NTNU

Norwegian University of
Science and Technology

Curved Boundary Conditions for the Lattice Boltzmann Method

Endre Joachim Mossige

Master of Science in Product Design and Manufacturing
Submission date: February 2011
Supervisor: Bernhard Müller, EPT
Co-supervisor: Joris Verschaeve, EPT

Norwegian University of Science and Technology
Department of Energy and Process Engineering

Problem Description

Background and objective.
The lattice Boltzmann method is a new numerical method of computational fluid dynamics (CFD). Conventional fluid solvers are based on the Navier-Stokes equations describing fluid motion based on a continuous picture of matter. The lattice Boltzmann method instead relies on discrete particles having an idealized movement on a lattice. Similar to the case of particles with continuous movement, it can be shown that the Navier-Stokes equations can be recovered from the statistical description of these particles. The numerical solution of these lattice Boltzmann equations has computational advantages over conventional solvers based on the Navier-Stokes equations. However, boundary conditions are still an issue for this method, since there is no clear picture what happens with these particles at walls or interfaces. In addition the method is only defined for equidistant Cartesian grids such that a special treatment of boundary conditions not aligned with the grid axes is necessary.
The master thesis will focus on curved boundary conditions of the lattice Boltzmann method.

The following questions should be considered in the project work:

1 Implementation of curved boundary conditions for the Lattice Boltzmann method.
2 Analytical derivation of a similar formulation for moving walls.
3 Implementation of this new formulation.
4 Verification of the boundary conditions.

Assignment given: 27. August 2010
Supervisor: Bernhard Müller, EPT

Norwegian University
of Science and Technology
NTNU

Department of Energy
and Process Engineering

NTNU

EPT-M-2010-45

MASTER THESIS

for

Endre Joachim Mossige
Fall 2010

Curved boundary conditions for the lattice Boltzmann method

Krumme grensebetingelser for lattice Boltzmann methoden

Background and objective.

The lattice Boltzmann method is a new numerical method of computational fluid dynamics (CFD). Conventional fluid solvers are based on the Navier-Stokes equations describing fluid motion based on a continuous picture of matter. The lattice Boltzmann method instead relies on discrete particles having an idealized movement on a lattice. Similar to the case of particles with continuous movement, it can be shown that the Navier-Stokes equations can be recovered from the statistical description of these particles. The numerical solution of these lattice Boltzmann equations has computational advantages over conventional solvers based on the Navier-Stokes equations. However, boundary conditions are still an issue for this method, since there is no clear picture what happens with these particles at walls or interfaces. In addition the method is only defined for equidistant Cartesian grids such that a special treatment of boundary conditions not aligned with the grid axes is necessary. The master thesis will focus on curved boundary conditions of the lattice Boltzmann method.

The following questions should be considered in the project work:

1- Implementation of curved boundary conditions for the Lattice Boltzmann method.
2- Analytical derivation of a similar formulation for moving walls.
3 -Implementation of this new formulation.
4 -Verification of the boundary conditions.

Within 14 days of receiving the written text on the diploma thesis, the candidate shall submit a research plan for his project to the department.

The thesis should be formulated as a research report with summary both in English and Norwegian, conclusion, literature references, table of contents etc. During the preparation of the text, the candidate should make an effort to produce a well-structured and easily readable reportreport. In order to ease the evaluation of the thesis, it is important that the cross-references are correct. In the making of the report, strong emphasis should be placed on both a thorough discussion of the results and an orderly presentation.

The candidate is requested to initiate and keep close contact with his/her academic supervisor(s) throughout the working period. The candidate must follow the rules and regulations of NTNU as well as passive directions given by the Department of Energy and Process Engineering.

Pursuant to "Regulations concerning the supplementary provisions to the technology study program/Master of Science" at NTNU §20, the Department reserves the permission to utilize all the results and data for teaching and research purposes as well as in future publications.

One – 1 complete original of the thesis shall be submitted to the authority that handed out the set subject. (A short summary including the author's name and the title of the thesis should also be submitted, for use as reference in journals (max. 1 page with double spacing)).
Two – 2 – copies of the thesis shall be submitted to the Department. Upon request, additional copies shall be submitted directly to research advisors/companies. A CD-ROM (Word format or corresponding) containing the thesis, and including the short summary, must also be submitted to the Department of Energy and Process Engineering

Department of Energy and Process Engineering, 23 June 2010

Olav Bolland
Department Manager

Bernhard Müller
Academic Supervisor

Research Advisor: Joris Verschaeve

Abstract

The lattice Boltzmann method is a modern method in computational fluid dynamics. Its primary use is the simulation of incompressible flows. It has computational advantages over conventional methods like the finite volume method. However, the implementation of boundary conditions is still an unsolved topic for this method. The method is defined on a Cartesian grid such that curved walls need special treatment as they are generally not aligned with the grid lines.

We investigated a number of straight and curved boundary conditions and performed four different benchmark tests to verify these. Based on a formulation for curved walls with no-slip from the literature, we showed that this method could be extended to simulate flows with arbitrary velocity boundary conditions. Our scheme conserved the second order accuracy of the lattice Boltzmann method in time and space.

Sammendrag

Lattice Boltzmann metoden er en moderne metode innen numeriske strømningsberegninger. Dens primære bruksområde er simulering av inkompressibel strømning. Den har numeriske fordeler i forhold til konvensjonelle løsere som endelig volum metoden. Imidlertid er implementeringen av grensebetingelser fortsatt et uløst emne for denne metoden. Lattice Boltzmann metoden er definert på et kartesisk nett slik at krumme vegger trenger spesiell behandling siden de generelt ikke sammenfaller med nettlinjene.

Vi undersøkte en rekke rette og krumme grensebetingelser og gjennomførte fire forskjellige rettesnorstester (benchmark tests) for å verifisere disse. Basert på en formulering for krumme vegger med heftbetingelse fra litteraturen viste vi at denne metoden kunne utvides til også å kunne simulere strømning med vilkårlige grensebetingelser for hastighet. Vårt skjema påviste å konservere lattice Boltzmann metodens andre ordens nøyaktighet i tid og sted.

Contents

Introduction

1 Theoretical Background **1**
- 1.1 Macroscopic quantities 4
- 1.2 From LBE to the Navier-Stokes Equations 5

2 Straight Boundary Conditions **8**
- 2.1 Jonas Lätt's Non-Local Boundary Condition 9

3 Curved Boundary Conditions **13**
- 3.1 Computing the Velocity on the Boundary Nodes 14
- 3.2 Computing the Rate of Strain on the Boundary Nodes 15
- 3.3 Computing the Density on the Boundary Nodes 15
- 3.4 Summary of the Algorithm 17

4 Verification **20**
- 4.1 Numerical Error 20
- 4.2 Flow in a Slit Driven by Gravity 21
- 4.3 Flow in a Slit Driven by a Pressure Gradient 22
- 4.4 Taylor-Couette Flow 24
- 4.5 Taylor-Vortex Flow 26

5 Conclusion **31**

Acknowledgments **32**

References **33**

List of Figures

1.0.1 The D2Q9 Lattice 2

2.1.1 Populations of Slitflow 10

3.4.1 Nodeclassification 18
3.4.2 Interpolationpoints 18
3.4.3 How to choose link $\vec{b}$ 19
3.4.4 How to calculate **S** on curved walls 19

4.2.1 Error of Gravity Driven Slit-flow 22
4.3.1 Velocity Error of Slit-Flow 23
4.3.2 Pressure Error of Slit-Flow 24
4.4.1 Velocity Profile of Taylor-Couette Flow 25
4.4.2 Velocity Error of Taylor-Couette Flow 26
4.5.1 Velocity and Pressure for Taylor-Vortex Flow 27
4.5.2 Velocity Error of Steady Taylor-Vortex Flow 29
4.5.3 Pressure Error of Steady Taylor-Vortex Flow 29
4.5.4 Velocity Error of Unsteady Taylor-Vortex Flow 30
4.5.5 Pressure Error of Unsteady Taylor-Vortex Flow 30

Introduction

The lattice Boltzmann method (LBM) has become popular in recent years to simulate fluid flows[1]. Conventional CFD-methods are generally based on a discretization procedure of the continuous partial differential equations that descibe fluid flows[2]. LBM instead uses a 'particle' model to decribe the dynamics of fluid flow. It can be shown via the so called Chapman-Enskog multiscale analysis[3], that LBM recovers the continuous Navier-Stokes equations[4]. Compared to conventional CFD-methods, LBM offers ease of programming of fluids near boundaries and interfaces, and is therefore suited to simulate e.g. multiphase flows[5] and flow in complex geometries[6].

The particles of lattice Boltzmann models live on the grid nodes of the computational domain. Because of their point-like structure, only two types of mechanism is sufficient to describe their dynamics. First is the local *collision* process, which corresponds to the collision these particles undergo at the grid nodes. The other type of mechanism is the non-local *streaming* process. Streaming corresponds to linear propagation of these particles between the nodes of the computational domain. The particle distribution function is the primary quantity in LBM. Macroscopic quantities, such as density and velocity, are recovered via statistical moments of this function.

In the first chapter of this thesis, we descibe LBM in more detail. References to books on the subject are given in the references[7][8][3] for the interested reader. In chapter 2, we move on by introducing some no-slip boundary conditions for straight walls. In chapter 3, we investigate a no-slip boundary condition for curved walls with movement.

This thesis treats both straight and curved boundaries with the main focus on the latter. We treat a general velocity boundary condition for curved walls, based on the article on curved no-slip boundary conditions by Verschaeve and Müller[9]. It proved to conserve the second order accuracy of LBM in time and space, for the benchmarks we tested in chapter 4. In addition, we verified some straight boundary conditions in this chapter. Finally, we sum up and conclude this thesis in chapter 5.

Chapter 1

Theoretical Background

This chapter treats the key elements of LBM necessary for the discussion in the rest of this thesis. We restrict ourselves to the lattice Boltzmann Bhatangar-Gross-Kross (LBGK) version of LBM[8][3]. The central quantity in the Boltzmann equation is the particle distribution function $f(t, \vec{x}, \vec{c})$, which is a function of time t, the position $\vec{x}$ and the particle speed $\vec{c}$. $f(t, \vec{x}, \vec{c})$ corresponds to the probability of finding a particle at position $\vec{x}$ with speed $\vec{c}$ at time t. Macroscopic quantities, such as fluid density and velocity, are the statistical moments of the distribution function.

As the name implies, lattice-Boltzmann methods solve the Boltzmann equation on a lattice. A lattice consists of the computational grid and an ensemble of velocity vectors and weights accociated with these vectors. The velocity vectors interconnect the grid nodes. They are restricted to certain values $c_i,\ i = 0, 1, ..., q-1$, where q is the number of possible velocities on the lattice. We treat only the D2Q9 lattice, see figure 1.0.1, having 9 possible velocities in 2 dimensions[7]. A necessary condition of LBM is that a particle travels exactly one grid spacing per time step. If Δx is the grid spacing and Δt is the time step, then

$$c = \frac{\Delta x}{\Delta t} \tag{1.0.1}$$

is the lattice constant. These constants are usually chosen as:

$$c = \Delta x = \Delta t = 1. \tag{1.0.2}$$

The speed of sound for the D2Q9 lattice is:

$$c_s = \frac{c}{\sqrt{3}}. \tag{1.0.3}$$

The neighboring nodes of a grid node with position $\vec{x}$ is thus $\vec{x} + \vec{c}_i$. The velocity vectors are chosen in such a way that the system is isotropic, meaning that the numerical solution to the macroscopic differential equation of interest does not depend to leading order on

the underlying grid structure. The magnitude of the velocity vector $\vec{c}_i$ is such that the particle propagates exactly the distance from one node to another in one time step. The lattice weights, t_i, adjust for the difference in length of the velocity vectors. For the D2Q9 lattice used throughout this thesis, the weights t_i are

$$t_i = \begin{cases} 4/9 & ,i = 0 \\ 1/9 & ,i = 1,2,3,4 \\ 1/36 & ,i = 5,6,7,8 \end{cases}$$

The set of vectors of the lattice is:

$$\vec{c}_i = \begin{cases} (0,0) & ,i = 0 \\ (1,0) & ,i = 1 \\ (0,1) & ,i = 2 \\ (-1,0) & ,i = 3 \\ (0,-1) & ,i = 4 \\ (1,1) & ,i = 5 \\ (-1,1) & ,i = 6 \\ (-1,-1) & ,i = 7 \\ (1,-1) & ,i = 8 \end{cases}$$

Note that $\vec{c}_0$ is the velocity vector of a particle at rest.

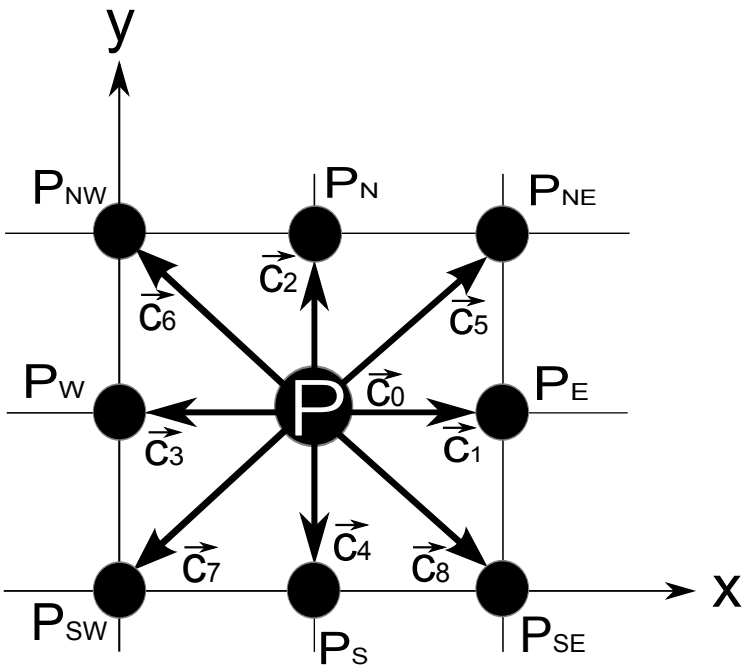


Figure 1.0.1: The D2Q9 lattice interconnecting grid node P to its neighboring nodes.

A physical quantity φ, e.g. length or velocity, is non-dimensionalized by $\varphi^* = \varphi/\varphi_{ch}$. Here, subscript $'ch'$ stands for *characteristic*. For instance, with $(N+1)$ points to resolve the length l^*, the grid resolution becomes $\delta x = l^*/N$. If l^* is chosen to be unity, then $\delta x = 1/N$. Since the lattice Boltzmann method is defined on a Cartesian grid, the resolution is the same in all spatial directions.

Since the velocity space is reduced to a finite set of velocity vectors in LBM, the continuous distribution function $f(t, \vec{x}, \vec{c})$ can be written as a set of discrete functions:

$$f(t,\vec{x},\vec{c}) = \begin{cases} f_0(t,\vec{x}) & \text{if} \quad \vec{c} = \vec{c}_0 \\ \vdots & \vdots \\ f_i(t,\vec{x}) & \text{if} \quad \vec{c} = \vec{c}_i \\ \vdots & \vdots \\ f_{q-1}(t,\vec{x}) & \text{if} \quad \vec{c} = \vec{c}_{q-1} \\ 0 & \text{otherwise} \end{cases} .$$

f_i is commonly referred to as a population and is responsible for carrying information from one node to the neighboring one. A neighboring node is connected to the initial node via the velocity vectors $\vec{c}_i$ of the lattice. Each f_i is carried along the direction of its associated $\vec{c}_i$. A set of q equations has to be solved for each f_i at each node to give the future distributions. The lattice Boltzmann equation reads:

$$f_i(t+1, \vec{x}+\vec{c}_i) = f_i(t,\vec{x}) + \Omega_i, \tag{1.0.4}$$

where Ω_i is the *collision operator*. Equation (1.0.4) can be divided into the collision and streaming step. We denote the populations entering a collision by *precoll* and the populations after collision by *postcoll*. The local collision step can then be written as:

$$f_i^{postcoll}(t,\vec{x}) = f_i^{precoll}(t,\vec{x}) + \Omega_i. \tag{1.0.5}$$

The *postcoll*-populations are then subject to the non-local streaming process, given by:

$$f_i^{poststream}(t+1, \vec{x}+\vec{c}_i) = f_i^{postcoll}(t,\vec{x}), \tag{1.0.6}$$

where the superscript *poststream* stands for 'post-streamed'. The superscript *precoll* will often be dropped for convenience.

In its complete form, the collision operator Ω_i is a non-linear integral operator. However, instead of using this complicated operator, the collision operator is modeled as a relaxation towards equilibrium. This is known as the BGK-collision:

$$\Omega_i = -\omega(f_i - f_i^{eq}), \tag{1.0.7}$$

where ω is the inverse of the collision relaxation time, i.e. the relaxation frequency. The

lattice Boltzmann equation (LBE) becomes:

$$f_i(t+1, \vec{x}+\vec{c}_i) = f_i(t, \vec{x}) - \omega(f_i - f_i^{eq}), \tag{1.0.8}$$

where the equilibrium distribution function f_i^{eq} is given by:

$$f_i^{eq} = t_i \rho \left(1 + \frac{1}{c_s^2}\vec{c}_i \cdot \vec{u} + \frac{1}{2c_s^4}\mathbf{Q}_i : \vec{u}\vec{u}\right). \tag{1.0.9}$$

In the special case $\vec{u} = \vec{0}$, f_i^{eq} is simply the product between the weights and the macroscopic fluid density. This is particularly useful for the calculation of the fluid density for walls at rest as will be shown in chapter 2. Eqn. (1.0.9) becomes in that case:

$$f_i^{eq} = t_i \rho^{wall}. \tag{1.0.10}$$

The tensor $\mathbf{Q}_i$ is defined as:

$$\mathbf{Q}_i = \vec{c}_i\vec{c}_i - c_s^2\mathbf{I} \tag{1.0.11}$$

where $\mathbf{I}$ is the identity matrix. Equation (1.0.9) can alternatively be written as:

$$f_i^{eq} = t_i \rho \left(1 + 3\vec{c}_i \cdot \vec{u} + \frac{9}{2}(\vec{c}_i \cdot \vec{u})^2 - \frac{3}{2}\vec{u}^2\right), \tag{1.0.12}$$

when c is unity.

1.1 Macroscopic quantities

Macroscopic quantities are linked to microscopic quantities via statistical moments of the distribution function:

$$\text{Density:} \quad \rho = \sum_{i=0}^{q-1} f_i \tag{1.1.1}$$

$$\text{Momentum density:} \quad \vec{j} \equiv \rho\vec{u} = \sum_{i=0}^{q-1} \vec{c}_i f_i \tag{1.1.2}$$

$$\text{Second-order tensor:} \quad \Pi = \sum_{i=0}^{q-1} \vec{c}_i\vec{c}_i f_i \tag{1.1.3}$$

The macroscopic velocity $\vec{u}$ is simply:

$$\vec{u} = \frac{1}{\rho}\vec{j} = \frac{1}{\rho}\sum_{i=0}^{q-1} \vec{c}_i f_i. \tag{1.1.4}$$

1.2 From LBE to the Navier-Stokes Equations

By means of the multiscale Chapman-Enskog expansion[10], it can be shown that the macroscopic quantities from section 1.1 obey the incompressible Navier-Stokes equations up to second order in time and space when compressible effects are negligible. The key idea is that the distribution functions can be expanded by a small parameter ϵ, identified as Kn, the Knudsen number[1]:

$$f_i = \sum_{n=0}^{nmax} f_i^{(n)} \epsilon^n. \tag{1.2.1}$$

The zeroth-order term of the expansion is the equilibrium distribution, $f_i^{eq} = f_i^{(0)}$. This term is responsible for the transport of information about ρ and $\vec{u}$ between the nodes of the computational grid. Information about their gradients is described by the first order distributions. Therefore, we identify the hydrodynamic terms of the Chapman-Enskog expansion as the zeroth and first order distribution functions. The remaining terms are thus neglected in the analysis. We take $f_i^{neq} = f_i - f_i^{eq} \approx \epsilon f_i^{(1)}$ for this reason. To avoid loss of mass during collision, the sum of the non-equilibrium parts has to be zero:

$$\sum_i f_i^{neq} = 0. \tag{1.2.2}$$

Conservation of momentum requires:

$$\sum_i \vec{c}_i f_i^{neq} = \vec{0}. \tag{1.2.3}$$

Conservation of mass and momentum are important for the stability of CFD-methods. Here, we refer to *local* conservation of these variables. Global conservation is not guaranteed, however, as one might lose or gain mass and momentum at boundaries[11].

The second-order tensor is linked to the rate-of-strain-tensor via the first-order distributions:

$$\Pi^{(1)} = \vec{c}_i \vec{c}_i f_i^{(1)} = -\frac{2c_s^2}{\omega} \rho \mathbf{S}. \tag{1.2.4}$$

where $\mathbf{S} = (\nabla \vec{u} + (\nabla \vec{u})^T)/2$ is the rate of strain tensor.

In the presence of an external source such as gravity, a correction term must be added prior to collision[12]:

$$\vec{j} = \sum_{i=1}^{q-1} \vec{c}_i \vec{f}_i = \underbrace{\sum_{i=1}^{q-1} \vec{c}_i f_i^{(0)}}_{\rho \vec{u}} - \frac{\vec{F}}{2}, \tag{1.2.5}$$

[1] $Kn = l_{mfp}/l_{ch}$, where l_{mfp} is the mean free path of the particles in the fluid

where $\vec{f}_i^{(0)}$ is the zeroth order moment of the distribution function. This result was obtained by means of a multiscale Chapman-Enskog analysis[10]. Introducing an external source can physically be interpreted as injecting momentum into the fluid[13]. Thus, it has to be corrected by a force term. The collision operator in this case takes the following form:

$$\Omega_i = \omega \left(f_i - f_i^{eq}\right) + Z_i, \tag{1.2.6}$$

where Z_i is the force term, given by:

$$Z_i = \left(1 - \frac{\omega}{2}\right) t_i \left(3\left(\vec{c}_i - \vec{u}\right) + 9\left(\vec{c}_i \cdot \vec{u}\right)\vec{c}_i\right) \cdot \vec{F} \tag{1.2.7}$$

when $c_s^2 = 1/3$. An alternative, however less accurate, force term is given by[12]:

$$Z_i = \frac{t_i}{c_s^2} \vec{c}_i \cdot \vec{F}. \tag{1.2.8}$$

The dimensionless formulation of the momentum and continuity equations for an isothermal and incompressible flow read:

$$\begin{aligned} \frac{\partial}{\partial t}\vec{u} \; + \; (\vec{u} \cdot \nabla)\vec{u} \;\; &= \;\; -\nabla p \; + \; \frac{1}{Re}\Delta\vec{u} \; + \; \vec{F} \\ \nabla \cdot \vec{u} \;\; &= \;\; 0. \end{aligned} \tag{1.2.9}$$

The pressure is linked to the density via the equation of state for an ideal gas:

$$p = c_s^2 \rho, \tag{1.2.10}$$

and the lattice-viscosity is linked to the relaxation parameter by:

$$\nu_L = c_s^2 \left(\frac{1}{\omega} - \frac{1}{2}\right), \tag{1.2.11}$$

which gives $0 < \omega < 2$. As $\omega \to 2$, numerical instabilities can arise as the viscosity becomes arbitrarily small[14]. As viscosity tends to dampen numerical oscillations in the solution, a very small viscosity allows these to grow freely which in turn gives numerical instabilities.

Equations (1.2.9) are solved with three error contributions by the lattice Boltzmann method: the spatial error, the temporal error and the error due to compressibility effects. They scale with δx^2, δt^2 and Ma^2, respectively, where Ma is the Mach number[2]. This means that the continuous equations are recovered up to second order in time and space if the error contribution from compressibility effects is negligible. Since the Mach number

[2] Ma=u/c_s, where u is the fluid velocity and c_s is the local speed of sound

itself scales with $(\delta t/\delta x)$, the time step has to be reduced as

$$\delta t \sim \delta x^2 \tag{1.2.12}$$

in order to obtain a second order accurate scheme. With this constraint on the time stepping, eqns.(1.2.9) are solved with second order accuracy in time and space.

The numerical viscosity is related to the Reynolds number by:

$$\nu_L = \frac{U_L N}{Re} = U_L N \nu, \tag{1.2.13}$$

where N is the spatial resolution, U_L the lattice velocity and ν the non-dimensional physical viscosity. For the benchmarks tests in chapter , we choose $\delta t = \delta x^2$ to suppress the influence from the compressible effects on the solution as much as possible. We choose the lattice velocity to be equal to the distance the populations propagate per iteration. That is, we choose a lattice velocity equal to one grid cell per time step. With this choice, the lattice velocity becomes

$$U_L = \frac{\delta t}{\delta x} = \delta x, \tag{1.2.14}$$

and the lattice-viscosity corresponds to the physical viscosity because

$$\nu = \frac{\nu_L}{U_L N} = \frac{\delta x^2}{\delta t|_{\delta t = \delta x^2}} \nu_L = \nu_L. \tag{1.2.15}$$

With the choice of units presented here, the last step is to calculate the relaxation frequency from eqn.(1.2.11). We adopt the present stategy for all the benchmark tests presented in chapter 4.

It is worth mentioning that, in the LBM-community, it is often of interest to impose a certain value for ω, e.g. when one wants to investigate the numerical stability of a LBM-scheme[10]. Since the method gets unstable as the value of the relaxation frequency approaches two, a LBM-scheme is said to have good stability properties if it is stable for relaxation frequencies close to two. With this choice of parameters, U_L is generally not equal to δx.

Choosing $\omega = 1$ is sometimes profitable because then only the equilibrium distributions, and none of the non-equilibrium distributions, enter the collision step. As instabilities usually originate at the boundaries, and f_i^{neq} is responsible for the propagation of information from the boundaries, choosing $\omega = 1$ provides stability.

Chapter 2

Straight Boundary Conditions

There exists two main classes of boundary conditions for the lattice Boltzmann method: bounce-back boundaries[7] and wet boundary conditions[1]. In the former case, nodes on the boundary of the computational domain are thought of as being dry nodes. On these nodes no collision or forcing is performed and the distributions are all bounced back[3] into their opposite directions after streaming. That is, the macroscopic quantities such as density are not computed on these nodes. Thus, all populations are known on the boundary nodes, next to the dry nodes, after streaming. With bounce-back boundaries, it is possible to achieve second order accuracy in time and space by defining the physical boundary to be midway between the boundary nodes and the dry nodes. This is however an *ad-hoc* approach and the bounce-back rule is considered only to be a first order accurate boundary condition. This is because only first order accuracy is achieved when bounce-back is implemented on the walls in LBM. The bounce-back rule works as a no-slip boundary condition and is popular because it assures exact conservation of mass and momentum on the boundary nodes and because it is easy to implement in a computer code.

The other group of boundary conditions is known as the wet boundary conditions. The name 'wet' comes from the fact that the macroscopic quantities are computed on all nodes, that is, the entire computational domain is wetted by the fluid. The boundary nodes for straight walls are in this case defined as the nodes being next to the boundary inside the computational domain. Opposed to the bounce-back boundaries, parts of the populations on a boundary node are unknown after streaming as they are streamed from populations stemming from dry nodes outside the computational domain. Therefore, the macroscopic density ρ and velocity $\vec{u}$ cannot be computed directly by eqn.(1.1.1), eqn.(1.1.2) and eqn.(1.1.4) on these nodes.

The simplest wet boundary condition is the equilibrium boundary condition, where the non-equilibrium distribution function is set to zero. Since only equilibrium distributions enter the collision step, important information about the gradients from the non-equilibrium distributions at the walls is missing. The solution in the entire domain suffers

from this, and the equilibrium boundary condition is only first order accurate for this reason. By this, we mean that the numerical solution for the entire domain will be only first order accurate.

We say that the order of a boundary condition is the order in which the global error is reduced with grid refinement. That is, when we double the number of sections to resolve a characteristic length in our computational domain, we expect the error to decrease by a factor four when using second order accurate boundary conditions. A boundary condition might only reduce the *local* error by first order. The *global* error, however, can still be reduced by second order when that current boundary condition is part of a LBM-algorithm. By 'error' and 'accuracy', we refer to the *global* quantities.

The accuracy of the boundary condition should not deteriorate the accuracy of the method. They should therefore be second order accurate for the lattice Boltzmann method. The wet boundary conditions discussed below are all second order accurate. We distinguish between local and non-local boundary conditions. Local boundary conditions get all the required information from the populations on the boundary node. The unknown populations after streaming are found from the known ones via closure relations. Therefore, all populations are known on a boundary node after streaming, and ρ and $\vec{u}$ are computed directly from eqn.(1.1.1), eqn.(1.1.2) and eqn.(1.1.4) as for the fluid nodes.

In general, local boundary conditions are accurate at low Reynolds numbers, but become unstable for high Reynolds number flows[11]. Non-local boundary conditions are slightly less accurate, but provide much better stability and are therefore better suited for high Reynolds number flows. We will use exclusively non-local boundary conditions as second order accurate boundary conditions for the benchmark tests in chapter 4. More specifically, we use the method derived by Lätt presented in the following.

2.1 Jonas Lätt's Non-Local Boundary Condition

The local tensor $\Pi^{(1)}$ appears in the hydrodynamic equations of the Chapman-Enskog expansion and is linked to the non-equilibrium distributions via eqn.(1.2.4). Parts of the populations in $\Pi^{(1)}$ are unknown at the boundary nodes after the streaming step, but these could be computed based on the known ones via closure relations.

The non-local 'Finite-difference velocity gradient method' by Lätt[11] abandons the idea of finding the unknown non-equilibrium distributions and replaces all distributions, unknown *and* known ones, on a boundary node prior to collision. This method exploits the fact that $\Pi^{(1)}$ is proportional to the rate-of-strain tensor and approximates f_i^{neq} by:

$$f_i^{neq} \approx -\frac{\rho t_i}{c_s^2 \omega} \mathbf{Q}_i : \mathbf{S}. \tag{2.1.1}$$

The rate of strain tensor $\mathbf{S}$ is computed by finite differencing schemes prior to collision

from $\vec{u}$. On straight walls, the velocity gradients of $\mathbf{S}$ normal to the wall are computed by asymmetric finite differences, and the gradients paralell to the wall are computed by central finite differences. The rate of strain tensor $\mathbf{S}$ gets particularly simple on straight walls with the no-slip condition as the diagonal elements, S_{xx} and S_{yy}, are zero. When the y-component of the velocity is zero, as for the slit flow treated below, the only non-zero components of $\mathbf{S}$ are $\partial u_x/\partial y$, which lie on the off-diagonal of $\mathbf{S}$. Finally, the distributions on the boundary nodes prior to collision are updated by:

$$f_i^{precoll}(\rho, \vec{u}, \mathbf{S}) = f_i^{eq}(\rho, \vec{u}) + f_i^{neq}(\rho, \mathbf{S}), \tag{2.1.2}$$

followed by the usual collision and streaming step. This procedure is called the regularized LBM. It is implemented for all benchmarks in chapter 4.

On straight walls, there are three unknown populations on a wall boundary node after streaming. A special case is when only one of the three macroscopic quantities ρ, u_x or u_y is unknown. We can then compute the unknown macroscopic quantity from the known ones and the known populations. This algorithm was derived by Zou and He[15].

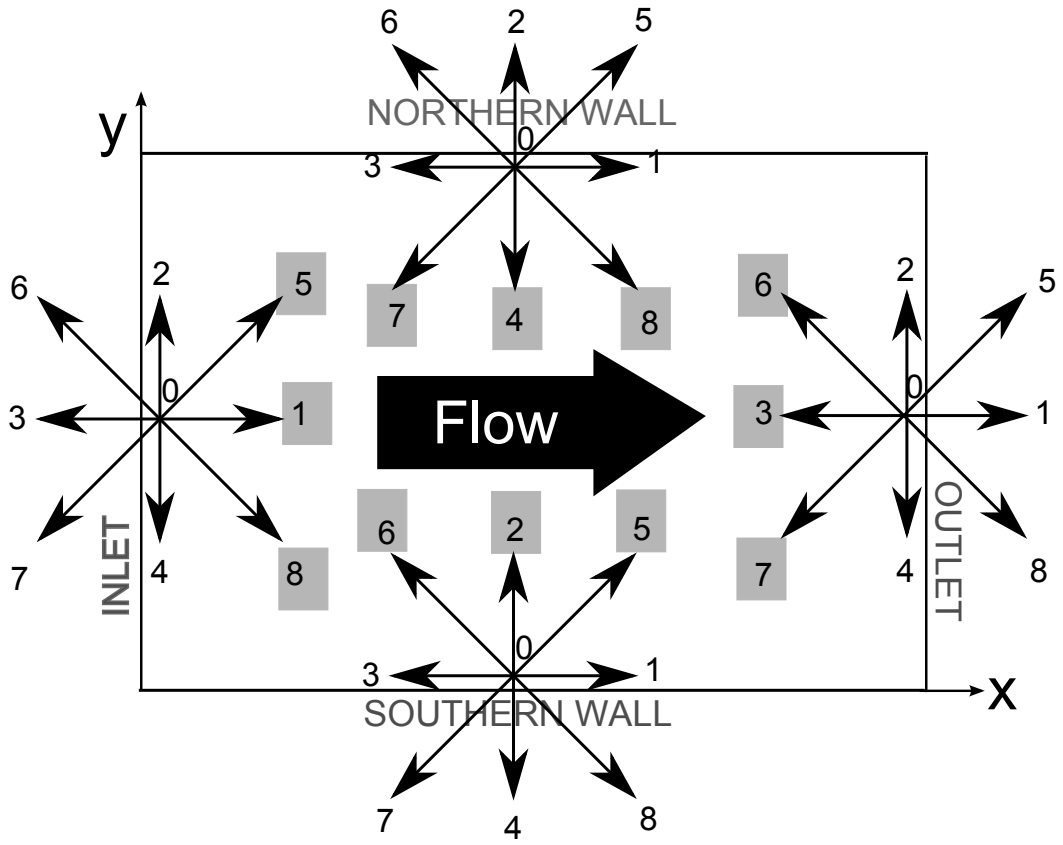


Figure 2.1.1: Figure showing pressure driven slit flow in section 2.1. The unknown populations on the boundary nodes after streaming are shaded. The inlet density, the wall density and the outlet x-velocity are the unknown macroscopic quantities.

Consider the pressure driven flow in a slit in figure 2.1.1. We impose a constant pressure at the outlet and a velocity profile at the inlet. The y-component of the velocity is set to zero. On the northern and southern walls, we impose the no-slip condition. We therefore have three unknown macroscopic quantities on the boundaries: the outlet x-velocity, the inlet density and the wall density. We solve for the unknown x-velocity at the outlet first. We combine eqn.(1.1.1) with eqn(1.1.2) and get:

$$\rho = \sum_i f_i = \underbrace{f_3 + f_6 + f_7}_{Unknown} + \underbrace{f_1 + f_2 + f_4 + f_5 + f_8 + f_0}_{Known\ after\ streaming} \tag{2.1.3}$$

and

$$\rho u_x = \sum_i c_{ix} f_i = -\underbrace{f_3 + f_6 + f_7}_{Unknown} \quad \underbrace{(f_1 + f_5 + f_8)}_{Known\ after\ streaming} \quad . \tag{2.1.4}$$

We solve for the x-velocity at the outlet:

$$u_{x,outlet} = -1 + \frac{f_2 + f_4 + f_9 + 2\,(f_1 + f_5 + f_8)}{\rho}. \tag{2.1.5}$$

With the non-local boundary condition of Lätt, the unknown populations, f_3, f_6 and f_7, are not computed via closure relations from the known ones as is the procedure with local boundary conditions. Instead, they are replaced by the 'pre-collision' populations in the regularization procedure given by eqn.(2.1.1)-(2.1.2).

After $u_{x,outlet}$ has been computed from eqn.(2.1.5), the velocity is known on all the boundary nodes. The next step is to compute the rate of strain tensor $\mathbf{S}$ from the velocity $\vec{u}$, which is used in eqn.(2.1.1) to approximate the non-equilibrium distributions.

In the particular case described above, the main flow is in the x-direction and the velocity component in the y-direction is zero on all nodes. The latter is not always the case, however. Placing a obstacle inside the channel, for instance, will induce periodic vortices downstream for a given range of the Reynolds number. This phenomenon is known as the Von-Karman vortex street. The y-velocity will be non-zero in this case. As the expression for u_y with the method described above will give more unknowns than there are equations, it is not possible to solve for this quantity at the outlet. It must therefore be set to zero. This will introduce a small local error contribution. However, this error is advected downstream with the flow and out of the domain. The global error will therefore not suffer from this approximation.

The next step is to compute the inlet density. We combine eqn.(1.1.1) with eqn.(1.1.2) and get:

$$\rho = \sum_i f_i = \underbrace{f_1 + f_5 + f_8}_{Unknown} + \underbrace{f_2 + f_3 + f_4 + f_6 + f_7 + f_0}_{Known\ after\ streaming} \tag{2.1.6}$$

and

$$\rho u_x = \sum_i c_{ix} f_i = \underbrace{f_1 + f_5 + f_8}_{Unknown} \quad - \quad \underbrace{(f_3 + f_6 + f_7)}_{Known\ after\ streaming} \quad . \tag{2.1.7}$$

The unknown density at the inlet is found by combining equation (2.1.6) and (2.1.7):

$$\rho_{inlet} = \frac{f_2 + f_4 + f_0 \ + 2\,(f_3 + f_6 + f_7)}{1 - u_x}. \tag{2.1.8}$$

We see from this that the inlet density is computed from the known post-streamed populations and the known velocity. As for the computation of the outlet velocity, the unknown populations, f_1, f_5 and f_8, are replaced in the regularization step given by eqn.(2.1.1)-(2.1.2).

Zou and He's algorithm is simple to implement in a computer program but does not work on curved boundaries. Also, it does not guarantee local mass conservation:

$$m^{postcoll} = m^{poststream}. \tag{2.1.9}$$

Local mass conservation requires that the sum of the populations streamed onto a boundary node from its fluid neighbors after streaming exactly balances the outgoing populations from the collision step on the node.

From eqn.(3.3.8) in chapter 3, it is possible to obtain a simple relation for the density for walls at rest[11]. The result, which aims at conserving mass locally on a boundary node, reads:

$$\rho_{wall} = 6m^{poststream}. \tag{2.1.10}$$

Here, $m^{poststream}$ is the sum of the known populations after streaming, i.e. the sum of the populations streamed from the fluid nodes onto a boundary node. From figure 2.1.1, we get:

$$m^{poststream}_{NORTHERN} = f_2 + f_5 + f_6 \tag{2.1.11}$$

$$m^{poststream}_{SOUTHERN} = f_4 + f_7 + f_8 \tag{2.1.12}$$

In general, internal corner nodes need special treatment because there are more unknown populations than there are equations on these nodes. In section 3.3, we present a method to approximate the density on curved boundary nodes which we also use to compute the density on corner nodes.

An alternative approach to approximate the corner density is to extrapolate the density by means of second-order extrapolation schemes from the neighboring fluid nodes[11].

Chapter 3

Curved Boundary Conditions

For curved boundaries, as for straight boundaries, parts of the populations on the boundary nodes are unknown after the streaming step. Curved boundaries differ from straight ones in that the number of unknown populations is different for each boundary node. Also, the distance from the boundary nodes to the wall is different for each node for curved boundaries.

One could define a boundary node in different ways. Some authors, like Guo et. al[16], define a boundary node as being part of the dry domain, but having at least one link to a fluid node. Here, we define a boundary node instead as being a node in the fluid domain (wet node) having at least one of its velocity vectors, $\vec{c}_i$ pointing onto a solid node (dry node). By this configuration, boundary nodes lie in-between fluid nodes where $\vec{u}$ is known after streaming and the wall where we impose the no-slip boundary condition. The velocity on a boundary node is therefore calculated by means of an *interpolation* scheme. We will take a closer look at this procedure in section 3.1.

We group the neighboring nodes of a boundary node into fluid nodes $\mathcal{F}$, boundary nodes $\mathcal{B}$ and solid nodes $\mathcal{S}$. The indices i of $\vec{c}_i$ of a boundary node pointing onto fluid nodes are then the elements of $\mathcal{F}$. In figure 3.4.1, we get

$$\mathcal{F} \;=\; \{6\} \tag{3.0.1}$$

The ensemble of indices pointing on to itself or other boundary nodes in figure 3.4.1 is

$$\mathcal{B} \;=\; \{0, 2, 3, 5, 7\}\,, \tag{3.0.2}$$

and the remaining populations are the solid indices $\mathcal{S}$. The entire ensemble of indices is then $\mathcal{M} = \mathcal{F} \cup \mathcal{B} \cup \mathcal{S}$. In figure 3.4.1 the solid indices are:

$$\mathcal{S} \;=\; \{1, 4, 8\}\,. \tag{3.0.3}$$

After streaming, the unknown populations on a boundary node are those streamed from

nodes belonging to the dry domain, outside the wet domain. Thus, these are the opposite of the solid nodes indices. In figure 3.4.1, these are

$$\mathcal{S}^{opposite} \;=\; \{2,3,6\}\,. \tag{3.0.4}$$

After streaming, the macroscopic quantities $\vec{u}$ and ρ can be computed on the fluid nodes by equations (1.1.1), (1.1.2) and eqn.(1.1.4) from the post-streamed populations. On the boundary nodes, parts of the post-streamed populations are missing. The macroscopic quantities $\vec{u}$, $\mathbf{S}$ and ρ needed for the regularization procedure can therefore not be computed by eqn.(1.1.1)-(1.1.4) on these nodes. We therefore need approximation schemes to compute $\vec{u}$, $\mathbf{S}$ and ρ on the boundary nodes.

3.1 Computing the Velocity on the Boundary Nodes

On the fluid nodes, $\vec{u}$ is found after streaming by eqn.(1.1.1) and eqn.(1.1.2). On the boundary nodes, the velocity is unknown because parts of the populations are streamed from dry nodes, which lie outside the computational domain. The wall velocity $\vec{u}_0$ is given by the velocity boundary condition there. The velocity $\vec{u}_P$ on boundary node P in figure 3.4.2 is found by interpolating the velocity between the known velocities at $H2$, $H1$ and $H0$. When the number of elements in $\mathcal{F} > 1$, we choose the link that forms the smallest angle with the wall normal. That is, we choose the link $\vec{b}$ by: $\vec{b} = \vec{c}_j / \left\|\vec{c}_j\right\|$, where j gives the maximum value of $\vec{c}_i \cdot \vec{n} / \left\|\vec{c}_i\right\|$, $i \in \mathcal{F}$. Here, $\vec{n}$ is the wall normal through point P, see figure 3.4.3. The approximation of $\vec{u}_P$ is performed with Lagrangian interpolation schemes, given by

$$\vec{u}_P = \vec{u}_0 l_0 \;+\; \vec{u}_1 l_1 \;+\; \vec{u}_2 l_2, \tag{3.1.1}$$

where $\vec{u}_0$, $\vec{u}_1$ and $\vec{u}_2$ are the velocities at points $H0$, $H1$ and $H2$, respectively. The interpolation polynomials of the distances b_{H0}, b_{H1} and b_{H2} evaluated at point P are given by $l_i, i = 0, 1, 2$:

$$l_0(b_P) \;=\; \frac{(b_P - b_{H1})(b_P - b_{H2})}{b_{H1} b_{H2}}, \tag{3.1.2}$$

$$l_1(b_P) \;=\; \frac{b_P(b_P - b_{H2})}{b_{H1}(b_{H1} - b_{H2})}, \tag{3.1.3}$$

$$l_2(b_P) \;=\; \frac{b_P(b_P - b_{H1})}{b_{H2}(b_{H2} - b_{H1})}, \tag{3.1.4}$$

where b_{Hi}, $i = 0, 1, 2, P$ is the distance from node Hi to the wall. By this definition of b_{Hi}, we have $b_{H0} = 0$.

3.2 Computing the Rate of Strain on the Boundary Nodes

Having approximated the velocity on the boundary nodes, we compute the rate of strain. We define nodes intersected by the wall as being dry nodes, thus b_p is always positive and central finite differences could be used both for the derivatives in x, and for the derivatives in y, of the velocity $\vec{u}$ on the boundary nodes, see figure 3.4.4. The nodes used to compute the velocity gradients at node P are the four neighbors P_N, P_S, P_E and P_W, where the subscripts stand for the four cardinal points relative to P. If the wall intersects the line interconnecting P with $P_{N,S,E,W}$, then the neighboring velocity takes the velocity at the intersection, that is, the wall velocity.

It would be appropriate to perform a weighting procedure between the velocities that enter the velocity gradient computation. However, this is not done for the benchmarks tested in chapter 4. The velocity gradients at P are computed by:

$$\frac{\partial \vec{u}}{\partial x} \approx \frac{\vec{u}_E - \vec{u}_W}{\left\| \overrightarrow{P_W P_E} \right\|}, \tag{3.2.1}$$

and

$$\frac{\partial \vec{u}}{\partial y} \approx \frac{\vec{u}_N - \vec{u}_S}{\left\| \overrightarrow{P_S P_N} \right\|}. \tag{3.2.2}$$

Expanding eqn.(3.2.1)-(3.2.2) by Taylor series yields only a first order scheme. However, approximating the rate of strain tensor $\mathbf{S}$ by these equations did not detoriorate the second order accuracy of LBM for the benchmarks in chapter 4.

3.3 Computing the Density on the Boundary Nodes

We present here an algorithm to compute the density on corner nodes. It takes advantage of the fact that the distributions are separable in ρ, such that:

$$f_i^{eq} = \rho \underbrace{t_i \left(1 + \frac{1}{c_s^2} \vec{c}_i \cdot \vec{u} + \frac{1}{2c_s^4} \mathbf{Q}_i : \vec{u}\vec{u} \right)}_{g_i^{eq}}. \tag{3.3.1}$$

Similarly, for the non-equilibrium part:

$$f_i^{neq} \approx \rho \underbrace{\left(-\frac{t_i}{c_s^2 \omega} \mathbf{Q}_i : \mathbf{S} \right)}_{g_i^{neq}}. \tag{3.3.2}$$

Thus, g is not a function of ρ, and eqn.(3.3.1) and eqn.(3.3.2) can be written as:

$$f_i^{eq}(\rho, \vec{u}) = \rho g_i^{eq}(\vec{u}) \tag{3.3.3}$$

$$f_i^{neq}(\rho, \mathbf{S}) = \rho g_i^{neq}(\mathbf{S}). \tag{3.3.4}$$

That is, as soon we know $\vec{u}$ and $\mathbf{S}$, then g_i^{eq} and g_i^{neq} can be computed directly. The ensemble of known indices after streaming is the opposite fluid indices plus the opposite boundary indices:

$$\mathcal{KN} = \mathcal{F}^{opposite} \cup \mathcal{B}^{opposite}. \tag{3.3.5}$$

The known indices could alternatively be chosen only to include the populations streamed from the fluid nodes:

$$\mathcal{KN} = \mathcal{F}^{opposite}. \tag{3.3.6}$$

Eqns.(3.3.3)-(3.3.6), together with assuming that the mass of the post-streamed populations approximately equals the mass entering a collision:

$$m^{poststream} \approx m^{precoll}, \tag{3.3.7}$$

yields an approximate formulae for the corner or boundary node density:

$$\rho \approx \frac{\sum_{i \in \mathcal{KN}} f_i^{poststream}}{\sum_{i \in \mathcal{KN}} g_i^{precoll}} \tag{3.3.8}$$

where $g_i^{precoll} = g_i^{eq} + g_i^{neq}$. Choosing $\mathcal{KN}$ as in eqn.(3.3.5) is in line with the definition of a wet boundary as it includes the populations streamed from the boundary nodes in the mass budget. It has been observed, however, that this approach can give rise to numerical instabilities[17] for reasons that are beyond the scope of this work. Thus, we choose $\mathcal{KN} = \mathcal{F}^{opp}$ for the benchmark tests in chapter 4. This gives $\mathcal{KN} = 8$ in figure 3.4.1.

3.4 Summary of the Algorithm

We conclude this chapter by summing up the algorithm:

1: Determine the direction of link $\vec{b}$ based on the angle the possible candidates for $\vec{b}$ form with the wall normal.

2: Approximate the velocity $\vec{u}$ on the boundary nodes by interpolation schemes.

3: Approximate the rate of strain tensor $\mathbf{S}$ from $\vec{u}$ by central finite difference stencils.

4: Compute the density ρ on the boundary nodes locally.

5: Apply the collision and streaming step on all nodes in that order.

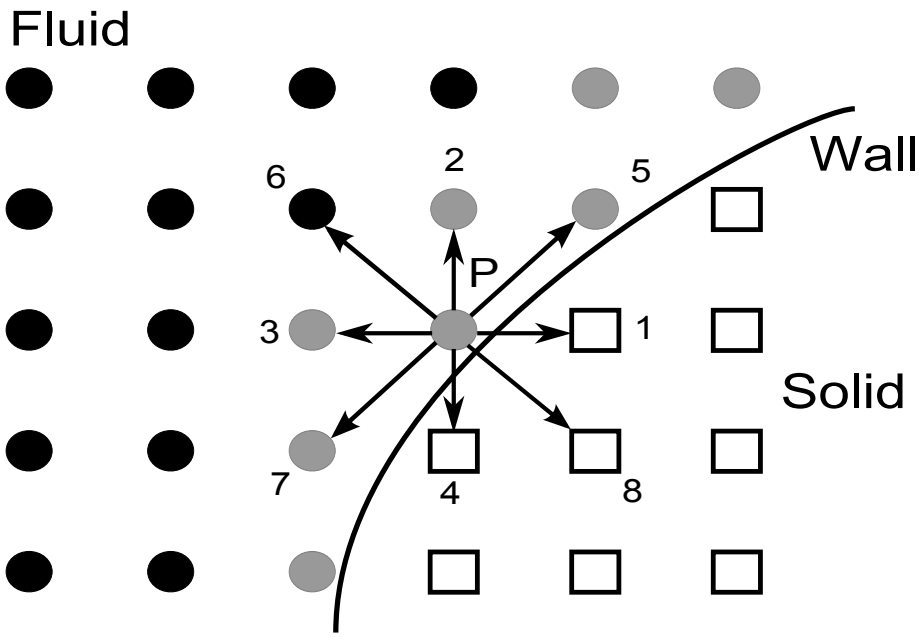


Figure 3.4.1: Close up look of boundary node P and its neighbors. The ensemble of fluidnodes is $\mathcal{F} = \{6\}$, the ensemble of boundary nodes $\mathcal{B} = \{0, 2, 3, 5, 7\}$, and the ensemble of solid nodes is $\mathcal{S} = \{1, 4, 8\}$. The boundary nodes are shaded grey.

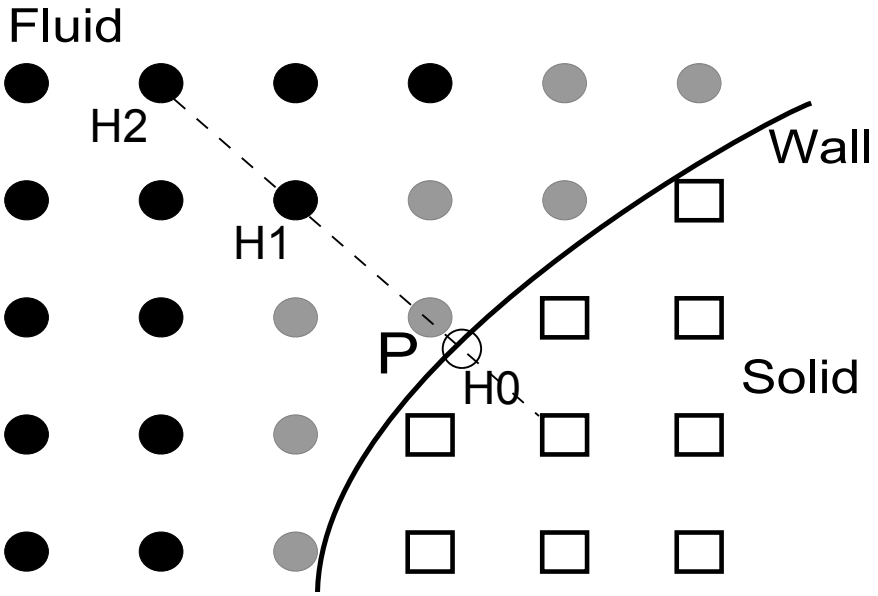


Figure 3.4.2: The velocity at boundary point P is approximated by an interpolation polynomial interpolating the velocity at $H0$, $H1$ and $H2$ along the link.

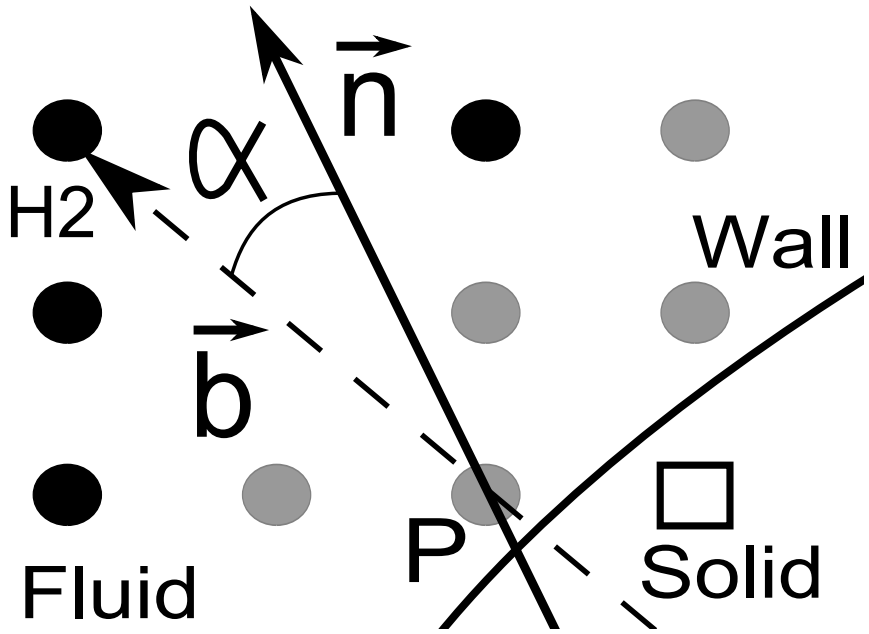


Figure 3.4.3: When there are multiple possible links to choose between, we choose the link $\vec{b}$ that forms the smallest angle α with the wall normal $\vec{n}$.

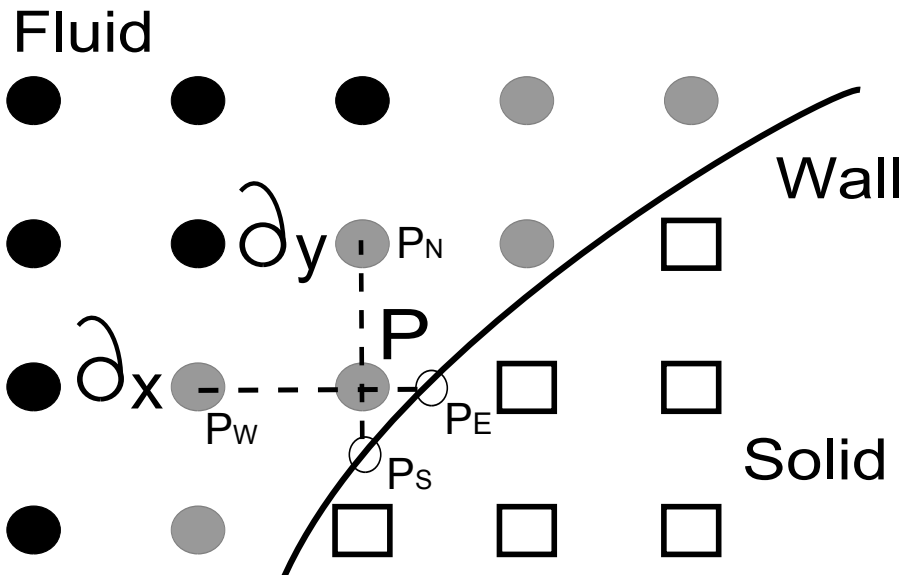


Figure 3.4.4: The figure shows how to compute **S** on boundary node P. P_N,P_S,P_E and P_W indicate the four cardinal points relative to P. Derivatives in x , ∂_x, are computed along the horizontal line, and derivatives in y, ∂_y, along the vertical line.

Chapter 4

Verification

We present in this chapter results from four different benchmark test cases. The two first are meant to verify the boundary conditions for straight walls: flow in a slit driven by gravity and flow in a slit driven by a pressure gradient. The last two benchmarks aim to verify the curved boundary conditions. They are: Taylor-Couette flow in-between two rotating cylinders, and steady and time-periodic Taylor-vortex flow. All flows are steady except from the last one.

4.1 Numerical Error

For incompressible flows, the error is always computed after the compressible effects have decayed. The spatial error contribution is computed for steady flows such that the temporal error can be neglected. The lattice-Boltzmann method is defined on a Cartesian grid. Therefore the spatial resolution is the same in both directions i.e. $\delta x = \delta y = 1/N$ for two dimensional problems. For steady flows, the l_2 norm of the spatial error in velocity is computed by:

$$\epsilon(N_I)_{l_2} = \sqrt{\frac{\sum_{(i,j)\in N_I} \left|\vec{u}(\vec{x})_{num} U_L^{-1} - \vec{u}^*(\vec{x})\right|^2}{N_I}} \leq C N_I^{-\kappa}, \tag{4.1.1}$$

where N_I is the ensemble of sections in the domain, C is an error constant and κ is the order in which the error is decreased. With the refinement strategy presented in section 1.2, we unscale the numerical velocity $\vec{u}_{num}$ by the inverse lattice velocity U_L^{-1}. $\vec{u}^*$ is taken as the non-dimensional solution. The pressure error is found in a similar way.

We compute the error of a cross-section by means of the l_1-norm instead. When, for instance, we want to compute the error of a velocity profile in a slit and want to minimize the influence from the inlet or outlet on the global error, we choose the l_1-norm. We compute the l_1-norm by:

$$\epsilon(N_I)_{l_1} = \frac{1}{N_I} \sum_{(i,j)\in N_I} |\phi(\vec{x})_{num} - \phi^*(\vec{x})| \leq C N_I^{-\kappa}, \tag{4.1.2}$$

where N_I is the ensemble of sections to resolve the cross-sectional length and ϕ is ρ, u or v. When we compute the l_1-norm, we must also unscale the numerical quantities by the lattice velocity.

4.2 Flow in a Slit Driven by Gravity

Both the source term and the boundary conditions get particularly simple for this benchmark, which we solve on a square computational domain. The gravitational forcing term, which drives the flow, is in the x-direction. The domain is unbounded in x. Numerically, this is represented by periodic boundaries on the inlet and on the outlet (eastern and western walls). On the northern and southern walls, we impose a no-slip condition which we implement with bounce-back, the equilibrium boundary and finally with the second order finite-difference boundary condition of Lätt. In the latter two cases, we compute the wall density after streaming by eqn.(2.1.10). The solution of the incompressible Navier-Stokes equations for this flow is:

$$u_x(y) = \frac{4u_{max}}{h^2}\,(h-y)\,y, \tag{4.2.1}$$

where u_{max} is the maximum velocity which occurs midway between the walls at $y = h/2$. With $u_{ch} = u_{max} = 1$ and $l_{ch} = h = 1$, the non-dimensional solution is:

$$u_x^*(y^*) = 4\,(1-y^*)\,y^*, \tag{4.2.2}$$

with the source term:

$$\vec{F}_x^* = \frac{8}{Re}. \tag{4.2.3}$$

The Reynolds number is fixed to $Re = 10$ for all simulations. The relaxation frequency becomes:

$$\omega = \frac{1}{\frac{3}{Re} + \frac{1}{2}} = \frac{5}{4}. \tag{4.2.4}$$

The influence of the force term is corrected in the collision step by eqn.(1.2.7).

From figure 4.2.1 we see that bounce-back and the equilibrium boundary conditions

both gave first order accuracy for the velocity, as expected. There is a discrepancy in the error constant C between the two obtained solutions of a factor 2 in favor of the equilibrium boundary condition.

The finite difference boundary condition, presented in section 2.1 gave the expected second order accuracy (see figure 4.3.1). All three simulations were performed with between 10 and 80 sections.

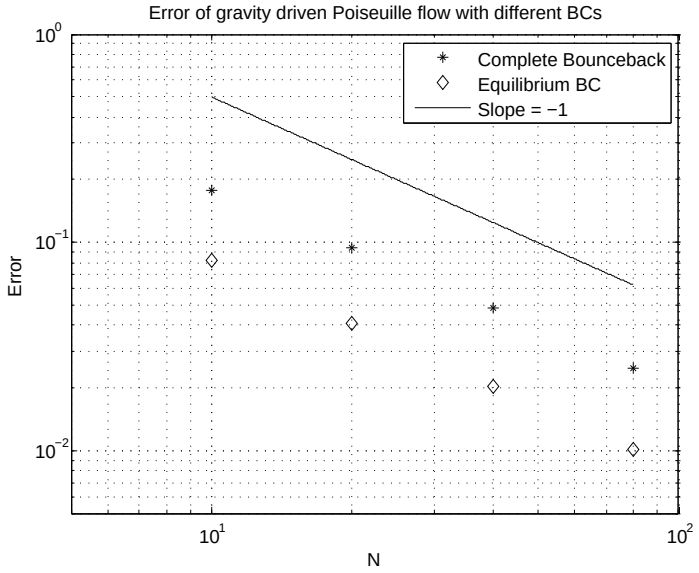


Figure 4.2.1: Error plot for the gravity driven flow in a slit with first order boundary conditions. The points obtained with bounceback is indicated by $*$ and the points obtained by the equilibrium boundary condition is indicated by $\diamond$. The solid line has slope -1.

4.3 Flow in a Slit Driven by a Pressure Gradient

The flow in section 4.2 was driven by gravity. We now choose to let a pressure difference drive the flow instead. The Reynolds number is $Re = 10$ and $\omega = 5/4$ as for the gravity driven flow. This benchmark is also solved numerically on a square domain. We impose the no-slip condition on the walls using Lätt's finite-difference boundary condition. The velocity field is given by eqn.(4.2.2) with pressure gradient:

$$\nabla^* p^* = -\frac{8}{Re}. \tag{4.3.1}$$

The Poiseuille flow profile given by eqn.(4.2.2) is the inlet boundary condition, and a constant zero pressure is the outlet boundary condition. We use Zou and He's method to compute the unknown inlet pressure and outlet x-velocity. The resulting equations are eqn.(2.1.8) for the inlet pressure, and eqn.(2.1.5) for the x-velocity.

As expected, the error in velocity was decreased with second order. It has been shown[18] that the introduction of velocity boundary conditions leads to a decrease of the

order of accuracy of the pressure. Therefore, the pressure error is only decreased with first order. The results for velocity and pressure are presented in figure 4.3.1 and figure 4.3.2, respectively. We computed the error using the l_1-norm on a section mid-way between the inlet and the outlet. We observe that the pressure driven slit flow is more accurate than the one driven by gravity which is the same conclusion obtained by Guo and Zheng[16].

To study the influence of entry and exit effects on the solution, we simulated the pressure driven flow in-between two parallel plates of length $l_x = 2l^* = 2$ as well. The characteristic length is still the separation distance h. Numerically, we represent this by solving this benchmark test on a rectangle with length 2 and height 1. The analytical solution is the same as for the pressure driven flow solved on a square domain described earlier in this section.

The velocity and pressure are still recovered with second and first order accuracy, respectively, using the l_1-norm mid-way between the inlet and the outlet to compute the error. However, there was a discrepancy between the two error constants for the pressure of a factor 1.5 in favor of the slit flow with length 2. First, this is natural since the entry and exit effects introduce additional error, which have a relatively smaller influence on the pressure field in a longer domain. Second, the distance from the inlet and outlet to where the l_1-norm of the error was computed was twice as long for the long channel compared to the shorter one. The velocity error was decreased with approximately the same error constant for the short as for the longer channel. The results are presented in figure 4.3.1 and figure 4.3.2.

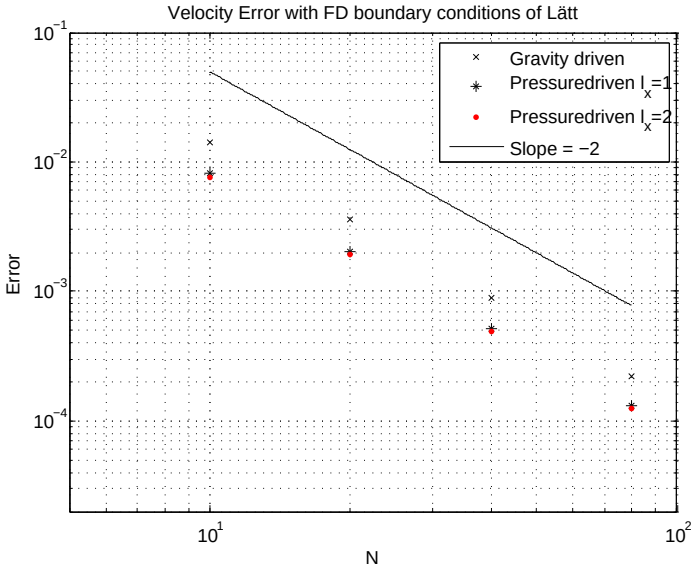


Figure 4.3.1: Error plot of velocity for the flow in a slit with second order boundary conditions of Lätt. The symbol **x** indicates gravity driven, $*$ indicates pressure driven with $l_x = 1$, and the dots $\cdot$ indicate pressure driven with $l_x = 2$. The solid line has slope -2.

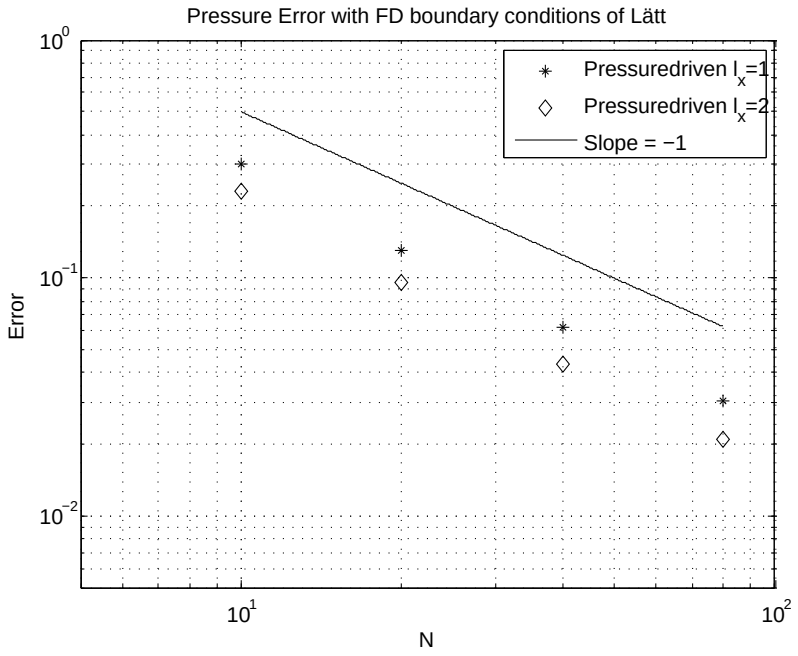


Figure 4.3.2: Error plot of the pressure for the pressure driven flow in a slit with Lätt's finite-difference boundary conditions. The symbol $*$ indicates pressure driven with $l_x = 1$, and the diamonds $\diamond$ indicate $l_x = 2$. The solid line has slope -1.

4.4 Taylor-Couette Flow

A classical benchmark case to verify no-slip curved boundary conditions for moving walls is flow between two circular cylinders. The inner cylinder, with radius r_0, rotates with a constant angular velocity, whereas the outer cylinder, with $R_0 > r_0$, is at rest. There exists an analytical solution to this problem, given in non-dimensionless units by

$$\vec{u}(r) = \frac{u_0 \beta}{1-\beta^2}\left(\frac{R_0}{r} - \frac{r}{R_0}\right)\vec{e}_\theta, \tag{4.4.1}$$

where $u_0 = r_0\Omega$ is the rotational speed of the inner wall and $\beta = r_0/R_0$. The superscript (*) is skipped for convenience. Note that the velocity field is independent of the angular position, i.e. the fluid particles experience no acceleration in this direction. The Reynolds number is Re$= u_0 (R_0 - r_0)/\nu$ and $\beta = (1/2)$. The Reynolds number is $Re = 10$ and $\omega = 5/4$. We impose a zero pressure at the outer cylinder wall, that is, $p(R_0)$=0. The pressure distribution takes the following form:

$$p^* = \frac{p(r)}{p_0} = \left(\frac{\beta}{1-\beta^2}\right)^2 \left(\frac{r^2}{R_0^2} - \frac{R_0^2}{r^2} - 4\,\ln\frac{r}{R_0}\right), \tag{4.4.2}$$

where $p_0 = \rho u_0^2/2$. We compute the density on the inner boundary nodes locally by equation (3.3.8) and the nodes on the outer boundary by the interpolation scheme used

to compute $\vec{u}$ at the boundary nodes.

Whereas u_0 on the inner cylinder in polar coordinates is constant, its x-component and y-component vary with x and y. By decomposing the normal vector $\vec{n}$ from figure 3.4.3 into its x and y-component, we obtain the resulting x and y-components of $u0$: $\langle u_{0x}, u_{0y} \rangle^T = \langle -n_y u_0, n_x u_0 \rangle^T$.

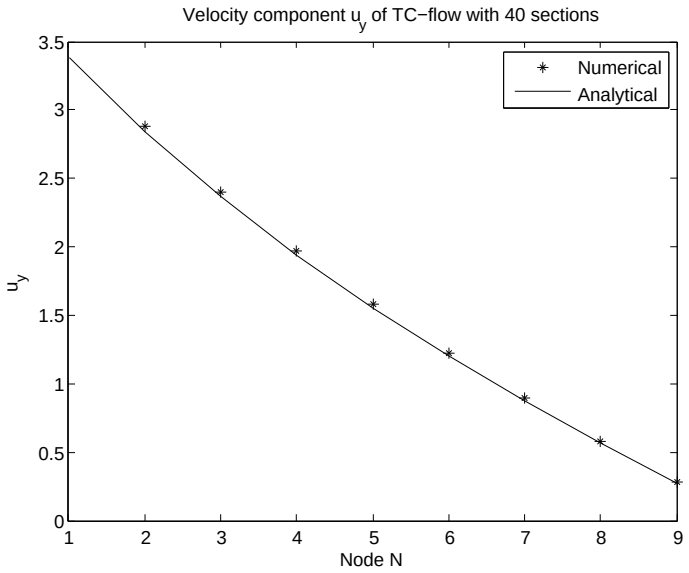


Figure 4.4.1: Profile of the u_y-component of $\vec{u}$ for the Taylor-Couette flow with N=40 sections to resolve the length. The analytical solution is indicated by the solid line, and the numerical solution by $*$.

Opposed to the benchmarks used to verify the straight boundary conditions, generally all components of $\mathbf{S}$ are non-zero on curved boundary nodes. Thus, the error stemming from the approximation of the diagonal components of $\mathbf{S}$ contributes to the total error for curved boundary conditions. As we saw in section 3.2 and in figure 3.4.4, the velocity gradients on boundary nodes close to the wall are approximated without taking into account the distance from P, where we compute u_P, to its neighbors, where we get the velocity components we need to compute ∂_x and ∂_y. We expect this approximation to increase the error introduced by $\mathbf{S}$, but not to detoriorate the order of the method.

Our results show that the error decreases with δx^2 for this stationary flow. The accuracy of the method is dictated by how we approximate ρ, $\vec{u}$ and $\mathbf{S}$. The reason why these approximations dictate the order in which the error is decreased is that we know that the other steps in the algorithm, like reconstruction, streaming and collision on the boundary nodes, do not detoriorate the second order accuracy of the lattice Boltzmann method. Any error introduced by these approximation schemes propagates through the entire domain and thus dictates the overall error. We therefore conclude that the approximations of ρ, $\vec{u}$ and $\mathbf{S}$ for curved walls presented in chapter 3 conserve the second order accuracy in space of the lattice Boltzmann method for the current benchmark test.

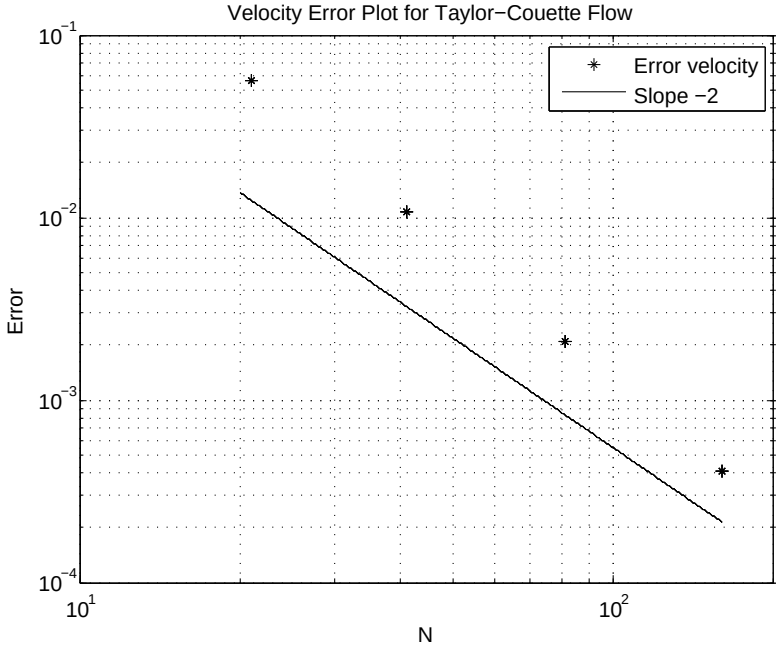


Figure 4.4.2: Velocity error of Taylor-Couette flow. The solid line has slope -2.

4.5 Taylor-Vortex Flow

We simulated steady and unsteady Taylor-vortex flow with and without curved boundaries in an infinite physical domain. This benchmark deviates from the other benchmarks in several ways. First, we do not assume the velocity components normal to the wall to be zero as is the case of the no-slip condition. Instead, we impose a more general velocity boundary condition at the wall. Second, both the pressure gradient and the external force are non-zero for this benchmark test. Third, the current flow is time-dependent.

We represent the unbounded physical domain numerically by solving this benchmark on a unity square with periodic boundaries. On this domain, we construct a solution by the method of manufactured solutions[19]. The velocity components in the x and y-direction, respectively, are given by:

$$u_x(x,y,t) = -\frac{1}{2\pi}\cos(\omega t)\cos(2\pi x)\sin(2\pi y) \tag{4.5.1}$$

$$u_y(x,y,t) = \frac{1}{2\pi}\cos(\omega t)\sin(2\pi x)\cos(2\pi y). \tag{4.5.2}$$

The pressure is given by:

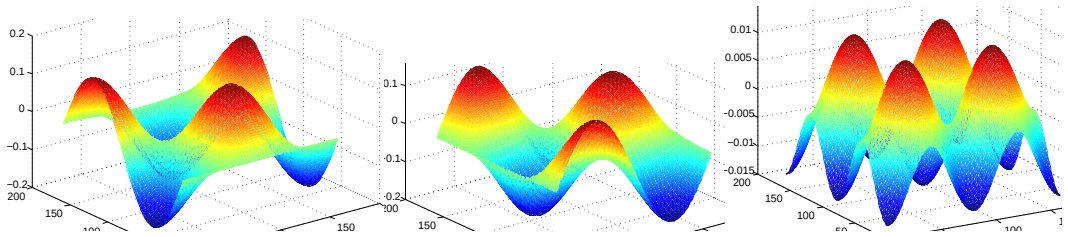
$$p(x,y,t) = -\frac{1}{16\pi^2}(\cos(4\pi x) + \cos(4\pi y))\cos(\omega t)^2, \tag{4.5.3}$$

where ω is the frequency determining the period of the flow. The velocity and pressure are given in non-dimensional units. The time-dependent forcing term is obtained by injecting

$\vec{u} = \langle u_x, u_y \rangle^T$ into the momentum equation (eqn.(1.2.9)). We chose $Re = 1$ such that the different terms in the momentum equation get the same influence. With $Re = 1$, the value of the relaxation frequency becomes 5/40 for this benchmark.

First, we solved this problem without curved boundaries. No reconstruction formalism is performed in this case, and thus the populations prior to collision are identical to the post-streamed populations. We used eqn. (1.2.5) and (1.2.7) from section 1.2 to introduce the force into the lattice Boltzmann method. We performed a series of simulations with different grid resolution between 11 and 81 nodes. We simulated the steady flow ($\omega = 0$) in addition flow the time-dependent flow. With 81 nodes to resolve the length, the frequency had to be really low ($\omega \approx 5\pi/2 \cdot 10^4$) to capture the time-evolution of the flow. As soon as the flow had come to a stable oscillation, we computed the error on one period.

subfig



(a) X-Velocity (b) Y-Velocity (c) Pressure

Figure 4.5.1: Velocity and Pressure for Taylor-Vortex Flow

We then placed a cylinder @ (x,y) = (1/2, 1/2), i.e. in the middle of the domain, with radius r_0=1/4. On the cylinder walls, we imposed the velocity field given by eqn.(4.5.1)-(4.5.2). We computed the density locally at the boundary nodes by eqn. (3.3.8) and (3.3.6). We simulated the steady case ($\omega = 0$) first. Thereafter, we simulated the unsteady case ($\omega \neq 0$). The error contribution from the approximation of ρ, $\vec{u}$ and $\mathbf{S}$ is found by comparing the error from the simulation *without* a curved boundary with the error from the simulation *with* a curved boundary. This is because the approximations of these quantities is the only additional source of error introduced by inserting the curved boundaries.

For the case without curved boundaries, we got the expected second order error decrease for the velocity. Figure 4.5.2 shows the error plot for steady flow, and figure 4.5.4 shows the time-dependent error reduction. As explained in section 4.3, we expect the pressure error to decrease with second order, since we do not impose a velocity boundary condition. The error plots in figure 4.5.3 and 4.5.5 show that the pressure was recovered with the expected second order accuracy.

The results for the pressure recovery show that by inserting the cylinder, and thereby introducing velocity boundary conditions, the order in which the pressure error was reduced went from two to one. The accuracy in velocity, on the other hand, actually

increased when we inserted the cylinder into the domain. This is somewhat counter intuitive, since imposing the wall boundary condition introduces error from the approximation of ρ, $\vec{u}$ and $\mathbf{S}$. However, at the wall, we impose the analytical solution for $\vec{u}$ for every time step, which is used in the approximation of the velocity at the boundary nodes. Therefore, the velocity $\vec{u}$ is accurately represented on these nodes. This effect propagates to the bulk fluid nodes, and the result is that the velocity field with a cylinder inside the domain is more accurately represented than the velocity field without the cylinder inside. We computed the error based on the value of ρ and $\vec{u}$ on the wet nodes (r>1/4) for the simulations with *and* for the simulations without the cylinder inside.

The method described in chapter 3 proved to resolve the velocity and pressure field with second and first order accuracy, respectively, for the time-dependent Taylor-Vortex flow with arbitrary curved velocity boundary conditions.

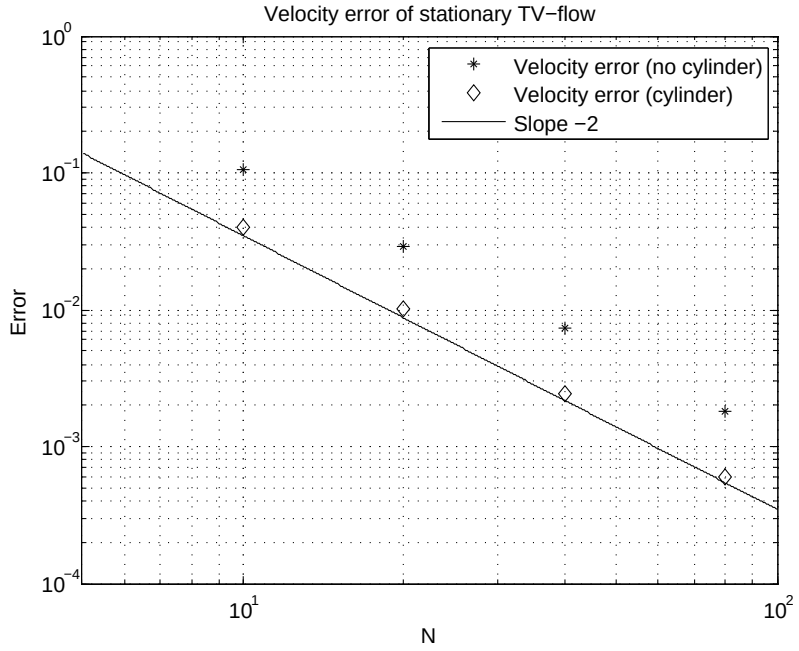


Figure 4.5.2: Velocity error of steady Taylor-Vortex flow. Points $*$ indicate 'without cylinder', and points $\diamond$ indicate 'with cylinder'. Solid line has slope -2.

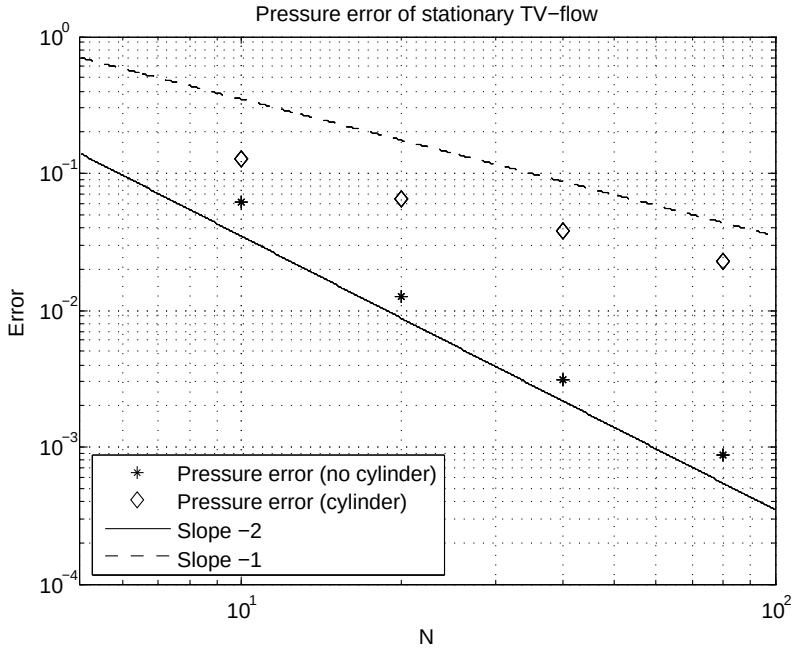


Figure 4.5.3: Pressure error of steady Taylor-Vortex flow. Points $*$ indicate 'without cylinder', and points $\diamond$ indicate 'with cylinder'. Solid line has slope -2, and dashed line has slope -1.

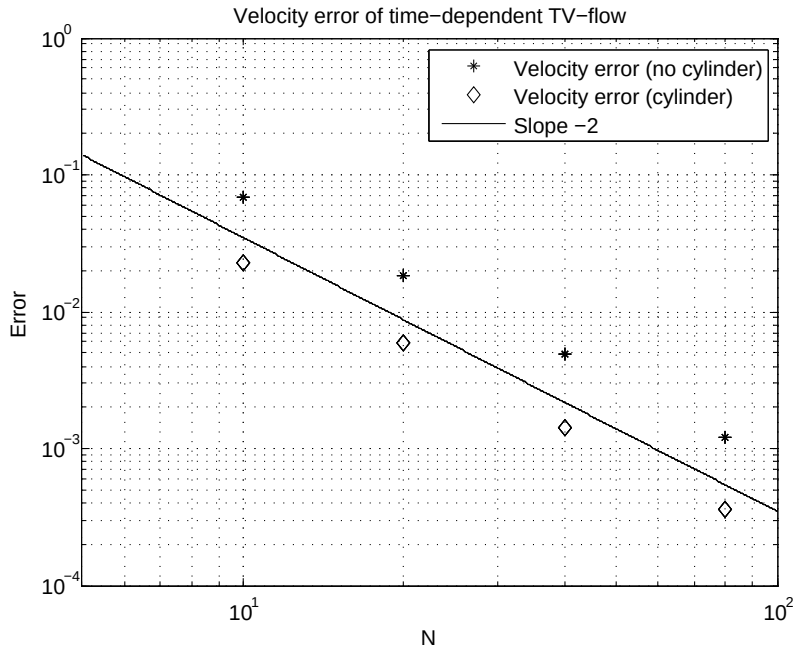


Figure 4.5.4: Velocity error of unsteady Taylor-Vortex flow. Points $*$ indicate 'without cylinder', and points $\diamond$ indicate 'with cylinder'. Solid line has slope -2.

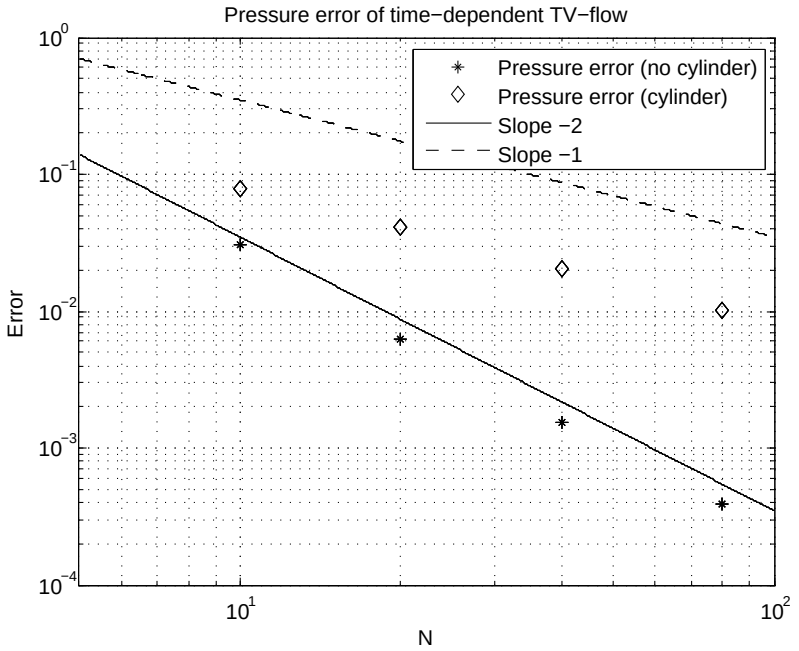


Figure 4.5.5: Pressure error of unsteady Taylor-Vortex flow. Points $*$ indicate 'without cylinder', and points $\diamond$ indicate 'with cylinder'. Solid line has slope -2, and dashed line has slope -1.

Chapter 5

Conclusion

We treated a number of local and non-local boundary conditions for straight and curved walls for the LBGK version of LBM in this thesis. The straight boundary conditions were discussed in chapter 2. In chapter 3, we investigated a no-slip boundary condition for curved walls with non-zero velocity aimed at solving two dimensional fluid flow problems. This curved boundary condition was derived by Verschaeve and Müller[9] and is based on the regularization procedure developed by Lätt[11] for straight walls.

We performed four different benchmark tests in chapter 4 to verify the straight and curved boundary conditions. The gravity driven flow in a slit and the same flow driven by a pressure gradient were the benchmark tests performed to verify the straight boundary conditions. We verified the curved no-slip boundary condition from chapter 3 by simulating steady Taylor-Couette flow in-between two rotating cylinders. The last benchmark test was the time-periodic Taylor-Vortex flow. With this benchmark, we verified a more general case of the curved boundary conditions than the no-slip condition, where the velocity components normal to the wall are zero. Instead, we imposed an arbitrary velocity field at the wall with generally no non-zero velocity components. The results from this benchmark showed that our scheme conserved the second order accuracy of the lattice Boltzmann method in time and space.

Further work should focus on a method to move curved walls between the nodes of the computational domain.

Acknowledgments

First, I would like to thank my research advisor Joris Verschaeve for follow-up through every step of the process of writing this thesis. I would also like to thank my adademic supervisor Berhard Müller for good advise and interest in the progress.

Bibliography

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

[2] W Malalasekera H K Versteeg. *An Introduction to Computational Fluid Dynamics. The Finite Volume Method*. Pearson, 2007.

[3] Dieter A. Wolf-Gladrow. *Lattice-Gas Cellular Automata and Lattice Boltzmann Models. An Introduction*. Springer, 2000.

[4] Ronald L. Panton. *Incompressible Flow, Third Edition*. Wiley, 2005.

[5] R. R. Nourgaliev, T. N. Dinh, T. G. Theofanous, and D. Joseph. The lattice boltzmann equation method: theoretical interpretation, numerics and implications. *International Journal of Multiphase Flow*, 29(1):117 – 169, 2003.

[6] S. J. Humby, M. J. Biggs, and U. Tüzün. Explicit numerical simulation of fluids in reconstructed porous media. *Chemical Engineering Science*, 57(11):1955 – 1968, 2002.

[7] Michael C. Sukop and Daniel T. Thorne. *Lattice Boltzmann Modeling. An Introduction for Geoscientists and Engineers*. Springer, 2005.

[8] Sauro Succi. *The Lattice Boltzmann Equation for Fluid Dynamics and Beyond*. Oxford Science Publications, 2001.

[9] Joris C. G. Verschaeve and Bernhard Müller. A curved no-slip boundary condition for the lattice boltzmann method. *J. Comput. Phys.*, 229:6781–6803, September 2010.

[10] J. Lätt. *Hydrodynamic Limit of Lattice Boltzmann Equations*. PhD thesis, Université de Genève, 2007.

[11] Jonas Latt, Bastien Chopard, Orestis Malaspinas, Michel Deville, and Andreas Michler. Straight velocity boundaries in the lattice boltzmann method. *Phys. Rev. E*, 77(5), 2008.

[12] Michael Junk, Axel Klar, and Li-Shi Luo. Asymptotic analysis of the lattice boltzmann equation. *Journal of Computational Physics*, 210(2):676 – 704, 2005.

[13] Junfeng Zhang. Lattice boltzmann method for microfluidics: models and applications. *Microfluidics and Nanofluidics*, 10:1–28, 2011. 10.1007/s10404-010-0624-1.

[14] 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.

[15] Q. Zou and X. He. On pressure and velocity boundary conditions for the lattice boltzmann bgk model. *Phys. Fluids*, 9:1591, 1997.

[16] Zhaoli Guo, Chuguang Zheng, and Baochang Shi. Discrete lattice effects on the forcing term in the lattice boltzmann method. *Phys. Rev. E*, 65(4):046308, Apr 2002.

[17] Joris C. G. Verschaeve. Analysis of the lattice boltzmann bhatnagar-gross-krook no-slip boundary condition: Ways to improve accuracy and stability. *Phys. Rev. E*, 80(3):036703, Sep 2009.

[18] M. Junk and Z. Yang. Analysis of lattice boltzmann boundary conditions. *PAMM*, 3(1):76–79, 2003.

[19] Patrick J. Roache. Code verification by the method of manufactured solutions. *Journal of Fluids Engineering*, 124(1):4–10, 2002.