1  1  1  1  1  1  1  1 -2 -2 -2 -2
  . -1 -1 -1 -1  1  1  1  1  .  .  .  .
  .  1 -1 -1  1  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  1 -1 -1  1
  .  .  .  .  .  1 -1 -1  1  .  .  .  .
  .  1  1 -1 -1  .  .  .  . -1 -1  1  1
  . -1  1 -1  1  1  1 -1 -1  .  .  .  .
  .  .  .  .  . -1  1 -1  1  1 -1  1 -1
106 add
17 mul
```

Table 3.35: Orthogonal basis matrix M for the fcc_d3q13_a lattice.

The orthogonal vectors for the `fcc_d3q13_a` lattice are;

$$\begin{aligned}
\psi_0 &= c^0 \\
\psi_1 &= c_x \\
\psi_2 &= c_y \\
\psi_3 &= c_z \\
\psi_4 &= 13c_\alpha^2 - 24c^0 \\
\psi_5 &= 3c_y^2 - c_\alpha^2 \\
\psi_6 &= c_z^2 - c_x^2 \\
\psi_7 &= c_x c_y \\
\psi_8 &= c_x c_z \\
\psi_9 &= c_y c_z \\
\psi_{10} &= c_x(c_y^2 - c_z^2) \\
\psi_{11} &= c_y(c_z^2 - c_x^2) \\
\psi_{12} &= c_z(c_x^2 - c_y^2)
\end{aligned}$$

The resulting equilibrium moments for the `fcc_d3q13_a` lattice are;

$$
\begin{aligned}
m_0^{eq} &= \rho \\
m_1^{eq} &= \rho u_x \\
m_2^{eq} &= \rho u_y \\
m_3^{eq} &= \rho u_z \\
m_4^{eq} &= 13\rho(u_x^2 + u_y^2 + u_z^2) - 11\rho \\
m_5^{eq} &= \rho(2u_y^2 - u_x^2 - u_z^2) \\
m_6^{eq} &= \rho(u_z^2 - u_x^2) \\
m_7^{eq} &= \rho u_x u_y \\
m_8^{eq} &= \rho u_x u_z \\
m_9^{eq} &= \rho u_y u_z \\
m_{10}^{eq} &= 0 \\
m_{11}^{eq} &= 0 \\
m_{12}^{eq} &= 0
\end{aligned}
$$

The orthogonal matrix required 106 add operations and 17 mul operations to calculate, a total of 123 operations. The factorized matrix $N^0 N^1 Q^0 Q^1$ requires $9 + 61 + 3 + 2 = 75$ add operations and 3 mul operations to calculate, a total of 78 operations.

The temperature and speed of sound are constant for this lattice with values,

$$RT = c_s^2 = \frac{1}{3}.$$

The lattice vectors are equal to,

$$\begin{aligned}\sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} &= 1, \\ \sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} &= 0, \\ \sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} &= c_s^2 \delta_{\alpha\beta}, \\ \sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} &= 0, \\ \sum_{\boldsymbol{i}=0}^{q} W_{\boldsymbol{i}} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} c_{\boldsymbol{i}\delta} &= c_s^4 \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma}\right) - c_s^4 \delta_{\alpha\beta\gamma\delta}.\end{aligned}$$

The rank four delta term can be dealt with by using the modified equilibrium distribution that we have developed,

$$f_{\boldsymbol{i}}^{eq} = \rho W_{\boldsymbol{i}} \left[1 + \frac{c_{\boldsymbol{i}\alpha} u_\alpha}{c_s^2} + \frac{1}{2c_s^4}\left(\delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\beta\gamma\delta}\right) c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} u_\gamma u_\delta - \frac{u_\alpha u_\alpha}{c_s^2}\right].$$

The equilibrium moments take their correct values, only the rank zero, and two moments involve different terms to the standard distribution from Section 3.2. The calculation for the rank zero moment is straightforward,

$$\begin{aligned}\sum_{\boldsymbol{i}=0}^{q} f_{\boldsymbol{i}}^{eq} &= \rho \left(1 + \frac{c_s^2}{2c_s^4}\left(\delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\beta\gamma\delta}\right)\delta_{\alpha\beta} u_\gamma u_\delta - \frac{u_\alpha u_\alpha}{c_s^2}\right), \\ &= \rho \left(1 + \frac{c_s^2}{2c_s^4} 2\delta_{\gamma\delta} u_\gamma u_\delta - \frac{u_\alpha u_\alpha}{c_s^2}\right) = \rho.\end{aligned}$$

The calculation for the rank two moment contains more terms,

$$\begin{aligned}
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} f_{\boldsymbol{i}}^{eq} &= \rho \Bigg(c_s^2 \delta_{\alpha\beta} - c_s^2 \delta_{\alpha\beta} \frac{u_\gamma u_\gamma}{c_s^2} \\
&\quad + c_s^4 \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma} - \delta_{\alpha\beta\gamma\delta} \right) \frac{u_\gamma u_\delta}{2c_s^4} \\
&\quad + c_s^4 \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\gamma}\delta_{\beta\delta} + \delta_{\alpha\delta}\delta_{\beta\gamma} - \delta_{\alpha\beta\gamma\delta} \right) \delta_{\gamma\delta\epsilon\zeta} \frac{u_\epsilon u_\zeta}{2c_s^4} \Bigg), \\
&= \rho \Bigg(c_s^2 \delta_{\alpha\beta} - \delta_{\alpha\beta} u_\gamma u_\gamma + \frac{1}{2} \delta_{\alpha\beta} u_\gamma u_\gamma + u_\alpha u_\beta - \frac{1}{2} \delta_{\alpha\beta\gamma\delta} u_\gamma u_\delta \\
&\quad + \frac{1}{2} \delta_{\alpha\beta} u_\gamma u_\gamma + \frac{1}{2} \delta_{\alpha\beta\gamma\delta} u_\gamma u_\delta \Bigg), \\
&= \rho \left(c_s^2 \delta_{\alpha\beta} + u_\alpha u_\beta \right).
\end{aligned}$$

3.A.4 bcc_d3q15_ab

We have developed the following data for the bcc_d3q15_ab lattice. The basis vectors are;

$$\begin{aligned}
n_0 &= c^0 \\
n_1 &= c_x \\
n_2 &= c_y \\
n_3 &= c_z \\
n_4 &= c_x^2 \\
n_5 &= c_y^2 \\
n_6 &= c_z^2 \\
n_7 &= c_x c_y \\
n_8 &= c_x c_z \\
n_9 &= c_y c_z \\
n_{10} &= c_x c_y^2 \\
n_{11} &= c_y c_x^2 \\
n_{12} &= c_z c_x^2 \\
n_{13} &= c_x c_y c_z \\
n_{14} &= c_x^2 c_z^2
\end{aligned}$$

The T matrix is,

$$T = \mathrm{diag}(15, 16, 16, 16, 360, 192, 64, 8, 8, 8, 16, 16, 16, 8, 288).$$

```
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   .   2  -2   .   .   .   .   1   1   1   1  -1  -1  -1  -1
   .   .   .   2  -2   .   .   1   1  -1  -1   1   1  -1  -1
   .   .   .   .   .   2  -2   1  -1   1  -1   1  -1   1  -1
   .   4   4   .   .   .   .   1   1   1   1   1   1   1   1
   .   .   .   4   4   .   .   1   1   1   1   1   1   1   1
   .   .   .   .   .   4   4   1   1   1   1   1   1   1   1
   .   .   .   .   .   .   .   1   1  -1  -1  -1  -1   1   1
   .   .   .   .   .   .   .   1  -1   1  -1  -1   1  -1   1
   .   .   .   .   .   .   .   1  -1  -1   1   1  -1  -1   1
   .   .   .   .   .   .   .   1   1   1   1  -1  -1  -1  -1
   .   .   .   .   .   .   .   1   1  -1  -1   1   1  -1  -1
   .   .   .   .   .   .   .   1  -1   1  -1   1  -1   1  -1
   .   .   .   .   .   .   .   1  -1  -1   1  -1   1   1  -1
   .   .   .   .   .   .   .   1   1   1   1   1   1   1   1
 139 add operations
```

Table 3.36: Lattice vector matrix N for the `bcc_d3q15_ab` lattice.

```
   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .
   .   2   .   .   .   .   .   .   .   .   1   .   .   .   .
   .   .   2   .   .   .   .   .   .   .   .   1   .   .   .
   .   .   .   2   .   .   .   .   .   .   .   .   1   .   .
   .   .   .   .   4   .   .   .   .   .   .   .   .   .   1
   .   .   .   .   .   4   .   .   .   .   .   .   .   .   1
   .   .   .   .   .   .   4   .   .   .   .   .   .   .   1
   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .
   .   .   .   .   .   .   .   .   1   .   .   .   .   .   .
   .   .   .   .   .   .   .   .   .   1   .   .   .   .   .
   .   .   .   .   .   .   .   .   .   .   1   .   .   .   .
   .   .   .   .   .   .   .   .   .   .   .   1   .   .   .
   .   .   .   .   .   .   .   .   .   .   .   .   1   .   .
   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .
   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1
 6 add operations
 6 mul operations
```

Table 3.37: Lattice vector matrix N^0 for the `bcc_d3q15_ab` lattice.

```
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   .   1  -1   .   .   .   .   .   .   .   .   .   .   .   .
   .   .   .   1  -1   .   .   .   .   .   .   .   .   .   .
   .   .   .   .   .   1  -1   .   .   .   .   .   .   .   .
   .   1   1   .   .   .   .   .   .   .   .   .   .   .   .
   .   .   .   1   1   .   .   .   .   .   .   .   .   .   .
   .   .   .   .   .   1   1   .   .   .   .   .   .   .   .
   .   .   .   .   .   .   .   1   1  -1  -1  -1  -1   1   1
   .   .   .   .   .   .   .   1  -1   1  -1  -1   1  -1   1
   .   .   .   .   .   .   .   1  -1  -1   1   1  -1  -1   1
   .   .   .   .   .   .   .   1   1   1   1  -1  -1  -1  -1
   .   .   .   .   .   .   .   1   1  -1  -1   1   1  -1  -1
   .   .   .   .   .   .   .   1  -1   1  -1   1  -1   1  -1
   .   .   .   .   .   .   .   1  -1  -1   1  -1   1   1  -1
   .   .   .   .   .   .   .   1   1   1   1   1   1   1   1
91 add operations
```

Table 3.38: Lattice vector matrix N^1 for the `bcc_d3q15_ab` lattice.

```
   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .
   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .
   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .
   .   .   .   1   .   .   .   .   .   .   .   .   .   .   .
   .   .   .   .   1   1   1   .   .   .   .   .   .   .   .
   .   .   .   .   .   1   .   .   .   .   .   .   .   .   .
   .   .   .   .  -1   .   1   .   .   .   .   .   .   .   .
   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .
   .   .   .   .   .   .   .   .   1   .   .   .   .   .   .
   .   .   .   .   .   .   .   .   .   1   .   .   .   .   .
   .   .   .   .   .   .   .   .   .   .   1   .   .   .   .
   .   .   .   .   .   .   .   .   .   .   .   1   .   .   .
   .   .   .   .   .   .   .   .   .   .   .   .   1   .   .
   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .
   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1
3 add operations
```

Table 3.39: Transformation matrix Q^0 for the `bcc_d3q15_ab` lattice.

```
   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .
   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .
   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .
   .   .   .   1   .   .   .   .   .   .   .   .   .   .   .
 -16   .   .   .   5   .   .   .   .   .   .   .   .   .   .
   .   .   .   .  -1   3   .   .   .   .   .   .   .   .   .
   .   .   .   .   .   .   1   .   .   .   .   .   .   .   .
   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .
   .   .   .   .   .   .   .   .   1   .   .   .   .   .   .
   .   .   .   .   .   .   .   .   .   1   .   .   .   .   .
   .  -1   .   .   .   .   .   .   .   .   2   .   .   .   .
   .   .  -1   .   .   .   .   .   .   .   .   2   .   .   .
   .   .   .  -1   .   .   .   .   .   .   .   .   2   .   .
   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .
  -8   .   .   .   1   .   .   .   .   .   .   .   .   .   9
7 add operations
8 mul operations
```

Table 3.40: Transformation matrix Q^1 for the `bcc_d3q15_ab` lattice.

```
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   .   2  -2   .   .   .   .   1   1   1   1  -1  -1  -1  -1
   .   .   .   2  -2   .   .   1   1  -1  -1   1   1  -1  -1
   .   .   .   .   .   2  -2   1  -1   1  -1   1  -1   1  -1
 -16   4   4   4   4   4   4  -1  -1  -1  -1  -1  -1  -1  -1
   .  -4  -4   8   8  -4  -4   .   .   .   .   .   .   .   .
   .  -4  -4   .   .   4   4   .   .   .   .   .   .   .   .
   .   .   .   .   .   .   .   1   1  -1  -1  -1  -1   1   1
   .   .   .   .   .   .   .   1  -1   1  -1  -1   1  -1   1
   .   .   .   .   .   .   .   1  -1  -1   1   1  -1  -1   1
   .  -2   2   .   .   .   .   1   1   1   1  -1  -1  -1  -1
   .   .   .  -2   2   .   .   1   1  -1  -1   1   1  -1  -1
   .   .   .   .   .  -2   2   1  -1   1  -1   1  -1   1  -1
   .   .   .   .   .   .   .   1  -1  -1   1  -1   1   1  -1
  -8  -4  -4  -4  -4  -4  -4   4   4   4   4   4   4   4   4
147 add operations
44 mul operations
```

Table 3.41: Orthogonal basis matrix M for the `bcc_d3q15_ab` lattice.

The orthogonal vectors for the `bcc_d3q15_ab` lattice are;

$$\begin{aligned}
\psi_0 &= c^0 \\
\psi_1 &= c_x \\
\psi_2 &= c_y \\
\psi_3 &= c_z \\
\psi_4 &= 5c_\alpha^2 - 16c^0 \\
\psi_5 &= 3c_y^2 - c_\alpha^2 \\
\psi_6 &= c_z^2 - c_x^2 \\
\psi_7 &= c_x c_y \\
\psi_8 &= c_x c_z \\
\psi_9 &= c_y c_z \\
\psi_{10} &= 2c_x c_y^2 - c_x \\
\psi_{11} &= 2c_y c_x^2 - c_y \\
\psi_{12} &= 2c_z c_x^2 - c_z \\
\psi_{13} &= c_x c_y c_z \\
\psi_{14} &= 9c_x^2 c_z^2 + c_\alpha^2 - 8c^0
\end{aligned}$$

The resulting equilibrium moments for the `bcc_d3q15_ab` lattice are;

$$
\begin{aligned}
m_0^{eq} &= \rho \\
m_1^{eq} &= \rho u_x \\
m_2^{eq} &= \rho u_y \\
m_3^{eq} &= \rho u_z \\
m_4^{eq} &= 5\rho(u_x^2 + u_y^2 + u_z^2) - 6\rho \\
m_5^{eq} &= \rho(2u_y^2 - u_x^2 - u_z^2) \\
m_6^{eq} &= \rho(u_z^2 - u_x^2) \\
m_7^{eq} &= \rho u_x u_y \\
m_8^{eq} &= \rho u_x u_z \\
m_9^{eq} &= \rho u_y u_z \\
m_{10}^{eq} &= \frac{1}{3}\rho u_x \\
m_{11}^{eq} &= \frac{1}{3}\rho u_y \\
m_{12}^{eq} &= \frac{1}{3}\rho u_z \\
m_{13}^{eq} &= 0 \\
m_{14}^{eq} &= \rho(u_x^2 + u_y^2 + u_z^2) - 2\rho
\end{aligned}
$$

The orthogonal matrix required 147 add operations and 44 mul operations to calculate, a total of 191 operations. The factorized matrix $N^0 N^1 Q^0 Q^1$ requires $6 + 91 + 3 + 7 = 107$ add operations and $6 + 8 = 14$ mul operations to calculate, a total of 121 operations.

The temperature and speed of sound are constant for this lattice with values,

$$RT = c_s^2 = \frac{2}{3}.$$

This is the only value that allows the correct form for the third order moments,

$c_x c_y^2$ and $c_x c_z^2$ which are both equal to,

$$(0,0,0,0,0,0,0,1,1,1,1,-1,-1,-1,-1)\,.$$

The third order lattice vector c_x^3 is also dependent,

$$c_x^3 = 4c_x - 3c_x c_y^2.$$

In order for the third order equilibrium moments to take their correct values,

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} f_{\boldsymbol{i}}^{eq} = \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\beta}\delta_{\gamma\delta}\right) \rho c_s^2 u_\delta, \tag{3.27}$$

equation 3.27 implies a constraint on the speed of sound,

$$3c_s^2 = 4 - 3c_s^2, \quad c_s^2 = \frac{2}{3}.$$

The fourth order lattice vectors $c_x^2 c_z^2$, $c_x^2 c_y^2$ and $c_y^2 c_z^2$ are all equal to

$$(0,0,0,0,0,0,0,1,1,1,1,1,1,1,1)\,.$$

The conclusion from this is that the only form possible for the fourth order equilibrium moment is to be proportional to,

$$\left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\beta}\delta_{\gamma\delta}\right).$$

The fourth order vectors c_x^4, c_y^4 and c_z^4 are also dependent,

$$c_x^4 = 4c_x^2 - c_x^2 c_z^2,$$
$$c_y^4 = 4c_y^2 - c_x^2 c_z^2,$$
$$c_z^4 = 4c_z^2 - c_x^2 c_z^2.$$

The only solution is,

$$\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}\alpha} c_{\boldsymbol{i}\beta} c_{\boldsymbol{i}\gamma} c_{\boldsymbol{i}\delta} f_{\boldsymbol{i}}^{eq} = \left(\delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\beta}\delta_{\gamma\delta} + \delta_{\alpha\beta}\delta_{\gamma\delta}\right) \rho c_s^4.$$

The previously calculated value for the speed of sound works once again,

$$\begin{aligned}
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} f_{\boldsymbol{i}}^{eq} &= 3\rho c_s^4 = \frac{4}{3}\rho, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} c_{\boldsymbol{i}z} c_{\boldsymbol{i}z} f_{\boldsymbol{i}}^{eq} &= \rho c_s^4 = \frac{4}{9}\rho, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} f_{\boldsymbol{i}}^{eq} &= \rho c_s^2 = \frac{2}{3}\rho, \\
\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} f_{\boldsymbol{i}}^{eq} &= 4\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} f_{\boldsymbol{i}}^{eq} - 3\sum_{\boldsymbol{i}=0}^{q} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} c_{\boldsymbol{i}x} f_{\boldsymbol{i}}^{eq}.
\end{aligned}$$

The other directions can be similarly verified.

3.A.5 `cub_d3q27_abc`

We have developed the following data for the `cub_d3q27_abc` lattice. The basis vectors are;

$$n_0 = c^0$$

$$n_1 = c_x$$

$$n_2 = c_y$$

$$n_3 = c_z$$

$$n_4 = c_x^2$$

$$n_5 = c_y^2$$

$$n_6 = c_z^2$$

$$n_7 = c_x c_y$$

$$n_8 = c_x c_z$$

$$n_9 = c_y c_z$$

$$n_{10} = c_x c_y^2$$

$$n_{11} = c_x c_z^2$$

$$n_{12} = c_y c_x^2$$

$$n_{13} = c_y c_z^2$$

$$n_{14} = c_z c_x^2$$

$$n_{15} = c_z c_y^2$$

$$n_{16} = c_x c_y c_z$$

$$n_{17} = c_x^2 c_y^2$$

$$n_{18} = c_x^2 c_z^2$$

$$n_{19} = c_y^2 c_z^2$$

$$n_{20} = c_x^2 c_y c_z$$

$$n_{21} = c_y^2 c_x c_z$$

$$n_{22} = c_z^2 c_x c_y$$

$$n_{23} = c_x^2 c_y^2 c_z$$

$$n_{24} = c_x^2 c_z^2 c_y$$

$$n_{25} = c_y^2 c_z^2 c_x$$

$$n_{26} = c_x^2 c_y^2 c_z^2$$

The base lattice vector matrix N is shown in table 3.42.

```
  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
  .  1 -1  .  .  .  .  1 -1 -1  1  1 -1 -1  1  .  .  .  .  1 -1  1 -1  1 -1 -1  1
  .  .  .  1 -1  .  .  1 -1  1 -1  .  .  .  .  1 -1 -1  1  1 -1  1 -1 -1  1  1 -1
  .  .  .  .  .  1 -1  .  .  .  .  1 -1  1 -1  1 -1  1 -1  1 -1 -1  1  1 -1  1 -1
  .  1  1  .  .  .  .  1  1  1  1  1  1  1  1  .  .  .  .  1  1  1  1  1  1  1  1
  .  .  .  1  1  .  .  1  1  1  1  .  .  .  .  1  1  1  1  1  1  1  1  1  1  1  1
  .  .  .  .  .  1  1  .  .  .  .  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
  .  .  .  .  .  .  .  1  1 -1 -1  .  .  .  .  .  .  .  .  1  1  1  1 -1 -1 -1 -1
  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1  .  .  .  .  1  1 -1 -1  1  1 -1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1  1  1 -1 -1 -1 -1  1  1
  .  .  .  .  .  .  .  1 -1 -1  1  .  .  .  .  .  .  .  .  1 -1  1 -1  1 -1 -1  1
  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1  .  .  .  .  1 -1  1 -1  1 -1 -1  1
  .  .  .  .  .  .  .  1 -1  1 -1  .  .  .  .  .  .  .  .  1 -1  1 -1 -1  1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1  1 -1  1 -1 -1  1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  1 -1  1 -1  .  .  .  .  1 -1 -1  1  1 -1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1  1 -1  1 -1 -1  1  1 -1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1 -1  1 -1  1
  .  .  .  .  .  .  .  1  1  1  1  .  .  .  .  .  .  .  .  1  1  1  1  1  1  1  1
  .  .  .  .  .  .  .  .  .  .  .  1  1  1  1  .  .  .  .  1  1  1  1  1  1  1  1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1  1  1  1  1  1  1  1  1  1  1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1 -1 -1  1  1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1  1  1 -1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1  1  1 -1 -1 -1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1  1 -1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1  1 -1 -1  1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1  1 -1  1 -1 -1  1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1  1  1  1  1  1  1
343 add operations
```

Table 3.42: Lattice vector matrix N for the `cub_d3q27_abc` lattice.

```
 1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
 .  1  .  .  .  .  .  .  .  .  1  1  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .
 .  .  1  .  .  .  .  .  .  .  .  .  1  1  .  .  .  .  .  .  .  .  .  .  1  .  .
 .  .  .  1  .  .  .  .  .  .  .  .  .  .  1  1  .  .  .  .  .  .  .  1  .  .  .
 .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  1  1  .  .  .  .  .  .  .  1
 .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  1  .  1  .  .  .  .  .  .  1
 .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  1  1  .  .  .  .  .  .  1
 .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .
 .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .
 .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .
 .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .
 .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .
 .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  1  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  1  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  1  .  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  1  .  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  1
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  1
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  1
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1
30 add operations
```

Table 3.43: Matrix N^0 for the `cub_d3q27_abc` lattice.

```
  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
  .  1 -1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  1 -1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  1 -1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  1  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  1  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  1  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  1  1 -1 -1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  1 -1 -1  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  1 -1  1 -1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  1 -1  1 -1  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1  1 -1  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1 -1  1 -1  1
  .  .  .  .  .  .  .  1  1  1  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  1  1  1  1  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1  1  1  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1 -1 -1  1  1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1  1  1 -1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1  1  1 -1 -1 -1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1  1 -1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1  1 -1 -1  1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1  1 -1  1 -1 -1  1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1  1  1  1  1  1  1
151 add operations
```

Table 3.44: Matrix N^1 for the `cub_d3q27_abc` lattice.

```
  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
 -2  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
 -2  .  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
 -2  .  .  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  . -2  .  .  .  .  .  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  . -2  .  .  .  .  .  .  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  . -2  .  .  .  .  .  .  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  . -2  .  .  .  .  .  .  .  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  . -2  .  .  .  .  .  .  .  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .  .
  .  .  . -2  .  .  .  .  .  .  .  .  .  .  .  3  .  .  .  .  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  .  .  .  .  .  .  .  .  .  .
  4  .  .  . -6 -6  .  .  .  .  .  .  .  .  .  .  .  9  .  .  .  .  .  .  .  .  .
  4  .  .  . -6  . -6  .  .  .  .  .  .  .  .  .  .  .  9  .  .  .  .  .  .  .  .
  4  .  .  .  . -6 -6  .  .  .  .  .  .  .  .  .  .  .  .  9  .  .  .  .  .  .  .
  .  .  .  .  .  .  .  .  . -6  .  .  .  .  .  .  .  .  .  .  9  .  .  .  .  .  .
  .  .  .  .  .  .  .  . -6  .  .  .  .  .  .  .  .  .  .  .  .  9  .  .  .  .  .
  .  .  .  .  .  .  . -6  .  .  .  .  .  .  .  .  .  .  .  .  .  .  9  .  .  .  .
  .  .  .  4  .  .  .  .  .  .  .  .  .  . -6 -6  .  .  .  .  .  .  .  9  .  .  .
  .  .  4  .  .  .  .  .  .  .  .  . -6 -6  .  .  .  .  .  .  .  .  .  .  9  .  .
  .  4  .  .  .  .  .  .  .  . -6 -6  .  .  .  .  .  .  .  .  .  .  .  .  .  9  .
 -8  .  .  . 12 12 12  .  .  .  .  .  .  .  .  .  .-18-18-18  .  .  .  .  .  . 27
37 add operations
56 mul operations
```

Table 3.45: Transformation matrix Q for the `cub_d3q27_abc` lattice.

```
  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
  .  1 -1  .  .  .  .  1 -1 -1  1  1 -1 -1  1  .  .  .  .  1 -1  1 -1  1 -1 -1  1
  .  .  .  1 -1  .  .  1 -1  1 -1  .  .  .  .  1 -1 -1  1  1 -1  1 -1 -1  1  1 -1
  .  .  .  .  .  1 -1  .  .  .  .  1 -1  1 -1  1 -1  1 -1  1 -1 -1  1  1 -1  1 -1
 -2  1  1 -2 -2 -2 -2  1  1  1  1  1  1  1  1 -2 -2 -2 -2  1  1  1  1  1  1  1  1
 -2 -2 -2  1  1 -2 -2  1  1  1  1 -2 -2 -2 -2  1  1  1  1  1  1  1  1  1  1  1  1
 -2 -2 -2 -2 -2  1  1 -2 -2 -2 -2  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
  .  .  .  .  .  .  .  1  1 -1 -1  .  .  .  .  .  .  .  .  1  1  1  1 -1 -1 -1 -1
  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1  .  .  .  .  1  1 -1 -1  1  1 -1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1  1 -1 -1  1  1 -1 -1 -1 -1  1  1
  . -2  2  .  .  .  .  1 -1 -1  1 -2  2  2 -2  .  .  .  .  1 -1  1 -1  1 -1 -1  1
  . -2  2  .  .  .  . -2  2  2 -2  1 -1 -1  1  .  .  .  .  1 -1  1 -1  1 -1 -1  1
  .  .  . -2  2  .  .  1 -1  1 -1  .  .  .  . -2  2  2 -2  1 -1  1 -1 -1  1  1 -1
  .  .  . -2  2  .  . -2  2 -2  2  .  .  .  .  1 -1 -1  1  1 -1  1 -1 -1  1  1 -1
  .  .  .  .  . -2  2  .  .  .  .  1 -1  1 -1 -2  2 -2  2  1 -1 -1  1  1 -1  1 -1
  .  .  .  .  . -2  2  .  .  .  . -2  2 -2  2  1 -1  1 -1  1 -1 -1  1  1 -1  1 -1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  1 -1 -1  1 -1  1 -1  1
  4 -2 -2 -2 -2  4  4  1  1  1  1 -2 -2 -2 -2 -2 -2 -2 -2  1  1  1  1  1  1  1  1
  4 -2 -2  4  4 -2 -2 -2 -2 -2 -2  1  1  1  1 -2 -2 -2 -2  1  1  1  1  1  1  1  1
  4  4  4 -2 -2 -2 -2 -2 -2 -2 -2 -2 -2 -2 -2  1  1  1  1  1  1  1  1  1  1  1  1
  .  .  .  .  .  .  .  .  .  .  .  .  .  .  . -6 -6  6  6  3  3 -3 -3 -3 -3  3  3
  .  .  .  .  .  .  .  .  .  .  . -6 -6  6  6  .  .  .  .  3  3 -3 -3  3  3 -3 -3
  .  .  .  .  .  .  . -6 -6  6  6  .  .  .  .  .  .  .  .  3  3  3  3 -3 -3 -3 -3
  .  .  .  .  .  4 -4  .  .  .  . -2  2 -2  2 -2  2 -2  2  1 -1 -1  1  1 -1  1 -1
  .  .  .  4 -4  .  . -2  2 -2  2  .  .  .  . -2  2  2 -2  1 -1  1 -1 -1  1  1 -1
  .  4 -4  .  .  .  . -2  2  2 -2 -2  2  2 -2  .  .  .  .  1 -1  1 -1  1 -1 -1  1
 -8  4  4  4  4  4  4 -2 -2 -2 -2 -2 -2 -2 -2 -2 -2 -2 -2  1  1  1  1  1  1  1  1
512 add operations
193 mul operations
```

Table 3.46: Orthogonal basis matrix M for the `cub_d3q27_abc` lattice.

The T matrix is,

$$T = \text{diag}(27, 18, 18, 18, 54, 54, 54, 12, 12, 12, 36, 36, 36, 36, 36, 36,$$
$$8, 108, 108, 108, 216, 216, 216, 72, 72, 72, 216).$$

The orthogonal vectors for the `cub_d3q27_abc` lattice are;

$$\psi_0 = c^0$$
$$\psi_1 = c_x$$
$$\psi_2 = c_y$$
$$\psi_3 = c_z$$
$$\psi_4 = 3c_x^2 - 2\Delta x^2 c^0$$
$$\psi_5 = 3c_y^2 - 2\Delta y^2 c^0$$
$$\psi_6 = 3c_z^2 - 2\Delta z^2 c^0$$
$$\psi_7 = c_x c_y$$
$$\psi_8 = c_x c_z$$
$$\psi_9 = c_y c_z$$

$$\psi_{10} = 3c_x c_y^2 - 2\Delta y^2 c_x$$
$$\psi_{11} = 3c_x c_z^2 - 2\Delta z^2 c_x$$
$$\psi_{12} = 3c_y c_x^2 - 2\Delta x^2 c_y$$
$$\psi_{13} = 3c_y c_z^2 - 2\Delta z^2 c_y$$
$$\psi_{14} = 3c_z c_x^2 - 2\Delta x^2 c_z$$
$$\psi_{15} = 3c_z c_y^2 - 2\Delta y^2 c_z$$
$$\psi_{16} = c_x c_y c_z$$

$$\psi_{17} = 9c_x^2c_y^2 - 6\Delta x^2c_y^2 - 6\Delta y^2c_x^2 + 4\Delta x^2\Delta y^2$$

$$\psi_{18} = 9c_x^2c_z^2 - 6\Delta x^2c_z^2 - 6\Delta z^2c_x^2 + 4\Delta x^2\Delta z^2$$

$$\psi_{19} = 9c_y^2c_z^2 - 6\Delta y^2c_z^2 - 6\Delta z^2c_y^2 + 4\Delta y^2\Delta z^2$$

$$\psi_{20} = 9c_x^2c_yc_z - 6\Delta x^2c_yc_z$$

$$\psi_{21} = 9c_y^2c_xc_z - 6\Delta y^2c_xc_z$$

$$\psi_{22} = 9c_z^2c_xc_y - 6\Delta z^2c_xc_y$$

$$\psi_{23} = 9c_x^2c_y^2c_z - 6\Delta x^2c_y^2c_z - 6\Delta y^2c_x^2c_z + 4\Delta x^2\Delta y^2c_z$$

$$\psi_{24} = 9c_x^2c_z^2c_y - 6\Delta x^2c_z^2c_y - 6\Delta z^2c_x^2c_y + 4\Delta x^2\Delta z^2c_y$$

$$\psi_{25} = 9c_y^2c_z^2c_x - 6\Delta y^2c_z^2c_x - 6\Delta z^2c_y^2c_x + 4\Delta y^2\Delta z^2c_x$$

$$\begin{aligned}\psi_{26} = 27c_x^2c_y^2c_z^2 - 8\Delta x^2\Delta y^2\Delta z^2c^0 \\ - 18(\Delta x^2c_y^2c_z^2 + \Delta y^2c_x^2c_z^2 + \Delta z^2c_x^2c_y^2) \\ + 12(\Delta x^2\Delta y^2c_z^2 + \Delta x^2\Delta z^2c_y^2 + \Delta y^2\Delta z^2c_x^2)\end{aligned}$$

The resulting equilibrium moments for the `cub_d3q27_abc` lattice are;

$$m_0^{eq} = \rho$$

$$m_1^{eq} = \rho u_x$$

$$m_2^{eq} = \rho u_y$$

$$m_3^{eq} = \rho u_z$$

$$m_4^{eq} = \rho(3c_s^2 - 2\Delta x^2) + 3\rho u_xu_x$$

$$m_5^{eq} = \rho(3c_s^2 - 2\Delta y^2) + 3\rho u_yu_y$$

$$m_6^{eq} = \rho(3c_s^2 - 2\Delta z^2) + 3\rho u_zu_z$$

$$m_7^{eq} = \rho u_xu_y$$

$$m_8^{eq} = \rho u_xu_z$$

$$m_9^{eq} = \rho u_yu_z$$

$$
\begin{aligned}
m_{10}^{eq} &= \rho u_x(3\gamma_1 - 2\Delta y^2)\\
m_{11}^{eq} &= \rho u_x(3\gamma_1 - 2\Delta z^2)\\
m_{12}^{eq} &= \rho u_y(3\gamma_1 - 2\Delta x^2)\\
m_{13}^{eq} &= \rho u_y(3\gamma_1 - 2\Delta z^2)\\
m_{14}^{eq} &= \rho u_z(3\gamma_1 - 2\Delta x^2)\\
m_{15}^{eq} &= \rho u_z(3\gamma_1 - 2\Delta y^2)\\
m_{16}^{eq} &= 0\\
m_{17}^{eq} &= \rho(9\gamma_2 - 6c_s^2(\Delta x^2 + \Delta y^2) + 4\Delta x^2\Delta y^2)\\
&\quad + \rho u_x u_x(9\gamma_3 - 6\Delta dy^2) + \rho u_y u_y(9\gamma_3 - 6\Delta dx^2)\\
m_{18}^{eq} &= \rho(9\gamma_2 - 6c_s^2(\Delta x^2 + \Delta z^2) + 4\Delta x^2\Delta z^2)\\
&\quad + \rho u_x u_x(9\gamma_3 - 6\Delta z^2) + \rho u_z u_z(9\gamma_3 - 6\Delta x^2)\\
m_{19}^{eq} &= \rho(9\gamma_2 - 6c_s^2(\Delta y^2 + \Delta z^2) + 4\Delta y^2\Delta z^2)\\
&\quad + \rho u_y u_y(9\gamma_3 - 6\Delta z^2) + \rho u_z u_z(9\gamma_3 - 6\Delta y^2)\\
m_{20}^{eq} &= \rho u_y u_z(9\gamma_4 - 6\Delta x^2)\\
m_{21}^{eq} &= \rho u_x u_z(9\gamma_4 - 6\Delta y^2)\\
m_{22}^{eq} &= \rho u_x u_y(9\gamma_4 - 6\Delta z^2)\\
m_{23}^{eq} &= \rho u_z(9\gamma_5 - 6\gamma_1(\Delta x^2 + \Delta y^2) + 4\Delta x^2\Delta y^2)\\
m_{24}^{eq} &= \rho u_y(9\gamma_5 - 6\gamma_1(\Delta x^2 + \Delta z^2) + 4\Delta x^2\Delta z^2)\\
m_{25}^{eq} &= \rho u_x(9\gamma_5 - 6\gamma_1(\Delta y^2 + \Delta z^2) + 4\Delta y^2\Delta z^2)\\
m_{26}^{eq} &= \rho[27\gamma_6 - 18\gamma_2(\Delta x^2 + \Delta y^2 + \Delta z^2) - 8\Delta x^2\Delta y^2\Delta z^2\\
&\quad + 12c_s^2(\Delta x^2\Delta y^2 + \Delta x^2\Delta z^2 + \Delta y^2\Delta z^2)]\\
&\quad + \rho u_x u_x(27\gamma_7 - 18\gamma_3(\Delta y^2 + \Delta z^2) + 12\Delta y^2\Delta z^2)\\
&\quad + \rho u_y u_y(27\gamma_7 - 18\gamma_3(\Delta x^2 + \Delta z^2) + 12\Delta x^2\Delta z^2)\\
&\quad + \rho u_z u_z(27\gamma_7 - 18\gamma_3(\Delta x^2 + \Delta y^2) + 12\Delta x^2\Delta y^2)
\end{aligned}
$$

The orthogonal matrix required 512 add operations and 193 mul operations to calculate, a total of 705 operations. The factorized matrix N^0N^1Q requires $30 + 151 + 37 = 218$ add operations and 56 mul operations to calculate, a total of 274 operations.

The temperature and speed of sound are constant for this lattice with values,

$$RT = c_s^2 = \frac{1}{3}.$$

3.A.6 `bcc_d3q27_abc`

We have developed the following data for the `bcc_d3q27_abc` lattice. The basis vectors are;

$$
\begin{aligned}
n_0 &= c^0 \\
n_1 &= c_x \\
n_2 &= c_y \\
n_3 &= c_z \\
n_4 &= c_x^2 \\
n_5 &= c_y^2 \\
n_6 &= c_z^2 \\
n_7 &= c_x c_y \\
n_8 &= c_x c_z \\
n_9 &= c_y c_z \\
n_{10} &= c_x^3 \\
n_{11} &= c_x c_z^2 \\
n_{12} &= c_x c_y^2 \\
n_{13} &= c_y^3 \\
n_{14} &= c_y c_x^2 \\
n_{15} &= c_y c_z^2 \\
n_{16} &= c_z^3 \\
n_{17} &= c_z c_y^2 \\
n_{18} &= c_z c_x^2
\end{aligned}
$$

$$
\begin{aligned}
n_{19} &= c_x c_y c_z \\
n_{20} &= (c_x^2 + c_y^2 + c_z^2) c_x^2 \\
n_{21} &= (c_x^2 + c_y^2 + c_z^2) c_y^2 \\
n_{22} &= (c_x^2 + c_y^2 + c_z^2) c_z^2 \\
n_{23} &= (c_x^2 + c_y^2 + c_z^2) c_x c_y \\
n_{24} &= (c_x^2 + c_y^2 + c_z^2) c_x c_z \\
n_{25} &= (c_x^2 + c_y^2 + c_z^2) c_y c_z \\
n_{26} &= c_x^2 c_y^2 + c_x^2 c_z^2 + c_y^2 c_z^2
\end{aligned}
$$

The base lattice vector matrix N is shown in table 3.47.

```
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
  .   2  -2   .   .   .   .   2  -2  -2   2   2  -2  -2   2   .   .   .   .   1  -1   1  -1   1  -1  -1   1
  .   .   .   2  -2   .   .   2  -2   2  -2   .   .   .   .   2  -2  -2   2   1  -1   1  -1  -1   1   1  -1
  .   .   .   .   .   2  -2   .   .   .   .   2  -2   2  -2   2  -2   2  -2   1  -1  -1   1   1  -1   1  -1
  .   4   4   .   .   .   .   4   4   4   4   4   4   4   4   .   .   .   .   1   1   1   1   1   1   1   1
  .   .   .   4   4   .   .   4   4   4   4   .   .   .   .   4   4   4   4   1   1   1   1   1   1   1   1
  .   .   .   .   .   4   4   .   .   .   .   4   4   4   4   4   4   4   4   1   1   1   1   1   1   1   1
  .   .   .   .   .   .   .   4   4  -4  -4   .   .   .   .   .   .   .   .   1   1   1   1  -1  -1  -1  -1
  .   .   .   .   .   .   .   .   .   .   .   4   4  -4  -4   .   .   .   .   1   1  -1  -1   1   1  -1  -1
  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   4   4  -4  -4   1   1  -1  -1  -1  -1   1   1
  .   8  -8   .   .   .   .   8  -8  -8   8   8  -8  -8   8   .   .   .   .   1  -1   1  -1   1  -1  -1   1
  .   .   .   .   .   .   .   .   .   .   .   8  -8  -8   8   .   .   .   .   1  -1   1  -1   1  -1  -1   1
  .   .   .   .   .   .   .   8  -8  -8   8   .   .   .   .   .   .   .   .   1  -1   1  -1   1  -1  -1   1
  .   .   .   8  -8   .   .   8  -8   8  -8   .   .   .   .   8  -8  -8   8   1  -1   1  -1  -1   1   1  -1
  .   .   .   .   .   .   .   8  -8   8  -8   .   .   .   .   .   .   .   .   1  -1   1  -1  -1   1   1  -1
  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   8  -8  -8   8   1  -1   1  -1  -1   1   1  -1
  .   .   .   .   .   8  -8   .   .   .   .   8  -8   8  -8   8  -8   8  -8   1  -1  -1   1   1  -1   1  -1
  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   8  -8   8  -8   1  -1  -1   1   1  -1   1  -1
  .   .   .   .   .   .   .   .   .   .   .   8  -8   8  -8   .   .   .   .   1  -1  -1   1   1  -1   1  -1
  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1  -1  -1   1  -1   1  -1   1
  .  16  16   .   .   .   .  32  32  32  32  32  32  32  32   .   .   .   .   3   3   3   3   3   3   3   3
  .   .   .  16  16   .   .  32  32  32  32   .   .   .   .  32  32  32  32   3   3   3   3   3   3   3   3
  .   .   .   .   .  16  16   .   .   .   .  32  32  32  32  32  32  32  32   3   3   3   3   3   3   3   3
  .   .   .   .   .   .   .  32  32 -32 -32   .   .   .   .   .   .   .   .   3   3   3   3  -3  -3  -3  -3
  .   .   .   .   .   .   .   .   .   .   .  32  32 -32 -32   .   .   .   .   3   3  -3  -3   3   3  -3  -3
  .   .   .   .   .   .   .   .   .   .   .   .   .   .   .  32  32 -32 -32   3   3  -3  -3  -3  -3   3   3
  .   .   .   .   .   .   .  16  16  16  16  16  16  16  16  16  16  16  16   3   3   3   3   3   3   3   3
415 add operations
236 mul operations
```

Table 3.47: Lattice vector matrix N for the `bcc_d3q27_abc` lattice.

```
    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    2    .    .    .    .    .    .    .    .    1    2    2    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    2    .    .    .    .    .    .    .    .    .    .    1    2    2    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    2    .    .    .    .    .    .    .    .    .    .    .    .    1    2    2    .    .    .    .    .    .    .    .
    .    .    .    .    4    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    4    .    .    .    .    .    1
    .    .    .    .    .    4    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    4    .    .    .    .    1
    .    .    .    .    .    .    4    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    4    .    .    .    1
    .    .    .    .    .    .    .    4    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .    .    .
    .    .    .    .    .    .    .    .    4    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .    .
    .    .    .    .    .    .    .    .    .    4    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .
    .    8    .    .    .    .    .    .    .    .    1    8    8    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    1    8    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    1    .    8    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    8    .    .    .    .    .    .    .    .    .    .    1    8    8    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    1    8    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .    8    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    8    .    .    .    .    .    .    .    .    .    .    .    .    1    8    8    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    8    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .    8    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .    .    .    .    .    .    .
    .    .    .    .   16    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .   32    .    .    .    .    .    3
    .    .    .    .    .   16    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .   32    .    .    .    .    3
    .    .    .    .    .    .   16    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .   32    .    .    .    3
    .    .    .    .    .    .    .   32    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    3    .    .    .
    .    .    .    .    .    .    .    .   32    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    3    .    .
    .    .    .    .    .    .    .    .    .   32    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    3    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    8    8    8    .    .    .    3
45 add operations
52 mul operations
```

Table 3.48: Matrix N^1 for the `bcc_d3q27_abc` lattice.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
.	1	-1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	1	-1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	1	-1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	1	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	1	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	1	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	1	1	-1	-1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	1	1	-1	-1	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	1	-1	-1	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	-1	1	-1	1	-1	-1	1
.	.	.	.	.	.	.	.	.	.	.	1	-1	-1	1	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	1	-1	-1	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	-1	1	-1	-1	1	1	-1
.	.	.	.	.	.	.	1	-1	1	-1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	-1	-1	1	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	-1	-1	1	1	-1	1	-1
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	-1	1	-1	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	1	-1	1	-1	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	-1	-1	1	-1	1	-1	1
.	.	.	.	.	.	.	1	1	1	1	1	1	1	1	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	1	1	1	1	.	.	.	.	1	1	1	1	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	1	1	1	1	1	1	1	1	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	1	1	1	-1	-1	-1	-1
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	1	-1	-1	1	1	-1	-1
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	1	-1	-1	-1	-1	1	1
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	1	1	1	1	1	1	1

`163 add operations`

Table 3.49: Matrix N^0 for the `bcc_d3q27_abc` lattice.

```
 1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   1   1   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .  -1   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   1   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .  -1   1   .   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   1   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .  -1   1   .   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   1   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .  -1   1   .   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .   .   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   1   1   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .  -1   .   1   .   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1   .
 .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   1
12 add operations
```

Table 3.50: Transformation matrix Q^0 for the `bcc_d3q27_abc` lattice.

```
    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
  -16    .    .    .    3    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .   -1    3    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .   -7    .    .    .    .    .    .    .    .    2    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .   -8    .    .    .    .    .    .    .    .   -2    5    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    1    .    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .   -7    .    .    .    .    .    .    .    .    .    .    2    .    .    .    .    .    .    .    .    .    .    .    .    .
    .    .   -8    .    .    .    .    .    .    .    .    .    .   -2    5    .    .    .    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .    .    .    .    .    .    .    .    .    .    .
    .    .    .   -7    .    .    .    .    .    .    .    .    .    .    .    .    2    .    .    .    .    .    .    .    .    .    .
    .    .    .   -8    .    .    .    .    .    .    .    .    .    .    .    .   -2    5    .    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .    .    .    .    .    .    .    .
    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    1    .    .    .    .    .    .    .
  440    .    .    . -219    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .   21    .    .    .    .    .    .
    .    .    .    .   20  -60    .    .    .    .    .    .    .    .    .    .    .    .    .    .   -3    9    .    .    .    .    .
    .    .    .    .    .    .  -20    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    3    .    .    .    .
    .    .    .    .    .    .    .  -67    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    9    .    .    .
    .    .    .    .    .    .    .    .  -67    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    9    .    .
    .    .    .    .    .    .    .    .    .  -67    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .    9    .
 -320    .    .    .  228    .    .    .    .    .    .    .    .    .    .    .    .    .    .    .  -57    .    .    .    .    .  135
23 add operations
37 mul operations
```

Table 3.51: Transformation matrix Q^1 for the bcc_d3q27_abc lattice.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
.	2	-2	.	.	.	.	2	-2	-2	2	2	-2	-2	2	.	.	.	.	1	-1	1	-1	1	-1	-1	1
.	.	.	2	-2	.	.	2	-2	2	-2	.	.	.	.	2	-2	-2	2	1	-1	1	-1	-1	1	1	-1
.	.	.	.	.	2	-2	.	.	.	.	2	-2	2	-2	2	-2	2	-2	1	-1	-1	1	1	-1	1	-1
-16	-4	-4	-4	-4	-4	-4	8	8	8	8	8	8	8	8	8	8	8	8	-7	-7	-7	-7	-7	-7	-7	-7
.	-4	-4	8	8	-4	-4	4	4	4	4	-8	-8	-8	-8	4	4	4	4	.	.	.	.	.	.	.	.
.	-4	-4	.	.	4	4	-4	-4	-4	-4	.	.	.	.	4	4	4	4	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	4	4	-4	-4	.	.	.	.	.	.	.	.	1	1	1	1	-1	-1	-1	-1
.	.	.	.	.	.	.	.	.	.	.	4	4	-4	-4	.	.	.	.	1	1	-1	-1	1	1	-1	-1
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	4	4	-4	-4	1	1	-1	-1	-1	-1	1	1
.	2	-2	.	.	.	.	2	-2	-2	2	2	-2	-2	2	.	.	.	.	-5	5	-5	5	-5	5	5	-5
.	-32	32	.	.	.	.	8	-8	-8	8	8	-8	-8	8	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	8	-8	-8	8	-8	8	8	-8	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	2	-2	.	.	2	-2	2	-2	.	.	.	.	2	-2	-2	2	-5	5	-5	5	5	-5	-5	5
.	.	.	-32	32	.	.	8	-8	8	-8	.	.	.	.	8	-8	-8	8	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	-8	8	-8	8	.	.	.	.	8	-8	-8	8	.	.	.	.	.	.	.	.
.	.	.	.	.	2	-2	.	.	.	.	2	-2	2	-2	2	-2	2	-2	-5	5	5	-5	-5	5	-5	5
.	.	.	.	.	-32	32	.	.	.	.	8	-8	8	-8	8	-8	8	-8	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	8	-8	8	-8	-8	8	-8	8	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	-1	-1	1	-1	1	-1	1
440	-100	-100	-100	-100	-100	-100	32	32	32	32	32	32	32	32	32	32	32	32	-28	-28	-28	-28	-28	-28	-28	-28
.	32	32	-64	-64	32	32	16	16	16	16	-32	-32	-32	-32	16	16	16	16	.	.	.	.	.	.	.	.
.	32	32	.	.	-32	-32	-16	-16	-16	-16	.	.	.	.	16	16	16	16	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	20	20	-20	-20	.	.	.	.	.	.	.	.	-40	-40	-40	-40	40	40	40	40
.	.	.	.	.	.	.	.	.	.	.	20	20	-20	-20	.	.	.	.	-40	-40	40	40	-40	-40	40	40
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	20	20	-20	-20	-40	-40	40	40	40	40	-40	-40
-320	-320	-320	-320	-320	-320	-320	16	16	16	16	16	16	16	16	16	16	16	16	256	256	256	256	256	256	256	256

```
410 add operations
327 mul operations
```

Table 3.52: Orthogonal basis matrix M for the `bcc_d3q27_abc` lattice.

The T matrix is,

$$
\begin{aligned}
T = \mathrm{diag}(&27, 48, 48, 48, 1\,512, 576, 192, 72, 72, 72,\\
&240, 2\,560, 512, 240, 2\,560, 512, 240, 2\,560, 512,\\
&8, 272\,160, 18\,432, 6\,144, 14\,400, 14\,400, 14\,400, 1\,244\,160).
\end{aligned}
$$

The orthogonal vectors for the `bcc_d3q27_abc` lattice are;

$$
\begin{aligned}
\psi_0 &= c^0 & \psi_{10} &= 2c_x^3 - 7c_x\\
\psi_1 &= c_x & \psi_{11} &= 5(c_x c_y^2 + c_x c_z^2) - 2c_x^3 - 8c_x\\
\psi_2 &= c_y & \psi_{12} &= c_x c_y^2 - c_x c_z^2\\
\psi_3 &= c_z & \psi_{13} &= 2c_y^3 - 7c_y\\
\psi_4 &= 3c_\alpha^2 - 16c^0 & \psi_{14} &= 5(c_y c_z^2 + c_y c_x^2) - 2c_y^3 - 8c_y\\
\psi_5 &= 3c_y^2 - c_\alpha^2 & \psi_{15} &= c_y c_z^2 - c_y c_x^2\\
\psi_6 &= c_z^2 - c_y^2 & \psi_{16} &= 2c_z^3 - 7c_z\\
\psi_7 &= c_x c_y & \psi_{17} &= 5(c_z c_x^2 + c_z c_y^2) - 2c_z^3 - 8c_z\\
\psi_8 &= c_x c_z & \psi_{18} &= c_z c_x^2 - c_z c_y^2\\
\psi_9 &= c_y c_z & \psi_{19} &= c_x c_y c_z
\end{aligned}
$$

$$\psi_{20} = 21c_\alpha^4 - 219c_\alpha^2 + 440c_0$$

$$\psi_{21} = 9c_\alpha^2 c_y^2 - 3c_\alpha^4 - 60c_y^2 + 20c_\alpha^2$$

$$= (3c_\alpha^2 - 20)(3c_y^2 - c_\alpha^2)$$

$$\psi_{22} = (3c_\alpha^2 - 20)(c_z^2 - c_x^2)$$

$$\psi_{23} = (9c_\alpha^2 - 67)c_x c_y$$

$$\psi_{24} = (9c_\alpha^2 - 67)c_x c_z$$

$$\psi_{25} = (9c_\alpha^2 - 67)c_y c_z$$

$$\psi_{26} = 135(c_x^2 c_y^2 + c_x^2 c_z^2 + c_y^2 c_z^2) - 57c_\alpha^4 + 228c_\alpha^2 - 320c_0$$

The resulting equilibrium moments are;

$$m_0^{eq} = \rho$$

$$m_1^{eq} = \rho u_x$$

$$m_2^{eq} = \rho u_y$$

$$m_3^{eq} = \rho u_z$$

$$m_4^{eq} = 3(3\rho RT + \rho u_x^2 + \rho u_y^2 + \rho u_z^2) - 16\rho$$

$$m_5^{eq} = 2\rho u_y^2 - \rho u_x^2 - \rho u_z^2$$

$$m_6^{eq} = \rho u_z^2 - \rho u_y^2$$

$$m_7^{eq} = \rho u_x u_y$$

$$m_8^{eq} = \rho u_x u_z$$

$$m_9^{eq} = \rho u_y u_z$$

$$m_{10}^{eq} = (6RT - 7)\rho u_x + 2\rho u_x^3$$

$$m_{11}^{eq} = (4RT - 8)\rho u_x + 5\rho u_x u_y^2 + 5\rho u_x u_z^2 - 2\rho u_x^3$$

$$m_{12}^{eq} = \rho u_x(u_y^2 - u_z^2)$$

$$m_{13}^{eq} = (6RT - 7)\rho u_y + 2\rho u_y^3$$

$$m_{14}^{eq} = (4RT - 8)\rho u_y + 5\rho u_y u_z^2 + 5\rho u_y u_x^2 - 2\rho u_y^3$$

$$m_{15}^{eq} = \rho u_y(u_z^2 - u_x^2)$$

$$
\begin{aligned}
m_{16}^{eq} &= (6RT-7)\rho u_z + 2\rho u_z^3 \\
m_{17}^{eq} &= (4RT-8)\rho u_z + 5\rho u_z u_x^2 + 5\rho u_z u_y^2 - 2\rho u_z^3 \\
m_{18}^{eq} &= \rho u_z(u_x^2 - u_y^2) \\
m_{19}^{eq} &= \rho u_x u_y u_z \\
m_{20}^{eq} &= 21\biggl[\rho u_x^4 + 6RT\rho u_x^2 + 3(RT)^2\rho + \rho u_y^4 + 6RT\rho u_y^2 + 3(RT)^2\rho \\
&\qquad + \rho u_z^4 + 6RT\rho u_z^2 + 3(RT)^2\rho \\
&\quad + 2(\rho u_x^2 u_y^2 + \rho RT(u_x^2 + u_y^2) + (RT)^2\rho + \rho u_x^2 u_z^2 + \rho RT(u_x^2 + u_z^2) + (RT)^2\rho \\
&\qquad + \rho u_y^2 u_z^2 + \rho RT(u_y^2 + u_z^2) + (RT)^2\rho)\biggr] \\
&\quad - 219(3\rho RT + \rho u_x^2 + \rho u_y^2 + \rho u_z^2) + 440\rho \\
&= 21(\rho u_x^4 + \rho u_y^4 + \rho u_z^4) + 42(\rho u_x^2 u_y^2 + \rho u_x^2 u_z^2 + \rho u_y^2 u_z^2) \\
&\quad + (210RT - 219)(\rho u_x^2 + \rho u_y^2 + \rho u_z^2) \\
&\quad + (315(RT)^2 - 657RT + 440)\rho \\
m_{21}^{eq} &= 9(\rho u_x^2 u_y^2 + \rho u_y^4 + \rho u_z^2 u_y^2 + \rho RT(u_x^2 + u_y^2 + 6u_y^2 + u_z^2 + u_y^2) + 5(RT)^2\rho) \\
&\quad - 3\biggl[\rho u_x^4 + 6RT\rho u_x^2 + 3(RT)^2\rho + \rho u_y^4 + 6RT\rho u_y^2 + 3(RT)^2\rho \\
&\qquad + \rho u_z^4 + 6RT\rho u_z^2 + 3(RT)^2\rho) \\
&\quad + 2(\rho u_x^2 u_y^2 + \rho RT(u_x^2 + u_y^2) + (RT)^2\rho + \rho u_x^2 u_z^2 + \rho RT(u_x^2 + u_z^2) + (RT)^2\rho \\
&\qquad + \rho u_y^2 u_z^2 + \rho RT(u_y^2 + u_z^2) + (RT)^2\rho)\biggr] \\
&\quad - 60(RT\rho + \rho u_y^2) + 20(3\rho RT + \rho u_x^2 + \rho u_y^2 + \rho u_z^2) \\
&= 6\rho u_y^4 - 3(\rho u_x^4 + \rho u_z^4) + 3(\rho u_x^2 u_y^2 + \rho u_z^2 u_y^2) - 6u_x^2 u_z^2 \\
&\quad + (42RT - 40)\rho u_y^2 + (20 - 21RT)(\rho u_x^2 + \rho u_z^2) \\
m_{22}^{eq} &= 3(\rho u_y^2 u_z^2 + RT\rho u_y^2 + RT\rho u_z^2 + (RT)^2\rho - \rho u_x^2 u_y^2 - RT\rho u_x^2 - RT\rho u_y^2 - (RT)^2\rho \\
&\quad + \rho u_z^4 + 6RT\rho u_z^2 + 3(RT)^2\rho - \rho u_x^4 - 6RT\rho u_x^2 - 3(RT)^2\rho) - 20\rho u_z^2 + 20u_x^2 \\
&= 3(\rho u_y^2 u_z^2 - \rho u_x^2 u_y^2 + \rho u_z^4 - \rho u_x^4) + (21RT - 20)(\rho u_z^2 - \rho u_x^2) \\
&= \bigl[3(u_x^2 + u_y^2 + u_z^2) + 21RT - 20\bigr](\rho u_z^2 - \rho u_x^2) \\
m_{23}^{eq} &= 9(\rho u_x^3 u_y + 3RT\rho u_x u_y) + 9(\rho u_y^3 u_x + 3RT\rho u_x u_y)
\end{aligned}
$$

$$
\begin{aligned}
&+ 9(\rho u_z^2 u_x u_y + RT\rho u_x u_y) - 67\rho u_x u_y \\
&= \big[9(u_x^2 + u_y^2 + u_z^2) + 63RT - 67\big]\rho u_x u_y \\
m_{24}^{eq} &= 9(\rho u_x^3 u_z + 3RT\rho u_x u_z) + 9(\rho u_z^3 u_x + 3RT\rho u_x u_z) \\
&+ 9(\rho u_y^2 u_x u_z + RT\rho u_x u_z) - 67\rho u_x u_z \\
&= \big[9(u_x^2 + u_y^2 + u_z^2) + 63RT - 67\big]\rho u_x u_z \\
m_{25}^{eq} &= 9(\rho u_y^3 u_z + 3RT\rho u_y u_z) + 9(\rho u_z^3 u_y + 3RT\rho u_y u_z) \\
&+ 9(\rho u_x^2 u_y u_z + RT\rho u_y u_z) - 67\rho u_y u_z \\
&= \big[9(u_x^2 + u_y^2 + u_z^2) + 63RT - 67\big]\rho u_y u_z \\
m_{26}^{eq} &= 135(\rho u_x^2 u_y^2 + RT\rho u_x^2 + RT\rho u_y^2 + (RT)^2\rho \\
&\quad + \rho u_x^2 u_z^2 + RT\rho u_x^2 + RT\rho u_z^2 + (RT)^2\rho \\
&\quad + \rho u_y^2 u_z^2 + RT\rho u_y^2 + RT\rho u_z^2 + (RT)^2\rho) \\
&- 57\Bigg[\rho u_x^4 + 6RT\rho u_x^2 + 3(RT)^2\rho + \rho u_y^4 + 6RT\rho u_y^2 + 3(RT)^2\rho \\
&\quad + \rho u_z^4 + 6RT\rho u_z^2 + 3(RT)^2\rho \\
&+ 2(\rho u_x^2 u_y^2 + \rho RT(u_x^2 + u_y^2) + (RT)^2\rho + \rho u_x^2 u_z^2 + \rho RT(u_x^2 + u_z^2) + (RT)^2\rho \\
&\quad + \rho u_y^2 u_z^2 + \rho RT(u_y^2 + u_z^2) + (RT)^2\rho)\Bigg] \\
&+ 228(3RT\rho + \rho u_x^2 + \rho u_y^2 + \rho u_z^2) - 320\rho \\
&= 21(\rho u_x^2 u_y^2 + \rho u_x^2 u_z^2 + \rho u_y^2 u_z^2) - 57(\rho u_x^4 + \rho u_y^4 + \rho u_z^4) \\
&(228 - 300RT)(\rho u_x^2 + \rho u_y^2 + \rho u_z^2) + (-450(RT)^2 + 684RT - 320)\rho
\end{aligned}
$$

The orthogonal matrix required 410 add operations and 327 mul operations to calculate, a total of 737 operations. The factorized matrix $N^0 N^1 Q^0 Q^1$ requires $45 + 163 + 12 + 23 = 242$ add operations and $52 + 37 = 89$ mul operations to calculate, a total of 331 operations.

The temperature for this lattice is variable.

Chapter 4

Finite volume schemes

In this chapter we will consider the solution of the advection-diffusion equation and the Navier-Stokes equations using finite volume schemes. We first review temporal schemes before moving to flux-integral methods which are used specifically to solve the advection equation. We use flux integral methods, particularly our NEWTOPIA scheme to simulate the evolution of the buoyancy field using a Boussinesq approximation. A highly accurate scheme for advection of the buoyancy field is particularly important in high Prandtl or Schmidt number buoyancy driven flows which can be encountered in the environmental [226] as well as high Schmidt number passive scalar transport problems [209]. The scheme we present is third order accurate, is truly multidimensional and is total variation diminishing due to the use of a flux limiter, thus the accuracy and stability are superior to lattice Boltzmann schemes for advection diffusion equations [210, 101, 97] which are at best second order accurate and suffer from stability issues. The finite volume scheme we present, due to its explicit and local nature, is also able to be implemented efficiently on highly parallel implementations, especially the graphics processing units (GPU's) that we use in our code. Lattice Boltzmann methods do not offer any performance advantage in this case

We describe incompressible projection methods as well and compressible methods as alternatives to the LB method for the solution of the Navier-Stokes equations. In the final section, we compare our graphics processing unit (GPU) implementation of the compressible Kurganov-Tadmor scheme to our LB implementation. We have chosen lattice Boltzmann methods to solve the Navier-Stokes equations mainly for parallel performance reasons due to their explicit and local nature. As we will explain, incompressible Navier-Stokes solvers require non-local communication to solve the pressure Poisson equation. While it is possible to formulate compressible solvers in an explicit and local manner, the pressure-velocity coupling significantly increases computational effort. We find a large increase in performance comparing LB methods with even the simplest compressible solver we have implemented.

Consider the scalar advection diffusion equation,

$$\partial_t \phi + \partial_\alpha u_\alpha \phi = \alpha \partial_\alpha \partial_\alpha \phi.$$

Integrating over a control volume with volume ΔV,

$$\Delta V \partial_t \bar{\phi} + \iiint_{\Delta V} \partial_\alpha \left(u_\alpha \phi - \alpha \partial_\alpha \phi \right) \mathrm{d}V = 0.$$

Using the divergence theorem, the volume integral becomes a surface integral,

$$\Delta V \partial_t \bar{\phi} + \oiint_S \left(u_\alpha \phi - \alpha \partial_\alpha \phi \right) n_\alpha \, \mathrm{d}S = 0.$$

Assuming a polyhedral control volume with n faces, the integral can be expressed as a sum,

$$\partial_t \bar{\phi} + \frac{1}{\Delta V} \sum_{\boldsymbol{i}=0}^{q} \left[u_\alpha \phi - \alpha \partial_\alpha \phi \right]_{\boldsymbol{i}} n_{\boldsymbol{i}\alpha} S_{\boldsymbol{i}} = 0. \tag{4.1}$$

This is referred to as the semi-discrete form since the temporal derivative has not yet been discretized.

4.1 Time stepping schemes

Considering only the time derivative we seek to approximate the solution to the ordinary differential equation,

$$\frac{d}{dt}u(x,t) = f(u(x,t),x,t).$$

The change over one time step may be treated as an initial value problem and solved by integrating,

$$\begin{aligned} \int_t^{t+\Delta t} \frac{d}{dt}u(x,t)\,\mathrm{d}t &= \int_t^{t+\Delta t} f(u(x,t),x,t)\,\mathrm{d}t, \\ u(x,t+\Delta t) - u(x,t) &= \int_t^{t+\Delta t} f(u(x,t),x,t)\,\mathrm{d}t. \end{aligned} \tag{4.2}$$

The solution amounts to finding the best possible approximation to the integral on the right hand side.

4.1.1 Euler methods

The simplest possible solution is to assume that the value of $f(u(x,t),x,t)$ is a constant, its value can be approximated by the value at the start of the time step to form the forward (or explicit) Euler method,

$$u(x,t+\Delta t) - u(x,t) = \Delta t f(u(x,t),x,t).$$

This is an explicit scheme, meaning that the only unknown is the value of u at the next time step. Approximating the value of f using the new value of u gives the backward (or implicit) Euler method,

$$u(x,t+\Delta t) - u(x,t) = \Delta t f(u(x,t+\Delta t),x,t).$$

Like all implicit methods, the value of u at the next time step must be found by solving an equation or a system of equations. When applied to partial differential equations, the function f will include spatial derivatives of u which are approximated in terms of surrounding points resulting in a large system of coupled equations at each grid point. This can be treated as a matrix inversion problem if the terms are linear.

4.1.2 Higher order polynomial methods – Linear multi step

In order to improve accuracy, the function may be represented by a higher order polynomial. For example, using two points in time, f may be approximated by a straight line, using three points allows a parabola. Integrating the resulting polynomial over the time interval gives a scheme with the same order of accuracy as the polynomial. Using values from previous time steps only gives an explicit scheme, these are referred to as Adams-Bashforth schemes. First, second and third order versions are shown below,

$$\begin{aligned}
u^{n+1} - u^n &= \Delta t f(u^n, x, t^n), \\
u^{n+1} - u^n &= \Delta t \left(\frac{3}{2} f(u^n, x, t^n) - \frac{1}{2} f(u^{n-1}, x, t^{n-1}) \right), \\
u^{n+1} - u^n &= \Delta t \left(\frac{32}{12} f(u^n, x, t^n) - \frac{4}{3} f(u^{n-1}, x, t^{n-1}) \right. \\
&\qquad \left. + \frac{5}{12} f(u^{n-2}, x, t^{n-2}) \right).
\end{aligned}$$

Here we have introduced the notation,

$$u^{n+1} = u(x, t + \Delta t).$$

Including the new time step gives an implicit scheme referred to as Adams-Moulton. The first, second and third order Adams-Moulton schemes are,

$$
\begin{aligned}
u^{n+1} - u^n &= \Delta t f(u^{n+1}, x, t^{n+1}), \\
u^{n+1} - u^n &= \Delta t \frac{1}{2}\Big(f(u^{n+1}, x, t^{n+1}) + f(u^n, x, t^n) \Big), \\
u^{n+1} - u^n &= \Delta t \Big(\frac{5}{12} f(u^{n+1}, x, t^{n+1}) + \frac{2}{3} f(u^n, x, t^n) \\
&\quad - \frac{1}{12} f(u^{n-1}, x, t^{n-1}) \Big).
\end{aligned}
$$

Another common name for the second order Adams-Moulton scheme is the Crank-Nicolson scheme or the trapezoid rule. The Euler schemes are equivalent to the first order versions of these schemes.

4.1.3 Runge-Kutta methods

The Runge-Kutta methods are another family of time stepping algorithm that can provide higher order accuracy. Rather than using values of u at previous time steps, a series of partial time steps are performed and combined to form a more accurate estimate. For example, the midpoint rule is a second order Runge-Kutta scheme. First the Euler method is used to estimate u mid time step, then that estimate is used to calculate f and hence the total change,

$$
\begin{gathered}
k_1 = \Delta t f(u^n, x, t^n), \\
k_2 = \Delta t f\left(u^n + \frac{1}{2} k_1, x, t^n + \frac{1}{2}\Delta t \right), \\
u^{n+1} - u^n = k_2.
\end{gathered}
$$

The intermediate calculations require additional calculations compared to the multi-step methods already mentioned that use information from previous time steps. Another second order Runge-Kutta method is also referred to as the modifier Euler method or Heun's method,

$$\begin{aligned}
k_1 &= \Delta t f(u^n, x, t^n), \\
k_2 &= \Delta t f(u^n + k_1, x, t^n + \Delta t), \\
u^{n+1} - u^n &= \frac{1}{2}(k_1 + k_2).
\end{aligned}$$

The Runge-Kutta methods with s steps can be generalized by the following formula,

$$u^{n+1} - u^n = \sum_{i=1}^{s} b_i k_i.$$

where each of the k_i are an approximation for the time integral of f.

$$k_i = \Delta t f\left(u^n + \sum_{j=1}^{i-1} a_{ij} k_j, x, t^n + c_i \Delta t\right).$$

Each coefficient k_i only depends on those that came before as seen by the upper limit of $i - 1$ in the summation above. The coefficients may be presented neatly in a Butcher tableau [25, 23, 24].

$$\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & \vdots & \ddots & & \\
c_s & a_{s1} & a_{s2} & \dots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \dots & b_{s-1} & b_s
\end{array}$$

In general, an explicit Runge-Kutta method with s steps can be made accurate up to order s by considering a Taylor expansion. The second order methods may be expressed using the tableau below.

$$\begin{array}{c|cc} 0 & & \\ \alpha & \alpha & \\ \hline & \left(1-\frac{1}{2\alpha}\right) & \frac{1}{2\alpha} \end{array}$$

Three notable values are $\alpha = \frac{1}{2}$ giving the already mentioned midpoint method.

$$\begin{array}{c|cc} 0 & & \\ \frac{1}{2} & \frac{1}{2} & \\ \hline & 0 & 1 \end{array}$$

$\alpha = 1$ gives the modified Euler method, also referred to as Heun's method.

$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

$\alpha = \frac{2}{3}$ gives Ralston's method [194] which minimises error.

$$\begin{array}{c|cc} 0 & & \\ \frac{2}{3} & \frac{2}{3} & \\ \hline & \frac{1}{4} & \frac{3}{4} \end{array}$$

The classic fourth order method, sometimes referred to as the RK4 method is given by the tableau.

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

Like the second order methods, there are also multiple fourth order methods.

The Runge-Kutta methods can also be extended to implicit schemes. The difference is that now each k_i can depend on each other one. These implicit relationships must then be solved by some means,

$$k_i = \Delta t f\left(u^n + \sum_{j=1}^{s} a_{ij} k_j, x, t^n + c_i \Delta t\right).$$

Upper triangular and diagonal entries are now included in the tableau.

$$\begin{array}{c|cccc}
c_1 & a_{11} & a_{12} & \dots & a_{1,s} \\
c_2 & a_{21} & a_{22} & \dots & a_{2,s} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
c_s & a_{s1} & a_{s2} & \dots & a_{s,s} \\
\hline
 & b_1 & b_2 & \dots & b_s
\end{array}$$

The simplest such method is the backwards Euler method with tableau.

$$\begin{array}{c|c}
1 & 1 \\
\hline
 & 1
\end{array}$$

The Crank-Nicolson method or trapezoidal rule, which we have also seen as a second order Adams-Moulton scheme is also a second order Runge-Kutta method with tableau.

$$\begin{array}{c|cc}
0 & 0 & 0 \\
1 & \frac{1}{2} & \frac{1}{2} \\
\hline
 & \frac{1}{2} & \frac{1}{2}
\end{array}$$

The update may be written as,

$$\begin{aligned}
k_1 &= \Delta t f(u^n, x, t^n), \\
k_2 &= \Delta t f\left(u^n + \frac{1}{2}(k_1 + k_2), x, t^n + \Delta t\right), \\
u^{n+1} - u^n &= \frac{1}{2}(k_1 + k_2).
\end{aligned}$$

Substituting the overall update into the equation for k_2 gives,

$$k_2 = \Delta t f(u^{n+1}, x, t^n + \Delta t).$$

An important family of RK methods are the Gauss-Legendre methods. These are highly stable and accurate with order $2s$ and are suitable for the solution of stiff equations. The tableau for the second order Gauss-Legendre method is below.

$$\begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array}$$

The tableau for the fourth order Gauss-Legendre method is below.

$$\begin{array}{c|cc} \frac{1}{2} - \frac{1}{6}\sqrt{3} & \frac{1}{4} & \frac{1}{4} - \frac{1}{6}\sqrt{3} \\ \frac{1}{2} + \frac{1}{6}\sqrt{3} & \frac{1}{4} + \frac{1}{6}\sqrt{3} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

In contrast to implicit Adams-Moulton methods, inplicit RK methods have an increased computational cost due to the calculation of intermediate steps however they have advantages in stability and accuracy. Dahlquist [61] introduced the concept of A-stability.

> *"A k-step method is called A-stable, if all solutions of (the scheme) tend to zero, as $n \to \infty$, when the method is applied with fixed positive h to any differential equation of the form,*
>
> $$\frac{dx}{dt} = qx,$$
>
> *where q is a complex constant with negative real part."*

Although not a necessary property, this stability criteria is important to certain classes of differential equations. The highest order A-stable Adams-Moulton methods is of order two while A-stable Gauss-Legendre RK methods can be constructed to any order [61].

4.2 Flux integral methods

The time stepping schemes from the previous section are well suited to solving general differential equations. It is possible to tailor a scheme more specifically for the advection diffusion equation by considering the integral over one time step (Eqn. 4.2) of the semi-discrete finite volume form (Eqn. 4.1).

$$\bar{\phi}\left(x_\alpha, t+\Delta t\right) - \bar{\phi}\left(x_\alpha, t\right) + \frac{1}{\Delta V}\sum_{\boldsymbol{i}=0}^{q}\left(\int_t^{t+\Delta t}\left[u_\alpha\phi - \alpha\partial_\alpha\phi\right]_{\boldsymbol{i}}\,\mathrm{d}t\right)n_{\boldsymbol{i}\alpha}S_{\boldsymbol{i}} = 0.$$

Flux integral methods assume that the velocity may be treated as a constant over the course of a time step and within the vicinity of the face under consideration. In that case, the time integral can be replaced by a time averaged face value implied by the brackets <>,

$$\bar{\phi}\left(x_\alpha, t+\Delta t\right) - \bar{\phi}\left(x_\alpha, t\right) + \sum_{\boldsymbol{i}=0}^{q}\left[c_{n\boldsymbol{i}}\left\langle\phi\right\rangle_{\boldsymbol{i}} - \frac{\Delta t\alpha}{\Delta x_{n\boldsymbol{i}}}\left\langle\partial_\alpha\phi\right\rangle\right]n_{\boldsymbol{i}\alpha} = 0.$$

Here we have used the normal grid size $\Delta x_{n\boldsymbol{i}}$, and the normal Courant number c_n,

$$\begin{aligned}\Delta x_{n\boldsymbol{i}} &= \frac{S_{\boldsymbol{i}}}{\Delta V},\\ c_{n\boldsymbol{i}} &= \frac{u_{n\boldsymbol{i}}\Delta t}{\Delta x_{n\boldsymbol{i}}}.\end{aligned}$$

Due to the constant velocity assumption, the time average can be replaced by an average over the flux integral parallelogram, or parallelepiped in three dimensions (FIP),

$$\left\langle\phi\right\rangle_{\boldsymbol{i}} = \frac{1}{\Delta V_{\text{FIP}}}\iiint_{\text{FIP}}\phi\,\mathrm{d}V_{\text{FIP}}.$$

The FIP is shown schematically in Figure 4.2, c_t is the Courant number in the transverse direction. The variables ξ and η are scaled so that the cell is uniform and of unit size, thus the volume of the FIP is equal to c_n.

$$\langle\phi\rangle_{\boldsymbol{i}} = \frac{1}{c_n} \iiint_{\text{FIP}} \phi \, \mathrm{d}V_{\text{FIP}}. \tag{4.3}$$

In one and two dimensions the volume integral is a replaced by a line or surface integral as appropriate. Some one dimensional schemes that can be derived using this approach include the first order upwind scheme, the second order Lax-Wendroff scheme and the third order QUICKEST scheme. Results obtained in two and three dimensions by applying one dimensional schemes to each dimension are generally not satisfactory, showing reduced stability and accuracy. A common solution, operator splitting, involves performing partial updates in each direction using the latest update in the calculation of the flux terms at each stage. Various techniques have been devised to reduce error introduced by this approach [142]. Flux integral methods avoid the need for operator splitting or any other additional effort by taking into account spatial variations in all directions simultaneously. Semi-Lagrangian schemes [190, 250] are another technique that takes into account motion in all directions, these schemes trace back the path of a Lagrangian particle at the previous time step, typically using some method more accurate than the constant velocity assumption of flux integral methods, then the value at the Lagrangian point is determined using interpolation and used to update the new cell value. A drawback of semi-Lagrangian methods is that they are not conservative in contrast to flux integral methods which are conservative.

4.2.1 Outflow implementation

The flux integral formulation as we present it invites a new computer implementation with the goal of reducing redundant calculations. While the calculation of face fluxes is nothing new, the formulas presented in this section for the NEWTOPIA scheme are novel. In conventional update schemes the interpolation within a cell is performed on average three times in two dimensions and four times in three dimensions during each time step in order to calculate face values for different cell updates. We have found performance increases up to a factor of two from our outflow based implementation which iterates over control volumes in two passes. During the first pass the interpolation (Eqn. 4.4) is calculated in each cell and any outflow face has its flux integral calculated using the direct integrals presented below. Since each face is an outflow to precisely one cell depending on flow direction, each face flux is calculate precisely once. During the second pass, the face fluxes are used in order to update the cell value. Pseudocode for this implementation is shown in Figure 4.1. Branching is also restricted to four if statements which determine whether a face is an outflow or not. There is no upwinding required to figure out the stencil as there is using a conventional update implementation.

Each cell needs to store three variables in two dimensions. It's cell average scalar and the north and east fluxes. South and west flux values are read from the neighbours. This scheme stores 3 values per cell as opposed to two values (old and new) for conventional time stepping schemes. In three dimensions four values must be stored per cell. The advantage is that the interpolation and associated memory access is only done once per cell. Caching is expected to mitigate some of the memory access cost however there are also savings in floating point operations. Numerical experiments, which will be further discussed in Section 4.2.6, show an increase in speed up to a factor of two (Table 4.2). At small grid sizes where caching is more effective there is a smaller difference with the speed improvement increasing up to a maximum of twofold for larger grids.

4.2.2 Two dimensional advection schemes

The order of accuracy of flux integral schemes is determined by the order of interpolation used in the FIP. Polynomial interpolation up to the second degree is considered with the value of ϕ within a cell is approximated by the formula,

$$\phi = C_1 + C_2\xi + C_3\eta + C_4\xi^2 + C_5\eta^2 + C_6\xi\eta. \tag{4.4}$$

```
/* Allocation.  */
u        = (double*)malloc(2*nX*nY*sizeof(double));
phi      = (double*)malloc(  nX*nY*sizeof(double));
phi_flux = (double*)malloc(2*nX*nY*sizeof(double));

/* Initialization.  */
/* ... */

/* Time stepping loop.  */
for (iT=0; iT<nT; iT++) {

    /* First pass.  */
    for (iX=1; iX<(nX-1); iX++) for (iY=1; iY<(nY-1); iY++) {
        id = iX * nY*iY;
        C_1 = *(phi + id);
        C_2 = 0.5*( *(phi + id + 1) - *(phi + id - 1) );
        /* ... */
        u_face_rgh = 0.5*(*(u + id) + *(u + id + 1));
        u_face_lft = 0.5*(*(u + id) + *(u + id - 1));
        /* ... */
        if(u_face_rgh >  0.)
          *(phi_flux + id)              = u_face_rgh*( C1 + ... );
        if(u_face_lft <= 0.)
          *(phi_flux + id - 1)          = u_face_lft*( C1 + ... );
        if(u_face_top >  0.)
          *(phi_flux + id + nX*nY)      = u_face_top*( C1 + ... );
        if(u_face_bot <= 0.)
          *(phi_flux + id + nX*nY - nX) = u_face_bot*( C1 + ... );
    }

    /* Second pass.  */
    for (iX=1; iX<(nX-1); iX++) for (iY=1; iY<(nY-1); iY++) {
        id = iX * nY*iY;
        *(phi + id) -= (*(phi_flux + id)
                      - *(phi_flux + id - 1)
                      + *(phi_flux + id + nX*nY)
                      - *(phi_flux + id + nX*nY - nX));
    }

    /* Boundary conditions.  */
    /* ... */
}
```

Figure 4.1: Pseudocode for outflow scheme.

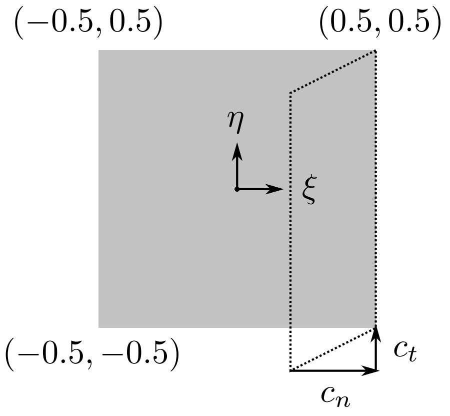


Figure 4.2: Flux integral parallelogram.

The change of co-ordinates is such that ξ and η are centered within the control volume and the grid size is equal to one in each direction. The use of non-dimensional co-ordinates and Courant numbers allows the same formulas to be used for non unit grid sizes. The integral generally covers space spanning multiple control volumes. Leonard's original UTOPIA scheme [144] treats each control volume separately. We use a more simple formulation where the approximate functional form within the cell is assumed to extend outside of the cell for the purpose of calculating the flux integral.

The flux integral is calculated by using the following range of integration within the FIP,

$$\begin{aligned}
&\xi : \xi_0 \to \xi_1, && \eta : \eta_0 (\xi) \to \eta_1 (\xi), \\
&\xi_0 = 0.5 - c_n, && \eta_0 (\xi) = \frac{c_t}{c_n} (\xi - 0.5) - 0.5, \\
&\xi_1 = 0.5, && \eta_1 (\xi) = \frac{c_t}{c_n} (\xi - 0.5) + 0.5.
\end{aligned}$$

The following identities,

$$\begin{aligned}
(a+b)^2 - (a-b)^2 &= 4ab, \\
(a+b)^3 - (a-b)^3 &= (a+b)\left(a^2+2ab+b^2\right) - (a-b)\left(a^2-2ab+b^2\right), \\
&= 4a^2b + 2b\left(a^2+b^2\right) = 6a^2b + 2b^3, \\
a^2 - (a-b)^2 &= 2ab - b^2, \\
a^3 - (a-b)^3 &= a^2 - (a-b)\left(a^2-2ab+b^2\right), \\
&= 3a^2b - 3ab^2 + b^3,
\end{aligned}$$

are used in order to evaluate the definite integrals,

$$\begin{aligned}
[\xi]_{\xi_0}^{\xi_1} &= c_n, & [\eta]_{\eta_0(\xi)}^{\eta_1(\xi)} &= 1, \\
\left[\xi^2\right]_{\xi_0}^{\xi_1} &= c_n - c_n^2, & \left[\eta^2\right]_{\eta_0(\xi)}^{\eta_1(\xi)} &= 2\left(\frac{c_t}{c_n}\right)(\xi - 0.5), \\
\left[\xi^3\right]_{\xi_0}^{\xi_1} &= \frac{3}{4}c_n - \frac{3}{2}c_n^2 + c_n^3, & \left[\eta^3\right]_{\eta_0(\xi)}^{\eta_1(\xi)} &= 3\left(\frac{c_t}{c_n}\right)^2(\xi - 0.5)^2 + \frac{1}{4}.
\end{aligned}$$

We will consider the integration of each term separately,

$$\begin{aligned}
I_1 &= \int_{\xi_0}^{\xi_1}\int_{\eta_0(\xi)}^{\eta_1(\xi)} C_1 \,\mathrm{d}\eta\,\mathrm{d}\xi, \\
&= \int_{\xi_0}^{\xi_1} C_1 \,\mathrm{d}\xi, \\
&= C_1 c_n.
\end{aligned}$$

$$\begin{aligned}
I_2 &= \int_{\xi_0}^{\xi_1}\int_{\eta_0(\xi)}^{\eta_1(\xi)} C_2\xi \,\mathrm{d}\eta\,\mathrm{d}\xi, \\
&= \int_{\xi_0}^{\xi_1} C_2\xi \,\mathrm{d}\xi, \\
&= C_2\frac{1}{2}\left[\xi^2\right]_{\xi_0}^{\xi_1}, \\
&= C_2\frac{1}{2}\left(c_n - c_n^2\right).
\end{aligned}$$

$$\begin{aligned}
I_3 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} C_3 \eta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= \frac{1}{2} C_3 \int_{\xi_0}^{\xi_1} \left[\eta^2\right]_{\eta_0(\xi)}^{\eta_1(\xi)} \mathrm{d}\xi, \\
&= C_3 \int_{\xi_0}^{\xi_1} \frac{c_t}{c_n} \left(\xi - 0.5\right) \mathrm{d}\xi, \\
&= C_3 \frac{c_t}{c_n} \frac{1}{2} \left[\xi^2 - \xi\right]_{\xi_0}^{\xi_1} = C_3 \frac{c_t}{c_n} \frac{1}{2} \left(c_n - c_n^2 - c_n\right), \\
&= -C_3 \frac{1}{2} c_t c_n.
\end{aligned}$$

$$\begin{aligned}
I_4 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} C_4 \xi^2 \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= \int_{\xi_0}^{\xi_1} C_4 \xi^2 \, \mathrm{d}\xi, \\
&= C_4 \frac{1}{3} \left[\xi^3\right]_{\xi_0}^{\xi_1}, \\
&= C_4 \frac{1}{4} c_n - C_4 \frac{1}{2} c_n^2 + C_4 \frac{1}{3} c_n^3.
\end{aligned}$$

$$\begin{aligned}
I_5 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} C_5 \eta^2 \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= C_5 \frac{1}{3} \int_{\xi_0}^{\xi_1} \left[\eta^3\right]_{\eta_0(\xi)}^{\eta_1(\xi)} \mathrm{d}\xi, \\
&= C_5 \int_{\xi_0}^{\xi_1} \left(\left(\frac{c_t}{c_n}\right)^2 \left(\xi - 0.5\right)^2 + \frac{1}{12} \right) \mathrm{d}\xi, \\
&= C_5 \left\{ \frac{1}{3} \left(\frac{c_t}{c_n}\right)^2 \left[\xi^3\right]_{\xi_0}^{\xi_1} - \frac{1}{2} \left(\frac{c_t}{c_n}\right)^2 \left[\xi^2\right]_{\xi_0}^{\xi_1} + \left(\frac{1}{4} \left(\frac{c_t}{c_n}\right)^2 + \frac{1}{12} \right) \left[\xi\right]_{\xi_0}^{\xi_1} \right\}, \\
&= C_5 \Bigg\{ \left(\frac{c_t}{c_n}\right)^2 \left(\frac{1}{4} c_n - \frac{1}{2} c_n^2 + \frac{1}{3} c_n^3 \right) - \frac{1}{2} \left(\frac{c_t}{c_n}\right)^2 \left(c_n - c_n^2\right) \\
&\qquad + \left(\frac{1}{4} \left(\frac{c_t}{c_n}\right)^2 + \frac{1}{12} \right) c_n \Bigg\}, \\
&= C_5 \frac{1}{3} c_n c_t^2 + C_5 \frac{1}{12} c_n.
\end{aligned}$$

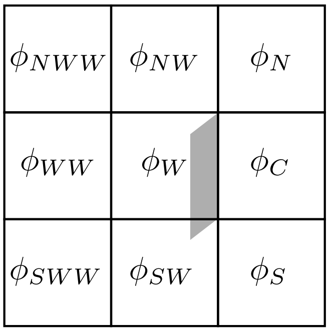


Figure 4.3: Flux integral neighbours.

$$
\begin{aligned}
I_6 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} C_6 \xi \eta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= C_6 \frac{1}{2} \int_{\xi_0}^{\xi_1} \xi \left[\eta^2\right]_{\eta_0(\xi)}^{\eta_1(\xi)} \mathrm{d}\xi, \\
&= C_6 \int_{\xi_0}^{\xi_1} \xi \left(\frac{c_t}{c_n}\right) (\xi - 0.5) \, \mathrm{d}\xi, \\
&= C_6 \left(\frac{c_t}{c_n}\right) \left(\frac{1}{3}\left[\xi^3\right]_{\xi_0}^{\xi_1} - \frac{1}{4}\left[\xi^2\right]_{\xi_0}^{\xi_1}\right), \\
&= C_6 \frac{1}{3} c_n^2 c_t - C_6 \frac{1}{4} c_n c_t.
\end{aligned}
$$

Depending on which constants are included in the interpolation and on which way the constants are defined, various schemes can be obtained. The neighbours are referred to using conventional east, west, north and south notation as shown in Figure 4.3. The face values are determined by summing all of the applicable integrals (I_1, I_2, etc.) then dividing by c_n as per Eqn. 4.3.

The first order upwind scheme is obtained by setting,

$$
\begin{aligned}
&C_1 = \phi_W, \\
&C_2 = C_3 = C_4 = C_5 = C_6 = 0. \\
&\langle \phi \rangle = \phi_W.
\end{aligned}
$$

The Lax-Wendroff scheme [137, 138] is obtained by using a downwind bias.

$$\begin{aligned}
C_1 &= \phi_W, \\
C_2 &= (\phi_C - \phi_W), \\
C_3 &= C_4 = C_5 = C_6 = 0. \\
\langle\phi\rangle &= \phi_W + \frac{1}{2}(1 - c_n)(\phi_C - \phi_W), \\
&= \frac{1}{2}(\phi_C + \phi_W) - c_n\frac{1}{2}(\phi_C - \phi_W).
\end{aligned}$$

A central approximation of the gradient results in Fromm's method [83], a second order upwind scheme.

$$\begin{aligned}
C_1 &= \phi_W, \\
C_2 &= \frac{1}{2}(\phi_C - \phi_{WW}), \\
C_3 &= C_4 = C_5 = C_6 = 0. \\
\langle\phi\rangle &= \phi_W + \frac{1}{4}(1 - c_n)(\phi_C - \phi_{WW}).
\end{aligned}$$

Adding the curvature component results in Leonard's QUICKEST scheme [140, 141].

$$\begin{aligned}
C_1 &= \phi_W - \frac{1}{24}(\phi_C - 2\phi_W + \phi_{WW}), \\
C_2 &= \frac{1}{2}(\phi_C - \phi_{WW}), \\
C_4 &= \frac{1}{2}(\phi_C - 2\phi_W + \phi_{WW}), \\
C_3 &= C_5 = C_6 = 0. \\
\langle\phi\rangle &= \phi_W + \frac{1}{4}(1 - c_n)(\phi_C - \phi_{WW}) \\
&\quad + (\phi_C - 2\phi_W + \phi_{WW}) . \left(\frac{1}{8} - \frac{1}{24} - \frac{1}{4}c_n + \frac{1}{6}c_n^2\right), \\
&= \frac{1}{2}(\phi_C + \phi_W) - c_n\frac{1}{2}(\phi_C - \phi_W) \\
&\quad - \frac{1}{6}\left(1 - c_n^2\right)(\phi_C - 2\phi_W + \phi_{WW}).
\end{aligned}$$

The following three schemes are new multidimensional schemes that we have devised. Taking the Lax-Wendroff scheme and adding an upwind biased transverse gradient term results in the following two-dimensional second order scheme.

$$\begin{aligned}
C_1 &= \phi_W, \\
C_2 &= (\phi_C - \phi_W), \\
C_3 &= (\phi_W - \phi_{SW}), \\
C_4 &= C_5 = C_6 = 0. \\
\langle\phi\rangle &= \frac{1}{2}(\phi_C + \phi_W) - c_n\frac{1}{2}(\phi_C - \phi_W) - c_t\frac{1}{2}(\phi_W - \phi_{SW}).
\end{aligned}$$

Taking the Fromm scheme and adding a central transverse gradient term results in the following two-dimensional scheme,

$$\begin{aligned}
C_1 &= \phi_W, \\
C_2 &= \frac{1}{2}(\phi_C - \phi_{WW}), \\
C_3 &= \frac{1}{2}(\phi_{NW} - \phi_{SW}), \\
C_4 &= C_5 = C_6 = 0. \\
\langle\phi\rangle &= \phi_W + \frac{1}{4}(1 - c_n)(\phi_C - \phi_{WW}) - c_t\frac{1}{4}(\phi_{NW} - \phi_{SW}).
\end{aligned}$$

Taking the QUICKEST scheme and adding centrally calculated transverse gradient, transverse curvature and twist ($\xi\eta$) terms results in our third order multidimensional scheme NEWTOPIA.

$$
\begin{aligned}
C_1 &= \phi_W - \frac{1}{24}\left(\phi_C - 2\phi_W + \phi_{WW}\right) - \frac{1}{24}\left(\phi_{NW} - 2\phi_W + \phi_{SW}\right), \\
C_2 &= \frac{1}{2}\left(\phi_C - \phi_{WW}\right), \\
C_3 &= \frac{1}{2}\left(\phi_{NW} - \phi_{SW}\right), \\
C_4 &= \frac{1}{2}\left(\phi_C - 2\phi_W + \phi_{WW}\right), \\
C_5 &= \frac{1}{2}\left(\phi_{NW} - 2\phi_W + \phi_{SW}\right), \\
C_6 &= \frac{1}{4}\left(\phi_N - \phi_{NWW} - \phi_S + \phi_{SWW}\right). \\
\langle\phi\rangle &= \frac{1}{2}\left(\phi_C + \phi_W\right) - c_n\frac{1}{2}\left(\phi_C - \phi_W\right) \\
&\quad - \frac{1}{6}\left(1 - c_n^2\right)\left(\phi_C - 2\phi_W + \phi_{WW}\right) - c_t\frac{1}{4}\left(\phi_{NW} - \phi_{SW}\right) \\
&\quad + c_t^2\frac{1}{6}\left(\phi_{NW} - 2\phi_W + \phi_{SW}\right) \\
&\quad + \frac{1}{4}\left(\frac{1}{3}c_nc_t - \frac{1}{4}c_t\right)\left(\phi_N - \phi_{NWW} - \phi_S + \phi_{SWW}\right).
\end{aligned}
$$

Leonard's UTOPIA scheme [144] is obtained in a slightly different way. The flux integral is calculated in two parts with a different interpolation function used in each cell that the FIP intersects. Taking the difference between the triangular segments in ϕ_W and ϕ_{SW} introduces higher order derivative terms. Despite the cross term C_6 not being included in the interpolation, a cross difference term still appears in the scheme due to the difference in normal gradient between the two triangular segments. A transverse third derivative (second last) term and third mixed derivative (last) term are likewise introduced. The latter two are fourth order terms which increase the maximum allowable Courant number of the scheme.

$$\begin{aligned}
C_1 &= \phi_W - \frac{1}{24}\left(\phi_C - 2\phi_W + \phi_{WW}\right) - \frac{1}{24}\left(\phi_{NW} - 2\phi_W + \phi_{SW}\right), \\
C_2 &= \frac{1}{2}\left(\phi_C - \phi_{WW}\right), \\
C_3 &= \frac{1}{2}\left(\phi_{NW} - \phi_{SW}\right), \\
C_4 &= \frac{1}{2}\left(\phi_C - 2\phi_W + \phi_{WW}\right), \\
C_5 &= \frac{1}{2}\left(\phi_{NW} - 2\phi_W + \phi_{SW}\right), \\
C_6 &= 0. \\
\langle\phi\rangle &= \frac{1}{2}\left(\phi_C + \phi_W\right) - c_n\frac{1}{2}\left(\phi_C - \phi_W\right) \\
&\quad - \frac{1}{6}\left(1 - c_n^2\right)\left(\phi_C - 2\phi_W + \phi_{WW}\right) - c_t\frac{1}{2}\left(\phi_W - \phi_{SW}\right) \\
&\quad - c_t\left(\frac{1}{4} - c_t\frac{1}{6}\right)\left(\phi_{NW} - 2\phi_W + \phi_{SW}\right) \\
&\quad - c_t\left(\frac{1}{4} - c_n\frac{1}{3}\right)\left(\phi_C - \phi_W - \phi_S + \phi_{SW}\right) \\
&\quad + c_t\left(\frac{1}{12} - c_n^2\frac{1}{8}\right)\left(\left(\phi_C - 2\phi_W + \phi_{WW}\right) - \left(\phi_S - 2\phi_{SW} + \phi_{SWW}\right)\right) \\
&\quad + c_t\left(\frac{1}{12} - c_t^2\frac{1}{24}\right)\left(\phi_{NW} - 3\phi_W + 3\phi_{SW} - \phi_{SSW}\right).
\end{aligned}$$

An earlier version of the UTOPIA scheme which was introduced by Leonard, MacVean and Lock [143] uses different coefficients and drops the fourth order terms.

$$\begin{aligned}
\langle\phi\rangle &= \frac{1}{2}\left(\phi_C + \phi_W\right) - c_n\frac{1}{2}\left(\phi_C - \phi_W\right) \\
&\quad - \frac{1}{6}\left(1 - c_n^2\right)\left(\phi_C - 2\phi_W + \phi_{WW}\right) - c_t\frac{1}{2}\left(\phi_W - \phi_{SW}\right) \\
&\quad - c_t\frac{1}{4}\left(1 - c_t\right)\left(\phi_{NW} - 2\phi_W + \phi_{SW}\right) \\
&\quad - c_t\frac{1}{4}\left(1 - c_n\right)\left(\phi_C - \phi_W - \phi_S + \phi_{SW}\right).
\end{aligned}$$

This form has been encountered in other work such as Thuburn's multidimensional limiter [242].

4.2.3 Three dimensional advection schemes

In three dimensions, the interpolation takes the form:

$$\phi = C_1 + C_2\xi + C_3\eta + C_7\zeta + C_4\xi^2 + C_5\eta^2 + C_8\zeta^2 + C_6\xi\eta + C_9\xi\zeta + C_{10}\eta\zeta. \quad (4.5)$$

The limits of integration are similar to the two dimensional case although there are two transverse velocities,

$$\begin{aligned}
&\xi : \xi_0 \rightarrow \xi_1, && \eta : \eta_0(\xi) \rightarrow \eta_1(\xi), && \zeta : \zeta_0(\xi) \rightarrow \zeta_1(\xi), \\
&\xi_0 = 0.5 - c_n, && \eta_0(\xi) = \frac{c_{t1}}{c_n}(\xi - 0.5) - 0.5, && \zeta_0(\xi) = \frac{c_{t2}}{c_n}(\xi - 0.5) - 0.5, \\
&\xi_1 = 0.5, && \eta_1(\xi) = \frac{c_{t1}}{c_n}(\xi - 0.5) + 0.5, && \zeta_1(\xi) = \frac{c_{t2}}{c_n}(\xi - 0.5) + 0.5.
\end{aligned}$$

With some calculations being based on previous results, the flux integrals are equal to,

$$\begin{aligned}
I_1 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_1 \, \mathrm{d}\zeta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= \int_{\xi_0}^{\xi_1} C_1 \, \mathrm{d}\xi, \\
&= C_1 c_n.
\end{aligned}$$

$$\begin{aligned}
I_2 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_2\xi \, \mathrm{d}\zeta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= \int_{\xi_0}^{\xi_1} C_2\xi \, \mathrm{d}\xi, \\
&= C_2 \frac{1}{2} \left[\xi^2\right]_{\xi_0}^{\xi_1}, \\
&= C_2 \frac{1}{2} \left(c_n - c_n^2\right).
\end{aligned}$$

$$
\begin{aligned}
I_3 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_3 \eta \, \mathrm{d}\zeta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= \frac{1}{2} C_3 \int_{\xi_0}^{\xi_1} \left[\eta^2\right]_{\eta_0(\xi)}^{\eta_1(\xi)} \mathrm{d}\xi, \\
&= C_3 \int_{\xi_0}^{\xi_1} \frac{c_{t1}}{c_n} (\xi - 0.5) \, \mathrm{d}\xi, \\
&= C_3 \frac{c_{t1}}{c_n} \frac{1}{2} \left[\xi^2 - \xi\right]_{\xi_0}^{\xi_1} = C_3 \frac{c_{t1}}{c_n} \frac{1}{2} \left(c_n - c_n^2 - c_n\right), \\
&= -C_3 \frac{1}{2} c_{t1} c_n.
\end{aligned}
$$

$$
\begin{aligned}
I_4 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_4 \xi^2 \, \mathrm{d}\zeta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= \int_{\xi_0}^{\xi_1} C_4 \xi^2 \, \mathrm{d}\xi, \\
&= C_4 \frac{1}{3} \left[\xi^3\right]_{\xi_0}^{\xi_1}, \\
&= C_4 \frac{1}{4} c_n - C_4 \frac{1}{2} c_n^2 + C_4 \frac{1}{3} c_n^3.
\end{aligned}
$$

$$
\begin{aligned}
I_5 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_5 \eta^2 \, \mathrm{d}\zeta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= C_5 \frac{1}{3} \int_{\xi_0}^{\xi_1} \left[\eta^3\right]_{\eta_0(\xi)}^{\eta_1(\xi)} \mathrm{d}\xi, \\
&= C_5 \int_{\xi_0}^{\xi_1} \left(\left(\frac{c_{t1}}{c_n}\right)^2 (\xi - 0.5)^2 + \frac{1}{12} \right) \mathrm{d}\xi, \\
&= C_5 \left\{ \frac{1}{3} \left(\frac{c_{t1}}{c_n}\right)^2 \left[\xi^3\right]_{\xi_0}^{\xi_1} - \frac{1}{2} \left(\frac{c_{t1}}{c_n}\right)^2 \left[\xi^2\right]_{\xi_0}^{\xi_1} + \left(\frac{1}{4} \left(\frac{c_{t1}}{c_n}\right)^2 + \frac{1}{12} \right) \left[\xi\right]_{\xi_0}^{\xi_1} \right\}, \\
&= C_5 \left\{ \left(\frac{c_{t1}}{c_n}\right)^2 \left(\frac{1}{4} c_n - \frac{1}{2} c_n^2 + \frac{1}{3} c_n^3 \right) - \frac{1}{2} \left(\frac{c_{t1}}{c_n}\right)^2 \left(c_n - c_n^2\right) \right. \\
&\qquad \left. + \left(\frac{1}{4} \left(\frac{c_{t1}}{c_n}\right)^2 + \frac{1}{12} \right) c_n \right\}, \\
&= C_5 \frac{1}{3} c_n c_{t1}^2 + C_5 \frac{1}{12} c_n.
\end{aligned}
$$

$$
\begin{aligned}
I_6 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_6 \xi \eta \, \mathrm{d}\zeta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= C_6 \frac{1}{2} \int_{\xi_0}^{\xi_1} \xi \left[\eta^2\right]_{\eta_0(\xi)}^{\eta_1(\xi)} \, \mathrm{d}\xi, \\
&= C_6 \int_{\xi_0}^{\xi_1} \xi \left(\frac{c_{t1}}{c_n}\right) (\xi - 0.5) \, \mathrm{d}\xi, \\
&= C_6 \left(\frac{c_{t1}}{c_n}\right) \left(\frac{1}{3} \left[\xi^3\right]_{\xi_0}^{\xi_1} - \frac{1}{4} \left[\xi^2\right]_{\xi_0}^{\xi_1}\right), \\
&= C_6 \frac{1}{3} c_n^2 c_{t1} - C_6 \frac{1}{4} c_n c_{t1}.
\end{aligned}
$$

$$
\begin{aligned}
I_7 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_7 \zeta \, \mathrm{d}\zeta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= \frac{1}{2} C_7 \int_{\xi_0}^{\xi_1} \left[\zeta^2\right]_{\zeta_0(\xi)}^{\zeta_1(\xi)} \, \mathrm{d}\xi, \\
&= C_7 \int_{\xi_0}^{\xi_1} \frac{c_{t2}}{c_n} (\xi - 0.5) \, \mathrm{d}\xi, \\
&= C_7 \frac{c_{t2}}{c_n} \frac{1}{2} \left[\xi^2 - \xi\right]_{\xi_0}^{\xi_1} = C_7 \frac{c_{t2}}{c_n} \frac{1}{2} \left(c_n - c_n^2 - c_n\right), \\
&= -C_7 \frac{1}{2} c_{t2} c_n.
\end{aligned}
$$

$$
\begin{aligned}
I_8 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_8 \zeta^2 \,\mathrm{d}\zeta \,\mathrm{d}\eta \,\mathrm{d}\xi, \\
&= C_8 \frac{1}{3} \int_{\xi_0}^{\xi_1} \left[\zeta^3\right]_{\zeta_0(\xi)}^{\zeta_1(\xi)} \,\mathrm{d}\xi, \\
&= C_8 \int_{\xi_0}^{\xi_1} \left(\left(\frac{c_{t2}}{c_n}\right)^2 (\xi - 0.5)^2 + \frac{1}{12} \right) \mathrm{d}\xi, \\
&= C_8 \left\{ \frac{1}{3} \left(\frac{c_{t2}}{c_n}\right)^2 \left[\xi^3\right]_{\xi_0}^{\xi_1} - \frac{1}{2} \left(\frac{c_{t2}}{c_n}\right)^2 \left[\xi^2\right]_{\xi_0}^{\xi_1} + \left(\frac{1}{4} \left(\frac{c_{t2}}{c_n}\right)^2 + \frac{1}{12} \right) \left[\xi\right]_{\xi_0}^{\xi_1} \right\}, \\
&= C_8 \left\{ \left(\frac{c_{t2}}{c_n}\right)^2 \left(\frac{1}{4} c_n - \frac{1}{2} c_n^2 + \frac{1}{3} c_n^3 \right) - \frac{1}{2} \left(\frac{c_{t2}}{c_n}\right)^2 \left(c_n - c_n^2\right) \right. \\
&\qquad \left. + \left(\frac{1}{4} \left(\frac{c_{t2}}{c_n}\right)^2 + \frac{1}{12} \right) c_n \right\}, \\
&= C_8 \frac{1}{3} c_n c_{t2}^2 + C_8 \frac{1}{12} c_n.
\end{aligned}
$$

$$
\begin{aligned}
I_9 &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_9 \xi \zeta \,\mathrm{d}\zeta \,\mathrm{d}\eta \,\mathrm{d}\xi, \\
&= C_9 \frac{1}{2} \int_{\xi_0}^{\xi_1} \xi \left[\zeta^2\right]_{\zeta_0(\xi)}^{\zeta_1(\xi)} \,\mathrm{d}\xi, \\
&= C_9 \int_{\xi_0}^{\xi_1} \xi \left(\frac{c_{t2}}{c_n}\right) (\xi - 0.5) \,\mathrm{d}\xi, \\
&= C_9 \left(\frac{c_{t2}}{c_n}\right) \left(\frac{1}{3} \left[\xi^3\right]_{\xi_0}^{\xi_1} - \frac{1}{4} \left[\xi^2\right]_{\xi_0}^{\xi_1} \right), \\
&= C_9 \frac{1}{3} c_n^2 c_{t2} - C_9 \frac{1}{4} c_n c_{t2}.
\end{aligned}
$$

$$
\begin{aligned}
I_{10} &= \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \int_{\zeta_0(\xi)}^{\zeta_1(\xi)} C_{10} \eta \zeta \, \mathrm{d}\zeta \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= C_{10} \frac{1}{2} \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \eta \left[\zeta^2\right]_{\zeta_0(\xi)}^{\zeta_1(\xi)} \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= C_{10} \int_{\xi_0}^{\xi_1} \int_{\eta_0(\xi)}^{\eta_1(\xi)} \eta \left(\frac{c_{t2}}{c_n}\right) (\xi - 0.5) \, \mathrm{d}\eta \, \mathrm{d}\xi, \\
&= C_{10} \frac{1}{2} \int_{\xi_0}^{\xi_1} \left(\frac{c_{t2}}{c_n}\right) (\xi - 0.5) \left[\eta^2\right]_{\eta_0(\xi)}^{\eta_1(\xi)} \, \mathrm{d}\xi, \\
&= C_{10} \int_{\xi_0}^{\xi_1} \left(\frac{c_{t2}}{c_n}\right) \left(\frac{c_{t2}}{c_n}\right) (\xi - 0.5)^2 \, \mathrm{d}\xi, \\
&= C_{10} \left(\frac{c_{t2}}{c_n}\right) \left(\frac{c_{t2}}{c_n}\right) \left(\frac{1}{3} \left[\xi^3\right]_{\xi_0}^{\xi_1} - \frac{1}{2} \left[\xi^2\right]_{\xi_0}^{\xi_1} + \frac{1}{4} \left[\xi\right]_{\xi_0}^{\xi_1}\right), \\
&= C_{10} \left(\frac{c_{t2}}{c_n}\right) \left(\frac{c_{t2}}{c_n}\right) \left(\left(\frac{1}{4} c_n - \frac{1}{2} c_n^2 + \frac{1}{3} c_n^3\right) - \frac{1}{2} \left(c_n - c_n^2\right) + \frac{1}{4} c_n\right), \\
&= C_{10} \frac{1}{3} c_n c_{t1} c_{t2}.
\end{aligned}
$$

The following three schemes are new three dimensional schemes that we have devised. The notation used to specify neighbours in the following section is different to the previous section. Here we use $\phi^{i,j,k}$ to refer to the cell that the interpolation is centered around. The neighbour in the positive x direction is $\phi^{i+,j,k}$ and similarly for other directions. Taking the Lax-Wendroff scheme and adding two upwind biased transverse gradient terms results in the following two-dimensional second order scheme. The direction of the upwind and downwind biases used in calculating the constants must be adjusted depending on the direction of the transverse velocities and depending on which face value is being calculated.

$$
\begin{aligned}
C_1 &= \phi^{i,j,k}, \\
C_2 &= \left(\phi^{i+,j,k} - \phi^{i,j,k}\right), \\
C_3 &= \left(\phi^{i,j,k} - \phi^{i,j-,k}\right), \\
C_7 &= \left(\phi^{i,j,k} - \phi^{i,j,k-}\right), \\
C_4 &= C_5 = C_6 = C_8 = C_9 = C_{10} = 0. \\
\langle\phi\rangle &= C_1 + C_2\frac{1}{2}\left(1 - c_n\right) - C_3\frac{1}{2}c_{t1} - C_7\frac{1}{2}c_{t2}. \\
\langle\phi\rangle &= \frac{1}{2}\left(\phi^{i+,j,k} + \phi^{i,j,k}\right) - c_n\frac{1}{2}\left(\phi^{i+,j,k} - \phi^{i,j,k}\right) \\
&\quad - c_{t1}\frac{1}{2}\left(\phi^{i,j,k} - \phi^{i,j-,k}\right) - c_{t2}\frac{1}{2}\left(\phi^{i,j,k} - \phi^{i,j,k-}\right).
\end{aligned}
$$

Taking the Fromm scheme and adding a central transverse gradient term results in the following three-dimensional scheme. In comparison to the Lax-Wendroff based scheme, this scheme may result in a larger stencil being used although if the outflow based algorithm is used, there is no difference in stencil size.

$$
\begin{aligned}
C_1 &= \phi^{i,j,k}, \\
C_2 &= \frac{1}{2}\left(\phi^{i+,j,k} - \phi^{i-,j,k}\right), \\
C_3 &= \frac{1}{2}\left(\phi^{i,j+,k} - \phi^{i,j-,k}\right), \\
C_7 &= \frac{1}{2}\left(\phi^{i,j,k+} - \phi^{i,j,k-}\right), \\
C_4 &= C_5 = C_6 = C_8 = C_9 = C_{10} = 0. \\
\langle\phi\rangle &= C_1 + C_2\frac{1}{2}\left(1 - c_n\right) - C_3\frac{1}{2}c_{t1} - C_7\frac{1}{2}c_{t2}, \\
\langle\phi\rangle &= \phi^{i,j,k} + \frac{1}{4}\left(1 - c_n\right)\left(\phi^{i+,j,k} - \phi^{i-,j,k}\right) \\
&\quad - c_{t1}\frac{1}{4}\left(\phi^{i,j+,k} - \phi^{i,j-,k}\right) - c_{t2}\frac{1}{4}\left(\phi^{i,j,k+} - \phi^{i,j,k-}\right).
\end{aligned}
$$

Taking the QUICKEST scheme and adding centrally calculated gradient, curvature and twist ($\xi\eta$) terms in both transverse directions results in our third order three dimensional scheme NEWTOPIA.

$$
\begin{aligned}
C_1 &= \phi^{i,j,k} - \frac{1}{12}\left(C_4 + C_5 + C_8\right), \\
C_2 &= \frac{1}{2}\left(\phi^{i+,j,k} - \phi^{i-,j,k}\right), \\
C_3 &= \frac{1}{2}\left(\phi^{i,j+,k} - \phi^{i,j-,k}\right), \\
C_7 &= \frac{1}{2}\left(\phi^{i,j,k+} - \phi^{i,j,k-}\right), \\
C_4 &= \frac{1}{2}\left(\phi^{i+,j,k} - 2\phi^{i,j,k} + \phi^{i-,j,k}\right), \\
C_5 &= \frac{1}{2}\left(\phi^{i,j+,k} - 2\phi^{i,j,k} + \phi^{i,j-,k}\right), \\
C_8 &= \frac{1}{2}\left(\phi^{i,j,k+} - 2\phi^{i,j,k} + \phi^{i,j,k-}\right), \\
C_6 &= \frac{1}{4}\left(\phi^{i+,j+,k} - \phi^{i-,j+,k} - \phi^{i+,j-,k} + \phi^{i-,j-,k}\right), \\
C_9 &= \frac{1}{4}\left(\phi^{i+,j,k-} - \phi^{i-,j,k+} - \phi^{i+,j,k-} + \phi^{i-,j,k-}\right), \\
C_{10} &= \frac{1}{4}\left(\phi^{i,j+,k+} - \phi^{i,j-,k+} - \phi^{i,j+,k-} + \phi^{i,j-,k-}\right). \\
\langle\phi\rangle &= C_1 + C_2\frac{1}{2}\left(1 - c_n\right) - C_3\frac{1}{2}c_{t1} - C_7\frac{1}{2}c_{t2} \\
&\quad + C_4\left(\frac{1}{4} - \frac{1}{2}c_n + \frac{1}{3}c_n^2\right) + C_5\left(\frac{1}{3}c_{t1}^2 + \frac{1}{12}\right) + C_8\left(\frac{1}{3}c_{t2}^2 + \frac{1}{12}\right) \\
&\quad \left(C_6 c_{t1} + C_9 c_{t2}\right)\left(\frac{1}{3}c_n - \frac{1}{4}\right) + C_{10}\frac{1}{3}c_{t1}c_{t2}, \\
\langle\phi\rangle &= \frac{1}{2}\left(\phi^{i+,j,k} + \phi^{i,j,k}\right) - c_n\frac{1}{2}\left(\phi^{i+,j,k} - \phi^{i,j,k}\right) \\
&\quad - \frac{1}{6}\left(1 - c_n^2\right)\left(\phi^{i+,j,k} - 2\phi^{i,j,k} + \phi^{i-,j,k}\right) \\
&\quad - c_{t1}\frac{1}{4}\left(\phi^{i,j+,k} - \phi^{i,j-,k}\right) + c_{t1}^2\frac{1}{6}\left(\phi^{i,j+,k} - 2\phi^{i,j,k} + \phi^{i,j-,k}\right) \\
&\quad - c_{t2}\frac{1}{4}\left(\phi^{i,j,k+} - \phi^{i,j,k-}\right) + c_{t2}^2\frac{1}{6}\left(\phi^{i,j,k+} - 2\phi^{i,j,k} + \phi^{i,j,k-}\right) \\
&\quad + \frac{1}{4}\left(\frac{1}{3}c_n c_{t1} - \frac{1}{4}c_{t1}\right)\left(\phi^{i+,j+,k} - \phi^{i-,j+,k} - \phi^{i+,j-,k} + \phi^{i-,j-,k}\right) \\
&\quad + \frac{1}{4}\left(\frac{1}{3}c_n c_{t2} - \frac{1}{4}c_{t2}\right)\left(\phi^{i+,j,k-} - \phi^{i-,j,k+} - \phi^{i+,j,k-} + \phi^{i-,j,k-}\right) \\
&\quad + \frac{1}{12}c_{t1}c_{t2}\left(\phi^{i,j+,k+} - \phi^{i,j-,k+} - \phi^{i,j+,k-} + \phi^{i,j-,k-}\right).
\end{aligned}
$$

4.2.4 Diffusion

If a zeroth order interpolation is used, the value of ϕ in each cell is assumed to be a constant, hence there is no first order diffusion scheme. A linear interpolation results in the conventional second order central scheme for diffusion, the transverse gradient does not contribute. Using second order interpolation in order to calculate the diffusion term can result in a third order scheme. Calculations for this scheme have not been performed. We use the second order central diffusion scheme in all our simulations.

4.2.5 Flux limiters

Total variation diminishing (TVD) schemes [238] are a second order accurate means of simulating systems of hyperbolic conservation laws which do not introduce any spurious oscillations typical of second order and higher schemes. No new extrema are introduced and the magnitude of existing extrema are not increased. Although TVD schemes had been introduced prior (eg. van Leer [249]), Sweby [238] showed that they could all be expressed as different limiter functions within the same framework. These schemes are presented as a linear combination of the first order upwind (FU) scheme and the central Lax-Wendroff (LW) scheme [137, 138] combined according to a limiter function ψ where $\psi = 0$ implies the upwind scheme and $\psi = 1$ implies the LW scheme.

$$\phi = \psi\phi^{\text{LW}} + (1 - \psi)\,\phi^{\text{FU}}$$

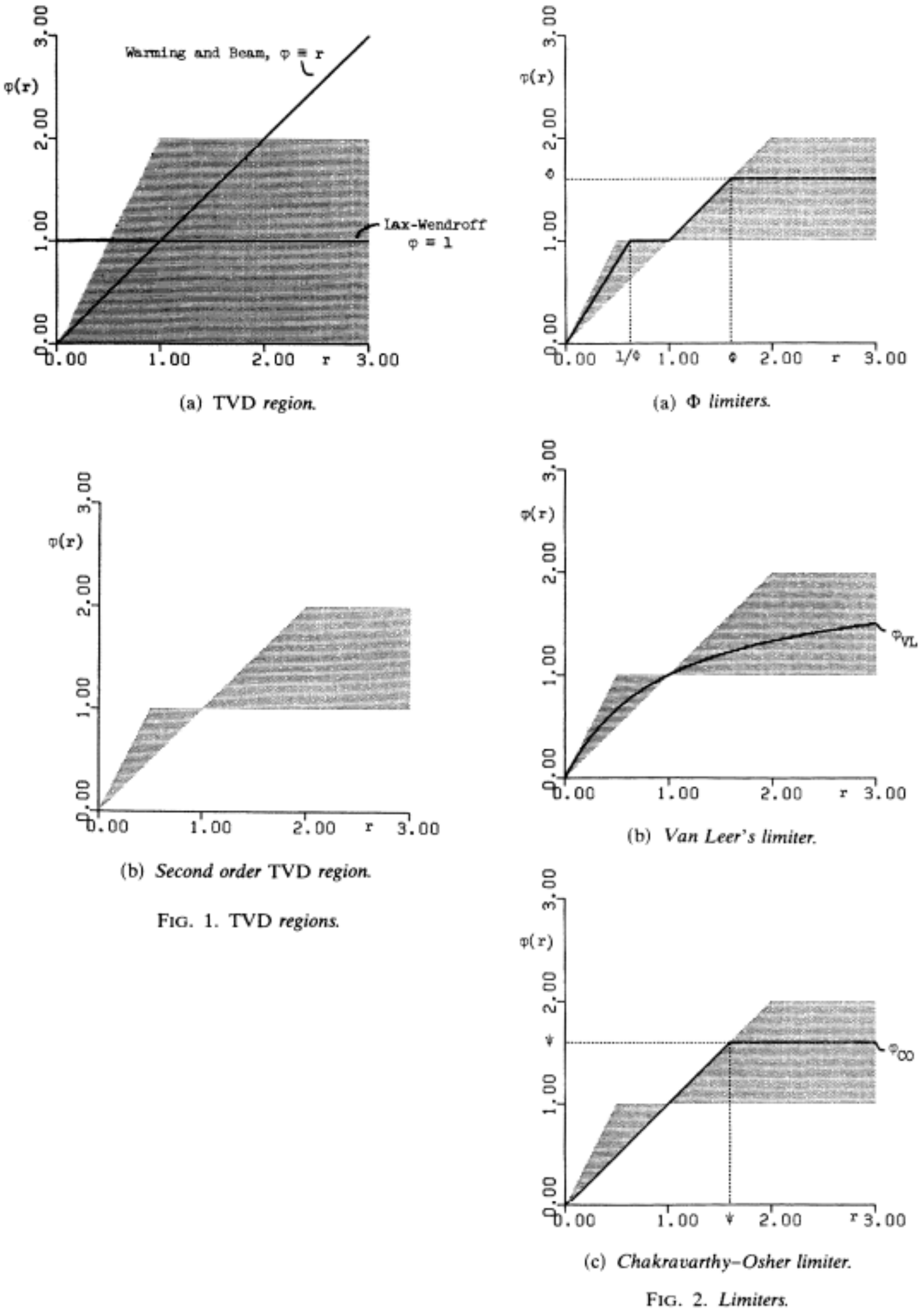
The limiter function varies in response to the successive ratio,

$$r = \frac{\phi^{x+} - \phi^{x}}{\phi^{x} - \phi^{x-}}.$$

The range of values that satisfy the TVD property can be plotted on a $\psi - r$ diagram (also called a Sweby diagram) and using the fact that all second order schemes can be expressed as a linear combination of the LW scheme and the upwind Warming and Beam [252] scheme, the second order accurate region can also be plotted on the Sweby diagram. Most limiters of interest lie in the intersection of the second order and TVD regions. The Sweby diagram and some common limiters are shown in Figure 4.4. Second order TVD limiter functions include Van Leers limiter [249], Roe's minmod and the superbee limiters [198], the Osher limiter [35] and many others. All second order accurate, the different limiter functions exhibit varying characteristics, for example Roe's superbee limiter is compressive, it sharpens interfaces without introducing new extrema. Fringer, Armfield and Street [81] have tested various limiters applied to the advection of a buoyancy field particularly studying the effect of compressive limiters on the conservation of background potential energy. Another related family of schemes are the flux corrected transport schemes [18, 260, 111]. These use a two pass algorithm to blend the first order upwind and some higher order scheme in order to eliminate new extrema.

We will use Thuburn's multi-dimensional limiter [242]. This limiter, like Leonard's one dimensional ULTIMATE limiter [141] is derived under under the assumption of the scalar advection equation. As such these limiters are not applicable to general systems of conservation laws. Leonard notes that the TVD region is too restrictive and finds a more relaxed set of limits that can increase the order of accuracy, essentially by reducing the application of the limiter. This improvement is only applicable to pure advection and is not an oversight of the TVD framework. Leonard presents his limiter using a so called normalized variable diagram which is similar to a Sweby diagram which introduces the possibility of other limiter functions within these revised acceptable bounds.

The Thuburn limiter is derived by applying logic to the finite volume form of the advection equation with the goal of preventing the formation of new local extrema. Unlike the previously mentioned TVD limiters, there are no free parameters in the Turburn limiter, it simply seeks to find the widest possible limits that will prevent the formation of new extrema. It is likely that there exist some parameterization analogous to a multidimensional Sweby diagram that could produce other limiter



(a) TVD *region.*

(b) *Second order* TVD *region.*

FIG. 1. TVD *regions.*

(a) Φ *limiters.*

(b) *Van Leer's limiter.*

(c) *Chakravarthy–Osher limiter.*

FIG. 2. *Limiters.*

Figure 4.4: Sweby diagrams for various limiters (from Sweby [238]).

functions with similar properties.

The first step in Thuburn's limiter is to calculate the upstream maximum $\phi^{(\mathtt{us})}_{\boldsymbol{i},\mathtt{max}}$ and minimum $\phi^{(\mathtt{us})}_{\boldsymbol{i},\mathtt{min}}$ at each face. While Thuburn suggests considering only the transverse neighbour in the upwind direction, we have found better results using both transverse directions. For example, the upstream neighbours we use for the face shown in Figure 4.3 would be ϕ_{NW}, ϕ_W and ϕ_{SW}, The downstream cell from any inflow face is simply the cell under consideration ϕ, it is used in the first approximation to the inflow bounds,

$$\begin{aligned}\phi^{(\mathtt{in})}_{\boldsymbol{i},\mathtt{min}} &= \min\left(\phi^{(\mathtt{us})}_{\boldsymbol{i},\mathtt{min}}, \phi\right),\\ \phi^{(\mathtt{in})}_{\boldsymbol{i},\mathtt{max}} &= \max\left(\phi^{(\mathtt{us})}_{\boldsymbol{i},\mathtt{max}}, \phi\right).\end{aligned}$$

Next the face value $\phi_{\boldsymbol{i}}$ is set to the face value of the base scheme, $\phi^{(\mathtt{base})}_{\boldsymbol{i}}$

$$\phi_{\boldsymbol{i}} = \phi^{(\mathtt{base})}_{\boldsymbol{i}}.$$

The face value is constrained to lie between the inflow bounds,

$$\begin{aligned}\phi_{\boldsymbol{i}} &= \min\left(\phi_{\boldsymbol{i}}, \phi^{(\mathtt{in})}_{\boldsymbol{i},\mathtt{max}}\right),\\ \phi_{\boldsymbol{i}} &= \max\left(\phi_{\boldsymbol{i}}, \phi^{(\mathtt{in})}_{\boldsymbol{i},\mathtt{min}}\right).\end{aligned}$$

The flux limiter will only shift the face value closer towards the upstream maximum so the inflow bounds may optionally be revised at this stage,

$$\boxed{\begin{aligned}\phi^{(\mathtt{in})}_{\boldsymbol{i},\mathtt{min}} &= \min\left(\phi^{(\mathtt{us})}_{\boldsymbol{i},\mathtt{min}}, \phi_{\boldsymbol{i}}\right),\\ \phi^{(\mathtt{in})}_{\boldsymbol{i},\mathtt{max}} &= \max\left(\phi^{(\mathtt{us})}_{\boldsymbol{i},\mathtt{max}}, \phi_{\boldsymbol{i}}\right).\end{aligned}}$$

This modification significantly relaxes the limits imposed by the limiter improving accuracy however it introduces a new data dependence. Omitting this step allows the outflow bounds to be calculated at the same time as the outflow face values of the basic scheme, hence the flux limited outflows can be calculated in one pass. We will refer to this as the "one pass" limiter. In order to adjust the inflow bounds using inflow face values from the basic scheme, outflow face values from adjacent cells must be known. This requires the limiter to be applied in a second pass, the basic outflow values having been calculated during the first. There is also an additional storage requirement, as the flux limiting pass writes out its limited face values, the values from the basic scheme are overwritten, cells which are updated later will then use flux limited face values in their inflow limits. In our testing, we have not seen any negative effect resulting from allowing this to occur however in order to prevent any unintended consequences from this interdependence we have used additional variables to store the flux limited outflows. Parallel implementations may not allow the programmer to specify the update order of cells. This data access problem bears some similarity to the difference between Gauss-Seidel and Gauss-Jacobi schemes, the order of updates can be disambiguated for the former using a red black checkerboard pattern where red nodes are updated first followed by black. Nodes of the same color have no interdependence. There also exists the possibility of performing additional passes where inflow bounds for each face could be further narrowed to lie between the basic value and the limited value from the previous pass, we have not tested this modification yet. We use separate storage in what we will subsequently refer to as our "two pass" scheme.

Now that the inflow bounds have been specified, the maximum and minimum cell value are calculated. Using the physical intuition that no new extrema should be introduced, these limits are calculated based on the previously calculated maxima and minima in the upstream neighbourhood of the cells faces,

$$\begin{aligned}\phi_{\mathtt{min}} &= \min\left(\phi^{(\mathtt{up})}_{0,\mathtt{min}}, \phi^{(\mathtt{up})}_{1,\mathtt{min}}, \dots, \phi^{(\mathtt{up})}_{n,\mathtt{min}}\right),\\ \phi_{\mathtt{max}} &= \max\left(\phi^{(\mathtt{up})}_{0,\mathtt{max}}, \phi^{(\mathtt{up})}_{1,\mathtt{max}}, \dots, \phi^{(\mathtt{up})}_{n,\mathtt{max}}\right).\end{aligned}$$

By considering the total flux into the control volume during one time step, a set of outflow bounds may be derived,

$$
\begin{aligned}
\phi_{i,\mathtt{min}}^{(\mathtt{out})} &= \frac{1}{\sum_{\mathtt{out}} c_i}\left[\phi + \sum_{\mathtt{in}} c_i \phi_{i,\mathtt{max}}^{(\mathtt{in})} - \phi_{\mathtt{max}}\left(1 + \sum_{\mathtt{in}} c_i - \sum_{\mathtt{out}} c_i\right)\right], \\
\phi_{i,\mathtt{max}}^{(\mathtt{out})} &= \frac{1}{\sum_{\mathtt{out}} c_i}\left[\phi + \sum_{\mathtt{in}} c_i \phi_{i,\mathtt{min}}^{(\mathtt{in})} - \phi_{\mathtt{min}}\left(1 + \sum_{\mathtt{in}} c_i - \sum_{\mathtt{out}} c_i\right)\right].
\end{aligned}
$$

Finally, the face values are constrained to lie within the outflow bounds,

$$
\begin{aligned}
\phi_i &= \min\left(\phi_i, \phi_{i,\mathtt{max}}^{(\mathtt{out})}\right), \\
\phi_i &= \max\left(\phi_i, \phi_{i,\mathtt{min}}^{(\mathtt{out})}\right).
\end{aligned}
$$

4.2.6 Numerical experiments

We set out to test the stability, accuracy and speed of our NEWTOPIA advection scheme and the outflow implementation both with and without Thuburn's limiter. We have also tested the UTOPIA scheme for comparison. We find minimal difference between the NEWTOPIA and UTOPIA schemes with the counter intuitive result that the NEWTOPIA scheme showed slightly improved accuracy, despite Leonard's UTOPIA scheme having some fourth order terms present. The outflow implementation achieves a significant performance improvement, particularly with larger domain sizes. Thuburn's limiter functions as expected, preventing the introduction of any new local extrema and reduces the order of accuracy to second order as expected. We have tested three benchmark cases, a rotating cylinder, a constant velocity Gaussian hill and a Taylor-Green vortex flow field with a step change in scalar.

All experiments were performed on a quad core Intel i5 950 CPU using four processes. The CPU clock speed was 2,667 MHz while the memory speed was 1,333 MHz. The operating system used was Arch Linux with Linux kernel 4.1.6 and `gcc` 5.2.0 was used to compile the code.

The rotating cylinder test aims to verify the ability of the scheme under test to capture a discontinuous change in the scalar. Square domains with $300 - 1200$ cells in each direction are used. The grid size is unity so the domain size l_0 corresponds to the number of cells. The diameter of the cylinder, $d_0 = 0.1 \times l_0$ and the cylinder is offset from the center of rotation by $r_0 = 0.35 \times l_0$. The rotational velocity ω is chosen so that tangential velocity at the center of the cylinder, $u = 0.525$ while the maximum velocity in the domain occurs at the corners and is approximately one, with a time step of half this leads to a maximum Courant number of half at the corners of the domain and approximately a quarter at the center of the cylinder. The gradient of the scalar normal to the boundary is set to zero. The simulation was allowed to progress until the non dimensional time,

$$t^* = t\omega,$$

reached a value of 2.5.

Table 4.1 shows the L_2 norm of the error over the domain compared to the exact solution at the end of the simulation. The third order schemes have a similar error while the flux limited schemes improve accuracy by a few percent. There is no difference between the accuracy of the one pass and two pass flux limiters. The errors are plotted against grid size in Figure 4.5. Although the errors do reduce with grid size, due to the poor resolution of the discontinuity no asymptotic convergence is observed. The beneficial effect of the flux limiter is apparent in the visualization of Figure 4.6 demonstrating that the over and under shoot of the third order schemes is eliminated by the flux limiter.

The performance of various schemes in terms of mega updates per second (MUPS) is shown in Table 4.2 and Figure 4.7. This performance metric is obtained by dividing the total number of cell updates by the time taken. The performance of all schemes is reduced as the number of cells is increased as a result of cache effects becoming less dominant. The outflow implementation is significantly faster at all grid sizes with a performance improvement of roughly 5 MUPS. Using larger

Cells	UTOPIA	NEWTOPIA	NEWTOPIA (1 pass lim)	NEWTOPIA (2 pass lim)
300^2	2.049×10^{-2}	2.032×10^{-2}	1.948×10^{-2}	1.948×10^{-2}
420^2	1.837×10^{-2}	1.822×10^{-2}	1.747×10^{-2}	1.747×10^{-2}
600^2	1.617×10^{-2}	1.604×10^{-2}	1.537×10^{-2}	1.537×10^{-2}
840^2	1.440×10^{-2}	1.428×10^{-2}	1.369×10^{-2}	1.369×10^{-2}
1200^2	1.269×10^{-2}	1.259×10^{-2}	1.208×10^{-2}	1.208×10^{-2}

Table 4.1: $L2$ average error for various schemes and grid sizes – Rotating cylinder

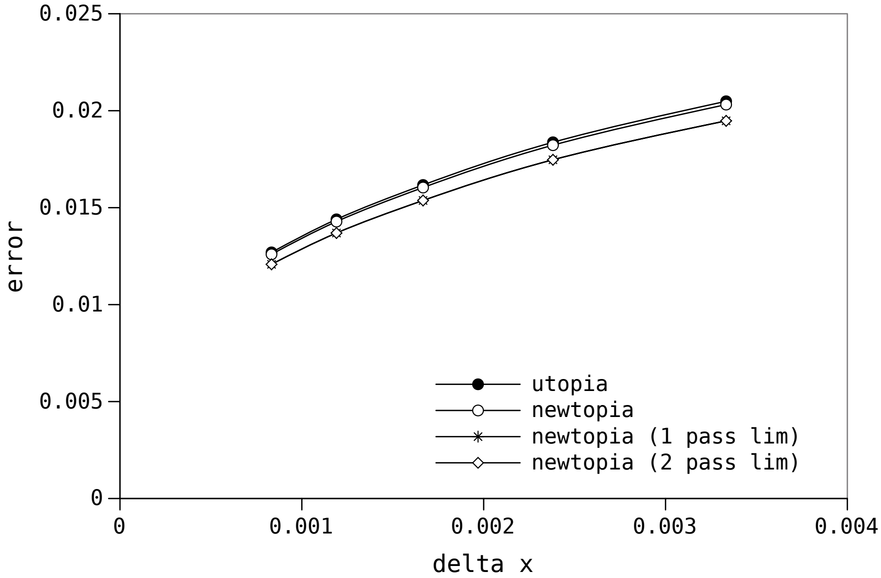


Figure 4.5: $L2$ average error for various schemes and grid sizes – Rotating cylinder

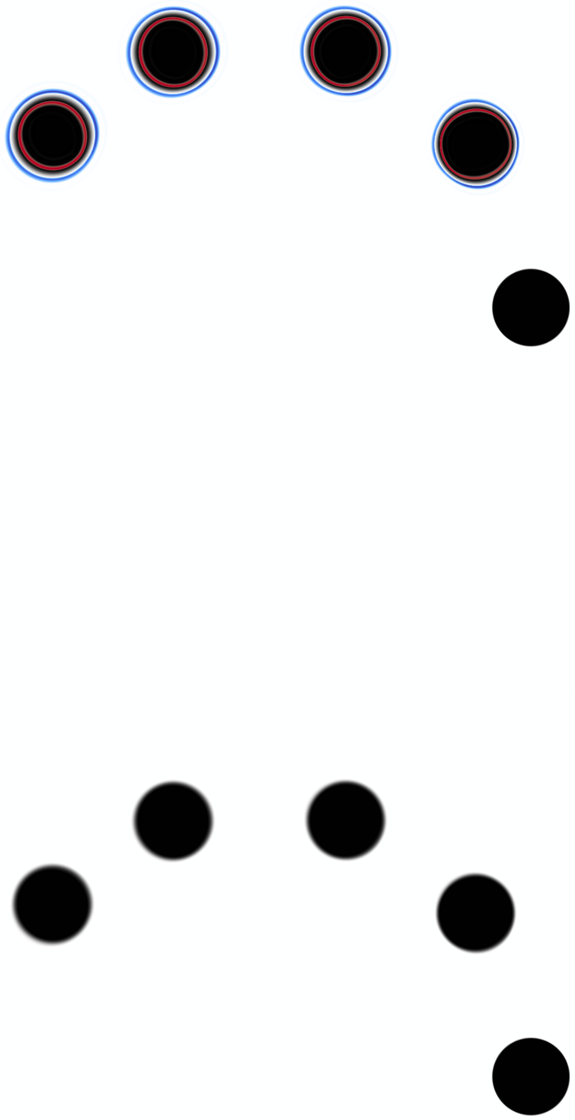


Figure 4.6: Rotating cylinder advection test. Top shows over and under shoot of NEWTOPIA and UTOPIA schemes in blue and red. Flux limited schemes, shown in bottom panel, remain bounded.

Cells	UTOPIA	NEWTOPIA	NEWTOPIA (outflow)	NEWTOPIA (1 pass lim)	NEWTOPIA (2 pass lim)
300^2	13.40	15.04	18.00	14.37	12.17
420^2	10.20	12.13	15.04	11.87	9.379
600^2	7.676	9.718	14.18	7.987	6.693
840^2	4.787	6.087	10.73	7.703	6.644
1200^2	3.165	5.044	9.506	5.923	6.075

Table 4.2: Update speed in mega updates per second (MUPS) – Rotating cylinder

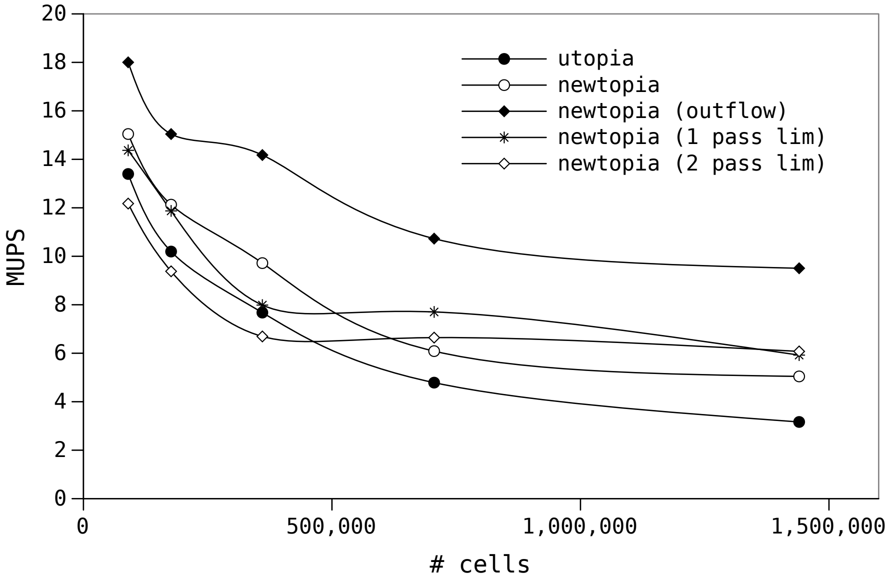


Figure 4.7: Update speed in mega updates per second (MUPS) – Rotating cylinder

domains, as the overall speed of all schemes is reduced, this 5 MUPS improvement becomes more significant representing a doubling of performance. The flux limited implementation is also an outflow based implementation and despite the additional calculations involved in flux limiting, the one pass limiter is nevertheless faster than the standard implementation of the basic schemes. The two pass limiter is slower as expected.

The advection of a Gaussian hill by a constant velocity is designed to test the asymptotic convergence of the schemes under test. The same range of domain sizes from the rotating cylinder test is used and the velocity is set at 45 deg to the grid with the time step such that the Courant number is half. A periodic boundary condition is used and the scalar is initialised to a Gaussian hill according to the formula,

$$\phi = \exp\left(\frac{x_\alpha x_\alpha}{-2B^2}\right).$$

The parameter was set to $B = 0.05 \times l_0$. The gradients are sufficiently well resolved by the chosen grid sizes that there is no over or undershoot using the third order UTOPIA or NEWTOPIA schemes as shown in Figure 4.9 and asymptotic convergence is observed. The simulation was allowed to proceed until the non dimensional time,

$$t^* = t\frac{u_0}{l_0},$$

reached a value of $0.8\dot{3}$.

Figure 4.8 shows the error behaviour of all tested schemes with respect to grid size on logarithmic axes. The dashed lines indicate second and third order asymptotic convergence slopes. Error values are presented in Table 4.3 with the divided differences shown in the last row. The divided difference is equal to four for a second order scheme and eight for a third order scheme. The gradient of both the UTOPIA scheme and our NEWTOPIA scheme indicates third order accuracy, a divided difference of 7.98 confirms this. The flux limiters degrade accuracy somewhat, with the two pass limiter having minimal effect, a divided difference of 7.55 indicates that the order of accuracy remains close to third order. The one pass limiter has a more significant effect. With a divided difference of 6.36 the accuracy is still closer to third order than to second. This may be considered a reasonable trade off considering the reduction in computational effort and memory required.

We have also performed a simulation of a step change being advected by a Taylor-Green vortex. The velocity field used is a constant velocity variation of the Taylor-

Cells	UTOPIA	NEWTOPIA	NEWTOPIA	NEWTOPIA
			(1 pass lim)	(2 pass lim)
300^2	7.286×10^{-5}	6.733×10^{-5}	9.922×10^{-5}	7.303×10^{-5}
420^2	2.672×10^{-5}	2.469×10^{-5}	4.045×10^{-5}	2.746×10^{-5}
600^2	9.131×10^{-6}	8.437×10^{-6}	1.577×10^{-5}	9.741×10^{-6}
840^2	3.343×10^{-6}	3.089×10^{-6}	6.609×10^{-6}	3.731×10^{-6}
1200^2	1.141×10^{-6}	1.055×10^{-6}	2.644×10^{-6}	1.353×10^{-6}
Divided difference	7.98	7.98	6.36	7.55

Table 4.3: $L2$ average error for various schemes and grid sizes – Gaussian hill

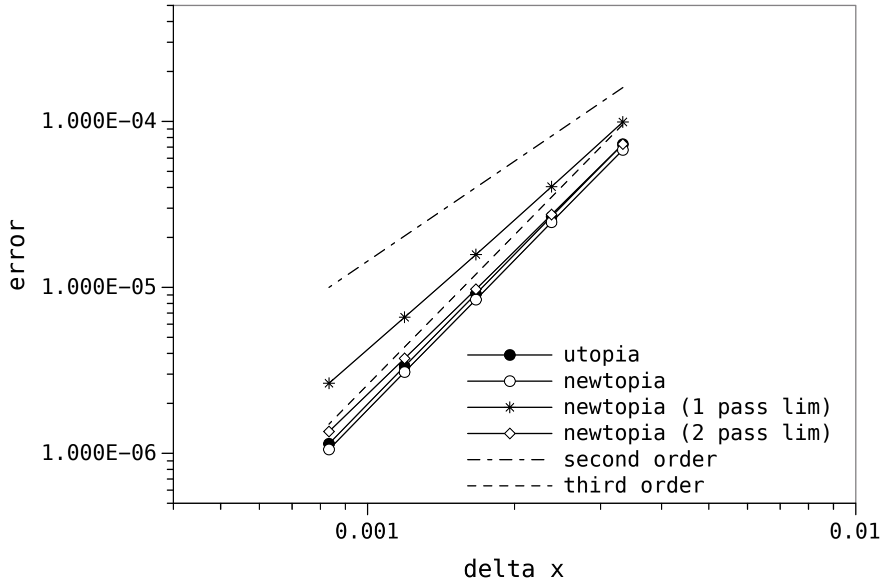


Figure 4.8: $L2$ average error for various schemes and grid sizes – Gaussian hill

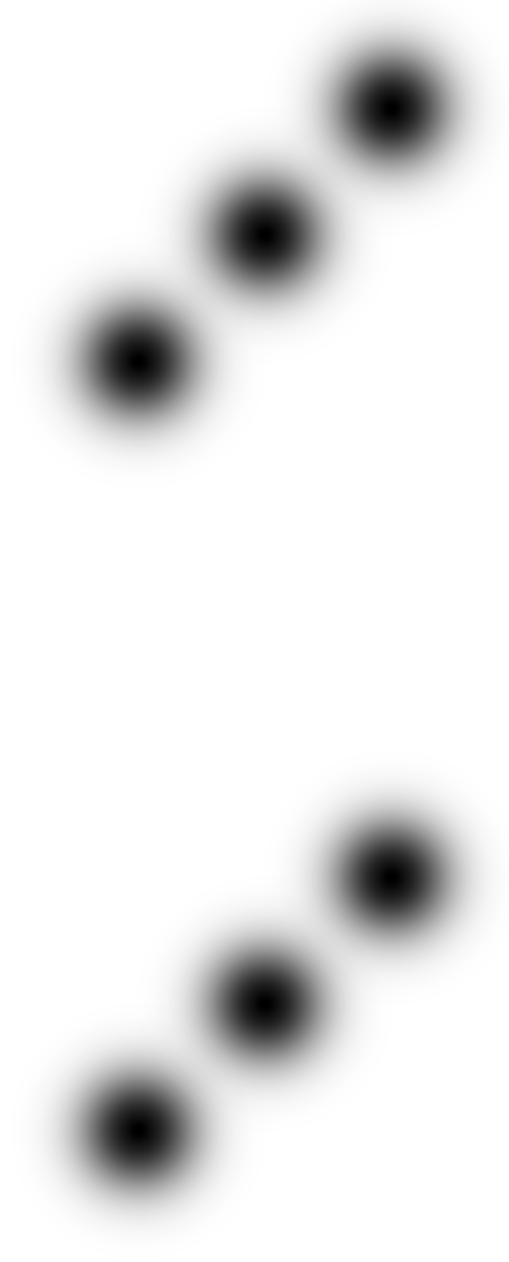


Figure 4.9: Gaussian hill advection test. Top panel shows NEWTOPIA and UTOPIA schemes, no over or under shoot present. Flux limited schemes are shown in bottom panel.

Green vortex velocity, shown in Eqn. 5.6,

$$\begin{aligned} u(x,y,t) &= \quad u_0 \cos(k_0 x)\sin(k_0 y), \\ v(x,y,t) &= -u_0 \sin(k_0 x)\cos(k_0 y), \\ k_0 &= l_0^{-1}. \end{aligned}$$

The characteristic length l_0 is equal to the domain size. Periodic boundary conditions are used and the time step and characteristic velocity are chosen so that the Courant number is,

$$Cr = \frac{u_0 \Delta t}{\Delta x} = \frac{1}{2}.$$

The scalar field is initialized to a square wave with wavelength equal to domain size and with the step changes coinciding with the centers of vortices as shown in Fig 4.10. The scalar field at non dimensional time,

$$t^* = t\frac{u_0}{l_0} = 5.8\dot{3},$$

using 840 cells is shown in Fig 4.11. Results using the UTOPIA and NEWTOPIA schemes are indistinguishable so only one is included. Both schemes effectively capture the sharp gradients generated and exhibit some slight over and under shoot as shown in blue and red. As with the rotating cylinder test, the flux limiters effectively prevent this inconsistency.

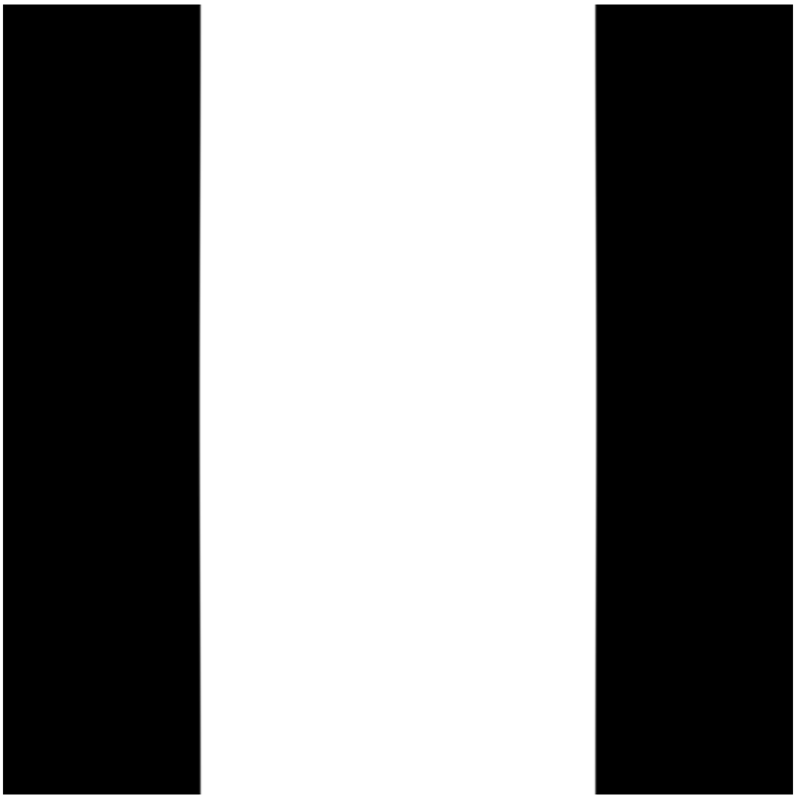


Figure 4.10: Taylor vortex advection test – Initial scalar field.

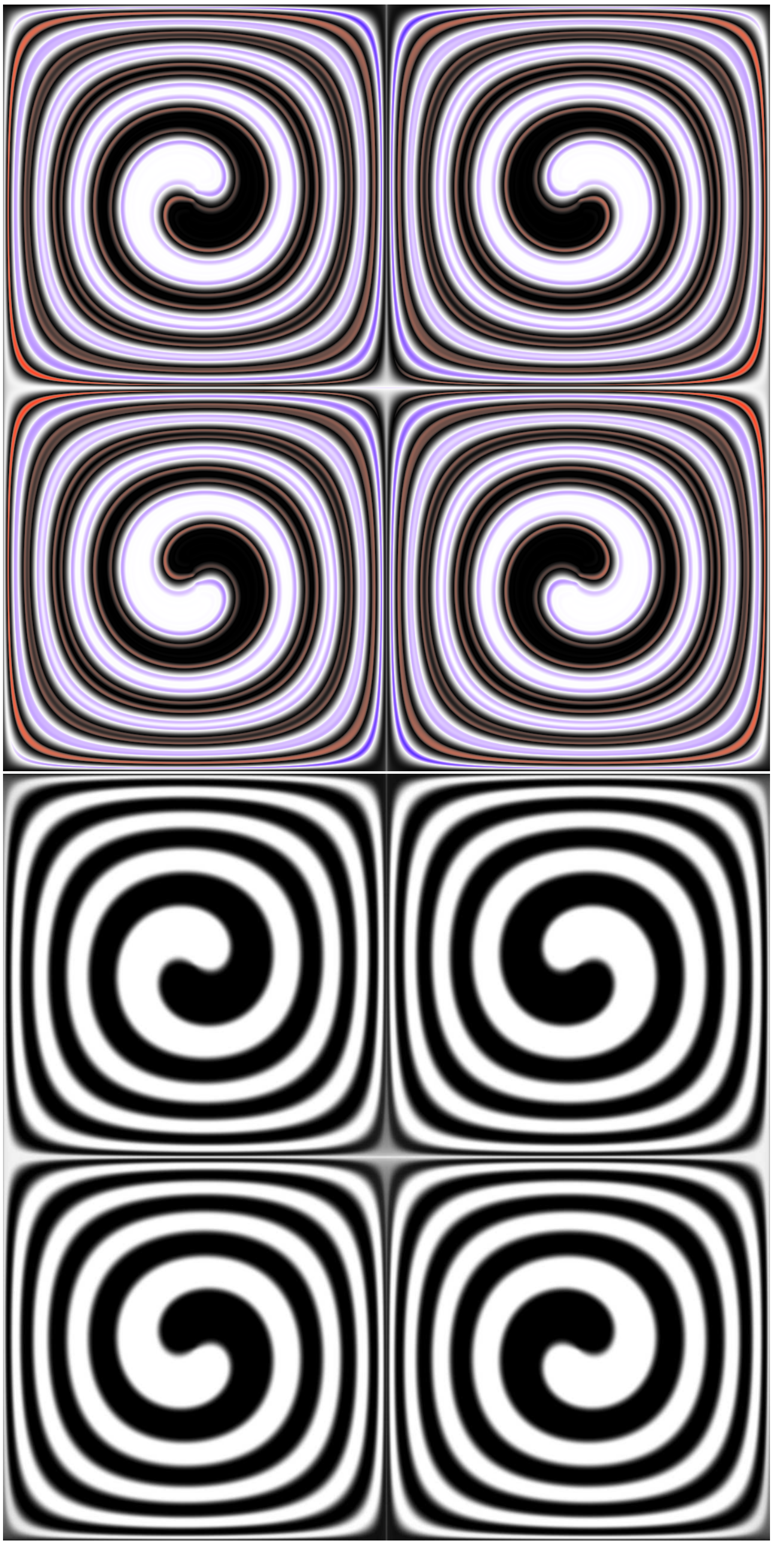


Figure 4.11: Taylor vortex advection test. Top shows over and under shoot of NEWTOPIA and UTOPIA schemes in blue and red. Flux limited scheme, shown in bottom panel remains bounded.

4.3 Projection methods

Projection methods, also referred to as fractional step methods or predictor-corrector methods are commonly used to simulate incompressible fluid flow. The solution of the pressure field is de-coupled from the solution of the velocity field. The pressure is no longer considered a free thermodynamic variable related to density, instead the density is constant and the pressure is a scalar field whose gradient enforces a divergence free velocity. In the low Mach number limit, the speed of sound is sufficiently high that the time taken for acoustic perturbations to reach equilibrium is effectively instantaneous allowing changes in one part of the domain to instantly be communicated to the entire domain. It is possible to express the incompressible Navier-Stokes equations as a projection without using pressure at all.

Chorin's projection method [47] is the classic example. Approximating the time derivative with a first order Euler finite difference allows the velocity at the next time step to be written in terms of the velocity and pressure at the current time step,

$$u_\alpha^{t+} = u_\alpha^t + \Delta t \left(-\partial_\alpha P^t + \rho\nu\partial_\beta\partial_\beta u_\alpha^t - \rho u_\beta^t \partial_\beta u_\alpha^t \right).$$

The pressure at the next time step can then be calculated based on this velocity and the Poisson equation for pressure (Eqn. 2.15). Chorin's method is sometimes written in terms of an intermediate velocity,

$$u_\alpha^* = u_\alpha^t + \Delta t \left(\rho\nu\partial_\beta\partial_\beta u_\alpha^t - \rho u_\beta^t \partial_\beta u_\alpha^t \right).$$

The Poisson equation that must be solved is then the equivalent,

$$\partial_\alpha\partial_\alpha P = \frac{1}{\Delta t}\partial_\alpha u_\alpha^*$$

and the true velocity is found by the correction,

$$u_\alpha^{t+} = u_\alpha^* - \Delta t \partial_\alpha P.$$

A more accurate time stepping scheme such as the second order implicit time centered Crank-Nicolson scheme may be used on the momentum,

$$\frac{1}{\Delta t}\left(u_\alpha^{t+} - u_\alpha^t\right) + \frac{1}{2}(\partial_\alpha P^{t+} + \partial_\alpha P^t + \rho u_\beta^{t+}\partial_\beta u_\alpha^{t+} + \rho u_\beta^t \partial_\beta u_\alpha^t \\ - \rho\nu\partial_\beta\partial_\beta u_\alpha^{t+} - \rho\nu\partial_\beta\partial_\beta u_\alpha^t) = 0$$

and mass equations,

$$\partial_\alpha u_\alpha^{t+} = 0.$$

In order to arrive at a linear system, the advective terms may be linearized. The resulting linear system contains DIM + 1 unknowns, u_α^{t+} and P^{t+} and has the same number of equations so it can be solved in principle. Unfortunately the problem is poorly conditioned [248] so direct solution is not often used.

An iterative scheme may be used to determine a solution,

$$\frac{1}{\Delta t}\left(u_\alpha^{*} - u_\alpha^t\right) + \frac{1}{2}(\partial_\alpha P^{t+,k} + \partial_\alpha P^t + \rho u_\beta^{*}\partial_\beta u_\alpha^{*} + \rho u_\beta^t \partial_\beta u_\alpha^t \\ - \rho\nu\partial_\beta\partial_\beta u_\alpha^{*} - \rho\nu\partial_\beta\partial_\beta u_\alpha^t) = 0$$

with the next velocity estimate corrected,

$$\frac{1}{\Delta t}(u_\alpha^{t+} - u_\alpha^{*}) = -\frac{1}{2}(\partial_\alpha P^{t+,k+1} - \partial_\alpha P^{t+,k}).$$

The pressure correction is solved in terms of the intermediate velocity,

$$\frac{1}{\Delta t}\partial_\alpha u_\alpha^{*} = \frac{1}{2}(\partial_\alpha\partial_\alpha P^{t+,k+1} - \partial_\alpha\partial_\alpha P^{t+,k}),$$

or directly,

$$\frac{1}{2}(\partial_\alpha\partial_\alpha P^{t+,k+1} + \partial_\alpha\partial_\alpha P^t) = \frac{1}{2}(\rho\nu\partial_\beta\partial_\beta u_\alpha^{*} + \rho\nu\partial_\beta\partial_\beta u_\alpha^t - \rho u_\beta^{*}\partial_\beta u_\alpha^{*} - \rho u_\beta^t\partial_\beta u_\alpha^t).$$

As the pressure correction approaches zero, the pressure and velocity approach the second order time accurate values of the Crank-Nicolson scheme. Van Kan's method [248] is equivalent to the first iteration of this scheme and it is shown in his paper that only one iteration is sufficient to give an approximation for velocity that is second order accurate in time.

The iterative procedure is in some ways similar to Patankar's SIMPLE algorithm [172, 170] which was formulated as a means of solving the steady state equations using finite volumes. Patankars work also spawned the SIMPLER (revised) method [171] and the SIMPLEC (consistent) method [247]. Jang et al. [116] give a comparison between these schemes and Issa's PISO scheme [114] which is likewise similar to two iterations of the pressure correction method. Bell [11] also presents a very similar algorithm where one iteration gives a second order accurate velocity field that can be further iterated to give a more accurate pressure. There is a small but key difference in that Bell calculates the mid timestep pressure directly which results in his scheme missing the factor of 1/2 in the pressure correction equation. There are other schemes too, Runge-Kutta (RK) schemes may be implemented in a similar fashion to Chorin's method where a PPE is solved for each partial time step based on the velocity calculated in the RK method [204, 205, 118].

Armfield and Street [8, 9] discuss these schemes and demonstrate their accuracy using numerical experiments. He and Sun [106] produce some rigorous results on the convergence of these schemes in the context of a finite element Galerkin discretization. He [105] extends this work to include non-smooth initial data.

The simplest pressure boundary condition, often used at no slip walls is a zero normal gradient. This amounts to dropping the term $\partial_x P = \rho\nu\partial_x\partial_x u_x$ and may be an acceptable approximation if the viscosity is small although Gresho [92] has the following to say,

> *" ... the scheme thus appears to be both hare brained and doomed."*

Despite the harsh words, Gresho goes on to show that the errors introduced by this BC are minor especially away from the wall. Kim and Moin [123] use a BC involving the pressure from the previous time step.

The iterative schemes mentioned above do not solve a Poisson equation for pressure but rather a Poisson equation for the change in pressure between iterations of the inner loop. Since this correction approaches zero, the zero gradient boundary condition is appropriate. Likewise, since the intermediate velocity approaches the actual velocity, using the velocity BC on the intermediate velocity is also natural. These simple boundary conditions are sufficient to give second order accuracy in both space and time even when only iteration of the algorithm is performed as shown in the original papers [248, 11] and is also confirmed by many subsequent papers such as Gresho [92], Gresho and Chan [93], Shen [215, 214] and Armfield and Street [8, 9] to name a few examples.

4.4 Compressible schemes

The advection terms in the compressible Navier-Stokes equations (Eqn. 2.14) cannot be solved by the flux integral schemes of this chapter. Boundaries between finite volume cells represent Riemann problems that must be solved at each time step. The second order TVD schemes, mentioned in Section 4.2.5 can be used in order to create a monotonic reconstruction within cells (sometimes called a MUSCL reconstruction) which is then used with a Riemann solver such as Roe's approximate Riemann solver [199] or a Riemann free scheme such as the Kurganov and Tadmor (KT) scheme [129] or its predecessor the Nessyahu and Tadmor (NT) scheme [162].

There are also a family of so called artificial compressibility (AC) methods, originating with Chorin's scheme [46]. These schemes do not attempt to solve the inter-cell Riemann problems correctly and generally need to employ some means of stabilization [192, 165] to yield acceptable results. Ohwada et al [166] compare these AC schemes to low Mach number lattice Boltzmann (LB) schemes by analysing the diffusive and acoustic timescales of the former using a type of multi-scale analysis typically employed in the analysis of LB methods. They perform numerical experiments using a Taylor-Green vortex and find improved accuracy using AC methods as expected due to their use of a higher order difference scheme.

In order to compare the performance and effectiveness of LB methods to compressible methods, we have implemented the KT scheme using the midpoint rule (a second order Runge-Kutta scheme) for time stepping and the MINMOD limiter function. We have simplified the governing equations by assuming a constant temperature. While this is a non-physical assumption, it was chosen in order to make the scheme similar to the LB methods we are comparing against. We have copied the implementation details directly from the original paper [129]. It is also possible to use higher order time stepping schemes from Section 4.1 however again, we have not done so in order to keep a similar order of accuracy to the LB methods we are comparing against.

Due to the explicit and local nature of these schemes, a similar characteristic to LB methods, good performance on graphics processing units (GPUs) is expected. We have found that the programming effort required to implement the KT scheme on the GPU is higher than for LB methods, this is true even with the simplified governing equations and time stepping scheme we have chosen. The performance is significantly lower, time taken is increased by 731% in our test below. The main advantages of the KT scheme is in its stability and potential for higher order accuracy, even when extended to thermal compressible flows with discontinuities. We have decided to continue our use of lattice Boltzmann methods as a quasi-incompressible solver due to their high performance.

We have run a sample problem using our GPU based KT scheme and benchmarked it against a `cub_d3q19_ab` lattice based LB method, also programmed for the GPU. The test was performed on a machine running Arch Linux with Linux kernel version 4.1.6, `gcc` version 5.2.0 and an Nvidia GTX580 GPU running driver version 352.41 and CUDA version 7.5.18.

The domain size used for testing was $l = 222$ cells with periodic boundaries. The sample problem is a zero velocity initial condition with a step change in density within a sphere centered in the domain. This test was chosen to demonstrate the ability of the KT scheme to deal with discontinuity. The density jumps from 1.0 outside the sphere to 1.1 inside the sphere. The viscosity is 1.6835×10^{-3}, this corresponds to a relaxation factor of 1.98 for the LB method. The time step and grid size are unity. The speed of sound is 0.57735 for the LB method and has to be reduced to 0.26943 with the KT scheme to ensure stability due to the CFL condition, this number was chosen so that the speeds of sound of the two schemes were related by a neat number,

$$\frac{c_s^{\mathrm{LB}}}{c_s^{\mathrm{KT}}} = \frac{150}{70}.$$

The non-dimensional time is defined as,

$$t^* = t * \frac{c_s}{l}.$$

Pseudocolor plots of the density at two non-dimensional times are shown in Figure 4.12 and Figure 4.13. Due to the both the faster update speed and its higher speed of sound, the lattice Boltzmann solver took 22.5 seconds while the KT scheme took 164.4 seconds, making the LB solver 731% faster overall. It is clear that the lattice Boltzmann scheme suffers from spurious oscillations near the sharp density gradients however these do not ultimately lead to any catastrophic failure. If the solution is sufficiently well resolved, the LB method will not introduce these oscillations. The LB method uses significantly more memory. The KT scheme uses 8 values per cell, new and old density and velocity while the LB scheme must store 23 values, the 19 particle velocity probability distribution functions, density and 3 velocities.

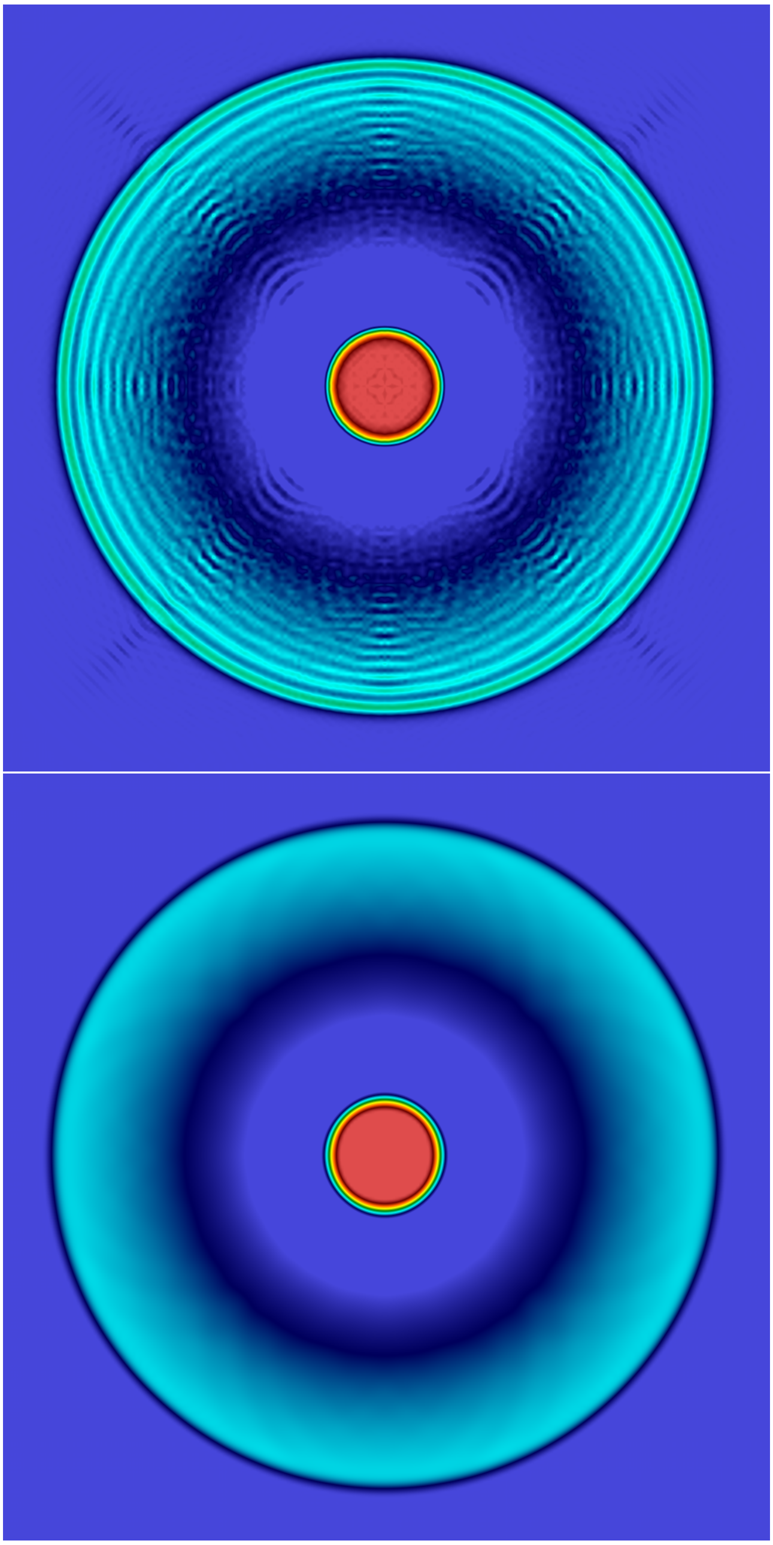


Figure 4.12: Ball test at $t^* = 0.1820$ with lattice Boltzmann simulation in top panel and Kurganov-Tadmor scheme in bottom.

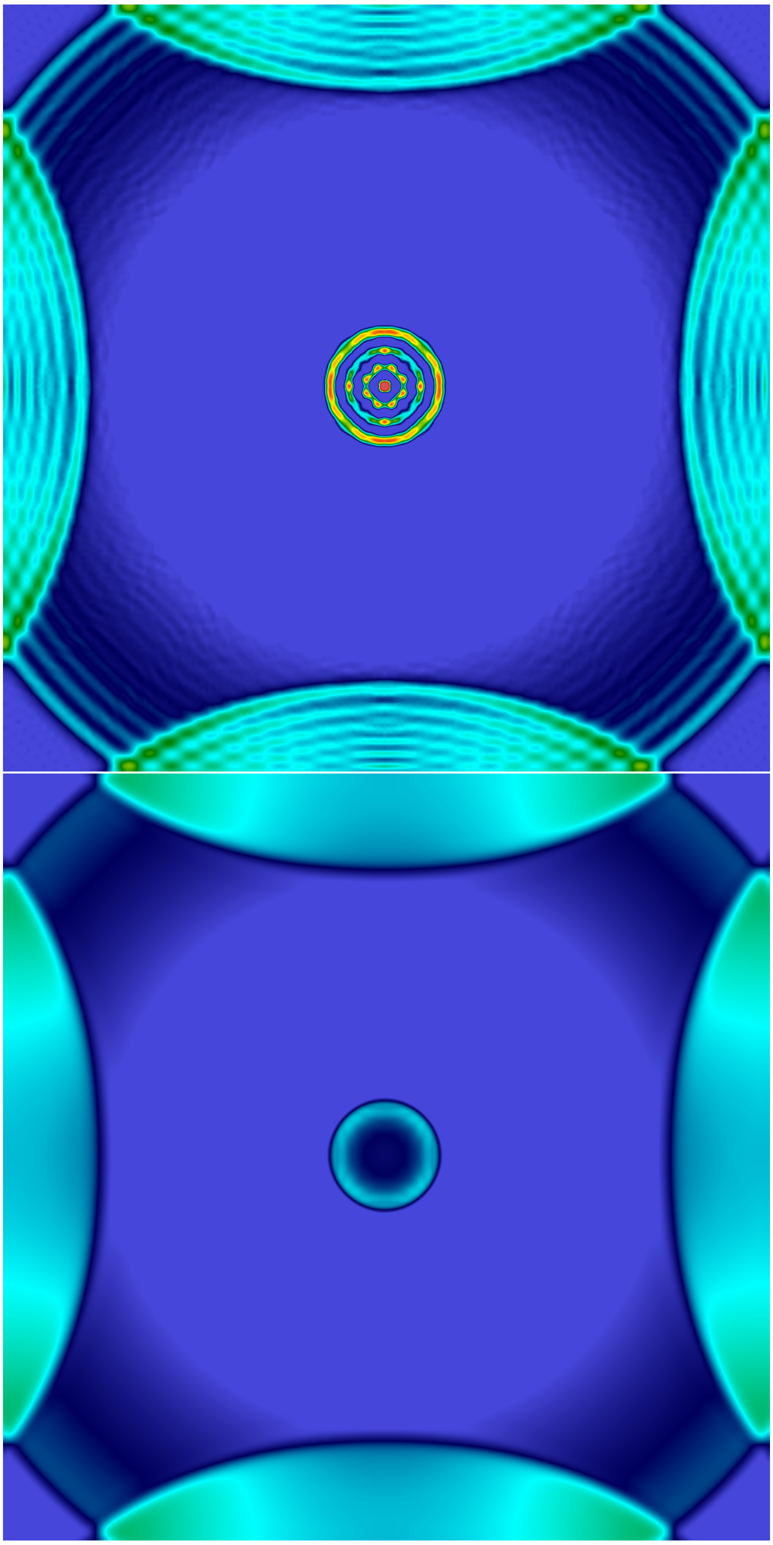


Figure 4.13: Ball test $t^* = 0.3641$ with lattice Boltzmann simulation in top panel and Kurganov-Tadmor scheme in bottom.

4.5 Boussinesq approximation – Our coupled scheme

The Boussinesq approximation assumes that density is constant except in the buoyancy term where the body force per unit volume due to gravity is,

$$B_\alpha = \rho g_\alpha \quad \left(\frac{kg}{m^2 s^2}\right),$$

where g_α is the acceleration vector due to gravity. It is convenient to subtract the component due to the average density which has no effect on an incompressible solution,

$$B_\alpha = g_\alpha(\rho - \rho_0).$$

We replace the density variation with some scalar,

$$\rho - \rho_0 = \phi.$$

The evolution of the scalar follows the advection diffusion equation,

$$\partial_t \phi + \partial_\alpha u_\alpha \phi = \alpha \partial_\alpha \partial_\alpha \phi.$$

The buoyancy force density is coupled to the lattice Boltzmann solver via the force term,

$$F_\alpha = B_\alpha = \phi g_\alpha.$$

We will use a lower case b to denote the buoyancy force per unit mass, the acceleration due to buoyancy,

$$b_\alpha = \frac{B_\alpha}{\rho_0} \quad \left(\frac{m}{s^2}\right).$$

The scalar density can be used to represent changes in temperature or concentration when dealing with mixtures of components with different densities such as salinity gradients in water. We have found that third order flux integral schemes with Thuburn's flux limiter for the buoyancy scalar coupled in this way with lattice

Boltzmann methods for the hydrodynamic equations results in a highly capable solver for buoyancy driven flows. This solver has been implemented to work in parallel across a cluster of machines with graphics processing units and achieves high computational throughput and parallel efficiency. We will present numerical simulations of various buoyancy driven problems in Chapter 5.

Chapter 5

Numerical simulations

This chapter begins with a section outlining the performance of our code. Next we discuss laminar channel flow and Taylor-Green vortex flow simulations which aim to verify the order of accuracy of the lattice Boltzmann schemes. Next we have simulated turbulent channel flow in order to benchmark our code against established direct numerical simulation results. We also show linearly forced homogeneous turbulence. We present some figures showing various simulations performed of Kelvin-Helmholtz and Holmboe instabilities, Rayleigh-Benard and Rayleigh-Taylor convection. We finally present simulations over a range of parameters of sheared convective boundary layers.

5.1 Performance

I have developed a code named BX that uses the lattice Boltzmann methods and coupled finite volume buoyancy solver that have been discussed in previous sections. The code has been written mostly in the C programming language with code that executes on the graphics processing units (GPUs) using Nvidia's CUDA language. Domain decomposition using one dimensional slicing is performed in order to run in parallel across multiple machines communicating using MPI. Xian and

Takayuki [258] have investigated the use of two and three dimensional slicing using pencils and cubes in order to increase parallel efficiency finding better results with higher order decomposition as expected. We have calculated that for domain sizes currently feasible on our cluster, parallel communication overhead is not a significant issue with one dimensional slicing. Our numerical experiments also support this assertion.

Thanks to generous funding from my supervisor, I have set up a small cluster to perform the simulations presented in this thesis. The cluster consists of 8 machines which have been obtained after being superseded from the school's cluster. They use Intel E8400 Core 2 Duo processors. The outdated processors are not a problem since minimal calculations are performed on the CPU. We have purchased 8 Nvidia GTX580 GPUs which perform the bulk of the calculations. Parallel efficiency has been improved by fitting the machines with an Infiniband interconnect. After testing several versions, OpenMPI 1.6.5 was determined to work optimally with the other libraries and Infiniband hardware. I have set the machines up running Debian Linux 7 and found optimal stability using an older version of Nvidia's drivers (295.75) and CUDA (4.2). The cluster uses standard UNIX utilities, NIS to synchronise logins and NFSv4 to provide a shared filesystem.

Performance results in terms of mega lattice updates per second (MLUPS) are presented in Table 5.1. Where multiple GPUs are used the performance is listed per card. The column labeled "Update" specifies the lattice Boltzmann update algorithm used. Most LB methods use what is known as the two-step algorithm where the collision step is performed locally and then streaming is performed in a second pass. The Lagrangian nature of streaming allows an in place swap to occur between adjacent lattice sites. This two-step algorithm only needs to store one copy of the lattice data however performance is slightly degraded since lattice data needs to be read during each pass. In contrast, the two-lattice update algorithm keeps two copies of lattice data, swapping between the two at each time step and updating in one pass increasing performance.

Tests were performed using a domain size of $480 \times 152 \times 80$ nodes in the x, y and z directions. Our code always ensures that array dimensions in the primary indexing direction are a multiple of 32 in order to maximise performance on Nvidia's GPU architecture. In cases where the domain size in the x direction is not a multiple of 32 we manually pad all arrays. This procedure also ensures optimal memory alignment. As a result, performance does not significantly depend on domain size as long as the total number of nodes is sufficiently high. This is because the GPU needs a large number of threads to be 'in flight' compared to the number of individual processing units so that execution may be overlapped with memory access in order to hide latency. The domain we have used results in around 6 million threads which is ample.

Comparing the performance of the `cub_d3q19_ab` lattice between two types of CPU and two types of GPU makes the high throughput of the GPU apparent. The speed up moving from the Intel i7 920 CPU using all four cores to the GTX580 is a factor of 130 in single precision and 50 in double precision. The large difference between single and double precision results is due to the architecture of the Nvidia GTX processors. The double precision performance is crippled by the manufacturer to a factor of one eighth of the single precision performance by laser cutting the relevant circuits on the die in order to drive sales of more expensive Tesla models. Tests using the same lattice comparing performance of the GTX285 and GTX580 show the large increases in performance which have been characteristic of GPU development cycles.

Parallel scaling for the `cub_d3q19_ab` lattice is between 97% and 100% as shown by the performance figures such as GTX580 × 2 meaning the per card performance when running across two GPUs. We have overlaid parallel execution and communication using threads in order to eliminate latency. Due to our implementation, there is no performance penalty using multiple machines as long as the interconnect bandwidth is sufficient, as is the case for our cluster and other clusters that we have tested.

These speeds are impressive when compared to traditional incompressible finite-volume based Navier-Stokes solvers. We have tested Puffin, a highly developed incompressible finite volume based research code [126]. Performance of this code on a similar turbulent channel flow configuration has been measured at 0.8 MLUPS per Intel Xeon processor core. In this context we can achieve higher overall throughput on a single GPU using our LB code than we could using the finite volume code on 128 processor cores of a research cluster. The Courant number used in simulations must also be taken into account, the finite volume simulation was performed using a Courant number of 0.15 while the LB method used an effective Courant number of 0.057 closing the performance gap. We calculate the rate of simulation time to wall time and compare a single GTX580 to a single Intel Xeon processor core in order to obtain a speedup factor of 70:1 for our code. In other words a single GPU is equivalent to around 70 CPU cores.

We have also tested Flamenco, a state of the art compressible finite volume code [241]. This code achieves around 0.02 MLUPS per Intel Xeon processor core. This code remains stable using a Courant number up to one due to it's use of a strong stability preserving Runge-Kutta tine stepping scheme. A Mach number of 0.1 is used, as with lattice Boltzmann simulations in order to reduce compressibility effects results in an effective Courant number of 0.1 compared to incompressible simulations. A comparison of simulation time per wall time between the single GTX580 and a single Intel Xeon processor core running FLAMENCO results in a 4200:1 speedup factor using our GPU code.

It should also be noted that the double precision performance of the GTX580 is severely crippled by the manufacturer, the same manufacturer also supplies GPUs with four times the double precision throughput however we have not tested those devices at present. The GTX580 is also an aging model from 2010, newer models by the same manufacturer offer significantly improved performance.

Addition of the flux limited NEWTOPIA scheme results in approximately 40% performance penalty due to the additional effort of calculating updates to the buoyancy field. These results are shown in rows where the "Buoyancy" column contains the

Lattice	Prec.	Update	Buoyancy	Proc.	Speed (MLUPS)
cub_d3q19_ab	float	two step	-	x6 1090t	2.4
cub_d3q19_ab	double	two step	-	x6 1090t	2.1
cub_d3q19_ab	float	two step	-	i7 920	3.2
cub_d3q19_ab	double	two step	-	i7 920	3.0
cub_d3q19_ab	float	two step	-	GTX285	71
cub_d3q19_ab	double	two step	-	GTX285	42
cub_d3q19_ab	float	two step	-	GTX580	428
cub_d3q19_ab	double	two step	-	GTX580	149
cub_d3q19_ab	float	two step	-	GTX580 × 2	422
cub_d3q19_ab	double	two step	-	GTX580 × 2	145
cub_d3q19_ab	float	two step	-	GTX580 × 4	428
cub_d3q19_ab	double	two step	-	GTX580 × 4	145
cub_d3q19_ab	float	two step	-	GTX580 × 8	422
cub_d3q19_ab	double	two step	-	GTX580 × 8	145
cub_d3q19_ab	float	two step	fluxlim	GTX580	258
cub_d3q19_ab	double	two step	fluxlim	GTX580	91
cub_d3q27_abc	float	two step	-	GTX580	300
cub_d3q27_abc	double	two step	-	GTX580	105
cub_d3q27_abc	float	two lat	-	GTX580	308
cub_d3q27_abc	double	two lat	-	GTX580	127
cub_d3q27_abc	float	two lat MRT	-	GTX580	336
cub_d3q27_abc	double	two lat MRT	-	GTX580	129

Table 5.1: Performance of various solvers, settings and platforms (per processor for multiple GPU cases).

text fluxlim. Other simulations are for hydrodynamics only. The explicit and local nature of the NEWTOPIA scheme means that the parallel efficiency is similarly high.

Tests using the cub_d3q27_abc lattice demonstrate the high computational efficiency of our MRT eigendecomposition. The two lattice algorithm improves performance from the two step algorithm and then the MRT algorithm improves performance again. Typically MRT algorithms have worse performance than conventional single relaxation time techniques due to the number of matrix operations required. As explained in the previous chapter, our sparse matrix implementation dramatically reduces the number of operations required.

5.2 Laminar channel flow

Laminar channel flow is used to test the numerical accuracy of lattice Boltzmann methods. This simple steady state flow requires either a pressure gradient or equivalently a body force to overcome friction at the wall. The analytical solution is,

$$
\begin{aligned}
u &= u_0 \left[1 - \frac{4}{\delta^2} \left(y - \frac{\delta}{2} \right)^2 \right], \\
v &= 0, \\
P &= 0.
\end{aligned}
$$

The x momentum equation gives a relationship for the body force,

$$
\begin{aligned}
F_\alpha &= -\rho\nu\partial_\beta\partial_\beta u_\alpha, \\
F_x &= \frac{8u_0\rho\nu}{\delta^2}.
\end{aligned}
$$

The flow only has gradients in the y direction and no contribution from the advective terms of the Navier-Stokes equations meaning that the full capabilities of the numerical scheme are not utilized or tested, only the balance between viscous terms and the body force is tested. We have performed simulations using the standard single relaxation time LB method with Lagrangian streaming and our modified time stepping scheme with a maximum Courant number of one half. The modified time stepping scheme has been tested using both the first order upwind and second order Lax-Wendroff differencing.

The simulation uses the no-slip, bounce back boundary condition at the walls with a channel height of δ. The domain is cube shaped with a uniform aspect ratio. Simulations were performed in three dimensions using the `cub_d3q19_ab` lattice however the flow is one dimensional with no gradients in the span-wise or stream-

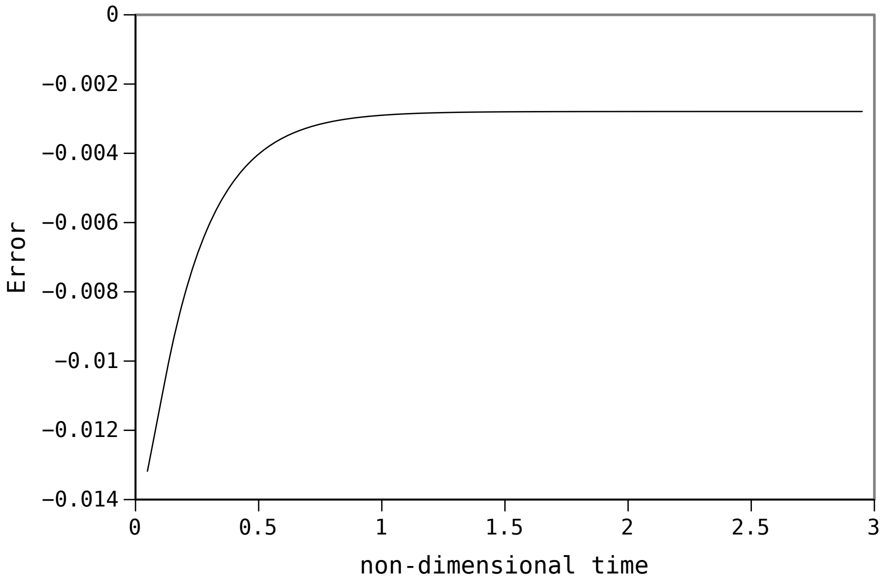


Figure 5.1: Centerline error vs non-dimensional time for laminar channel flow with 192 grid points and a Mach number of 0.08.

wise directions which use periodic boundaries. The Reynolds number,

$$\mathrm{Re} = \frac{u_0 \delta}{\nu},$$

was set to one in all tests in order to ensure laminar flow. The non-dimensional time is defined by,

$$t^* = t\frac{u_0}{\delta}.$$

The flow was allowed to evolve from a uniform zero velocity initial condition until a steady state, represented by $t^* \sim 3$ was reached. By this stage the error value, printed using eight significant figures every 200 time steps was not changing. The time development of error is illustrated in Figure 5.1. Double precision arithmetic was used in all simulations.

Mach numbers ranging from 0.02 → 0.1 were tested using a variety of grid sizes ranging from 64 → 192 nodes. As shown in Figure 5.2 the Mach number dependence is linear. This observation is at odds with the supposed second order in Mach number accuracy of LB methods however the trend is clear – perfectly straight lines

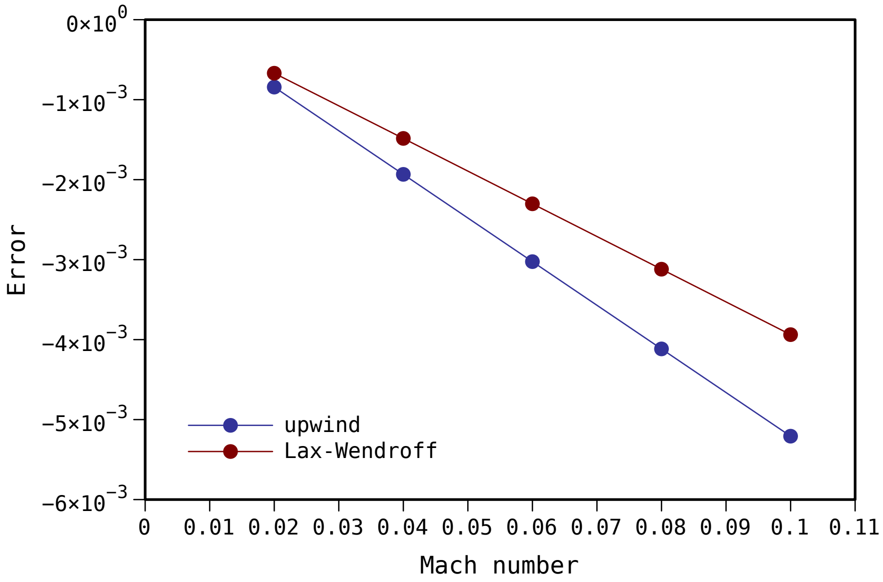


Figure 5.2: Centerline error vs Mach number for laminar channel flow with 128 grid points.

on linear axes for all tests. These results are also inconsistent with the results from the next section using Taylor-Green vortex flow which find the Mach number dependence closer to second order. Figure 5.3 shows second order grid size dependence as expected using zero Mach number extrapolated solutions. The Lax-Wendroff scheme has lower error magnitude but the same order of accuracy. Again these results are not in agreement with Taylor-Green vortex simulations of the next section which show the upwind scheme being more accurate. A possible mechanism that might confound Mach number error dependence is errors that are related to the relaxation rate. As the Mach number is decreased, the viscosity has to be simultaneously decreased in order to maintain the desired Reynolds number. We have found reduced stability and the introduction of non-physical oscillations in other simulations when viscosity was decreased. Although no oscillations were observed in this test, it is nevertheless possible that errors inversely proportional to viscosity were unaccounted for.

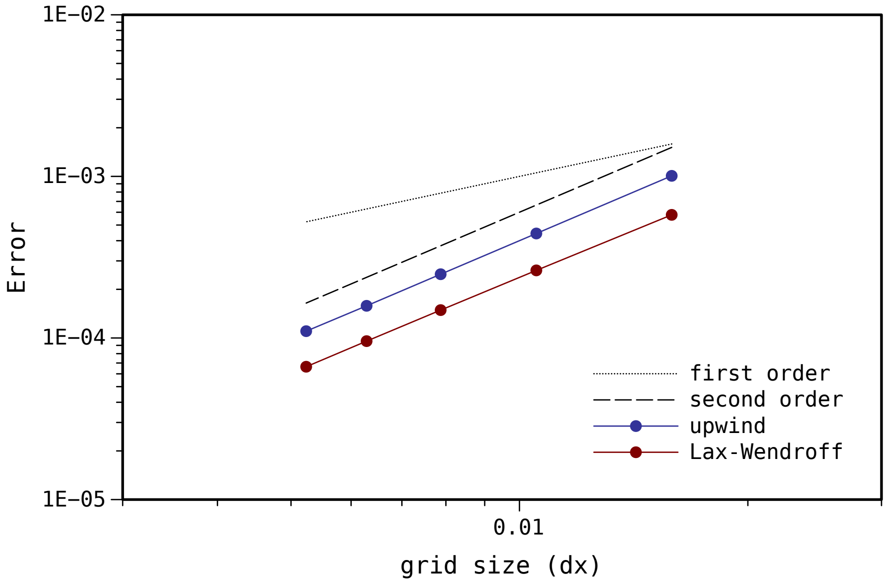


Figure 5.3: Centerline error vs grid size for laminar channel flow, zero Mach number extrapolation.

5.3 Travelling wave

In analysing acoustic problems, a common simplifying assumption is that the flow consists of small acoustic perturbations with zero mean velocity,

$$u_\alpha = \overline{u_\alpha} + u',$$
$$\rho = \overline{\rho} + \rho'.$$

This allows us to some non-linear terms that arise in the Navier-Stokes equations to be dropped. Assuming zero viscosity, the small signal acoustic wave equation is obtained.

$$\overline{\rho}\partial_t u'_\alpha + \partial_\alpha P = 0, \tag{5.1}$$
$$\partial_t \rho' + \overline{\rho}\partial_\alpha u'_\alpha = 0. \tag{5.2}$$

Using the simplified equation of state,

$$P = c_s^2 \rho',$$

and taking a time derivative of Eqn. 5.1 and a space derivative of Eqn. 5.2 results in,

$$\partial_t \partial_t u'_\alpha - c_s^2 \partial_\alpha \partial_\beta u'_\beta = 0.$$

Likewise a similar equation for the density perturbation is obtained.

$$\partial_t \partial_t \rho' - c_s^2 \partial_\alpha \partial_\alpha \rho' = 0.$$

These are standard forms of the wave equation, restricted to one dimension,

$$\partial_t \partial_t u' - c_s^2 \partial_x \partial_x u' = 0. \tag{5.3}$$

A general solution is found by setting,

$$\begin{aligned} \xi &= x - c_s t, \\ \eta &= x + c_s t. \end{aligned}$$

Using the chain rule,

$$\begin{aligned} \frac{\partial}{\partial x} &= \frac{\partial \eta}{\partial x}\frac{\partial}{\partial \eta} + \frac{\partial \xi}{\partial x}\frac{\partial}{\partial \xi} & \frac{\partial}{\partial t} &= \frac{\partial \eta}{\partial t}\frac{\partial}{\partial \eta} + \frac{\partial \xi}{\partial t}\frac{\partial}{\partial \xi} \\ &= \frac{\partial}{\partial \eta} + \frac{\partial}{\partial \xi}, & &= c_s\left(\frac{\partial}{\partial \eta} - \frac{\partial}{\partial \xi}\right), \\ \frac{\partial}{\partial x}\frac{\partial}{\partial x} &= \left(\frac{\partial}{\partial \eta} + \frac{\partial}{\partial \xi}\right)\left(\frac{\partial}{\partial \eta} + \frac{\partial}{\partial \xi}\right) & \frac{\partial}{\partial t}\frac{\partial}{\partial t} &= c_s^2\left(\frac{\partial}{\partial \eta} - \frac{\partial}{\partial \xi}\right)\left(\frac{\partial}{\partial \eta} - \frac{\partial}{\partial \xi}\right) \\ &= \frac{\partial^2}{\partial \eta^2} + 2\frac{\partial^2}{\partial \eta \partial \xi} + \frac{\partial^2}{\partial \xi^2}, & &= c_s^2\left(\frac{\partial^2}{\partial \eta^2} - 2\frac{\partial^2}{\partial \eta \partial \xi} + \frac{\partial^2}{\partial \xi^2}\right). \end{aligned}$$

Substituting the partial derivatives into the one dimensional wave equation Eqn. 5.3 results in,

$$\frac{\partial^2 u'}{\partial\eta\partial\xi} = 0$$

The general solution to which is a combination of any twice differentiable functions F and G,

$$\begin{aligned} u'(\xi,\eta) &= F(\xi) + G(\eta) \\ u'(x,t) &= F(x - c_s t) + G(x + c_s t). \end{aligned} \tag{5.4}$$

Viscous damping may also be included,

$$\begin{gathered} \overline{\rho}\partial_t u'_\alpha = -\partial_\alpha P + \overline{\rho}\nu\partial_\beta\partial_\alpha u'_\beta + \overline{\rho}\nu_b\partial_\alpha\partial_\beta u'_\beta, \\ \partial_t\rho' + \overline{\rho}\partial_\alpha u'_\alpha = 0, \end{gathered}$$

although the travelling wave solution Eqn. 5.4 is no longer applicable. Combining the momentum and density equations as before,

$$\partial_t\partial_t u'_\alpha - c_s^2\partial_\alpha\partial_\beta u'_\beta = \nu\partial_t\partial_\beta\partial_\alpha u'_\beta + \nu_b\partial_t\partial_\alpha\partial_\beta u'_\beta$$

Restricted to one dimension again,

$$\partial_t\partial_t u' - c_s^2\partial_x\partial_x u' = (\nu + \nu_b)\partial_t\partial_x\partial_x u'$$

Solutions using travelling sine waves with an added exponential decay term are tested. The density will also need to have cosine factors to balance terms appearing in the time derivative.

$$\begin{aligned} u' &= u_0 e^{-\alpha t}\{\sin(k(x + c_p t)) + \sin(k(x - c_p t))\}, \\ \rho' &= e^{-\alpha t}\{\, A\sin(k(x + c_p t)) + C\sin(k(x - c_p t)) \\ &\qquad B\cos(k(x + c_p t)) + D\cos(k(x - c_p t))\}. \end{aligned}$$

Here, c_p is now the phase velocity which may not be equal to the speed of sound c_s.

$$\begin{aligned}
\partial_x u' &= u_0 k e^{-\alpha t}\{\cos(k(x+c_p t)) + \cos(k(x-c_p t))\},\\
\partial_t \rho' &= e^{-\alpha t}\{ -\alpha A \sin(k(x+c_p t)) - \alpha B \cos(k(x+c_p t))\\
&\qquad + k c_p A \cos(k(x+c_p t)) - k c_p B \sin(k(x+c_p t))\\
&\qquad - \alpha C \sin(k(x-c_p t)) - \alpha D \cos(k(x-c_p t))\\
&\qquad - k c_p C \cos(k(x-c_p t)) + k c_p D \sin(k(x-c_p t))\}.
\end{aligned}$$

In order to cancel the sine terms the following relationships are obtained,

$$\begin{aligned}
-\alpha A &= -k c_p B, & B &= \frac{-\alpha A}{k c_p},\\
-\alpha C &= k c_p D, & D &= \frac{\alpha C}{k c_p}.
\end{aligned}$$

In order to balance the conservation of mass the following relationships are obtained,

$$\begin{aligned}
-\alpha B + A k c_p &= -u_0 \overline{\rho} k, & A &= \frac{-u_0 \overline{\rho} k}{\frac{\alpha^2}{k c_p} + k c_p}, & B &= \frac{\alpha u_0 \overline{\rho} k}{\alpha^2 + k^2 c_p^2},\\
-\alpha D - C k c_p &= -u_0 \overline{\rho} k, & C &= \frac{u_0 \overline{\rho} k}{\frac{\alpha^2}{k c_p} + k c_p}, & D &= \frac{\alpha u_0 \overline{\rho} k}{\alpha^2 + k^2 c_p^2}.
\end{aligned}$$

The equation for density is then,

$$\begin{aligned}
\rho' = \frac{u_0 \overline{\rho}}{c_p \left(\frac{\alpha^2}{k^2 c_p^2} + 1\right)} e^{-\alpha t} \Big\{ &\sin(k(x-c_p t)) - \sin(k(x+c_p t))\\
&+ \frac{\alpha}{k c_p} \cos(k(x-c_p t)) + \frac{\alpha}{k c_p} \cos(k(x+c_p t)) \Big\}.
\end{aligned}$$

Substituting into the differential equations,

$$
\begin{aligned}
\partial_x\partial_x u' &= -u_0 k^2 e^{-\alpha t}\{\sin(k(x+c_p t)) + \sin(k(x-c_p t))\},\\
\partial_t u' &= u_0 e^{-\alpha t}\{\, kc_p\cos(k(x+c_p t)) - kc_p\cos(k(x-c_p t))\\
&\qquad - \alpha\sin(k(x+c_p t)) - \alpha\sin(k(x-c_p t))\},\\
\partial_t\partial_t u' &= u_0 e^{-\alpha t}\{\, -k^2c_p^2\sin(k(x+c_p t)) - k^2c_p^2\sin(k(x-c_p t))\\
&\qquad - \alpha k c_p\cos(k(x+c_p t)) + \alpha k c_p\cos(k(x-c_p t))\\
&\qquad + \alpha^2\sin(k(x+c_p t)) + \alpha^2\sin(k(x-c_p t))\\
&\qquad - \alpha k c_p\cos(k(x+c_p t)) + \alpha k c_p\cos(k(x-c_p t))\},\\
\partial_x\partial_x\partial_t u' &= -u_0 k^2 e^{-\alpha t}\{\, kc_p\cos(k(x+c_p t)) - kc_p\cos(k(x-c_p t))\\
&\qquad\quad - \alpha\sin(k(x+c_p t)) - \alpha\sin(k(x-c_p t))\}.
\end{aligned}
$$

Equating sine and cosine terms respectively gives,

$$
\begin{aligned}
\alpha^2 - k^2c_p^2 + c_s^2k^2 &= \nu_{\text{tot}}k^2\alpha,\\
2\alpha k c_p &= \nu_{\text{tot}}k^3c_p.
\end{aligned}
$$

These imply that,

$$\alpha = \frac{\nu_{\text{tot}}k^2}{2},$$

and

$$\frac{\nu_{\text{tot}}^2k^4}{4} - k^2c_p^2 + c_s^2k^2 = \frac{\nu_{\text{tot}}^2k^4}{2} \quad\Rightarrow\quad c_p = \sqrt{c_s^2 - \frac{\nu_{\text{tot}}^2k^4}{4}}. \tag{5.5}$$

The dispersion relationship Eqn. 5.5 results in a different rate of propagation depending on frequency. The sine wave (and cosine) solutions are thus only affected by an attenuation rate however more complicated waveforms will now also be deformed by the dispersive effects.

At this stage we have implemented a standing and traveling wave initial condition and performed several simulations to verify the speed of sound. We have confirmed the speed of sound of the `bcc_d3q15_ab`, `cub_d3q27_abc` and `bcc_d3q27_abc` lattices. Future work will be to confirm dispersion relationships for various schemes.

5.4 Taylor-Green vortex

The two dimensional Taylor-Green vortex (TGV) is used as a more advanced test for the order of accuracy of LB methods. In contrast to laminar channel flow, the TGV has gradients in two directions resulting in a more thorough test utilizing both advective and viscous terms of the Navier-Stokes equations. The analytical solution to this time varying flow is given by,

$$\begin{aligned} u(x,y,t) &= \quad u_0 e^{-2k_0^2 \nu t} \cos(k_0 x) \sin(k_0 y), \\ v(x,y,t) &= -u_0 e^{-2k_0^2 \nu t} \sin(k_0 x) \cos(k_0 y), \\ P(x,y,t) &= P_0 + \frac{u_0^2}{4} e^{-4k_0^2 \nu t} \cos(2k_0 x) \cos(2k_0 y). \end{aligned} \tag{5.6}$$

We have used a wavelength equal to the domain size,

$$k_0 = l_0^{-1}.$$

Various simulations were performed however unfortunately, a clear picture regarding the order of accuracy has not emerged [263]. The following results were obtained at a Reynolds number of 10 using grid sizes of 96, 128 and 256 cells. Result were also obtained using a stretched grid with a 2 : 1 aspect ratio and our rectangular grid scheme described in Section 3.10. Stretched grid simulations use the nominal number of nodes in the y direction and half that number in the x direction. Both stretched and normal grid simulations were performed using the multiple relaxation time collision operator. The flow was initialized to the zero time solution of the governing equations (Eqn. 5.6). The simulation was allowed to proceed until the non-dimensional time,

$$t^* = 2k_0^2 \nu t,$$

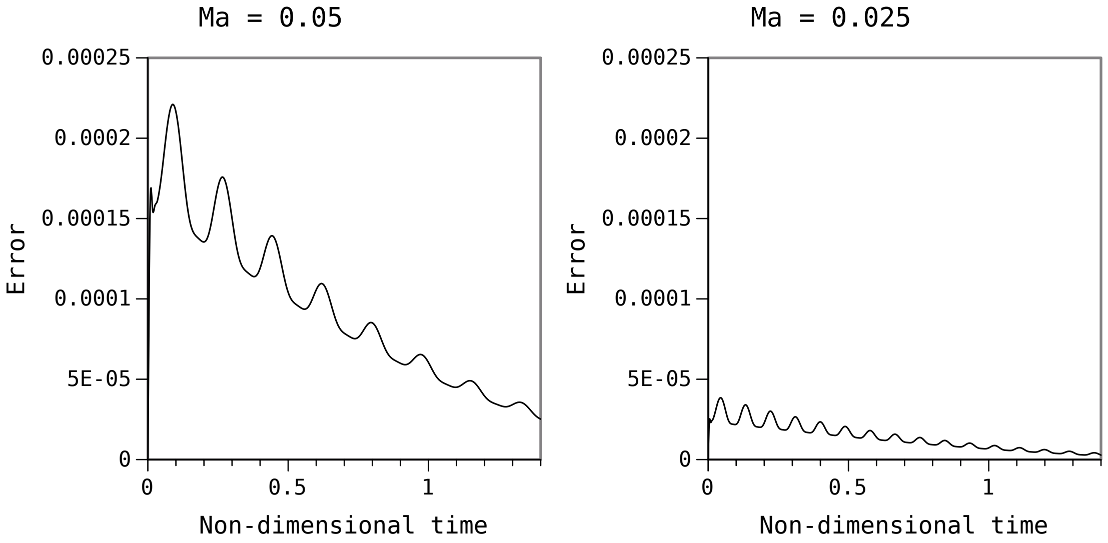


Figure 5.4: Error vs time for Taylor-Green vortex simulation, $Re = 10$ and $Ma = 0.025$ and $Ma = 0.0125$ with 256 nodes.

reached a value of,

$$e^{-t^*} = \frac{1}{4}, \quad \Rightarrow \quad t^* = -\ln\left(\frac{1}{4}\right) \sim 1.3863.$$

An example of the error behaviour with respect to time is shown in Figure 5.4. The oscillations observed in the solution are not consistent with an accurate numerical solution. The period of oscillation in all cases is consistent with an acoustic wave with wavelength equal to half the domain length, $l_0/2$. We suspect that these oscillations are related to initialization since the amplitude decays with time. We have attempted to thermalize the flow in order to initialize the non-equilibrium moments as explained in Section 3.9 however results did not improve noticeably.

The error behaviour as a function of Mach number is shown in Figure 5.5. Different characteristic regions are observed. For small Mach numbers around 0.1, the slope appears to be close to second order. As the Mach number continues to be reduced, a point is reached where the slope rapidly changes. In this region, at Mach numbers of around 0.003, the slope indicates an order of accuracy worse than first order. In particular there are some anomalous data points at the transition between the two

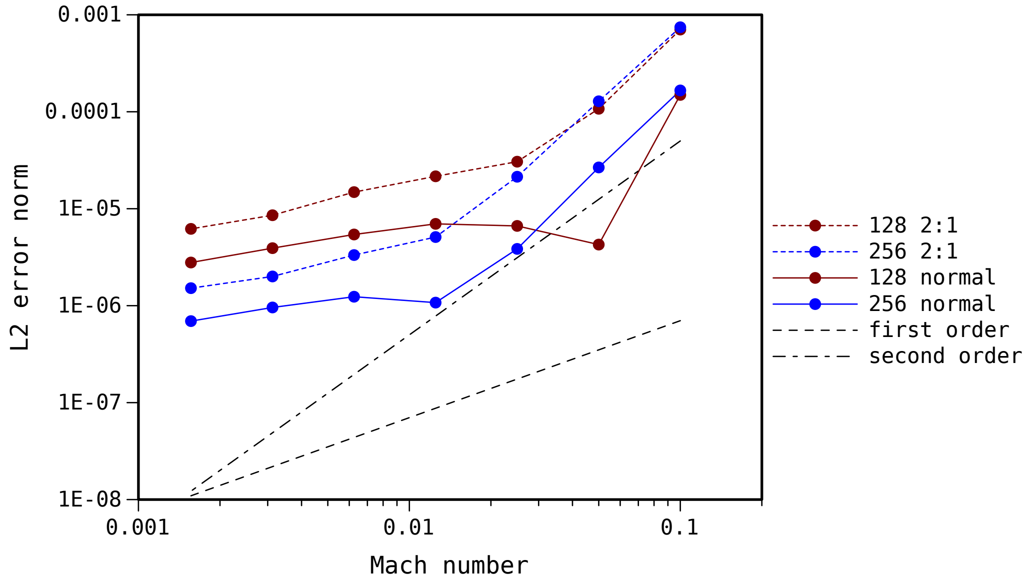


Figure 5.5: Error vs Mach number for Taylor-Green vortex simulation, $Re = 10$, various grid sizes and schemes shown.

regions, for example the 128 normal grid simulation at a Mach number of 0.05 and the 256 node normal grid simulation at a Mach number of 0.0125 exhibit lower lower error than the general trend. We suspect that a new mechanism comes into play at these very small Mach numbers, perhaps due to the larger relaxation factors that must be used in order to maintain a constant Reynolds number as the Mach number is reduced. We observe that the effect is less pronounced in our other simulations at a Reynolds number of one shown in Figure 5.7 where relaxation factors are lower, this is consistent with our hypothesis that the anomaly is caused by a high relaxation factor.

At all grid sizes and Mach numbers the error of our stretched grid scheme are larger than for a uniform grid. Despite these seemingly disappointing results, our scheme is significantly more accurate than the rectangular grid method that has been published by Bouzidi et al. [19]. As we have explained in Section 3.10, our rectangular LB method eliminates errors introduced by the choice of eigenbasis. Figure 5.6 shows a comparison of error behaviour between our scheme and the scheme of Bouzidi et al. using a Mach number of 0.00625, a Reynolds number of

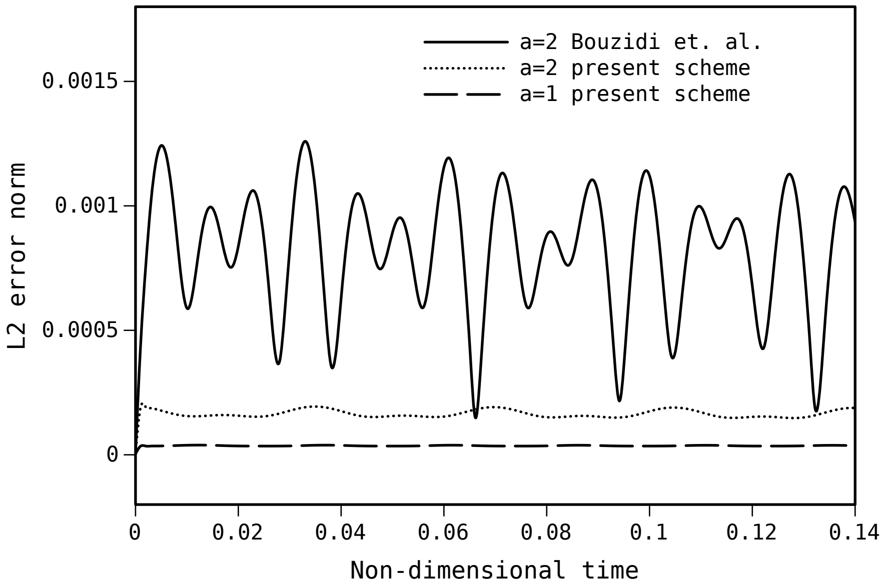


Figure 5.6: Error vs time for Taylor-Green vortex simulation, $Re = 10$ and $Ma = 0.00625$ with 256 nodes.

10 and a grid size of 256 × 128 for the rectangular grids and 256^2 for the uniform grid. Our scheme is an order of magnitude more accurate.

We have also performed additional experiments in order to further test the accuracy of the difference schemes we have devised which were introduced in Section 3.4.2. The error results were taken at the same non-dimensional time, $t^* = \ln(0.25)$ however a Reynolds number of one was used in this case.

Results using a grid size of 128 are shown in Figure 5.7. Again, the slope indicates second order accuracy for Mach numbers near 0.1 with the order of accuracy decreasing as the Mach number is reduced further. In this case the change in order of accuracy is not abrupt. The previously mentioned anomalous behaviour is not apparent until the very smallest Mach number in this case. The most accurate results are generally obtained using Lagrangian streaming, in other words a Courant number of unity. Our modified Lax-Wendroff (LW) scheme with a reduced Courant number only shows improved error at the smallest Mach number. This is in contrast to the laminar channel flow experiment which showed the best accuracy using our modified LW scheme. Figure 5.8 demonstrates that these simulations did not

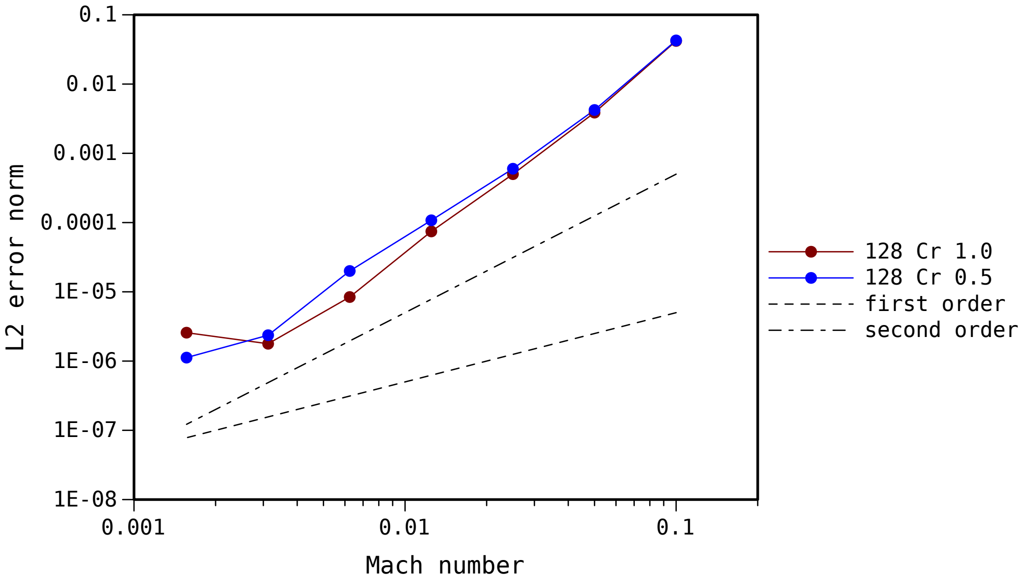


Figure 5.7: Error vs Mach number for Taylor-Green vortex simulation, $Re = 1$, 128 nodes and various schemes shown.

show the same oscillations as the aforementioned tests at a Reynolds number of 10. More investigation is required in order to fully understand the error behaviour of LB methods when used to simulate Taylor-Green vortex flows.

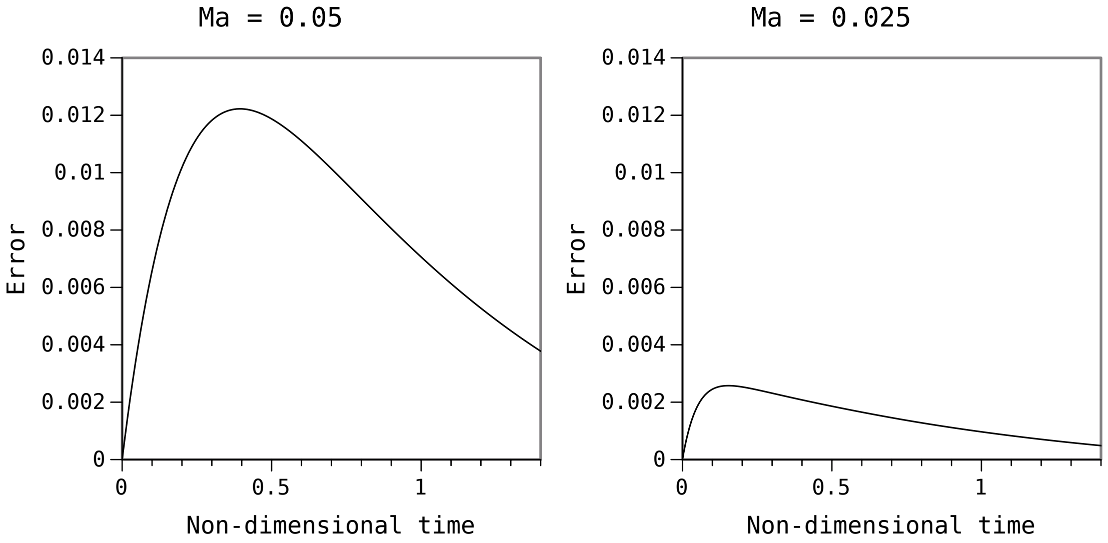


Figure 5.8: Error vs time for Taylor-Green vortex simulation, $Re = 1$ and $Ma = 0.025$ and $Ma = 0.0125$ with 128 nodes.

5.5 Initial condition perturbation

The velocity perturbation field u_α^{pert} is formed by taking the curl of a three dimensional random noise field n_γ,

$$u_\alpha^{\text{pert}} = \varepsilon_{\alpha\beta\gamma}\partial_\beta n_\gamma.$$

Using the identity,

$$\partial_\alpha \varepsilon_{\alpha\beta\gamma}\partial_\beta a_\gamma = 0, \quad \forall a_\gamma.$$

The divergence of the velocity perturbation field is thus zero and it is suitable for the initialization of incompressible flow.

The random noise field is smoothed by applying the diffusion equation. This is equivalent to a low pass filter.

$$\frac{\partial n}{\partial t} = \alpha \frac{\partial^2 n}{\partial x^2}$$

The diffusion equation is solved using a simple finite difference method operating

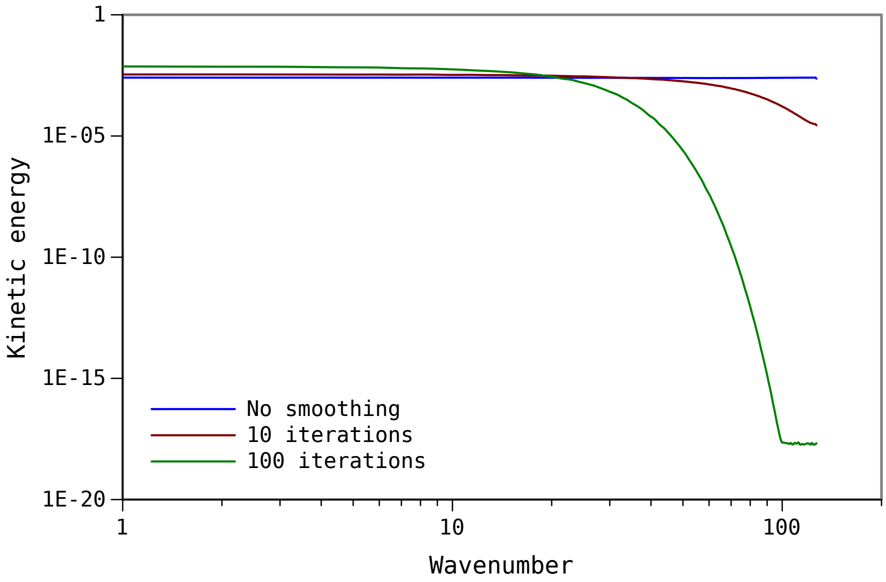


Figure 5.9: Velocity perturbation kinetic energy spectrum, various amounts of smoothing.

just under the Von-Neumann stability limit [37],

$$\alpha \Delta t \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2} \right) < \frac{1}{2}.$$

The diffusive length and time scales are related by,

$$t^0 = \frac{(x^0)^2}{\alpha}. \tag{5.7}$$

These can be used to adjust the cutoff frequency.

As shown in Figure 5.9, the random noise field initially has a roughly uniform distribution of energy vs wavenumber. Although uniformly distributed "white" noise can be used directly [226, 224], we have found that having energy present at the smallest scales is undesirable. Some numerical schemes, including LB methods, generate oscillations due to dispersive errors which may grow with time causing instability when the shortest length scales are unable to be sufficiently resolved. As shown in Figure 5.9, 100 iterations is sufficient to ensure that no energy (within the limits of numerical truncation) remains at the Nyquist frequency. The number of iterations of diffusion can be tuned to give the desired cutoff frequency using Eqn. 5.7. Visualizations of the resulting velocity field are shown in Figure 5.10.

Some applications, such as the simulation of homogeneous isotropic turbulence require the energy spectrum of the initial velocity to better approximate turbulence [201, 264]. It is common to use the Passot-Pouquet [169] spectrum given by,

$$E(k) = \frac{16}{\sqrt{\frac{\pi}{2}}} \frac{u_0^2 k^4}{k_0^5} \exp\left(-\frac{2k^2}{k_0^2}\right).$$

Phases are uniformly randomised and a reverse Fourier transform is used in order to generate a velocity field from the spectral data.

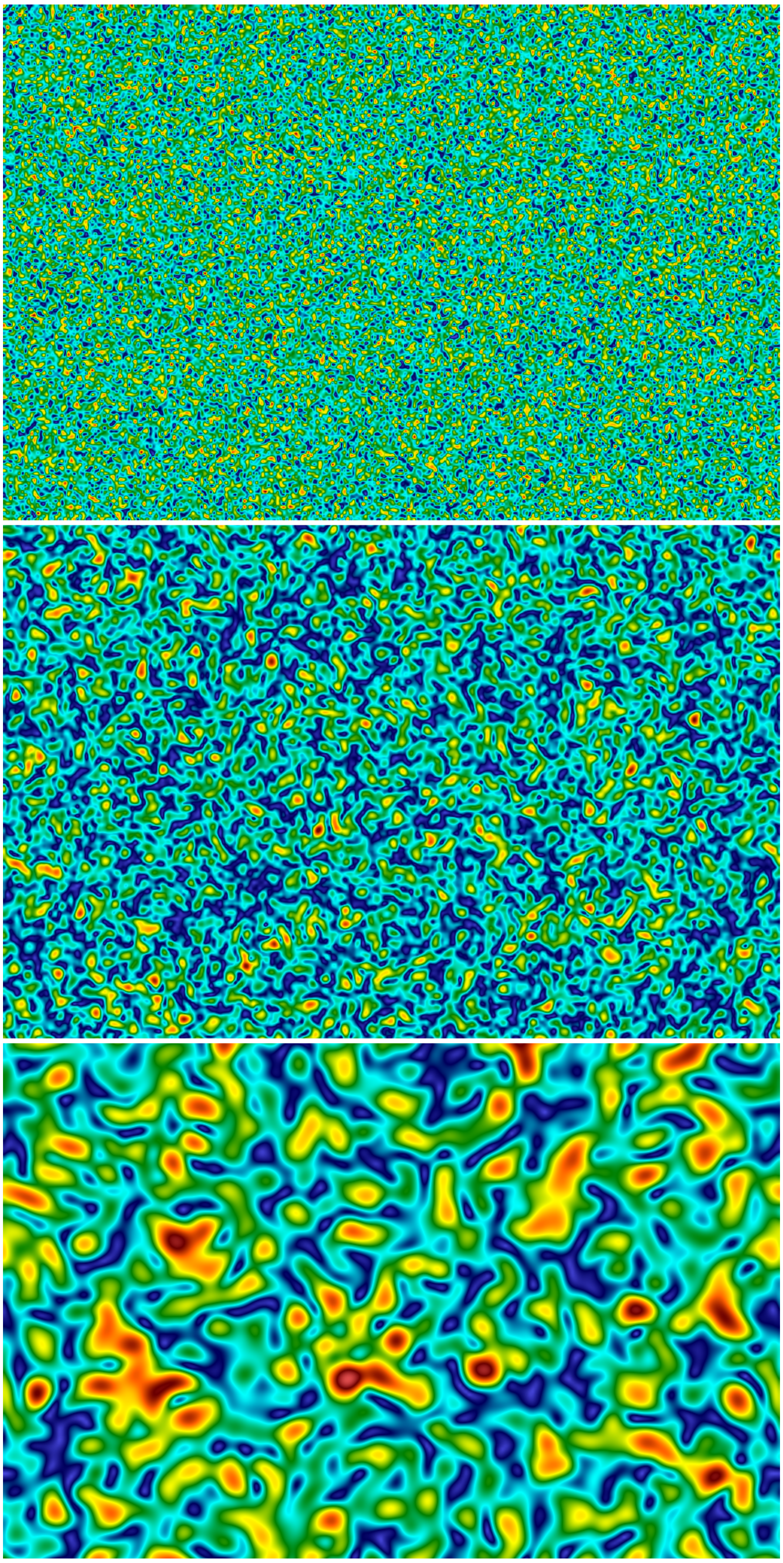


Figure 5.10: Pseudocolor plots of velocity perturbation field, no smoothing (top), 10 iterations of diffusion (mid) and 100 iterations of diffusion (bottom).

5.6 Gathering statistics

Statistics are saved at adjustable intervals. The following procedure allows conversion of these statistics to equivalent ones over a larger time interval. For example during channel flow simulations, mean and fluctuating velocity profiles are saved at a selected time interval. Saving at each time step would be excessive especially due to the small effective Courant number of LB simulations. Once the simulation has been completed and a suitable window has been selected over which to determine the fully developed statistics, the saved profiles must be converted to mean and fluctuating profiles over this new period.

In the case of mean profiles taken on xz planes, increasing the period of time averaging is trivially simple,

$$\begin{aligned} \overline{u}_{j,t} &= \frac{1}{n_{XZ}} \sum_{i=0}^{nX-1} \sum_{k=0}^{nZ-1} u_{i,j,k,t}, \\ <\overline{u}>_j^T &= \frac{1}{n_T} \sum_{t \in T} \overline{u}_{j,t}. \end{aligned}$$

Consider a disjoint set of n_{T_A} discrete time periods,

$$T_A = \{T_1, T_2, \ldots, T_h, \ldots, T_{n_{T_A}}\}.$$

Then the mean over T_A is simply,

$$<\overline{u}>_j^{T_A} = \frac{1}{n_{T_A}} \sum_{h}^{n_{T_A}} <\overline{u}>_j^{T_h}.$$

The fluctuating velocity profile which varies in the y direction is defined,

$$\hat{u}_{i,j,k,t} = u_{i,j,k,t} - \overline{u}_{j,t}.$$

It averages out to zero over planes,

$$\overline{\hat{u}}_{j,t} = \frac{1}{n_{XZ}} \sum_{i=0}^{nX-1} \sum_{k=0}^{nZ-1} (u_{i,j,k,t} - \overline{u}_{j,t}) = \overline{u}_{j,t} - \overline{u}_{j,t} = 0.$$

Non linear quantities involving fluctuating quantities however are non zero,

$$\overline{\hat{u}\hat{u}}_{j,t} = \frac{1}{n_{XZ}} \sum_{i=0}^{nX-1} \sum_{k=0}^{nZ-1} (u_{i,j,k,t} - \overline{u}_{j,t}) \cdot (u_{i,j,k,t} - \overline{u}_{j,t}) \neq 0.$$

A new symbol u'^T is introduced, the prime indicates fluctuations from the temporal and spatial average,

$$u'^T_{i,j,k,t} = u_{i,j,k,t} - <\overline{u}>^T_j.$$

This is in contrast to the hat which only indicated deviations from the spatial average. The desired mean over some time period of the fluctuations squared where the fluctuations are also based on the mean from the same period is not simply equal to the mean of the instantaneous fluctuations squared.

$$\begin{aligned}
<\overline{u'u'}>^T_j &= \frac{1}{nT} \sum_{t \in T} \frac{1}{n_{XZ}} \sum_{i=0}^{nX-1} \sum_{k=0}^{nZ-1} \big(u_{i,j,k,t} - <\overline{u}>^T_j\big) \cdot \big(u_{i,j,k,t} - <\overline{u}>^T_j\big), \\
<\overline{\hat{u}\hat{u}}>^T_j &= \frac{1}{nT} \sum_{t \in T} \overline{\hat{u}\hat{u}}_{j,t}, \\
<\overline{u'u'}>^T_j &\neq <\overline{\hat{u}\hat{u}}>^T_j.
\end{aligned}$$

The desired mean quantity can be calculated based on the saved instantaneous values using,

$$
\begin{aligned}
<\overline{u'u'}>^T_j &= \frac{1}{nT}\sum_{t\in T}\frac{1}{n_{XZ}}\sum_{i=0}^{nX-1}\sum_{k=0}^{nZ-1}\Big[\big(u_{i,j,k,t}-\overline{u}_{j,t}+\overline{u}_{j,t}-<\overline{u}>^T_j\big) \\
&\quad\cdot\big(u_{i,j,k,t}-\overline{u}_{j,t}+\overline{u}_{j,t}-<\overline{u}>^T_j\big)\Big], \\
&= \frac{1}{nT}\sum_{t\in T}\frac{1}{n_{XZ}}\sum_{i=0}^{nX-1}\sum_{k=0}^{nZ-1}\Big[\big(u_{i,j,k,t}-\overline{u}_{j,t}\big)\cdot\big(u_{i,j,k,t}-\overline{u}_{j,t}\big) \\
&\quad+2\big(u_{i,j,k,t}-\overline{u}_{j,t}\big)\cdot\big(\overline{u}_{j,t}-<\overline{u}>^T_j\big)+\big(\overline{u}_{j,t}-<\overline{u}>^T_j\big)^2\Big].
\end{aligned}
$$

Using,

$$\sum_{i=0}^{nX-1}\sum_{k=0}^{nZ-1}(u_{i,j,k,t}-\overline{u}_{j,t})=0,$$

the relationship can be simplified,

$$<\overline{u'u'}>^T_j=\frac{1}{nT}\sum_{t\in T}\Big[\overline{\hat{u}\hat{u}}_{j,t}+\big(\overline{u}_{j,t}-<\overline{u}>^T_j\big)^2\Big].$$

The second term takes into account deviations of the spatial mean from the combined spatial and temporal mean. The same technique can be applied to a set of time averages,

$$<\overline{u'u'}>^{T_A}_j=\frac{1}{n_{T_A}}\sum_{h}^{n_{T_A}}\Big[<\overline{u'u'}>^{T_h}_j+\big(<\overline{u}>^{T_h}_j-<\overline{u}>^{T_A}_j\big)^2\Big].$$

This procedure can be generalised to any product of two properties and any ensemble average over a collection of other averages.

5.7 Turbulent channel flow

We have performed direct numerical simulations of turbulent channel flow using a variety of configurations in order to further benchmark our code. The domain is a rectangular prism with periodic boundaries in the streamwise x and spanwise

z directions and solid wall boundaries in the wall-normal y direction. The half height of the channel δ is equal to one. Results are obtained for streamwise domain lengths of 2π and 4π and spanwise widths of π and $4/3\pi$ and compared to the spectral results of Kim, Moin, Moser and Mansour [124, 160].

Details of the domain size and lattice resolutions used here, and those of Moser et al., are given in Table 5.2. The first column of the table lists code-names for each scheme tested, these codes summarise configuration information. The first number is equal to the number of nodes in the y direction followed by either an `n` suffix indicating a normal cubic grid or an `s` suffix indicating a stretched grid with a reduced grid size in the y direction. Our stretched grid simulation uses the multiple relaxation time method we have developed in Section 3.10. Stretched and normal grid simulations that use the `cub_d3q27_abc` lattice as indicated by the `q27` code use our multiple relaxation time collision operator. Simulations using the `cub_d3q19_ab` lattice are assigned the `q19` code and all use a single relaxation time collision operator. Two different domain sizes are used, the streamwise domain length is used as the next code, either `4pi` or `2pi`. Our implementation requires the use of a uniform grid size, even when using a stretched grid resulting in a coarser mesh at the wall compared to the simulations of Moser et al. which use a nonuniform grid. The distance to the node closest to the wall is given in terms of wall units,

$$y^+ = \frac{u_\tau y}{\nu}.$$

The uniform grids result in a slightly higher wall normal resolution in the centre of the channel and a much higher resolution in the streamwise and spanwise directions. The last code indicates the boundary condition used, the Guo boundary condition (`gb`) places the first node at a distance of one from the wall while the bounce back boundary condition (`bb`) places the first node at a distance of half from the wall hence resulting in a reduced y^+_{wall} value for a given grid size.

Codename	$\frac{\Delta x}{\Delta y}$	N_y	N_x	N_z	l_y	l_x	l_z	N_{tot}	y^+_{wall}
`092n-q19-2pi-gb`	1	92	288	176	2	2π	π	4.6×10^6	3.82
`092n-q19-4pi-gb`	1	92	576	198	2	4π	$4/3\pi$	10.4×10^6	3.82
`112n-q19-2pi-gb`	1	112	352	176	2	2π	π	6.9×10^6	3.16
`112n-q19-4pi-gb`	1	112	704	234	2	4π	$4/3\pi$	18.0×10^6	3.16
`132n-q19-2pi-gb`	1	132	416	212	2	2π	π	11.6×10^6	2.69
`142n-q27-4pi-bb`	1	142	896	296	2	4π	$4/3\pi$	37.7×10^6	1.27
`152n-q19-2pi-gb`	1	152	480	246	2	2π	π	17.8×10^6	2.35
`160n-q27-4pi-bb`	1	160	992	333	2	4π	$4/3\pi$	52.9×10^6	1.13
`178s-q27-4pi-bb`	1.4	178	800	266	2	4π	$4/3\pi$	37.9×10^6	1.01
`Moser et al.`	–	129	128	128	2	4π	$4/3\pi$	2.1×10^6	0.05

Table 5.2: Configurations tested. All simulations use $\mathrm{Re}_\tau = 180$.

5.7.1 Parameters

The flow is defined by the Reynolds number Re_τ based on the friction velocity u_τ. All simulations are performed at $\mathrm{Re}_\tau = 180$. The wall shear τ_w is related to the applied body force f which drives the flow, by a static balance of forces as,

$$f\delta = \tau_w = \mu \left. \frac{d\bar{u}}{dy} \right|_{y=0} .$$

Here $\bar{u}$ is the average streamwise velocity over xz planes. The friction velocity u_τ and corresponding Reynolds number Re_τ are defined in terms of f as,

$$u_\tau = \sqrt{\frac{\tau_w}{\rho}} = \sqrt{\frac{f\delta}{\rho}},$$
$$\mathrm{Re}_\tau = \frac{u_\tau \delta}{\nu} = \frac{\delta}{\nu}\sqrt{\frac{f\delta}{\rho}}.$$

Lattice spacing and time step are both fixed at one in lattice units. The size of the domain δ is determined by the resolution, while the viscosity and the body force remain variable.

The Reynolds number is specified, with the added constraint that the Mach number should be below some threshold in order to decrease the compressibility error. The

Mach number plays a similar role to the CFL number,

$$\text{CFL} = \frac{\Delta t \cdot u_{\max}}{\Delta x}.$$

With Δt and Δx both equal to one in lattice units, the relationship between CFL number and Mach number is,

$$\begin{aligned}
\text{CFL} &= u_{\max}, \\
c_s &= \frac{1}{\sqrt{3}}, \\
\text{Ma} &= \frac{u_{\max}}{c_s} = \frac{\text{CFL}}{c_s} = \sqrt{3} \times \text{CFL}.
\end{aligned}$$

We use a Mach number of 0.1, equivalent to a CFL number of approximately 0.05774 for all simulations.

5.7.2 Results

The friction velocity turnover time is defined as,

$$t^* = \frac{\delta}{u_\tau}.$$

As shown in Figure 5.11, parameters such as wall shear and peak turbulent kinetic energy (TKE) reach steady values after approximately 30 turnover times. This development period depends on the initial condition used. We have used our smoothed, divergence free perturbation from Section 5.5 and a 1/7 power law velocity profile. In order to be certain that fully developed conditions have been reached, we allow the simulation to progress for 200 turnover times and then sample statistics for a further 200 turnover times. This is a drastically longer sampling period than the 10 turnover times used for the entire original simulation of Kim et al. [124]. Although the sampling period is not specified by Moser et al. [160], they do mention that the sampling period is marginal and the statistics contain wiggles. This highlights the advantage of our code, extremely long running simulations are made feasible.

Streamwise TKE spectra along the centerline and at a location near the TKE peak are shown in Figure 5.12. The wavenumber in these plots is defined in terms of wavelength λ as,

$$k = \frac{2\pi}{\lambda}.$$

The spectra of Moser et al. and of our simulations are similar overall, our simulations use a significantly higher streamwise resolution resulting in profiles that extend further into the dissipation range. The spectra at $y^+ = 19$ show closer agreement than the centerline spectra. The gradient of the 5/3 power law is also shown for reference. One possible cause for the differences in spectra may be due to compressibility effects. It is possible that the Mach number of 0.1 which was used was not sufficiently small to eliminate all compressibility effects which would be more pronounced in regions of high mean velocity near the centerline. This observation is consistent with the larger deviation of our spectra from those of Moser et. al. at the centerline.

Figure 5.13 shows mean velocity and TKE profiles in wall units for a large number of configurations. Overall, results are quantitatively similar to those of Moser et al. and approach them as the resolution is increased. A log-law region is apparent in the mean velocity profile for values of y^+ greater than approximately 30. We have not found significant difference between the small and large domains (`2pi` and `4pi` codes) when all other factors are accounted for. Both domain sizes are shown for the `112n-q192-xxx-gb` simulations and the results are virtually on top of each other. There is a significant difference between results obtained using the Guo and bounce back boundary conditions (`gb` and `bb` codes). We attribute the superior accuracy of the bounce back boundary condition simulations to the placement of a node at half grid spacing from the wall. It seems that the height of the TKE peak is closely related to the placement of the first fluid node. The stretched grid simulation achieves accurate results however it is not clear if there is any improvement compared to a traditional square grid. Both the `142n-q27-4pi-bb` and

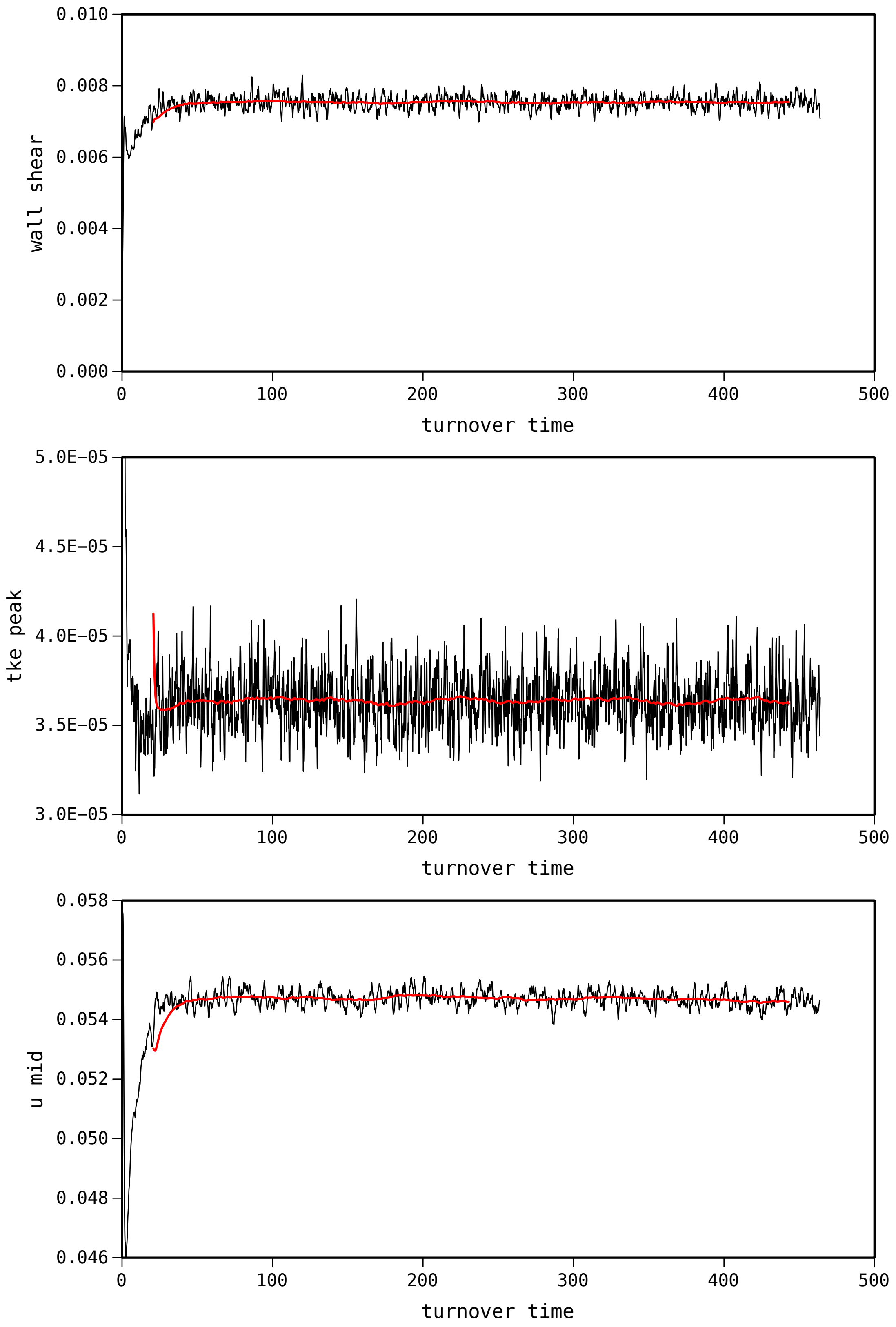


Figure 5.11: Time variation of wall shear, peak TKE and centerline velocity for `142n-q27-4pi-bb` simulation.

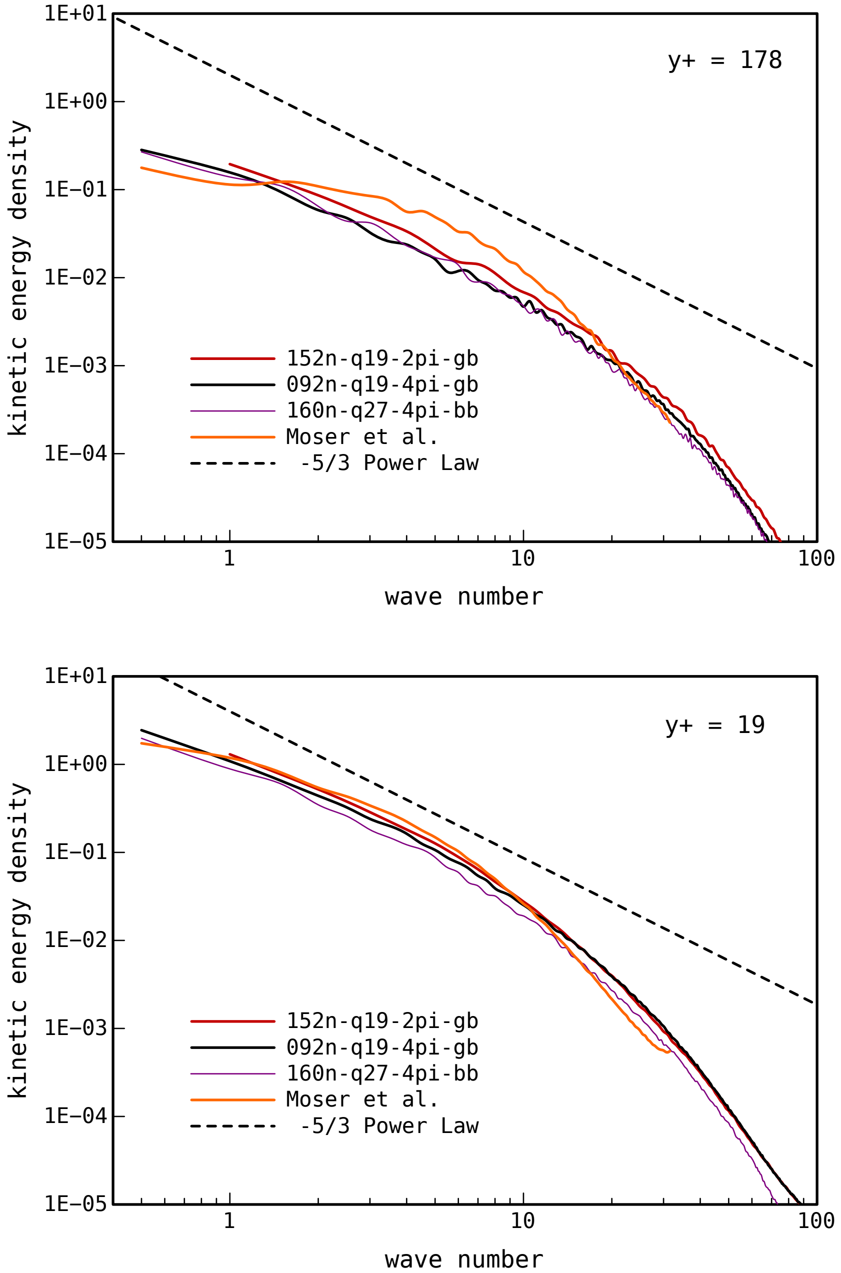


Figure 5.12: Streamwise turbulent kinetic energy spectra along centerline ($y^+ = 178$) and near TKE peak ($y^+ = 19$) for various simulations.

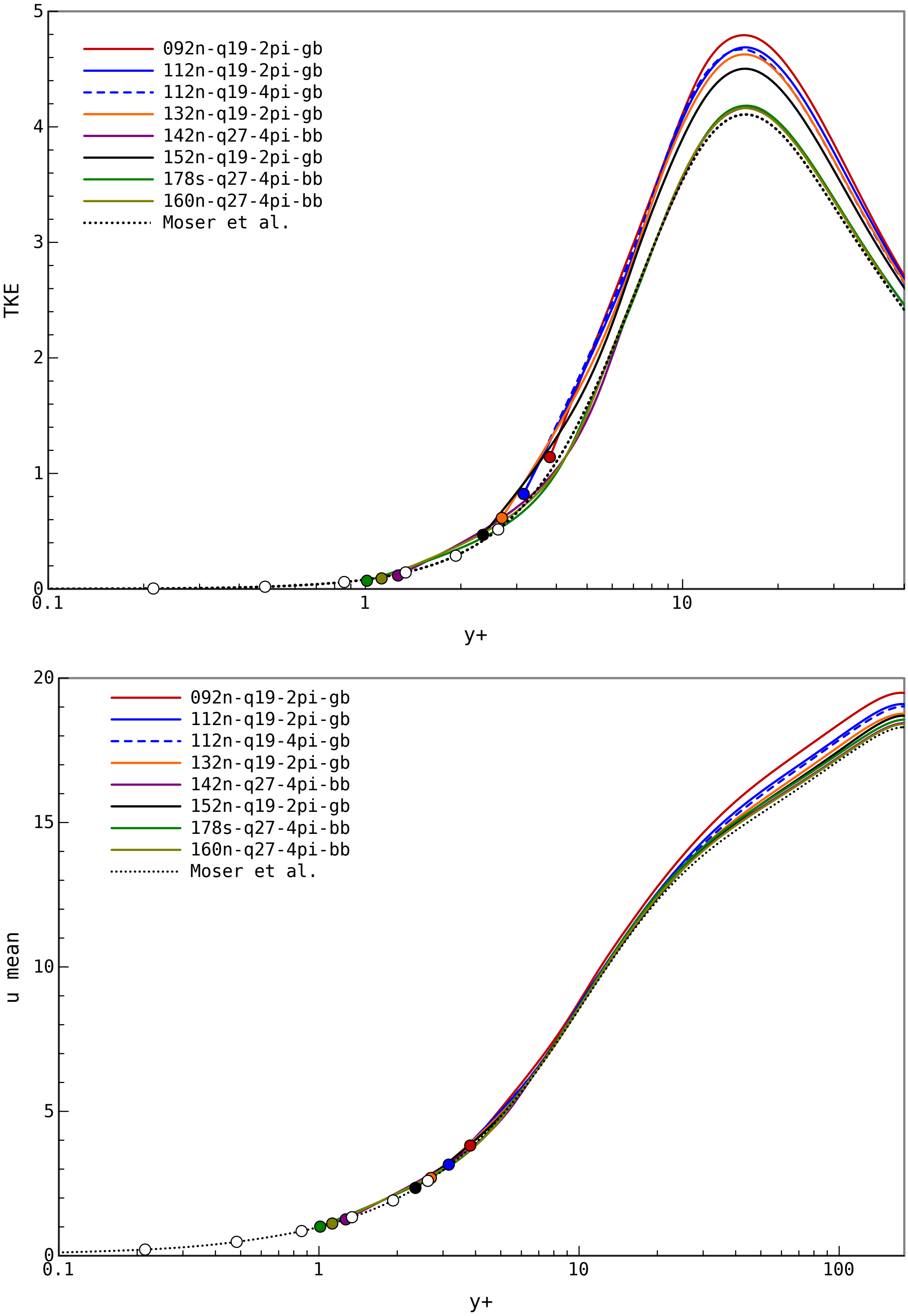


Figure 5.13: Mean velocity and TKE profiles for various configurations.

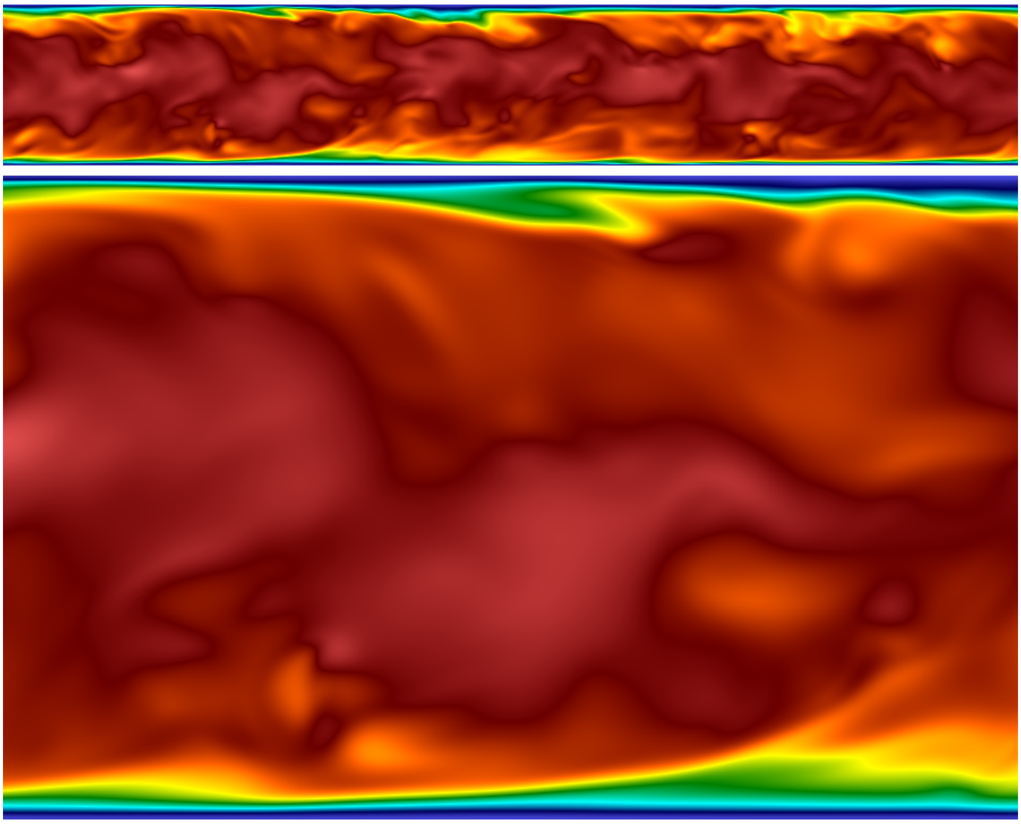


Figure 5.14: Pseudocolor velocity field of `142n-q27-4pi-bb` simulation at $t^* = 277$.

`178s-q27-4pi-bb` simulations use the same number of cells however the latter has a finer resolution in the wall normal direction due to the stretched grid. Both of these simulations achieve similar results. This simulation demonstrates the stability and accuracy of the stretched grid scheme I have devised in Section 3.10. This is in contrast to other rectangular lattice methods which have not demonstrated turbulent flow simulations. A visualisation of the velocity field is shown in Figure 5.14.

5.8 Sheared convective boundary layer

The sheared convective boundary layer is an environmentally occurring fluid flow consisting of a convective boundary layer topped by a stably stratified shear flow. The velocity profile is characterised by a series of layers moving away from the surface. The layer closest to the surface is the boundary layer followed by a mixed

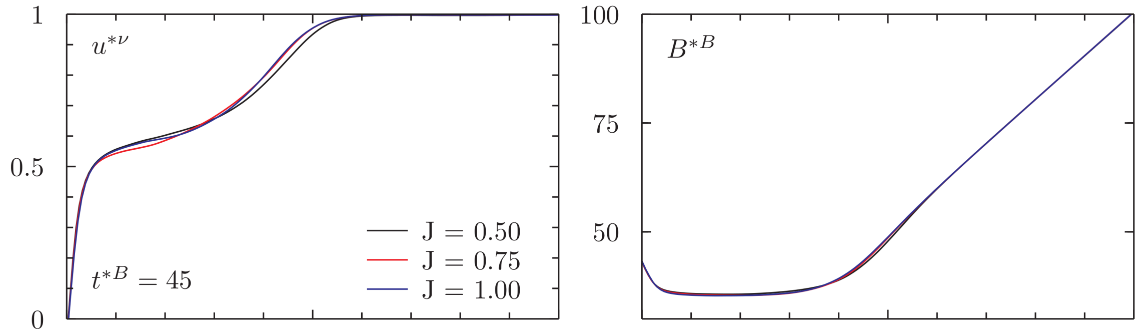


Figure 5.15: Various example velocity and buoyancy profiles for the sheared convective boundary layer.

layer then a shear layer and finally the free stream which has a uniform velocity profile. The buoyancy field is characterised by surface convection in the boundary layer followed by a fairly uniform region in the mixed layer and then a stratification beginning at the velocity shear layer and continuing into the free stream. Characteristic velocity and buoyancy fields from our simulations are shown in Figure 5.15 with further details to be presented subsequently. Research efforts have concentrated on parameterizing the rate of entrainment for use in larger scale simulations or other environmental and engineering models. There are many possible variations to this flow. In the case of zero mean velocity, this is referred to as the convective boundary layer (CBL). There has been considerably more progress characterizing the CBL. Research has progressed along three fronts, field observations of actual weather events, large eddy simulations and experiments using wind tunnels and water channels.

We consider flows driven by a geostrophic wind. Such flows can be approximated as a flow with a constant velocity leading to the formation of a boundary layer. All of the numerical studies mentioned assume that there is a pre existing inversion in the potential temperature. Some studies assume that the lower fluid has a uniform density, followed by a step jump at the inversion and then a linear stable stratification in the stable atmosphere [72, 73, 117, 74, 75, 76, 182, 125, 228, 181]. Others assume a linear stable stratification throughout the domain [56, 57, 183, 58, 236]. The density profile used by Pino et al. [182] was based on field observations on a particular day and later work [181] made use of the same data. Other studies cited do not justify their choice of density profile. The characteristics of the inversion strongly influence the behaviour of the system.

The formation of an inversion may occur due to numerous mechanisms including night time surface cooling, moving warm fronts or as a result of atmospheric subsidence – that is a negative mean vertical velocity. Inversions due to surface cooling often occur during the night when the surface loses heat due to radiation and hence cools the boundary layer forming a stable stratification. Winds will lead to mixing however turbulent transport will be inhibited by the stratification. This flow, commonly referred to as the stable boundary layer (SBL) has been extensively studied and has been shown to lead to an almost linear potential temperature profile in a large section of the boundary layer between 30% and 100% of boundary layer height. There is a sharper gradient near the surface [77]. Churilov has published several papers [48, 49, 50, 51, 52] analysing the stability of stratified shear layers with an inflection free velocity profile. These flows have an interesting property that they are stable in the absence of stratification.

Once developed, the SCBL may be thought of as having two distinct components, the convective boundary layer at the ground, a mixed layer and a stratified shear layer. There may exist some limit, where the separation between these regions is large enough that they can be treated separately. For example, it may be possible to approximate the mixing in the shear layer while neglecting the dynamics below it in the mixed layer.

The convective boundary layer will grow in thickness along the spanwise direction beginning as a laminar Blasius profile and then undergoing a transition to turbulence and continuing to grow with an ever increasing Reynolds number. The shear layer will also grow along the streamwise direction. Depending upon the Richardson number and Reynolds number, the shear layer may thicken in a laminar fashion or mixing may be accelerated by shear instabilities. For a shear layer with sharp changes in velocity and buoyancy, the stability behaviour is dependent on the Richardson number (J). Low values of J, will lead to a Kelvin-Helmholtz (KH) type instability while higher values may either be stable or subject to a Holmboe (H) type instability [226]. Either unstable mode will lead to increased mixing and potentially turbulence production. Although most research has concentrated on Richardson numbers below 5, it has also been shown that a secondary Holmboe mode is present at J as high as 20 [3].

Stratified shear flows have typically been simulated in a periodic box large enough to fit one or more wavelengths of the dominant unstable mode [254, 225, 175, 29, 226, 21, 224, 3, 28, 55]. When boundary layers are simulated in a periodic domain, it is usually referred to either as a "temporal" simulation or "parallel flow" [230]. These simulations cannot capture any streamwise spatial variation and hence do not capture effects such as asymmetric entrainment of plane mixing layers [151].

In contrast "spatial" boundary layer simulations use some prescribed inflow condition followed optionally by an area designated for flow development and then a region of interest. These simulations can be broadly grouped into simulations that aim to capture the transition to turbulence and those that aim to generate turbulence as an inflow. Simulations using a laminar inflow require sufficient spatial extent for turbulence to develop and thus are limited to lower Reynolds numbers. The transition to turbulence is typically accelerated by introducing a perturbation ranging from synthetic turbulence [193, 115] to more accurate techniques including passing wakes [257] or periodic patches of isotropic turbulence [256]. The last two techniques are somewhat similar to the use of trip wires in wind tunnel experiments.

The method of Spalart [229] transforms a rectangular domain using a change of variable in order to allow stream-wise gradients to develop in a periodic box. Using this method Spalart produced DNS results that have been widely used as reference data ever since. Although some early studies have reported success using parallel flow [127], most contemporary works focus on spatial simulations [208, 139].

Lund, Wu and Squires [147] devised a method of turbulent inflow generation where a simplified form of Spalart's transformation is used to run a temporal simulation, the purpose of which is to provide input data for a downstream spatial simulation. The spatial simulation is performed using a domain 2.4 times the stream-wise length of the periodic simulation. Lund et al. achieved results that agree quite well with those using the Spalart method and superior to results obtained using a purely periodic 'parallel' upstream section. They note that a parallel development may be used given a sufficient development zone of around 10 boundary layer thicknesses.

5.8.1 Parameters

The simulation is fully specified by a combination of fluid properties, boundary conditions and initial conditions. The relevant fluid properties are viscosity ν, thermal diffusivity α and density ρ_0. The boundary conditions are a no-slip wall with prescribed buoyancy flux $\vec{B}_{surf}$ at the ground and an infinite domain extending vertically. The infinite domain is approximated by a free-slip boundary with an inflection free buoyancy profile used to allow a constant buoyancy gradient in the vertical direction. The initial conditions are a constant velocity u_0 and a constant buoyancy gradient $\frac{dB}{dz}$. These parameters are summarized in Table 5.3 and visualised in Figure 5.16. The streamwise, spanwise and surface-normal directions are the x, y and z coordinate axes.

The free stream velocity and buoyancy stratification may also be viewed as boundary conditions and the initial condition being that the boundary layer height is zero. In some cases, it may be desirable to begin the simulation with some pre-existing velocity and buoyancy profiles, in this case these profiles will introduce additional parameters.

We perform a temporal simulation with periodic boundary conditions in the spanwise and stream-wise directions. The domain size used is significantly larger than the largest expected flow feature.

The six parameters specifying the fluid properties and boundary conditions are

Parameter	Unit	Description
ν	$\mathrm{m^2s^{-1}}$	viscosity
α	$\mathrm{m^2s^{-1}}$	thermal diffusivity
ρ_0	$\mathrm{kg\,m^{-3}}$	density
$\vec{B}_{surf}$	$\mathrm{kg\,m^{-1}s^{-3}}$	buoyancy flux at the surface
u_0	$\mathrm{m\,s^{-1}}$	free stream velocity
$\frac{dB}{dz}$	$\mathrm{kg\,m^{-3}s^{-2}}$	buoyancy stratification

Table 5.3: Parameters used in the DNS simulation

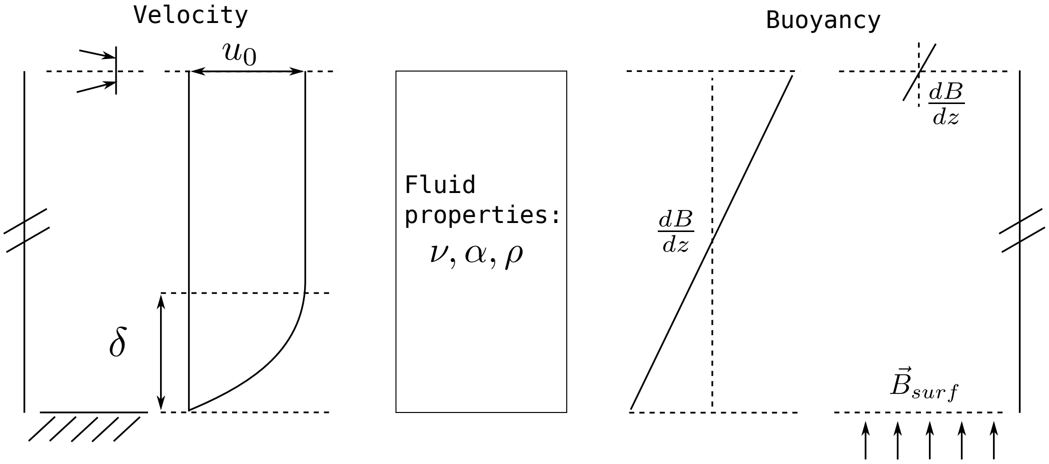


Figure 5.16: Parameters used in the DNS simulation

made up of three fundamental properties so using the Buckingham Π theorem, the problem may be equivalently specified using three non-dimensional parameters. The repeating variables are chosen to be ν, u_0 and $\vec{B}_{surf}$ and the remaining variables are expressed in terms of these.

$$[\alpha] = [\nu] \quad \Rightarrow \quad \Pi_1 = \mathrm{Pr} = \frac{\nu}{\alpha}.$$

The Prandtl number Pr relates momentum and buoyancy diffusivity. Length and time are expressed as,

$$\mathrm{m} = \frac{[\nu]}{[u_0]} \quad \text{and} \quad \mathrm{s} = \frac{[\nu]}{[u_0^2]},$$

in order to express the remaining dimensionless groups,

$$\begin{aligned}
[\rho_0] = [\vec{B}_{surf}]\frac{[u_0^2]}{[\nu^2]}\cdot\frac{[\nu^3]}{[u_0^6]} \quad &\Rightarrow \quad \Pi_2 = \frac{\vec{B}_{surf}\nu}{u_0^4\rho_0},\\
[\frac{dB}{dz}] = [\vec{B}_{surf}]\frac{[u_0^2]}{[\nu^2]}\cdot\frac{[\nu]}{[u_0^2]} \quad &\Rightarrow \quad \Pi_3 = \frac{\vec{B}_{surf}}{\nu\frac{dB}{dz}}.
\end{aligned}$$

The third group Π_3 and first group Pr are combined to give what we have called the buoyancy shape Ш,

$$Ш = \Pi_3 \cdot \Pi_1 = \frac{\vec{B}_{surf}}{\alpha\frac{dB}{dz}}.$$

The buoyancy shape is represented by the Cyrillic letter Sha (for shape), we have found the shape factor to be the most important influence on the characteristic shape of the flow field. For $Ш = -1$ the buoyancy field will maintain a linear gradient. Increasing Ш will result in a larger gradient at the wall. For the laminar case, the hydrodynamic fields and the buoyancy field are decoupled. The buoyancy field can be characterized using the following dimensionless parameters,

$$\begin{aligned} t^B &= \sqrt{\rho_0 \left(\frac{dB}{dz}\right)^{-1}}, & t^{*B} &= \frac{t}{t^B}, \\ y^B &= \sqrt{\alpha t^B}, & y^{*B} &= \frac{y}{y^B}, \\ B^B &= \frac{dB}{dz} y^B, & B^{*B} &= \frac{B}{B^B}. \end{aligned}$$

Normalizing by these characteristic scales allows all laminar profiles without convection to collapse to one solution specified by Ш and time as shown in Fig. 5.17. The velocity profile will scale according to the following viscous scales,

$$\begin{aligned} u^\nu &= u_0, & u^{*\nu} &= \frac{u}{u^\nu}, \\ t^\nu &= \frac{\nu}{u_0^2}, & t^{*\nu} &= \frac{t}{t^\nu}, \\ y^\nu &= \frac{\nu}{u_0}, & y^{*\nu} &= \frac{y}{y^\nu}. \end{aligned}$$

The laminar solution is not dependent on the second non-dimensional group, only the shape of the buoyancy field depends on Ш. Due to the periodic boundary condition, the velocity profile is not a Blasius boundary layer but a more simple solution to momentum diffusion only.

The Reynolds number for boundary layer flows may be based on a number of length scales including the 99% length δ, the displacement thickness,

$$\delta^* = \int_0^\infty \left(1 - \frac{u}{u_0}\right) dy,$$

or the momentum thickness,

$$\theta = \int_0^\infty \frac{u}{u_0} \left(1 - \frac{u}{u_0}\right) dy.$$

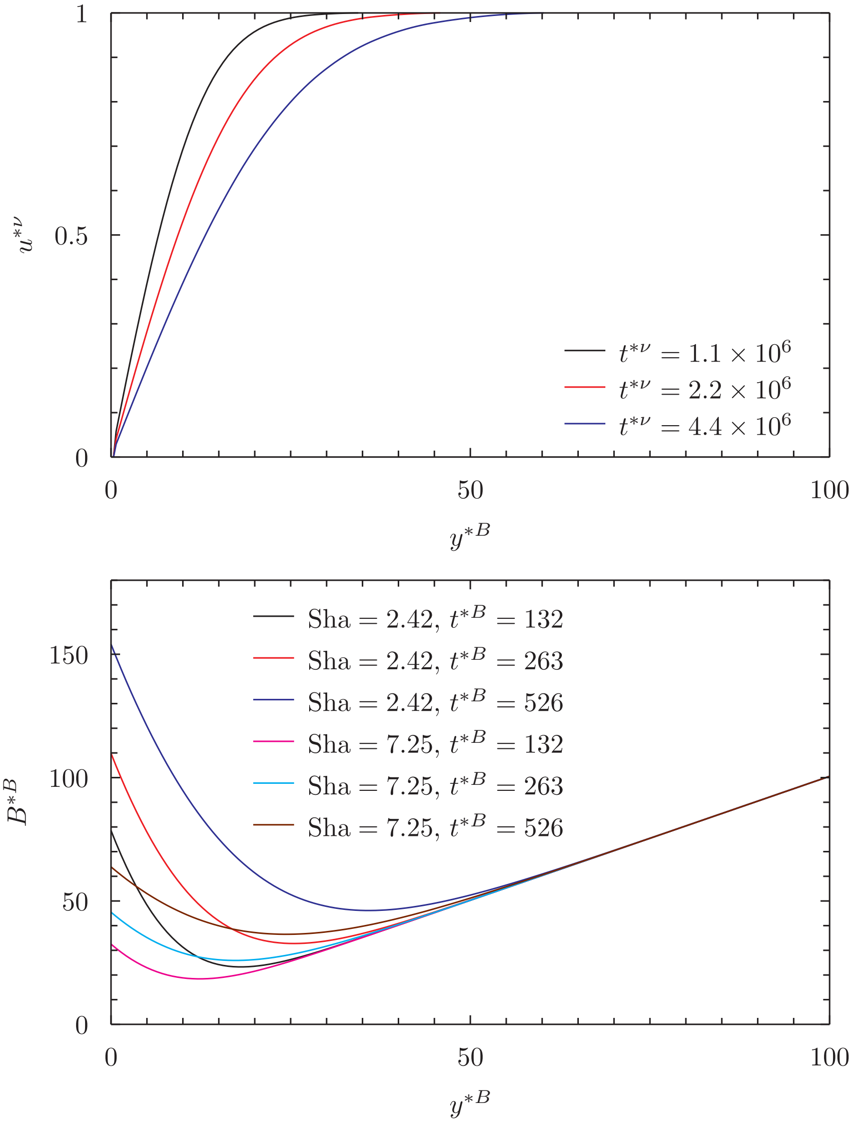


Figure 5.17: Velocity and buoyancy at various times for laminar flow.

The resulting Reynolds numbers are,

$$Re_\delta = \frac{\delta\, u_0}{\nu} \quad \text{or} \quad Re_{\delta^*} = \frac{\delta^*\, u_0}{\nu} \quad \text{or} \quad Re_\theta = \frac{\theta\, u_0}{\nu}.$$

All of these boundary layer length scales will increase from their initial value as the simulation progresses changing with them their respective Reynolds numbers and other dependent dimensionless numbers. We express the second group Π_2 as a combination of other parameters in order to yield some insight into its significance,

$$\begin{aligned} \frac{\Pi_2}{\Pi_3} &= \frac{\frac{dB}{dz}\nu^2}{u_0^4 \rho_0} = \frac{\frac{dB}{dz}\delta^2}{u_0^2 \rho_0 \,\mathrm{Re}_\delta^2} = \frac{J_\delta}{\mathrm{Re}_\delta^2}, \\ \Pi_2 &= \frac{\text{Ш} J_\delta}{\mathrm{Pr}\,\mathrm{Re}_\delta^2}. \end{aligned} \tag{5.8}$$

Here J is some bulk Richardson number relating the bulk stabilising energy across a shear layer to the generally destabilising kinetic energy.

$$J = \frac{\Delta B_\delta\, \delta}{\rho_0\, u_0^2}.$$

ΔB_δ is the difference in specific buoyancy across the boundary layer with characteristic length δ. Given a uniform initial stratification, this can be simplified to,

$$\Delta B_\delta = \frac{dB}{dz} \cdot \delta.$$

The stability ratio (SR) uses the Monin-Obukhov length (L_{MO}) to describe the relative contributions of shear and buoyancy effects in the boundary layer,

$$\mathrm{SR} = \frac{\delta}{L_{\mathrm{MO}}} \quad \text{with} \quad L_{\mathrm{MO}} = \frac{u_\tau^3\, \rho_0}{\kappa\, \vec{B}_{surf}}$$

The von Kármán constant κ is roughly 0.41. The wall shear will change over time, influenced mainly by the boundary layer height but also by the turbulent structures that ordinarily depend on Reynolds number but will also depend on buoyancy parameters.

In order to predict the wall shear, additional information regarding the velocity profile is required. The law of the wake has been shown to accurately predict mean velocity profiles for a range of wall bounded shear flows [185, 54],

$$\frac{u}{u_\tau} = f\left(\frac{yu_\tau}{\nu}\right) + \frac{\Pi}{\kappa} w\left(\frac{y}{\delta}\right).$$

Evaluating at $y = \delta$ and approximating f by the log law results in,

$$\frac{u_0}{u_\tau} = \frac{1}{\kappa} \ln\left(\frac{\delta u_\tau}{\nu}\right) + c + \frac{2\Pi}{\kappa}.$$

The equation above has three constants, $\kappa \sim 0.41$ is the von Kármán constant, $c \sim 5.2$ is the log law constant. These have been experimentally derived for wall bounded boundary layers and it is tempting to see how well it works in this case despite the immediately obvious problem that the mean velocity profile of the SCBL is significantly different to a normal boundary layer far away from the wall. The wake parameter Π also depends on the flow, Coles [54] predicts a value of 0.5 for a zero pressure gradient boundary layer however Pope [185] uses a value of 0.55.

Lattice Boltzmann simulations are weakly compressible requiring a sufficiently small Mach number in order to minimise errors. Thus we add one additional variable, the speed of sound c_s and a corresponding non-dimensional parameter, the Mach number,

$$\mathrm{Ma} = \frac{u_0}{c_s}.$$

Flow parameters used by Conzemius and Fedorovich [56] (C&F) will be used as a guide in selecting the parameter range for study. Physical parameters are specified by C&F rather than non-dimensional ones, we calculate the non-dimensional parameters previously discussed in this section using the following values of kinematic viscosity, thermal diffusivity and density at 1 atm and 25° C,

$$\nu = 1.562 \times 10^{-5}\ \mathrm{m^2s^{-1}}, \quad \alpha = 2.141 \times 10^{-5}\ \mathrm{m^2s^{-1}} \quad \text{and} \quad \rho = 1.184\ \mathrm{kg\ m^{-3}}.$$

The simulations by C&F were allowed to proceed until the boundary layer reached approximately 60% of the 1.6 km domain height, around 960 m. They studied three initial conditions for velocity; zero velocity, constant shear and constant velocity. The initial velocity in the last case is 20 ms^{-1} leading to a Reynolds number of around 10^9.

The potential temperature, a measure of the buoyancy of air based on both temperature and pressure is used by C&F rather than a direct buoyancy field as used in our simulation. The potential temperature flux $\vec{\theta}$ can be related to the buoyancy flux using equation,

$$\vec{B} = \rho_0 g \frac{\vec{\theta}}{\theta_{\text{ref}}}.$$

The potential temperature fluxes used in CF are 0.03, 0.1 and 0.3. Using the law of the wake to predict a friction velocity $u_\tau = 0.752$, these fluxes correspond to stability ratios of 0.91, 3.0, and 9.1.

The potential temperature gradients of 0.01, 0.003 and 0.001 used by C&F are equivalent to bulk Richardson numbers of 0.75, 0.23 and 0.075 using the formula,

$$\frac{dB}{dz} = \frac{\rho_0\, g}{\theta_{\text{ref}}} \frac{d\theta}{dz},$$
$$J = \frac{\frac{dB}{dz}\delta^2}{\rho_0 u_0^2} = \frac{\delta^2\, g}{u_0^2\, \theta_{\text{ref}}} \frac{d\theta}{dz}.$$

The characteristic length scale used in our simulations is half of the domain height. The Richardson number specified will take the desired value when the boundary layer reaches that height. Although we have specified our simulations in terms of this Richardson number, an alternative choice that does not depend on boundary layer height is the second non-dimensional group Π_2. The two are related by Eqn 5.8.

The buoyancy flux into the domain at the top boundary $\vec{B}_{top}$ is set to,

$$\vec{B}_{top} = \alpha \cdot \frac{dB}{dz}$$

Codename	SR	J	Ш	Π_2
A1	6.520	0.50	2.42	3.23×10^{-8}
A2	9.770	0.75	2.42	4.84×10^{-8}
A3	13.05	1.00	2.42	6.46×10^{-8}
B1	19.52	0.50	7.25	9.67×10^{-8}
B2	29.30	0.75	7.25	1.45×10^{-7}
B3	39.05	1.00	7.25	1.93×10^{-7}

Table 5.4: Parameters used in various SCBL simulations.

in order to maintain the correct buoyancy gradient.

5.8.2 Results

We will present the results of six direct numerical simulations which were performed using periodic boundary conditions and a range of parameters detailed in Table 5.4. The stability ratio and Richardson numbers selected are similar to those used by Conzemius and Fedorovich [56] however our DNS needs to be run at a much lower Reynolds number since all turbulent length scales must be fully resolved. All simulations were performed using $\mathrm{Re}_\delta = 1500$ and $\mathrm{Pr} = 0.73$. Only two out of the three parameters SR, J, Ш and Π_2 may be specified, the rest being dependent. The first and second batch of numerical experiments are each run for a fixed values of Ш in order to demonstrate the similarity of simulations based on this parameter. The domain size used for all simulations was $608 \times 302 \times 190$ nodes in the streamwise x, spanwise y and surface normal z directions. The streamwise direction is around 2π times the mixed layer height and the spanwise domain size is approximately π times the mixed layer height noting that the mixed layer height never exceeds 50% of the domain height. These values were chosen based on experience with channel flow simulations. These simulations were performed using the single relaxation time collision operator and the `cub_d3q19_ab` lattice. The scalar buoyancy field was solved using our third order multi-dimensional flux limited NEWTOPIA scheme.

Velocity and buoyancy profiles for various Richardson numbers are shown at a series of times for $\text{Ш} = 2.42$ in Figure 5.18. The times are scaled using the buoyancy time scale t^B. Figure 5.19 shows the same data for $\text{Ш} = 7.25$ simulations. The development of distinct boundary layer, mixed layer (the upper part of the boundary layer below the shear layer) and shear layers is apparent. There is considerable agreement between profiles at different Richardson numbers at the same value of Ш. Pseudocolor plots at the same values are shown in Figure 5.21 and Figure 5.22.

The time development of shape factor, defined as δ^*/θ, displacement thickness (δ^*) and momentum thickness (θ) are shown in Figure 5.23. The shape factor and momentum thickness evolve according to the buoyancy time scale while the displacement thickness evolves according to the viscous time scale. Once again there is remarkable agreement between simulations at the same value of Ш. Figure 5.20 shows velocity and buoyancy profiles from Conzemius and Fedorovich [56] for comparison. As we have explained, our parameters do not exactly match those of C&F, in particular due to our lower Reynolds number chosen to facilitate direct numerical simulation rather than large eddy simulation, thus the figure only serves to show qualitative agreement between our simulations.

The scaled plots presented in this section combined with the preceding dimensional analysis represent the first steps towards establishing a scaling relationship relating various parameters of the flow to the non-dimensional variable Ш. The parameterization will also depend on the last dimensionless group Π_2 although the Ш dependence is expected to dominate. Further simulations in this parameter space are required in order to arrive at the functional form for this parameterization. Our code is well suited to carry out this further investigation due to high computational throughput.

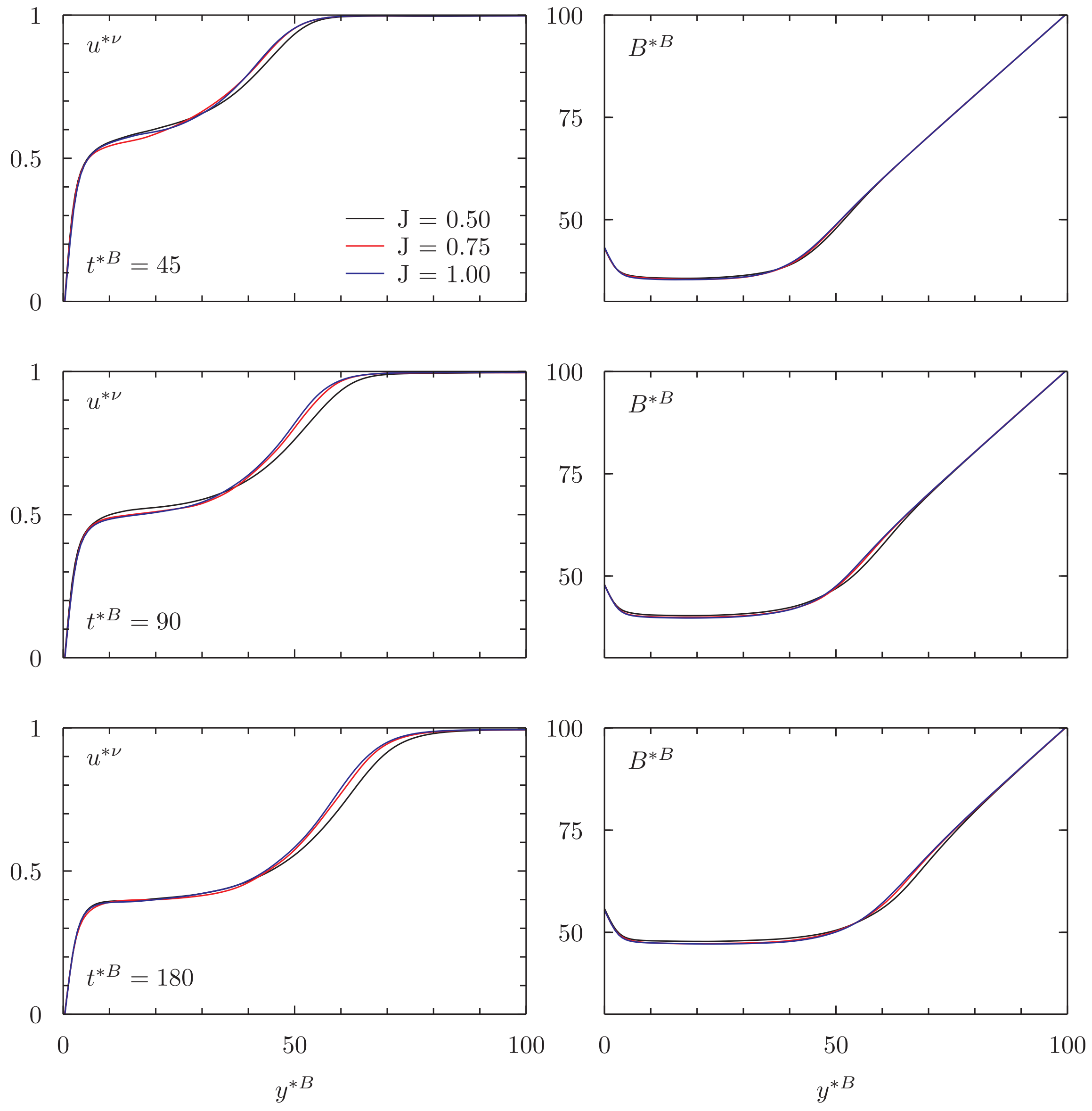


Figure 5.18: Velocity and buoyancy profiles at various times for Ш ~ 2.42.

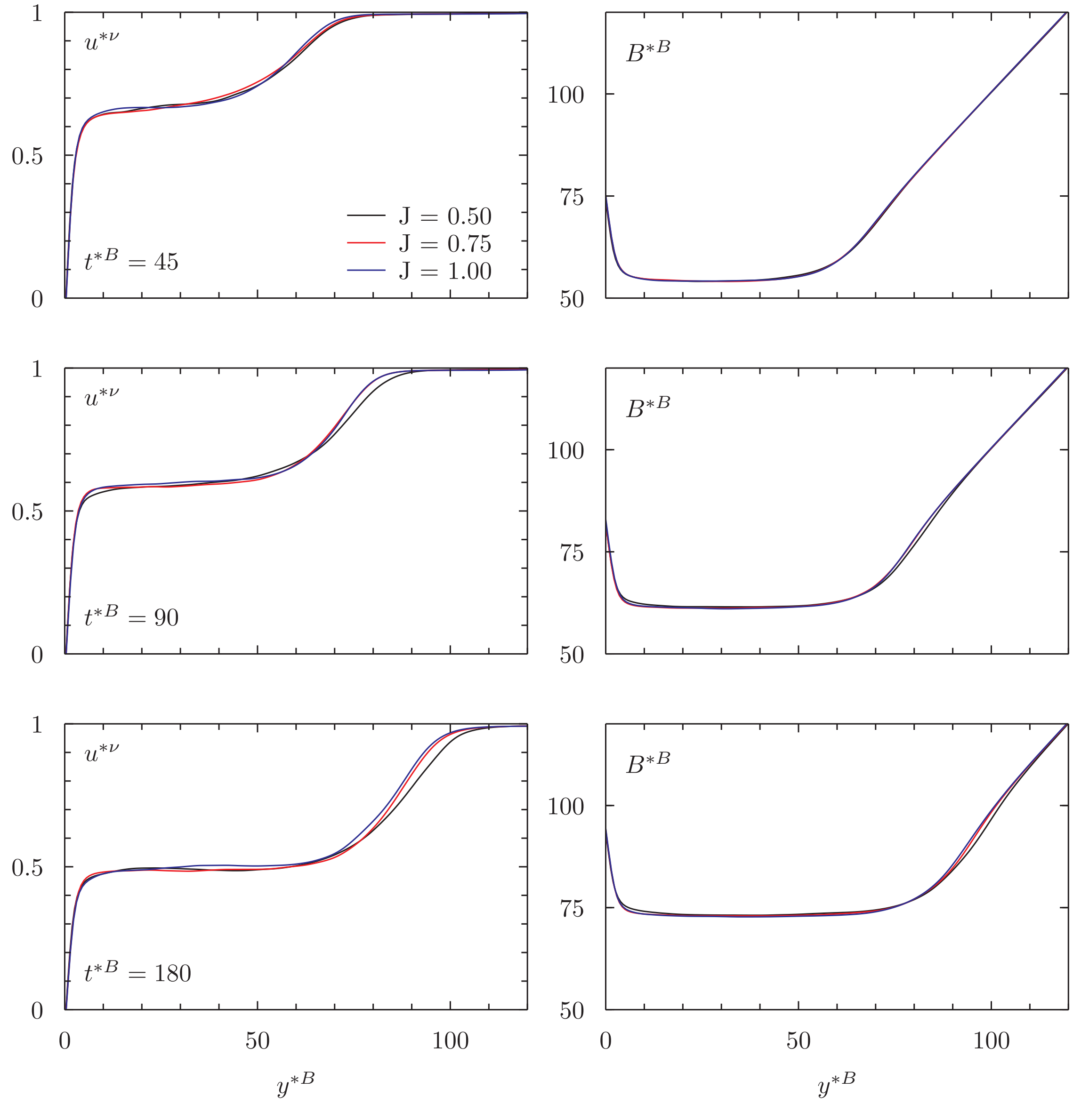


Figure 5.19: Velocity and buoyancy profiles at various times for Ш ∼ 7.25.

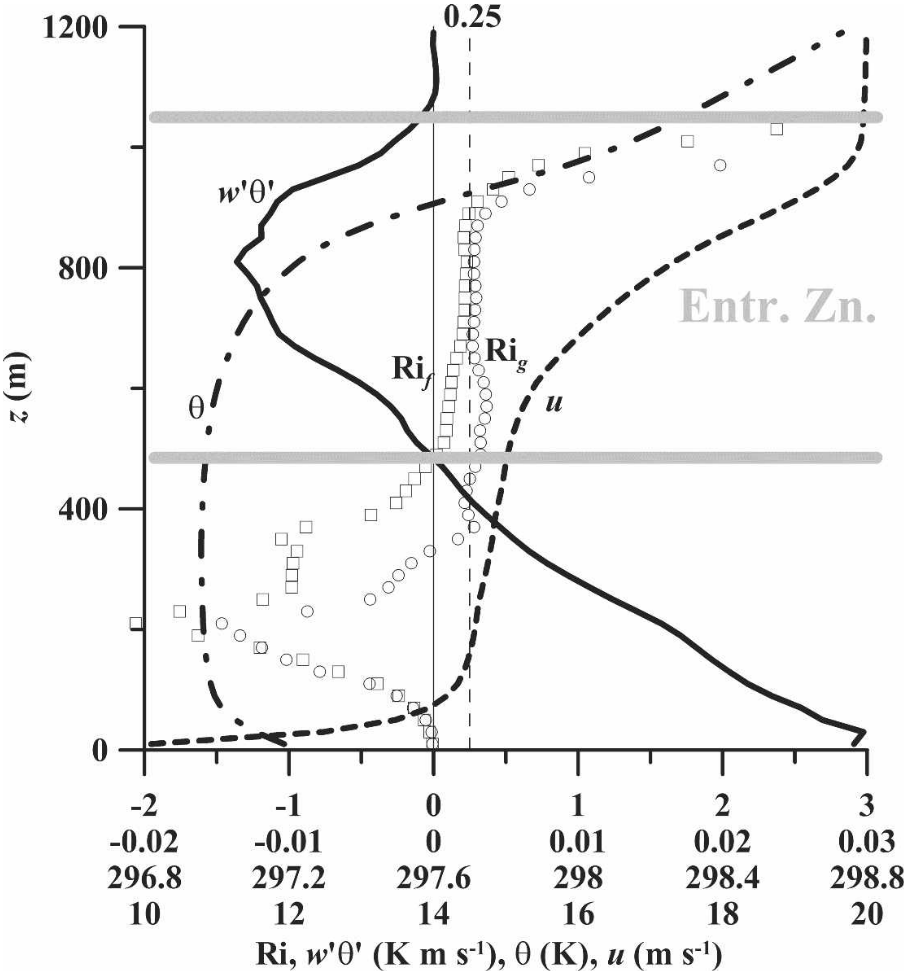


Figure 5.20: Comparison velocity and buoyancy profiles from Conzemius and Fedorovich [56].

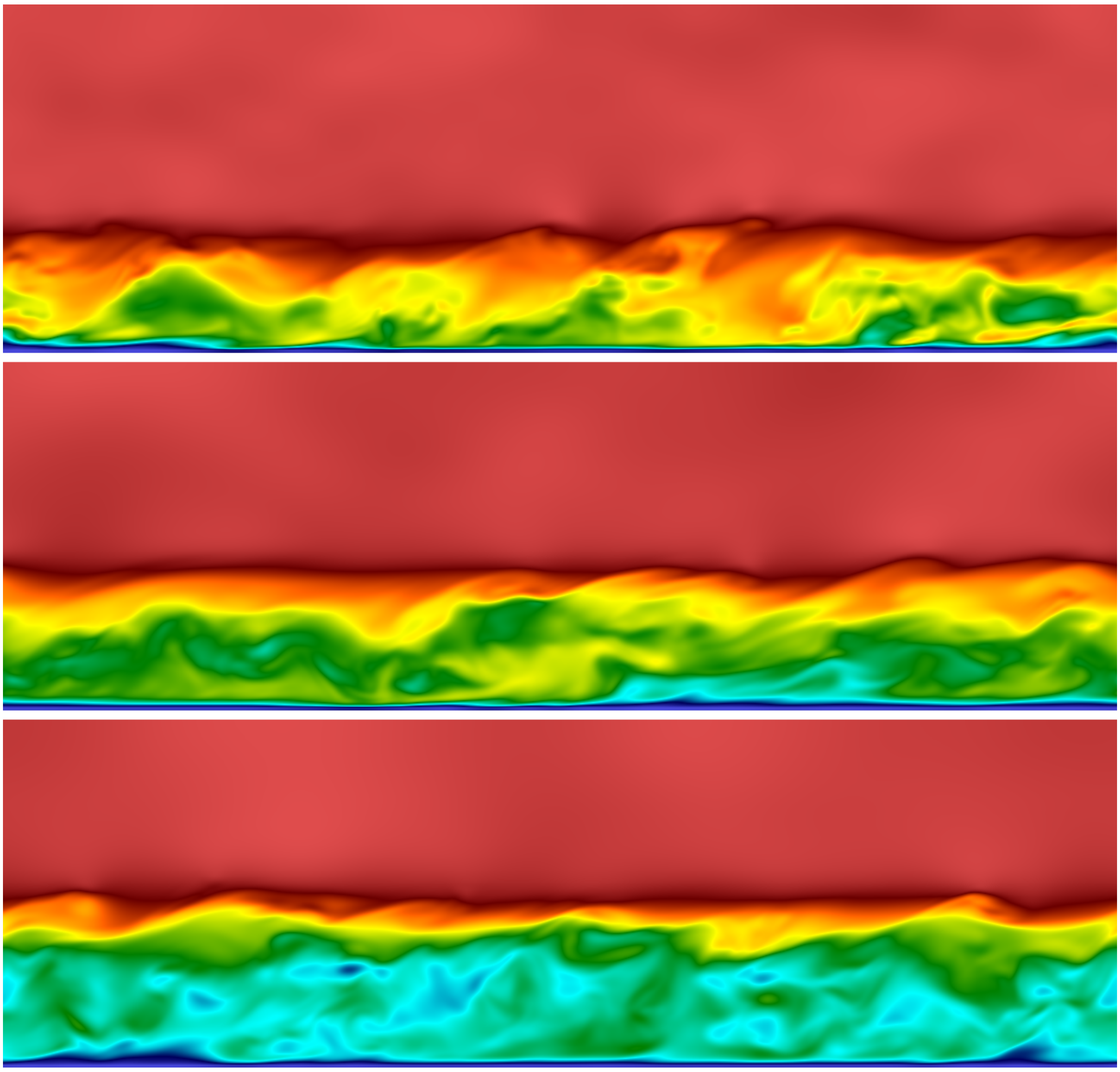


Figure 5.21: Pseudocolor velocity plot of $Ш = 2.42$, $J = 0.75$ simulation at t^{*B} values of 45, 90 and 180.

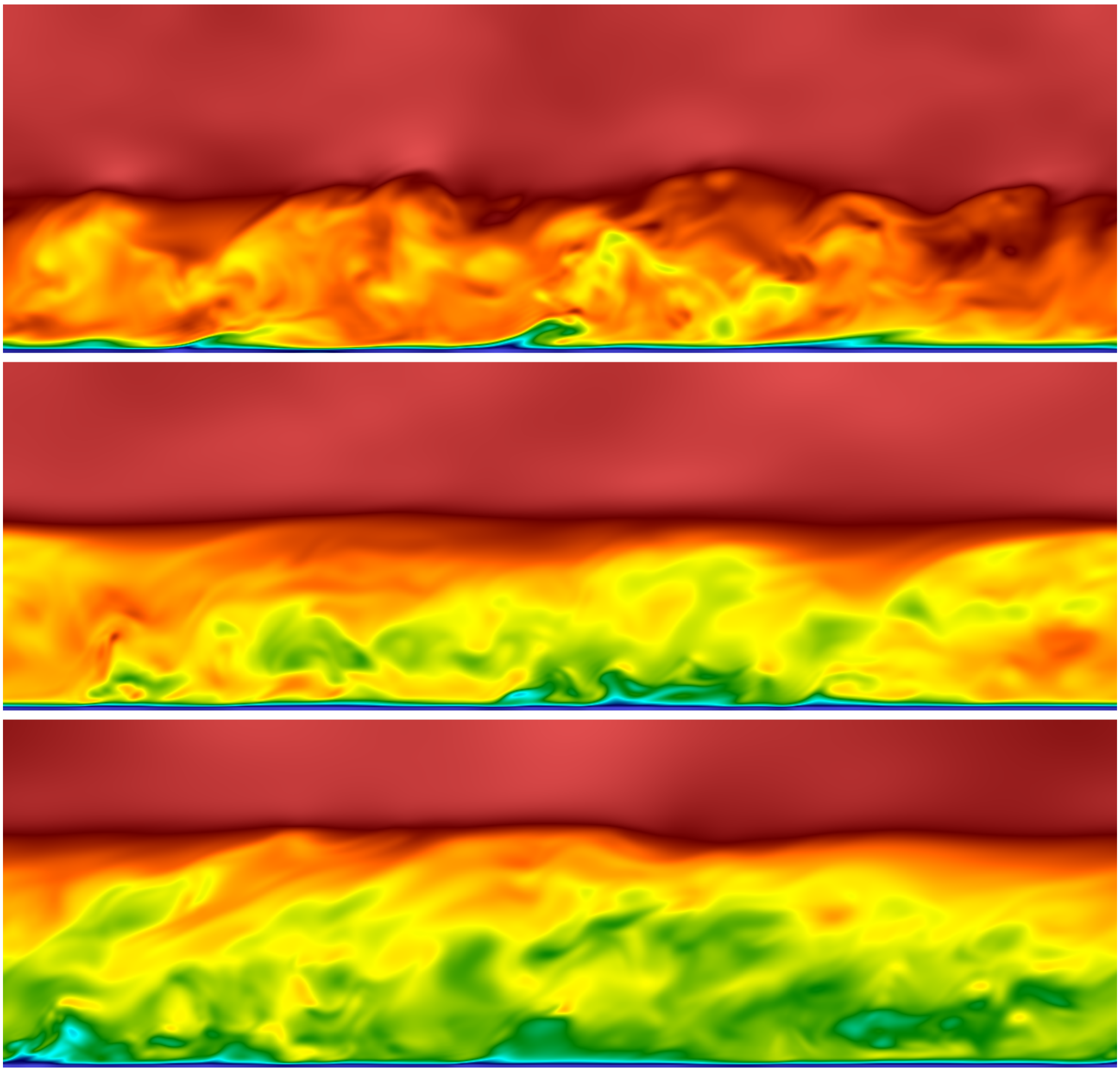


Figure 5.22: Pseudocolor velocity plot of Ш $= 7.25$, $J = 0.75$ simulation at t^{*B} values of 45, 90 and 180.

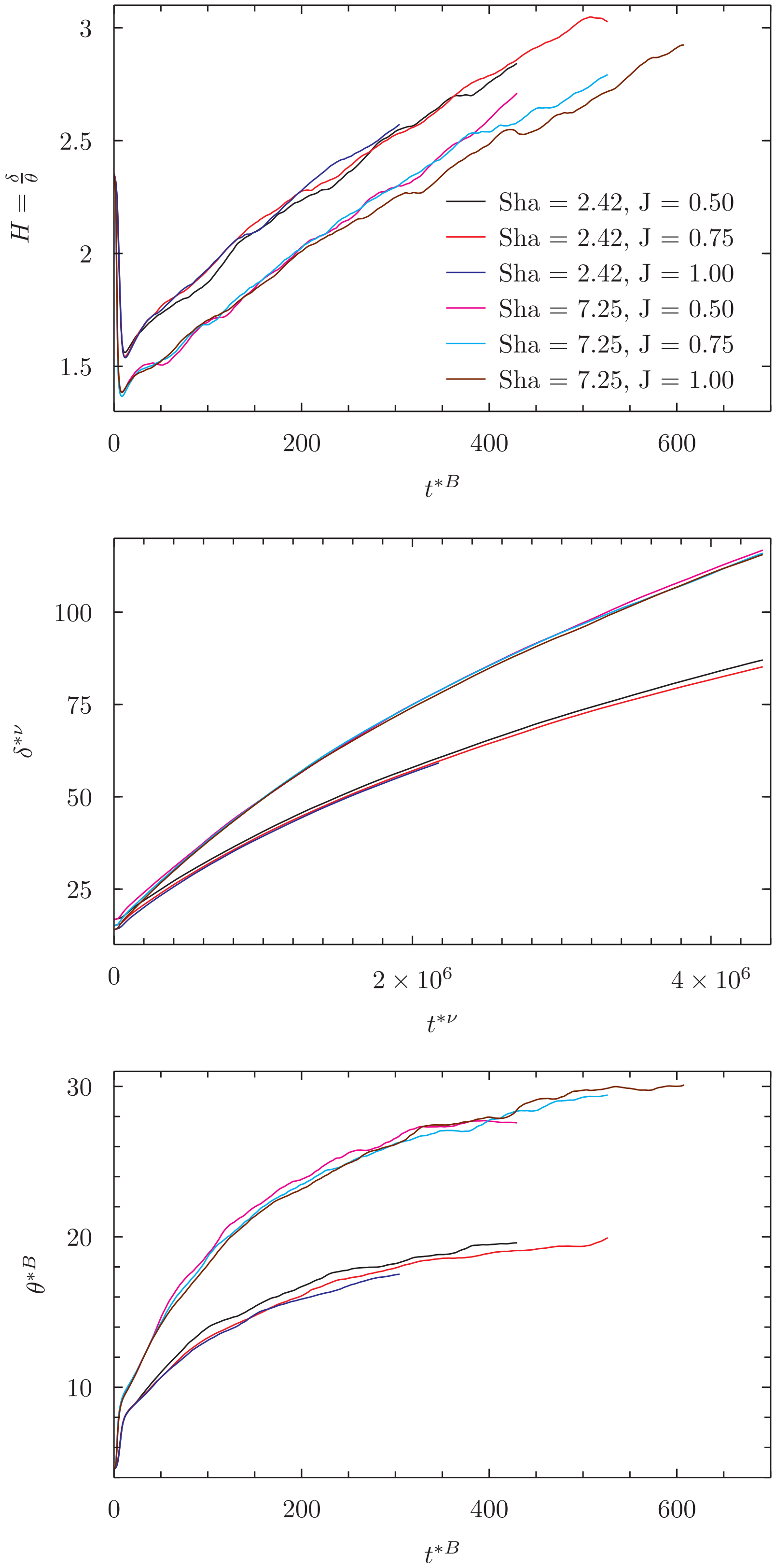


Figure 5.23: Time dependence of shape factor, displacement thickness and momentum thickness.

Chapter 6

Conclusion

Our study of lattice Boltzmann (LB) methods for the simulation of turbulent fluid flows has uncovered several algorithmic improvements. We have presented new non-unit Courant number schemes based on the second order Lax-Wendroff scheme and the first order upwind scheme which use an additional term to cancel errors that are introduced into the multi-scale expansion. These have been tested and shown to improve stability and accuracy. We address the order of accuracy of the force operator which has been the subject of considerable debate. The accuracy of the force operator is maintained when using our modified difference schemes. We have presented alternative topologies based on face and body centered cubic lattices which offer computational improvements compared to the typical cubic lattice. A new formulation of the multiple relaxation time collision operator has been presented. A generalized eigendecomposition has been used which allows a new freedom in tuning the eigenvectors of the linearised collision operator. Applications include a variable bulk viscosity and the use of a stretched grid, our implementation of which has reduced errors present in previous efforts. Our stretched grid implementation has been tested using Taylor-Green vortex flow and turbulent channel flow. We have included full details including all required matrices and moments for a wide range of lattices including new ones based on the body centered cubic lattice. We have performed linear stability analysis on several of the included lattices.

Small Mach number flows where density variations are negligible except in the buoyancy force term allow the use of a highly accurate finite volume solver to simulate the evolution of the buoyancy field. The solver is coupled to the LB simulation as an external force. We use a multidimensional flux limited third order flux integral based advection scheme. The simplified NEWTOPIA algorithm we have devised is easier to implement, has higher performance and does not sacrifice any accuracy compared to the original UTOPIA scheme it is based on. Our algorithm is particularly suited to an outflow based implementation which further simplifies implementation and results in additional performance improvements. Our algorithm is also well suited to the implementation of Thuburn's [242] multidimensional flux limiter which is outflow based. We have presented numerical experiments confirming the third order accuracy of our scheme when applied to multidimensional advection. We have also confirmed that the multidimensional flux limiter reduces the order of accuracy to somewhere between second and third order.

The coupled solver is implemented in a new code that runs in parallel across multiple machines using GPUs. Our code achieves high computational throughput and accuracy and has been used to simulate a range of turbulent flows. The code currently runs on Nvidia's CUDA enabled hardware however we have ported an older branch to use OpenCL. We have also developed two utility programs used to create and analyse collision matrices and to perform linear stability analysis of LB methods.

Details regarding turbulent channel flow have been presented demonstrating correct turbulent kinetic energy profiles and spectra. Our stretched grid scheme has been developed in order to improve grid resolution near the wall, we have successfully used it to simulate turbulent channel flow using moderate grid aspect ratios. We have also performed simulations of the sheared convective boundary layer and found new insight into the scaling properties of the flow, finding close agreement between results using the same shape parameter.

Future work includes further analysis concerning the effects the eigenstructure of the collision matrix on the stability and accuracy of LB methods. Based on preliminary investigations and on previous research [134], we anticipate stability improvements are possible as a result of tuning the eigenvalues corresponding to higher order moments. Our eigendecomposition allows tuning of eigenvectors as well as eigenvalues potentially allowing further stability and accuracy improvements. We also intend to further test and characterize our newly developed lattices. We believe that Thuburn's limiter can be generalized in order to create a new family of limiters similar to the one dimensional total variation diminishing schemes. Further exploration of the sheared convective boundary layer parameter space is expected to yield a parameterization of the flow properties such as boundary layer growth and entrainment rate in terms of relavent non-dimensional parameters. We also plan to port our solvers to Intel's Xeon Phi architecture and update our OpenCL port. Our code has been setup to simulate a variety of other flows which have not been presented in this thesis. We plan to publicly release the code and flow configuration files using an open source license.

Bibliography

[1] Top 500. Tiahne–2. `http://www.top500.org/system/177999/`, 2014. Accessed: 2014-05-29.

[2] C. K. Aidun and J. R. Clausen. Lattice Boltzmann method for complex flows. *Annu. Rev. Fluid Mech.*, 42:439–472, 2010.

[3] A. Alexakis. Stratified shear flow instabilities at large Richardson numbers. *Phys. Fluids*, 21(5), 2009.

[4] F. J. Alexander and A. L. Garcia. The direct simulation Monte Carlo method. *Comput. Phys.*, 11(6):588–593, 1997.

[5] U. R. Alim, A. Entezari, and T. Moeller. The lattice-Boltzmann method on optimal sampling lattices. *IEEE Trans. Vis. Comput. Graph.*, 15(4):630–641, 2009.

[6] Z. Alterman, K. Frankowski, and C. L. Pekeris. Eigenvalues and eigenfunctions of the linearized Boltzmann collision operator for a Maxwell gas and for a gas of rigid spheres. *Astrophys. J. Suppl*, 7:291, 1962.

[7] R. Aris. *Vectors, Tensors, and the Basic Equations of Fluid Mechanics*. Dover, 1989.

[8] S. Armfield and R. Street. The fractional-step method for the Navier-Stokes equations on staggered grids: The accuracy of three variations. *J. Comput. Phys.*, 153(2):660–665, 1999.

[9] S. Armfield and R. Street. An analysis and comparison of the time accuracy of fractional-step methods for the Navier-Stokes equations on staggered grids. *Int. J. Numer. Methods Fluids*, 38(3):255–282, 2002.

[10] M. K. Banda, W. A. Yong, and A. Klar. A stability notion for lattice Boltzmann equations. *SIAM J. Sci. Comput.*, 27(6):2098–2111, 2006.

[11] J. B. Bell, P. Colella, and H. M. Glaz. A second-order projection method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 85(2):257 – 283, 1989.

[12] R. Benzi, S. Succi, and M. Vergassola. The lattice Boltzmann equation — theory and applications. *Phys. Rep.*, 222(3):145–197, 1992.

[13] P. L. Bhatnager, E. P. Gross, and M. Krook. A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems. *Phys. Rev.*, 94(3):511–525, 1954.

[14] G. A. Bird. Breakdown of translational and rotational equilibrium in gaseous expansions. *AIAA J.*, 8(11):1998–2003, 1970.

[15] G. A. Bird. Direct simulation and the Boltzmann equation. *Phys. Fluids*, 13(11):2676–2681, 1970.

[16] G. A. Bird. *Molecular Gas Dynamics and the Direct Simulation of Gas Flows.* Oxford University Press, 1994.

[17] A. V. Bobylev. Instabilities in the Chapman-Enskog expansion and hyperbolic Burnett equations. *J. Stat. Phys.*, 124(2-4):371–399, 2006.

[18] J. P. Boris and D. L. Book. Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works. *J. Comput. Phys.*, 11(1):38–69, 1973.

[19] M. Bouzidi, D. d'Humieres, P. Lallemand, and L. S. Luo. Lattice Boltzmann equation on a two-dimensional rectangular grid. *J. Comput. Phys.*, 172(2):704–717, 2001.

[20] R. A. Brownlee, A. N. Gorban, and J. Levesley. Stability and stabilization of the lattice Boltzmann method. *Phys. Rev. E*, 75(3, Part 2), 2007.

[21] K. A. Brucker and S. Sarkar. Evolution of an initially turbulent stratified shear layer. *Phys. Fluids*, 19(10), 2007.

[22] J. M. Buick and C. A. Greated. Gravity in a lattice Boltzmann model. *Phys. Rev. E*, 61(5 A):5307–5320, 2000.

[23] J. C. Butcher. Coefficients for the study of Runge-Kutta integration processes. *J. Aust. Math. Soc.*, 3:185–201, 1963.

[24] J. C. Butcher. Implicit Runge-Kutta processes. *Math. Comput.*, 18(85):50–64, 1964.

[25] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations*. John Wiley and Sons, 2nd edition, 2008.

[26] A. Caiazzo. Analysis of lattice Boltzmann initialization routines. *J. Stat. Phys.*, 121:37–48, 2005.

[27] Y. Cao and A. Faghri. Simulation of the early startup period of high-temperature heat pipes from the frozen state by a rarefied vapor self-diffusion model. *J. Heat Transf. - Trans. ASME*, 115(1):239–245, 1993.

[28] J. R. Carpenter, N. J. Balmforth, and G. A. Lawrence. Identifying unstable modes in stratified shear layers. *Phys. Fluids*, 22(5), 2010.

[29] C. P. Caulfield and W. R. Peltier. The anatomy of the mixing transition in homogeneous and stratified free shear layers. *J. Fluid Mech.*, 413:1–47, 2000.

[30] H. M. Cave, K. C. Tseng, J. S. Wu, M. C. Jermy, J. C. Huang, and S. P. Krumdieck. Implementation of unsteady sampling procedures for the parallel direct simulation Monte Carlo method. *J. Comput. Phys.*, 227(12):6249–6271, 2008.

[31] Y. A. Cengel and M. A. Boles. *Thermodynamics: An Engineering approach.* McGraw-Hill, 5th edition, 2006.

[32] Y. A. Cengel and R. H. Turner. *Fundamentals of Thermal–Fluid Sciences.* McGraw-Hill, 2005.

[33] C. Cercignani. Methods of solution of linearized Boltzmann equation for rarefied gas dynamics. *J. Quant. Spectrosc. Radiat. Transf.*, 11(6):973–985, 1971.

[34] C. Cercignani. *Theory and application of the Boltzmann equation.* Scottish Academic Press, 1975.

[35] S. Chakravarthy and S. Osher. High resolution applications of the Osher upwind scheme for the Euler equations. In *Fluid Dynamics and Co-located Conferences*, pages 363–372. American Institute of Aeronautics and Astronautics, 1983.

[36] S. Chapman and T. G. Cowling. *The Mathematical Theory of Non-uniform Gases: An Account of the Kinetic Theory of Viscosity, Thermal Conduction and Diffusion in Gases.* Cambridge University Press, 3rd edition, 1970.

[37] J. G. Charney, R. Fjørtoft, and J. Von Neumann. Numerical integration of the barotropic vorticity equation. *Tellus*, 2(4):237–254, 1950.

[38] S. Chen and G. D. Doolen. Lattice Boltzmann method for fluid flows. *Annu. Rev. Fluid Mech.*, 30:329–364, 1998.

[39] S. Y. Chen, D. Martinez, and R. W. Mei. On boundary conditions in lattice Boltzmann methods. *Phys. Fluids*, 8(9):2527–2536, 1996.

[40] Y. Chen, H. Ohashi, and M. Akiyama. Thermal lattice Bhatnagar-Gross-Krook model without nonlinear deviations in macrodynamic equations. *Phys. Rev. E*, 50(4):2776–2783, 1994.

[41] Y. Cheng and J. Li. Introducing unsteady non-uniform source terms into the lattice Boltzmann model. *Int. J. Numer. Methods Fluids*, 56(6):629–641, 2008.

[42] S. S. Chikatamarla and I. V. Karlin. Entropy and Galilean invariance of lattice Boltzmann theories. *Phys. Rev. Lett.*, 97(19), 2006.

[43] S. S. Chikatamarla and I. V. Karlin. Complete Galilean invariant lattice Boltzmann models. *Comput. Phys. Commun.*, 179(1-3):140–143, 2008.

[44] S. S. Chikatamarla and I. V. Karlin. Lattices for the lattice Boltzmann method. *Phys. Rev. E*, 79(4), 2009.

[45] S. S. Chikatamarla and I. V. Karlin. Comment on "Rectangular lattice Boltzmann method". *Phys. Rev. E*, 83(4, Part 2), 2011.

[46] A. J. Chorin. A numerical method for solving incompressible viscous flow problems. *J. Comput. Phys.*, 2(1):12 – 26, 1967.

[47] A. J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comput.*, 22(104):745–762, 1968.

[48] S. M. Churilov. Stability analysis of stratified shear flows with a monotonic velocity profile without inflection points. *J. Fluid Mech.*, 539:25–55, 2005.

[49] S. M. Churilov. Stability analysis of stratified shear flows with a monotonic velocity profile without inflection points. Part 2. Continuous density variation. *J. Fluid Mech.*, 617:301–326, 2008.

[50] S. M. Churilov. Nonlinear stage of instability development in a stratified shear flow with an inflection-free velocity profile. *Phys. Fluids*, 21(7), 2009.

[51] S. M. Churilov. Three-dimensional instability of shear flows with inflection-free velocity profiles in stratified media with a high Prandtl number. *Izv. Atmos. Ocean. Phys.*, 46(2):159–168, 2010.

[52] S. M. Churilov. Resonant three-wave interaction of Holmboe waves in a sharply stratified shear flow with an inflection-free velocity profile. *Phys. Fluids*, 23(11), 2011.

[53] J. I. Cirac and P. Zoller. Goals and opportunities in quantum simulation. *Nat. Phys.*, 8(4):264–266, 2012.

[54] D. Coles. The law of the wake in the turbulent boundary layer. *J. Fluid Mech.*, 1(2):191–226, 1956.

[55] N. C. Constantinou and P. J. Ioannou. Optimal excitation of two dimensional Holmboe instabilities. *Phys. Fluids*, 23(7), 2011.

[56] R. J. Conzemius and E. Fedorovich. Dynamics of sheared convective boundary layer entrainment. Part I: Methodological background and large-eddy simulations. *J. Atmos. Sci.*, 63(4):1151–1178, 2006.

[57] R. J. Conzemius and E. Fedorovich. Dynamics of sheared convective boundary layer entrainment. Part II: Evaluation of bulk model predictions of entrainment flux. *J. Atmos. Sci.*, 63(4):1179–1199, 2006.

[58] R. J. Conzemius and E. Fedorovich. Bulk models of the sheared convective boundary layer: Evaluation through large eddy simulations. *J. Atmos. Sci.*, 64(3):786–807, 2007.

[59] R. Cornubert, D. d'Humieres, and D. Levermore. A Knudsen layer theory for lattice gases. *Physica D*, 47(1-2):241–259, 1991.

[60] M. S. Cramer. Numerical estimates for the bulk viscosity of ideal gases. *Phys. Fluids*, 24(6), 2012.

[61] G. G. Dahlquist. A special stability problem for linear multistep methods. *BIT Numer. Math.*, 3(1):27–43, 1963.

[62] D. d'Humieres, M. Bouzidi, and P. Lallemand. Thirteen-velocity three-dimensional lattice Boltzmann model. *Phys. Rev. E*, 63(6, Part 2), 2001.

[63] D. d'Humieres, I. Ginzburg, M. Krafczyk, P. Lallemand, and L. S. Luo. Multiple-relaxation-time lattice Boltzmann models in three dimensions. *Philos. Trans. R. Soc. A - Math. Phys. Eng. Sci.*, 360(1792):437–451, 2002.

[64] D. d'Humieres and P. Lallemand. Lattice gas automata for fluid-mechanics. *Physica A*, 140(1-2):326–335, 1986.

[65] D. d'Humieres and P. Lallemand. Numerical simulations of hydrodynamics with lattice gas automata in two dimensions. *Complex Syst.*, 1:599–632, 1987.

[66] D. d'Humieres, P. Lallemand, and U. Frisch. Lattice gas models for 3D hydrodynamics. *Europhys. Lett.*, 2(4):291–297, 1986.

[67] D. d'Humieres, P. Lallemand, and Y. H. Qian. Review of flow simulations using lattice gases. *Lect. Notes Math.*, 1402:56–68, 1989.

[68] H. N. Dixit and V. Babu. Simulation of high Rayleigh number natural convection in a square cavity using the lattice Boltzmann method. *Int. J. Heat Mass Transf.*, 49(3-4):727–739, 2006.

[69] R. Du and B. Shi. A novel scheme for force term in the lattice BGK model. *Int. J. Mod. Phys. C*, 17(7):945–958, 2006.

[70] M. Duongvan, M. D. Feit, P. Keller, and M. Pound. The nature of turbulence in a triangular lattice gas automaton. *Physica D*, 23(1-3):448–454, 1986.

[71] M. H. Ernst. Linearized Boltzmann equation - Navier-Stokes and Burnett transport coefficients. *Am. J. Phys*, 38(7):908–&, 1970.

[72] E. Fedorovich, P. Kaiser, M. Rau, and E. Plate. Wind tunnel study of turbulent flow structure in the convective boundary layer capped by a temperature inversion. *J. Atmos. Sci.*, 53(9):1273–1289, 1996.

[73] E. Fedorovich and R. Kaiser. Wind tunnel model study of turbulence regime in the atmospheric convective boundary layer. In *Buoyant convection in geo-*

physical flows, volume 513 of *NATO Advanced Science Institutes Series, Series C, Mathematical and Physical Sciences*, pages 327–370, 1998.

[74] E. Fedorovich, F. T. M. Nieuwstadt, and R. Kaiser. Numerical and laboratory study of a horizontally evolving convective boundary layer. Part I: Transition regimes and development of the mixed layer. *J. Atmos. Sci.*, 58(1):70–86, 2001.

[75] E. Fedorovich, F. T. M. Nieuwstadt, and R. Kaiser. Numerical and laboratory study of horizontally evolving convective boundary layer. Part II: Effects of elevated wind shear and surface roughness. *J. Atmos. Sci.*, 58(6):546–560, 2001.

[76] E. Fedorovich and J. Thater. Vertical transport of heat and momentum across a sheared density interface at the top of a horizontally evolving convective boundary layer. *J. Turbul.*, 2, 2001.

[77] E. Ferrero, L. Quan, and D. Massone. Turbulence in the stable boundary layer at higher Richardson numbers. *Boundary Layer Meteorol.*, 139(2):225–240, 2011.

[78] R. P. Feynman. Simulating physics with computers. *Int. J. Theor. Phys.*, 21(6-7):467–488, 1982.

[79] O. Filippova and D. Hanel. Grid refinement for lattice-BGK models. *J. Comput. Phys.*, 147(1):219–228, 1998.

[80] O. Filippova and D. Hanel. A novel lattice BGK approach for low Mach number combustion. *J. Comput. Phys.*, 158(2):139–160, 2000.

[81] O. B. Fringer, S. W. Armfield, and R. L. Street. Reducing numerical diffusion in interfacial gravity wave simulations. *Int. J. Numer. Methods Fluids*, 49(3):301–329, 2005.

[82] U. Frisch, B. Hasslacher, and Y. Pomeau. Lattice-gas automata for the Navier-Stokes equation. *Phys. Rev. Lett.*, 56(14):1505–1508, 1986.

[83] J. E. Fromm. A method for reducing dispersion in convective difference schemes. *J. Comput. Phys.*, 3(2):176 – 189, 1968.

[84] Y. Gan, A. Xu, G. Zhang, and Y. Li. Lattice Boltzmann study on Kelvin-Helmholtz instability: Roles of velocity and density gradients. *Phys. Rev. E*, 83(5, 2), 2011.

[85] L. S. Garcia-Colin, R. M. Velasco, and F. J. Uribe. Beyond the Navier-Stokes equations: Burnett hydrodynamics. *Phys. Rep. - Rev. Sec. Phys. Lett.*, 465(4):149–189, 2008.

[86] I. Ginzbourg and P. M. Adler. Boundary flow condition analysis for the 3-dimensional lattice Boltzmann model. *J. Phys. II*, 4(2):191–214, 1994.

[87] I. Ginzbourg and D. d'Humieres. Local second-order boundary methods for lattice Boltzmann models. *J. Stat. Phys.*, 84(5-6):927–971, 1996.

[88] H. Grad. Note on n-dimensional Hermite polynomials. *Commun. Pure Appl. Math.*, 2(4):325–330, 1949.

[89] H. Grad. On the kinetic theory of rarefied gases. *Commun. Pure Appl. Math.*, 2(4):331–407, 1949.

[90] H. Grad. The profile of a steady plane shock wave. *Commun. Pure Appl. Math.*, 5(3):257–300, 1952.

[91] R. E. Graves and B. M. Argrow. Bulk viscosity: Past to present. *J. Thermophys. Heat Transfer*, 13(3):337–342, 1999.

[92] P. M. Gresho. On the theory of semiimplicit projection methods for viscous incompressible-flow and its implementation via a finite-element method that also introduces a nearly consistent mass matrix. 1. Theory. *Int. J. Numer. Methods Fluids*, 11(5):587–620, 1990.

[93] P. M. Gresho and S. T. Chan. On the theory of semiimplicit projection methods for viscous incompressible-flow and its implementation via a finite-element

method that also introduces a nearly consistent mass matrix. 2. Implementation. *Int. J. Numer. Methods Fluids*, 11(5):621–659, 1990.

[94] P. M. Gresho and R. l. Sani. On pressure boundary-conditions for the incompressible Navier-Stokes equations. *Int. J. Numer. Methods Fluids*, 7(10):1111–1145, 1987.

[95] E. P. Gross and E. A. Jackson. Kinetic models and the linearized Boltzmann equation. *Phys. Fluids*, 2(4):433–441, 1959.

[96] Z. Guo, C. Zheng, B. Shi, and T. S. Zhao. Thermal lattice Boltzmann equation for low mach number flows: Decoupling model. *Phys. Rev. E*, 75(3, 2), 2007.

[97] Z. L. Guo, B. C. Shi, and C. G. Zheng. A coupled lattice BGK model for the Boussinesq equations. *Int. J. Numer. Methods Fluids*, 39(4):325–342, 2002.

[98] Z. L. Guo, C. G. Zheng, and B. C. Shi. Discrete lattice effects on the forcing term in the lattice Boltzmann method. *Phys. Rev. E*, 65(4, Part 2B), 2002.

[99] Z. L. Guo, C. G. Zheng, and B. C. Shi. An extrapolation method for boundary conditions in lattice Boltzmann method. *Phys. Fluids*, 14(6):2007–2010, 2002.

[100] Z. L. Guo, C. G. Zheng, and B. C. Shi. Non-equilibrium extrapolation method for velocity and pressure boundary conditions in the lattice Boltzmann method. *Chin. Phys.*, 11(4):366–374, 2002.

[101] X. He, S. Chen, and G. D. Doolen. A novel thermal model for the lattice Boltzmann method in incompressible limit. *J. Comput. Phys.*, 146(1):282–300, 1998.

[102] X. Y. He and G. D. Doolen. Lattice Boltzmann method on a curvilinear coordinate system: Vortex shedding behind a circular cylinder. *Phys. Rev. E*, 56(1, Part a):434–440, 1997.

[103] X. Y. He and L. S. Luo. A priori derivation of the lattice Boltzmann equation. *Phys. Rev. E*, 55(6, A):R6333–R6336, 1997.

[104] X. Y. He, L. S. Luo, and M. Dembo. Some progress in lattice Boltzmann method. 1. Nonuniform mesh grids. *J. Comput. Phys.*, 129(2):357–363, 1996.

[105] Y. He. The Crank-Nicolson/Adams-Bashforth scheme for the time-dependent Navier-Stokes equations with nonsmooth initial data. *Numer. Meth. Part Differ. Equ.*, 28(1):155–187, 2012.

[106] Y. He and W. Sun. Stability and convergence of the Crank-Nicolson/Adams-Bashforth scheme for the time-dependent Navier-Stokes equations. *SIAM J. Numer. Anal.*, 45(2):837–869, 2007.

[107] V. Henson. The kernel hacker's bookshelf: Ultimate physical limits of computation. `http://lwn.net/Articles/286233/`, 2008. Accessed: 2014-05-29.

[108] F. J. Higuera and S. Succi. Simulating the flow around a circular-cylinder with a lattice Boltzmann-equation. *Europhys. Lett.*, 8(6):517–521, 1989.

[109] F. J. Higuera, S. Succi, and R. Benzi. Lattice gas-dynamics with enhanced collisions. *Europhys. Lett.*, 9(4):345–349, 1989.

[110] J. R. Ho, C. P. Kuo, W. S. Jiaung, and C. J. Twu. Lattice Boltzmann scheme for hyperbolic heat conduction equation. *Numer. Heat Transf. B - Fundam.*, 41(6):591–607, 2002.

[111] E. V. Holm. A fully 2-dimensional, nonoscillatory advection scheme for momentum and scalar transport-equations. *Mon. Weather Rev.*, 123(10):3125, 1995.

[112] T. Inamuro and B. Sturtevant. Numerical study of discrete-velocity gases. *Phys. Fluids A*, 2(12):2196–2203, 1990.

[113] T. Inamuro, M. Yoshino, and F. Ogino. Non-slip boundary-condition for lattice Boltzmann simulations. *Phys. Fluids*, 7(12):2928–2930, 1995.

[114] R. I. Issa. Solution of the implicitly discretized fluid-flow equations by operator-splitting. *J. Comput. Phys.*, 62(1):40–65, 1986.

[115] R. G. Jacobs and P. A. Durbin. Simulations of bypass transition. *J. Fluid Mech.*, 428:185–212, 2001.

[116] D. S. Jang, R. Jetli, and S. Acharya. Comparison of the PISO, SIMPLER, and SIMPLEC algorithms for the treatment of the pressure velocity coupling in steady flow problems. *Numer. Heat Transf.*, 10(3):209–228, 1986.

[117] R. Kaiser and E. Fedorovich. Turbulence spectra and dissipation rates in a wind tunnel model of the atmospheric convective boundary layer. *J. Atmos. Sci.*, 55(4):580–594, 1998.

[118] N. A. Kampanis and J. A. Ekaterinaris. A staggered grid, high-order accurate method for the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 215(2):589–613, 2006.

[119] D. Kandhai, A. Koponen, A. Hoekstra, M. Kataja, J. Timonen, and P. M. A. Sloot. Implementation aspects of 3D lattice-BGK: Boundaries, accuracy, and a new fast relaxation method. *J. Comput. Phys.*, 150(2):482–501, 1999.

[120] S. M. Karim and L. Rosenhead. The 2nd coefficient of viscosity of liquids and gases. *Rev. Mod. Phys.*, 24(2):108–116, 1952.

[121] I. Karlin and P. Asinari. Factorization symmetry in the lattice Boltzmann method. *Physica A*, 389(8):1530–1548, 2010.

[122] A. Kaufman, Z. Fan, and K. Petkov. Implementing the lattice Boltzmann model on commodity graphics hardware. *J. Stat. Mech: Theory Exp.*, 2009.

[123] J. Kim and P. Moin. Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comput. Phys.*, 59(2):308–323, 1985.

[124] J. Kim, P. Moin, and R. Moser. Turbulence statistics in fully-developed channel flow at low Reynolds-number. *J. Fluid Mech.*, 177:133–166, 1987.

[125] S. Kim, S. Park, D. Pino, and J. Vila-Guerau de Arellano. Parameterization of entrainment in a sheared convective boundary layer using a first-order jump

model. *Boundary Layer Meteorol.*, 120(3):455–475, 2006.

[126] M. P. Kirkpatrick. *A large eddy simulation code for industrial and environmental flows.* PhD thesis, The University of Sydney, 2002.

[127] L. Kleiser and T. A. Zang. Numerical-simulation of transition in wall-bounded shear flows. *Annu. Rev. Fluid Mech.*, 23:495–537, 1991.

[128] P. Kowalczyk, A. Palczewski, G. Russo, and Z. Walenta. Numerical solutions of the Boltzmann equation: comparison of different algorithms. *Eur. J. Mech. B - Fluids*, 27(1):62–74, 2008.

[129] A. Kurganov and E. Tadmor. New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations. *J. Comput. Phys.*, 160(1):241–282, 2000.

[130] A. Kuzmin, Z. L. Guo, and A. A. Mohamad. Simultaneous incorporation of mass and force terms in the multi-relaxation-time framework for lattice Boltzmann schemes. *Philos. Trans. R. Soc. A - Math. Phys. Eng. Sci.*, 369(1944):2219–2227, 2011.

[131] F. Kuznik, J. Vareilles, G. Rusaouen, and G. Krauss. A double-population lattice Boltzmann method with non-uniform mesh for the simulation of natural convection in a square cavity. *Int. J. Heat Fluid Flow*, 28(5):862–870, 2007.

[132] A. J. C. Ladd and R. Verberg. Lattice-Boltzmann simulations of particle-fluid suspensions. *J. Stat. Phys.*, 104(5-6):1191–1251, 2001.

[133] P. Lallemand and L. S. Luo. Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability. *Phys. Rev. E*, 61(6, Part A):6546–6562, 2000.

[134] P. Lallemand and L. S. Luo. Theory of the lattice Boltzmann method: Acoustic and thermal properties in two and three dimensions. *Phys. Rev. E*, 68(3, Part 2), 2003.

[135] J. Latt, B. Chopard, O. Malaspinas, M. Deville, and A. Michler. Straight velocity boundaries in the lattice Boltzmann method. *Phys. Rev. E*, 77(5, Part 2), 2008.

[136] M. K. Lawrence and D. S. Glenn. Universal limits on computation. *arXiv*, astro-ph/0404510v2:1–4, 2004.

[137] P. Lax and B. Wendroff. Systems of conservation laws. *Commun. Pure Appl. Math.*, 13(2):217–237, 1960.

[138] P. D. Lax and B. Wendroff. Difference schemes for hyperbolic equations with high order of accuracy. *Commun. Pure Appl. Math.*, 17(3):381–&, 1964.

[139] J. H. Lee and H. J. Sung. Direct numerical simulation of a turbulent boundary layer up to Re_θ=2500. *Int. J. Heat Fluid Flow*, 32(1):1–10, 2011.

[140] B. P. Leonard. Stable and accurate convective modeling procedure based on quadratic upstream interpolation. *Comput. Meth. Appl. Mech. Eng.*, 19(1):59–98, 1979.

[141] B. P. Leonard. The ultimate conservative difference scheme applied to unsteady one-dimensional advection. *Comput. Meth. Appl. Mech. Eng.*, 88(1):17–74, 1991.

[142] B. P. Leonard, A. P. Lock, and M. K. MacVean. Conservative explicit unrestricted-time-step multidimensional constancy-preserving advection schemes. *Mon. Weather Rev.*, 124(11):2588–2606, 1996.

[143] B. P. Leonard, M. K. MacVean, and A. P. Lock. Positivity-preserving numerical schemes for multidimensional advection. Technical report, NASA technical memorandum, 1993.

[144] B. P. Leonard, M. K. MacVean, and A. P. Lock. The flux integral method for multidimensional convection and diffusion. *Appl. Math. Modell.*, 19(6):333–342, 1995.

[145] S. Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, 1996.

[146] S. Lloyd. Ultimate physical limits to computation. *Nature*, 406(6799):1047–1054, 2000.

[147] T. S. Lund, X. H. Wu, and K. D. Squires. Generation of turbulent inflow data for spatially-developing boundary layer simulations. *J. Comput. Phys.*, 140(2):233–258, 1998.

[148] R. Machado. On the moment system and a flexible Prandtl number. *Mod. Phys. Lett. B*, 28(6), 2014.

[149] M. Macrossan. Scaling parameters in rarefied flow: breakdown of the Navier-Stokes equations. Departmental Report 2006/03, Centre for Hypersonics, University of Queensland, Mechanical Engineering, 2006.

[150] N. S. Martys, X. Shan, and H. Chen. Evaluation of the external force term in the discrete Boltzmann equation. *Phys. Rev. E*, 58(5 B):6855–6857, 1998.

[151] J. Mathew. Extending temporal simulations. *Appl. Math. Lett.*, 23(3):222–225, 2010.

[152] K. K. Mattila, D. N. Siebert, L. A. Hegele, and P. C. Philippi. High-order lattice-Boltzmann equations and stencils for multiphase models. *Int. J. Mod. Phys. C*, 24(12, SI), 2013.

[153] G. Mcnamara and B. Alder. Analysis of the lattice Boltzmann treatment of hydrodynamics. *Physica A*, 194(1-4):218–228, 1993.

[154] G. R. McNamara, A. L. Garcia, and B. J. Alder. A hydrodynamically correct thermal lattice Boltzmann model. *J. Stat. Phys.*, 87(5-6):1111–1121, 1997.

[155] R. Mei, L. S. Luo, P. Lallemand, and D. d'Humieres. Consistent initial conditions for lattice Boltzmann simulations. *Comput. Fluids*, 35(8-9):855 – 862, 2006.

[156] R. W. Mei, L. S. Luo, and W. Shyy. An accurate curved boundary treatment in the lattice Boltzmann method. *J. Comput. Phys.*, 155(2):307–330, 1999.

[157] S. C. Mishra, B. Mondal, T. Kush, and B. S. R. Krishna. Solving transient heat conduction problems on uniform and non-uniform lattices using the lattice Boltzmann method. *Int. Commun. Heat Mass Transfer*, 36(4):322–328, 2009.

[158] A. A. Mohamad and A. Kuzmin. A critical evaluation of force term in lattice Boltzmann method, natural convection problem. *Int. J. Heat Mass Transf.*, 53(5-6):990–996, 2010.

[159] A. B. Morris, P. L. Varghese, and D. B. Goldstein. Monte Carlo solution of the Boltzmann equation via a discrete velocity model. *J. Comput. Phys.*, 230(4):1265–1280, 2011.

[160] R. D. Moser, J. Kim, and N. N. Mansour. Direct numerical simulation of turbulent channel flow up to Re_τ=590. *Phys. Fluids*, 11(4):943–945, 1999.

[161] E. P. Muntz. Rarefied-gas dynamics. *Annu. Rev. Fluid Mech.*, 21:387–417, 1989.

[162] H. Nessyahu and E. Tadmor. Non-oscillatory central differencing for hyperbolic conservation-laws. *J. Comput. Phys.*, 87(2):408–463, 1990.

[163] D. R. Noble, S. Y. Chen, J. G. Georgiadis, and R. O. Buckius. A consistent hydrodynamic boundary-condition for the lattice Boltzmann method. *Phys. Fluids*, 7(1):203–209, 1995.

[164] C. Obrecht, F. Kuznik, B. Tourancheau, and J. Roux. The `TheLMA` project: A thermal lattice Boltzmann solver for the GPU. *Comput. Fluids*, 54:118–126, 2012.

[165] T. Ohwada and P. Asinari. Artificial compressibility method revisited: Asymptotic numerical method for incompressible Navier-Stokes equations. *J. Comput. Phys.*, 229(5):1698–1723, 2010.

[166] T. Ohwada, P. Asinari, and D. Yabusaki. Artificial compressibility method and lattice Boltzmann method: Similarities and differences. *Comput. Math. Appl.*, 61(12):3461–3474, 2011.

[167] G. Ortiz, J. E. Gubernatis, E. Knill, and R. Laflamme. Quantum algorithms for fermionic simulations. *Phys. Rev. A*, 64(2):art. no.–022319, 2001.

[168] A. Palczewski, J. Schneider, and A. V. Bobylev. A consistency result for a discrete-velocity model of the Boltzmann equation. *SIAM J. Numer. Anal.*, 34(5):1865–1883, 1997.

[169] T. Passot and A. Pouquet. Numerical-simulation of compressible homogeneous flows in the turbulent regime. *J. Fluid Mech.*, 181:441–466, 1987.

[170] S. V. Patankar. *Numerical Heat Transfer and Fluid Flow.* Series in computational methods in mechanics and thermal sciences. Hemisphere Publishing Corporation, 1980.

[171] S. V. Patankar. A calculation procedure for two-dimensional elliptic situations. *Numer. Heat Transf.*, 4(4):409–425, 1981.

[172] S. V. Patankar and D. B. Spalding. Calculation procedure for heat, mass and momentum-transfer in 3-dimensional parabolic flows. *Int. J. Heat Mass Transfer*, 15(10):1787–&, 1972.

[173] D. V. Patil and K. N. Lakshmisha. Finite volume TVD formulation of lattice Boltzmann simulation on unstructured mesh. *J. Comput. Phys.*, 228(14):5262–5279, 2009.

[174] P. Pavlo, G. Vahala, and L. Vahala. Higher order isotropic velocity grids in lattice methods. *Phys. Rev. Lett.*, 80(18):3960–3963, 1998.

[175] W. R. Peltier and C. P. Caulfield. Mixing efficiency in stratified shear flows. *Annu. Rev. Fluid Mech.*, 35:135–167, 2003.

[176] A. Perez, F. J. Luque, and M. Orozco. Frontiers in molecular dynamics simulations of DNA. *Acc. Chem. Res.*, 45(2):196–205, 2012.

[177] K. Petkov, F. Qiu, Z. Fan, A. E. Kaufman, and K. Mueller. Efficient LBM visual simulation on face-centered cubic lattices. *IEEE Trans. Vis. Comput. Graph.*, 15(5):802–814, 2009.

[178] P. C. Philippi, L. A. Hegele, L. O. E. dos Santos, and R. Surmas. From the continuous to the lattice Boltzmann equation: The discretization problem and thermal models. *Phys. Rev. E*, 73(5, 2), 2006.

[179] P. C. Philippi, L. A. Hegele, R. Surmas, D. N. Siebert, and L. O. E. dos Santos. From the Boltzmann to the lattice-Boltzmann equation: Beyond BGK collision models. *Int. J. Mod. Phys. C*, 18(4):556–565, 2007.

[180] P. C. Philippi, K. K. Mattila, D. N. Siebert, L. O. E. dos Santos, L. A. Hegele, and R. Surmas. Lattice-Boltzmann equations for describing segregation in non-ideal mixtures. *J. Fluid Mech.*, 713:564–587, 2012.

[181] D. Pino and J. V. G. de Arellano. Effects of shear in the convective boundary layer: analysis of the turbulent kinetic energy budget. *Acta Geophys.*, 56(1):167–193, 2008.

[182] D. Pino, J. V. G. de Arellano, and P. G. Duynkerke. The contribution of shear to the evolution of a convective boundary layer. *J. Atmos. Sci.*, 60(16):1913–1926, 2003.

[183] D. Pino, J. V. G. de Arellano, and S. W. Kim. Representing sheared convective boundary layer by zeroth- and first-order-jump mixed-layer models: Large-eddy simulation verification. *J. Appl. Meteorol. Climatol.*, 45(9):1224–1243, 2006.

[184] T. Platkowski and W. Walus. An acceleration procedure for discrete velocity approximation of the Boltzmann collision operator. *Comput. Math. Appl.*, 39(5-6):151–163, 2000.

[185] S. B. Pope. *Turbulent Flows.* Cambridge University Press, 2000.

[186] N. I. Prasianakis and I. V. Karlin. Lattice Boltzmann method for thermal flow simulation on standard lattices. *Phys. Rev. E*, 76(1, 2), 2007.

[187] N. I. Prasianakis and I. V. Karlin. Lattice Boltzmann method for simulation of compressible flows on standard lattices. *Phys. Rev. E*, 78(1, 2), 2008.

[188] N. I. Prasianakis, I. V. Karlin, J. Mantzaras, and K. B. Boulouchos. Lattice Boltzmann method with restored Galilean invariance. *Phys. Rev. E*, 79(6), 2009.

[189] K. N. Premnath, M. J. Pattison, and S. Banerjee. Generalized lattice Boltzmann equation with forcing term for computation of wall-bounded turbulent flows. *Phys. Rev. E*, 79(2, Part 2), 2009.

[190] R. J. Purser and L. M. Leslie. An efficient interpolation procedure for high-order three-dimensional semi-Lagrangian models. *Mon. Weather Rev.*, 119:2492–2498, 1991.

[191] Y. H. Qian. Fractional propagation and the elimination of staggered invariants in lattice-BGK models. *Int. J. Mod. Phys. C*, 8(4):753–761, 1997.

[192] M. M. Rahman and T. Siikonen. An artificial compressibility method for incompressible flows. *Numer. Heat Transf. B - Fundam.*, 40(5):391–409, 2001.

[193] M. M. Rai and P. Moin. Direct numerical-simulation of transition and turbulence in a spatially evolving boundary-layer. *J. Comput. Phys.*, 109(2):169–192, 1993.

[194] A. Ralston. Runge-Kutta methods with minimum error bounds. *Math. Comput.*, 16:431–437, 1962.

[195] F. Reif. *Fundamentals of Statistical and Thermal Physics.* McGraw-Hil, 1965.

[196] S. Reinecke and G. M. Kremer. Method of moments of Grad. *Phys. Rev. A*, 42(2):815–820, 1990.

[197] M. Rheinlaender. On the stability structure for lattice Boltzmann schemes. *Comput. Math. Appl.*, 59(7):2150–2167, 2010.

[198] P. L. Roe. Characteristic-based schemes for the Euler equations. *Annu. Rev. Fluid Mech.*, 18:337–365, 1986.

[199] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *J. Comput. Phys.*, 135(2):250–258, 1997.

[200] M. Rohde, D. Kandhai, J. J. Derksen, and H. E. A. van den Akker. A generic, mass conservative local grid refinement technique for lattice-Boltzmann schemes. *Int. J. Numer. Methods Fluids*, 51(4):439–468, 2006.

[201] C. Rosales and C. Meneveau. Linear forcing in numerical simulations of isotropic turbulence: Physical space implementations and convergence properties. *Phys. Fluids*, 17(9), 2005.

[202] R. Rubinstein and L. S. Luo. Theory of the lattice Boltzmann equation: Symmetry properties of discrete velocity sets. *Phys. Rev. E*, 77(3, 2), 2008.

[203] T. Ruggeri. Breakdown of shock-wave-structure solutions. *Phys. Rev. E*, 47(6):4135–4140, 1993.

[204] B. Sanderse and B. Koren. Accuracy analysis of explicit Runge-Kutta methods applied to the incompressible Navier-Stokes equations. *J. Comput. Phys.*, 231(8):3041–3063, 2012.

[205] B. Sanderse and B. Koren. New explicit Runge-Kutta methods for the incompressible Navier-Stokes equations. In *Proceedings of the Seventh International Conference on Computational Fluid Dynamics*, 2012.

[206] M. Sbragaglia, R. Benzi, L. Biferale, H. Chen, X. Shan, and S. Succi. Lattice Boltzmann method with self-consistent thermo-hydrodynamic equilibria. *J. Fluid Mech.*, 628:299–309, 2009.

[207] A. Scagliarini, L. Biferale, M. Sbragaglia, K. Sugiyama, and F. Toschi. Lattice Boltzmann methods for thermal flows: Continuum limit and applications to compressible Rayleigh-Taylor systems. *Phys. Fluids*, 22(5), 2010.

[208] P. Schlatter, Q. Li, G. Brethouwer, A. V. Johansson, and D. S. Henningson. Simulations of spatially evolving turbulent boundary layers up to Re_θ=4300. *Int. J. Heat Fluid Flow*, 31(3, SI):251–261, 2010.

[209] F. Schwertfirm and M. Manhart. DNS of passive scalar transport in turbulent channel flow at high Schmidt numbers. *Int. J. Heat Fluid Flow*, 28(6):1204–1214, 2007.

[210] X. W. Shan. Simulation of Rayleigh-Benard convection using a lattice Boltzmann method. *Phys. Rev. E*, 55(3, Part A):2780–2788, 1997.

[211] X. W. Shan and H. Chen. A general multiple-relaxation-time Boltzmann collision model. *Int. J. Mod. Phys. C*, 18(4):635–643, 2007.

[212] X. W. Shan and X. Y. He. Discretization of the velocity space in the solution of the Boltzmann equation. *Phys. Rev. Lett.*, 80(1):65–68, 1998.

[213] X. W. Shan, X. F. Yuan, and H. D. Chen. Kinetic theory representation of hydrodynamics: a way beyond the Navier-Stokes equation. *J. Fluid Mech.*, 550:413–441, 2006.

[214] J. Shen. On error-estimates of projection methods for Navier-Stokes equations - 1st-order schemes. *SIAM J. Numer. Anal.*, 29(1):57–77, 1992.

[215] J. Shen. On error-estimates of some higher-order projection and penalty-projection methods for Navier-Stokes equations. *Numerische Mathematik*, 62(1):49–73, 1992.

[216] F. S. Sherman. A low-density wind-tunnel study of shock-wave structure and relaxation phenomena in gases. technical note 3298, National Advisory Committee for Aeronautics, 1955.

[217] J. W. Shim. Multidimensional on-lattice higher-order models in the thermal lattice Boltzmann theory. *Phys. Rev. E*, 88(5), 2013.

[218] J. W. Shim and R. Gatignol. Thermal lattice Boltzmann method based on a theoretically simple derivation of the Taylor expansion. *Phys. Rev. E*, 83(4, 2), 2011.

[219] C. Shu, Y. T. Chew, and X. D. Niu. Least-squares-based lattice Boltzmann method: A meshless approach for simulation of flows with complex geometry. *Phys. Rev. E*, 64(4, Part 2), 2001.

[220] D. N. Siebert, L. A. Hegele, R. Surmas, L. O. E. dos Santos, and P. C. Philippi. Thermal lattice Boltzmann in two dimensions. *Int. J. Mod. Phys. C*, 18(4):546–555, 2007.

[221] G. Silva and V. Semiao. A study on the inclusion of body forces in the lattice Boltzmann BGK equation to recover steady-state hydrodynamics. *Physica A*, 390(6):1085–1095, 2011.

[222] G. Silva and V. Semiao. First - and second - order forcing expansions in a lattice Boltzmann method reproducing isothermal hydrodynamics in artificial compressibility form. *J. Fluid Mech.*, 698:282–303, 2012.

[223] P. A. Skordos. Initial and boundary conditions for the lattice Boltzmann method. *Phys. Rev. E*, 48(6):4823–4842, 1993.

[224] W. D. Smyth, J. R. Carpenter, and G. A. Lawrence. Mixing in symmetric Holmboe waves. *J. Phys. Oceanogr.*, 37(6):1566–1583, 2007.

[225] W. D. Smyth and J. N. Moum. Length scales of turbulence in stably stratified mixing layers. *Phys. Fluids*, 12(6):1327–1342, 2000.

[226] W. D. Smyth and K. B. Winters. Turbulence and mixing in Holmboe waves. *J. Phys. Oceanogr.*, 33(4):694–711, 2003.

[227] S. W. Son, H. S. Yoon, H. K. Jeong, M. Y. Ha, and S. Balachandar. Discrete lattice effect of various forcing methods of body force on immersed boundary-lattice Boltzmann method. *J. Mech. Sci. Technol.*, 27(2):429–441, 2013.

[228] Z. Sorbjan. Statistics of scalar fields in the atmospheric boundary layer based on large-eddy simulations. Part II: Forced convection. *Boundary Layer Meteorol.*, 119(1):57–79, 2006.

[229] P. R. Spalart. Direct simulation of a turbulent boundary-layer up to R_θ=1410. *J. Fluid Mech.*, 187:61–98, 1988.

[230] P. R Spalart and K. S. Yang. Numerical study of ribbon-induced transition in Blasius flow. *J. Fluid Mech.*, 178:345–365, 1987.

[231] J. D. Sterling and S. Y. Chen. Stability analysis of lattice Boltzmann methods. *J. Comput. Phys.*, 123(1):196–206, 1996.

[232] M. Stiebler, J. Toelke, and M. Krafczyk. Advection-diffusion lattice Boltzmann scheme for hierarchical grids. *Comput. Math. Appl.*, 55(7):1576–1584, 2008.

[233] H. Struchtrup and M. Torrilhon. H theorem, regularization, and boundary conditions for linearized 13 moment equations. *Phys. Rev. Lett.*, 99(1), 2007.

[234] S. Succi. *The Lattice Boltzmann Equation for Fluid Dynamics and Beyond.* Oxford University Press, 2001.

[235] Y. K. Suh, J. Kang, and S. Kang. Assessment of algorithms for the no-slip boundary condition in the lattice Boltzmann equation of BGK model. *Int. J. Numer. Methods Fluids*, 58(12):1353–1378, 2008.

[236] J. Sun and Q. Xu. Parameterization of sheared convective entrainment in the first-order jump model: Evaluation through large-eddy simulation. *Boundary Layer Meteorol.*, 132(2):279–288, 2009.

[237] R. Surmas, C. E. P. Ortiz, and P. C. Philippi. Simulating thermohydrodynamics by finite difference solutions of the Boltzmann equation. *Eur. Phys. J. - Spec. Top.*, 171:81–90, 2009.

[238] P. K. Sweby. High-resolution schemes using flux limiters for hyperbolic conservation-laws. *SIAM J. Numer. Anal.*, 21(5):995–1011, 1984.

[239] A. Tamura, K. Okuyama, S. Takahashi, and M. Ohtsuka. Three-dimensional discrete-velocity BGK model for the incompressible Navier-Stokes equations. *Comput. Fluids*, 40(1):149–155, 2011.

[240] J. R. Taylor. *Classical mechanics.* University Science Books, 2005.

[241] B. Thornber. *Implicit Large Eddy Simulation for Unsteady Multi-Component Compressible Turbulent Flows.* PhD thesis, Cranfield University, 2007.

[242] J. Thuburn. Multidimensional flux-limited advection schemes. *J. Comput. Phys.*, 123(1):74–83, 1996.

[243] L. Tisza. Supersonic absorption and Stokes' viscosity relation. *Phys. Rev.*, 61(7/8):531–536, 1942.

[244] J. Toelke and M. Krafczyk. TeraFLOP computing on a desktop PC with GPUs for 3D CFD. *Int. J. Comput. Fluid Dyn.*, 22(7):443–456, 2008.

[245] T. Tokumasu and Y. Matsumoto. Dynamic molecular collision (DMC) model for rarefied gas flow simulations by the DSMC method. *Phys. Fluids*, 11(7):1907–1920, 1999.

[246] G. Vahala, P. Pavlo, L. Vahala, and N. S. Martys. Thermal lattice-Boltzmann models (TLBM) for compressible flows. *Int. J. Mod. Phys. C*, 9(8):1247–1261, 1998.

[247] J. P. van Doormaal and G. D. Raithby. Enhancements of the simple method for predicting incompressible fluid-flows. *Numer. Heat Transf.*, 7(2):147–163, 1984.

[248] J. van Kan. A second-order accurate pressure-correction scheme for viscous incompressible flow. *SIAM J. Sci. Stat. Comput.*, 7(3):870–891, 1986.

[249] B. van Leer. Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme. *J. Comput. Phys.*, 14(4):361 – 370, 1974.

[250] S. Verma and G. Blanquart. Effects of numerical diffusion and mass conservation errors on turbulent transport of high Schmidt number scalars. In *Proceedings of the Seventh International Conference on Computational Fluid Dynamics*, 2012.

[251] W. Wagner. A convergence proof for bird direct simulation Monte-Carlo method for the Boltzmann-equation. *J. Stat. Phys.*, 66(3-4):1011–1044, 1992.

[252] R. F. Warming and R. M. Beam. Upwind 2nd-order difference schemes and applications in aerodynamic flows. *AIAA J.*, 14(9):1241–1249, 1976.

[253] M. Watari and M. Tsutahara. Possibility of constructing a multispeed Bhatnagar-Gross-Krook thermal model of the lattice Boltzmann method. *Phys. Rev. E*, 70(1, 2), 2004.

[254] K. B. Winters, P. N. Lombard, J. J. Riley, and E. A. Dasaro. Available potential-energy and mixing in density-stratified fluids. *J. Fluid Mech.*, 289:115–128, 1995.

[255] D. A. Wolf-Gladrow. *Lattice-gas cellular automata and lattice Boltzmann models - An Introduction*, volume 1725 of *Lect. Notes Math.* Springer Berlin, 2000.

[256] X. Wu and P. Moin. Direct numerical simulation of turbulence in a nominally zero-pressure-gradient flat-plate boundary layer. *J. Fluid Mech.*, 630:5–41, 2009.

[257] X. H. Wu, R. G. Jacobs, J. C. R. Hunt, and P. A. Durbin. Simulation of boundary layer transition induced by periodically passing wakes. *J. Fluid*

Mech., 398:109–153, 1999.

[258] W. Xian and A. Takayuki. Multi-GPU performance of incompressible flow computation by lattice Boltzmann method on GPU cluster. *Parallel Comput.*, 37(9, SI):521–535, 2011.

[259] S. M. Yen. Monte Carlo solutions of nonlinear Boltzmann equation for problems of heat transfer in rarefied gases. *Int. J. Heat Mass Transfer*, 14(11):1865–1869, 1971.

[260] S. T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *J. Comput. Phys.*, 31(3):335–362, 1979.

[261] G. L. Zanetti. Hydrodynamics of lattice-gas automata. *Phys. Rev. A*, 40(3):1539–1548, 1989.

[262] V. Zecevic, M. P. Kirkpatrick, and S. W. Armfield. The lattice boltzmann method for turbulent channel flows using graphics processing units. In *Proceedings of the 15th Biennial Computational Techniques and Applications Conference, CTAC-2010*, volume 52 of *ANZIAM J.*, pages C914–C931, 2011.

[263] V. Zecevic, M. P. Kirkpatrick, and S. W. Armfield. Stability and accuracy of various difference schemes for the lattice Boltzmann method. In *Proceedings of the 10th Biennial Engineering Mathematics and Applications Conference, EMAC-2011*, volume 53 of *ANZIAM J.*, pages C494–C510, 2012.

[264] Z. Zeren and B. Bedat. Spectral and physical forcing of turbulence. In *Progress in Turbulence III*, volume 131 of *Springer Proceedings in Physics*, pages 9–12, 2010.

[265] H. Zhang. Lattice Boltzmann method for solving the bioheat equation. *Phys. Med. Biol.*, 53(3):N15–N23, 2008.

[266] L. Zheng, B. Shi, and Z. Guo. Multiple-relaxation-time model for the correct thermohydrodynamic equations. *Phys. Rev. E*, 78(2, 2), 2008.

[267] J. G. Zhou. Rectangular lattice Boltzmann method. *Phys. Rev. E*, 81(2, Part 2), 2010.

[268] J. G. Zhou. MRT rectangular lattice Boltzmann method. *Int. J. Mod. Phys. C*, 23(5), 2012.

[269] D. P. Ziegler. Boundary-conditions for lattice Boltzmann simulations. *J. Stat. Phys.*, 71(5-6):1171–1177, 1993.

[270] Q. S. Zou and X. Y. He. On pressure and velocity boundary conditions for the lattice Boltzmann BGK model. *Phys. Fluids*, 9(6):1591–1598, 1997.