UNIVERSITEIT VAN AMSTERDAM

Entropic and multiple relaxation time lattice Boltzmann methods compared for time harmonic flows

Joost Geerdink

11th January 2008

Student number: 0015873
E-mail address: ikku100@gmail.com

GRID COMPUTING — MASTER'S THESIS

Supervisor(s): Dr. Ir. Alfons G. Hoekstra

Entropic and multiple relaxation time lattice Boltzmann methods compared for time harmonic flows

Master's Thesis (*Afstudeerscriptie*)

written by

Joost Geerdink

under the supervision of **Dr. Ir. Alfons G. Hoekstra**, and submitted in partial fulfilment of the requirements for the degree of

MSc in Grid Computing

at the *Universiteit van Amsterdam.*

Date of the public defence:
25th January 2008

Members of the Thesis Committee:
Dr. P. van Emde Boas
Prof. C.R. Jesshope, PhD
Dr. Alfonso Caiazzo
Prof. P.M.A. Sloot, PhD
Dr. Ir. Alfons G. Hoekstra

UNIVERSITEIT VAN AMSTERDAM

Abstract

Contents

Preface

In this Preface I would like to describe how I came about simulating blood flows. Years ago, while studying physics here on the University of Amsterdam, I found myself in an awkward position. Physics could not interest me enough to properly finish my bachelors, let alone masters. Luckily it was possible to do a "vrije kandidaats", which means replacing a third of the courses of the bachelor with courses of some other study, in order to follow some specific direction.

As I was doing some small programming in my spare time, the option of making it a computational physics bachelor was not that far fetched. To find out what my future options for a masters were I spoke, thanks to Thomas Miedema, with Peter Sloot. At that moment he was the director of the Master Grid Computing and he told me about the possibilities of doing physics-related research. This sounded rather interesting and thus I changed my courses accordingly.

After some years I reached the moment at which I had to decide on my masters thesis subject. Although the master was called Grid Computing, my main focus had logically always been computational science and not computer science. My master piece would likewise be in this field and so I asked the around for optional subjects, keeping in mind I had a physics background. Alfons Hoekstra told me about the research his group was doing on blood flow simulations and amongst all the very interesting subjects, this attracted me most. Not only because there was a lot of physics involved but also because real progress could be made by enhancing the models they used. The number and size of the approximations used in their simulations astounded me!

And so I started off reading about the Lattice Boltzmann Model. Many directions for research were possible. Alfons told me they used a rather old model and that new models were probably much better but that their applicability to blood flows had not been tested yet. He also told me this specific subject could prove a daunting task with a smaller chance on success. A real challenge...

Acknowledgements

CHAPTER 1

Introduction

In the medical world the importance of computational simulations is ever-increasing. Models for cancer, aids resistance and entire bodies are created. Researchers are also relentlessly paving the way for blood flow simulations. When a bypass in an artery of the human body has to be made a docter decides, based on experience, when, how and where the bypass needs to be made. This educated guess solves the problem in most cases but sometimes the results are so unsatisfactory that the patient needs another surgery [6, 11, 42].

To lower these chances of failure a calculation of the flow caused by the intended bypass might be made as part of the pre-operative planning. Because of the complexity of this calculation a computer simulation is needed.

Another application is the simulation of aneurismas which are expansions of the artery which locally reduce its strength. When it grows too far it can rupture causing an internal bleeding. To prevent this new procedures including the use of stents are developed and instead of trying these procedures on patients blood flow simulations can be used.

The above two simulations require in general a compute time unsuitable for surgeons and researchers. The accuracy of the calculation is also very important as human lives depend on it. Therefore research is ongoing to find the fastest and most reliable model to simulate blood flows.

1.1 Different approaches

Calculations of blood flow can be done in many different ways [29]. The most elegant approach, namely analytically solving the physical equations for the problem at hand, however, does not even come close to a solution because of the complexity of the problem. Blood flows are non-Newtonian flows with unsmooth systolic pressure profiles with irregular non-static boundary conditions. Even for simple fluids flows in irregular geometries this method proves to be a too great a physical and mathematical challenge (Section 2.1.1).

In the field of Computational Fluid Dynamics the physical equations are solved by discretizing the continuum model and solving the approximations of the equations. This method is however more difficult to implement and in general requires a lot of computation power.

Since the appearance of Cellular Automata (CA) a very simple model for fluids exists: the Lattice Gas Cellular Automata (LGCA) [38]. In these models the fluid is represented by particles moving around on a lattice colliding with each other and bouncing on the wall boundaries. This resulted in very fast simulations of fluids with however large drawbacks [39].

As a response *Lattice Boltzmann Models (LBM)* were developed which, instead of point particles, use density functions to describe the fluid. The physical description of this model is derived from kinetic theory (see section 2.1.7)[39]. The first version was proposed in 1988 and was unable to model in three dimensions due to its complexity (see 2.1.7). Soon after, this and many other problems were solved by removing the direct link between physics and the model, resulting finally (1992) in the BGK (Bhatnagar-Gross-Krook) LBM. [1]

[1] BGK LBM is called LBGK (for Lattice BGK) in the rest of this thesis.

A lot of progress has since been made. The LBGK has been analyzed, tested and refined in many ways [14, 15, 39]. One large problem of the model is however its numerical instability. For some simulations the model becomes unstable and the causes are somewhat complicated. The many simplifications used to leap from kinetic theory to LBGK were the onset of a strong debate, as can be concluded from the many papers cited in this thesis.

In time two streams arose, both claiming to have the solution. The stream which links the instability to the loss of the H-theorem (see Section 2.1.6 and 2.1.7) derived a new lattice Boltzmann method which uses entropy: the Entropic LBM (ELBM). The other stream solves the instability problem by using Multiple Relaxion Times (MRT) in the collision step of the LBM.

Both models have evolved over the last decade and are claimed to be better than the LBGK. Because however blood flows are a special case of fluid flows, the applicability of these two new models has to be tested thoroughly before problem solving environments using the LBGK can be updated with either new model.

The goal of replacing LBGK with either ELBM or MRT is increasing the accuracy, speed and/or useability of the systems. Apart from this the conclusions drawn for the systems used in this paper, namely tubular time harmonic flows, can be of influence in many other research fields like engineering [18].

1.2 From blood to systolic flow

As already mentioned, blood flows are rather complex. To begin with, the vessel diameter is subject to a 5 to 10% change during the cardiac cycle in most major arteries [41]. In most blood flow simulations this is however approximated with a rigid vessel.

Also, the fluid is non-Newtonian, meaning that there is a non-linear relation between pressure and deformation. This property is however not taken into account as a simplification and results in good approximations for large vessels [6].

Blood is neither isotropic nor incompressible but is however approximated with incompressible isotropic models.

The major difference between present day blood flow simulations and other flow simulations is the systolic behaviour. Because of the time dependent pressure and flow rates the system is completely different from static systems. Therefore conclusions drawn for many testcases of the LBGK, MRT and ELBM do not necessarily apply and thus the three models have to be specifically tested for blood flows.

1.3 Aim of this thesis

It has already been shown [6, 7, 8, 9, 10, 11] that the LBGK adequately models specific blood flows.

The same tests should be applied to the new models. In this thesis the quality of the ELBM, MRT and LBGK is measured for time-harmonic flows in order to draw conclusions on their applicability on blood flows. Their relative stability, accuracy and speed will be tested.

The flow used for this comparison is a Womersley flow which is a circular pipe flow with a time harmonic pressure gradient. The flow is simple and its exact analytical solution is known [45] and thus it can be used as a simple benchmark for blood and other time harmonice flows [18].

In order to grasp the fundamentals of the LBM, a rather extensive literature study was needed to leap from computational sciences to fluid dynamics and kinetic theory including its discretization. To reflect this work, to create a starting point for new researchers in the LBM group and because this thesis is supposed to be readable by fellow students, an attempt is made to create an independent description of the Lattice Boltzmann Model. If this is closely read the main concepts as well as the most important steps of its derivation, including their alternatives and restrictions, should be clear.

This work starts with the theoretical description just mentioned, in the first part of the Theory Section (2). With this framework the steps can be taken from the

general LBM or the LBGK to the two new models, ELBM (Section 2.3) and MRT (Section 2.4). Then the more down to earth descriptions of the models follow in the Implementation Section (3). This section also includes a description of the implementation specific choices made and the setup of the experiment. The Results Section (4) gives the results obtained with the necessary graphs, tables and descriptions. In the Discussion Section (5) the, by then, rather large thirst for interpretation and explanation is hopefully quenched, after which the conclusions (Section 6) are drawn and future work is described. In the Appendix some MRT implementation specifics are given.

CHAPTER 2

Theory

2.1 General introduction

In the ever ongoing quest to simplify the world in order to understand it, a lot of effort has been spent on fluid dynamics. The physics behind fluids can be described in multiple ways on different scales. The classical approach shall be summerized first to give the general idea of how fluids work after which a newer theory called kinetics will be explained (starting in section 2.1.2).

2.1.1 Fluid dynamics

The original approach of describing fluids uses the continuous macro scale in which the conserved global quantities, density (ρ), impuls ($\rho\mathbf{v}$, $\mathbf{v}$ being the velocity vector) and energy ($\rho\mathbf{v}^2$) together with the pressure (p) and temperature (T) fully describe the system. Another possibility is the use of for instance the first three variables and two equations of state (EOSs) such that with the EOSs the pressure and the temperature can be calculated. In the system description small regions of space (volumes) are used, so small that no distinction in the description arises when the region size is decreased but large enough such that the amount of particles (atoms, molecules) in the volume can exhibit an equilibrium state. Here the fundamental formula of fluid dynamics, the Navier-Stokes equation, will be derived and explained [33, 35].

It all begins with the conservation of mass. If we have a fixed volume containing some gas, say V, then the total amount of mass inside that volume (m) is given by the integral of its mass density over its space:

$$m = \int_V \rho dV.$$

Now if some gas leaves the volume, the amount of gas leaving the volume is given by the integral of the gas flux over the area surrounding the volume (S):

$$\frac{dm}{dt} = \int_S \rho\mathbf{v} \cdot \mathbf{dS},$$

which equals the reduction of the mass inside the volume ($\frac{d}{dt} \equiv d_t$):

$$\frac{dm}{dt} = -d_t \int_V \rho dv = -\int_V \partial_t \rho dV$$

where use has been made of the property that the region is fixed ($d_t V = 0$), giving

$$\int_S \rho\mathbf{v} \cdot \mathbf{dS} = -\int_V \partial_t \rho dV.$$

This equation can be rewritten by calculating the integrals since the equation must hold for any volume. Using Gauss' divergence theorem gives

$$\int_S \rho\mathbf{v} \cdot \mathbf{dS} = \int_V \nabla(\rho\mathbf{v}) dV = -\int_V \partial_t \rho dV$$

and thus

$$\nabla(\rho\mathbf{v}) - \partial_t \rho = 0. \tag{2.1}$$

This mass conservation equation is usually called the continuity equation.

The next derivation is the momentum conservation. It starts with the equation for the force acting on a volume:

$$\begin{aligned} \mathbf{F}_{total} &= \mathbf{F}_{pressure} + \mathbf{F}_{rest} && (2.2)\\ &= \int_S P\mathbf{dS} + \int_V \mathbf{f}_{rest} dV && (2.3)\\ &= \int_v \nabla P + \mathbf{F}_{rest} dV, && (2.4)\\ & && (2.5) \end{aligned}$$

where again Gauss' divergence theorem is used and all variables are as before. Now the second law of Newton states:

$$\mathbf{F}_{total} = m\frac{d\mathbf{v}}{dt}.$$

Merely plugging in the mass and speed definitions and then expanding the derivative with partial derivatives:

$$\begin{aligned} \mathbf{F}_{total} &= m\frac{d\mathbf{v}}{dt} && (2.6)\\ &= \int_V \rho d_t \mathbf{v} dV && (2.7)\\ &= \int_V \rho(\partial_t \mathbf{v} + \mathbf{v}\cdot\nabla\mathbf{v}) dV && (2.8)\\ & && (2.9) \end{aligned}$$

Since this must hold for any volume, by rewriting, the so-called moment conservation equation can be obtained:

$$\partial_t \mathbf{v} + \mathbf{v}\cdot\nabla\mathbf{v} = \rho^{-1}(\nabla P + \mathbf{F}_{rest}). \tag{2.10}$$

To obtain the Navier-Stokes equation some things have first to be said about the nature of forces in fluids. In the above the force was described as a sum of a force due to pressure gradients ($F_{pressure}$) and some other force (F_{rest}). In fluids however the forces can also be described as forces working on volumes or bodies, and forces working at the surface.

Body forces are those forces that result from a force field which grow as some intensive property like density grows. The simplest example would probably be the gravitational force which doubles when the weight of the object doubles. The other force is the surface force which is due to the contact of one area with another area. These forces are called stresses and are proportional to the size of the contact area. The surface forces can be described with two components: the normal and the shear force. An example of surface forces is friction. The surface forces can be described by

$$\tau_n \equiv \frac{dF_n}{dA}, \tau_s \equiv \frac{dF_s}{dA},$$

with F_n the force normal and F_s tangential to the surface and dA the element of area of the surface. The body force can be expressed as gradients of potentials as they are conservative, e.g.

$$F_{gravity} = -\nabla\Pi,$$

with Π being the gravity force potential.

To describe these surface forces with one variable, a stress tensor (τ_{ij}) is used as two indices are needed to fully describe its behaviour: one index for the direction of the surface on which the force is working and one index for the direction in which the force is working. The surface force per unit volume is then accordingly $\frac{\partial \tau ij}{\partial xj}$.

If now instead of the $F_{pressure}$ and F_{rest} the body/surface force descriptions are used, the following is obtained:

$$\partial_t \mathbf{v} + \mathbf{v}\cdot\nabla\mathbf{v} = \rho^{-1}\frac{\partial \tau ij}{\partial xj} - \nabla\Pi. \tag{2.11}$$

This is again the moment conservation equation and is sometimes called Cauchy's equation of motion.

To be able to continue the stress tensor has to be resolved. If the fluid is static the only surface forces present are due to pressure. The relation between deformation (or strain) and stress depends on the type of fluid. For Newtonian fluids this relation is linear and quantified by the viscosity (ν). Then the stress tensor for static fluids becomes

$$\tau_{ij} = -p\delta_{ij}$$

with δ_{ij} being the Kronecker delta. Any other tensor (with non-zeroes also at the non-diagonal regions) would give a force that would not be purely normal to the surface.

The forces due to non-static forces are then still unresolved and are given the symbol σ_{ij}. Based on arguments related to isotropy, symmetry, Stokes assumtion and tensor properties this σ can be rewritten, ultimately resulting in

$$\tau_{ij} = -(P + \frac{2}{3}\mu\nabla \cdot \mathbf{v})\delta_{ij} + 2\mu e_{ij} \tag{2.12}$$

where μ is the thermodynamic scalar viscosity and

$$e_{ij} \equiv \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i})$$

is the strain rate tensor.

By substituting this latest definition of tau (equation 2.12) into the momentum conservation equation (equation 2.11) the Navier-Stokes equation (NSE) is obtained:

$$\partial_t \mathbf{v} + \mathbf{v} \cdot \nabla \mathbf{v} = \rho^{-1}(\nabla(p + 2\mu e_{ij} - \frac{2}{3}\mu\nabla \cdot \mathbf{v})\delta_{ij}) + \nabla\Pi. \tag{2.13}$$

As already has been stated blood can be described as an incompressible fluid. Using this and that viscosity is a function of temperature which thus becomes a constant as blood is always of some static temperature, the NSE becomes

$$\partial_t \mathbf{v} + \mathbf{v} \cdot \nabla \mathbf{v} = \rho^{-1}(\nabla p + \mu\nabla^2 \mathbf{v}) + \nabla\Pi. \tag{2.14}$$

The NSE gives a relation between pressure, speed and acceleration. When the initial conditions are known the behaviour of the fluid can be described using this equation. Solving this equation can be very complex as it is a non-linear partial differential equation. The field in which computers are used to solve this equation is called Computational Fluid Dynamics (CFD).

2.1.2 Kinetic theory

A completely different approach taken to describe fluids is the one starting as far back as five centuries BC, with Democritus who stated that matter is made up of small particles [13]. This theory has evolved slowly until the 19th century in which the *kinetic theory of gases* was greatly expanded by Maxwell and Boltzmann.

The bottom-up (micro to macro scale) approach, [13, 40], starts at the atomistic level in which the physics are based on Newtonian mechanics:

$$m_i \frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{F}_i,$$

where $\mathbf{x}_i$ and m_i are respectively the position and mass of atom i, and $\mathbf{F}_i$ is the force working on that atom, for N atoms. The speed of each particle, $\mathbf{v}_i$ is also an elementary part of the description of the system. The problem with the above description is that in order to work out the dynamics of a reasonable sized system, these 6N variables (N atoms, three directions in both normal and velocity space) have to be tracked, with N being in the order of Avogadro's number (10^{23}).

The above description can be reformulated by using the phase space function

$$f_N(\mathbf{x}_1, \mathbf{v}_1, \ldots, \mathbf{x}_N, \mathbf{v}_N, t)$$

which is a distribution function describing the probability of finding the molecule one at position $\mathbf{x}_1$ with velocity $\mathbf{v}_1$ *and* molecule two with $\mathbf{x}_2, \mathbf{v}_2$ up to molecule N. Instead of solving for N differential equations the Liouville equation has to be solved:

$$[\partial_t + \sum_{i=1}^{N} \mathbf{v}_i \cdot \partial_{\mathbf{x}_i} + \mathbf{a}_i \cdot \partial_{\mathbf{v}_i}] f_N = 0, \tag{2.15}$$

where $\mathbf{a}_i = \mathbf{F}_i / m_i$ are the intermolecular accelerations. This description in itself does not reduce the size of the problem. However it does allow for integration over parts of the phase space in order to reduce it from a distribution function of N atoms to one of only two atoms.

Two assumptions can now be made, the first being molecular chaos (also called the Stosszahlansatz), which means that particles move independantly prior to collision. The second being that the gas is rare, meaning that chances are negligible on a collision comprising more than two atoms. When these two hold the chance on the state ensemble of two atoms (f_{12}) equals the chance of both states (f_1 and f_2) being true, e.g.

$$f_{12} = f_1 f_2. \tag{2.16}$$

This fact can then be used to reduce the Liouville equation to an equation with a one particle distribution function, the Boltzmann equation (BE) (see [13] for details):

$$\partial_t f + \mathbf{v} \cdot \partial_{\mathbf{x}} f + \mathbf{a} \cdot \partial_{\mathbf{v}} f = C(f, f) \tag{2.17}$$

It states that the rate of change in f depends on the collision integral C and on the acceleration due to external forces ($\mathbf{a}$). The collision integral expresses the change in f due to two body collisions.

The rate of change can also be written as:

$$\partial_t + \mathbf{v} \cdot \partial_{\mathbf{x}} + \mathbf{a} \cdot \partial_{\mathbf{v}} = \mathscr{D}, \tag{2.18}$$

which simplifies the BE into:

$$\mathscr{D} f = C(f, f). \tag{2.19}$$

2.1.3 Collision invariants

Average properties $\overline{\phi}$ of the fluid can be calculated ([14]) by taking the integral over (velocity) space-time of the distribution function $f = f(\mathbf{x}, \mathbf{v}, t)$ times the sought for property $\phi = \phi(\mathbf{x}, \mathbf{v}, t)$. If the property is taken as the average quantity at $\mathbf{x}, d\mathbf{x}$ and at time t, dt then the integration becomes

$$\overline{\phi} = \int f \phi d\mathbf{c} \tag{2.20}$$

Taking ϕ as a one term polynomial of the speed $\mathbf{c}$ is called taking its moment. For instance the integral

$$\text{first moment} = \int \mathbf{c} f d\mathbf{c} \tag{2.21}$$

Some properties of fluids are collision invariant. In order to derive these properties the BE (2.17) is multiplied by a general property ϕ and integrated:

$$\int \phi \mathscr{D} f d\mathbf{c} = n \Delta \phi, \tag{2.22}$$

with

$$n \Delta \phi = \int \phi \frac{\partial_e f}{\partial_t} d\mathbf{c}. \tag{2.23}$$

Obviously $\Delta\phi$ stands for the change in ϕ, while $\frac{\partial_e f}{\partial t}$ is the rate of change in f due to collision, and thus $n\Delta\phi$ gives the rate of change of property ϕ.

By rewriting the collision integral and using the rate of change of properties it can be shown (see [13]) that there exist moments of the distribution function which are collision invariant:

$$I(v) = [1, \rho \mathbf{v}, \rho \mathbf{v}^2], \tag{2.24}$$

which simply states that mass, momentum and energy are conserved by the collision operator.

2.1.4 Hydrodynamic and kinetic subspaces

In kinetic theory collision invariants are very important. They are those physical quantities in the system that are unchanged by collision and henceforth represent the physical part of the distribution function. The part of the distribution function that dissipates, and finally disappears, is called the kinetic part. It is precisely the non-equilibrium part of the distribution function. In other words, the collision operator works only on the kinetic part of the distribution function. The other part is called the hydrodynamic part as it consists of the hydrodynamic variables, namely density, momentum and energy.

For applying vector calculus on kinetic theory it is worthwhile to distinguish two parts in the phase space, the hydrodynamic ($\mathcal{H}$) and the kinetic ($\mathcal{K}$) subspaces. These two together span the entire phase space. The collision operator works only in the kinetic space, since it maps hydrodynamic vectors onto themselves. In general the basis vectors of the two spaces will be orthoganol to each other. The basis vectors in one subspace do not necessarily have to be orthoganol to the other basis vectors of that same subspace.

2.1.5 Equilibrium distribution functions

Going from the Boltzmann equation to hydrodynamic equations, for instance the Navier Stokes, local equilibria are needed [39]. These are defined such that collisions do not change the shape of f:

$$C(f^{eq}, f^{eq}) = 0, \tag{2.25}$$

which leads to the *detailed balance* condition:

$$f_1' f_2' = f_1 f_2, \tag{2.26}$$

merely stating that the ensemble of the two distribution functions (f_1', f_2') after collision equals the ensemble of the two distribution functions prior to collision (f_1, f_2).

The equilibrium distribution functions can now be obtained in various ways. If we take the logarithm of this condition, we get:

$$\ln f_1' + \ln f_2' = \ln f_1 + \ln f_2, \tag{2.27}$$

which shows us that $\ln f$ is an additive collision invariant. Since we know the physical properties that are collision invariant, it follows that $\ln f$ must be a function only of those quantities:

$$\ln f = [A, B\rho\mathbf{v}, C\rho v^2], \tag{2.28}$$

for $f = f^{eq}$. These five constants can be obtained by using the moment equations and matching them to the physical properties:

$$\rho = \int f d\mathbf{c} \tag{2.29}$$

$$\rho\mathbf{v} = \int \mathbf{c} f d\mathbf{c} \tag{2.30}$$

$$\rho\mathbf{v}^2 = \int c^2 f d\mathbf{c}^2 \tag{2.31}$$

Using quadrature then results in the (global) Maxwell-Boltzmann equilibrium distribution:

$$f^{eq} = \rho(2\pi v_T^2)^{-D/2} e^{-c^2/2v_T^2}, \tag{2.32}$$

where D, is the dimension, T the temperature, $v_T = \sqrt{\frac{K_B T}{m}}$ the thermal speed, K_B the Boltzmann constant, m the mass of one particle and c the speed of the particles relative to the motion of the fluid. This function is translational and rotational invariant.

There also exist local equilibria. In fact, in order for a global equilibrium to be generated the local equilibria have to grow into one big equilibrium. The physical picture is this: on very small length scales atoms collide very frequently and thus on

these short ranges fluids can reach an equilibrium on a small time scale. As time goes by, the local properties of the fluid diffuse and advect, causing wider range equilibria to settle in. Finally, for slowly evolving flows, a global equilibrium is reached.

In order to check if this micro scale description of fluids is correct, the Navier-Stokes relation and the conservation equations have to be obtained from it. This can be done using the Chapman-Enskog method which approximates the variables with an expansion. These approximations are then substituted into the BE and solved. From this the conservation equations follow. Then by finding the equilibrium distribution function f^{eq} it can be checked if the Navier-Stokes relation holds.

The Chapman-Enskog expansion approximates f with a taylor series around f^{eq} with expansion parameter K_n. This is the Knudsen number and relates to the speed at which local equilibria settle in, and is defined by:

$$K_n = \frac{l_{mfp}}{l_M}, \tag{2.33}$$

where l_{mfp} is the particle mean free path length and l_M is the macroscopic length scale of the system. In other words, the expansion is around an equilibrium of a static fluid.

The approximation of the probability distribution function is as follows:

$$f \approx f^{(0)} + K_n f^{(1)} + K_n^2 f^{(2)} + \cdots + K_n^m f^{(m)} + O(K_N^{m+1}) \tag{2.34}$$

These functions are substituted into the BE and, starting at the lowest order, order by order they are solved such that the BE holds. Calculating the moments of the different orders then gives back the conservation equations.

2.1.6 H-Theorem

The former subsection gave a short introduction to the continous Boltzmann equation and summarized its meaning and the method which proves its correctness. There is however one big gap and that is the mechanism driving the distribution functions to equilibrium. Why would the collisions point into the direction of this local equilibrium? The question can be rephrased as: why is there a time direction, at all, in the kinetic mechanics?

The well known, but elusive, answer is that from Boltzmanns H-theorem, which roughly states that entropy does not decrease in time. In the case of a fluid this means that the energy diffuses and thus that collisions will tend to a certain equilibrium. The H-function is given by:

$$H(t) = \int f \ln f d\mathbf{v} d\mathbf{x}, \tag{2.35}$$

with the H-theorem as:

$$\frac{dH}{dt} \leq 0. \tag{2.36}$$

The equality is only true at equilibrium. In these states the indiviual particles may change in space and velocity, but the distribution funtion as a whole remains unaltered.

2.1.7 Problem size reduction

Solving the macroscopic NSE (2.14) has proven to be one of the hardest mathematical problems in physics, still left to be solved for the more complicated geometries. Using instead the micro scale for solving the problem is however unfeasable since the number of particles for any reasonable sized problem is, as already mentioned, far too large.

So instead of solving the problem on either scale, an intermediate scale is looked for. The first step is to use fictitious particles, each representing a bunch of atoms or molecules. This can be done by discretizing space onto a lattice, where each site represents a volume in space. In the continuous picture this is represented by the step from many Newtonian equations to one Liouville equation ensemble and then to one Boltzmann equation. These two steps reduce the problem significantly, however solving the BE is not that simple.

A second step can be taken to further reduce the state space. This is done by discretization in velocity space which means that instead of having any speed, particle speeds are in a limited, discretized, range. Possible speeds are those that fit onto the lattice, e.g. the particles can only move from lattice site to lattice site.

The speed and space-time discretizations together result in a new set of functions, one for each possible velocity, at each lattice site:

$$f = f_i(\mathbf{x}, t), \{i = 1, \ldots, b\}, \tag{2.37}$$

for b different velocities at site $\mathbf{x}$ at time t. These are called the velocity distribution funtions. Note that $\mathbf{x}$ is a discrete location in space and t is also discrete. In the Quadratures section, 2.2, above discretizations are explained in more detail.

The Boltzmann equation (2.17) now becomes the Lattice Boltzmann Equation (LBE):

$$f_i(\mathbf{x} + \mathbf{v}_i, t + 1) = f_i(\mathbf{x}, t) + C_i[f_i, \ldots, f_b], \tag{2.38}$$

where C_i is the collision operator as a polynomial function of degree b. This collision operator describes the effect of collision at lattice site $\mathbf{x}$ as a function of the one-body b-velocity distribution function. Since all the b parts can play a role in this, the collision operator is a function of b variables, which can always be reformulated as a polynomial of b variables.

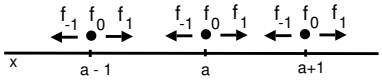

Figure 2.1: A simple example of a one dimensional lattice, with at each lattice site three densities, one for each velocity.

The LBE can best be explained with a simple example. If we take a one dimensional lattice, with a lattice spacing of one, and the set of possible velocities is given by: $\mathbf{v} = (-1, 0, 1)$, then each lattice node has three densities, f_{-1}, f_0, f_1. The LBE then gives for instance $f_1(a+1, t+1) = f_1(a, t) + C_2[f_{-1}, f_0, f_1]$, see figure 2.1. Thus the new density speed right-moving on lattice site $a + 1$ (left part of the equation) equals the old density speed right moving of its left neighbour site plus the possible appearing or disappearing speed density resulting from collisions at its left neighbour.

Collision operator The next subject of the reduction of the problem size is the collision operator. In general, the collision operator of the LBE is a polynomial of degree b of all the f_i at a lattice site. Since the number of velocities b is quite large for most higher dimensional models, using a high degree polynomial will take a lot of computation time. The physical reason of existance for the collision operator is clear, however its role in the model can be recast onto a more practical shape. What the collision operator actually does is pushing the velocity distribution functions in the direction of equilibrium. A tremendous accomplishment ([39] chapter 3.2) in the lattice Boltzmann theory was the insight that this collision operator could be transformed into a linear operator.

The whole trick of transforming the polynomial collision operator into a linear one, is using the Chapman-Enskog approximation for the description of the collision operator itself. Since the equilibrium part of the distribution functions will not equilibrate any further, this part remains unaltered. Hence, the collision operator works only on the non-equilibrium part. The assumption behind the linearization is that the fluid is close to equilibrium. Thus the collision operator can be described as a linear operator working on the difference between the velocity distributions and their equilibrium parts:

$$C_i[f_i, \ldots, f_b] \Rightarrow L[f_i - f_i^{eq}]. \tag{2.39}$$

There are many consequences of this linearisation. The positive side is mainly the gigantic increase in simplicity and hence the simulation speed of the model. On the negative side there is the loss of the second law of thermodynamics, e.g. the H-theorem does not hold anymore (see, for instance, [39] page 48). As a result the distribution functions can become negative which leads to instabilities.

The second law of thermodynics is lost because a certain equilibrium distribution is taken as the general form of all equilibrium distributions in the system [39]. Instead of letting the fluid equilibrate by itself due to collisions, an equilibrium is predefined and the fluid is pushed towards it. This predefined equilibrium is a polynomial approximation of the "real" equilibrium, around zero speed (see again section Quadratures, 2.2), which means that it will break down at speeds far away from zero. Thus, whenever the fluid moves with a high speed, or more correctly, has high Knudsen numbers, the direction of relaxation using the predefined equilibrium might be wrong. As a result the fluid might actually move away from its correct equilibrium and thereby increase its H-value.

A different point of view on the loss of the H-theorem is that discretization errors can occur when the continuous representation is lost, and since no explicit H-theorem is imposed on the discretization there is no reason to believe it still applies.

Operator L can be given as a matrix (since it is a linear operator working on vectors) and the first derivation relied on approximating C. However L can also be constructed from scratch in such a way that it retains as much physical laws as possible. If then the different mode speeds at which equilibrium is attained are all set to one speed (e.g. using one relaxation parameter, see also [3]), the matrix L can be diagonalised causing again an increase in simplicity. This step result in the Lattice Bhatnaga-Gross-Krook (LBGK) model, with its main step:

$$L_i[f_i - f_i^{eq}] \Rightarrow -\omega\delta_{ij}(f_i - f_i^{eq}) = -\omega(f_i - f_i^{eq}), \tag{2.40}$$

where $\omega = 1/\tau$ is the inverse relaxation time and τ is the single relaxation time.

As a result, the LBE (2.38) becomes:

$$f_i(\mathbf{x} + \mathbf{v_i}, t + 1) = f_i(\mathbf{x}, t) - \omega(f_i - f_i^{eq}), \tag{2.41}$$

Note that with this construction of the collision matrix, as opposed to above derivation, no relation to entropy exists whatsoever. The construction of the equilibria takes into account the conservation laws but does not depend on some H-function.

2.2 Quadratures

In the LBGK model, which can be called the final version of the classical Lattice Boltzmann Methods, the collision operator became a simple linear operator with its elements laid down by the physical laws it should uphold. A different angle from which to attain a LBM schema [37, 39] is the continuous, algebraic, kinetic description of fluid mechanics. In this framework there is also the BGK collision operator, but now it works with a continuous (velocity) space. To derive a discrete model a quadrature can be used. In general:

$$\int_a^b \omega(x)f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i), \tag{2.42}$$

where $\omega(x)$ is some weight function, $f(x)$ is the function that will be discretised, x_i are the nodes at which the sum has to be evaluated and w_i are the appropriate weights. The velocity modes can now be approximated, with e.g. the first order, momentum, equation becoming:

$$\rho\mathbf{v} = \int \omega\mathbf{c}fd\mathbf{c} = \sum w_i f_i. \tag{2.43}$$

Many weight functions can be used and their errors can be calculated. As a result of this method discretisation can be done as exact as need be. For some scientists this shows the correctness of LBGK as this description closely resembles the continous one.

The weight function used is usually e^{-x^2}, its associated polynomials are Hermitian. What these quadratures boil down to is the remapping of the state space function f. In the following such a remapping will be explained based on reference [37].

Since the Chapman-Enskog derivation depends only on some lower modes of the state function, keeping these modes intact in an approximation will make it possible

to derive the NSE. Because quadratures are exact up to some order, depending on the order of the polynomial used in the approximation, quadratures can be used to approximate the state function while upholding the NSE.

If the weight function is defined by:

$$\omega(\mathbf{v}) = \frac{e^{-(1/2\mathbf{v}^2)}}{(2\pi)^{D/2}}, \tag{2.44}$$

with D the dimension and if f is some nicely integrable function then the following approximation is valid:

$$f(\mathbf{x}_i, \mathbf{v}_i, t) = \omega(\mathbf{v}_i) \sum_{n=0}^{\infty} \frac{1}{n!} a_i^{(n)}(\mathbf{x}, t) \mathcal{H}_i^{(n)}(\mathbf{v}), \tag{2.45}$$

where $\mathcal{H}^{(n)}$ is the nth order Hermite polynomial, $\mathcal{H}, a$ are both tensors of order n and the multiplication is a contraction of order n. The coefficients a can be calculated from:

$$a^{(n)} = \int f \mathcal{H}^{(n)}(\mathbf{v}) d\mathbf{v} \tag{2.46}$$

As already mentioned, the quadrature of a function is exact up to some order, with the order as a function of the order of the polynomial used. For moment calculations with a Hermite expansion the approximation is exact when at least the degree of the moment is used in the approximation. Hence the function 2.45 can be be truncated at the order of the highest moment. With some tricks the weight function a can be approximated too and the fluid variables then become

$$\rho = \sum_{i=1}^{d} \frac{w_i f_i}{\omega(\mathbf{v_i})} \tag{2.47}$$

$$\rho \mathbf{u} = \sum_{i=1}^{d} \frac{w_i f_i \mathbf{v}_i}{\omega(\mathbf{v_i})} \tag{2.48}$$

$$2\rho\epsilon + \rho u^2 = \sum_{i=1}^{d} \frac{w_i f_i v_i^2}{\omega(\mathbf{v_i})} \tag{2.49}$$

where w_i and v_i are the specific weights and nodes for a quadrature of order $2N$. If now an auxiliary variable is defined as $g_i = w_i f_i / \omega(\mathbf{v_i})$ then the above equations become the LBM equations for the modes:

$$\rho = \sum_{i=1}^{d} g_i \tag{2.50}$$

$$\rho \mathbf{u} = \sum_{i=1}^{d} g_i \mathbf{v}_i \tag{2.51}$$

$$2\rho\epsilon + \rho u^2 = \sum_{i=1}^{d} g_i v_i^2. \tag{2.52}$$

This g function takes the place of the classic f. The nodes of the quadrature take over the symbolic meaning of the discrete velocity set v_i.

Next, the equilibrium density function has to be approximated. This then results in a discrete description of a continuous problem. There are some difficulties with this method, the biggest being the unsolved problem of finding the most optimal quadrature with a minimal number of nodes for a set system. Some solutions are believed to be minimal without proof. These solutions determine the degree of the polynomial and its nodes. In general these nodes do not fit on a regular lattice and as a result a finite difference scheme has to be used.

The positive side of the story is that this method encapsulates the classic LBM and is able to refine the discretisation scheme of the theory. It provides a mathematical foundation for the LBM and its exactness and stability can be improved.

A well known example of using quadratures to approximate the solution of the BE is the Grad's 13 moment method. The solution is assumed to be a function of 13

moments which represent density, speed (3), temperature, stress (5) and heat flux (3). The nodes of the quadrature relate to the physical quantities of the system and thus belong to the hydrodynamic space. The polynomials are Hermite tensors. A problem with Grad's method is that it can not easily be extended to polyatomic or degenerate gases [32]. Another problem is its limited Mach number range, since for higher Mach numbers the method does not converge, resulting in not 13, but an infinite amount of moments needed for approximating the phase space [14].

2.3 ELBM

In the LBM derivation (section **??**) the world is simplified step by step without breaking the continuity equations. The H-theorem is however lost somewhere in the process (paragraph 2.1.7). It was never even defined as such. Of course the principle behind it, namely energy dissipation and the general move of the distribution functions to equilibrium, *is* applied. In the LBE the new distrubution ($f(t+1)$) will in general be closer to its equilibrium (f^{eq}) than the old distribution ($f(t)$).

Whenever simulations contain discontinuities, like sharp edges on water riples, instabilities may occur. This has to do with the ability of the model to map the physical geometry to its discretised world. Decreasing the (space and/or time) discretisation stepsize may solve this problem, but will also increase the runtime of the simulation. When the edge itself is not of interest to the scientist, for instance when they occur merely temporary in the simulation as the result of a discretisation error, handling these edges in a different way might prove beneficial.

As already mentioned in the introduction, the Entropic Lattice Boltzmann method (ELBM) imposes the H-theorem on the LBM with the goal of attaining a higher stability. The trick of using the H-theorem was, to the author's knowledge, applied for the first time in 1996 by A. Gorban and I. Karlin [24] where a Grad-like description of state (having to do with quadratures, section 2.2) was created. A next attempt was more focussed on using some form of entropy and then creating appropriate equilibrium functions. This was done in 1998, [31], and from this article more ELBM theory threads spawned.

ELBM is evolving continuously (the latest paper for instance [17], when writing this article, throws the whole prospect of the Gauss-Hermite quadrature out of the window) and as no final conclusions can be drawn as to which version is the clearest or most stable, accurate or elegant, the contents of this section will be a summary of the ELBM as a whole. The general idea will first be explained, including some of its more important implications. Then some examples and a short summary of the conclusions drawn by various scientists on this work-in-progress theory will be given.

2.3.1 The basics

Most ELBM start with some H-function after which the equilibrium distribution functions are derived. Some however start with an equilibrium distribution, like Boltzmann's, and derive the H-function. Here the former approach is taken. The most general form for the discrete H-function is:

$$H = \sum_{i=1}^{N} h(f_i), \tag{2.53}$$

where h is a concave function. While evolving the system will never increase its H-value, therefore as long as h is concave there exists only one final solution to the problem. A well known example is the Boltzmann h:

$$h(f) = f \ln f \tag{2.54}$$

Some of the other functions that have been used are $h(f) = f^2$ [31], $h(f) = f\sqrt{f}, h(f) = -\ln f$. The latest version, [16, 17, 30, 40] is

$$h(f_i) = f_i \ln(\frac{f_i}{w_i}) \tag{2.55}$$

in which each f_i can have its own weight w_i.

Since we know from Section 2.1.5 the formulas for the collision invariants (equations 2.29) and the defining property of the equilibrium functions being that they minimise the H-function, they can now be derived. To put it in other words, the equilibrium distribution function f^{eq} is derived as the minimiser of the H-function 2.53 under the hydrodynamic constraints 2.29.

The derivation can be performed in many ways. One is via the usage of Lagrangian multipliers (see, for instance, [16]). Here the solution is written as follows:

$$f_i^{eq} = w_i \chi \zeta_x^{cix} \zeta_y^{ciy} \zeta_z^{ciz}, \tag{2.56}$$

where the Langrangian multipliers $\chi(\rho, \mathbf{v}), \zeta(\rho, \mathbf{v})$ are derived by substituting 2.56 into 2.29. The weights w_i are calculated using a low speed approximation for the NSE, similar to a Chapman-Enskog derivation.

A different derivation of the equilibrium distribution functions makes use of the division of the state space into a hydrodynamic ($\mathcal{H}$) and a kinetic subspace ($\mathcal{K}$) (see section 2.1.4 and, for instance, [2, 30]). Using this description the equilibrium distribution functions can be derived from the orthogonality of these functions to the kinetic subspace, when combined with the conservation equations. This translates to:

$$\nabla H|_{f^{eq}} \in \mathcal{H}, \tag{2.57}$$

$$\nabla H|_{f^{eq}} \perp \mathcal{K}, \tag{2.58}$$

after which the equilibrium can be found using vector calculus. Here ∇H is the gradient of H.

Once the equilibrium distribution function and the H-function are known, there is one last part in the LBM that needs to be changed. Since now the H-function is defined, the H-theorem can be applied to the model. This states that over time the H-function should never increase in value. The theorem, in which the H-value (that value of the H-function belonging to the f) of the new distribution equals or is smaller than the H-value of the distribution prior to collision, translates to:

$$H(f_{new}) \leq H(f_{old}). \tag{2.59}$$

Substituting f_{new} by using the equation for the evolution of the distribution function, the LBGK (2.41), gives:

$$H(f_{old} - \omega(f_{old} - f_{eq})) \leq H(f_{old}). \tag{2.60}$$

The equality will in general only hold when the system is at equilibrium. The H-theorem can be broken whenever ω is too large, causing the new H-value to be larger than the previous one. At these moments the physics of the model are broken. To solve this a new parameter α is added to the model. The LBGK becomes

$$f_i(\mathbf{x} + \mathbf{v_i}, t + 1) = f_i(\mathbf{x}, t) - \alpha\omega(f_i - f_i^{eq}), \tag{2.61}$$

which is reflected in the H-theorem equality as:

$$H(f_{old} - \alpha\omega(f_{old} - f_{eq})) = H(f_{old}). \tag{2.62}$$

The maximum value, α_{max} of the parameter α can now be calculated for each update. When alpha is larger than α_{max}, the entropy of the system decreases and the physics break down. When alpha is smaller the distribution functions approach their equilibrium values at a slower pace, which means that the system relaxates slower. However, the H-value will nowhere increase and thus the model does *not* break the second law of thermodynamics.

One might assume that the viscosity and hence the final distribution will change by the addition of the α parameter. This is however not the case as α approaches 1 as the system approaches equilibrium. This can easily be seen from equation 2.62. What does change is the way in which the system relaxates to this equilibrium. Whenever sharp edges in the distribution occur, instabilities are bound to occur. However, by demanding that the H-value will decrease the edges can not grow in sharpness. This does result in a slower relaxation as the update will over-relaxate less due to the lower α.

It should be realised that in the normal LBGK model, the update rule over-relaxates the distribution from its old value to a value that is not somewhere in between its old value and the equilibrium but rather on the other side. In a way the distribution function overshoots its equilibrium value. The result is a faster convergence of the model but appearently the trade-off is breaking the H-theorem.

2.3.2 Transformations

Bare collision integrals based on the kinetic subspace

When the subspace description of states is used, the whole mathematical framework can be transcribed. Since the collision operator leaves the hydrodynamic part of the state space unaltered, only the kinetic part is needed in the collision update rules. The result is that the equilibrium functions are not required in the mathematical derivation of the viscosity or in the simulation itself. To keep things simple, a one dimensional example from paper [2] will be given here.

The space is constructed as a one dimensional lattice with C length between the nodes. The velocity set consists of $(c_-, c_0, c_+) = (-C, 0, C)$ and each node has thus three population densities (N_-, N_0, N_+). An entropy function that is good for this system (concave, defined on all points and resulting in the NSE correct up to the fourth Mach order) is:

$$H = N_0 \ln(\frac{N_0}{4}) + N_- \ln(N_-) + N_+ \ln(N_+), \tag{2.63}$$

which is of the form of the latest H-function (eq. 2.55).

Now a "bare" collision integral is specified such that the rules 2.57 and 2.58 and the H-theorem hold. Such integrals are also called admissible. One possibility is the following:

$$\Delta = (g^+ - g^-) < \nabla H, g^- - g^+ >, \tag{2.64}$$

where $g^+ = (1, 0, 1)$ and $g^- = (0, 2, 0)$ are the positive and negative parts of the vector $g = g^+ - g^-$ which lies in the kinetic subspace $\mathcal{K}$. Then the update rule becomes

$$N(x + \mathbf{c}, t + 1) - N(x, t) = \Delta^*(N(x, t)), \tag{2.65}$$

where Δ^* is the (dressed, or) stabilised collision integral

$$\Delta^*(N) = \beta\alpha(N)\Delta(N), \tag{2.66}$$

with β a relaxation parameter related to viscosity and α the stabilisation parameter which guards the H-theorem. As a result, the bare collision operator presents the direction in which the populations move, and the β, α determine the step size.

In order to perform an update for a node its equilibrium values are no longer required. Instead only the inproduct of the entropy gradient and the kinetic subspace vector have to be calculated and the α determined. Note however that no three dimensional versions of model are known to the author.

Moment methods and quadratures

In the Quadratures section (2.2) the discretisation of the continuous model was attained by using Hermite quadratures starting with the linearized collision operator. The original Grad moment method is however unable to supply an entropy since the moments can not both be used as a measure for the entropy and as a function to balance the entropy at the same time [32].

Therefore the method has been reviewed. It is possible to ignore the whole LBGK path from the continuous Boltzmann equation (BE, 2.17) to a LBM and instead start again at the BE of section 2.1.2. This time however, a low Mach number expansion is first applied to the Maxwell-Boltzmann equilibrium distribution (2.32) after which the Grad-like quadrature is used to derive the lattice structure and the equilibrium distribution functions. As a result H-theorem does not have to be reinstated because it has not been lost during the process and the approximation of the NSE can be done up to any desireable order [26]. This method uses the same mathematical procedure

as given in the Quadratures section but applied slightly different. Hence it will not be repeated here.

The results of this method are claimed [17] to be merely a subset of the results obtained via another path. Instead this derivation of the lattice structure and the equilibrium function starts with the entropy function with to-be calculated weights. Then a series expansion is again used to derive the equilibrium functions which are in turn used for the calculation of the constitutive relations that result from the Maxwell-Boltzmann equation. The order of accuracy of these relations dictates the level of accuracy of the complete model. As these relations are known in the continuous case, the accuracy can be calculated as a function of the weights. It turns out that some of the sets of velocities which in turn dictate the weights, match the sets obtained from other models.

2.4 MRT

Where the ELBM solves the instability problem by reinstating the H-theorem, the Multiple Relaxation Time method (MRT) does this by approximating the non-linear collision operater with more than one relaxation time in the linear collision operator. The effects of having only one relaxation speed are plentifull as for instance the Prandtl number (a measure for the ratio of momentum diffusivity (viscosity) and thermal diffusivity) and the ratio of the bulk and kinematic viscosities are fixed [21].

2.4.1 Moments

The phase space of the LBGK is also used in the MRT with the same possible definitions for space and time discretisations (see 2.1.7). What is added is a transformation of this space into a so-called moment space where each dimension contains its own moment. These moments can be calculated as functions of the moments described in subsection 2.1.3. Now the collision operator is applied in moment space instead of the normal phase space and the authors of [34] claim that the reason for this is "somewhat obvious". Because the physical processes in kinetic theory can be approximately be described as interactions between the hydro and kinetic moments, or likewise, these new moments, the collision process can be better understood when these moments themselves are used. The result is that the relaxation parameter of each separate moment can be defined and/or calculated. This is usefull for making the leap from theoretical model to physical system.

Since a transformation of the phase space is used, it is possible to create a collision operator having the transformation vectors as its eigenvectors. The result is that the collision operator becomes a linear operation with precisely one relaxation parameter for each moment. What now follows is a more theoretical description of MRT [20, 21, 34].

2.4.2 Transformation

In the LBGK the phase space consists of velocity distributions $\{f_i(\mathbf{x},t)|i=0,1,...,b-1\}$, b being the number of different velocities used to describe the velocity space $\{e_i|i=0,1,...,b-1\}$. To obtain the moments a set of vectors ϕ must be defined which can be derived by the Gram-Schmidt orthogonalization procedure from polynomials of the speed vectors $\{e_d|d=0,1,...,D\}$ where D is the dimensionality. These vectors are simply multiplied with the velocity distributions to obtain the moments $\{m_i(\mathbf{x},t)|i=0,1,...,b-1\}$:

$$m_i \equiv \phi_{\mathbf{i}} \cdot \mathbf{f} \tag{2.67}$$

These moments represent the moment distribution in moment space $\mathbb{M}$ spanned by $\phi_{\mathbf{i}}$ obtained via the above transformation from the velocity distribution $\mathbf{f}$ in velocity space spanned by $\mathbf{e_i}$. When the set of $\phi_{\mathbf{i}}$ are compiled into a matrix $\mathbf{M}$ this transformation can also be described as

$$\mathbf{m} \equiv \mathbf{M}\mathbf{f} \tag{2.68}$$

To continue the explanation it is best to use the example of a typical athermal model. In such a model the only conserved quantities are the density and

the moments [speed times density]. A D2Q9 model could have for instance $\mathbf{m} = \{\rho, e, \epsilon, j_x, q_x, j_y, q_y, p_x x, p_x y\}$ with the "physical" significance of respectively density, energy, some function of energy squared, x momentum, x energy flux, y momentum, y energy flux, diagonal and off diagonal component of the stress tensor [34].

Then the equilibrium function for each moment can be written as a function of the conserved quantities. There are different ways to derive these functions, one option is based on the physical significance of the moments and another is to use the definitions of the moments. The only conditions are the conservation of isotropy and Gallilean invariance.

Next the collision operator $\mathbf{L}$ can be defined. Similar to the argument in paragraph 2.1.7, the only conditions for the actual value and shape of $\mathbf{L}$ are the upholding of the NSE in as high orders of accuracy as possible. Since the collision for MRT is performed in the moment space it is usefull to have a diagonal matrix as collision operator in moment space $\mathbb{M}$. This can easily be obtained if the $\phi_\mathbf{i}$ are the eigenvectors of the matrix $\mathbf{L}$. Then the collision matrix in moment space is defined by $\hat{\mathbf{L}} = \mathbf{M} \cdot \mathbf{L} \cdot \mathbf{M^{-1}}$ and now the LBE (2.38) becomes

$$f_i(\mathbf{x} + \mathbf{e_i}, t + 1) = f_i(\mathbf{x}, t) - \mathbf{M^{-1}}\hat{\mathbf{L}}(m_i - m_i^{eq}). \tag{2.69}$$

Since the diagonal matrix $\hat{\mathbf{L}} \equiv diag(l_0, l_1, ...l_{b-1})$ gives the linear relaxation parameter per moment (l_i) and moments are easier to interpret than velocity densities, the derivation of the separate parameters is usually done in moment space. The biggest difference between LBGK and MRT lies in this matrix: for LBGK all the relaxation parameters are equal and set to some function of the viscosity whereas for MRT the parameters can vary.

To derive the relaxation parameters the velocity set and the equilibrium functions have to be chosen. When this is done the Chapman-Enskog method can be used. However, when using the moments instead of the velocities the standard Von Neumann stability analysis [34] can instead be used where the approximation is not done in speed but in wave vector k. First the transformation into Fourier space is made. Then a general solution for the moments, periodic in space and time, is chosen and substituted in the evolution equation together with the general solution for the evolution equation. From this a dispersion relation between the fourier transform of the evolution operator and wave vector k follows. This dispersion relation can then be solved by setting the global constant flow of the fluid at some value and expanding the relation around some k. The result is that the relaxation parameters can be optimised for maximum isotropy and Galilean invariance. Note however that this procedure uses periodic boundaries in all directions and does not necessarily produce the optimal relaxation parameters. It remains an approximation and it is possible for some typical fluid to be more sensitive for some higher orders of the modes which are damped more slowly in this specific solution than in some other solution.

CHAPTER 3

Implementation

So far the more theoretical side of the models has been explained. In this section the practicalities will be discussed including pseudo-code of the actual models used, examples of specific problems and choices made. It begins with a description of general Lattice Boltzmann Methods after which the specifics of the three particular models are discussed, starting with LBGK, then ELBM and finally MRT. In what follows three dimensions are used unless otherwise specified.

3.1 General LBM

In all the Lattice Boltzmann Methods space is discretized with stepsizes dx and time with stepsize dt. The set of speed vectors has the requirement $dx = |v|dt$ with $|v|$ the lenght of $\mathbf{v}$ in any one dimension. This requirement basically means that every whole timestep dt the f_i of each node move exactly unto a neighboring node.

As already explained in the quadratures subsection (2.2) the set of speed vectors $\mathbf{v}$ can be derived using approximations. They can also be judiciously chosen. As a result there exist many different sets and for each dimension a whole new combination of speed vectors is possible. This is why people refer to specific sets using an abbreviation for dimension and number: DdQb, where d is the physical dimensional size and b is the number of vectors used. As most models have vectors with a maximum stepsize of dx in each dimension, this abbreviation usually is unambiguous. For instance the D1Q3 set is the one given in 2.1.7 and a D2Q9 set would be $\{(i,j)|i,j=-1,0,1\}$. Another two dimensional set is D2Q4 $\{(-1,0),(1,0),(0,-1),(0,1)\}$.

Each lattice node contains a velocity distribution f which contains b different f_i's. For every timestep each f_i is shifted in space to the appropriate neighboring node during the so-called advection step which can be written formally as

$$f_i(\mathbf{x}+\mathbf{v}_i, t+dt) = f_i(\mathbf{x}, t). \tag{3.1}$$

Another way of writing this formula using the indices instead of the vectors for space and time would be

$$f_i(r_{j+v_i^x,k+v_i^y,l+v_i^z}, t+1) = f_i(r_{j,k,l}, t), \tag{3.2}$$

with r the position with indices $(j,k,l|j,k,l=0,1,...,\frac{L}{dx})$ for the x,y,z dimensions, respectively. This step is also called the *propagation* step.

After each advection step the fluid undergoes a *collision* step during which all the f_i interact with each other per node during which they in general move closer to their equilibrium values. This is a relaxation and described by

$$f_i(\mathbf{x}, t+1) = f_i(\mathbf{x}, t) - L_{ij}(f_i(\mathbf{x}, t) - f_i^{eq}(\mathbf{x}, t)), \tag{3.3}$$

with $\mathbf{L}$ a linearized collision operator, like the one from (2.39), and f^{eq} the equilibrium functions. These equilibrium functions have to be calculated per node per direction.

One way of introducing forces into the LBM is using *body forces*. In this case the body force per node $\mathbf{G}$ is calculated and then using the definitions of the speeds v_i and the body force vector the change in speed density per direction can be calculated from

$$f_i(\mathbf{x}, t+1) = f_i(\mathbf{x}, t) + \mathbf{G}\cdot\mathbf{v_i}, \tag{3.4}$$

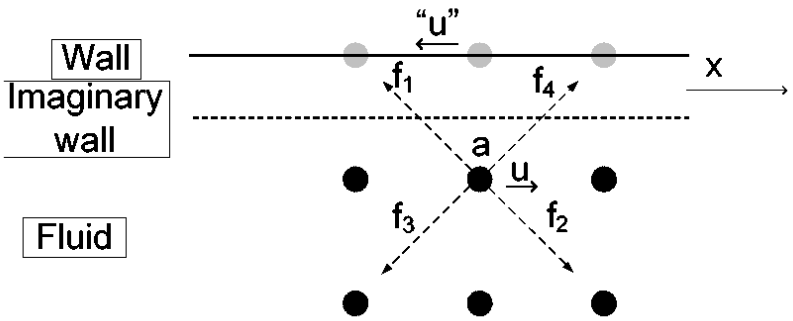

Figure 3.1: Bounce Back on Links with a wall on the top nodes, two layers of fluid nodes and an imaginary wall.

where $\cdot$ is the inproduct (over space).

Another way of introducing forces can be done by setting some boundary conditions, for instance a pressure difference between two boundaries. If one layer of nodes has a constant pressure which is higher than the neighboring nodes, there will be a higher flux of f_i coming out of that layer than going into that layer.

3.1.1 Boundaries

So far forces, propagation and collision have been described but boundary conditions have not been in the picture while they are clearly necessary for these models. Many different boundary types have been developed for the LBM, here a few popular types are discussed.

The simplest boundary is a non-existing one, or, in other words, a periodic boundary. If all the f_i that move out of bounds are remapped to the other side of the lattice nothing can go wrong with isotropy, Galilean invariance, the conservation of mass and momentum. A static wall might seem as simple but is however much more complicated.

One implementation for static walls is called Bounce Back on Links. All the f_i that hit the wall bounce back straight to where they came from:

$$f_i(\mathbf{x}, t+1) = f_i^{inv}(\mathbf{x}, t), \tag{3.5}$$

where f_i^{inv} is the f_j belonging to the v_j which is the exact opposite of the v_i. Obviously with BBL the density is conserved. The layers flowing directly next to the wall are however feeling incorrect forces [46]. This is best explained with a simple example. Take a two dimensional flow as drawn in figure 3.1.1. The flow is moving in the positive x-direction and a wall is placed along this direction. The node a has f_1 moving into the wall. If BBL is used its f_2 is updated with the value of this f_1. Its f_3 is updated with its f_4. As a result this node is updated with speeds from imaginary nodes inside the wall with exactly the opposite speed. If all the nodes along the x-direction are updated like this, all of them undergo forces as if the wall would be part of the fluid but moving in the exact opposite direction. Studies have shown that BBL results in a first order error in velocity [47]. This degrades the whole simulation because the models are of second order as can be proven with for instance Chapman Enskog analyses.

Another boundary implementation is the Half-way Bounce Back (HBB) where the wall is placed between the fluid nodes and the original wall nodes. This results in second order accuracy for some simple flows [46].

initialize global variables
initialize grid
while *!END_CONDITION* **do**
    apply bodyforce
    perform collision
    propagate
    impose boundary conditions
end

Algorithm 1: A typical LBM.

for *all nodes* **do**
    calculate the equilibrium distribution function
    update the density function according to 3.3
end

Algorithm 2: A standard LBM collision step.

It is also possible to use the non-equilibrium part of the distribution functions in the bounce back scheme. This way all the updates on the wall nodes take into account the known quantities like fluid velocity and density at the wall and their incoming and outgoing f_i's are calculated accordingly (as proposed in [47]). The result is a second order boundary condition.

The list of possible boundary conditions is quite big [15] and these are just some of the more basic and well known types. The problem with most of the boundary models is that they can become quite complicated for three dimensional irregular structures. The only exception known to the author is the BBL.

3.1.2 Overview

A typical simulation run is as in Algorithm 1.

Where `END_CONDITION` can be specified as a function of the accuracy or specified by the specific physical problem at hand. The collision step can be set up as in Algorithm 2.

3.1.3 Parallelization

In the LBM each lattice site depends per iteration only on its neighbours reachable via one step of v_i. Since most models use a set of v_i of length dx per direction the result is that LBM displays an extreme locality. Hence a parallelization of this method is rather straightforward as long as the geometry is simple. For more complex geometries like networks of thin tubes (like veins in the lungs) parallization becomes more difficult, but is still possible [11, 29]. The grid has to be split up geometrically in chunks. The only communication necessary between different processing units are the global information at startup and end of the simulation and each iteration the distribution functions of the nodes which lie on the edges of the areas of the chunks. If the lattice is big and the number of processing units small the layers of nodes in between the chunks are relatively small which results in a small amount of needed communication. All the acceleration tricks that apply in normal discretized models with only local forces can also be applied in the LBM.

3.1.4 Procedure

In order to get some hands on experience of lattice Bolzmann simulations first a two-dimensional simulation was created from scratch. This was a D2Q9 LBGK simulation of a Womersley flow with BBL boundary conditions on the walls and time-periodic pressure gradient in- and outlet conditions [6, 47]. All steps looping over the grid were optimised to prevent pipe-line breaking which means that all the loops that ran over the grid contained no conditional statements.

It was found that pressure gradient in- and outlet conditions can be quite tedious to implement as many exceptions have to be made at the corner nodes of the lattice. The predicted incorrect fluid flow velocity along the wall of BBL was found in the

velocity profiles. Some other small implementation specifics have been found which could be used in the real simulations.

Next the step was made to three dimensions. An implementation of the LBGK was supplied and used as a starting point for all models (see next paragraph). To test the implementations they were compared with this original version because the program was proven to be correct up to some order (see for instance [6]). Because the ELBM implementation had significant problems after testing with Womersley flows (see 3.3.1), Poiseuille flows were taken as a starting point. When everything worked the three models were put to the test using Womersley flows.

The models were implemented for a tube geometry periodic in its in- and outlet and the tube was oriented parallel to the x-axis. As a result the body force works only on the subset of speed vectors that have a non-zero x component. Also the periodicity implementation is then as simple as possible (but not that simple that no errors can occur in implementing it, see next paragraph).

To obtain Womersley flows the body force has a harmonic (sine wave) dependence in time. It is usefull to start with a sinus and not a cosinus because the sinus start with a zero body force. This prevents the simulation from breaking down simply because starting with a maximum speed with a static flow can easily trigger instabilities. The simulations were run for 40 periods to let them relaxate, a number typical for these simulations [6]. For a Poiseuille flow the body force is set to a constant value.

To run the simulations Lisa, a national computer cluster at SARA, is used. SARA is an ICT service center. Because big parameter sweeps are done, looping over Reynolds numbers, Viscosity, Mach numbers and lattice sizes for three different models, the program is run sequentially. As all models are or can be parallized in exactly the same manner, this does not influence the results of the comparison. A couple of scripts are created to automate these sweeps and to process their output data. Instead NIMROD could have been used but this is maybe the worst supported, documented and steady software package known to the author. It has been succesfully deployed in the past but for this project it did not work out well in combination with Lisa. Most of the figures in this report are made using Gnuplot.

3.1.5 Error calculation

> There are three kinds of lies: lies, damned lies, and statistics. [1]

The difference in quality of the models depends largely on the error and therefore on the error calculation. As the analytical solution of both the Womersley and Poseuille flow exist, the problem of finding the right formula would seem rather straightforward. Some formula along the lines of

$$Er = \frac{|u_m - u_{th}|}{u_{th}},$$

would probably do, with Er the local error and u_m and u_{th} the measured and theoretical fluid speed respectively. The problem with this definition is however that this Er blows up when u_{th} becomes very small. In many problems the u_m is also very small for those positions but due to the relatively large error caused by the bounce back on links (BBL) boundary condition (BC) near the walls, the absolute small errors near these walls can dominate the global error when this error definition is used. The author assumes this is the reason why multiple researchers (e.g. [6, 11]) use instead a definition based on the sum of the speeds along a cut-through line, lying in a plane perpendicular to the flow:

$$Er_{global} = \frac{\sum |u_m - u_{th}|}{\sum u_{th}}. \tag{3.6}$$

Thus the trick is to sum first and then divide instead of dividing first using the tiny speeds near the walls. The result is an error measure which is less sensitive to local errors at positions where the speed is very small. This should always be taken into account when dealing with this measure.

[1] Benjamin Disraeli

There is however one problem with the above definition for Womersley flows and it arises when the sum over the theoretical speed also becomes zero. This should not give problems except when there is a time lag between the theoretical solution and the simulation. With such a lag the error again blows up whenever the Womersley solution goes through zero. In the measured data there was such a lag, similar to the one reported in [6], thus another global error definition is used:

$$Er_{global} \equiv \frac{\sum |u_m - u_{th}|}{D u_{th}^{max}}, \tag{3.7}$$

with D the diameter of the tube and u_{th}^{max} the maximum theoretical speed which equals the maximum lattice speed U. Thus for a larger theoretical speed the error measure does not increase nor for a larger lattice size. Using the above definition the global error measure remains relative but is decreased by a large factor for those moments at which the theoretical speed is small.

As the object of this thesis is to compare the three models and the author has no background with either three, lieing with these statistics is done as equal for the models as possible. Ofcourse the newly defined error measure looses its statistical interpretation as a true relative error such as a chi-squared error but as long as the same measure is used for all measurements, the above definition does not result in any model-preferences.

Because the flow is time dependent the error also varies in time. Because a time independent measure of the error is needed the errors are averaged per period. Thus the final error formula becomes

$$Er(P_i) = \frac{1}{P} \sum_{t=iP}^{(i+1)P} Er_{global}(t). \tag{3.8}$$

3.1.6 Womersley parameter sweeping

To measure the quality, accuracy, stability and speed, of the three models they have to be run for some parameter sets. The parameters are Reynolds (Re) and Womersley (Wo) number, viscosity (ν), lattice (D) and period (P) size and Mach number (Ma) or lattice speed (U), related via

$$\begin{aligned} Re &= \frac{UD}{\nu} && (3.9)\\ Wo &= \frac{D}{2}\sqrt{\frac{2\pi}{P\nu}} && (3.10)\\ &= \frac{D}{2}\sqrt{\frac{2Re\pi}{PUD}} && (3.11)\\ P &= \frac{\pi}{2}\frac{Re}{Wo^2}\frac{D}{U} && (3.12)\\ Ma &= \frac{U}{C_s}, && (3.13)\\ & && (3.14) \end{aligned}$$

with $C_s = \sqrt{1/3}$ the sound speed.

The three qualities vary over these parameters. In order to get a good understanding of the underlying behaviour, the models should be tested over a whole range of these parameters. The question then rises: how should this parameter sweep be set up? Ideally all parameters are varied over extreme ranges, like a Reynolds number from as low as one to as large as ten thousand and a Womersley number from zero to sixty. This would however result in an impossibly large sweep taking A to much compute time and B resulting in too much data output. To give a feeling on how impossible exactly, here follows a small example.

Say a lattice size of 40 by 40 by 40 nodes is used, which is large enough to simulate complicated flows. A lattice speed of 0.1, which gives $Ma = 0.17$, a large Mach number. A Reynolds number of a thousand (which is normal in the abdominal aorta) and a Womersley number of one, then the needed number of time step per period, P, equals roughly six hundred thousand. As forty periods are needed for relaxation, this

means the simulation needs twentyfour million (!) iterations. On this grid size the used LBGK implementation needs about 0.1 second per iteration on one processor of the lisa cluster, which means a devilish execution time of 666 hours. Obviously doing parameter sweeps around these numbers is quite impossible, especially taking into account that the LBGK is the fastest model measured in iterations per second (more on this follows).

Because of these extremely many iterations per period needed for low Womersley numbers, it is decided that the simulations will only be done for a Womerley number of 4 and upward. This corresponds to the second isoline from the left in Figure 3.1.6: $P = 50000, D = 20, U = 0.1$. Note that typical numbers are 16 for the abdominal aorta and 9 for the carotic artery under resting conditions [6]. A very interesting result obtained in [11], for Womersley numbers below 4, can thus not be obtained. Note that for these results a high resolution in Reynolds numbers is also needed which would require even more compute time.

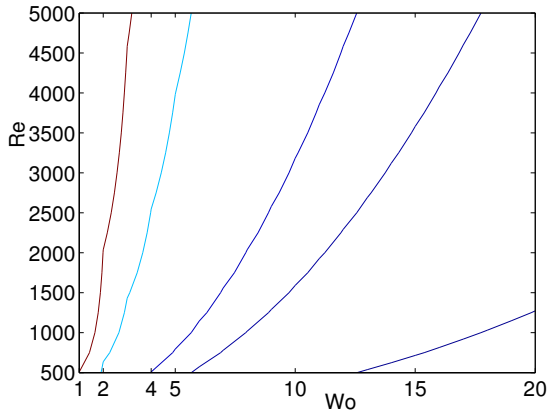

Figure 3.2: From left to right the isolines for period sizes 160000, 50000, 10000, 5000 and 1000 for a simulation with $D = 20, U = 0.1$ for varying Reynolds and Womersley numbers.

While there really are four parameters that can be varied, only two of them determine the flow that is simulated in reality: the Reynolds and the Womersley number. The other two parameters determine the discretization of the system. These two can be chosen and in this experiment the maximum lattice speed (U) and the lattice size (D) are used. To create a parameters space that takes into account the rather large need of compute time, some limitations are used. The different lattice sizes takes are ten, twenty and thirty as the compute time scales as $O(D^3)$ and a lattice size of fourty has proven to be rather slow for the ELBM. The maximum lattice speed set is $0.01, 0.05, 0.1$. Larger would completely break the assumption of low Knudsen numbers used in the derivation of the models, which sometimes can done without giving the expected troubles [11], probably due to highly symmetrical systems, but is not done in this work. Smaller lattice speeds would increase P and thus the runtime of the simulation. This set of lattice speeds follows from hands on experience from preliminary parameter sweeping which will not be discussed in this report.

The Reynolds number set is $\{Re \in (500, 800, 1000, 1200, 1400, 1600, 1800, 2000, 2500, 3000, 3500, 4000)\}$ as follows from the discussion above, and as the Womersley number should be larger than four, the Womersley set is $\{Wo \in (4, 8, 16)\}$. The result is a parameter sweep of $|Re| * |Wo| * |D| * |U| = 12 * 3 * 3 * 3 = 324$ runs of the simulation for each model.

To compare the quality of the models three parameters should, as already mentioned, be taken into account: stability, accuracy and execution time. Because there are four independent parameters, as follows from the set of equations 3.9, that can be varied, the resulting space that needs to be explored becomes rather large. It becomes even more complicated when it is realized that the speed of relaxation with which the different models move towards their respective equilibrium might not be the same. This means that for some parameter set model A would perhaps need

only five oscillations to obtain its "optimal" accuracy/speed moment, while model B might need fourty oscillations. With "optimal" accuracy/speed the relation between decrease in error and increase in runtime is meant. For instance if a simulation would take twice as much compute time for a reduction of its error of one percent, this would be rather non-optimal. Ofcourse the applicational demands determine the size of the maximum error but the goal is to keep the conclusions of this work as general as possible.

Thus, in order to compare the accuracy or execution time it is necessary to cut down the size of the parameter space and compare them step by step for different sets. Because the initial goal of this research project was to compare the models for blood flows, typical parameter sets applicable for those flows are taken as a starting point. In the abdominal aorta typical numbers are $Re = 1150, Wo = 16$ and in the carotid artery $Re = 500, Wo = 9$. For these two settings the relation between most of the parameters are made. The decrease in error as a function of execution time and oscillation number and the execution time and error for different lattice sizes and speeds is compared.

The relation between number of periods oscillated P_i and the decrease in error is, as explained above, rather essential for accuracy and execution time comparisons. To investigate this relation a small study was required. A small program was created to calculate the error per oscillation using Equation 3.8. When plotting these error versus P_i the error regression behaviour could be properly analysed. After a little search an exponential decay was found to be the underlying function combined with a constant minimal error. To properly compare all the models a fit was made for every simulation using this same program. Now all the error regression functions of all the simulations could easily be plotted and analyzed. to calculate the

The stability study is, compared to the execution time and accuracy studies, rather straightforward due to the nature of stability: a simulation is either stable or it is not. Thus the clearest view on stability might be obtained by simply drawing a table of all the stable/unstable parameter sets. The resulting relations will then follow. Apart from this, comparisons will be made to the literature. Note however that not much literature is to be found on *three* dimensional oscillatory pipe flows apart from those done with the LBGK. More on this will follow in the Discussion, Section 5.

3.1.7 Supplied code

For the three dimensional simulations code was supplied by the Section of Computational Science. The authors are Drona Kandhai and Anti Koponen (2000), Abdel M. Artoli (2003) and Lilit Abrahamyan (2004). It containes a complete workframe for LBGK simulations of systolic flows for irregular geometries using various boundary conditions. The geometry can be specified by using custom made Visual Toolkit-based programs, Magnetic Resonance Imaging scans or pre-defined geometries like pipes. The program can return time sliced speed vector fields of the flow, summaries of the momentum, and more.

The whole program is parallized including various possible decompositions. The Message Passing Interface is used and the code is partly C, partly C++. Due to the complexity resulting from the immensity of this application and the number of people that worked on it in different periods, it took some time to understand the logic behind the code. The short or often non-existing comments in the code did not really help either. There was also an inconsistensy/error found, described below. The error did not show or was relatively small when some combinations of settings were used so this does not have to say anything about the many results obtained with this code [6][29][11].

This program was used as the LBGK implementation and altered where necessary to obtain ELBM and MRT implementations. Before this was done a little bit of cleanup and error-fixing had to be done. Also some parts were extended (argument passing, parsing and checking) or removed. Some parallelization decompositions were said to be malfunctioning and after alot of husling all but the slice-parallelization were put on non-useable. Because of the complexity of non-BBL boundary conditions, especially when combined with various D3Qb models, only the BBL condition has been used. The same goes for the in- and outlet conditions, these were removed and the only flow that has been implemented is a tube flow with periodic in- and outlet.

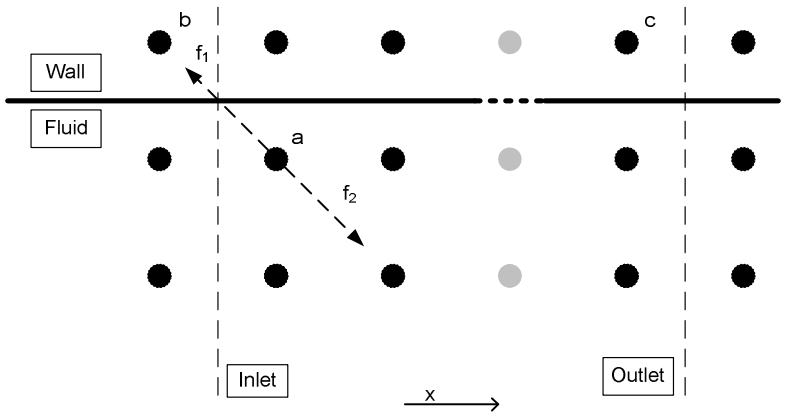

Figure 3.3: An example of an error occuring using BBL wall and periodic in-outlet boundary conditions in combination with parallized code.

One error of the supplied program, and its solution, will now be explained. It occors with the combination of periodic in- and outlet, BBL wall conditions and a parallized simulation where these in- and outlets have to be communicated. A simple two dimensional model will be used, the D2Q9, figure 3.3 illustrates the example. Two steps in this process are taken, propagation including BBL and communication.

First the f_1 of fluid node a is shifted to wall node b during the propagation phase. Then in the BBL phase the wall nodes "bounce back" their received f_i's which means that f_2 of node a is updated with the f_1 of b. Next comes the communication phase in which the periodicity of the lattice requires the incoming f_i of the first column of the lattice to be updated with the outgoing f_i of the last column. In this case this means that the f_2 of node a is again updated, now with the f_2 of node c which is zero because node c is a wall node and did not receive any f_i from nodes outside of the fluid region of the lattice. The solution to this problem is to not update those f_i's at the in- and outlet that are originating from the wall.

3.1.8 Statistical noise and initialization

The density distribution function of each node has to be initialized. It is safest to initialize all nodes on the lattice with zero speed. This way the relaxation of fluids from static to flowing state can be monitored. Initializing zero speed can be obtained by setting one f_i equal to the local density (which is equal to one in all simulations in this work). As a result the first timestep a huge difference exists between the equilibrium values of the f_i causing a big leap in the speed distribution functions. This may result in instabilities. Another option is to set all f_i equal to the equilibrium function values belonging to zero speed, which physically means that the simulation starts with a fluid relaxated at rest.

Because statistical noise is needed in the simulations (see 3.3.1)a randomization has to be imposed. This can be done at initialization as was done in the supplied code.

3.2 LBGK

The main formula of the LBGK is 2.41:

$$f_i(\mathbf{x} + \mathbf{v_i}, t + 1) = f_i(\mathbf{x}, t) - \omega(f_i - f_i^{eq}),$$

where ω is the inverse of the relaxation parameter τ and the equilibrium distribution can for instance be taken as [25]:

$$f_i^{eq}(\mathbf{x}, t) = w_i\rho\{1 + \frac{v_{ia}u_a}{c_s^2} + \frac{u_a u_b}{2c_s^2}(\frac{v_{ia}v_{ib}}{c_s^2} - \delta_{ab})\}, \tag{3.15}$$

with w_i depending on the specific model (DdQb) and a, b representing the cartesian coordinates with implied summation for repeated indices.

In the supplied code and thus in this research project the D3Q19 model is used with the speed vector set $\mathbf{v} \in \{(i,j,0),(i,0,j),(0,i,j)|i,j \in \{-1,0,1\}\}$.

The code has been partly optimized by the various authors with tricks including loop unrolling, static constants, no conditional statements inside lattice wide loops unless absolutely necessary, many compile time switches (preventing some runtime conditional switches) and more [28].

3.3 ELBM

As has been explained in the theoretical section (2.3) there exist many different Entropic Lattice Boltzmann models. Many ELBM theorists claim that ELBM is more stable than classical LBM due to the reinstatement of the H-theorem which makes it "more physical". Hence in the authors opinion only those models using a truly physical entropy, namely the Boltzmann entropy, should be used. The tendency of the entropy functions used in the published papers also shows this direction [40]. Within this set many different models still exist but the difference between them mostly lies in the speed vector set used.

Via different paths (see 2.3.2) it can be concluded that a good speed vector set in three dimensions is the one of D3Q27 [1, 26, 37], with the speed vector set $\mathbf{v} = \{(i,j,k)|i,j,k \in \{-1,0,1\}\}$.

The resulting equilibrium function is the product of three times the one-dimensional solution [1, 2, 16] and given by

$$f_i^{eq}(\mathbf{x},t) = \rho\omega_i \prod_{a=1}^{D} (2 - \sqrt{1+{u'_a}^2})(\frac{\frac{2}{\sqrt{3}}u'_a + \sqrt{1+{u'_a}^2}}{1-\frac{u'_a}{\sqrt{3}}})^{v_{ia}/\sqrt{3}c_s}, \tag{3.16}$$

where $D = 3$, a is the dimensional index, u' is the dimensionless velocity $u' = u/c_s$, ω_i weighting factors also used in the entropy calculation and $c_s = \frac{1}{\sqrt{T_0}}$. T_0 is some reference temperature, in the isothermal setting with $dx = dt = 1$ it becomes 3 and thus $c_s = \frac{1}{\sqrt{3}}$ giving for the equilibrium

$$f_i^{eq}(\mathbf{x},t) = \rho\omega_i \prod_{a=1}^{D} (2 - \sqrt{1+3{u_a}^2})(\frac{2u_a + \sqrt{1+3u_a^2}}{1-u_a})^{v_{ia}}, \tag{3.17}$$

where $v_{ia} \in -1, 0, 1$.

In this model the collision equation (3.3) is given by [2]

$$f_i(\mathbf{x}, t+1) = f_i(\mathbf{x},t) - \alpha\beta(f_i(\mathbf{x},t) - f_i^{eq}(\mathbf{x},t)), \tag{3.18}$$

with α as defined by (2.62) with however β instead of the ω, with $\beta = \frac{1}{6\nu+1}$ which gives an α of 2 when equilibrium is approached.

The collision step for the ELBM is as shown in Algorithm 3.

Note that as many articles refer to each other and small differences exist between the actual definitions used for τ, ν and α, β it was quite confusing figuring out one complete consistent set of equations. When then typographical errors exist in the articles (compare [1](4) with [43](10) for example) it becomes a daunting task to find the small errors which render the end result totally useless. Finally using the definitions from [1] gave good results, using *only* the definitions from this paper.

The implementation of the calculation of the equilibrium function uses as few divisions as possible and only calculates the various terms in (3.17) once.

The calculation of α, as suggested in many ELBM articles [2, 3, 44], uses a combination of the Newton-Raphson (NR) and the bisection method. This is necessary

for *all nodes* **do**
  calculate the equilibrium distribution function 3.17
  calculate α by solving 2.62
  update the density function according to 3.18
end

Algorithm 3: The ELBM collision step.

due to the fact that (3.17) is highly non-linear. The bisection method is used when the derivative of the function is too small because then the Newton-Raphson method does not work well. (3.17) can be rewritten, using the definition (2.55),

$$\begin{aligned} H(f_{new}) &= H(f_{old}) && (3.19)\\ H(f_{new}) - H(f_{old}) &= 0 && (3.20)\\ \sum_i (f_i - \alpha\beta(f_i - f_i^{eq})) \ln \frac{f_i - \alpha\beta(f_i - f_i^{eq})}{w_i} - f_i \ln \frac{f_i}{w_i} &= 0 && (3.21)\\ &= G(\alpha, f, f^{eq}) && (3.22)\\ & && (3.23) \end{aligned}$$

The derivative then becomes

$$dG(\alpha, f, f^{eq}) = \sum_i -\beta(f_i - f_i^{eq})(\ln \frac{f_i - \alpha\beta(f_i - f_i^{eq})}{w_i} + 1). \quad (3.24)$$

The actual procedure is given in Algorithm 4.

3.3.1 α problems

When the ELBM was implemented it seemed as if the α calculation did not work: it did not start, it did not find a solution or when it did give an α back the numbers showed no relation to the fluid mechanics. First it seemed as if the difference between the distribution functions and their equilibria were just too small so to solve this, so extreme fluctuations in the flow itself were induced. This resulted in a malfunctioning of Algorithm 4 because in order to calculate the G and dG values a logarithm is needed of the "new" f (see (3.22) and (3.24)). This new f was sometimes lower than zero because of the extreme fluctuations and thus the logarithms returned an error. To solve this a function was created to find the maximum α such that the smallest new f_i was bigger than zero. This prevented the errors but still the α calculation did not improve the stability of the simulation nor did the values come close to what was to be expected.

To figure out what was going wrong a step back was taken. Instead of Womersley flows a Poiseuille flow was taken. This is a laminar tube flow with a constant pressure gradient or body force and is abit simpler than the Womersley flow. However, because the three models behave very differently for mach numbers above 0.01 and this was not expected by the author it took quite some time to be able to actually get some decent results. Everything was checked and rechecked only to find that the simulation outcomes differed greatly regardless of the various parameters until in the end the mach number was lowered. As a result a small mach-Poiseuille relation check has been done and this will be reported in the results section (4.1).

There was still a problem. MRT, LBGK and ELBM were clearly able to simulate a Poiseuille flow. Their accuracy was not extremely high but this could be expected with the BBL and lattice sizes used. However, the stability of the models was very good, with no instabilities up to extremely high Reynolds number and the α calculation still did not work as intended. It was reasoned that the geometry was too symmetrical resulting in too perfect (unphysical) flows. To solve this two options are available: use a less regular geometry or impose a statistical noise in the system. The first option was chosen and implemented but the results were not satisfactory. The flows did become unstable but the ELBM simulation still showed no response in α except for Reynolds numbers above four thousand combined with the small lattice size of 8 by 10 by 10. If a lattice size of 18 by 20 by 20 was used the Reynolds number had to be even bigger.

Statistical noise To get the Poiseuille flows unstable on acceptable Reynolds numbers next a statistical noise was imposed. This was done according to 3.1.8.

As can be read in the results section this gave good instabilities.

Because it was now certain that instabilities occured the α calculation was again examined. Instead of making another alteration (the algorithm had been changed severely because it just did not seem to work), the original version was reinstated. Finally this resulted in α values like those that were initially expected (see 4.2).

input : $\alpha_{max}, \alpha_{min}, f, f_{eq}, stableDeviation, \alpha_{accuracy}$
output: α
foreach *speed vector* i **do**
  $deviation = \frac{|f_i - f_i^{eq}|}{f_i^{eq}}$
  if $deviation > maximumDeviation$ **then**
    $maximumDeviation = deviation$
  end
end
if $maximumDeviation > stableDeviation$ **then**
  $Gmin$ = Calculate G(α_{min}, f, f^{eq})
  $Gmax$ = Calculate G(α_{max}, f, f^{eq})
  $dGmin$ = Calculate dG(α_{min}, f, f^{eq})
  $dGmax$ = Calculate dG(α_{max}, f, f^{eq})
  if *(*$Gmin < 0$ **and** $Gmax < 0$*)* **or** *(*$Gmin > 0$ **and** $Gmax > 0$*)* **then**
    G does not cross zero
    return α_{max}
  end
  if $Gmin > 0$ **then**
    Flip $\alpha_{min}, \alpha_{max}$
  end
  $\alpha = 0.5(\alpha_{min} + \alpha_{max})$
  $d\alpha = |\alpha_{max} - \alpha_{min}|$
  $d\alpha_{old} = d\alpha$
  G = Calculate G(α, f, f^{eq})
  dG = Calculate dG(α, f, f^{eq})
  for *iterate till* $maximumIterations$ **do**
    if $(((\alpha - \alpha_{max})dG - G)((\alpha - \alpha_{min})dG - G) > 0)$**or**$(|2G| > |\alpha_{old} dG|)$
    then
      Do bisection update because NR would fail
      $d\alpha_{old} = d\alpha$
      $d\alpha = 0.5(\alpha_{max} - \alpha_{min})$
      $\alpha = \alpha_{min} - \alpha_{max}$
      if $\alpha_{min} == \alpha$ **then**
        Machine size accuracy reached
        return α
      end
    end
    else
      $d\alpha_{old} = d\alpha$
      $d\alpha = G/dG$
      $\alpha_{old} = \alpha$
      $\alpha - = d\alpha$
      if $\alpha_{old} == \alpha$ **then**
        Machine size accuracy reached
        return α
      end
    end
    if $|d\alpha| < \alpha_{accuracy}$ **then**
      Needed α *accuracy reached*
      return α
    end
    G = Calculate G(α, f, f^{eq})
    dG = Calculate dG(α, f, f^{eq})
    if $G < 0$ **then**
      $\alpha_{min} = \alpha$
    end
    else
      $\alpha_{max} = \alpha$
    end
  end
end
return α_{max}

Algorithm 4: The calculation of α via the Newton-Raphons / bisection method.

3.4 MRT

The implementation of MRT was relatively straightforward. The model described in [27] and in the appendix of [21] has been used. The speed vector set used in the LBGK implementation was the same except that the e_i had a different order in i and so all the functions working with speed distribution functions had to be rewritten. This proved to be a very tedious task, moreover because the speed set definition of [21] is slightly ambigues. The collision step in MRT is performed as in Algorithm 5 according to [21](Appendix A) with the main formula of the model as

$$f_i(\mathbf{x} + \mathbf{e_i}, t+1) = f_i(\mathbf{x}, t) - \mathbf{M^{-1}}\hat{\mathbf{L}}(m_i - m_i^{eq}) \quad (3.25)$$

$$= f_i(\mathbf{x}, t) + \mathbf{\Omega}(m_i - m_i^{eq}). \quad (3.26)$$

The transformation matrix $\mathbf{M}$, the equilibrium functions for m_i^{eq} and the $\mathbf{\Omega}$ are given in appendix A. Using $\mathbf{\Omega}$ saves compute time.

The used numbers are derived in [34] with the method explained in section 2.4. In the Results Section 4 it can be seen that the MRT is sometimes unstable at lower Reynolds number than is the LBGK while this contradicts the literature. An explanation was sought for and thus also Jonas Toelke, an expert in the field of MRT, was contacted. He stated that the set of relaxation parameters should not be as in [34] but unity, apart from those related to viscosity. As a result all non-physical moments disappear at every collision. Because own measurements showed this had a strong influence on stability, both those and the original relaxation parameters were used in the Poseuille and Womersley parameter sweeps. The implementation using [34] will be called MRT, the implementation using Jonas Toelkes relaxation parameters will be called MRTJ. The MRTJ relaxation parameter set is used in other research aswell (e.g. [22, 36]).

foreach *lattice site* **do**
  Calculate moments
  foreach *moment* m_i **do**
    foreach *speed vector* e_j **do**
      Using transformation matrix M
      $m_i = m_i + M_{i,j} * f_j$
    end
  end
  Calculate the equilibrium values of the moments
  foreach *moment* m_i **do**
    $m_i^{eq} =$ Calculate $m^{eq}(m_i)$
  end
  Calculate the non-equilibrium values of the moments
  foreach *moment* m_i **do**
    $m_i^{neq} = m_i - m_i^{eq}$
  end
  Update the speed distribution functions
  foreach *speed vector* e_i **do**
    foreach *non equilibrium moment* m_j^{neq} **do**
      $df_i = df_i + \Omega_{i,j} * m_j^{neq}$
    end
    $f_i = f_i + df_i$
  end
end

Algorithm 5: The collision step of MRT.

CHAPTER 4

Results

This chapter will start with the results of the simulations of the Poiseuille flow which was used as a ~~simple~~ test for the implementation of the three models (Section 4.1). Then some pecularities of the ELBM will shortly be discussed (Section 4.2) such that the ~~following~~ Womersley results can be interpreted better (Section 4.3).

In this chapter the Mach number Ma and the lattice speed U are used to describe the same variable as they are related only by a constant. Note however that whenever figures display "Ma" it should say "U". The lattice size L is often used as well. This number is the size of the grid used in the simulations, in these simulations the tube is constructed by creating fluid nodes for all those nodes inside a circle that fits inside this grid. Thus a lattice size $L = 10$ results in a circle of radius $R = L/2 - 0.5 = 4.5$ (minus 0.5 to create a tube which has wall nodes not outside of the lattice). Keep in mind though, that the real radius is not exactly ~~3~~4.5 because of discretization errors.

4.1 Poiseuille flow

In the poiseuille flow simulations the parameter sweep used the following sets: $Re \in \{100, 200, 300, 400, 500, 700, 1000, 1200, 1400, 1600, 1800, 2000, 2500, 3000, 4000\}$, $U \in \{0.01, 0.05, 0.1\}$ and $L \in \{10, 20, 40\}$. As a result small differences can not be measured but increasing the parameter space was impossible due to computational ~~demands.~~

Mach number effects The first results from the Poiseuille flow were, as already stated in the implementation section (3.3.1), flawed due to the wrong maximum speed used. This resulted in big errors due to the fact that the error scales as a function of the Mach number (because the Chapman-Enskog and all other analyses use approximations around zero speed). One graph (left figure of 4.1) is included in this report to give a good example of how big the differences can become. This figure shows the speed at a centerline of the fluid, where the tube has the fluid flowing in the x-direction and the cut is at middle-x and middle-z. The speed is the x-part of the total speed which equals the total speed because the flow is laminar. The line shows the theoretical value. Another graph is included to show how close the simulations approach each other when smaller values of this maximum speed in lattice units is used (right figure of 4.1).

The large influence of the lattice speed or Mach number on the error is logical due to the nature of the discretization of the Boltzmann equation. The variables are expanded around zero speed and thus if this speed moves too far away from zero the errors grow (see Section 2.1.5). The relation has been measured and reported for Poiseuille flows in the literature, mostly for two but also for three dimensions (see for instance [6]).

4.1.1 Sound waves

The results of a small quality comparison of the three models for Poiseuille flow will now be given. First comes the stability check because stability always has a final say: if a simulation becomes unstable its speed or accuracy become irrelevant. Usually checking if a simulation is unstable is very simple: when some values like densities

become infinite something has appearently gone wrong. However in some specific runs something special happened. Instead of breaking out of bounds when reaching high speeds the fluid shows a transient period after which it assumes a semi-stable flow with a lower speed. An example is given in figure 4.2.

There were two possible explanations for this behaviour. Because the flow relaxed to a new and lower speed it was possible that the flow lost speed due to internal friction arising from turbulence. Usually turbulence sets in only at higher Reynolds numbers but because the noise was induced every so-many time steps it was reasonable that turbulence would set in easier. The other explanation was that instabilities arose in the fluid but of such an order that they did result in a lower speed but not such that the whole simulation broke down.

To figure out which explanation was the correct one, a small detail study of the local speeds of the fluid was done. It could be seen that when the fluids reached high enough speeds, relatively big fluctuations arose near the wall of the tube. These fluctuations jumped up and down each timestep and they moved around the grid with the speed of sound. Thus the fluctuations could only be sound waves induced by speed related errors of the boundary conditions and thus the behaviour shown was not due to turbulence but due to instabilities. As a result, whenever this behaviour shows in the graphs, the simulation can be notated as unstable.

Stability The above described instability resulting from sound waves has been taken into account in the following. The measured Reynolds numbers at which the simulations became unstable are listed in Table 4.2.

Before the numbers can be interpreted some things should be mentioned. Many runs of the simulation with a lattice size of fourty did not converge fully yet, due to the maximum runtime. Because the sound waves can take very long to develop making sure they will not arise is sometimes impossible [34]. Thus the true Reynolds number at which the simulation would become unstable can be much lower than the ones mentioned when this happens. The Reynolds numbers at which the simulation was not yet relaxed can be seen in Table 4.1.

For a lattice size of ten, the LBGK is clearly the most unstable model, regardless of lattice speed U. The ELBM shows a consistent higher robustness resulting in almost twice the reachable Reynolds number. The MRT and MRTJ both show very different results for varying U. In some cases the simulation seems to be able to take any Reynolds number and in some cases the model becomes unstable as fast as the LBGK. The instabilities that occured for the MRT were mainly of the sound wave type. More on this in the Discussion Section (5).

At a lattice size of twenty the maximum Reynolds number for stable flows goes up for the LBGK which is due to the rise of the viscosity, a relation which is also reported in the literature. For the ELBM the same happens and its relation to the LBGK remains the same. The MRT(J) shows however a *decrease* in stability. The MRT behaves slightly worse than the MRTJ for $U = 0.05, 0.1$ and slightly better for $U = 0.01$. As the relaxation parameters of the MRT have a significant effect on the stability, it is possible that other numbers will increase the MRT performance. To figure this out a relaxation parameter optimization would be necessary which goes far beyond the scope of this thesis.

The stabilities for the lattice size of fourty are somewhat vague due to the large needed execution time. Table 4.1 shows at which Reynolds numbers the simulations have not yet converged. This means that for larger Reynolds numbers no found instability does not necessarily mean it is stable, whereas a found instability dictates above what Reynolds number the simulations are definately unstable and below which they might be unstable. Hence the number $Re = 300$ for LBGK $L = 40, U = 0.01$ in Table 4.1 and the corresponding 1400 from Table 4.2 means that the simulation is stable up to atleast $Re = 300$ and not above $Re = 1400$. The few conclusions that can be drawn are these: the MRTJ is less stable than the LBGK for $U = 0.01$ and the LBGK is atleast as stable as the MRTJ for $U = 0.05$.

Accuracy The accuracy of the models is measured as explained in Section 3.1.5. The measured errors for $Re = 100$ are in Table 4.3. It shows a close resemblenance of the LBGK, MRT and MRTJ and larger errors for ELBM. An increase of L results

in a large decrease in error which is due to the usage of Bounce Back on Links (BBL) Boundary Conditions (BC). The fact that ELBM behaves so poorly can not be explained, maybe it is due to the combination of ELBM with BBL or with body forces but this has not been tested.

The decrease in error for LBGK and MRT(J) for increasing U at $L = 40$ is also unexpected. This might be the result of an error in the calculation of the maximum speed of the analytical solution because a radius is needed for in its function and it is ambigues if the true radius or the measured radius or the imposed radius should be used, with the true radius the radius solved for halfway instead of normal BBL BC, the measured being the radius as measured from the surface area of a slice of the tube in the simulation and the imposed radius being the radius as explained in the introduction of this chapter.

The small difference in accuracy for LBGK and MRT is expected because the MRT should give a more stable and not necessarily a more accurate simulation [21].

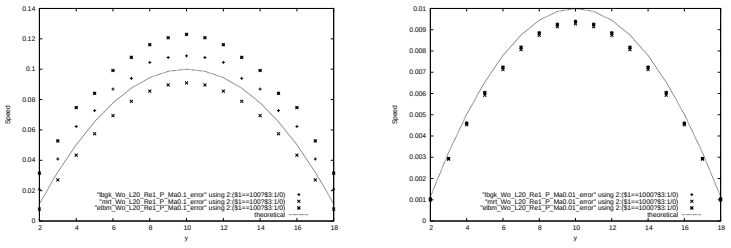

Figure 4.1: The speed of three Poiseuille flow simulations, plus for LBGK, cross for MRT, star for ELBM and the line for the theoretical value. These results were obtained with no noise, a lattice size of 18 by 20 by 20, Reynolds number 1 and a maximum lattice speed of 0.1 ($*1/c_s^2 = 0.3Mach$) for the left and 0.01 ($= 0.03Mach$) for the right figure. On the x-axis is the y-position in the tube and the centerline is taken halfway in x and z positions in the tube at the 100th iteration step in the simulation (the fluids were relaxed).

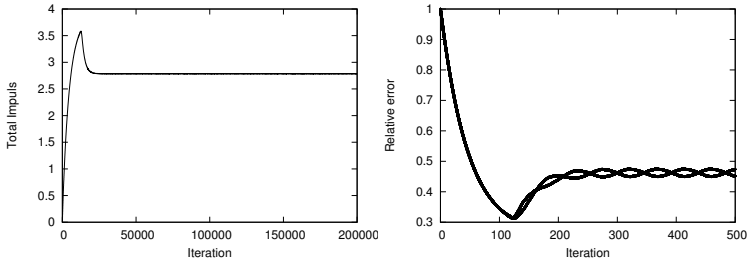

Figure 4.2: On the left the total impuls of a Poiseuille flow with on the x-axis the iteration number, first rising like normal but then suddenly dropping and assuming a "stable" new state. On the right the relative error of that flow summed over one centerline. There appear to becoming two lines but actually the error jumps up and down with every timestep. The graph was made with the output of an ELBM simulation ran with a Reynolds number of 1000, a maximum lattice speed of 0.05 on an 8 by 10 by 10 grid.

L	U	LBGK	MRT	MRTJ	ELBM
40	0.01	300	100	100	100
	0.05	1200	500	500	500
	0.1	2000	1000	1000	1000

Table 4.1: Reynolds numbers at which the simulations have not converged yet.

L	U	LBGK	MRT	MRTJ	ELBM
10	0.01	400	4000+	1600	700
	0.05	400	700	4000+	700
	0.1	400	400	1800	700
20	0.01	700	300	200	
	0.05	700	500	700	1600
	0.1	700	500	700	1600
40	0.01	1400	400	300	-
	0.05	-	2000	1200	4000+
	0.1	3000	4000	2500	4000+

Table 4.2: Reynolds numbers at which instabilities occur for Poiseuille flow.

L	U	LBGK	MRT	MRTJ	ELBM
10	0.01	9.7	10	10	26
	0.05	9.7	10	10	26
	0.1	9.7	10	10	26
20	0.01	5.0	5.1	5.0	11
	0.05	5.0	5.1	4.6	11
	0.1	5.0	5.2	4.3	11
40	0.01	1.4	*	*	*
	0.05	1.4	1.4	1.2	6.9+
	0.1	1.3	1.3	1.0	2.5

Table 4.3: Accurary for Poiseuille flow with $Re = 100$ in percentages. *: Not yet fully relaxed yet. +: Close to, but not completely relaxed.

4.2 ELBM: α

In this section the resulting α which is calculated in the ELBM collision phase will be discussed. Some examples of simulation runtimes in which the α damped the overrelaxating are given, starting with Poiseuille flow. Then some Womersley results are shown and it ends with a discussion of the usefullness of this α calculation.

In order to monitor the α values a virtual three dimensional colored movie moving through time showing the values of the reducing relaxed parameter α at each node in space would suit best. Because of limited time and the paper status of present day theses this has not been done. Instead static two dimensional projections have to be used. Because the geometry is periodic and completely symmetric in the x-direction, the direction of flow through the tube, no relevant information is lost when all layers of the tube perpendicular to x are projected unto one layer. A simple example of this is given in Figure 4.3 where the left "disc" is how the ensemble of the right discs will look like. Note that in this case the result would have been the same if instead of projection simple only one layer at any x would have been taken. When however α's appear at different x but not all x, these values would be lost without projection.

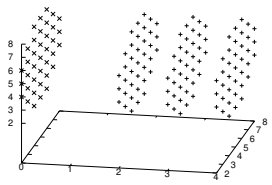

Figure 4.3: Example of projecting layers of a tube perpendicular to x (x = 2,3,4), to x = 0.

Using the projection in x, the x-axis can now be used for the iteration number. In Figure 4.4 the result is shown for the α behaviour in the first stage of the simulation. In the beginning the α's are not exactly 2. (called from now on "non-two") and thus the entropy at some of the nodes would have decreased if α would not have been altered. This is true for many nodes everywhere in the fluid. In a couple of iterations the non-two α's move away from the edges and finally disappear completely. In reality they remain non-two for the middle line in the fluid for quite some time because this is where the fluid gains the most speed. For clarity this is not shown in the Figure. The behaviour is as expected: in the beginning of the simulation the fluid is slightly unstable and thus the α's respond accordingly. Then in a few timesteps the wall already slows down the fluid near the wall and thus these fluid nodes can reach their equilibrium state faster than the fluid nodes in the middle. Soon the information of the geometry of the fluid has travelled through the entire tube and thus the fluid becomes stable.

A more interesting example is that of non-two α's related to instabilities caused by big velocities. When the Reynolds number increases, higher velocities are induced in the flow. For instance the run with a Reynolds number of 4096 for a lattice size of 8 by 10 by 10, using a maximum lattice speed of 0.05, becomes unstable for ELBM around iteration 12400 (see top part of Figure 4.5). The non-two α's are thus expected to show up somewhere before this moment. As can be seen in the bottom part of Figure 4.5 the simulation shows this behaviour. At iterations before the ones drawn, all α's remained two, apart from the start-up of course.

Next some α results are shown for the Womersley flow. In these simulations the flow starts with zero speed and the body force follows a sinus, thus with a zero force at the first iteration. As with the Poiseuille flows the fluid undergoes an initial transient period. The inner fluid nodes simply keep accelerating while the nodes close to the walls feel these walls and attain a lower speed. Because of this the change in speed close to the walls will be relatively smaller and thus here the fluid will be less unstable. Again the non-two α's are expected to move inward, away from the walls, as time passes. This corresponds to Figure 4.6 where darker crosses indicate a lower α. At the beginning all α's are low but they move towards two fast. In the last slice it can

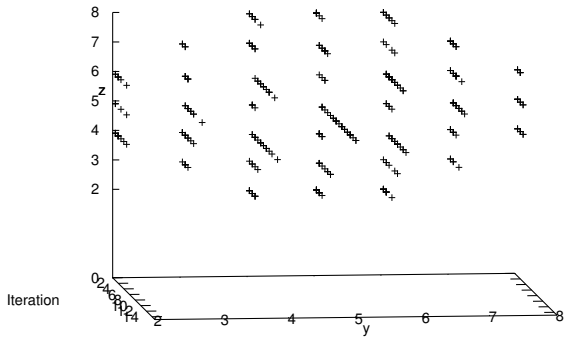

Figure 4.4: The α's for a Poiseuille flow in a tube on a 8 by 10 by 10 grid with a Reynolds number of 1024 and a maximum lattice speed of 0.01. The more the crosses form a line to the bottom-right, the longer they are non-two.

be seen that the outer layers contain higher, and the inner regions lower α's.

When a smaller Reynolds number is used simulations are more stable. Thus the α's should approach two faster. This can be clearly seen in Figure 4.7. In only fifty iterations the non-two α's have moved away from the wall nodes. In Figure 4.8 it can be seen that the non-two's have disappeared completely after about 150 iterations. Because in this simulation the Reynolds number is very small (8), the non-two's do not come back.

To show that non-two α's can reappear for systolic flows, Figure 4.9 has been included. In this simulation the Reynolds number (256) is big enough to create H-theorem breaking changes in the fluid period after period. After two periods (one period is 1438 iterations long) the flow has gone through five maxima in which each time the entropy was locally reduced, but less and less.

4.2.1 The α effects

It can be concluded from the previous section that the α calculation seems to work. The real test however is of course turning the calculation on and off and comparing the results. Strangely enough, no effects were found, *at all*. Whenever an instability occurs in the simulation, the non-two α's also appear, but they never prevent the fluid from becoming unstable. Detail studies have been performed to find the exact Reynolds numbers at which the instabilities occur and then around those numbers the calculation has been switched off and on, but the instabilities remained.

Many things have been tried to figure out the cause, one obscure fact has been found. The local entropy of fluid should decrease whenever the fluid becomes unstable, as can be concluded from the literature, see Section 2.3. However it has been found in some cases that the entropy did not decrease before the simulation became unstable. Then it did, but it was already too late and the present instabilities did not disappear anymore. So for some reason the entropy calculation is in some simulations not an exact measure for the instability, let alone, together with the α calculation, a guard against unstable simulations.

What physically happens inside the fluid whenever an α becomes lower than 2, is that the local viscosity is increased, the fluid moves slower to its equilibrium. For flows in which a global acceleration is imposed, such an alteration might cause weird changes. Say a node would have a small entropy decrease and its relaxation parameter is slightly lowered. Then the next time step its neighbours might all have a higher speed than this node and thus this node should undergo even larger changes. Its entropy might decrease even more and thus it would detach itself from reality in this vicious circle. This problem will spread due to the nature of the LBM and thus the

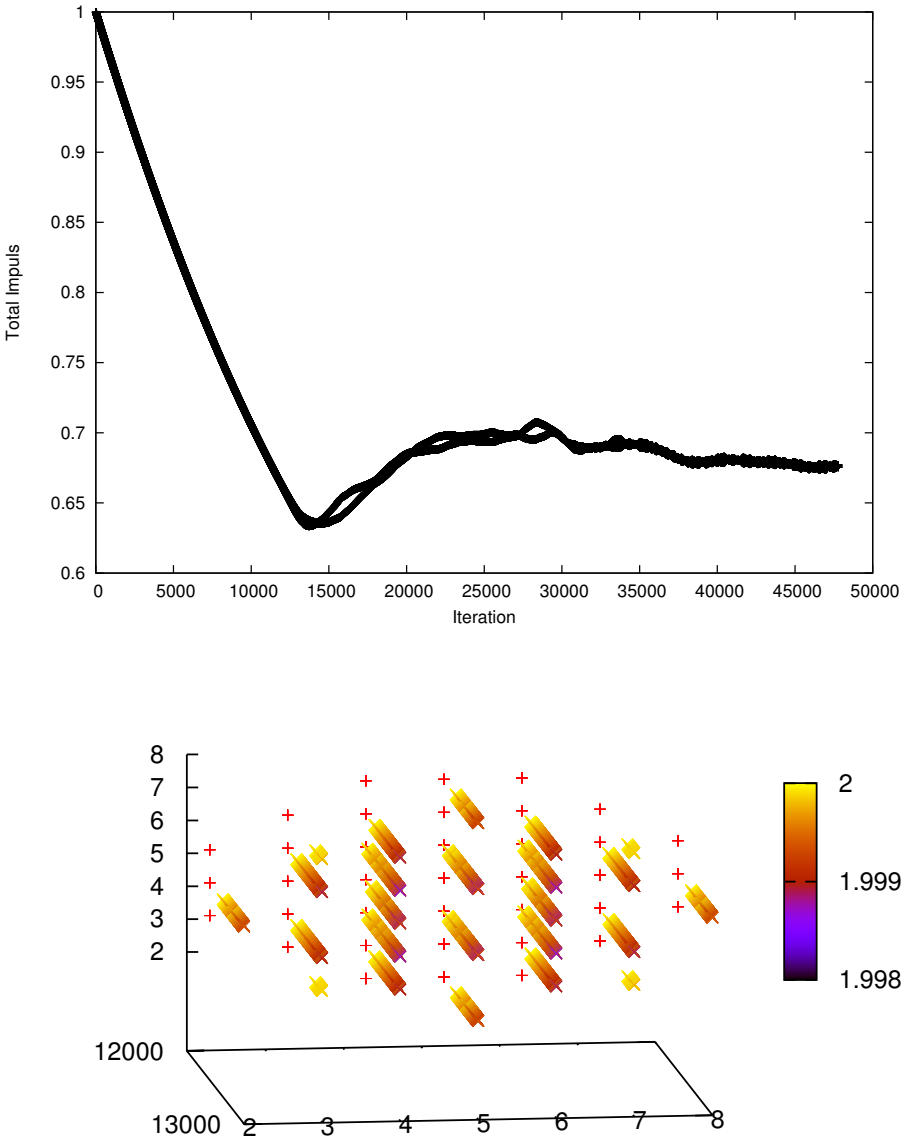

Figure 4.5: On the top the error of the simulation, summed over a centerline in the tube, which should move toward zero but at around iteration 12400 breaks. On the bottom the non-two α's of the simulation (colorod cross) and in the back the y,z-location of the fluid nodes (red plus).

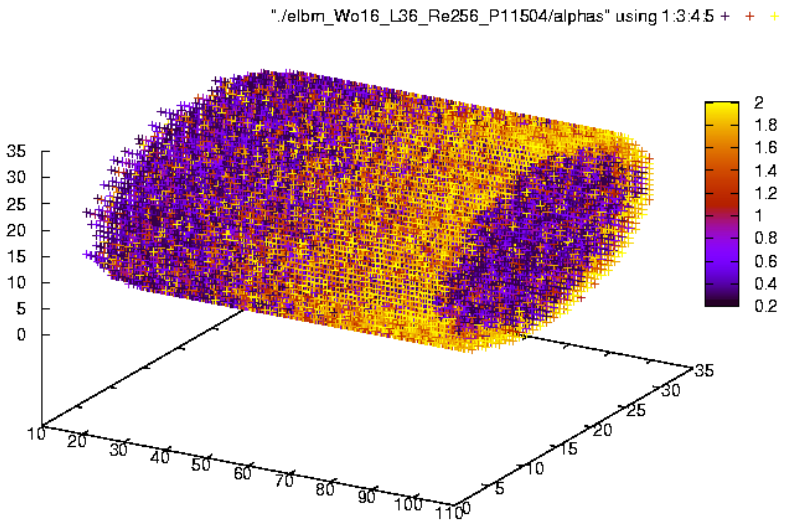

Figure 4.6: The α's for a Womersley flow at the very beginning, with $Wo = 16, Re = 256, P = 11504, L = 36$.

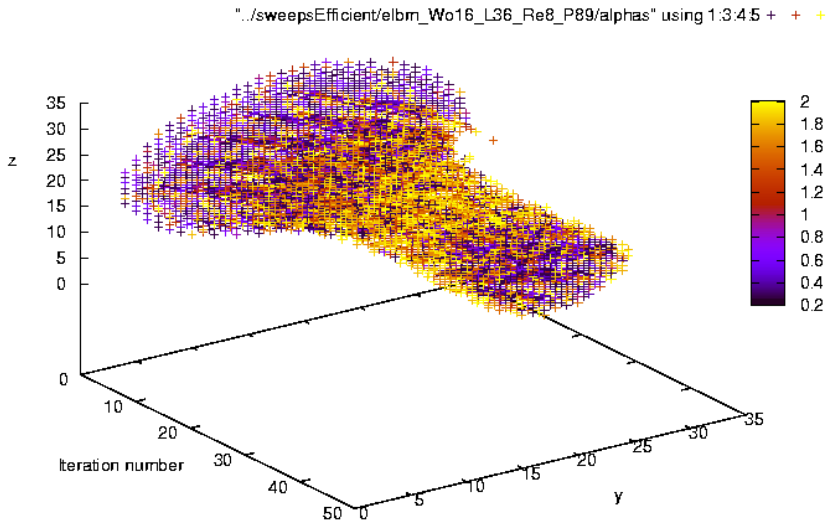

Figure 4.7: The α's for a Womersley flow at the beginning, with $Wo = 16, Re = 8, P = 89, L = 36$.

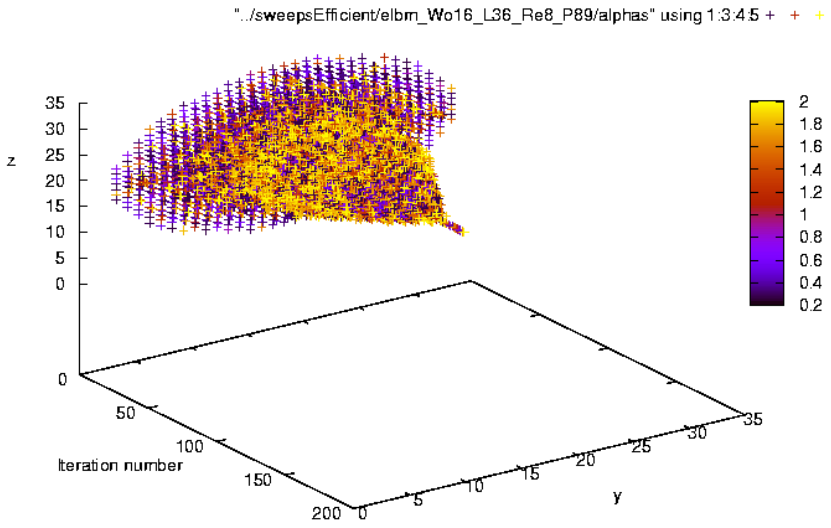

Figure 4.8: The α's for a Womersley flow, with $Wo = 16, Re = 8, P = 89, L = 36$.

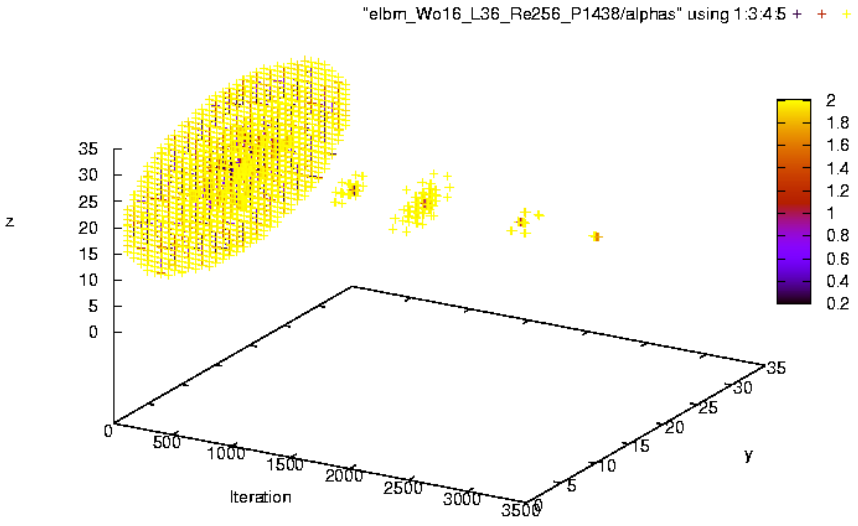

Figure 4.9: The α's for a Womersley flow, with $Wo = 16, Re = 256, P = 1438, L = 36$.

entire simulation can become unstable. Note that the chances on this are higher when there is one global force which moves all the density distributions of the nodes into one direction which will cause the nodes with reduces relaxation parameters to lag behind substantially. When this global force is small or the reduction in relaxation only small, the entropy increase in the next timestep is not necessarily negative and thus the node does not have to enter this vicious circle.

In Figure 4.9 the α's are moving away from two at the peak speeds for first couple of iterations. The reason that this does not cause instabilities is the reduction of the global force which prevents the nodes with lowered α's to lag behind too far. The reason why the α calculation does not help in preventing instabilities is that the whole simulation would not become unstable in the first place due to the decrease of the body force whenever nodes are about to become unstable.

The calculation does use alot of compute time as explained in Section 4.3.3. Its relevence will be discusses in Section 5.

4.3 Womersley flow

Because of the complexity of the parameter sweep study (see Section 3.1.6) a start is made with one combination of Reynolds and Womersley numbers, namely that of the carotic artery. Then the results for the abdominal aorta simulation are presented after which the execution time is analyzed. This in part needs the following erro regression analysis section. The whole section ends with the stability of the models.

4.3.1 Carotid artery

The carotid artery has typically the values $Re = 500, Wo = 8$. The two free parameters are then the lattice speed U and the lattice size L. In Figure 4.10 the error of this parameter set for $U = 0.1$ is shown. What can clearly be seen is the enormous decrease in error for increasing L and the large difference in error of the ELBM and the other two models (including the other version of MRT).

To check the influence of the *lattice speed (U)* on the error the nine parameter sets were compared: $U \in \{0.01, 0, 05, 0.1\}$, $L \in \{10, 20, 30\}$, shown in Figure 4.11. It was found that the lattice speed had only a minor influence on the errors for all the runs. The other differences are a small change in relaxation speed, but more on this in the Error regression Section 4.3.4. Also there do not exist nine lines per plot due to the fact that the instable runs are not shown. These are all the MRT and MRTJ runs with $U = 0.01$ and the LBGK run with $L = 30, U = 0.01$.

The increase in maximum accuracy, or decrease of minimum error, with increasing *lattice size* is measured for the LBGK and shown in Figure 4.12. A fit to the data is also plotted using the function $f(L) = a + bL^c$. The value of c gives the order of the error as a function of L and thus it is found that it goes as $O(L^{-1.65})$. Because the error of the LBGK scales as $O(L^{-2})$ this might seem wrong. The error of the LBGK and LBM in general however reduces to $O(L^{-1})$ when a bounce back on links (BBL) boundary condition (BC) is used. When a BBL is changed in such a way that the effective position of the boundary is halfway, its error scales quadratically again. All the positions in between create superlinear error scalings [7, 12, 19, 23, 46]. Thus a $O(L^{-1.65})$ can be explained because the BBL used in these simulations creates boundary nodes in three dimensions thus they can be anywhere in the range of perfect BBL and halfway BBL.

The lattice size does not influence the steepness of descent of the error and thus only the minimal error influences the period number at which the simulation is relaxed: for lower maximum error the simulation takes more time to relaxate. This has its effects on the execution time, more on this in Section 4.3.3.

The ELBM behaves worst for all parameter sets of the carotid artery apart from the fact that it is more stable. This increased stability is however not usefull as the simulations for which the other models are unstable do not cause the ELBM to excell in accuracy: for $U = 0.01$ the ELBM has still by far a worse error than the LBGK and MRT(J).

The LBGK, MRT and MRTJ behave very much alike. The lattice size does not create any differences at all, the lattice speed only significantly for the step from

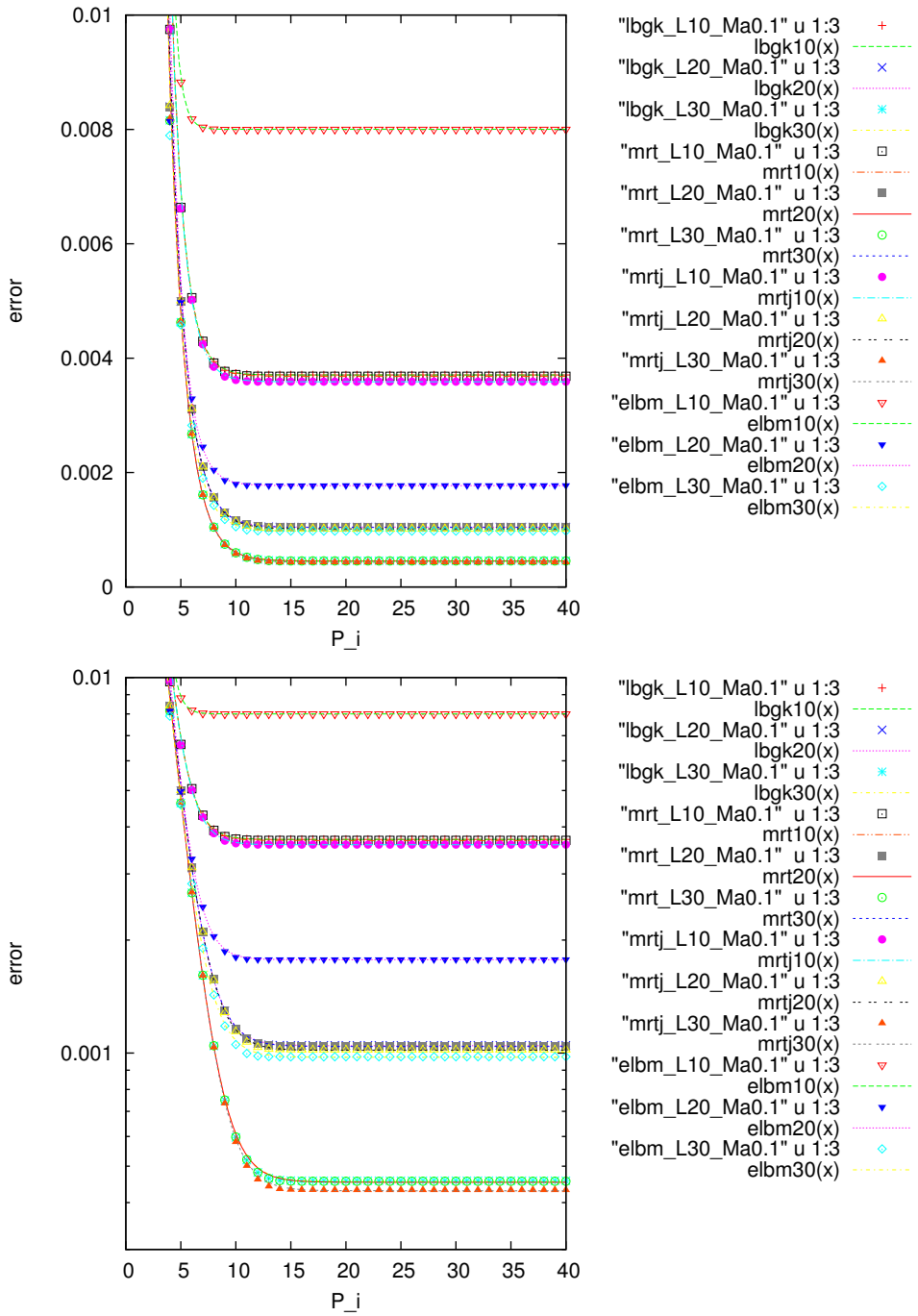

Figure 4.10: The error versus oscillation number P_i for $Re = 500, Wo = 8, U = 0.1$ and all four models with three different lattice sizes, including their fit (see Section 4.3.4). The bottom graph has a logarithmic scale on the y-axis such that the lower lines can be distinguished better. The name of the lines is of the format "model_latticeSize_latticeSpeed_".

$U = 0.05$ to $U = 0.1$. It changes the error behaviour of the LBGK slightly but causes only a minor change in the final error. The minimum errors of the four simulations with $U = 0.1$ are givin in Table 4.4. The MRTJ is giving the best accuracy with MRT on its tail and LBGK closely behind as well. ELBM has about half the accuracy.

4.3.2 Abdominal aorta

Now the other combination of Reynolds and Womersley numbers ($Re = 1200, Wo = 16$), the abdominal aorta, will be discussed. In Figure 4.13 the errors are plotted versus the period number for all lattice speeds and sizes, as was done in Figure 4.11

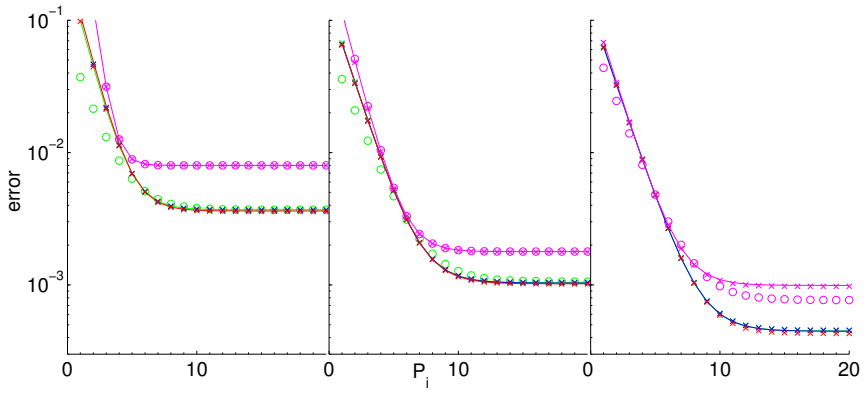

Figure 4.11: The fits of the error versus P_i for $Re = 500, Wo = 8$, models LBGK (green), MRT (blue), MRTJ (red), ELBM (magenta) and lattice speed U 0.01 ("o"), 0.05 (line), 0.1 ("x") and lattice sizes 10 (left plot), 20 (middle plot) and 30 (right plot). The LBGK results in the right plot are obscured by the MRT results.

L	LBGK	MRT	MRTJ	ELBM
10	0.00370	0.00368	0.00359	0.00800
20	0.00105	0.00104	0.00102	0.00178
30	0.000456	0.000456	0.000432	0.000978

Table 4.4: The minimum errors of the carotid artery runs for the three different lattice sizes and three models. $Re = 500, Wo = 8, U = 0.1$.

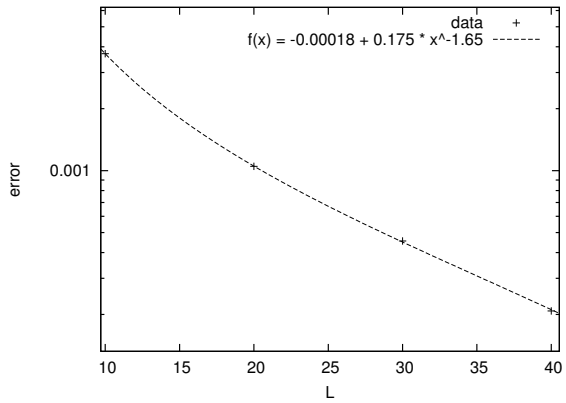

Figure 4.12: The optimal error versus the lattice size L for the LBGK for $Re = 500, Wo = 8, U = 0.1$. The line shows the fit of the data.

for the carotid artery. The main difference with those results is that the error is highly reduced. For ELBM the error is one third for $L = 10$ up to almost one fifth for $L = 30$ and for the other models the error reduces with a factor of two up to three (this non-linear error reduction is due to the $O(L^{1.x})$ relation described above).

The small change in error as a function of U, the constant steepness of the error regression and the reduced error for increasing lattice size L are again clearly visible. In Table 4.5 the minimal errors of the four runs with a lattice speed of 0.1 are given. The differences between LBGK and MRT(J) are now even smaller than for the carotid artery.

4.3.3 Execution time

The execution time of the simulations depends on many parameters. The model, the lattice size, the number of iterations per period and the relaxation-period speed all influence the execution time. Luckily, most of these parameters do not affect each other and thus their effect on the execution time can be examined seperately.

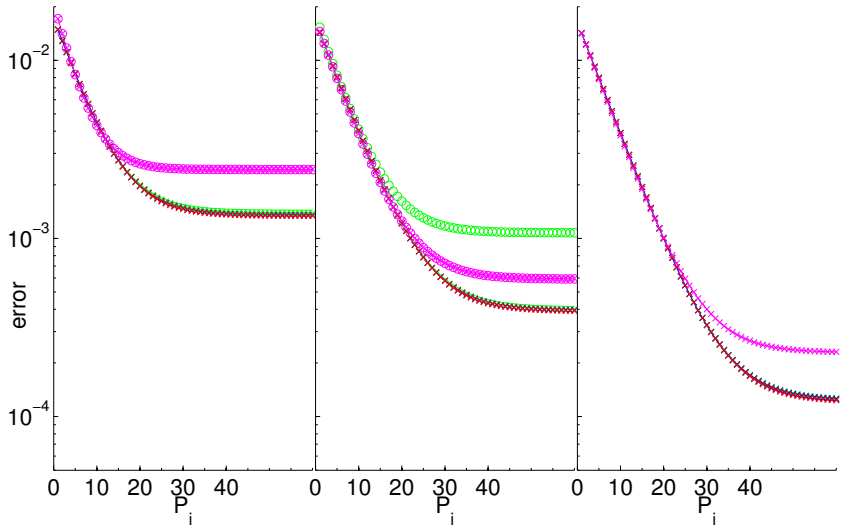

Figure 4.13: The fits of the error versus P_i for $Re = 1200, Wo = 16$, models LBGK (green), MRT (blue), MRTJ (red), ELBM (magenta) and lattice speed U 0.01 ("o"), 0.05 (line), 0.1 ("x") and lattice sizes 10 (left plot), 20 (middle plot) and 30 (right plot).

L	LBGK	MRT	MRTJ	ELBM
10	0.00144	0.00142	0.00141	0.00248
20	0.000451	0.000444	0.000441	0.000662
30	0.000179	0.000178	0.000176	0.000285

Table 4.5: The minimum errors of the abdominal aorta runs for the three different lattice sizes and three models. $Re = 1200, Wo = 16, U = 0.1$.

L	P	T_{ex-tot}	$T_{ex/iteration}$	$T_{ex/node/iteration}$
10	736	65	0.088	2.110^{-6}
20	1472	812	0.14	1.710^{-6}
30	2208	3940	0.45	1.710^{-6}

Table 4.6: The execution time (total, per iteration and per node per iteration) of the LBGK for $Re = 1200, Wo = 16, U = 0.1$.

Because the simulations were run sequentially the execution time per iteration depends linearly on the number of grid points up to some limits as the amount of operations per point is always the same (as follows straightforward from the Theory and Implementation Sections (2, 3)), per model. It only looses its linearity when the number requires memory sizes which grow out of certain cache level or main memory sizes thus increasing the memory latency. This is likely to occur when the *lattice size* L is increased as the number of points increases cubically as a function of the lattice size. However, for the lattice sizes used the simulation never required more memory than the size of main memory because the execution time per node was almost constant. As an example look at Table 4.6 which shows the execution times for the LBGK. The last column shows that the *execution time per node per iteration* ($T_{ex/node/iteration}$) is almost constant. The decrease for larger lattice size is not due to a difference in initialization time, which is negligible, but due to the fact that in the simulations the error calculations is always done for a line in the model which needs a costly computation of Bessel functions for the analytical Womersley solution. This line grows only linearly in L and thus for increasing L this takes relatively less time. As the required amount of memory depends only on the size of the grid, the three models require the same amount of memory and thus this table is sufficient. The used memory size of the program is constant in time due to the fact that this grid is of constant size and all other variables need negligible memory space.

L	LBGK	MRT	ELBM
10	$2.2 * 10^{-6}$	$7.3 * 10^{-6}$	$2.1 * 10^{-5}$
20	$1.7 * 10^{-6}$	$7.6 * 10^{-6}$	$2.1 * 10^{-5}$
30	$1.7 * 10^{-6}$	$7.6 * 10^{-6}$	$2.1 * 10^{-5}$

Table 4.7: The execution time per node per iteration (T_{exni}) of the three models. The values for MRT and MRTJ are ofcourse the same.

The *number of iterations per period, P,* depends inversely linearly on the lattice speed U. Because the fluid relaxation speed in number of oscillations increases slightly to not at all for decreasing U, as shown above and in the error regression analysis below, the total execution time increases linearly to super linearly for decreasing U. As already mentioned the error does not change more than a hundredth, relatively, and thus the U for the simulations should always be taken as 0.1. There is almost no difference between the three models for different lattice speeds.

From the error regression analysis it follow that the number of iterations needed for convergence shows:

$$i_r \propto \frac{ReL}{U} \tag{4.1}$$

and thus

$$T_{ex} \propto L^3 \frac{ReL}{U} = \frac{ReL^4}{U}. \tag{4.2}$$

because the execution time scales linearly with the number of nodes which themselves scale as L^3.

It also follows that the number of iterations needed to reach convergence is almost model independent: the only dependence being caused by the maximum obtainable accuracy which will cause the ELBM, due to its lower accuracy, to converge with 5 to 20% fewer iterations.

Clearly the lattice size is of huge importance, not only for the error which scales as $O(L^{1.65})$ but also for the execution time which scales as $O(L^4)$.

The last variable which determines the total execution time is the model which determines the execution time per node per iteration $T_{ex/node/iteration}$ or T_{exni}. These numbers have been calculated conform the calculation done in Table 4.6 and the results are in Table 4.7. The ELBM has by far the largest T_{exni}, roughly ten times that of LBGK, whereas MRT has roughly four times that of LBGK. The increase of T_{exni} for increasing L for MRT and ELBM contradict the already explained decrease for LBGK. The only possible explanation would be a higher memory latency which is possible for ELBM as it requires more memory because it uses 27 instead of 19 lattice speed vectors. The large difference in T_{exni} per model will be discussed in Section 5.

4.3.4 Error regression

The number of periods required for the simulations to ~~relaxate~~, P_r, or its period-relaxation speed relation, has a linear relationship with the execution time. The problem to calculate this period at which the fluid is converged is that there is no perfect or universal stopping condition, as already explained in Section 3.1.6. Thus the mentioned error regression study will be given here to tackle this problem. As can be seen in Figure 4.10 the fits show a good correspondence to the measured data. These fits have the formula

$$f(P_i) = a + be^{cP_i}, \tag{4.3}$$

which immediately shows that a will give the error if the fluid would continue relaxating infinately long ($f(\infty) = a$). The value c determines the steepness of descent and b and c together determine the initial error. For the same b, c a smaller a would cause the fit to approach its final error at a higher P_i. With approaching is meant: the decrease in error relative to its minimal size a becoming small, or in other words, the derivative of its difference becoming small:

$$g(P_i) = \frac{f'(P_i)}{a} \to 0. \tag{4.4}$$

The value of P_r can be calculated by solving $g(P_r) = \epsilon$, with ϵ some constant value depending on the required minimal relaxation speed, a.k.a. minimal change in error

as a function of maximum obtainable accuracy. These values can be easily calculated with the given fits. This would be usefull were it not for the high similarity of the error regression behaviour of the various models for all the parameter sets which will now be shown.

In Figure 4.14 the fits of the error versus period P_i are plotted for twelve simulations. When looking closely to this figure it can be seen that the steepness of the lines depends mostly on the Womersley number but most importantly, they do not depend on the model. The only real difference between the lines for the different models is the height at which the error stabilizes, given by the value a of the fits. Thus to continue the error regression analyses it suffices to look at a single model. Note that the author has checked all 500 fits to make sure this could be done.

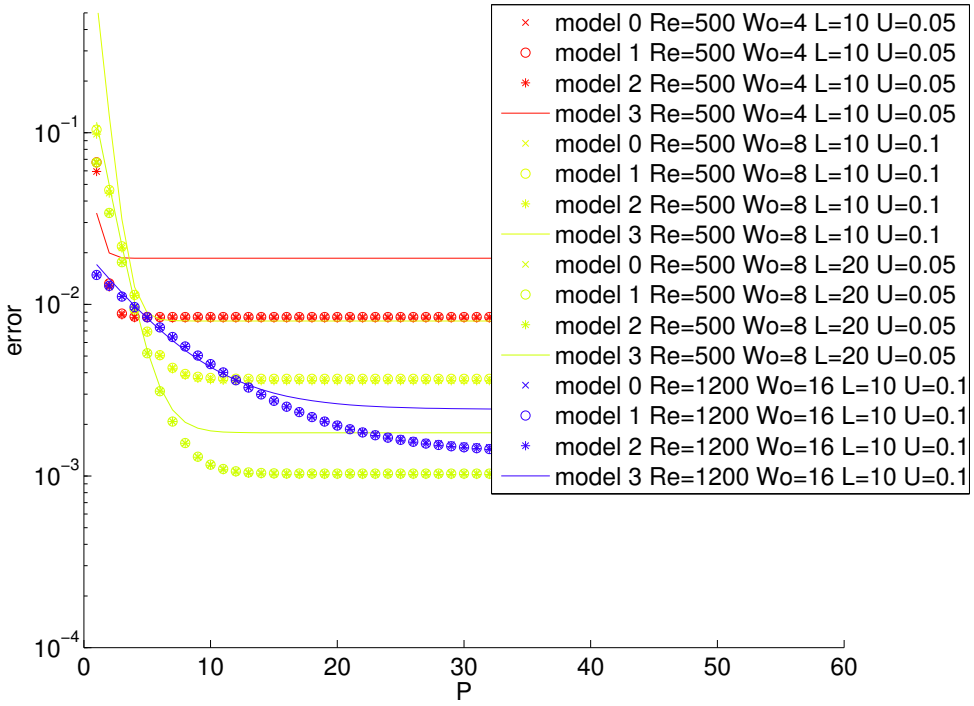

Figure 4.14: The error fit versus period number P_i for the models LBGK = 0, MRT = 1, MRTJ = 2, ELBM = 3 for various Reynolds numbers Re, Womersley numbers Wo, lattice speeds U, and lattice sizes L. The y-axis is logarithmic to create some space between the lower lines and to properly show the steepness of the fits.

In the following the results of LBGK will be used which is just an arbitrary choice. To analyze the error regression as a function of Re, Wo, U, L many plots would be needed. By using three different values of each parameter a good overview can be obtained and thus in Figure 4.15 the 3^4 curves of the LBGK are plotted. As this figure holds quite a lot of data, it can take some time to grasp its essence.

To start off, look at the top left plot, which shows the errors for a Reynolds number of 500 and a lattice size of 10. Thus this in part shows the same lines as Figure 4.14. What can be seen is that the simulation has the biggest $g(P)$, or relative error decrease, for the lowest Womersley number $Wo = 4$ (the green lines), then for $Wo = 8$ (the blue lines) and then $Wo = 16$ (the red line). The difference in steepness differs relatively only marginally to not at all for varying lattice speeds U (the different marks). The optimal obtainable error decreases both as a function of Wo and of U, with again the Womersley number having the largest influence.

Now take a look at the second plot (top middle) which containts the curves for the same Reynolds number and a lattice size of 20. The steepness is not changed visibly while the minimal error decreases and thus the Pr for same ϵ increases slightly. The effect of the Womersley and lattice speed numbers are the same as in the previous graph.

In the third plot (top right) the lattice size is 30. It shows the same behaviour as the second plot. Thus it can be concluded that for these three sets of parameters,

the Womersley number mainly influences the steepness of descent. The lattice speed changes the steepness only marginally and the optimal error largely for higher Womersley numbers. The lattice size does not change the steepness of descent but has a large influence on the optimal error.

When going down in the figure, the Reynolds number is increased. The steepness of descent remains however the same. What only changes slightly is the value of the optimal error.

The global conclusions of this small analysis in short are that A for increasing Womersley numbers the steepness of descent decreases and thus the P_r increases, B for increasing lattice speeds the optimal error increases and thus the P_r decreases, C the lattice size only increase the optimal error and thus P_r and D the Reynolds number slightly influences the optimal error and thus P_r. The P_r-Wo relation has not been reported in the literature, more on this in Section 5.

The relation between the error regression and the iteration number i_r, instead of the oscillation number, was calculated. This could be done by finding the P_r for some ϵ and then multiplying this with the number of iterations per oscillation P. It was found that this number shows a decrease of about 25% up to 30% going from $Wo = 4$ to $Wo = 8$ and a neglibible change between $Wo = 8, 16$. When a simulation has more iterations per period, the fluid has more iterations to adapt or to "feel" the global force and the geometry. As a result the fluid can relaxate more per period. The reason the number of iterations needed for convergence lowers for increasing Womersley numbers at small Womersley numbers is due to the fact that the simulation starts with a sine and not a cosine. The result is a flow which has as first maximum velocity much larger than its final maximum velocity and thus this will take atleast a few iterations to disappear. As an example Figure 4.17 is included which clearly shows the large change in the flow for the first one and a half iterations. When a force would start with as a cosine this would not happen but then the body force would be switched on at maximum force which could cause instabilities to develop more easily.

Now that it is clear that the number of iterations needed for convergence (i_r) does not depend on the Womersley number, as long as the Womersley number is big enough, the question rises what the influence of the other parameters is. Because however it has already been found that the other parameters do not influence the P_r significantly, the relations for $i_r(x)$, with x being one of the parameters, follow directly from Equations 3.9.

Stability This chapter ends with the stability analysis for the whole Womersley parameter sweep. In the top plot of Figure 4.18 the Reynolds/Womersley combinations can be seen for which none (red) or at least one (green) combination of lattice speed and size resulted in a stable simulation, for each model. It can be seen that the ELBM is the most robust model not able to remain stable only for the lowest Womersley number ($Wo = 4$) at Reynolds numbers above 2500. The LBGK follows closely having the same results for higher Womersley numbers and being unstable at $Wo = 4$ for Reynolds numbers above 1600. The MRT has abit more stability problems breaking down at all Womersley numbers at some point. The MRTJ behaves worst with no stable solutions for Reynolds numbers above 1600.

The reason that models behave worse for lower Womersley is because each oscillation has more iterations and thus if unstabilities develop they have more iterations to grow before the global force field slows the whole flow down again. Also the parameter set is not complete for smaller Wo because of the limitation on P. There is for instance only one parameter combination for $Wo = 4, Re = 4000 : L = 10, U = 0.1$. Thus if the models become more stable as their lattice size grows (which is the case, sometimes the maximum obtainable Reynolds number for stable simulations is even given as a number per lattice unit), the limitation on P prevents those models to have a high enough lattice size for stable simulations.

Keep in mind however that increasing the P and L would increase the execution time of the simulation drastically and thus whenever a Re, Wo combination is green for one model and red for another it really does say the stable model is better for that combination, if T_{exni} of the stable model is not much larger than the unstable model. For instance, to use MRT for $Re = 2500, Wo = 4$, a lattice size of 20 would probably be required. Then the execution time of this run would be larger than that of the

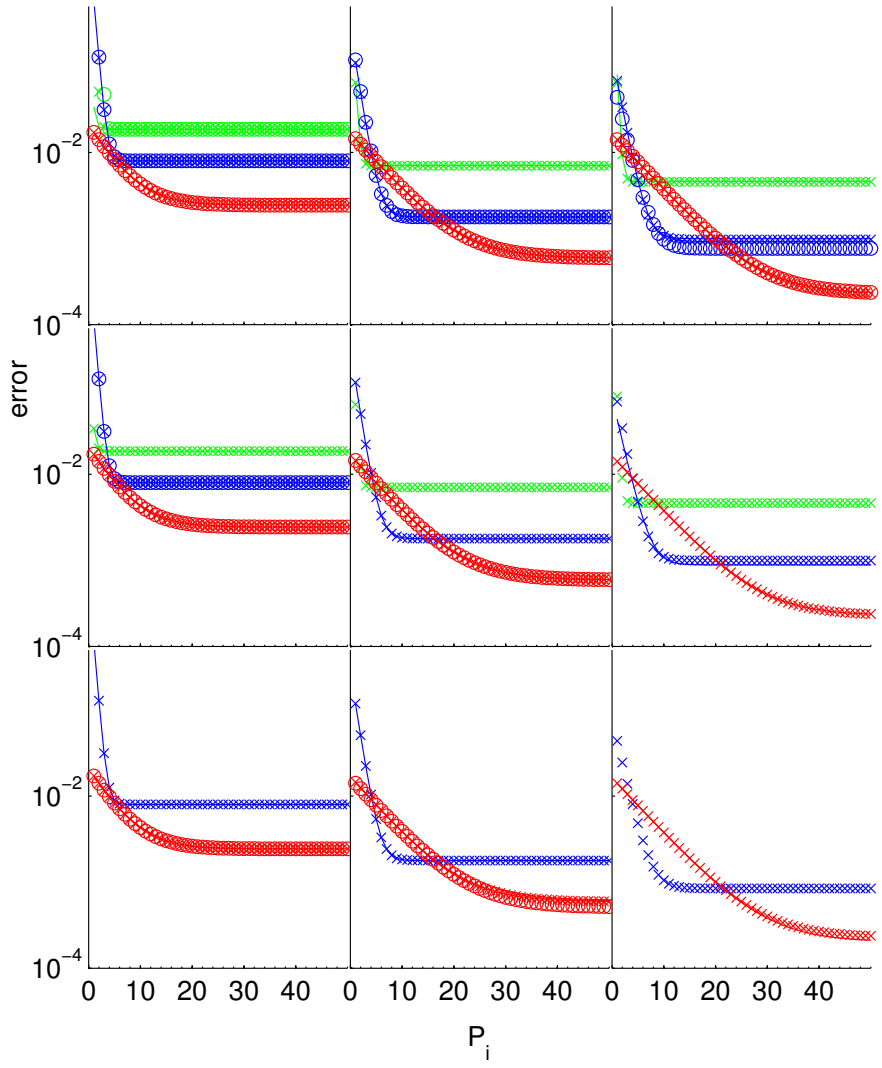

Figure 4.15: The error (y-axis) versus the period number P (x-axis) for the LBGK with, from top to bottom $Re = 500, 1600, 4000$, from left to right $L = 10, 20, 30$, coloring green, blue, red for $Wo = 4, 8, 16$ and marks "o", line, "x" for $U = 0.01, 0.05, 0.1$, respectively.

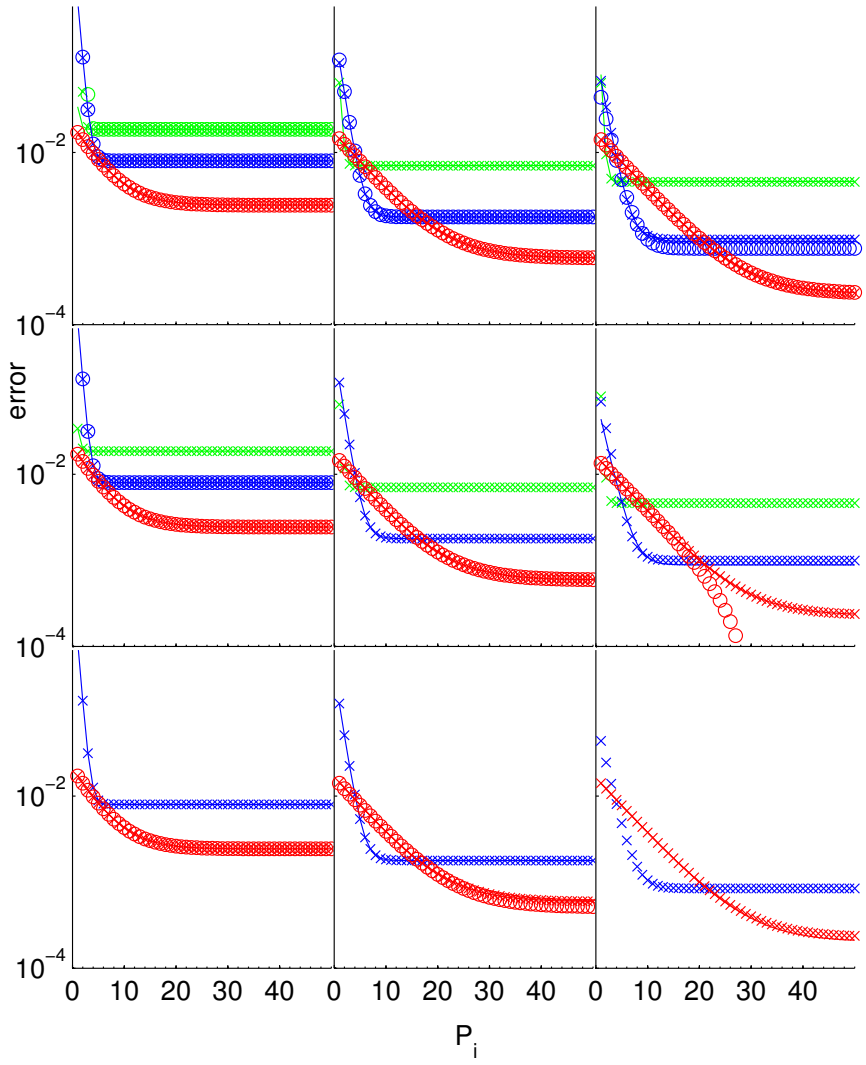

Figure 4.16: The error (y-axis) versus the period number P (x-axis) for the ELBM with, from top to bottom $Re = 500, 1600, 4000$, from left to right $L = 10, 20, 30$, coloring green, blue, red for $Wo = 4, 8, 16$ and marks "o", line, "x" for $U = 0.01, 0.05, 0.1$, respectively.

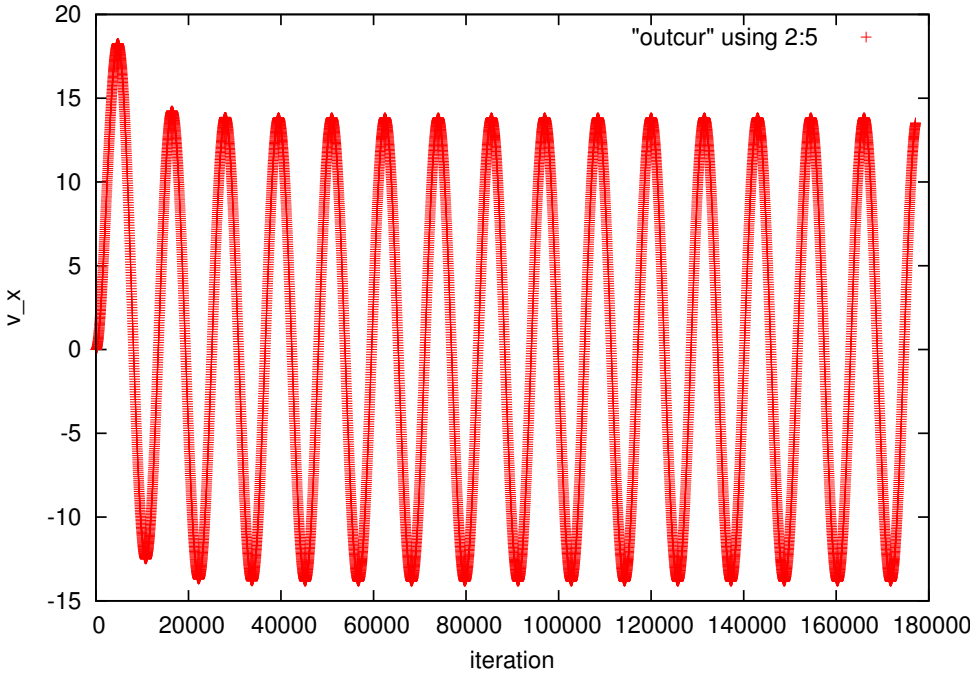

Figure 4.17: The total impuls in the direction of the flow of a LBGK simulation (for $Re = 256, WO = 4, L = 16, P = 11504$).

ELBM even though it has a higher T_{exni} and thus the ELBM can provide a faster solution, up to some accuracy (because the ELBM has half the accuracy of MRT for the same L and thus even lower for a lower L).

In the bottom plot of Figure 4.18 the highest obtained accuracy for each Wo, Re combination is drawn. The scale of the color is logarithmic because of the large range in accuracy (a factor of 100 between the largest and smallest error as defined by (3.7) in Section 3.1.5). The accuracy grows as the Womersley number grows.

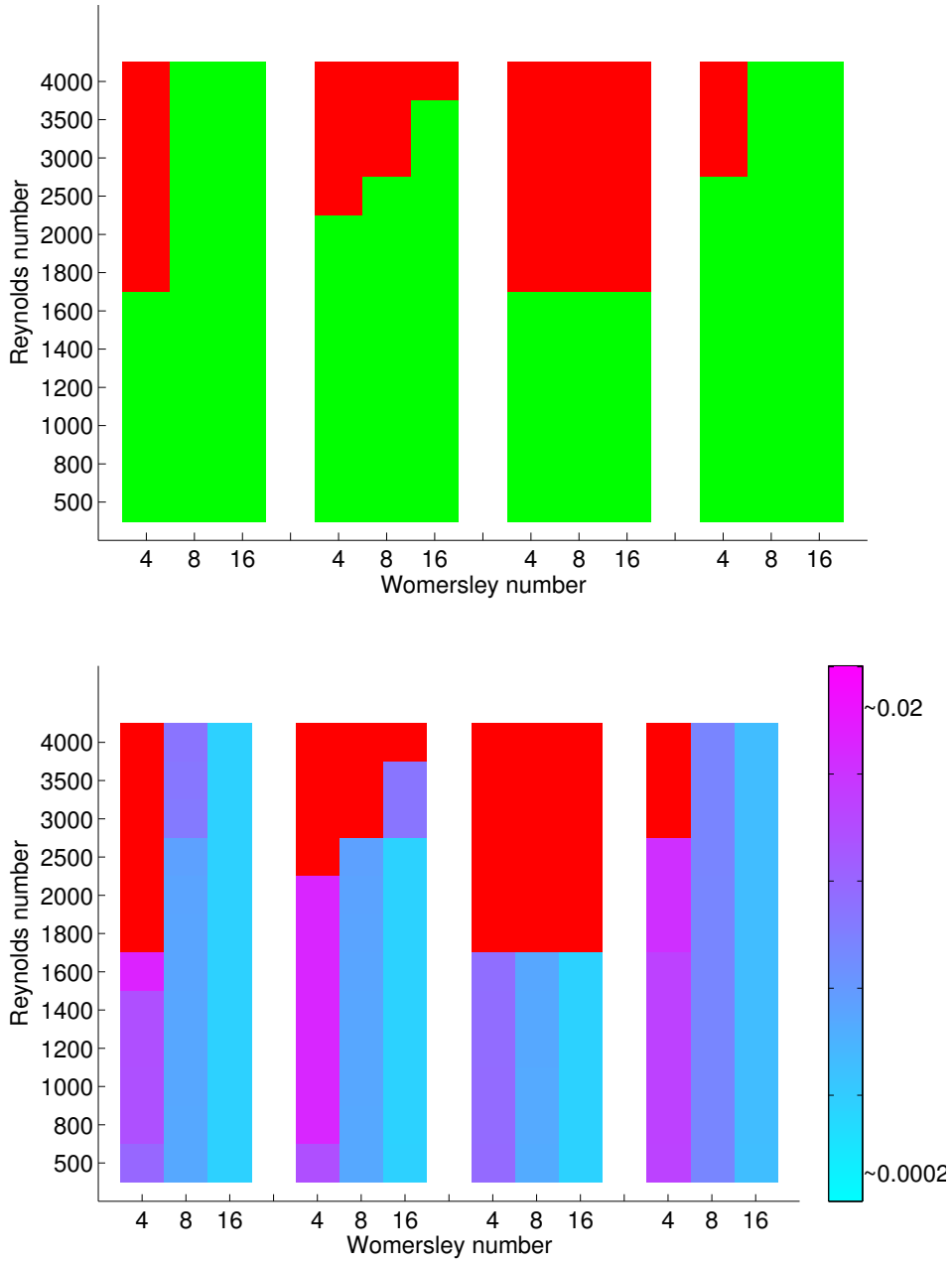

Figure 4.18: Grid plots of the best results of the Womersley parameter sweep. In the top plot, when at least one combination of U, L was successfull for the given Re, Wo, its corresponding square is colored green, else it is colored red. In the bottom plot the accuracy is also plotted, the cooler the color, the higher the accuracy. For each Re, Wo combination the highest accuracy is used. Grouped from left to right are LBGK, MRT, MRTJ and ELBM.

CHAPTER 5

Discussion

The Results section (4) calls for some explanations and comparisons to the literature, which will be given in this chapter.

Parameter sweeping As reported in the Results section (4.3) the error increase for increasing Mach number or maximum lattice speed U is negligible in the used parameter sweep ($U \in (0.01, 0, 05, 0.1)$). It is however reported in [6, 11] that for Womersley flows the Mach number can become even bigger than one without causing the simulation to break down. This is due to the relatively high geometrical symmetry of the problem and the fact that the effective speed of the flow in Womersley flows is only maximal part of the time. Thus to properly draw conclusions on the quality of the models as a function of U, its range should be larger, maybe up to $U = 1.5$ even.

The Womersley parameter sweep was in part done to find the optimal settings for each model for varying Reynolds and Womersley numbers. It was however found that all three models performed best with as high lattice speed as possible, in the used parameter range. Thus the only parameter left to distinguish the models was the lattice size. As the error changed dramatically as a function of this parameter and only three different sizes were used, it was impossible to find lattice sizes at which one model had a different lattice size but the same accuracy, apart from the ELBM versus the MRT and LBGK as the errors for $L = 20$ of ELBM were relatively close to those for $L = 10$ of MRT and LBGK. The computation time for ELBM was however also much larger for this bigger L and thus no "optimization" was achieved by varying the lattice size. Hence this optimization which is similar to that of [9] could not be performed. A more fine-grained parameter sweep would be necessary, which is beyond the scope of this work.

Sound waves and MRT As has been shown in the Poiseuille Section, 4.1.1, of the Results Chapter, there is a ~~bizarre~~ relation between unstableness of the MRT and MRTJ and Reynolds number for small lattice sizes. The instabilities were almost always of the nature of sound waves. In the literature the high sensitivity of the instabilities of the LBGK and MRT to the discretization used has been reported [34]. Depending on the geometry, the direction of the bodyforce and, for the MRT, the used relaxation parameters, sound waves will develop. This then clearly explains the large change of maximum Reynolds number of the MRT model for which no sound waves and hence instabilities occur.

There is not much literature to be found on three dimensional Poiseuille or Womersley LBM simulations. No ELBM versions were found, some two dimensional MRT versions exist and for the classical LBGK there is some. The problem is however that many different boundary conditions are possible, not only for the walls but also for the in- and outlet. Also this simulation uses a circular tube whereas some use a squared tube and some use planar Poiseuille flow. Hence comparing the results of this work to the literature is not straightforward. Luckily the LBGK and the supplied code has been thoroughly tested and thus this model could itself provide a semi-benchmark.

In article [36] the LBGK and the MRT are compared for plane Poiseuille flow in two dimensions. They report that the MRT becomes unstable for relatively low Reynolds numbers and that they have to solve this by rescaling some variables. The resulting MRT, the LBGK and their own new compressible MRT show no significant

	add/sub	multipy	divide	if	fabs	square root	power	logarithm
complete	3818	2724	33	77	37	9	81	891
eq & coll	162	135	3			9	81	

Table 5.1: The number of operations for an ELBM node. Complete means including an α calculation which needs 8 iterations, eq & coll stands for solely equilibrium and collision.

differences in error or stability for the plane Poiseuille flow, while for other flows there is a large difference. From this article it might be concluded that the MRT is not necessarily better for Poiseuille flows.

From the large differences in stability for the Poiseuille flow between the MRT and the MRTJ it can be concluded that the set of relaxation parameters used in the MRT largely influences the stability. Hence it might be possible to increase the stability beyond that of LBGK and ELBM by performing an optimization of these relaxation parameters for specific lattice sizes, Reynolds number and lattice speeds. Note however that such an optimization will not be simple and take a lot of compute time. Realizing that the ultimate goal of many applications is to use the LBM for varying system geometries and settings this optimization would be close to impossible. Hence even though MRT *might* be more stable, its applicability is very negotiable.

Execution time The execution time per node per iteration $T_{ex/node/iteration}$ or T_{exni}, listed in Section 4.3, Table 4.7, varies hugely per model. Many things have been written on this figure. As ELBM and MRT are the alledged successors of LBGK, their T_{exni} are usually compared to that of LBGK.

In the paper [21] the optimization of the MRT is discussed for three dimensions. They use three simple techniques to reduce the number of floating point operations needed per node update. The first is the use of hard coding the transformations (mapping the velocity densities distributions to moments and back), the second is the calculation of the equilibria in moment space and the third is the prevention of calculating subexpressions more than once. The second has been used in this project, the third partly and the first not. As a result the number of additions and multiplications is roughly respectively 760 and 720 whereas it could be as low as approximately respectively 150 and 50. Hence a large reduction in execution time would be possible. The numbers for LBGK they end up with are respectively about 100 and 50 whereas the supplied code (see Section 3.1.7) uses 150 and 110 respectively. Thus the MRT could gain tremendously in execution speed, but it would never be faster than the LBGK. The reduction in speed reported in [21] is *ca.* 17%.

The speed of ELBM is however much more difficult to compare as it has a varying number of operations per node due to the α calculation. First of all, this calculation does not always happen as the change in speed density distributions is not always large enough. Secondly the number of iterations needed in the Newton-Raphson search varies between 1 and 8. Apart from these variables the calculation uses many logarithms, power functions (including square roots) and fabs (the function returning the absolute value of its argument). The number of these operations is listed in Table 5.1.

Note that these numbers are for D3Q27 and not D3Q19 which thus needs more operations. The number of additions/substractions and multiplications for only the equilibrium and collision for a D3Q27 would approximately be 114 and 95 which is less than the LBGK but it also uses power functions, square roots and divisions. These the operations can take alot of compute time. It would be possible by hard coding the equilibria to substitute the power functions with multiplications and divisions. If then a very fast square root implementation would be used the speed of the ELBM without α calculations would be close to the LBGK speed.

Apart from the large number of operations needed take into account the fact that logarithms are expensive operations unless table lookups are used. This however requires fine tuning of the code and is not likely to occur. Also the conditional operations will cause the cpu pipeline to break slowing down the whole simulation. In paper [4] the compute time for the ELBM is 10 times larger than for the LBGK, which corresponds nicely with the measurements. The authors however state that

other methods which improve the LBGK, as the MRT which they refer to, need to solve a matrix of size $(b-M)^2$ with b from DdQb and M the number of conservation laws, which is very costly. Above though it has already been mentioned that no matrix multiplications are needed for the MRT.

A completely different figure for the computational overhead of ELBM is given in [1] where they claim in the conclusion, without any measurements shown, that the overhead is only 5-10% compared to "standard isothermal LBM". No explanation is given on how exactly they end up with this number, how many iterations are used on average in the α calculation, etc. In [16] they use an analytical function for the calculation of α whenever the difference between the density distribution and its equilibrium is small. Thus a large speed up can be realized "in 90% of grid points; thus, the nonlinear stability is achieved at virtually no computational overhead". The computation of this analytical function uses itself however tens of multiplications and additions and the 10% leftover still required the logarithms etc. In the authors opinion the "virtually no computational overhead" is therefore a gross underestimation.

In Section 4.2 the effect of α is reported. In summary the α calculation does not improve the ELBM for the Womersley or Poiseuille flows. If then this calculation would not be used, how would the ELBM perform compared to LBGK and MRT? The only thing that would change is its execution time which would approximate that of the LBGK. The model would still be more stable with still a lower accuracy. As a result ELBM could outperform LBGK and non-optimised (relaxation parameters) MRT when so-called Mach number annealing is used [5]. In the method the flow is relaxed first with a higher Mach number and then this number is rescaled to increase the accuracy. In the first periods there is a chance on instabilities and this is where the ELBM could perhaps outperform the LBGK and MRT. To investigate this will take some time and should be done in future work. Needless to say is that ELBM is better than MRT and LBGK for those Reynolds and Womersley combinations for with the other models have no stable solutions.

P_r as a function of Wo In the literature it can be found that after fourty periods Womersley simulations have converged [6, 11]. This is however contradicting the found relaxation speed-Womersley number relation (see 4.3.4) which shows that only about ten oscillations are needed for $Wo = 8$ and even less for lower Womersley numbers. As these results are quite firm (they are explainable and show changes of an order of magnitude) a possible explanation for this contradiction is that the error regression has not been studied for varying Womersley numbers in these works.

CHAPTER 6

Conclusions

In this chapter the conclusions drawn in the Results and Discussion sections 4 and 5 will be summarized.

The goal of this thesis was to compare the classical Lattice Boltzmann Method, LBGK, with two ~~modern~~ models, the Entropic LBM (ELBM) and the Multiple Relaxation Time LBM (MRT). In order to properly compare the three models a good understanding of their fundamentals and respective differences was needed. Thus a thorough literature studie was necessary, the result of which can be found in the Theory Section (2). This can function as a starting point for researchers new to the LBM field.

A good benchmark for blood flow simulations is the Womersley flow but before this flow could be used a Poiseuille flow analysis was needed. The results of these Poiseuille simulations were that the instabilities of intermediate Reynolds numbers could be explained with sound waves, which in turn explained why MRT became unstable at unexpectedly low Reynolds numbers. It became clear that many iterations were needed for these sound waves to develop, which agreed with the literature.

The ELBM showed a minimal error twice that of LBGK and MRT but was more robust. The MRT was very sensitive to the used relaxation parameter set.

The use of the reduced relaxation parameter (α) calculation of the ELBM showed no increase in stability, for neither the Poiseuille nor the Womersley flow. This was partly explained by the characteristics of Poiseuille and Womersley flows. If this calculation would not be used and the code would be optimised better, its execution time would approach that of LBGK and (optimised) MRT. With still the same higher robustness it could the outperform the two models.

As MRT was very sensitive to the used relaxation parameter set, finding the right set would probably enhance the model significantly. This search could however be a real challenge especially if it has to be done for various Reynolds and Womersley numbers and different geometries, which is usually the case for hemodynamics. Without this optimisation MRT does not outperform LBGK in a large part of the parameter sweeps used in this thesis, which is based in part on typical hemodynamical values.

Due to the simplicity of the LBGK and the fact that it provides a stable solution for the carotid artery and abdominal aorta, there is no reason to use ELBM or MRT for these two cases as long as no special tricks like Mach number annealing or the above MRT and ELBM improvements are applied.

Future Work There is still a lot to be done before a final conclusion can be drawn on the quality of the three models. Here is a small list:

- The execution speed of MRT can be increased by hard coding the matrix multiplication.
- The execution speed of ELBM can be increased by removing the α calculation.
- The Mach number annealing technique should be applied to make use of the higher stability of the ELBM [5].
- Instead of Bounce Back on Links a second order boundary condition should be implemented.

- The large difference in error for ELBM versus the other models should be investigated.
- Instead of body forces an in- and outlet pressure difference should be implemented and its effects measured.
- The optimization scheme described in [9] should be used to fully compare the models.

APPENDIX A

MRT

The specific equilibrium functions, transformation matrix $\mathbf{M}$ and relaxation matrix $\mathbf{\Omega}$ used for the MRT implementation are given here [21].
The moment equilibrium functions are

$$
\begin{aligned}
m_1^{eq} &= -11\rho + \frac{19}{/rho_0}\mathbf{j}\cdot\mathbf{j}, && \text{(A.1)}\\
m_2^{eq} &= \omega_\epsilon\rho + \frac{\omega_{\epsilon j}}{\rho_0}\mathbf{j}\cdot\mathbf{j}, && \text{(A.2)}\\
m_{4,6,8}^{eq} &= -\frac{2}{3}j_{x,y,z}, && \text{(A.3)}\\
m_9^{eq} &= \frac{1}{\rho_0}(2j_x^2 - (j_y^2 + j_z^2)), && \text{(A.4)}\\
m_{11}^{eq} &= \frac{1}{\rho_0}(j_y^2 - j_z^2), && \text{(A.5)}\\
m_{10}^{eq} &= \omega_{xx}m_9^{eq}, && \text{(A.6)}\\
m_{12}^{eq} &= \omega_{xx}m_{11}^{eq}, && \text{(A.7)}\\
m_{13}^{eq} &= \frac{1}{\rho_0}j_x j_y, && \text{(A.8)}\\
m_{14}^{eq} &= \frac{1}{\rho_0}j_y j_z, && \text{(A.9)}\\
m_{15}^{eq} &= \frac{1}{\rho_0}j_x j_z, && \text{(A.10)}\\
m_{16}^{eq} &= m_{17}^{eq} = m_{18}^{eq} = 0, && \text{(A.11)}\\
& && \text{(A.12)}
\end{aligned}
$$

with $\omega_\epsilon = \omega xx = 0$ and $\omega_{\epsilon j} = \frac{-475}{63}$.
Because the transformation matrix $\mathbf{M}$ is constructed in such a way that its row vectors (ϕ) are orthogonal to each other, its inproduct with its transpose ($M_{i,j}^\dagger \equiv M_{j,i}$) gives a diagonal matrix with elements equal to the size of each row squared:

$$M_{i,j}M_{j,i} = diag(||\phi_i||^2) = \Phi_{i,j}$$

and thus the following holds:

$$\mathbf{M}\mathbf{M}^\dagger\mathbf{\Phi^{-1}} = \mathbf{1}.$$

When multiplied with $\mathbf{M}^\dagger$ it gives

$$\mathbf{M^{-1}} = \mathbf{M}^\dagger\mathbf{Phi^{-1}}.$$

Thus the matrix $\mathbf{\Omega}$ becomes

$$\mathbf{\Omega} = \mathbf{M^{-1}}\hat{\mathbf{L}} = \mathbf{M}^\dagger\mathbf{Phi^{-1}}\hat{\mathbf{L}}.$$

The diagonal collision matrix in momemtum space $\hat{\mathbf{L}}$ has the following diagonal:

$$\hat{\mathbf{L}} \equiv diag(0, s_1, s_2, 0, s_4, 0, s_4, 0, s_4, s_9, s_{10}, s_9, s_{10}, s_{13}, s_{13}, s_{13}, s_{16}, s_{16}, s_{16}),$$

with $s_1 = 1.19, s_2 = s_{10} = 1.4, s_4 = 1.2, s_9 = s_{13}$, viscosity $\nu = \frac{1}{3}(\frac{1}{s_9} - \frac{1}{2})$.

$$
M = \begin{vmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-30 & -11 & -11 & -11 & -11 & -11 & -11 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\
12 & -4 & -4 & -4 & -4 & -4 & -4 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & -4 & 4 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & -1 \\
0 & 0 & 0 & -4 & 4 & 0 & 0 & 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & -4 & 4 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 \\
0 & 2 & 2 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -2 & -2 & -2 & -2 & -2 \\
0 & -4 & -4 & 2 & 2 & 2 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -2 & -2 & -2 & -2 & -2 \\
0 & 0 & 0 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -2 & -2 & 2 & 2 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 1
\end{vmatrix}
$$

Bibliography

[1] S. Ansumali, I. V. Karlin, and H. C. Öttinger. Minimal entropic kinetic models for hydrodynamics. *Europhysics Letters*, 63(6):798–804, September 2003.

[2] Santosh Ansumali and Iliya V. Karlin. Stabilization of the lattice Boltzmann method by the h theorem: a numerical test. *Phys. Rev. E*, 62(6):7999–8003, Dec 2000.

[3] Santosh Ansumali and Iliya V. Karlin. Entropy function approach to the lattice Boltzmann method. *J. of Stat. Phys*, 107(1,2):291–308, April 2002.

[4] Santosh Ansumali and Iliya V. Karlin. Single relaxation time model for entropic lattice Boltzmann methods. *Phys. Rev. E*, 65(5):056312, May 2002.

[5] A. M. Artoli, A. G. Hoekstra, and P. M. A. Sloot. Accelerated Lattice BGK Method for Unsteady Simulations Through Mach Number Annealing. *International Journal of Modern Physics C*, 14:835–845, 2003.

[6] Abdel Monim Artoli. *Mesoscopic computational haemodynamics*. PhD thesis, University of Amsterdam, October 2003.

[7] A.M. Artoli, A.G. Hoekstra, and P.M.A. Sloot. 3d pulsatile flow with the lattice boltzmann bgk method. *International Journal of Modern Physics C*, 13 (8):1119 – 1134, 2002.

[8] A.M. Artoli, A.G. Hoekstra, and P.M.A. Sloot. Mesoscopic simulations of systolic flow in the human abdominal aorta. *Journal of Biomechanics*, 39(5): 873–884, 2006.

[9] L. Axner, A. G. Hoekstra, and P. M. A. Sloot. Simulating time harmonic flows with the lattice Boltzmann method. *Physical Reviews E*, 75(3):036709–+, March 2007.

[10] L. Axner, J. Latt, A. G. Hoekstra, B. Chopard, and P. M. A. Sloot. Simulating Time Harmonic Flows with the Regularized L-Bgk Method. *International Journal of Modern Physics C*, 18:661–666, 2007.

[11] Lilit Axner. *High performance computational hemodynamics*. PhD thesis, University of Amsterdam, December 2007.

[12] M'hamed Bouzidi and Mouaouia Firdaouss Pierre Lallemand. Momentum transfer of a Boltzmann-lattice fluid with boundaries. *Physics of Fluids*, 13: 3452–3459, 2001.

[13] C. Cercignani, R. Illner, and M. Pulvirenti. *The mathematical theory of dilute gases*. Applied mathematical sciences, 106. Springer-Verlag, 1994.

[14] S. Chapman and T.G. Cowling. *The mathematical theory of non-uniform gases*. Cambridge mathematical library, third, with foreword by c. cercignani edition, 1990.

[15] Shiyi Chen and Gary D. Doolen. Lattice Boltzmann method for fluid flows. *Annual Review of Fluid Mechanics*, 30(1):329–364, 1998.

[16] S. S. Chikatamarla, S. Ansumali, and I. V. Karlin. Entropic lattice Boltzmann models for hydrodynamics in three dimensions. *Physical Review Letters*, 97(1): 010201, 2006.

[17] Shyam S. Chikatamarla and Iliya V. Karlin. Entropy and galilean invariance of lattice Boltzmann theories. *Physical Review Letters*, 97(19):190601, November 2006.

[18] J. A. Cosgrove, J. M. Buick, S. J. Tonge, C. G. Munro, C. A. Greated, and D. M. Campbell. Application of the lattice boltzmann method to transition in oscillatory channel flow. *Journal of Physics A: Mathematical and General*, 36 (10):2609–2620, 2003.

[19] Kandhai D., Koponen A., Hoekstra A., Kataja M., Timonen J., and Sloot P.M.A. Implementation aspects of 3d lattice-bgk: Boundaries, accuracy, and a new fast relaxation method. *Journal of Computational Physics*, 150: 482–501(20), April 1999.

[20] D. d'Humières. Generalized lattice Boltzmann equations. *Rarefied gas dynamics: theory and simulations*, 159:450–458, 2002.

[21] D. d'Humières, I. Ginzburg, M. Krafczyk, P. Lallemand, and L. Luo. Multiple-relaxation-time lattice Boltzmann models in three dimensions. *Phil. Trans. R. Soc. A*, 360:437–451, 2002.

[22] I. Ginzbourg and P. M. Adler. Boundary flow condition analysis for the three-dimensional lattice Boltzmann model. *Journal de Physique II*, 4:191–214, February 1994.

[23] I. Ginzburg. Generic boundary conditions for lattice Boltzmann models and their application to advection and anisotropic dispersion equations. *Advances in Water Resources*, 28:1196–1216, November 2005.

[24] Alexander N. Gorban and Iliya V. Karlin. Scattering rates versus moments: Alternative grad equations. *Phys. Rev. E*, 54(4):R3109–R3112, Oct 1996.

[25] Qian Y. H., D'Humières D., and Lallemand P. Lattice bgk model for navier-stokes equation. *Europhysics letters*, 17(6):479–484, January 1992.

[26] Xiaoyi He and Li-Shi Luo. Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation. *Phys. Rev. E*, 56(6): 6811–6817, Dec 1997.

[27] Li-Shi Luo Huidan Yu and Sharath S. Girimaji. Les of turbulent square jet flow using an mrt lattice Boltzmann model. *Computers & Fluids*, 35:957–965, September 2006.

[28] D. Kandhai, A. Koponen, A. G. Hoekstra, M. Kataja, J. Timonen, and P. M. A. Sloot. Lattice-Boltzmann hydrodynamics on parallel systems. *Computer Physics Communications*, 111:14–26, June 1998.

[29] Drona Kandhai. *Large scale lattic-Boltzmann simulations*. PhD thesis, University of Amsterdam, December 1999.

[30] I. V. Karlin, S. S. Chikatamarla, and S. Ansumali. Elements of the lattice Boltzmann method II: Kinetics and hydrodynamics in one dimension. *Commun. Comput. Phys.*, 2(2):196–238, April 2007.

[31] Iliya V. Karlin, Alexander N. Gorban, S. Succi, and V. Boffi. Maximum entropy principle for lattice kinetic equations. *Phys. Rev. Lett.*, 81(1):6–9, Jul 1998.

[32] A. M. Kogan. Derivation of grad's type equations and study of their relaxation properties by the method of maximization of entropy. *Journal of Applied Mathematics and Mechanics*, 29(1):130–142, 1965.

[33] Pijush K. Kundu and Ira M. Cohen. *Fluid Mechanics*. Academic Press, 2002.

[34] Pierre Lallemand and Li-Shi Luo. Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, galilean invariance, and stability. *Phys. Rev. E*, 61(6):6546–6562, Jun 2000.

[35] L. D. Landau and E.M. Lifshitz. *Fluid Mechanics*. Butterworth and Heinemann, second edition, 1987.

[36] Kannan N. Premnath and John Abraham. Three-dimensional multi-relaxation time (mrt) lattice-Boltzmann models for multiphase flow. *Physics letters A*, 359:564–572, 2006.

[37] Xiaowen Shan and Xiaoyi He. Discretization of the velocity space in the solution of the Boltzmann equation. *Phys. Rev. Lett.*, 80(1):65–68, Jan 1998.

[38] P.M.A. Sloot and A.G. Hoekstra. Modelling dynamic systems with cellular automata. *Handbook of dynamic system modeling*, pages 21.1–6, 2007.

[39] S. Succi. *The Lattice Boltzmann Equation For Fluid Dynamics and Beyond*. Oxford University Press, Oxford, 2001.

[40] Sauro Succi, Iliya V. Karlin, and Hudong Chen. Colloquium: Role of the h theorem in lattice Boltzmann hydrodynamic simulations. *Rev. Mod. Phys.*, 74 (4):1203–1220, Nov 2002.

[41] CA Taylor, TJR Hughes, and CK Zarins. Finite element modeling of blood flow in arteries. *Computer methods in applied mechanics and engineering*, 1-2 (158):155–198, May 1998.

[42] Charles A. Taylor and Mary T. Draney. Experimental and computational methods in cardiovascular fluid mechanics. *Annual Review of Fluid Mechanics*, 36(1):197–231, 2004.

[43] F. Tosi, S. Ubertini, S. Succi, H. Chen, and I.V. Karlin. A comparison of single-time relaxation lattice boltzmann schemes with enhanced stability. *International Journal of Modern Physic C*, 17(10):1375–1390, 2006.

[44] F. Tosi, S. Ubertini, S. Succi, and I.V. Karlin. Optimization strategies for the entropic lattice Boltzmann method. *Journal of Scientific Computing*, June 2006.

[45] J. R. Womersley. Method for the calculation of velocity, rate of flow and viscous drag in arteries when the pressure gradient is known. *Journal of Physiology*, 127:553–563, 1955.

[46] Donald P. Ziegler. Boundary conditions for lattice Boltzmann simulations. *Journal of Statistical Physics*, 71:1171–1177, 1993.

[47] Qisu Zou and Ziaoyi He. On pressure and velocity boundary conditions for the lattice Boltzmann bgk model. *Physics of Fluids*, 9(6):1591, Jun97.