UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Química

Lucas Giuliano Murdiga de Moraes

Flow in Porous Media and Adsorption of Binary Fluids via Lattice Boltzmann Method

Escoamento em Meios Porosos e Adsorção de Misturas Binárias pelo Método Lattice Boltzmann

Campinas

2020

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Química

Lucas Giuliano Murdiga de Moraes

Flow in Porous Media and Adsorption of Binary Fluids via Lattice Boltzmann Method

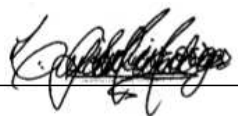
Escoamento em Meios Porosos e Adsorção de Misturas Binárias pelo Método Lattice Boltzmann

Dissertation presented to the School of Chemical Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Chemical Engineering.

Dissertação apresentada à Faculade de Egenharia Química da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Química.

Orientador: Prof. Dr. Luís Fernando Mercier Franco

Este exemplar corresponde à versão final da tese defendida pelo aluno Lucas Giuliano Murdiga de Moraes, e orientada pelo Prof. Dr. Luís Fernando Mercier Franco



Campinas

2020

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Luciana Pietrosanto Milla - CRB 8/8129

M791f

Moraes, Lucas Giuliano Murdiga de, 1995-
Flow in porous media and adsorption of binary fluids via Lattice Boltzmann Method / Lucas Giuliano Murdiga de Moraes. – Campinas, SP : [s.n.], 2020.

Orientador: Luís Fernando Mercier Franco.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Química.

1. Lattice Boltzamann, Metodo de. 2. Fluidodinâmica computacional (CFD). 3. Escoamento. 4. Meios porosos. 5. Adsorção. I. Franco, Luís Fernando Mercier, 1988-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Química. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Escoamento em meios porosos e adsorção de misturas binárias pelo Método Lattice Boltzmann
Palavras-chave em inglês:
Lattice Boltzmann methods
Computational fluid dynamics (CFD)
Flow
Porous media
Adsorption
Área de concentração: Engenharia Química
Titulação: Mestre em Engenharia Química
Banca examinadora:
Luís Fernando Mercier Franco [Orientador]
Caetano Rodrigues Miranda
Dirceu Noriler
Data de defesa: 19-02-2020
Programa de Pós-Graduação: Engenharia Química

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: 0000-0001-8878-8194
- Currículo Lattes do autor: http://lattes.cnpq.br/1054810539383714

Folha de Aprovação da Dissertação de Mestrado defendida por Lucas Giuliano Murdiga de Moraes, em 19 de fevereiro de 2020 pela banca examinadora constituída pelos doutores.

Prof. Dr. Luis Fernando Mercier Franco - Presidente e Orientador
FEQ / UNICAMP

Prof. Dr. Dirceu Noriler
FEQ / UNICAMP

Dr. Caetano Rodrigues Miranda
Universidade de São Paulo

A Ata da defesa com as respectivas assinaturas dos membros encontra-se no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria do Programa da Unidade.

Acknowledgements

I would like to thank the person who initiated this journey with me, professor Luís Fernando Mercier Franco of the University of Campinas, for the advice and guidance during the elaboration of this dissertation.

I would like to thank all my fellow students of the Complex Systems Engineering Laboratory (LESC, from Portuguese *Laboratório de Engenharia de Sistemas Complexos*) that accompanied me during the elaboration of this work and advised me several times.

I thank the team of the School of Chemical Engineering of the University of Campinas, for providing the necessary structure and institutional support for the completion of this work.

Special thanks to the student Nadine Zandoná Rafagnim for providing the CFD simulations that are presented in this work, which enriched this dissertation, and to the undergraduate student Guilherme Milhoratti Lopes, who implemented the Monte Carlo simulations for the Ising model used to generate the porous meshes presented. Also, I thank my fellow student Rodrigo Presence Bagarolo who helped me several times when I needed guidance to continue my work.

I would like to thank my mother for the support during this time, and my family as a whole for always being by my side in the moments I needed the most. I am very grateful for having you by my side.

This study was financially supported in part by the *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil* (CAPES) - Finance Code 001.

To the people directly or indirectly mentioned here, thank you very much! This work could not be accomplished without your support.

Lucas Giuliano Murdiga de Moraes

Abstract

The Lattice Boltzmann Method (LBM) is an alternative to traditional Computational Fluid Dynamics (CFD). It has recently gained popularity due to its special capabilities, especially at scales where microscopic effects become important. The LBM originated in statistical mechanics, which implies the existence of particles contrasting with the continuum hypothesis assumed in CFD. Furthermore, LBM also contrasts with Molecular Dynamics (MD) and Monte Carlo (MC) simulations, since it does not focus on each individual particle but in their distribution, which allows for less computational demand in bigger domains. The meshes (or lattices) commonly used in LBM are simple, and the way the method is structured allows for code parallelization and versatility in simulating complex geometries, such as porous media. The present work focuses on the implementation and study of the LBM for simulating flows in porous media and under the effect of adsorption forces. These are areas of interest with great perspectives, especially for chemical engineering. A program was created and the code validated by performing simulations for cases that can be compared to analytical solutions (Poiseuille and Couette). More complex geometries (Poiseuille flow around an infinite cylinder, lid-driven cavity) were also used for evaluating the code, comparing the results to those available in the literature and CFD simulations. Flow in porous media was simulated in meshes generated using Monte Carlo simulations for the Ising model. The results were analyzed and compared to the Darcy's Law. Forces were also implemented and used to simulate body-forces and adsorption. Results were analyzed qualitatively and compared to other studies when possible. They proved to be very satisfactory for the simple geometries, and the flow behaved as expected in the complex ones. An alternative model for a distribution function was proposed to study binary flows of non-interacting particles, yielding good results when simulating adsorption. The results show the feasibility of using the LBM to simulate the studied systems.

Keywords: Lattice Boltzmann, CFD, porous media, multi-component flow, adsorption.

Resumo

O método da Rede Boltzmann (LBM, do inglês *Lattice Boltzmann Method*) é uma alternativa às metodologias tradicionais usadas na Fluidodinâmica Computacional (CFD, do inglês *Computational Fluid Dynamics*). Sua popularidade tem aumentado devido às suas capacidades únicas, especiamente na escala em que efeitos microscópicos se tornam importantes. O LBM tem origem Mecânica Estatística, fazendo uma descrição discreta da matéria em contraposição à hipótese do contínuo, na qual os métodos tradicionais de CFD se baseiam. O LBM também contrasta com a dinâmica molecular (MD) e simulações de Monte Carlo (MC) por não considerar as moléculas individualmente, permitindo uma demanda computacional menor para domínios muito grandes. As malhas computacionais (*lattices*) são simples, e o modo como o método é estruturado permite paralelização e versatilidade para simular geometrias complexas, como meios porosos. O presente trabalho foca na implementação e estudo do LBM para escoamentos em meios porosos e adsorção, áreas de grande interesse e com muitas perspectivas. Um código foi escrito para as simulações, o qual foi validado comparando os resultados com soluções analíticas conhecidas dos escoamentos de Poiseulle e Couette. Geometrias mais complexas (escomento de Poiseulle em torno de um cilindro infinito e *lid-driven cavity*) também foram avaliadas qualitativamente, comparando a resultados disponíveis na literatura e com simulações de CFD. O escoamento em meios porosos foi simulado em malhas geradas por simulações de Monte Carlo para o modelo Ising bidimensional. As simulações foram avaliadas qualitativamente e a lei de Darcy foi verificada. Forças de corpo (*body-forces*) e locais (para adsorção) também foram implementadas. Os resultados foram avaliados qualitativamente e comparados com outros estudos, quando possível. As simulações em geometrias simples foram bastante satisfatórias, apresentando erros pequenos. Para os casos mais complexos, o comportamento também foi como esperado. Uma distribuição de equilíbrio alternativa foi proposta para simular sistemas binários sem interação entre os componentes, apresentando bons resultados para adsorção. Assim, os resultados comprovam a aplicabilidade do método para os sistemas apresentados.

Palavras-chave:: Rede Boltzmann, CFD, escoamento multi-componente, meios porosos, adsorção.

List of Figures

Contents

1 Introduction

Fluid mechanics is the field of science and engineering associated with the study of fluids in motion. This branch of physics models the behavior of fluids associated with transport phenomena of heat, mass, and momentum in attempts to understand these phenomena. It usually consists in solving equations associated with the balance of such quantities in control volumes, which tends to result in a series of differential equations to be solved (KRÜGER *et al.*, 2017).

Those equations, however, cannot always be solved analytically, especially in the case of mixtures, complex geometries, and coupled transport phenomena. It is then necessary to resort to numerical approximation methods. At first, these methods had to be performed manually. Nonetheless, with the advent of computers in the late XX Century, computational methods and tools were developed and became increasingly popular for solving such problems, giving birth to the field of Computational Fluid Dynamics (CFD) used to this day to simulate flows.

As time passed, processes increased in complexity, and efficiency became an an important factor, whist the computational power also increased exponentially. Thus, CFD and other numerical tools became popular, and currently several commercial and free software are available to perform such calculations. Other methodologies were also developed focusing on molecular interactions in a microscopic level, as is the case for Molecular Dynamics (MD). Therefore, they have larger computational domains to simulate each molecule and are more computationally intensive.

Those are completely different scales in which problems can be analyzed. The continuum hypothesis is said to be a macroscopic approach, whereas the atomic and molecular scales can be interpreted as performed from microscopic approaches. In CFD, the term microscopic may be used when simulating the flow within pores, microchannels, and other microscopic systems. Nevertheless, for the purposes of this work those are still said to be in the macroscopic scale.

The Lattice Boltzmann Method (LBM) has gained popularity due to its unique characteristics. It presents itself as an intermediate method, physically associated with

statistical mechanics (SM) through the Boltzmann Transport Equation (BTE) instead of macroscopic balances or interactions between molecules. The LBM, instead, considers the distribution of molecules with respect to time, space, and velocities by using a probability distribution function. The space is discretized in a lattice structure and this distribution function of the molecules is operated in each point of the lattice to recover the characteristic macroscopic behavior.

This statistical approach allows for the implementation of microscopic effects in a more straightforward manner when compared to CFD methods. Computational cost can also be reduced when compared to MD, since several particles can be incorporated into a single distribution function. However, LBM simulations can also be very intensive, and for the same system can be more computationally costly than CFD. Yet, the calculations are performed mostly locally, allowing for easy parallelization, and the lattice nature of the method allow for implementation in complex structures (KRÜGER *et al.*, 2017).

Overall, the LBM presents a good compromise between the macroscopic and molecular scales and thus, can be said to operate in between. The prospects for multicomponent systems and porous media are promising, especially with the several extensions the method has gained within its lifetime until now, allowing for simulation of multiphase-multicomponent systems, colloidal solutions, diffusivity, turbulence, and coupled transport phenomena (CHEN; DOOLEN, 1998; SUCCI, 2001; NOURGALIEV *et al.*, 2003; PENG, 2011).

LBM has grown in popularity in recent years with novel applications, as the simulating blood flow, for example (ZHANG; JOHNSON; POPEL, 2008; PETERS *et al.*, 2010). The method is well-suited to simulating microscopic systems and complex geometries. Applied to multicomponent and reactive flow, LBM thus allows for simulation of systems of interest in science and engineering, as simulations of oil and gas reservoirs, chemical reactors, and biological systems. Overall, LBM is suitable to simulate systems where the confinement effect is important for describing the system. Examples are reactive flows, adsorption and catalysis, microfluidcs, microreactors, mass transfer, phase and component separation, bubble dynamics and droplets formation, fuel cells, batteries, nanoporous electrodes, and even biological systems (SHARMA; STRAKA; TAVARES, 2019).

LBM is also used in multi-scale problems in combination with other approaches. Recently, Wang *et al.* (2015) proposed a flux solver based in the LBM which applies concepts of the finite difference methods (FDM). Poonoosamy *et al.* (2019) used the LBM in microfluidic simulations for crystal nucleation in combination with FDM, comparing the different approaches. Başağaoğlu and Succi (2010) implemented various terms for modeling the internal and external forces, accounting for a series of possible interactions, including the two-pair Lennard-Jones potential.

Nonetheless, there are still aspects to improve in the LBM. Several manners of implementing boundary conditions and multicomponent-multiphase flow exist. Moreover, works simulating mass transfer and adsorption in multi-component systems are yet a minority of published works and, thus, various possibilities remain to be explored. Also, CFD simulations are still more popular than LBM, specially in the engineering community where CFD is consolidated and well established. It has matured more within its lifespan when compared to LBM, which creates certain resistance to LBM usage in engineering, despite its advantages in several aspects.

As such, the present work focuses on studying the Lattice Boltzmann Method for complex geometries and multicomponent flows. This work will present a general overview of fluid dynamic simulations and introduce the LBM. Various methodologies associated to the implementation of LBM will be explored and explained. The simulation results are then presented and used to validating the code and simulating complex geometries.

1.1 Objectives

The main objective of this work in to study and implement the Lattice Boltzmann Method (LBM) to simulate flow in porous media, adsorption and non-interacting binary flows.

The first goal is writing a code to implement the LBM, and validating it using flows of known well-established behavior. Poiseuille, Couette, and combined flows are compared to the analytical solutions. Poiseulle flow around an infinite cylinder and Lid-Driven Cavity geometries are also simulated.

Next, body-forces are implemented and tested in the direction (Poiseulle geometry) and perpendicular (external field) to the flow.

The code is then used to simulate flows in computationally generated porous media, comparing the results to Darcy's Law.

Then, adsorption forces are implemented to simulate the phenomenon. Finally, multicomponent model for a binary ideal fluid with non-interacting components is introduced and used to simulate adsorption of the individual components.

2 Fluid Dynamics Overview

The mathematical description of fluids is usually treated in terms of the macroscopic behavior, relying on the continuum hypothesis, which treats the fluids as if they were continuous instead of composed of particles. This is a reasonable approximation in most cases, since the scale of the problems are usually much greater than the molecular scale, making the errors negligible. Nonetheless, such a hypothesis is physically inaccurate since it neglects the existence of molecules, introducing errors when the effects associated to the molecular scale become significant.

The continuum macroscopic fluid is divided into fluid elements of volume V_0 with mass density ρ_m. The mass of such volume element is thus given by Equation 2.1.

$$m = \int_{V_0} \rho \, \mathrm{d}V \tag{2.1}$$

The variation of any specific variable ϕ (that corresponds to extensive variable Φ) with time t, within a volume element $\mathrm{d}V$, is due to flow of such variable into and out of the boundaries of the volume element contained within the enclosed surface A_0 with velocity $\vec{v}$, the flux due to an external field represented as $\vec{j_\phi}$, and a source term associated with ϕ given by $\dot{S}_\phi$, as represented in Equation 2.2.

$$\int_{V_0} \frac{\partial}{\partial t}(\rho\phi)\,\mathrm{d}V = -\oint_{A_0} (\rho\phi\vec{v}) \cdot \mathrm{d}\vec{A} - \oint_{A_0} \vec{j_\phi} \cdot \mathrm{d}\vec{A} + \int_{V_0} \dot{S}_\phi \, \mathrm{d}V \tag{2.2}$$

The surface integrals can be converted into volume integrals using the divergence theorem, resulting in Equation 2.3.

$$\int_{V_0} \frac{\partial}{\partial t}(\rho\phi)\,\mathrm{d}V = -\int_{V_0} \vec{\nabla} \cdot (\rho\phi\vec{v})\,\mathrm{d}V - \int_{V_0} \vec{\nabla} \cdot \vec{j_\phi}\,\mathrm{d}V + \int_{V_0} \dot{S}_\phi \, \mathrm{d}V \tag{2.3}$$

Equation 2.3 can then be simplified to a general transport equation for ϕ in the form of Equation 2.4.

$$\frac{\partial}{\partial t}(\rho\phi) = -\vec{\nabla} \cdot (\rho\phi\vec{v}) - \vec{\nabla} \cdot \vec{j_\phi} + \dot{S}_\phi \tag{2.4}$$

When $\Phi = m$ represents the total mass of the system, $\phi = 1$ implied. Because of mass conservation, the source and flux terms are nonexistent and, thus, Equation 2.4 reduces to the Continuity Equation (2.5).

$$\frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot (\rho \vec{v}) = 0 \tag{2.5}$$

To recover the momentum transport equations, the variable used is the velocity of the flow: $\phi = \vec{v}$. The flux is given by differences in pressure p, and the source term is due to external forces $\vec{F}$, reducing Equation 2.4 to the Euler Equation (2.6).

$$\frac{\partial}{\partial t}(\rho \vec{v}) = -\vec{\nabla} \cdot (\rho \vec{v}\,\vec{v}) - \vec{\nabla} p + \vec{F} \tag{2.6}$$

The term $\vec{v}\,\vec{v}$ is a second-order tensor formed by the multiplication of the velocity components. This equation can also be written in a more general form as the Cauchy Momentum Equation (2.7).

$$\frac{\partial}{\partial t}(\rho \vec{v}) = -\vec{\nabla} \cdot \vec{\vec{\Pi}} + \vec{F} \tag{2.7}$$

In this more general equation $\vec{\vec{\Pi}} = \rho \vec{v}\,\vec{v} - \vec{\vec{\sigma}}$ is a second-order tensor, with $\vec{\vec{\sigma}}$ as the stress tensor. It can represent more complex fluids by separating the pressure gradient that appears in the Euler Equation (2.6) and a viscous stress tensor included to account for dissipative transfer of moment due to internal friction and viscosity.

$$\vec{\vec{\sigma}} = \vec{\vec{\sigma}}_\mu - p\vec{\vec{I}} \tag{2.8}$$

Here, $\vec{\vec{I}}$ is defined as the identity tensor such that $p\vec{\vec{I}}$ is diagonal and $\vec{\nabla} \cdot p\vec{\vec{I}} = \vec{\nabla} p$. $\vec{\vec{\sigma}}_\mu$ is a viscous stress tensor which is related to the velocity gradient of the fluid by the viscosity μ. By considering the general form of the stress tensor $\vec{\nabla} \cdot \vec{\vec{\sigma}}_\mu = \mu \nabla^2 \vec{v} = \nu \nabla^2 (\rho \vec{v})$, the Navier Stokes Equation (NSE) is recovered:

$$\frac{\partial}{\partial t}(\rho \vec{v}) + \vec{\nabla} \cdot (\rho \vec{v}\,\vec{v}) = -\vec{\nabla} p + \nu \nabla^2 (\rho \vec{v}) + \vec{F} \tag{2.9}$$

By choosing other forms for the viscous stress tensor, and adding forces, a series of modified NSEs can be recovered to account for more complex fluids. The viscous

stress tensor simplified in the NSE (2.9) by the kinematic viscosity ν can also be treated as a higher order tensor to account for anisotropic fluids when necessary.

There are other important transport equations used in specific cases. Energy and mass for different components of a mixture can also be implemented using the general transport equation (2.4). To account for heat transfer, it is possible to use the Navier-Stokes-Fourier Equations (KRÜGER *et al.*, 2017) as an example, but a series of methods exist.

Another important set of equations are Advection-Diffusion Equations (ADEs). These can be used to model problems consisting of mixing in multi-component fluids and heat diffusion. The general form of such equations for a scalar field Ψ is:

$$\frac{\partial}{\partial t}\Psi + \vec{\nabla}\cdot(\Psi\vec{v}) = \vec{\nabla}\cdot(D\vec{\nabla}\Psi) + \dot{S} \tag{2.10}$$

This approach shares several similarities with the NSE, and in fact, can be understood as an special case of those. The variable of interest, Ψ, is usually the temperature or concentration, which can be considered a mass or number density for a specific component in a mixture. The term D is a diffusivity coefficient, equivalent to the viscosity in the NSE, and can be simplified in an isotropic fluid. $\dot{S}$ is a source term for Ψ.

The ADEs and NSEs, despite being different in nature, share various similarities and are commonly used to solve a great range of real problems. These equations are the most used in the field known as Computational Fluid Dynamics (CFD), which focuses on discretization and solution of these equations computationally to obtain the macroscopic behavior of a fluid.

The finite elements method (FEM) first appeared in 1956, and was popularized in the following decade, along with the finite difference methods (FDM) (MOHAMAD, 2011). In the 1980's, the finite volumes method (FVM) was developed and gained popularity, being used to this date in CFD calculations. The aforementioned methods consist in dividing a greater volume of fluid into small pieces and solving NSEs for each of the parts. The main difference is in the discretization method used to solve the NSE, which alters the calculations. Usually, the values also depend on the neighboring regions.

Nonetheless, as robust as the continuum hypothesis may be for a series of cases, they do not consider the microscopic effects directly (KRÜGER *et al.*, 2017). Matter is

actually made of molecules, atoms, and particles. The methods presented until now are unable of representing such structures, since they assume a continuous matter and, thus, require compatible scales of length and time. Therefore, the addition of models become necessary, which tend to average the microscopic behavior and are hard to achieve. They also tend to be less predictive, since they tend to depend on fittings of experimental data.

To better understand the limitations imposed by the traditional CFD methods, it is necessary to consider the typical scales in which different problems can be translated. Traditional CFD methods are bound to classical mechanics (CM) and, as such, are adequate to represent macroscopic problems. High energies, velocities, and distances may require the consideration of relativistic effects. New methodologies have to be introduced in this case, but in fluid dynamics studies for engineering these cases are rare and very specific. The molecular and atomic scales, on the other hand, present very visible effects in a series of common problems and are of great interest in science and engineering.

Atomic and subatomic scales are described using quantum mechanics (QM), which focuses on the study of particles, atoms, and molecules. Time scales are also very small, and other physical models and mathematical tools were developed to solve quantum systems in these scales (GEORGESCU; ASHHAB; NORI, 2014). The problems are usually focused on the simulation of specific molecules and small systems to obtain macroscopic properties, but are very hard to simulate. They are very computationally intensive, becoming impractical for large systems. To simulate 1 mole of any specific substance, a number in the order of 10^{23} molecules is necessary, making the simulations prohibitive for large systems. These techniques will be promising when quantum computers become a reality, but are not yet adequate to simulate the dynamic behavior of macroscopic fluids. The Lattice Boltzmann Method (LBM) was also modified to simulate quantum systems by solving the Schrödinger and Dirac equations (PALPACELLI; SUCCI, 2008), but not many works are found in this area yet.

Still at the atomic and molecular scale, other methodologies were developed with focus on the interactions between molecules, but without relying on quantum mechanics. These are the origins of molecular dynamics (MD), which consists in integrating Newton equations of motion for individual molecules in a system. Also, there is the Monte Carlo (MC) method consisting of stochastic assessments of the position of the

molecules within a system. Those approaches, despite requiring less computational effort than QM problems, are still much more intensive than the macroscopic treatments, and can only be performed in small system (thousands of particles) with a defined quantity of molecules.

The limitations imposed by the continuum hypothesis in the macroscopic treatment of fluids led to the development of alternative approaches in the form of statistical mechanics (SM), which considers the existence of atoms and molecules but recognized the difficulties in analyzing the interactions between each one of them. As such, statistical approaches are based on the average properties of the whole system. Those are intermediate approaches resulting in very powerful albeit more complicated tools. SM theories and models were developed specifically for this intermediate scale, notably, the Kinetic Theory (KT) and the field of Statistical Thermodynamics (STD), which caused a revolution in the way thermodynamics was understood.

As those methods are intermediate between the macroscopic and microscopic (molecular) scale, they can be said to be mesoscopic approaches. The predecessor of the LBM is also found in this category as the Lattice Gas Automata (LGA). Heavily based on the KT, the LGA was introduced by Frisch, Hasslacher and Pomeau (1986). It is centered in the idea of representing the macroscopic behavior of a gaseous fluid using representative molecules contained within the nodes of a lattice structure. However, the method presents instability and numerical noise, and is limited to simulations of gaseous fluids (MOHAMAD, 2011).

The Lattice Boltzmann Method (LBM) itself is an evolution of the LGA based on the Boltzmann Transport Equation (BTE), which grants it a strong physical basis. The BTE is centered in a distribution function of the particles in respect to space, time, and velocities. LBM preserves the lattice structure and operates these distribution functions instead of the representative molecules used in the original LGA, improving the stability and reducing noise (MOHAMAD, 2011; PENG, 2011).

LBM has gained popularity in recent years for its applicability. As stated, a series of extensions were proposed and promising applications do exist. The method is a good alternative to simulate microscopic effects, being less intensive than MD and MC simulations, but better representing the corresponding effects than the classical methods.

3 The Boltzmann Transport Equation

Ludwig Boltzmann (1844-1906) was an Austrian physicist well known for his contributions in the field of statistical mechanics. The Boltzmann Transport Equation (BTE) was developed around 1872 and constitutes one of his most important contributions. The BTE (3.1) statistically describes the motion of particles within a fluid and is also in the centerpiece of the Lattice Boltzmann Method (LBM).

$$\frac{\partial f}{\partial t} + \vec{\xi} \cdot \vec{\nabla}_{\vec{x}} f + \frac{\vec{F}}{\rho} \cdot \vec{\nabla}_{\vec{\xi}} f = \Omega(f) \tag{3.1}$$

The BTE follows the evolution of a distribution function f that represents the probability density of particles at a determined time t, in a certain position $\vec{x}$, and with a velocity $\vec{\xi}$. The microscopical units of the fluids are from this point called generically as particles. This term may refer to individual molecules, but also can represent generic fluid particles that might not correspond to the real molecules.

The space comprised of the spatial coordinates x plus the velocity coordinates $\vec{\xi}$ denotes all possible configurations for each particle and is called phase space. Therefore:

$$f = f(\vec{x}, \vec{\xi}, t) \tag{3.2}$$

The function f represents a probability density for finding particles within points of this phase space, and is defined such that its integration in the velocity space results in the local number density of particles ρ_n in the position $\vec{x}$ at time t:

$$\rho_n(\vec{x}, t) = \int f(\vec{x}, \vec{\xi}, t) \mathrm{d}\vec{\xi} \tag{3.3}$$

This is the zeroth moment of f in relation to the velocity of the particles. The higher moments recover other macroscopic properties, such as the macroscopic velocity

($\rho\vec{u}$), and total (ρE) and internal energy (ρe):

$$\rho_n(\vec{x},t)\vec{u}(\vec{x},t) = \int \vec{\xi} f(\vec{x},\vec{\xi},t)\mathrm{d}\vec{\xi} \tag{3.4}$$

$$\rho_n(\vec{x},t)e(\vec{x},t) = \frac{1}{2}\int |\vec{\xi}|^2 f(\vec{x},\vec{\xi},t)\mathrm{d}\vec{\xi} \tag{3.5}$$

$$\rho_n(\vec{x},t)u(\vec{x},t) = \frac{1}{2}\int |\vec{\xi}-\vec{u}|^2 f(\vec{x},\vec{\xi},t)\mathrm{d}\vec{\xi} \tag{3.6}$$

These relations show mass, momentum, and energy transport as the momenta of a unique variable linked to the BTE. The term $\vec{\xi}-\vec{u}$ in Equation 3.4 is the relative velocity between the average fluid velocity $\vec{u}$ and the velocity of the individual particles $\vec{\xi}$, and can be denoted as a relative velocity $\vec{v} = \vec{\xi}-\vec{u}$ for simplicity.

The notation $a = |\vec{a}|$ is used to denote the magnitude a of a vector $\vec{a}$, meaning $a = |\vec{a}| = \sqrt{\sum \vec{a}_i}$, where $\vec{a}_i$ denotes the component of the vector in the direction i, which may be represented as the unitary vector $\hat{i}$. Therefore, the squared norm can also be represented as $a^2 = |a|^2 = \vec{a}\cdot\vec{a}$.

To derive the BTE, the changes of f with time has to be evaluated, which can be accomplished by analyzing its total differential. As f is a function of $\vec{x}$, $\vec{\xi}$, and t, the total derivative of f can be calculated as:

$$\mathrm{d}f(\vec{x},\vec{\xi},t) = \left(\frac{\partial f}{\partial t}\right)_{\vec{x},\vec{\xi}}\mathrm{d}t + \left(\frac{\partial f}{\partial \vec{x}}\right)_{\vec{\xi},t}\cdot\mathrm{d}\vec{x} + \left(\frac{\partial f}{\partial \vec{\xi}}\right)_{\vec{x},t}\cdot\mathrm{d}\vec{\xi} \tag{3.7}$$

Expliciting the differential for of Equation 3.7 in terms of time, and recognizing the terms $\frac{\partial f}{\partial \vec{x}}$ and $\frac{\partial f}{\partial \vec{\xi}}$ as $\vec{\nabla}_{\vec{x}} f$ and $\vec{\nabla}_{\vec{\xi}} f$ respectively, when the indices are omitted, the equation reduces to:

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial t} + \frac{\mathrm{d}\vec{x}}{\mathrm{d}t}\vec{\nabla}_{\vec{x}} f + \frac{\mathrm{d}\vec{\xi}}{\mathrm{d}t}\vec{\nabla}_{\vec{\xi}} f \tag{3.8}$$

It is possible to further simplify the equation. The derivative of the position in time corresponds to the velocity of the particles $\frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = \vec{\xi}$. As for the derivative of the velocity in time, it corresponds to an acceleration $\frac{\mathrm{d}\vec{\xi}}{\mathrm{d}t} = \vec{a}$, which can be correlated to a force density $\vec{F}$ by Newton law:

$$\vec{F} = \rho\vec{a} \tag{3.9}$$

With such substitutions, the final total derivative of f in time results in:

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial t} + \vec{\xi} \cdot \vec{\nabla}_{\vec{x}} f + \frac{\vec{F}}{\rho} \cdot \vec{\nabla}_{\vec{\xi}} f \tag{3.10}$$

This total derivative is called the collision operator, and is usually represented as $\Omega(f) = \frac{\mathrm{d}f}{\mathrm{d}t}$, recovering Equation 3.1. As for the physical meaning of the collision operator, it indicates the way the distribution function varies with time. To better understand the meaning of this term, it is possible to derive the BTE using physical arguments, as shown by Hirschfelder *et al.* (1954), and Mohamad (2011).

To grasp the meaning of the collision operator, the effects of an external force acting on the particles contained within an infinitesimal region of the phase space given by $\mathrm{d}\vec{x}\mathrm{d}\vec{\xi}$ can be considered. The distribution function f was shown to represent the distribution of particles in the position $\vec{x}$ with velocity $\vec{\xi}$, at a specific time t. If an external force density $\vec{F}$ acts on these particles during an infinitesimal time interval $\mathrm{d}t$, then the velocity of these particles will consequently change to $\vec{\xi} + \vec{F}\mathrm{d}t$ and the position to $\vec{x} + \vec{\xi}\mathrm{d}t$. The new distribution can, therefore, be written as $f(\vec{x} + \vec{\xi}\mathrm{d}t, \vec{\xi} + \vec{F}\mathrm{d}t, t + \mathrm{d}t)$. The difference in the value of f, given by $\mathrm{d}f$ can be calculated as the difference between those two expressions:

$$\mathrm{d}f = f(\vec{x} + \vec{\xi}\mathrm{d}t, \vec{\xi} + \vec{F}\mathrm{d}t, t + \mathrm{d}t) - f(\vec{x}, \vec{\xi}, t) \tag{3.11}$$

Moreover, in the absence of collision, the density within the volume $\mathrm{d}\vec{x}\mathrm{d}\vec{\xi}$ remains unchanged, implying $\mathrm{d}f\mathrm{d}\vec{x}\mathrm{d}\vec{\xi} = 0$. Substituting the definition of $\mathrm{d}f$ obtained, it is possible to write:

$$\mathrm{d}f\mathrm{d}\vec{x}\mathrm{d}\vec{\xi} = f(\vec{x} + \vec{\xi}\mathrm{d}t, \vec{\xi} + \vec{F}\mathrm{d}t, t + \mathrm{d}t)\mathrm{d}\vec{x}\mathrm{d}\vec{\xi} - f(\vec{x}, \vec{\xi}, t)\mathrm{d}\vec{x}\mathrm{d}\vec{\xi} = 0 \tag{3.12}$$

If the resulting equation is then divided by $\mathrm{d}\vec{x}\mathrm{d}\vec{\xi}\mathrm{d}t$ and evaluated at the limit $\mathrm{d}t \to 0$, the mathematical definition of the total differential of f in relation to time is obtained. Therefore, $\Omega(f) = 0$ in the absence of collisions. However, when collisions occur the equality in Equation 3.12 does not hold true and, thus, $\Omega(f)$ is not null anymore. This is the reason why $\Omega(f)$ is known as the collision operator, as it represents the results of the collisions of the particles within the system.

Real fluids are composed of particles that interact with each other. The collision operator can describe any interaction between the particles and, as such, can be very challenging to calculate. In its true form, $\Omega(f)$ can be represented as an integration of the potential that represents the interactions between each of the particles of the fluid with all other particles, resulting in an integro-differential equation for f.

Boltzmann applied the molecular chaos hypothesis to simplify the form of the collision operator. It assumes the moments of colliding particles as uncorrelated and independent of the position, which allows for the use of a generic distribution function for any representative particle. Therefore, the Boltzmann collision operator (3.13) consider binary interactions between particles and uses only two distribution functions f_1 and f_2 of two representative particles.

$$\Omega(f) = \iint (f_1' f_2' - f_1 f_2) |\vec{\xi}_1 - \vec{\xi}_2| \sigma(\omega) \mathrm{d}\omega \mathrm{d}\vec{v}_2 \tag{3.13}$$

Boltzmann original collision operator represents two particles that have a distribution f before and f' after the collision. ω is the solid angle between the velocities before and after the collision, and $\sigma(\omega)$ is analog to a cross-section of the collision, which depends on the potential of interaction between the particles.

For some purposes, however, the simplifications made by Boltzmann do not yield satisfactory results, and its use still imposes a complex transport equation not easily solvable for presenting differential and integrals of the same quantities. Thereafter, other models for this collision operator have been proposed in the literature.

Since f represents a distribution function, assuming a general f that directly results in the total density of particles of a kind and not a distribution for each single particle, as proposed by Boltzmann, it is possible to make assumptions about f when the particles are in equilibrium. One possible assumption is that f follows a Maxwell-Boltzmann distribution so as to write an equilibrium function f_{eq} as:

$$f_{\mathrm{eq}} = \rho \left(\frac{1}{2\pi RT} \right)^{3/2} \exp \left\{ -\frac{|\vec{v}|^2}{2RT} \right\} \tag{3.14}$$

This distribution depends on the temperature T, on the density ρ, and on the magnitude of the relative velocity $\vec{v} = \vec{u} - \vec{\xi}$. R is the gas constant. Bhatnagar, Gross

and Krook (1954) proposed a collision operator that follows the Maxwell-Boltzmann distribution in the form of:

$$\Omega(f) = -\frac{1}{\tau}(f - f_{\text{eq}}) \tag{3.15}$$

Essentially, the BGK collision operator implies that populations of particles tend to a local equilibrium within a relaxation time τ. This approach does not take into consideration the individual collision of the particles but the average, following the principles of statistical mechanics. Within this approach, the individual results are unimportant, as one large set of results can predict correctly the tendency of a system in a predictive manner (MOHAMAD, 2011). The combination of the BTE with the BGK collision operator results in the BE-BGK, which is written as:

$$\frac{\partial f}{\partial t} + \vec{\xi} \cdot \vec{\nabla}_{\vec{x}} f + \frac{\vec{F}}{\rho} \cdot \vec{\nabla}_{\vec{\xi}} f = -\frac{1}{\tau}(f - f_{\text{eq}}) \tag{3.16}$$

The number density ρ_n is also related to the mass density by a constant: the particles mass. Therefore, it is also possible to write the distribution f for mass instead of particle number. A generic ρ will then be used to write the momenta of f, since those two densities are interchangeable.

Integrating the BE-BGK in the velocity space and comparing to the moments presented in Equations 3.3 and 3.4, the continuity equation (2.5) can be recovered.

$$\int \frac{\partial f}{\partial t} \mathrm{d}\vec{\xi} + \int \vec{\xi} \cdot \vec{\nabla}_{\vec{x}} f \mathrm{d}\vec{\xi} + \int \frac{\vec{F}}{\rho} \cdot \vec{\nabla}_{\vec{\xi}} f \mathrm{d}\vec{\xi} = \int \Omega(f) \mathrm{d}\vec{\xi} \tag{3.17}$$

To guarantee the conservation of mass, momentum, and energy during the collisions, it is necessary that the moments of the collision operator are also equal to zero, so that:

$$\int \Omega(f) \mathrm{d}\vec{\xi} = 0 \tag{3.18}$$

The term $\vec{\xi} \cdot \vec{\nabla}_{\vec{x}} f$ can also be rearranged considering:

$$\vec{\xi} \cdot \vec{\nabla}_{\vec{x}} = \frac{\partial \vec{x}}{\partial t} \cdot \frac{\partial}{\partial \vec{x}} = \sum \frac{\partial x_i}{\partial t} \frac{\partial}{\partial x_i} = \sum \frac{\partial}{\partial x_i} \frac{\partial x_i}{\partial t} = \frac{\partial \vec{x}}{\partial t} \cdot \frac{\partial}{\partial \vec{x}} = \vec{\nabla}_{\vec{x}} \cdot \vec{\xi} \tag{3.19}$$

Therefore, using the definitions for the momenta of f (Equations 3.3, 3.4), in the absence of external forces ($\vec{F} = 0$), and making the adequate substitutions, the Con-

tinuity Equation (2.5) can be recovered:

$$\begin{aligned} \frac{\partial}{\partial t}\int f \mathrm{d}\vec{\xi} + \vec{\nabla}_{\vec{x}} \cdot \int \vec{\xi} f \mathrm{d}\vec{\xi} + \frac{\vec{F}}{\rho} \cdot \int \vec{\nabla}_{\vec{\xi}} f \mathrm{d}\vec{\xi} &= \int \Omega(f) \mathrm{d}\vec{\xi} \\ \implies \frac{\partial \rho}{\partial t} + \vec{\nabla}_{\vec{x}} \cdot (\rho \vec{u}) + 0 &= 0 \end{aligned} \tag{3.20}$$

Likewise, integrating the first moment of the BTE recovers the Cauchy Equation (2.7). It is equivalent to multiply the BTE for the velocity $\vec{\xi}$ and integrating in the velocity space:

$$\int \vec{\xi} \frac{\partial f}{\partial t} \mathrm{d}\vec{\xi} + \int \vec{\xi}\vec{\xi} \cdot \vec{\nabla}_{\vec{x}} f \mathrm{d}\vec{\xi} + \int \vec{\xi} \frac{\vec{F}}{\rho} \cdot \vec{\nabla}_{\vec{\xi}} f \mathrm{d}\vec{\xi} = \int \Omega(f) \mathrm{d}\vec{\xi} = 0 \tag{3.21}$$

Rearranging the terms:

$$\frac{\partial}{\partial t}\int \vec{\xi} f \mathrm{d}\vec{\xi} + \vec{\nabla}_{\vec{x}} \cdot \int \vec{\xi}\vec{\xi} f \mathrm{d}\vec{\xi} + \frac{\vec{F}}{\rho} \cdot \int \vec{\xi} \vec{\nabla}_{\vec{\xi}} f \mathrm{d}\vec{\xi} = 0 \tag{3.22}$$

The first term is the second momentum and the integral reduces to $\rho\vec{u}$. As for the second term, it has to be decomposed using the fact that $\vec{\xi} = \vec{u} + \vec{v}$ as:

$$\int \vec{\xi}\,\vec{\xi} f \mathrm{d}\vec{\xi} = \int \vec{u}\,\vec{\xi} f \mathrm{d}\vec{\xi} + \int \vec{v}\,\vec{\xi} f \mathrm{d}\vec{\xi} \tag{3.23}$$

Further decomposition of the first term by integrating by parts in $\vec{u}$ and $\vec{\xi} f \mathrm{d}\vec{\xi}$, and using the moments presented in Equations 3.3 and 3.4, leads to:

$$\int (\vec{u})(\vec{\xi} f) \mathrm{d}\vec{\xi} = \rho \vec{u}\vec{u} - \int (\rho \vec{u}) \left(\frac{1}{\rho} \vec{\xi} f \right) \mathrm{d}\vec{\xi} \tag{3.24}$$

Therefore:

$$\int \vec{u}\,\vec{\xi} f \mathrm{d}\vec{\xi} = \frac{1}{2} \rho \vec{u}\vec{u} \tag{3.25}$$

The second term has to be decomposed again using the fact that $\vec{v} = \vec{\xi} - \vec{u}$, yielding:

$$\int \vec{v}\vec{\xi} f \mathrm{d}\vec{\xi} = \int \vec{v}\vec{v} f \mathrm{d}\vec{\xi} + \int \vec{u}\vec{\xi} f \mathrm{d}\vec{\xi} - \int \vec{u}\vec{u} f \mathrm{d}\vec{\xi} \tag{3.26}$$

The second term is the same as in Equation 3.25, and can be promptly substituted. The third term can be shown to reduce to zero by decomposing it using an integration by parts:

$$\int \vec{u}\vec{u} f \mathrm{d}\vec{\xi} = \rho \vec{u}\vec{u} - \int \rho d(\vec{u}\vec{u}) \tag{3.27}$$

Therefore, Equation 3.23 can be written as:

$$\int \vec{\xi}\,\vec{\xi} f \mathrm{d}\vec{\xi} = \rho \vec{u}\vec{u} - \int \vec{v}\vec{v} f \mathrm{d}\vec{\xi} \tag{3.28}$$

As for the third component in Equation 3.22, performing a multi-variable integration by parts of the forcing terms it is possible to obtain:

$$\int \vec{\nabla}_{\vec{\xi}} f \mathrm{d}\vec{\xi} = 0 \tag{3.29}$$

$$\int \vec{\xi} \vec{\nabla}_{\vec{\xi}} f \mathrm{d}\vec{\xi} = -\int f \vec{\nabla}_{\vec{\xi}} \vec{\xi} \mathrm{d}\vec{\xi} = -\rho \tag{3.30}$$

Substituting all results in Equation 3.22 and simplifying by combining the terms yields:

$$\frac{\partial}{\partial t}(\rho\vec{\xi}) + \vec{\nabla}_{\vec{x}} \cdot (\rho \vec{u}\vec{u}) + \vec{\nabla}_{\vec{x}} \cdot \int \vec{v}\vec{v} f \mathrm{d}\vec{\xi} - \vec{F} = 0 \tag{3.31}$$

This is the Cauchy Equation (2.7) for a stress tensor $\vec{\vec{\sigma}}$:

$$\vec{\vec{\sigma}} = -\int \vec{v}\vec{v} f \mathrm{d}\vec{\xi} \tag{3.32}$$

Likewise, the second momentum of the equation by integrating it in relation to the velocity after being multiplied by ξ^2, yields:

$$\frac{\partial}{\partial t}(\rho E) + \vec{\nabla}_{\vec{x}} \cdot (\rho \vec{u} E) - \vec{\nabla}_{\vec{x}} \cdot (\vec{u} \cdot \vec{\vec{\sigma}}) - \vec{F} \cdot \vec{u} + \vec{\nabla}_{\vec{x}} \cdot \vec{q} = 0 \tag{3.33}$$

This is a total energy equation for a heat flux q:

$$\vec{q} = \frac{1}{2} \int \vec{v} \cdot \vec{v}\vec{v} f \mathrm{d}\vec{\xi} \tag{3.34}$$

By calculating the internal product of Equation 3.31 with $\vec{u}$ and subtracting it from Equation 3.33, which removes the bulk energy $\frac{1}{2}\rho\vec{u}^2$, the end result is an equation for the internal energy:

$$\frac{\partial}{\partial t}(\rho e) + \vec{\nabla}_{\vec{x}} \cdot (\rho \vec{u} e) - \vec{\vec{\sigma}} \cdot (\vec{\nabla}_{\vec{x}} \cdot \vec{u}) + \vec{\nabla}_{\vec{x}} \cdot \vec{q} = 0 \tag{3.35}$$

All these equations, calculated from the moments of the BTE, resemble the form of NSEs and ADEs, with macroscopic properties being then derived from the distribution function. Therefore, it has been proven that the Boltzmann Transport Equation

is adequate to represent such equations and should recover the behavior of the fluid expressed by those equations. Nonetheless, the macroscopic properties depend on the form of f directly and, thus, it would be necessary to write it in an explicit form if the macroscopic properties need to be directly obtained.

4 The Lattice Boltzmann Method

The Lattice Boltzmann Method (LBM) is based on the discretized form of the BTE (Equation 4.1) with respect to space, time, and velocities.

$$f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) = f_i(\vec{x},t) + \Omega_i(\vec{x},t) + S_i\Delta t \tag{4.1}$$

This is the general form of the Lattice Boltzmann Equation (LBE), the center piece of the Lattice Boltzmann Method. It differentiates itself for using the discrete $f_i(\vec{x},t)$ instead of the continuous $f(\vec{x},\vec{\xi},t)$. The notation f_i implies each distribution function is associated with a velocity $\vec{c}_i$ of the set of velocities $\{\vec{c}\}$. There are q equations, one for each velocity, which describes the behavior of the distribution function. The terms are associated as:

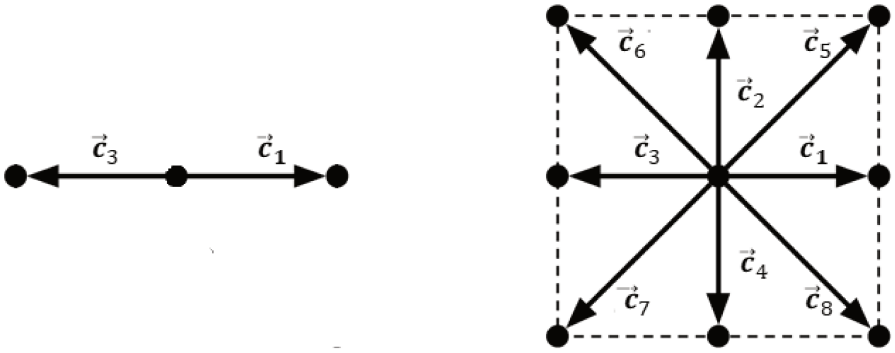
$$f_i(\vec{x},t) = f(\vec{x},\vec{c}_i,t) \tag{4.2}$$

The term S_i is linked to the force in the original BTE. The term on the right side represents the propagation of the populations to neighboring positions respecting the velocities in each direction. Krüger *et al.* (2017) show that a discrete ensemble of velocities is enough to recover the correct macroscopic momenta, as long the velocities in the set respect some restrictions.

Further details of the discretization steps are provided in Section 4.2 of this work. The LBM using the BGK collision operator (3.15) can recover the macroscopic NSEs, making it a suitable substitute for CFD computations. This can be demonstrated by the Chapman-Enskog analysis, explained in detail for the BGK collision operator in Section 4.3.

There are, however, established velocity sets consistently used in most LBM calculations (KRÜGER *et al.*, 2017). These are usually characterized by the number of dimensions d considered in the simulation, as well as the number of velocities in the set q, denoted as DdQq. Examples for the representation of these sets can be seen in Figure 1.

Usually, the sets are composed of symmetrical velocities pointing in opposite directions, as well as a resting velocity in the center. For one-dimensional simulations



(a) (b)

Figure 1 – Common velocity sets for the Lattice Boltzmann Method: (a) unidimensional D1Q3; (b) bidimensional D2Q9.

the usual set is D1Q3 (Figure 1a). For bidimensional problems, more than one option is available. At first, the lattice of choice was a hexagonal D2Q7 lattice, mostly substituted now by the D2Q9 velocity set (Figure 1b) in most simulations. As for tridimensional problems, there are more possible choices. Minor sets are D3Q15 and D3Q19. However, they may not represent accurately non-linear effects and cases where anisotropy is present, such as systems with elevated Reynolds (Re) or Mach (Ma) numbers (KRÜGER *et al.*, 2017). In such cases, a higher set is preferable and the D3Q27 is a suitable choice.

Discretization in space and time is more intuitive, as LBM uses a structured grid with nodes separated by distances Δx that are usually the same for all points, as detailed in Figure 2. This kind of mesh allows for easy application in complex geometries, one of the great attractive points of the LBM (BOEK; VENTUROLI, 2010). Nonetheless, there are works developed in unstructured and curved grids and locally refined grids (UBERTINI; SUCCI, 2005; ROSSI *et al.*, 2005; SANDOVAL, 2012; MISZTAL *et al.*, 2015).

Besides the spatial discretization, there is a time step Δt for each interaction of the LBM algorithm. The velocity of the grid can be then calculated as $c = \frac{\Delta x}{\Delta t}$. These are normally tuned to unity and converted to real unities after the simulations, as detailed in Section 4.6.

The movement of f_i is such that a population in a position $\vec{x}$ with velocity $\vec{c}$ at

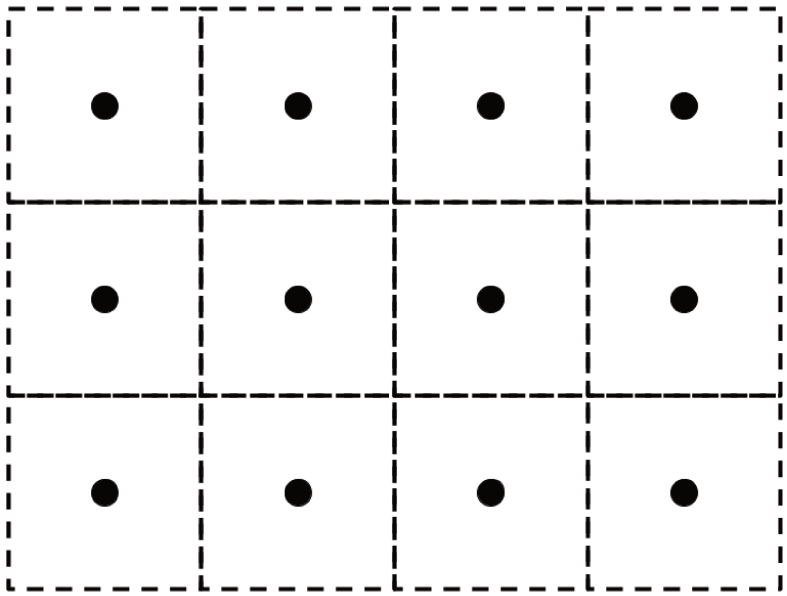


Figure 2 – Structured grid for the LBM. Each square represents a computational cell, and each point within these cells is a node with several functions f_i defined for a specific velocity set.

time t propagates to a position $\vec{x} + \vec{c}\Delta t$ in an interval Δt of time. The force also acts in each node and alters the value of the velocity in them. If the grid is uniform and velocity is a multiple of $\frac{\Delta x}{\Delta t}$, then it is clear that for each time-step the population in one node will move exactly to the center of another node. The grid values of Δx and Δt are also usually set to unity in grid values and converted after the simulation, with all other parameters converted accordingly, as already mentioned.

The same macroscopic properties, previously given by the moments of the distribution function f in Equations 3.3-3.6, can also be obtained by a summation of the populations f_i and its moments in respect to the velocities in $\{\vec{c}_i\}$, as shown in Equations 4.3-4.6. The density ρ can also be substituted by the mass density without loss of generalization, since both terms are directly related through the mass of each particle, which is a constant and does not affect the equation as a whole. The important thing is

to keep consistency, and apply the forces acting on the particles also consistently.

$$\rho(\vec{x},t) = \sum_i f_i(\vec{x},t) \tag{4.3}$$

$$\rho(\vec{x},t)\vec{u}(\vec{x},t) = \sum_i \vec{c}_i f_i(\vec{x},t) \tag{4.4}$$

$$\rho(\vec{x},t)e(\vec{x},t) = \frac{1}{2}\sum_i |\vec{c}_i|^2 f_i(\vec{x},t) \tag{4.5}$$

$$\rho(\vec{x},t)u(\vec{x},t) = \frac{1}{2}\sum_i |\vec{c}_i - \vec{u}|^2 f_i(\vec{x},t) \tag{4.6}$$

The collision operator also has to be discretized to implement the LBM. Other models exist, despite the BGK operator being widely used in LBM simulations in its discretized form (Equation 4.7). The choice of an adequate operator is important since it influences directly the accuracy of the results.

$$\Omega_i(\vec{x},t) = -\frac{\Delta t}{\tau}\left(f_i(\vec{x},t) - f_i^{\mathrm{eq}}(\vec{x},t)\right) \tag{4.7}$$

The complete equation reads as follows:

$$f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) = f_i(\vec{x},t) - \frac{\Delta t}{\tau}\left(f_i(\vec{x},t) - f_i^{\mathrm{eq}}(\vec{x},t)\right) + S_i\Delta t \tag{4.8}$$

To implement the BGK collision operator, however, it is recommended to rearrange the terms in the LBE as in Equation 4.9 to access the vector in memory less times, resulting in a more efficient algorithm.

$$f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) = \left(1 - \frac{\Delta t}{\tau}\right) f_i(\vec{x},t) + \frac{\Delta t}{\tau} f_i^{\mathrm{eq}}(\vec{x},t) + S_i\Delta t \tag{4.9}$$

The use of the BGK collision operator has a major problem associated with it, leading to a solution dependent on the relaxation time τ. This constant is related to the fluid kinematic viscosity, denoted as ν, through Equation 4.10, which obtained using the Chapman-Enskog analysis (Section 4.3). Therefore, the result is itself dependent on the viscosity of the fluid (KRÜGER *et al.*, 2017), contradicting traditional fluid dynamics. This nonphysical result introduces errors, which can mostly be neglected but exist nonetheless.

$$\nu = c_s^2\left(\tau - \frac{\Delta t}{2}\right) \tag{4.10}$$

The term c_s corresponds to the speed of sound, given by the isentropic thermodynamic relation:

$$c_s^2 = \left(\frac{\partial p}{\partial \rho}\right)_S \tag{4.11}$$

Therefore, the isothermal model for the speed of sound is obtained as:

$$p = \rho c_s^2 \tag{4.12}$$

This is also a simple equation of state (EoS) for the isothermal LBM. For the velocity sets presented, this constant can be calculated as $c_s^2 = \frac{1}{3}\left(\frac{\Delta x}{\Delta t}\right)^2$.

Other models with higher accuracy and stability for more complex collision operators do exist. They can correct the problem of the viscosity dependent solution. The two-relaxation-time (TRT) (GINZBURG; VERHAEGHE; D'HUMIERES, 2008) and the multi-relaxation-time (MRT) operators are well-established examples. These models introduce various relaxation constants related to other macroscopic properties (in the case of the MRT operator) and have been widely explored in recent works, especially when the flow is non-isothermal and monophasic (QIU *et al.*, 2019; LUO; XU, 2019; TAO *et al.*, 2019).

As for the equilibrium functions originated from the Maxwell-Boltzmann distribution, they can be discretized into polynomials of the velocity. More details of the discretization process are provided in Section 4.2. The final form of the discrete equilibrium function reads:

$$f_i^{\mathrm{eq}} = w_i \rho \left(1 + \frac{\vec{u}\cdot\vec{c}_i}{c_s^2} + \frac{(\vec{u}\cdot\vec{c}_i)^2}{2c_s^4} - \frac{\vec{u}\cdot\vec{u}}{2c_s^2}\right) \tag{4.13}$$

Each equilibrium function is associated with a weight w_i that appears naturally when performing the discretization process for creating the velocity sets. These values are associated with a Gaussian Quadrature (GQ) performed in one of the intermediate steps and are well specified for the established aforementioned sets.

The hexagonal set D2Q7 is composed of several velocities of equal value arranged pointing to the vertices of a regular hexagonal cell, resulting in equal weights. But in most cases, the length of the vectors for each velocity $\vec{c}_i$ differs in size, as it is the only manner to obtain the perfect squared and cubic cells to use in the method.

Therefore, the velocities are also different, and so are the weights. These details were considered when first establishing these grids, and now these values are tabled for the main sets. A complete scheme for these sets is presented in Table 1.

Table 1 – General properties of sets D1Q3, D2Q9, D3Q15, D3Q19, D3Q27

Set	Velocity Directions (x,y,z)	Quantity	Length (c_i)	Weight (w_i)
D1Q3	(0)	1	0	2/3
	($\pm$1)	2	1	1/6
D2Q9	(0,0)	1	0	4/9
	($\pm$1,0) (0,$\pm$1)	4	1	1/9
	($\pm$1,$\pm$1)	4	$\sqrt{2}$	1/36
D3Q15	(0,0,0)	1	0	2/9
	($\pm$1,0,0) (0,$\pm$1,0) (0,0,$\pm$1)	6	1	1/9
	($\pm$1,$\pm$1,$\pm$1)	8	$\sqrt{3}$	1/72
D3Q19	(0,0,0)	1	0	1/3
	($\pm$1,0,0) (0,$\pm$1,0) (0,0,$\pm$1)	6	1	1/18
	($\pm$1,$\pm$1,0) ($\pm$1,0,$\pm$1) (0,$\pm$1, $\pm$1)	12	$\sqrt{2}$	1/36
D3Q27	(0,0,0)	1	0	8/27
	($\pm$1,0,0) (0,$\pm$1,0) (0,0,$\pm$1)	6	1	2/27
	($\pm$1,$\pm$1,0) ($\pm$1,0,$\pm$1) (0,$\pm$1, $\pm$1)	12	$\sqrt{2}$	1/54
	($\pm$1,$\pm$1,$\pm$1)	8	$\sqrt{3}$	1/216

The BTE, and consequently the LBE, can sometimes be implemented without the terms associated with external forces, for they can be disregarded in some specific cases. The Couette and Poiseuille flows, as an example, are obtained without this term, like various other cases in which it is negligible. The forces are discretized in a similar manner to the collision operator, and the inclusion of such terms will be detailed in Section 4.5.

As for the boundary conditions, they are not straightforward to implement since mesoscopic variables cannot be measured. The conversion of the mesoscopic distribution functions into the macroscopic measured variables is well established, but the conversion of macroscopic variables into the probability distribution functions is not trivial, and there is no unique manner in which this conversion can be made. This allows for the easy implementation of microscopic effects but turns the definition of boundary conditions somewhat cumbersome. The main approaches will, therefore, be presented in Section 4.1.

Because of the way the LBM is structured, most calculations are performed locally within one node, and then propagated to neighboring nodes according to Equation 4.1. Therefore, the LBM is usually divided into two parts: collision and propagation, which have corresponding parts in the LBE.

First, the collision consists of performing the calculation within the node to obtain the value of $\Omega_i(\vec{x},t)$. The functions $f_i(\vec{x},t)$ and $f_i^{\text{eq}}(\vec{x},t)$ are used to obtain a function $f_i^*(\vec{x},t)$ as:

$$f_i^*(\vec{x},t) = f_i(\vec{x},t) + \Omega_i(\vec{x},t) + S_i\Delta t \tag{4.14}$$

Then, the time step propagation within a time step Δt consists in propagating the populations f_i^* to the corresponding neighboring nodes in the position $\vec{x}+\vec{c}_i\Delta t$ for each direction $\vec{c}_i$, as:

$$f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) = f_i^*(\vec{x},t) \tag{4.15}$$

It is useful to differentiate those steps since they are usually performed separately within the code. Therefore, when implementing the LBM, the collision step is performed with the distribution functions in the node, whereas the propagation consists only in the substitution of the values of the distribution function by the values in the neighboring node. The forces may depend on neighboring nodes, but these detail will be considered in Sections 4.5 and 7.

Moreover, this structure of LBM calculations also presents the possibility of easy parallelization, since most of the computational requirement is located within the nodes. That is an important point for optimizing simulating time. LBM is also usually implemented in uniform grids that allow for simulation of flows in porous media and other complex geometries. For such geometries, it would be very difficult to create meshes when working with traditional CFD methods (KRÜGER *et al.*, 2017).

Another advantage is the simple implementation of complex microscopic dynamics, which are lost in classical methods due to the continuum hypothesis. Microscopic effects are very difficult to implement in classical CFD methods since it is necessary to create models based on macroscopic behavior. They are more straightforward in LBM, which can naturally incorporate such effects in its distribution functions (MOHAMAD, 2011; KRÜGER *et al.*, 2017).

For these reasons, the LBM was shown to be suitable for various situations. Despite being so recent, it has spread recently and advances further in several areas (PENG, 2011). Eggels (1996), Ladd and Verberg (2001), Guo and Zhao (2002), Shan and Chen (1993) have proposed manners of using LBM to simulate turbulent flows, colloidal suspensions, porous media, and multiphase-multicomponent flows, respectively.

As for the negative points, LBM requires greater computational effort due to the presence of several distribution functions in each node in comparison to only the velocity components and density for each node in CFD. The LBM is also naturally compressible, but Zou *et al.* (1995) circumvented this problem, and as of now there are ways to simulate incompressible flows. Another problem is the difficulty that arises when working in high velocities and high Reynolds and Mach numbers, when compressibility becomes an important factor to be considered (SUCCI, 2001; NOURGALIEV *et al.*, 2003; KRÜGER *et al.*, 2017).

As an overview of the LBM has been made, the next sections will focus on the details of specific parts of the method, as well as its implementation.

4.1 Boundary Conditions

The implementation of the LBM in the bulk of the fluid is straightforward, as the calculation of the macroscopic properties from the distribution functions. The inverse, however, is not true.

Taking the D2Q9 velocity set, the distribution function is separated in 9 populations associated to each of the velocities in the set. However, the equations for calculating the macroscopic properties are only three, two of them obtained dividing the velocity momentum into its components:

$$\begin{aligned} \rho(\vec{x},t)u_x(\vec{x},t) &= \sum_i f_i(\vec{x},t)(\vec{c}_i \cdot \hat{x}) \\ \rho(\vec{x},t)u_y(\vec{x},t) &= \sum_i f_i(\vec{x},t)(\vec{c}_i \cdot \hat{y}) \end{aligned} \tag{4.16}$$

where the terms $\vec{c}_i \cdot \hat{x}$ and $\vec{c}_i \cdot \hat{y}$ represent the projection of the velocity $\vec{c}_i$ in the direction of the axis x and y, respectively.

In a wall node, only the three distribution functions with velocities directed towards it are known after the propagation, there are yet six other variables that have to be determined. Moreover, conditions of pressure and velocity apply, and one of them may be undefined. Since the equations for the known momenta are limited, the known values are usually insufficient to calculate all distribution functions. Thus, there are various approaches to calculate such values, and these differ one from another.

Overall, there are two main types of boundary conditions: contact with a solid wall; and boundaries specified by known macroscopic properties (density, pressure, velocities). Solid walls may also occur within the computational domain, examples being the flow in porous media and flow around obstacles. These solid walls can also be in motion, but they do not allow flow to pass through them. As for the fluid nodes, they are generally points of known pressure, density, or velocity and usually the inlets and outlets of the computational domain.

Boundary conditions are very important because a small error in the boundary can affect the whole domain and invalidate the results (KRÜGER *et al.*, 2017). They also define the results in most cases, since they specify the conditions of the problem. Thus, it is necessary to be very careful when implementing such conditions and to choose adequate methods.

As for the location of the boundaries relative to the nodes, there are two distinct families of methods: the walls can be placed on the nodes or half-way between the nodes, in the links of the nodes (edge of each computational cell that contains one node). These two possibilities are called, respectively, wet-node approach and link-wise approach, both illustrated in Figure 3

Note that Figures 3a and 3b are both the same conditions but are represented in a different manner. Figure 3a shows the lattice limits for each node, whereas Figure 3b shows the same domain as Figure 3c, but the computational cells are presented. In this approach the nodes are placed at the corners of the computational cells, differently of the link-wise approach, where the lattice of each node matches the computational cells. The number of nodes is also different depending on the approach, as the wet-node boundaries need more nodes for the same computational domain, as seen in Figure 3

There are some more complex boundary conditions for placing irregular walls,

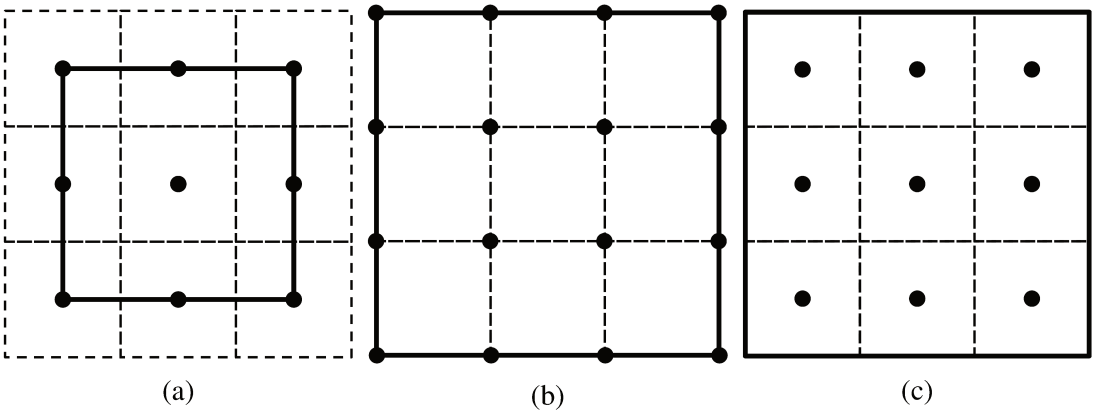


Figure 3 – Placement of the boundaries in (a,b) wet-node, and (c) link-wise approaches.

but most problems can be approximated by an image composed of pixels that allows for the easy implementation of the presented methodologies. This approximation can, however, introduce errors if this condition is not satisfied and the walls cut the nodes. That is the case when using a less refined grid to save computational power.

As for the corners in the intersection of boundaries, the principles are the same, except the existence of less known values for the distribution functions when compared to a usual straight boundary. Due to location of these nodes and the geometry, a corner node lacks all values except those propagated to it. In the D2Q9 set, as an example, only three velocities are directed towards other fluid or boundary nodes. Thus, only the values of three distribution functions, directed opposite to these three velocities, are known out of all velocities in the set.

Therefore, caution is needed when calculating the populations in these intersections. Sometimes, different boundary conditions may apply and, thus, complexity is increased. Corners that occur within the domain as the separation between the fluid and a solid wall are usually easier to calculate, but corners that connect external boundary conditions are not usually trivial.

Other important observation is that not all populations will be propagated into the fluid domain, especially in the case of corners. Some nodes receive populations from neighboring wall nodes and propagate to other wall nodes. These populations will not affect the domain, but will interfere in the velocity and density of the affected node,

needing to be set accordingly.

In reality, the true problem consists of finding values for the mesoscopic functions that respect the fluid-dynamics and macroscopic properties and, thus, there is not a unique possibility. This results in a plethora of different methodologies valid for specific cases.

Several studies were performed trying to implement satisfactory boundary conditions, starting by using the equilibrium function with given values of density and pressure (GRUNAU; CHEN; EGGERT, 1993), which introduces significant errors. The solution proposed by Skordos (1993) was the addition of terms to the equilibrium function with the gradients of pressure and velocity at the boundaries. Inamuro, Yoshino and Ogino (1995) proposed an equilibrium function modified with the addition of a counter velocity. Maier, Bernard and Grunau (1996) used different approaches for pressure and velocity when applying boundary conditions in their work. Chen, Martínez and Mei (1996) calculated the boundary conditions by adding nodes outside of the computational domain and extrapolating the values from the internal nodes, then propagating the values contained into these external nodes into the boundaries. Finally, Zou and He (1997) proposed the bounce-back rules could be applied to the non-equilibrium part of the function. Latt *et al.* (2008) summarized some of the approaches that can be used in straight boundaries, as Ho *et al.* (2009) generalized and extended the methodologies. Several other methods exist but will not be presented here, since the literature in the field is extensive.

Thereafter, the following sections will deepen further into some of the most used boundary conditions, presenting them in detail. For simplicity, the velocity set adopted in the following sections will be a D2Q9 with velocities enumerated as in Figure 1b.

4.1.1 Bounce-Back Method

The simplest and most used way to implement a boundary condition is the bounce-back method. As the name states, it consists basically of reflecting the distribution functions that reach a boundary node with the same value, but in the opposite direction, as seen in Figure 4.

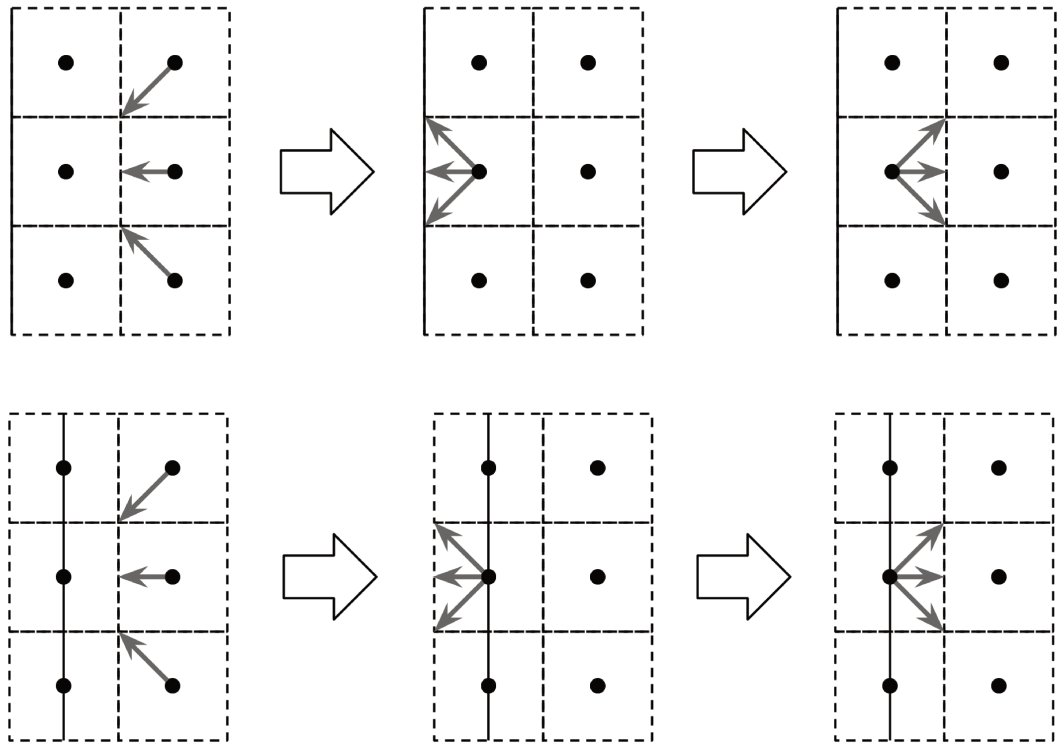


Figure 4 – Schematic representation of the link-wise (top) and wet-node (bottom) bounce-back approaches. The arrows represent the populations being propagated.

This approach states that for the unknown populations:

$$f_i(\vec{x}_b, t+\Delta t) = f_{-i}(\vec{x}_b, t) \tag{4.17}$$

Here f_{-i} corresponds to the distribution function in the opposite direction of f_i. As an example, for the D2Q9 velocity set $f_{-1} = f_3$. $\vec{x}_b$ corresponds to the position of a boundary wall.

The difference Δt is due to the propagation steps that occur between the propagation of a population into a boundary node, and the reflection. If the populations are reflected in the same time-step the following equation ensues:

$$f_i(\vec{x}_b, t) = f_{-i}(\vec{x}_b, t) \tag{4.18}$$

As for the populations parallel to the wall, the values are established for link-wise approaches. For a wet-node approach, one possible approximation is that there is no propagation in this direction, which may result in appropriate results in some cases. This is only possible when there is no velocity applied to the wall. For the link-wise approach, the boundary nodes may be placed within the solid. In this case, the propagation parallel to the wall does not matter, since these populations will never affect the fluid domain.

Now, a wider approach is to consider the propagation, in which case the populations parallel to the wall are known. The density can be calculated from the known values without need to use the uncertain information.

For the calculation of the density and velocity along a top wall, from Equations 4.3 and 4.4:

$$\begin{aligned} \rho &= (f_0 + f_1 + f_2 + f_3 + f_5 + f_6)_{\text{known}} + (f_4 + f_7 + f_8)_{\text{unknown}} \\ \rho u_y &= (f_2 + f_5 + f_6)_{\text{known}} - (f_4 + f_7 + f_8)_{\text{unknown}} \end{aligned} \tag{4.19}$$

The sum of these equations yields:

$$\rho(1 + u_y) = f_0 + f_1 + f_3 + 2(f_2 + f_5 + f_6) \tag{4.20}$$

For a resting wall, $u_y = 0$. All the other values are known and, thus, the density can be determined without compromising the results with the unknown values.

The bounce-back approaches can also be used for symmetric problems by implementing the boundary nodes in the plane of symmetry.

4.1.2 Bounce-Back with Moving Wall

The bounce-back method can also be applied to a boundary with a defined velocity $\vec{u}_b$. This is the case for a no-slip condition of a fluid with a moving wall. In this case, the fluid assumes the same velocity as the wall, which reflects in the distribution function.

The distribution functions may, in this case, be transformed to the frame of the wall to perform the bounce-back method. this results in:

$$f_i(\vec{x}_b, t + \Delta t) = f_{-i}(\vec{x}_b, t) - 2w_i\rho_b \frac{\vec{c}_i \cdot \vec{u}}{c_s^2} \tag{4.21}$$

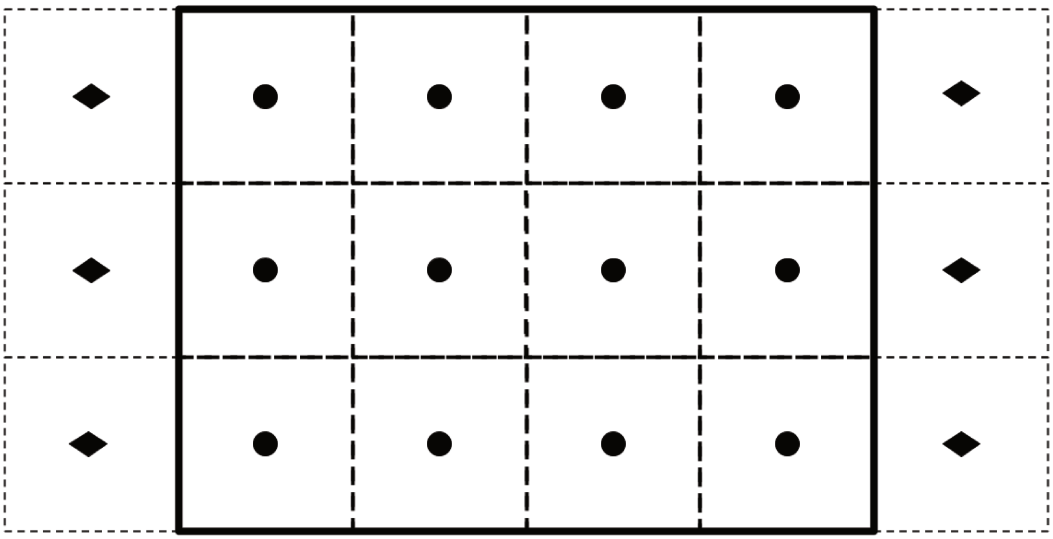


Figure 5 – Representation of the virtual nodes outside of the computational domain.

where ρ_b is the density at the boundary. The term associated with the velocity vanishes when $u_b = 0$, recovering the original bounce-back method.

4.1.3 Periodic Domains

In some specific cases, symmetry allows for the implementation of a periodic boundary condition. When a repeated flow pattern exists within a domain, a periodic solution can be found, allowing for the implementation of periodic boundary conditions to simulate the pattern. This method is implemented by adding one layer of virtual nodes outside of the boundaries, and the values that exit the domains on one side are transferred to the opposite side, re-entering the domain.

These virtual nodes are truly located outside of the domain and, therefore, are not part of the solution. Figure 5 presents an scheme for the virtual nodes on the inlet and outlet.

A very basic periodic flow is the Couette flow. It is characterized by fluid contained between two plates, with the flow caused by the motion of one of them resulting in a linear velocity profile.

In the case of a periodic flow in the x direction through a domain with N_x nodes

in the x direction, it is possible to write:

$$f_i(0,t) = f_i(\Delta x N_x, t) \tag{4.22}$$

$$f_i(\Delta x N_x, t) = f_i(0,t) \tag{4.23}$$

In this case, the velocities and densities of the virtual nodes do not need to be accurate and, thus, there is no necessity of calculating the unknown values of the distribution functions. All the necessary values are provided by the populations flowing out of the domain in the opposite side.

4.1.4 Periodic Domains with Pressure Drop

In some cases, boundaries may be periodic but a pressure drop can be present along the direction of the periodic boundaries. One such example is the Poiseuille flow.

As stated in Equation 4.12, the pressure can be related to the density, which can be implemented in the LBM through the distribution functions. This leads to a density difference $\Delta\rho$, which yields:

$$\rho(0,t) = \rho(\Delta x N_x, t) + \Delta\rho \tag{4.24}$$

$$\rho(0,t)\vec{u}(0,t) = \rho(\Delta x N_x, t)\vec{u}(\Delta x N_x, t) \tag{4.25}$$

This is clearly a compressible flow, since a variation in the density occurs. As the periodic boundary conditions are enforced for the momentum of the fluid, mass and velocity conservation cannot be guaranteed. Hence, an incompressible form of the LBM has to be used in such cases. This is achieved by using the pressure p as:

$$p(0,t) = p(\Delta x N_x, t) + \Delta p \tag{4.26}$$

$$\vec{u}(0,t) = \vec{u}(\Delta x N_x, t) \tag{4.27}$$

To impose the periodic boundary conditions with pressure drop, one manner is to separate the distribution function within the nodes in an equilibrium and a non-equilibrium part (KIM; PITSCH, 2007). This non-equilibrium part can be understood as the difference between the distribution in one point and the correspondent equilibrium. Therefore:

$$f_i = f_i^{\mathrm{eq}} + f_i^{\mathrm{neq}} \tag{4.28}$$

This leads to:

$$f_i = f_i^{\mathrm{eq}} + (f_i - f_i^{\mathrm{eq}}) \tag{4.29}$$

This equation is equivalent to add and to subtract the equilibrium distribution function f_i^{eq}. The equilibrium part is evaluated using the density prescribed at the boundaries, related to the pressure through Equation 4.12, whereas the non-equilibrium part is obtained from the opposite side. The final result reads:

$$f_i(0,t) = f_i^{\mathrm{eq}}(p_1, u(\Delta x N_x, t)) + (f_i(\Delta x N_x, t) - f_i^{\mathrm{eq}}(\Delta x N_x, t)) \tag{4.30}$$

$$f_i(\Delta x N_x, t) = f_i^{\mathrm{eq}}(p_N, u(0,t)) + (f_i(0,t) - f_i^{\mathrm{eq}}(\Delta x N_x, t)) \tag{4.31}$$

The pressure at one of the sides can be set at unity, the other being related to it through the known pressure drop Δp.

4.1.5 Equilibrium Scheme

The equilibrium scheme is also simple to implement. It consists of setting the distribution functions at the boundaries to the equilibrium populations of the same density and velocity. Therefore:

$$f_i(\vec{x}_b, t) = f_i^{\mathrm{eq}}(\rho_b, \vec{u}_b) \tag{4.32}$$

This equilibrium applies to all distribution functions within the node, including the known ones.

This is a method with high stability (KRÜGER *et al.*, 2017) and easy applicability, but sometimes it may be not so accurate as other methods. Furthermore, it needs both prescribed density and velocity. This problem can be mitigated by using the values of neighboring nodes, or those obtained after propagation and before collision. However, these approximations can reduce accuracy and applicability.

4.1.6 Non-Equilibrium Bounce-Back Zou-He Method

The non-equilibrium bounce-back method, as the name states, consists of equating the non-equilibrium parts of the distribution functions and reflecting these parts instead of the incoming populations directly.

Zou and He (1997) proposed the bounce-back method could be applied to the non-equilibrium parts of the distribution functions to calculate the unknown values. This means the non-equilibrium distribution function propagating in direction $\vec{c}_i$ will be reversed, that is, the distribution function will be transferred to the direction opposed to its original direction given by $\vec{c}_{-i}$.

$$f_i^{\text{neq}} = f_{-i}^{\text{neq}} \tag{4.33}$$

These non-equilibrium distribution functions are given by Equation 4.28. Thereafter:

$$f_i - f_i^{\text{eq}} = f_{-i} - f_{-i}^{\text{eq}} \tag{4.34}$$

Zou and He conditions result in the addition of more equations to those used for calculating the macroscopic momentum, allowing the computation of the unknown values. Their approach became popular in implementing such conditions for LBM applications. Ho *et al.* (2009) generalized the ideas presenting consistent boundary conditions that recover some of these other methods as well.

These conditions can be used if density or velocity are specified at a boundary node. Specifying density is uncommon, as the pressure value is usually the known variable. This pressure can be converted into density with Equation 4.12.

Rearranging Equation 4.34 results in:

$$f_i = f_{-i} + (f_i^{\text{eq}} - f_{-i}^{\text{eq}}) \tag{4.35}$$

Since the equilibrium function is well known and can be calculated as detailed in Equation 4.13, this assumption adds equations to the problem. Noting that $\vec{c}_i^2 - \vec{c}_{-i}^2$, the equation can be further simplified to:

$$f_i = f_{-i} + w_i \rho \left(\frac{1}{c_s^2} \vec{u} \cdot (\vec{c}_i - \vec{c}_{-i}) \right) \tag{4.36}$$

The density and velocities are still given by:

$$\rho = f_0 + f_1 + f_2 + f_3 + f_4 + f_5 + f_6 + f_7 + f_8 \tag{4.37}$$

$$\rho u_x = (f_1 + f_5 + f_8) - (f_3 + f_6 + f_7) \tag{4.38}$$

$$\rho u_y = (f_2 + f_5 + f_6) - (f_4 + f_7 + f_8) \tag{4.39}$$

Depending on the location of the boundary, Equation 4.38 or 4.39 is added or subtracted from Equation 4.37 to eliminate the unknown functions and determine the density or velocity on the boundary node, according to the other defined variable. For a top wall:

$$\rho(1+u_y) = f_0 + f_1 + f_3 + 2(f_2 + f_5 + f_6) \tag{4.40}$$

To close the system, the non-equilibrium bounce-back is assumed, using Equation 4.36 to find the unknown populations:

$$f_4 = f_2 - \frac{2}{c_s^2}\rho u_y \tag{4.41}$$

$$f_7 = f_5 + \frac{1}{2}(f_1 - f_3) - \frac{3}{2c_s^2}\rho u_x - \frac{1}{2c_s^2}\rho u_y \tag{4.42}$$

$$f_8 = f_6 - \frac{1}{2}(f_1 - f_3) + \frac{3}{2c_s^2}\rho u_x - \frac{1}{2c_s^2}\rho u_y \tag{4.43}$$

These boundary conditions can, therefore, be used to calculate the distribution functions for a given pressure or velocity if the other value is given, using Equation 4.40.

As this method incorporates the non-equilibrium parts of the distribution functions, it is more accurate than other methods. This method and other derived methodologies became very popular due to its simplicity, and are still used to this day. The difficulty lies in expanding the Zou-He method to three dimensions.

4.1.7 Anti-Bounce-Back Method

The anti-bounce-back approach can be used to prescribe a density (or pressure) in the boundary nodes. This technique is similar to the non-equilibrium bounce-back method, but it changes the sign of the populations, resulting in different terms being considered:

$$f_i = -f_{-i} + f_i^{\mathrm{eq}} + f_{-i}^{\mathrm{eq}} \tag{4.44}$$

Substituting Equation 4.13 and simplifying the terms that reduce to zero:

$$f_i = -f_{-i} + 2w_i\rho_b\left(1 + \frac{(\vec{c}_i \cdot \vec{u}_b)^2}{2c_s^4} - \frac{u_b^2}{2c_s^2}\right) \tag{4.45}$$

If the velocity is unknown, it may be extrapolated from the neighboring nodes and inserted into Equation 4.45.

4.1.8 Virtual Nodes

Another possibility is to use virtual nodes by extending the computational domains. These nodes are not part of the solution but the required bounce-back is performed in them. The real physical boundary, however, is not located on these nodes but on the neighboring nodes. It is an approximation when it is impossible to obtain the distribution functions for specific boundary conditions.

This approach is similar to that performed by Shan and Chen (1993) for multicomponent fluids, yielding satisfactory results and being easy to implement. On the virtual nodes, one of the presented methods to prescribe the properties are yet to be performed, but in some cases a simple bounce-back approach may be enough, turning a complicated problem into a much simpler one.

As it does not present anything new, at least in terms of the calculations of the populations, this method will not be detailed further at this moment.

4.1.9 Further Considerations

These are some of the approaches used in the context of LBM simulations to prescribe conditions at the boundaries. They show how a variety of methods exists. Many more techniques were proposed, but the ones detailed in this work are enough for an introduction to the topic and allow for comprehending the general approach.

The final results of an LBM simulation are also heavily dependent on the choice of the boundary conditions (KRÜGER *et al.*, 2017). Specific problems may also ask for different approaches, depending on the desired result. Therefore, it is useful to know how to implement these different techniques and how they are derived. It is also important to know how these methods were created to derive new conditions if necessary by using the same tools.

With the acquired knowledge of the method and boundary conditions, the next sections will be dedicated to exploring the discretization of the BTE to acquire the LBE.

The macroscopic equations of transport will also be shown to be recovered by performing the Chapman-Enskog analysis. Then, the details about the implementation will be provided, and LBM will be expanded to more complex flows, subjected to external forces and multicomponent fluids.

4.2 Discretization of the Boltzmann Transport Equation

The Lattice Boltzmann Equation (LBE) was already presented, as it was explained its origins from the Boltzmann Transport Equation (BTE). Now, this section is dedicated to show how this discretization is performed, and how these two equations are linked.

First, the general BTE reads:

$$\frac{\partial f}{\partial t} + \vec{\xi} \cdot \vec{\nabla}_{\vec{x}} f + \frac{\vec{F}}{\rho} \cdot \vec{\nabla}_{\vec{\xi}} f = \Omega(f) \tag{4.46}$$

As its definition states, f is dependent on the position $\vec{x}$ and time t, but also of the velocity of the particles $\vec{\xi}$. This dependence on the velocity is what differs the Lattice Boltzmann Method (LBM) from most CFD methods, since a discretization in the velocity space is also to be performed.

Note the absolute values of f are of little interest, since the moments of f entail what describes the macroscopic behavior. There are different functions that yields the same values as the moments of f that can be chosen, since the microscopic physics does not need to be completely known. Therefore, a finite number of velocities can be used to represent the behavior of the fluid.

Two approaches can be taken in the discretization of the BTE. It is possible to perform a Mach number (Ma) expansion of the distribution function (HE; LUO, 1997) or to perform an expansion using Hermite polynomials (SHAN; YUAN; CHEN, 2006). Both approaches yield the same results, but the latter was chosen on this work due to its strong mathematical basis.

First, a dimensionless form for the BTE reads the same as the original LBE (KRÜGER *et al.*, 2017). The equilibrium distribution function is given in its dimen-

sionless form as:

$$f_{\text{eq}}(\rho, \vec{u}, \theta, \vec{\xi}) = \rho \left(\frac{1}{2\pi\theta}\right)^{d/2} \exp\left\{-\frac{\upsilon^2}{2\theta}\right\} \tag{4.47}$$

where d is the number of spatial dimensions, and $\theta = \frac{RT}{\upsilon_0^2}$ is a dimensionless temperature related to a characteristic velocity υ_0. As the collision operator conserves the momenta of f, the momenta of f^{eq} coincide with f.

Now, for the discretization, the Hermite polynomials are defined as:

$$H^{(n)}(r) = \frac{(-1)^n}{\omega(r)} \frac{\mathrm{d}^n}{\mathrm{d}x^n} \omega(r) \tag{4.48}$$

For a dependent variable r, which is not necessarily the position, n indicates the order of the respective polynomial with a generating function $\omega(r)$ defined as:

$$\omega(r) = \frac{1}{\sqrt{2\pi}} e^{-r^2/2} \tag{4.49}$$

For more dimensions, these polynomials can be extended to:

$$\vec{H}^{(n)}(\vec{r}) = \frac{(-1)^n}{\omega(\vec{r})} \vec{\nabla}^{(n)} \omega(\vec{r}) \tag{4.50}$$

$$\omega(\vec{r}) = \frac{1}{(2\pi)^{d/2}} e^{-r^2/2} \tag{4.51}$$

The variable $\vec{r}$ is now a vector and both $\vec{H}^{(n)}$ and $\vec{\nabla}^{(n)}$ are tensors of rank n, with the latter being a generalization of the gradient operator $\vec{\nabla}$.

These polynomials can be shown to be orthogonal in respect to r, thus forming a complete basis in which any function can be represented:

$$\int_{-\infty}^{+\infty} \omega(r) H^{(n)}(r) H^{(m)}(r) dr = n! \delta_{nm} \tag{4.52}$$

where δ_{nm} is the usual Kronecker delta, defined as 1 if $n = m$, and 0 otherwise. This assumption, albeit not demonstrated here, also holds true for the vector variable $\vec{r}$.

Therefore, any function $h(\vec{r})$ can be represented as a series of Hermite polynomials:

$$h(\vec{r}) = \omega(\vec{r}) \sum_{n=0}^{\infty} \frac{1}{n!} \vec{a}^{(n)} \cdot \vec{H}^{(n)} \tag{4.53}$$

The term $\vec{a}^{(n)}$ is a tensor of rank n, equivalent to coordinates for the function h in relation to the basis formed by the Hermite polynomials. These coordinates uniquely define the function h in terms of Hermite polynomials, and can be calculated as:

$$\vec{a}^{(n)} = \int h(\vec{r})\vec{H}^{(n)}(\vec{r})\mathrm{d}\vec{r} \tag{4.54}$$

The equilibrium distribution function f_{eq} given by Equation 4.47 is an exponential function, which is similar to the exponential generating function ω and can be written as:

$$f_{\mathrm{eq}}(\rho,\theta,\vec{v}) = \frac{\rho}{\theta^{d/2}}\omega\left(\frac{\vec{v}}{\sqrt{\theta}}\right) \tag{4.55}$$

Representing the equilibrium function in the Hermite polynomials basis yields:

$$f_{\mathrm{eq}}(\rho,\theta,\vec{v}) = \omega(\vec{\xi})\sum_{n=0}^{\infty}\frac{1}{n!}\vec{a}_{\mathrm{eq}}^{(n)}\cdot\vec{H}^{(n)}(\vec{\xi}) \tag{4.56}$$

The coefficients $\vec{a}_{\mathrm{eq}}^{(n)}$ can be calculated as:

$$\vec{a}_{\mathrm{eq}}^{(n)}(\rho,\vec{u},\theta) = \int f_{\mathrm{eq}}(\rho,\theta,\vec{v})\vec{H}^{(n)}(\vec{\xi})\mathrm{d}\vec{\xi} = \frac{\rho}{\theta^{d/2}}\int\omega\left(\frac{\vec{v}}{\sqrt{\theta}}\right)\vec{H}^{(n)}(\vec{\xi})\mathrm{d}\vec{\xi} \tag{4.57}$$

Moreover, the coefficients can also be expressed in terms of the momenta of the distribution functions as:

$$\vec{a}_{\mathrm{eq}}^{(0)} = \int f_{\mathrm{eq}}\mathrm{d}\vec{\xi} = \rho = \int f\mathrm{d}\vec{\xi} = \vec{a}^{(0)} \tag{4.58}$$

$$\vec{a}_{\mathrm{eq}}^{(1)} = \int f_{\mathrm{eq}}\vec{\xi}\mathrm{d}\vec{\xi} = \rho\vec{u} = \int f\vec{\xi}\mathrm{d}\vec{\xi} = \vec{a}^{(1)} \tag{4.59}$$

$$\vec{a}_{\mathrm{eq}}^{(2)} = \int f_{\mathrm{eq}}\xi^2\mathrm{d}\vec{\xi} - \rho d = 2\rho E - \rho d = \int f\xi^2\mathrm{d}\vec{\xi} - \rho d = \vec{a}^{(2)} \tag{4.60}$$

These relations are the core reason as to why the expansion in Hermite polynomials are useful to represent the distribution functions. Note the coefficient for the functions are the same as those of the equilibrium functions, which can be calculated, since the equilibrium functions have an explicit form.

To reproduce the macroscopic behavior, Equations 4.58 to 4.60 show that only the first three terms are necessary, since the important macroscopic momenta are those

up to the second-order, given by $\vec{a}^{(2)}$. This allows for a significant reduction of the computational effort.

Therefore, rewriting the distribution functions with the approximation up to the second-order (third term in the Hermite polynomial series):

$$f = \omega(\vec{\xi}) \left(\vec{a}^{(0)} \cdot \vec{H}^{(0)} + \frac{1}{2} \vec{a}^{(1)} \cdot \vec{H}^{(1)} + \frac{1}{6} \vec{a}^{(2)} \cdot \vec{H}^{(2)} \right) \tag{4.61}$$

There is a mathematical tool that allows an integral of a polynomial $\vec{P}^{(n)}$ to be calculated as the sum over a finite set of points. The results are exact when the number of points in the set is equal or greater than the order n of the polynomial. This is the Gauss-Hermite quadrature (Equation 4.62), which can be used to integrate the Hermite polynomial form of f.

$$\int_{-\infty}^{+\infty} \omega(\vec{r}) \vec{P}^{(n)}(\vec{r}) \mathrm{d}\vec{r} = \sum_{i=1}^{N} w_i \vec{P}^{(n)}(\vec{r}) \tag{4.62}$$

As shown in Equation 4.62, the integration can be broken into a sum evaluated in a discrete set of N points with associated weights w_i. If the polynomial evaluated corresponds to the Hermite polynomials expansion of f, then:

$$\vec{P}^{(2)} = \vec{a}^{(0)} \cdot \vec{H}^{(0)} + \frac{1}{2} \vec{a}^{(1)} \cdot \vec{H}^{(1)} + \frac{1}{6} \vec{a}^{(2)} \cdot \vec{H}^{(2)} \tag{4.63}$$

$$\int_{-\infty}^{+\infty} \omega(\vec{\xi}) \vec{P}^{(2)}(\vec{\xi}) \mathrm{d}\vec{\xi} = \sum_{i=1}^{N} w_i \vec{P}^{(2)}(\vec{\xi}_i) \tag{4.64}$$

To ensure the macroscopic properties are respected, it is necessary to correctly integrate this polynomial up to the second-order term, equivalent to the second momentum. Therefore, the macroscopic momenta are exactly calculated using the set of discrete velocities $\left\{ \vec{\xi}_i \right\}$ of Equation 4.64.

The final distribution function for a isothermal case, for each velocity in the set $\vec{\xi}_i$, is:

$$f_i^{\mathrm{eq}} = w_i \rho \left(1 + \vec{u} \cdot \vec{\xi}_i + \frac{1}{2} \left(\vec{u} \cdot \vec{\xi}_i \right)^2 - \frac{1}{2} u^2 \right) \tag{4.65}$$

Finally, to simplify the velocities, $\vec{\xi}_i$ is substituted for $\vec{c}_i$, related through:

$$\vec{c}_i = c_s \vec{\xi}_i \tag{4.66}$$

This results in the discretized distribution function as presented in Equation 4.13. The distribution function f can be simplified using the same velocity sets as f_i^{eq} as:

$$f_i(\vec{x},t) = \frac{w_i}{\omega(\vec{c}_i)} f(\vec{x},\vec{c}_i,t) \tag{4.67}$$

The term $\frac{w_i}{\omega(\vec{c}_i)}$ is added to satisfy the Gauss-Hermite quadrature. This discrete distribution functions recover the same macroscopic moments as the equilibrium distribution functions and, therefore, are viable. So, the BTE is rewritten substituting the functions f for the discrete counterparts f_i.

Forces are discretized in the same manner, resulting in a contribution:

$$F_i(\vec{x},t) = -\frac{w_i}{\omega(\vec{c}_i)} \frac{\vec{F}}{\rho} \cdot \vec{\nabla}_{\vec{c}_i} f \tag{4.68}$$

Performing the same expansion as performed for f_eq, respecting the second-order momenta, yields:

$$F_i = w_i \left(\frac{\vec{c}_i}{c_s^2} + \frac{(\vec{c}_i \cdot \vec{u}) \cdot \vec{c}_i}{c_s^4} - \frac{\vec{u}}{c_s^2} \right) \cdot \vec{F} \tag{4.69}$$

However, the discretization is yet to be performed in space and time. There is a strong coupling in the original LBM, since the space and time steps $\Delta\vec{x}$ and Δt are linked through the velocities in the set $(\vec{c}_i)$. This happens because it is desired that the populations in one node propagate to the exactly position of one other node within the grid. This is the reason why most LBM simulations focuses on structure grids, despite methods for unstructured grids and local refinement do exist (UBERTINI; SUCCI, 2005; ROSSI *et al.*, 2005; SANDOVAL, 2012; MISZTAL *et al.*, 2015).

There are several ways to discretize the BTE in relation to space and time. In particular, it is possible to write the distribution function parameterized in relation to a variable ζ as:

$$f_i(\vec{x},t) = f_i(\vec{x}(\zeta), t(\zeta)) \tag{4.70}$$

The total differential is given as:

$$\frac{df_i}{d\zeta} = \left(\frac{\partial f_i}{\partial t} \right) \frac{\partial t}{\partial \zeta} + \vec{\nabla} f_i \cdot \frac{\partial \vec{x}}{\partial \zeta} \tag{4.71}$$

By equating the total differential to the collision operator summed to the force term, it is possible to recover the BTE for $\frac{\partial t}{\partial \zeta} = 1$ and $\frac{\partial \vec{x}}{\partial \zeta} = \vec{c}_i$. The resulting equation is then integrated:

$$\int_t^{t+\Delta t} \frac{df_i}{d\zeta} d\zeta = f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) - f_i(\vec{x},t) = \int_t^{t+\Delta t} (\Omega_i + F_i) d\zeta \tag{4.72}$$

The collision operator in its discretized form in velocity space is assumed to depend only on the discretized forms of f and f_{eq} as $\Omega_i = \Omega(f_i, f_i^{\text{eq}})$. Therefore, it is unnecessary to perform the discretization of the collision operator in the velocity space.

As the right side of Equation 4.72 is ready and exact, the only part missing is to integrate the collision operator and forcing term. This is usually achieved by making a first or second-order approximation of the integral.

The first order discretization is accomplished by approximating the integral for the absolute values in a point as:

$$\int_t^{t+\Delta t} (\Omega_i + F_i) d\zeta = \Delta t \left(\Omega_i(\vec{x},t) + \vec{F}(\vec{x},t)\right) \tag{4.73}$$

As for the second-order integration, the integral is approximated as:

$$\int_t^{t+\Delta t} (\Omega_i + F_i) d\zeta = \Delta t \left(\frac{\Omega_i(\vec{x},t) + \Omega(\vec{x}+\vec{c}_i\Delta t, t+\Delta t)}{2} + \frac{F_i(\vec{x},t) + F_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t)}{2} \right) \tag{4.74}$$

A change of variables can be performed as:

$$f_i' = f_i - \Delta t \left(\frac{\Omega_i + F_i}{2} \right) \tag{4.75}$$

This change results in a equation for the substituted variable f' of the form:

$$f_i'(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) - f_i'(\vec{x},t) = (\Omega_i' + F_i') \tag{4.76}$$

The collision operator and force term are also modified. Such modifications are actually very simple. In the case of the BGK collision operator given by Equation 4.7:

$$\Omega_i' = -\frac{\Delta t}{\tau + \frac{\Delta t}{2}} (f_i' - f_i^{\text{eq}}) = -\frac{\Delta t}{\tau'} (f_i' - f_i^{\text{eq}}) \tag{4.77}$$

The equilibrium distribution f_i^{eq} is the same as the function for f_i, and it is possible to redefine the relaxation constant as $\tau' = 1 + \frac{\Delta t}{2}$.

Since the absolute value of f is unimportant, as long as the macroscopic momenta are recovered, the forms of the equations for the first and second-order discretization are equivalent. Therefore, for the LBE without forces these two forms are equivalent.

The force term is usually substituted by a generic S_i, which may include the second-order discretization or not. For the first order discretization $S_i = F_i$, directly recovering the LBE (Equation 4.1). For the second-order discretization, however:

$$S_i = F_i' = \left(1 - \frac{\Delta t}{2\tau'}\right) F_i \tag{4.78}$$

Therefore, the second-order discretization can be omitted and, as the forms are equivalent. Representing f_i' as f_i the final form of the LBE can be written as:

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = f_i(\vec{x}, t) + \Omega_i(\vec{x}, t) + S_i \Delta t \tag{4.79}$$

This is the same form seen in Equation 4.1, granting a second-order accuracy if the correct force terms are used.

There are yet other methods to solve the integral, as any numerical method for integration could be used. Nonetheless, these require intensive computational steps that affect the simplicity of LBM and, thus, its attractiveness.

4.3 The Chapman-Enskog Analysis

The Boltzmann Transport Equation (BTE) was previously shown to recover its macroscopic counterparts: the Continuity Equation (CE), Navier-Stokes Equations (NSEs), and Advection-Diffusion Equations (ADEs). Nonetheless, it does not automatically extend for the Lattice Boltzmann Equation (LBE). Therefore, the present section provides a means to recover these macroscopic equations in the form of the so-called Chapman-Enskog analysis. This provides further proof of the validity of the Lattice Boltzmann Method (LBM) as an NSE solver for fluids.

The Chapman-Enskog analysis was developed separately by Sydney Chapman and David Enskog around 1917, then combined and summarized by the former (CHAPMAN; COWLING, 1952). This analysis aims to recover macroscopic equations of movement from the original BTE and initiates with an expansion of f around f^{eq}, using the Knudsen number (Kn) as the expansion parameter.

Applying the Chapman-Enskog analysis to the LBM demonstrates the applicability of LBM for solving problems related to NSEs by recovering the macroscopic balance equations. The first step is to expand the LBE (Equation 4.9) with the BGK collision operator in a Taylor series up until the second-order term as:

$$\Delta t\left(\frac{\partial f_i}{\partial t}+\vec{c}_i\cdot\nabla f_i\right)+\frac{\Delta t^2}{2}\left(\frac{\partial^2 f_i}{\partial t^2}+c_i^2\nabla^2 f_i\right)+O\left(\Delta t^3\right)=-\frac{\Delta t}{\tau}\left(f_i-f_i^{\text{eq}}\right) \tag{4.80}$$

The terms in $O\left(\Delta t^3\right)$ represent the expansion of the terms higher than second-order, which will not be considered in the next steps of the analysis.

By subtracting $\frac{\Delta t}{2}\left(\frac{\partial}{\partial t}+\vec{c}_i\cdot\nabla\right)$ applied Equation 4.80 from itself:

$$\Delta t\left(\frac{\partial}{\partial t}+\vec{c}_i\cdot\nabla\right)f_i=-\frac{\Delta t}{\tau}\left(f_i-f_i^{\text{eq}}\right)+\frac{\Delta t^2}{2\tau}\left(\frac{\partial}{\partial t}+\vec{c}_i\cdot\nabla\right)\left(f_i-f_i^{\text{eq}}\right) \tag{4.81}$$

Now, for the next step it is necessary to expand the derivatives of f using the Knudsen number (Kn) as in Equation 4.82. The spacial derivative is not expanded beyond first order.

$$\Delta t\frac{\partial f_i}{\partial t}=\Delta t\left(\text{Kn}\frac{\partial f_i}{\partial t}+\text{Kn}^2\frac{\partial^2 f_i}{\partial t^2}+\text{Kn}^3\frac{\partial^3 f_i}{\partial t^3}+\ldots\right) \tag{4.82}$$

$$\Delta t\vec{c}_i\cdot\nabla f_i=\Delta t\,\text{Kn}\,\vec{c}_i\cdot\nabla f_i \tag{4.83}$$

Considering the terms of the expansion bellow second-order and applying Equations 4.82 and 4.83 to Equation 4.81:

$$\Delta t\left(\text{Kn}\frac{\partial}{\partial t}+\text{Kn}^2\frac{\partial^2}{\partial t^2}+\text{Kn}\,\vec{c}_i\cdot\nabla\right)f_i+\frac{\Delta t}{\tau}\left(f_i-f_i^{\text{eq}}\right)=\frac{\Delta t^2}{2\tau}\left(\text{Kn}\frac{\partial}{\partial t}+\text{Kn}^2\frac{\partial^2}{\partial t^2}+\text{Kn}\,\vec{c}_i\cdot\nabla\right)\left(f_i-f_i^{\text{eq}}\right) \tag{4.84}$$

Then, it is possible to perform a perturbation expansion of f_i around f_i^{eq} also using Kn as a parameter:

$$f_i = f_i^{\text{eq}} + \text{Kn} f_i^{(1)} + \text{Kn}^2 f_i^{(2)} + \text{Kn}^3 f_i^{(3)} + ... \tag{4.85}$$

Substituting the terms of the expansion up to second-order in Equation 4.84:

$$\begin{aligned}\Delta t \left(\text{Kn}\frac{\partial}{\partial t} + \text{Kn}^2\frac{\partial^2}{\partial t^2} + \text{Kn}\,\vec{c}_i\cdot\nabla\right)\left(f_i^{\text{eq}} + \text{Kn} f_i^{(1)} + \text{Kn}^2\right) + \frac{\Delta t}{\tau}\left(\text{Kn} f_i^{(1)} + \text{Kn}^2 f_i^{(2)}\right) \\ = \frac{\Delta t^2}{2\tau}\left(\text{Kn}\frac{\partial}{\partial t} + \text{Kn}^2\frac{\partial^2}{\partial t^2} + \text{Kn}\,\vec{c}_i\cdot\nabla\right)\left(\text{Kn} f_i^{(1)} + \text{Kn}^2 f_i^{(2)}\right)\end{aligned} \tag{4.86}$$

By separating the terms of the same order in Kn:

$$\frac{\partial}{\partial t} f_i^{\text{eq}} + \vec{c}_i\cdot\nabla f_i^{\text{eq}} = -\frac{1}{\tau} f_i^{(1)} \tag{4.87}$$

$$\frac{\partial^2}{\partial t^2} f_i^{\text{eq}} + \Delta t\left(1 - \frac{\Delta t}{2\tau}\right)\left(\frac{\partial}{\partial t} f_i^{(1)} + \vec{c}_i\cdot\nabla f_i^{(1)}\right) = -\frac{1}{\tau} f_i^{(2)} \tag{4.88}$$

At this point, the force term was not considered in the calculations. In the presence of forces, the term F_i is expanded as the gradient as $\text{Kn} F_i^{(1)}$. The final result of the equations in this case is as follows:

$$\frac{\partial}{\partial t} f_i^{\text{eq}} + \vec{c}_i\cdot\nabla f_i^{\text{eq}} - \left(1 - \frac{\Delta t}{2\tau}\right) F_i^{(1)} = -\frac{1}{\tau} f_i^{(1)} \tag{4.89}$$

$$\frac{\partial^2}{\partial t^2} f_i^{\text{eq}} + \left(1 - \frac{\Delta t}{2\tau}\right)\left(\frac{\partial}{\partial t} + \vec{c}_i\cdot\nabla\right) f_i^{(1)} + \frac{\Delta t}{2}\left(1 - \frac{\Delta t}{2\tau}\right)\left(\frac{\partial}{\partial t} + \vec{c}_i\cdot\nabla\right) F_i^{(1)} = -\frac{1}{\tau} f_i^{(2)} \tag{4.90}$$

By taking the momenta of these equations it is possible to obtain macroscopic quantities. Calculating the first moment of Equation 4.89 yields:

$$\frac{\partial}{\partial t}\sum_i f_i^{\text{eq}} + \nabla\cdot\sum_i \vec{c}_i f_i^{\text{eq}} - \left(1 - \frac{\Delta t}{2\tau}\right)\sum_i F_i^{(1)} = -\sum_i \frac{1}{\tau} f_i^{(1)} \tag{4.91}$$

As stated in Section 3, the collision operator has to conserve mass, momentum, and energy. Therefore, the momenta of the collision operator are zero in the case of a system without forces. For general systems, this conservation is ensured including

forces. In the absence of these forces ($F_i = 0$), the conservation equations revert back to the momenta of the collision operator being zero. In the case of the BGK collision operator:

$$\Omega_i = -\frac{\Delta t}{\tau} f_i^{\text{neq}} = -\frac{\Delta t}{\tau}(f_i - f_i^{\text{eq}}) \tag{4.92}$$

To respect the conserved quantities:

$$\sum_i (f_i - f_i^{\text{eq}}) + \frac{\Delta t}{2}\sum_i F_i^{(1)} = 0 \tag{4.93}$$

$$\sum_i (f_i - f_i^{\text{eq}}) + \frac{\Delta t}{2}\sum_i F_i^{(1)} = 0 \tag{4.94}$$

If these equations are assumed to hold for every order of discretization, then:

$$\sum_i f_i^{(1)} + \frac{\Delta t}{2}\sum_i F_i^{(1)} = 0 \qquad \sum_i f_i^{(n)} = 0 \ (n > 1) \tag{4.95}$$

$$\sum_i \vec{c}_i f_i^{(1)} + \frac{\Delta t}{2}\sum_i F_i^{(1)} = 0 \qquad \sum_i \vec{c}_i f_i^{(n)} = 0 \ (n > 1) \tag{4.96}$$

Returning to Equation 4.91, by applying these definitions and substituting the other moments as in Equations 4.3-4.6, it directly recovers the continuity equation:

$$\frac{\partial}{\partial t}\rho + \nabla \cdot (\rho \vec{u}) = 0 \tag{4.97}$$

The second moment is calculated in the same manner, yielding:

$$\frac{\partial}{\partial t}(\rho \vec{u}) + \nabla \cdot \vec{\vec{\Pi}}_{\text{eq}} = \vec{F} \tag{4.98}$$

The tensor $\vec{\vec{\Pi}}_{\text{eq}}$ is the second moment of the equilibrium distribution function given by:

$$\vec{\vec{\Pi}}_{\text{eq}} = \rho \vec{u}\vec{u} + (\rho c_s^2)\vec{\vec{I}} \tag{4.99}$$

For the LBM, $\rho c_s^2 = p$. Thus, Equation 4.98 is equivalent to the Euler Equation (2.6) seen previously in Section 2. Therefore, the first order approximation recovers the Euler Equation.

As for the second-order equation (4.90), the moments are given by:

$$\nabla^2 \rho = 0 \tag{4.100}$$

$$\frac{\partial^2}{\partial t^2}(\rho \vec{u}) + \left(1 - \frac{\Delta t}{2\tau}\right) \nabla \cdot \left(\sum_i f_i^{(1)} \vec{c}_i \vec{c}_i + \frac{\Delta t}{2} \sum_i F_i^{(1)} \vec{c}_i \vec{c}_i \right) = 0 \tag{4.101}$$

Here, the term $\sum_i f_i^{(1)} \vec{c}_i \vec{c}_i + \frac{\Delta t}{2} \sum_i F_i^{(1)} \vec{c}_i \vec{c}_i$ can be represented as $\vec{\vec{\Pi}}^{(1)}$. It appears when calculating the second moment of Equation 4.90 using Equations 4.95 and 4.96.

If the sum of the first and second momentum equations for the parameters of different orders are calculated:

$$\left(\mathrm{Kn} \frac{\partial}{\partial t} + \mathrm{Kn}^2 \frac{\partial^2}{\partial t^2} \right) \rho + \mathrm{Kn} \nabla \cdot (\rho \vec{u}_i) = 0 \tag{4.102}$$

$$\left(\mathrm{Kn} \frac{\partial}{\partial t} + \mathrm{Kn}^2 \frac{\partial^2}{\partial t^2} \right) (\rho \vec{u}_i) + \mathrm{Kn} \nabla \cdot \vec{\vec{\Pi}}_{\mathrm{eq}} = \mathrm{Kn} \vec{F} - \mathrm{Kn}^2 \left(1 - \frac{\Delta t}{2\tau}\right) \nabla \cdot \vec{\vec{\Pi}}^{(1)} \tag{4.103}$$

The terms can be recombined by reverting the expansion made using Kn:

$$\frac{\partial}{\partial t}(\rho \vec{u}) + \nabla \cdot \vec{\vec{\Pi}}_{\mathrm{eq}} = \vec{F} - \left(1 - \frac{\Delta t}{2\tau}\right) \nabla \cdot \vec{\vec{\Pi}}^{(1)} \tag{4.104}$$

To finish the equations, an explicit value for $\vec{\vec{\Pi}}^{(1)}$ must be calculated. The second moment of Equation 4.89 yields:

$$\frac{\partial}{\partial t} \vec{\vec{\Pi}}_{\mathrm{eq}} + \nabla \cdot \vec{\Pi}_{\mathrm{eq}}^{(3)} - \left(1 - \frac{\Delta t}{2\tau}\right) \sum_i F_i^{(1)} \vec{c}_i \vec{c}_i = -\frac{1}{\tau} \vec{\vec{\Pi}}^{(1)} \tag{4.105}$$

This is a tensorial equation. The third order tensor $\vec{\Pi}_{\mathrm{eq}}^{(3)}$ corresponds to the third momentum of f_i^{eq}, defined as:

$$\vec{\Pi}_{\mathrm{eq}}^{(3)} = \sum_i f_i^{\mathrm{eq}} \vec{c}_i \vec{c}_i \vec{c}_i \tag{4.106}$$

Equation 4.105 can be rearranged as:

$$\vec{\vec{\Pi}}^{(1)} = -\tau \left(\frac{\partial}{\partial t} \vec{\vec{\Pi}}_{\mathrm{eq}} + \nabla \cdot \vec{\Pi}_{\mathrm{eq}}^{(3)} - \left(1 - \frac{\Delta t}{2\tau}\right) \sum_i F_i^{(1)} \vec{c}_i \vec{c}_i \right) \tag{4.107}$$

The force term $\sum_i F_i^{(1)} \vec{c}_i \vec{c}_i$ ensues a correction when force is applied. It does not need to appear in the definition of $\vec{\vec{\Pi}}^{(1)}$, as it is clearly related to the viscous stress tensor as seen by comparing Equation 4.104. Thus, a definition for the tensor can be written as:

$$\vec{\vec{\Pi}}^{(1)} = -\tau \left(\frac{\partial}{\partial t} \vec{\vec{\Pi}}_{\text{eq}} + \nabla \cdot \vec{\Pi}_{\text{eq}}^{(3)} \right) \tag{4.108}$$

Both moments can be calculated by their definitions as:

$$\vec{\vec{\Pi}}_{\text{eq}} = \sum_i f_i^{\text{eq}} \vec{c}_i \vec{c}_i = \rho \vec{u}\vec{u} + (\rho c_s^2) \vec{\vec{I}} \tag{4.109}$$

$$\vec{\Pi}_{\text{eq}}^{(3)} = \sum_i f_i^{\text{eq}} \vec{c}_i \vec{c}_i \vec{c}_i = \rho \vec{u}\vec{u}\vec{u} + (\rho c_s^2) \vec{u}\vec{\vec{I}} \tag{4.110}$$

The time derivative in 4.107 can be rewritten as spatial derivatives:

$$\frac{\partial}{\partial t}\rho = -\nabla \cdot (\rho \vec{u}) \qquad \frac{\partial}{\partial t}(\rho \vec{u}) = -\nabla \cdot \left(\rho \vec{u}\vec{u} + \rho c_s^2 \vec{\vec{I}} \right) \tag{4.111}$$

The time derivative of $\vec{\vec{\Pi}}_{\text{eq}}$ thus reads:

$$\frac{\partial}{\partial t} \vec{\vec{\Pi}}_{\text{eq}} = \frac{\partial}{\partial t}(\rho \vec{u}\vec{u}) + \frac{\partial}{\partial t}(\rho c_s^2 \vec{\vec{I}}) = 2\vec{u} \frac{\partial}{\partial t}(\rho \vec{u}) - \vec{u}\vec{u} \frac{\partial}{\partial t}\rho + c_s^2 \vec{\vec{I}} \frac{\partial}{\partial t}\rho \tag{4.112}$$

$$\frac{\partial}{\partial t} \vec{\vec{\Pi}}_{\text{eq}} = -2\vec{u}\nabla \cdot \left(\rho \vec{u}\vec{u} + \rho c_s^2 \vec{\vec{I}} \right) + \vec{u}\vec{u}\nabla \cdot (\rho \vec{u}) - c_s^2 \vec{\vec{I}} \nabla \cdot (\rho \vec{u}) \tag{4.113}$$

$$\frac{\partial}{\partial t} \vec{\vec{\Pi}}_{\text{eq}} = -2\vec{u}\nabla \cdot (\rho \vec{u}\vec{u}) + \vec{u}\vec{u}\nabla \cdot (\rho \vec{u}) - 2c_s^2 \vec{u}\nabla\rho - c_s^2 \vec{\vec{I}} \nabla \cdot (\rho \vec{u}) \tag{4.114}$$

$$\frac{\partial}{\partial t} \vec{\vec{\Pi}}_{\text{eq}} = -\nabla \cdot (\rho \vec{u}\vec{u}\vec{u}) - 2c_s^2 \vec{u}\nabla\rho - c_s^2 \vec{\vec{I}} \nabla \cdot (\rho \vec{u}) \tag{4.115}$$

As for the third order tensor $\vec{\Pi}_{\text{eq}}^{(3)}$, it can be calculated as:

$$\nabla \cdot \vec{\Pi}_{\text{eq}}^{(3)} = \nabla(\rho \vec{u}\vec{u}\vec{u}) + c_s^2 \nabla \cdot (\rho \vec{u}\vec{\vec{I}}) = \nabla(\rho \vec{u}\vec{u}\vec{u}) + 2c_s^2 \nabla(\rho \vec{u}) + c_s^2 \vec{\vec{I}} \nabla \cdot (\rho \vec{u}) \tag{4.116}$$

$$\nabla \cdot \vec{\Pi}_{\text{eq}}^{(3)} = \nabla(\rho \vec{u}\vec{u}\vec{u}) + 2c_s^2 \rho \nabla \vec{u} + 2c_s^2 \vec{u}\nabla\rho + c_s^2 \vec{\vec{I}} \nabla \cdot (\rho \vec{u}) \tag{4.117}$$

Substituting the calculated derivatives in Equation 4.108 and simplifying the equal terms yields:

$$\vec{\vec{\Pi}}^{(1)} = -\tau 2 c_s^2 \rho \nabla \vec{u} \tag{4.118}$$

Finally, Equation 4.104 can be written as:

$$\frac{\partial}{\partial t}(\rho\vec{u}) + \nabla\cdot\left(\rho\vec{u}\vec{u} + (\rho c_s^2)\vec{\vec{I}}\right) = \vec{F} - \left(1 - \frac{\Delta t}{2\tau}\right)\nabla\cdot\vec{\vec{\Pi}}^{(1)} \tag{4.119}$$

Comparing Equations 2.9 with the NSE (4.119) it is clear that the NSE is recovered. The viscous stress tensor $\vec{\vec{\sigma}}_\mu$ is given as:

$$\vec{\vec{\sigma}}_\mu = \frac{1}{\tau\rho c_s^2}\vec{\vec{\Pi}}^{(1)} \tag{4.120}$$

And the viscosity is found to be:

$$\frac{\mu}{\rho} = \nu = \tau c_s^2\left(1 - \frac{\Delta t}{2\tau}\right) \tag{4.121}$$

This is the same as Equation 4.10. Therefore, these two macroscopic properties were shown to be intimately related to the relaxation constant τ of LBM. Furthermore, the LBE was shown to be an adequate alternative as an NSE solver. For other collision operators, the analysis follows the same steps. The parameters of other collision operators are also related to other macroscopic constants.

4.4 Compressibility

The Lattice Boltzmann Method is naturally compressible, since the equation of state for the LBM (Equation 4.12) establishes a direct relation between pressure and density. It is a strength when simulating compressible flows. However, it is difficult to adequately simulate incompressible flows, which the NSEs naturally do. Thereafter, several modifications were proposed with the simulation of incompressible flows.

For a density variation $\Delta\rho$ that is small when compared to a reference density of the system ρ_0, the compressibility errors are also small and sometimes negligible. Another approach, which was used at the beginning of the LBM, is to substitute the pressure difference for a body-force that acts as a momentum source for the particles of the fluid. This force approach presents some limitations, but it eliminates the compressibility effects. More details about force implementation will be explained in Section 4.5, and the relation between pressure gradients and forces will be exemplified in Section 5.3. The extension to other systems is trivial.

Nonetheless, most approaches propose to calculate incompressible flows, when the density variation is negligible, are built by altering the equilibrium functions. These new distributions are made to recover the incompressible NSE upon being analyzed through the Chapman-Enskog analysis (Section 4.3).

Krüger *et al.* (2017) present a model for incompressibility in the LBM. It is achieved by implementing a new equilibrium function that reads:

$$f_i^{\text{eq}} = w_i\rho + w_i\rho_0\left(\frac{\vec{u}\cdot\vec{c}_i}{c_s^2} + \frac{(\vec{u}\cdot\vec{c}_i)^2}{2c_s^4} - \frac{\vec{u}\cdot\vec{u}}{2c_s^2}\right) \tag{4.122}$$

In this proposal, the only density that depends on the LBM is the local density ρ. Therefore, the part of the equilibrium function related to the velocities is multiplied by a reference density ρ_0, which is common to the whole computational domain.

Zou *et al.* (1995) proposed the use of the moment densities given by $\vec{U} = \rho\vec{u}$ in the equilibrium functions to guarantee $\nabla(\rho\vec{u}) = 0$ in the steady-state. The equilibrium distribution for their model is calculated as:

$$f_i^{\text{eq}} = w_i\left(\rho + \frac{\vec{U}\cdot\vec{c}_i}{c_s^2} + \frac{\left(\vec{U}\cdot\vec{c}_i\right)^2}{2c_s^4} - \frac{\vec{U}\cdot\vec{U}}{2c_s^2}\right) \tag{4.123}$$

In a general manner, alterations of the equilibrium distribution will alter the macroscopic equations recovered by the Chapman-Enskog analysis, which allows for the inclusion of other effects unexplored in this moment. More examples of modifications in the case of multiple components will be presented in Section 7. For a single component fluid, a general form may be written as:

$$f_i^{\text{eq}} = w_i\rho\left(1 + a_0\frac{\vec{u}\cdot\vec{c}_i}{c_s^2} + a_1\frac{(\vec{u}\cdot\vec{c}_i)^2}{2c_s^4} - a_2\frac{\vec{u}\cdot\vec{u}}{2c_s^2}\right) \tag{4.124}$$

The recovered macroscopic equation, thus, will depend on the value of a_1, a_2, and a_3, which can be constants or functions of other variables such as the local and general densities, as in the cases presented.

4.5 Forces

The Lattice Boltzamnn Method (LBM), as presented, can be promptly implemented for simple fluid flows in the absence of forces and other source terms. Implementing forces in the LBM method, however, is not trivial and some modifications have to be made to properly represent these terms. The forces are implemented as force densities. In the case of $\vec{g}$ (gravity), for example:

$$\vec{g} = \frac{\vec{F}_g}{\rho} \tag{4.125}$$

Force densities that generate a constant force field, as in the case of gravity for the scales considered, can be implemented as a body-force that influences or drive the flow. Due to the force terms, differential densities or temperatures of mixing fluids can give rise to buoyancy effects. These forces can also be implemented as a way to drive the flow by substituting or supplementing a differential pressure.

The velocity in redefined to guarantee second-order accuracy as:

$$\rho\vec{u} = \sum_i (f_i\vec{c}_i) + \vec{F}\frac{\Delta t}{2} \tag{4.126}$$

The forces are discretized as in Equation 4.78 in a term as S_i:

$$S_i = \left(1 - \frac{\Delta t}{2\tau'}\right) F_i \tag{4.127}$$

In this scheme, the velocity that enters the equilibrium function is given by Equation 4.126, and f_i^{eq} is still calculated as in Equation 4.13. This method is called Guo-force, since it was derived by Guo, Zheng and Shi (2002).

However, this is not the only approach to decompose the equilibrium function in the collision operator and the force term. There are several manners to relate these terms that recover the NSEs in the macroscopic level. So, as long as the term $\Omega_i + S_i$ is kept the same, the independent forms of Ω_i and S_i do not matter at a macroscopic level.

A generalization of the velocities that enters the equilibrium function $\vec{u}_{\text{eq}}$, yields:

$$\rho\vec{u}_{\text{eq}} = \sum_i (f_i\vec{c}_i) + a\vec{F}\frac{\Delta t}{2} \tag{4.128}$$

There are several models that vary the value of the constant a. The forcing term S_i is also altered accordingly to keep $\Omega_i + S_i$ constant. Guo-force is, therefore, given by:

$$a = \frac{1}{2} \qquad\qquad S_i = \left(1 - \frac{\Delta t}{2\tau}\right) \tag{4.129}$$

Another example of a different force method was proposed by Shan and Chen (1993) to calculate multi-phase flows. Their model disregards the force term S_i, implementing the necessary modifications within the collision operator by modifying the value of a:

$$a = \tau \Delta t \qquad\qquad S_i = 0 \tag{4.130}$$

An alternative was proposed by Kupershtokh (2004) based on kinetic theory to include forces in a manner that would not affect the collision operator, only the force term. Therefore, the velocity is redefined as:

$$\rho \vec{u}^* = \sum_i (f_i \vec{c}_i) \qquad\qquad \rho \Delta \vec{u}^* = \vec{F} \Delta t \tag{4.131}$$

The collision is calculated at $\vec{u}^*$ and the force term becomes:

$$S_i = f_i^{\mathrm{eq}}(\rho, \vec{u}^* + \Delta \vec{u}^*) - f_i^{\mathrm{eq}}(\rho, \vec{u}^*) \tag{4.132}$$

This scheme basically states that the equilibrium is altered by a force that causes a difference in velocity $\Delta \vec{u}^*$ to appear.

Guo forcing is easier to implement, since the necessary modifications are made in the velocity and the other parts of the method continue the same. The scheme proposed by Shan-Chen was developed for multicomponent flows, but can be used to simulate simple flows. Nonetheless, this scheme adds a term involving τ to the force scheme that leads to a dependence of the surface tension with the viscosity (KRÜGER *et al.*, 2017). As for the Kupershtokh method, it has the advantage to lead to an equilibrium solution after the force is implemented and mix the collision and force terms. Nonetheless, the Guo-force scheme is enough in most simulations.

4.6 Units Conversion

The final step for implementing the Lattice Boltzmann Method (LBM) consists of establishing the units to convert physical variables to simulation units and to obtain the results in the usual units. Due to the strong coupling of space and time because of the velocity discretization, it is also important to choose proper units when simulating a physical system using the LBM.

The simulations are usually implemented for a system of dimensionless lattice units (LU), usually set to unity. These are associated with real units according to each problem. The usual step is to choose proper spatial and temporal steps, respectively Δx and Δt, which will be associated with physical units. The other variables will be then calculated from the characteristic equations of the LBM and their definitions. Another common LU is the density, which can be represented in lattice units as ρ_0, usually also set to unity.

A conversion factor can be established using the relation between the lattice units and the physical units. The basic conversion factors are associated with length C_L, time C_t, and mass or density C_ρ. They can be represented as follows:

$$\text{Length:} \quad C_L = \frac{(\Delta x)_{\text{physical}}}{(\Delta x)_{\text{lattice}}} = \Delta x^* \tag{4.133}$$

$$\text{Time:} \quad C_t = \frac{(\Delta t)_{\text{physical}}}{(\Delta t)_{\text{lattice}}} = \Delta t^* \tag{4.134}$$

$$\text{Mass:} \quad C_\rho = \frac{\rho}{\rho_0} = \rho^* \tag{4.135}$$

The conversion factors given by Δx^*, Δt^*, and ρ^* are the physical values corresponding to the simulation steps Δx, Δt, and ρ_0, which are set to unity, and all parameters represented using the superscript as in the equations of conversion are to be treated as representing the physical units, whereas the variable without superscript represents the same variables in the lattice units system.

The other factors are calculated from these and the definitions. The factor for conversion of lattice velocity c, for example, can be calculated as:

$$c = \frac{\Delta x}{\Delta t} \Rightarrow C_u = \frac{\Delta x^*}{\Delta t^*} \tag{4.136}$$

Therefore, any physical velocity u^* can be calculated as $u^* = u\frac{\Delta x^*}{\Delta t^*}$. The sound speed ($c_s = \frac{1}{\sqrt{3}}$ in lattice units), as an example, can be calculated as:

$$c_s^* = \frac{1}{\sqrt{3}}\frac{\Delta x^*}{\Delta t^*} \tag{4.137}$$

Another example is the kinematic viscosity given by Equation 4.10:

$$\begin{gathered} \nu = c_s^2\left(\tau - \frac{\Delta t}{2}\right) \Rightarrow \nu^* = \frac{1}{3}\left(\frac{\Delta x^*}{\Delta t^*}\right)^2\left(\tau - \frac{\Delta t}{2}\right)\Delta t^* \\ \therefore \nu^* = \frac{1}{3}\left(\tau - \frac{1}{2}\right)\frac{(\Delta x^*)^2}{\Delta t^*} \end{gathered} \tag{4.138}$$

The pressure may be calculated as indicated in Equation 4.12. However, only the differences in pressure are correctly calculated by the indicated equation of state. In reality, any reference pressure p_0^* can be simulated if this equation is divided as follows:

$$\begin{gathered} p^* = p_0^* + p'^* = p_0^* + \rho'^*(c_s^2)^* \\ p^* = p_0^* + \rho' c_s^2 \rho^* \left(\frac{\Delta x^*}{\Delta t^*}\right)^2 \end{gathered} \tag{4.139}$$

In this case ρ' is given by $\rho = \rho_0 + \rho'$. So, differences of density convert into differences of pressure, but the absolute values are allowed to be arbitrary, as long as the conversion factor $C_p = \rho^*\left(\frac{\Delta x^*}{\Delta t^*}\right)^2$ is respected (KRÜGER *et al.*, 2017).

The conversion of forces has to be performed carefully, since the factors are different for force densities, acceleration, and forces. The term force is usually used to represent all those quantities, but it is necessary to be careful when converting such factors.

The conversion for acceleration is obtained as $C_g = \frac{\Delta x}{(\Delta t)^2}$. So, the gravity can be converted to lattice units as:

$$g = g^*\frac{(\Delta t)^2}{\Delta x} \tag{4.140}$$

The force density has to consider the density to be properly converted:

$$F = \rho g = \rho g^*\frac{(\Delta t)^2}{\Delta x} \tag{4.141}$$

The conversion factor for the force reads $C_F = \rho^* \frac{\Delta x}{(\Delta t)^2}$. All other usual variables can be calculated in the same manner, and reduce to the same three conversion factors. Therefore, it is possible to correctly set a system that represents a real flow.

Furthermore, dimensionless numbers such as Reynolds (Re), Biot (Bi), Knudsen (Kn), Mach (Ma), and several others, are characterized by a division between two terms of the same units to correlate the system state to its intrinsic properties. Therefore, the same Re holds for the system, no matter if it is calculated in lattice units, or in physical units. So, the Reynolds number is enough to characterize the system, and the other properties must be adjusted to guarantee maximum accuracy between the behavior of a given physical system and its simulation.

One final thing to keep in mind when simulating using the LBM is the strong dependence between the spatial steps, the time steps, and the pressure related to the density by Equation 4.12. Therefore, for the same system, changes in the value of one unit cause changes in the value of the other. This is a dependency characteristic of the LBM, and in extreme cases can lead to instability (KRÜGER *et al.*, 2017).

5 Implementation and Validation

In this work, a program was elaborated and implemented using the Lattice Boltzmann Method (LBM) to simulate flow in several conditions. The first step to test the code was the comparison to well-known analytical solutions, as commonly done in the literature (HO *et al.*, 2009; PENG, 2011; PORTINARI, 2015). The results were compared to analytical solutions of Poiseuille and Couette flows, and the error computed. Other more complex geometries were also tested. Since for such geometries no analytical solution can be obtained, qualitative comparisons were made to experimental results available in the literature for the field and CFD simulations. This was the methodology used to evaluate the LBM for more complex systems.

The LBM simulations were performed using the code implemented by the author, which is available in Appendix A. The used computer language was Python (version 2.6), which is an interpreted language. This language type does not need to be compiled and can be executed directly. The other possibility was the use of compiled languages, such as C and Fortran, that are converted to a machine language file prior to execution.

Compiled languages tend to be faster in execution, and the compilation step tend to remove errors and can be used to optimize the code. For simple program the difference is barely noticeable, but as the complexity and execution time increase, the differences can become significant. Nonetheless, interpreted languages tend to be very intuitive, fast to write, and easy to understand. The programs also tend to be smaller, containing less lines when compared to an equivalent program written in C or Fortran. Moreover, Python is faster to program, contains several additional libraries, and is a well-established language, widely used for scientific computing.

5.1 Simulation Conditions

All simulations within the scope of this study were implemented in two dimensions using a D2Q9 velocity set, as represented in Figure 6. This is a common grid for

the LBM with several boundary conditions already available. Furthermore, due to its symmetry and square shape, it can be easily implemented in regular grids and pixelated images. Regular meshes were used, and the size of the grids varied according to the number of points, which are divided into fluid or solid (boundary) nodes.

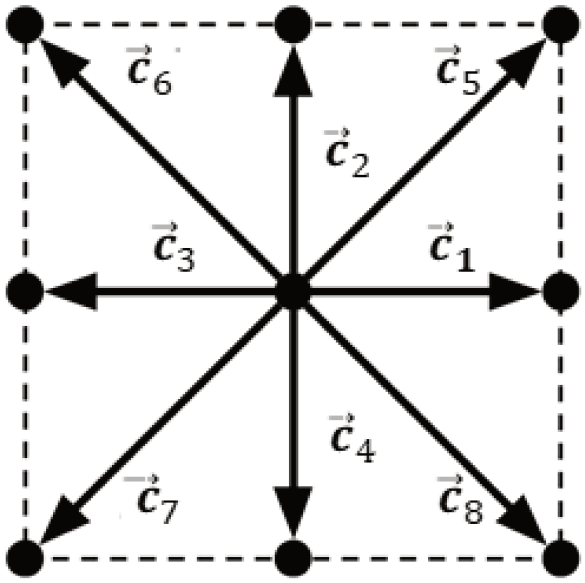


Figure 6 – Schematic representation of the D2Q9 velocity set.

The fluid was an ideal Newtonian fluid with non-interacting particles, which translates in the absence of internal forces between the nodes. Flow was monophasic and non-reactive. The flow in inherently isothermal, since other versions of LBM should be used when considering heat transfer and temperature gradients.

As LBM is compressible, to compare the results to the analytical incompressible solutions available for the NSE, the equilibrium distribution function proposed by Zou *et al.* (1995) (Equation 4.123) was used:

$$f_i^{\mathrm{eq}} = w_i \left(\rho + \frac{\vec{U} \cdot \vec{c}_i}{c_s^2} + \frac{\left(\vec{U} \cdot \vec{c}_i\right)^2}{2c_s^4} - \frac{\vec{U} \cdot \vec{U}}{2c_s^2} \right)$$

where $\vec{U} = \rho \vec{u}$ represents the momentum density instead of the usual velocity.

As previously stated, due to the nature of the Lattice Boltzmann Method (LBM), several associated methodologies exist for its implementation. The wet-node approach presented in Section 4.1 was mostly used. For all stationary solid walls presented, including internal walls within the domain, the used approach was the bounce-back (BB) technique as described in Section 4.1.1. Conditions of pressure and velocity are set on the fluid boundaries using the non-equilibrium bounce-back (NEBB) method (Section 4.36).

Simulations were evaluated in the stationary state. Flow evolved until convergence was achieved. The criteria used was a tolerance value $\varepsilon = 10^{-8}$ for the total sum of the velocities:

$$\sum \left(|\vec{u}_n(\vec{x},t) - \vec{u}_m(\vec{x},t)| \right) < \varepsilon \tag{5.1}$$

This sum is performed within the whole computational domain, but excluding the virtual nodes when they are used. The indices n and m indicate different iterations. The convergence is not tested at the end of each iteration, but after an established number of time steps given by $n-m$, which was set to 100 in most studied cases. It was considered that an error smaller than ε in the sum of velocities was enough to guarantee the convergence, as the value is very small.

The difference between the velocity found with the LBM simulations u and the calculated analytical velocity $u^\star$ was then evaluated by calculating the Average Absolute Relative Deviation (AARD) and the Root Mean Square Deviation (RMSD), represented respectively in Equations 5.2 and 5.3, for all N points in the lattice. The AARD is a relative deviation, independent of the valor of the variable and, thereby, easy to analyze. The RMSD is an absolute deviation, associated with the values at each point and giving a tangible value:

$$\text{AARD} = \frac{1}{N} \sum \left| \frac{u^\star - u}{u^\star} \right| \tag{5.2}$$

$$\text{RMSD} = \sqrt{\frac{1}{N} \sum (u^\star - u)^2} \tag{5.3}$$

Used units were lattice units (LU), unitary for space (Δx), time (Δt), and reference density (ρ_0). Therefore, from now on, the units will be omitted and must be interpreted as such, unless indicated otherwise. The units can be easily converted to real units by defining conversion factors for space and time as explained in Section 4.6.

The viscosity can be calculated from the value set to the relaxation constant τ as indicated in Equation 4.10. Various values around unity were tested to see the effects on the results for the Poiseuille and Couette flows. Based on the results, the value $\tau = 1$ was chosen for all other simulations.

5.2 Implementation Steps

LBM algorithm consists of several steps, as shown in Figure 7. In the ensuing sections, the explanation for these steps will be provided.

5.2.1 Initialization

To initialize the populations, a profile of density and velocity may be specified, and the distribution functions may be found to satisfy those profiles or proceed to calculate the equilibrium functions from those values. This is the case for tests that will cause modifications in the pre-established behavior of a flow, such as the addition of an obstacle, and are useful for time-dependent flows.

Another approach may be simply setting the velocity to zero and the density to a specified value, usually unity. The distributions functions can be placed with the values of the equilibrium populations. This second approach is more easily applicable

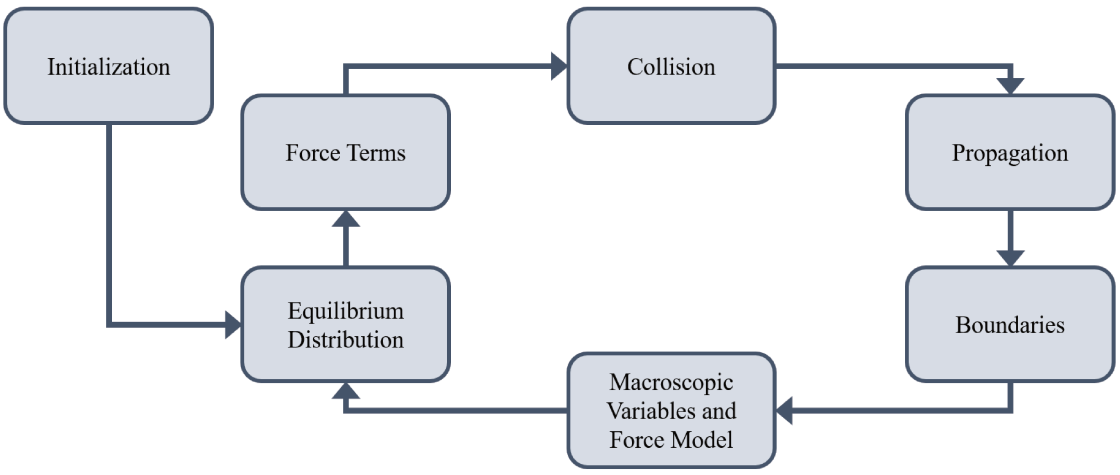


Figure 7 – Execution steps for the Lattice Boltzmann Method.

to steady-state flows, since the initial position is not as important, or for stagnated flows that starts moving along the simulation.

The initialization steps for steady-state (time-independent) flows are usually not as important, since the terms associated with transient values tend to disappear. That is also the case for periodic flows in time evaluated after a long period has elapsed. The non-cyclic time dependency also tends to disappear after a number of cycles have passed.

Nonetheless, for time-dependent flows, when there is a need to evaluate how the system evolves with time, the initialization becomes relevant. For a better initialization, the non-equilibrium distributions can be calculated to improve the values placed for f_i on each node.

In this work, the initial values did not affect much the final results. Density was set to 1 and the velocity to 0 in all simulations, unless stated otherwise. The distribution populations were set to the equilibrium with these values.

5.2.2 Collision

First, it is necessary to calculate the equilibrium functions if not already done. Then, the collision step is performed by executing all steps in the Lattice Boltzmann Equation, which does not involve interaction with neighboring nodes, as previously indicated in Equation 4.14, which is centered only in each position $\vec{x}$.

The force term must be applied before the collision steps finish. The order is not important in most cases as it does not affect the results. However, some cases may make use of the values before the collision or neighboring nodes and must be carefully executed. It is important to notice what are the effects of the force term, to make sure the changes are being applied to the correct populations.

5.2.3 Propagation

The propagation step is indicated by the displacement of the distribution functions to neighboring nodes, in the direction of the correspondent velocity. The bulk of the fluid propagates normally according to Equation 4.15, but in the boundaries, precau-

tionary steps need to be taken according to the chosen method.

Usually, LBM when implemented needs two populations because of the propagation steps, since it substitutes the values present in one node before it can be propagated in one of the directions. It is possible to avoid such situations by propagating in one direction only after the other is finished, or by propagating in different ways in each direction, but such steps make the code more complicated.

5.2.4 Boundaries

After propagation, the values at boundaries are calculated with proper methods. Section 4.1 present a list of possible methods, but several more exist in LBM related literature. Some methodologies may invert the order of the collision, propagation, and boundary steps, so it is necessary to be very careful when writing the LBM code.

5.2.5 Macroscopic Variables Update

This step relates to the update of the macroscopic values from the distribution functions. It takes place after all other steps are concluded and finishes by incrementing the time variable, proceeding to the next step.

The first part of the process is to compute the model for the force in the corresponding step, but not yet the complete force term, since it requires the velocity in the process according to Equation 4.69. This first step depends on the model used to calculate the force, and not of LBM itself. Then, the macroscopic momenta are updated as in Equations 4.3-4.6.

Next, the equilibrium distributions are calculated using the obtained macroscopic variables. Finally, the force term is calculated and the algorithm proceeds to the collision step described in Section 5.2.2.

5.3 Poiseuille Flow

The Poiseuille flow is commonly used to validate LBM implementation, as analytical solutions are avilable for this geometry. The geometry is characterized as a

two-dimensional flow between two infinite plates established by a difference of pressure between the inlet and outlet. This flow is characterized by a parabolic profile of velocity, as shown in Figure 8.

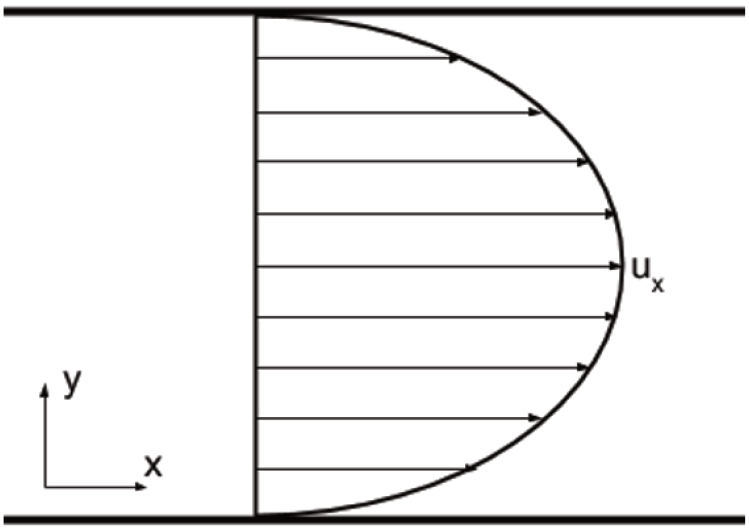


Figure 8 – Schematic representation of the Poiseuille flow.

Consider the NSE (2.9). If in stationary state and for a velocity $\vec{u} = \vec{u}_x$, the NSE reduces to:

$$\frac{\partial^2 u_x}{\partial y^2} = \frac{\partial p}{\partial x} \tag{5.4}$$

A force term can also be present in the direction of the flow as $\vec{F} = \vec{F}_x$. This should be a body-force acting equally in the whole domain.

$$\frac{\partial^2 u_x}{\partial y^2} = \frac{\partial p}{\partial x} - F_x \tag{5.5}$$

This equation can be solved analytically for $u_x(y)$ as:

$$u_x(y) = \frac{1}{2\mu}\left(\frac{\partial p}{\partial x} - F_x\right)\left[y^2 - \left(\frac{H}{2}\right)^2\right] \tag{5.6}$$

The pressure gradient can be approximated as a linear pressure drop, yielding:

$$\frac{\partial p}{\partial x} = -\frac{\Delta p}{L} \tag{5.7}$$

The values H and L correspond, respectively, to the size of the y and x dimensions of the computational domain. The simplified solution reads:

$$u_x(y) = \frac{1}{2\mu}\left(\frac{\Delta p}{L} + F_x\right)\left[\left(\frac{H}{2}\right)^2 - y^2\right] \tag{5.8}$$

Furthermore, a no-slip condition is adopted at the top and at bottom walls, which implies zero velocity in those nodes. The BB method (Section 4.1.1) was used in these walls.

Different boundary conditions were tested for the fluid nodes in the inlet and outlet. The first tested methodology (a) was the periodic domain with pressure drop, as presented in Section 4.1.4. The pressure drop was converted to a density difference using Equation 4.12.

The second methodology (b) was the definition of pressure in the inlet and outlet using the NEBB scheme presented in Section 4.1.6. Again, the pressure drop was converted in a density difference using Equation 4.12, and the densities were applied to each side. The reference outlet density was set to unity, and the inlet density was calculated by subtracting the density difference.

This pressure drop is also linked to the maximum velocity observed in the system, given by:

$$u_{\max} = \frac{1}{2\mu}\left(\frac{\Delta p}{L} + F_x\right)\left(\frac{H}{2}\right)^2 \tag{5.9}$$

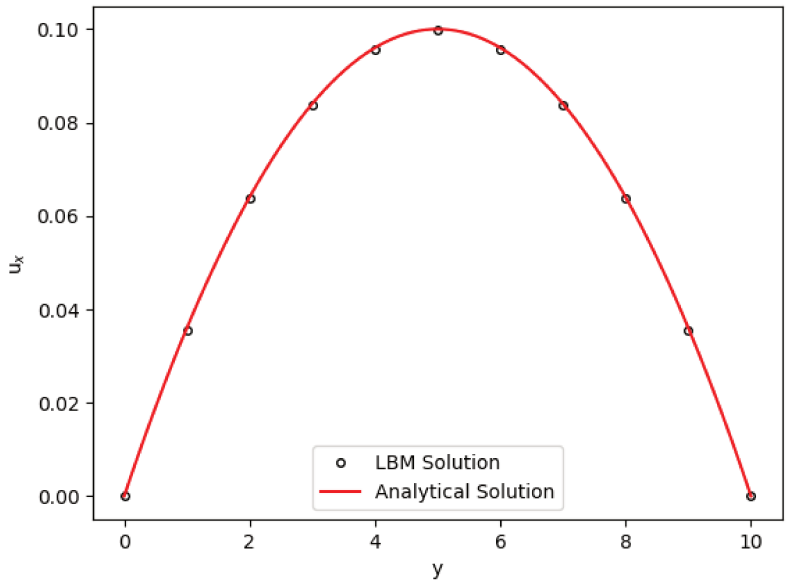
As the objective was to obtain a velocity profile by simulating a flow with the LBM, the chosen variable was the maximum velocity $u_{\max} = 0.1$ given by Equation 5.9. The velocity was defined, and the pressure drop was then calculated from the referred equation.

5.3.1 Results

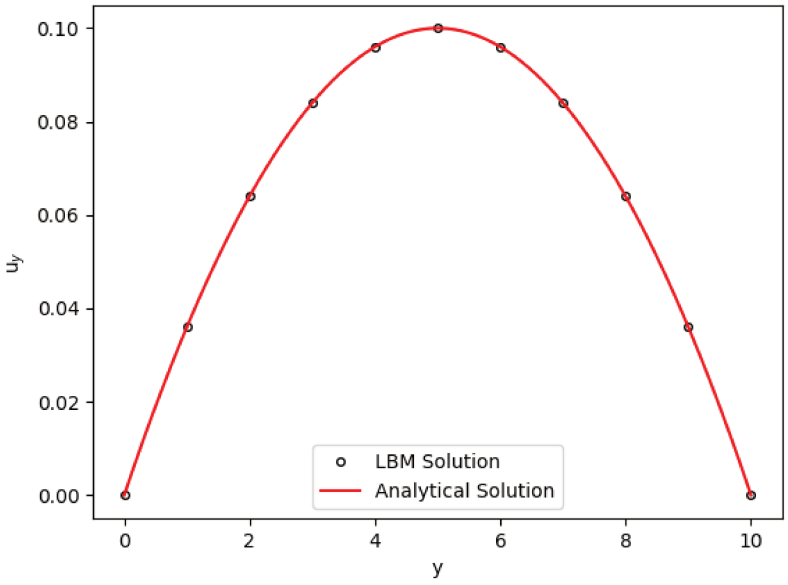
Both methods presented good concordance with the analytical solution for $\tau = 1$, as demonstrated in Figure 9. The NEBB method was chosen to be implemented for

more complex geometries due to being easier to implement in irregular grids, as porous media. Furthermore, pressure drop also proves to be harder to calculate in such systems.

Additional images are presented in Figure 10, such as the velocity field and the flow velocity representation for the NEBB boundary conditions. The errors found for the NEBB approach were AARD $= 0.0048$, and RMSD $= 0.00028$. The results shown were obtained for a square 11x11 grid containing 11 points in each direction. Better precision would be expected as the mesh is refined and more points are included. Nonetheless, the chosen methods proved to be good enough to simulate the Poiseuille profile with great accuracy, even with such small number of points.

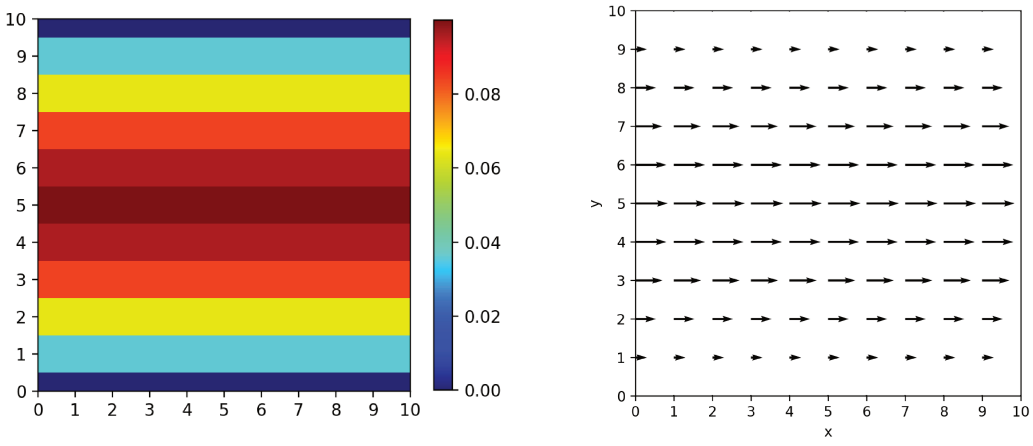


(a) Periodic domain (virtual nodes) with pressure drop approach.



(b) NEBB boundary conditions.

Figure 9 – Velocity profiles obtained for the LBM method.



(a) (b)

Figure 10 – Results obtained for the Poiseuille flow: (a) velocity; (b) velocity field.

5.4 Couette Flow

Another well-established flow is the Couette flow, which is also bidimensional, occurring between two infinite plates. In this case, the motion is generated by the relative movement of one plate in relation to another with a velocity u_p, any pressure difference is absent. It is characterized by a linear velocity profile, as shown in Figure 11.

Starting from the NSE, the same assumptions used to calculate the Poiseuille flow holds true, except for the absence of a pressure gradient. Therefore:

$$\frac{\partial^2 u_x}{\partial y^2} = \frac{\partial p}{\partial x} \tag{5.10}$$

The solution yields:

$$u_x(y) = u_p \frac{y}{H} \tag{5.11}$$

where H is the size of the computational domain in the y axis. A no-slip condition is assumed in both walls, which means the fluid does not have velocity in these points. Nonetheless, in the moving wall (top), that condition ensures the velocity of the fluid is the same as the wall velocity u_p. In the static wall (bottom) the BB method is used.

The variable chosen as a parameter was the velocity of the top wall, set to $u_p = u_{\max} = 0.1$ using the NEBB scheme (Section 4.1.6) to establish the velocity.

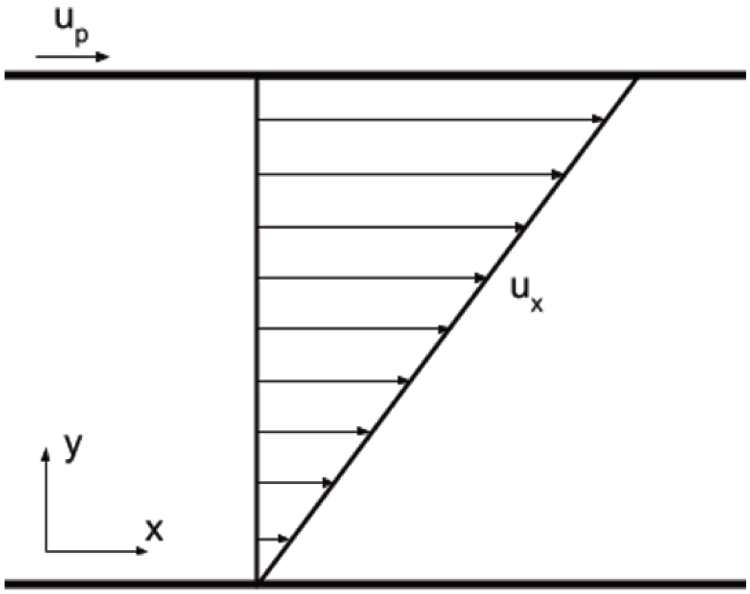


Figure 11 – Schematic representation of the Couette flow.

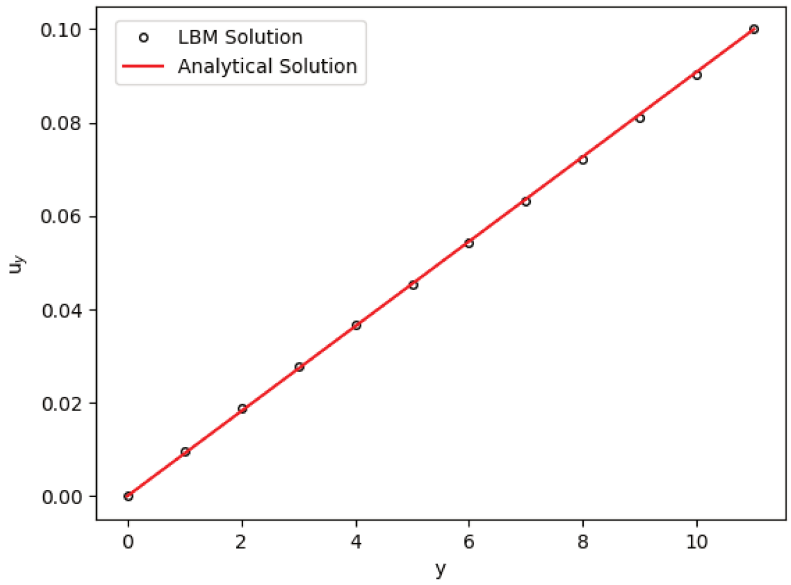
Two conditions were used to set the boundary conditions in the fluid nodes. Simulations were performed (a) using periodic domains with virtual nodes at the extremes (inlet and outlet), as described in Section 4.1.3. The other approach was (b) using the NEBB to set the same density on both laterals of the computational domain.

5.4.1 Results

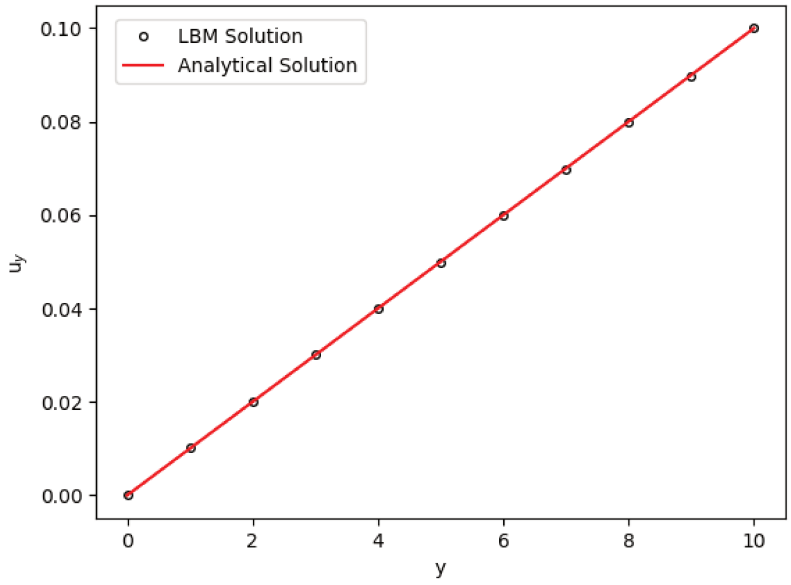
As seen in Figure 12, both approaches presented good results. As the NEBB boundary conditions held good results. This method will be used in the other simulations due to the situations formally discussed in previous sections, such as facility of implementation for complex geometries and versatility.

The velocity field and flux are shown in Figure 13 for better visualization. The errors were found to be AARD = 0.0030, and RMSD = 0.00034. The results shown were obtained for a square 11x11 grid containing 11 points in each direction. The same conclusions drawn about the Poiseuille flow apply here.

All results are considered satisfactory, and greater accuracy could be achieved by

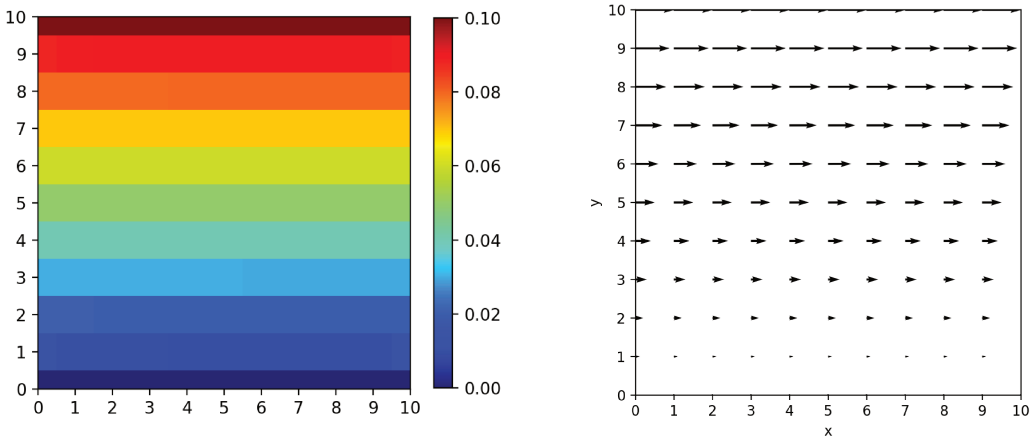


(a) Periodic domain (virtual nodes) approach.



(b) NEBB boundary conditions.

Figure 12 – Velocity profiles obtained for the Couette flow.



(a) (b)

Figure 13 – Results obtained for the Poiseuille flow: (a) velocity; (b) velocity field.

using more refined grids. Nonetheless, the methods proved to be efficient in simulating the Couette flow. Therefore, it is concluded that LBM is adequate to simulate Couette and Poiseuille flows.

5.4.2 Adequate Boundary Conditions

The importance of choosing adequate boundary conditions can be seen in Figure 14, which shows errors obtained when inadequate boundary conditions were applied in the nodes of the two top corners. The error in these two boundary nodes resulted in errors for the whole system, generating wrong results.

These errors in the corners tend to become less important as the grid is refined, and for bigger grids (100x100 points) they are barely noticeable. Nonetheless, this result illustrates the necessity of choosing carefully the adequate boundary conditions to avoid unnecessary errors.

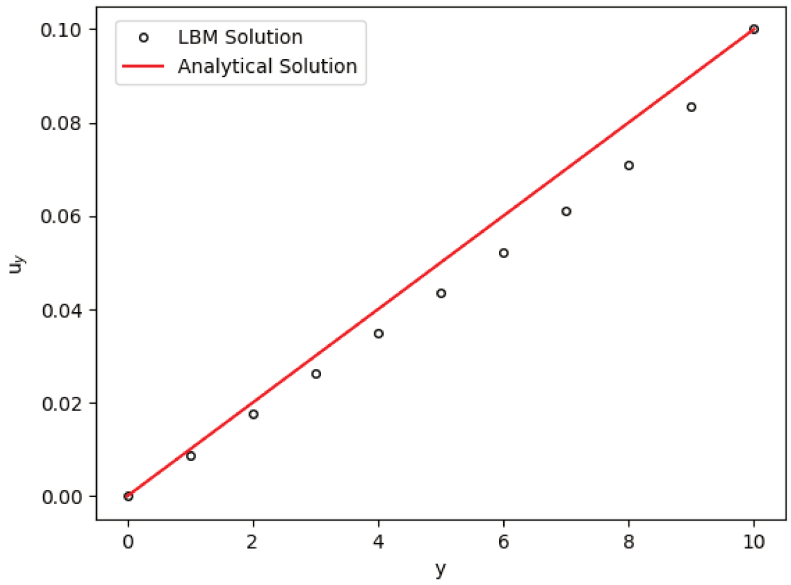


Figure 14 – Results obtained by applying inadequate boundary conditions in the top corners.

5.5 Combined Poiseuille and Couette Flows

Another possible flow is the mixture of the Poiseuille and Couette flows, that is, a pressure driven flow between two non-stationary infinite plates, meaning relative movement in relation to each other exist. The result is a velocity profile that falls in between the Poiseuille and Couette flows and has analytical solution given by Equation 5.12:

$$u_x(y) = \frac{1}{2\mu}\left(\frac{\Delta p}{L} + F_x\right)\left[\left(\frac{H}{2}\right)^2 - y^2\right] + u_p\frac{y}{H} \tag{5.12}$$

The conditions are those already explained to the Poiseulle and Couette flows, but applied within the same domain to define a pressure drop (laterals) and a wall velocity (top) with NEBB boundary conditions (Section 4.1.6). The BB method was used in the bottom wall.

The values used were a pressure drop equivalent to that tested on the Poiseuille flow ($u_{\max,\text{pois}} = 0.1$), and the same velocity to the plate previously used for the Couette

flow ($u_p = 0.1$).

The results obtained (Figure 15) show good agreement with the expected behavior. It provides further proof of the suitability of the LBM when using different boundary conditions for the computational domain.

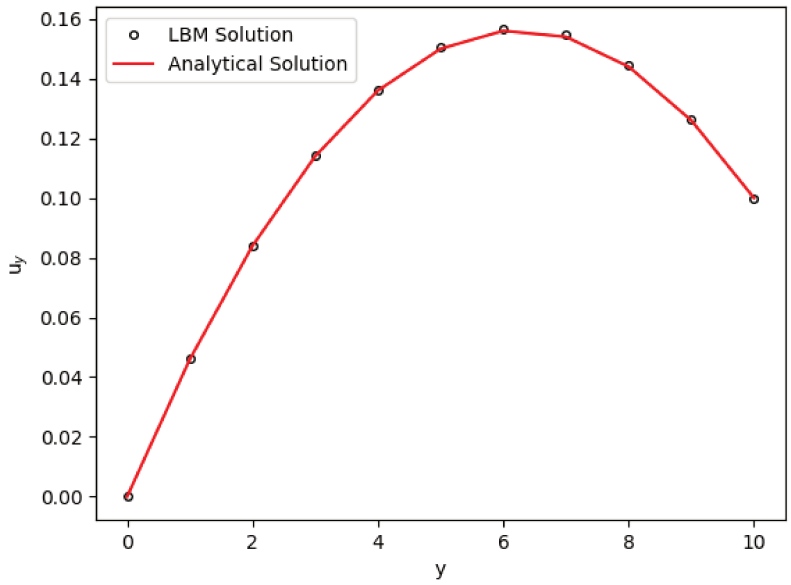


Figure 15 – Profile for the combined Poiseuille and Couette flows.

5.6 Relaxation Constant Dependence

As stated in Section 4, the use of the BGK collision operator introduces a non-physical effect characterized by a viscosity dependent result. This occurs because the viscosity is calculated from the relaxation constant τ by Equation 4.10, which is an input for the LBM when using the BGK as the collision operator. As the value for τ is not fixed, the resulting viscosity also varies. The LBM results turn to be clearly different for each value of τ, as seen for the Poiseuille flow in Figure 16.

The solution becomes very discrepant when the values of τ are distant from unity. Values of τ less than or equal to 0.5 result in instability and, thus, cannot be used. This result is related to the second-order discretization described in Section 4.2,

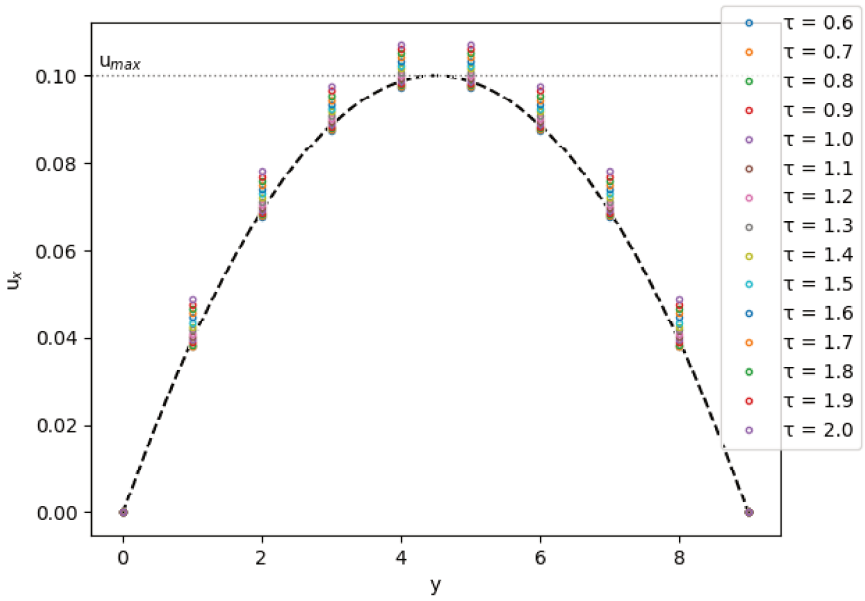


Figure 16 – Variation observed in the velocity profile as the value of τ changes. The black line represents the analytical solution corresponding to the situation.

which results in the factor $\left(\tau - \frac{\Delta t}{2}\right)$, which cannot be zero or negative, appearing in LBM equations. The viscosity, as given by Equation 4.10, also presents this term. As the physical meaning of viscosity determines it to be positive, if Δt is set to unity then the value of τ has to be greater than 0.5 to guarantee such a condition is obeyed.

Only one specific value for τ yields the exact solution, but the profiles are satisfactory in a range of values around unity. Changes in the velocity profile for the Couette flow (results omitted) were not observed with different values of the relaxation constant. Therefore, $\tau = 1$ was adopted for the other simulations performed in this work.

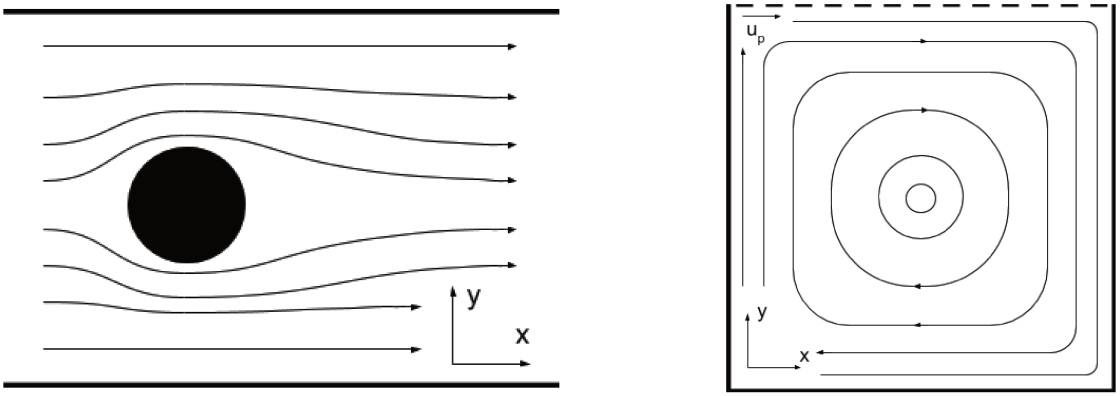
5.7 Other Geometries

Following the validation of the LBM method for Poiseulle and Couette flows, other more complex geometries were also simulated. The simulations cannot be compared to analytical solutions but can be qualitatively analyzed and compared to other CFD simulations and experimental results. The primary objective was to produce re-

sults comparable to those of (HO *et al.*, 2009) and (PORTINARI, 2015), observing the formation of vortices as seen in those works.

The first of these geometries was the Poiseulle Flow around an infinite cylinder (Figure 17a), which has no analytical solution. It is the same as the Poiseuille flow, but a round obstacle is placed within the computational domain, causing the flow to go around. The conditions are the same as the Poiseuille flow described in Section 5.3, and the condition applied to the cylinder was a common bounce-back scheme.

The second geometry analyzed was the so called Lid-Driven Cavity (LDC) flow (Figure 17b). It is characterized as the motion of a fluid within a cavity consisting of three solid resting walls. The fourth wall is moving with velocity u_p in relation to the others, and no normal velocities to the movement exist. The usual bounce-back scheme was applied to the resting walls and the Zou-He method was applied to the top wall to define the velocity in the same manner as done to the Couette flow in Section 5.4.



(a) Poiseuille flow around an infinite cylinder.

(b) Lid-Driven Cavity flow.

Figure 17 – Schemes of the geometries used in the simulations.

The results were compared to CFD simulations, which show the same behavior. The CFD simulations were performed by the Ms. Nadine Zandoná Rafagnim using the open software OpenFOAM. The solver used was the transient icoFoam for the variables velocity and pressure. The necessary input are the boundary conditions, the initial values, the kinematic viscosity, and the geometric grid.

The grids used were the same as the grids implemented in the LBM simulations. The number of nodes was not the same, but the relationship between all distances in

the system was. The boundary conditions were calculated from the unitary lattice units as presented in Section 4.6, and the converted values were implemented in the CFD simulations.

The results were obtained after approximately 50 s of simulated time, with the convergence criteria $\varepsilon = 10^{-6}$ used to infer the stationary state. Despite the numerical values not being compared, the results allow for a qualitative evaluation of the LBM.

5.8 Poiseuille Flow Around Infinite Cylinder

This flow is a pressure-driven flow between two parallel infinite plates analogous to the Poiseuille flow, but with a infinite cylindrical object placed obstructing the channel. A more refine 200x100 grid was used in this case to better represent the results. More nodes are simulated in the direction of the flow to allow for a full development of the flow after the cylinder.

The Reynolds number Re in this case was calculated as:

$$\mathrm{Re} = \frac{uD}{\nu} \tag{5.13}$$

where D is a characteristic length given by the diameter of the cylinder; u is the maximum velocity observed in a point where the effects of the cylinder are minimized (full developed flow or velocity profile before affected by the cylinder); and ν is the kinematic viscosity calculated from the relaxation constant τ as indicated in Equation 4.10.

It is also possible to observe the influence of Re by increasing the pressure difference between the inlet and outlet while maintaining the other parameters constant.

Figure 18 shows the laminar flow ($\mathrm{Re} \approx 5$) around the surface of the cylindrical object. The low pressure drop leads to a solution with no vortices forming.

Higher pressure differences resulted in the formation of vortices after the cylinder, as shown in Figure 19 ($\mathrm{Re} \approx 70$). Comparing both scenarios, the increase in the pressure drop by a factor of 10 resulted in the maximum velocity increasing from approximately 0.04 to 0.3 and Re from 5 to 70.

Figure 20 shows the effect of increasing the pressure drop and Reynolds number.

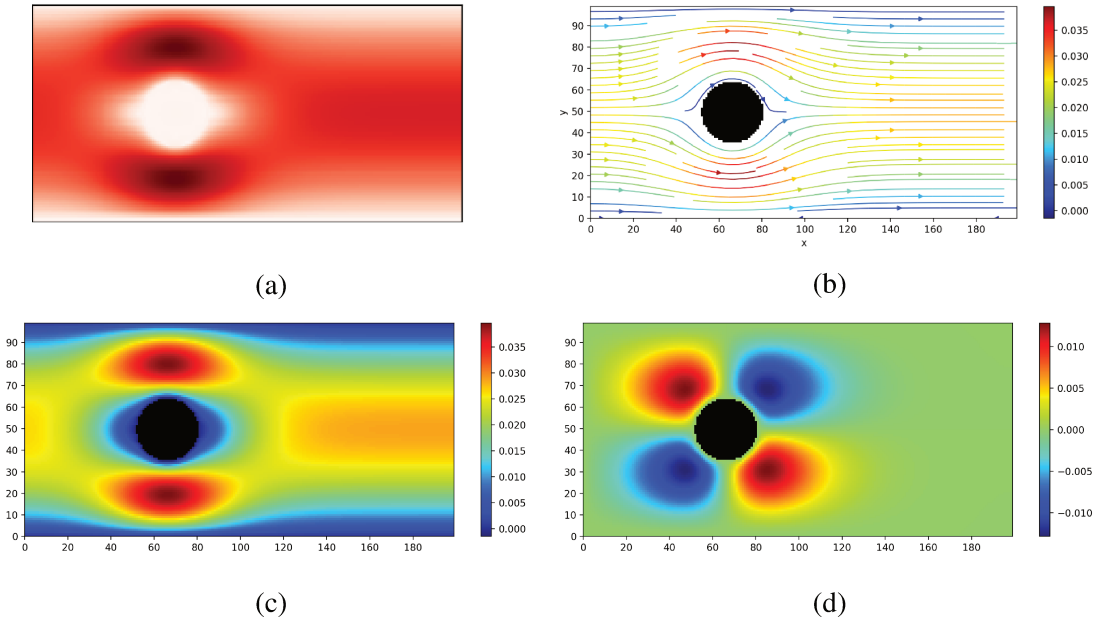


Figure 18 – Results for the Poiseulle laminar flow around an infinite cylinder (Re ≈ 5): (a) absolute velocity; (b) flow lines; (c) horizontal component of velocity; (d) vertical component of velocity (d).

Figure 21 shows the comparison between the LBM simulation results and CFD. The steps of the CFD simulations were set to $\Delta x = 2$ mm. Viscosity and pressure were calculated as specified in Section 4.6. Outlet pressure was set to 0. The inlet pressure was obtained from the pressure drop in LBM. The initial velocity was set to 0. The boundaries were specified as to not allow gradients at the inlet and outlet, and no-slip ($u = 0$) conditions at the walls.

LBM was able to simulate in great agreement with CFD, behaving as expected. Furthermore, Champmartin and Ambari (2007) show experimental and numeric profile for various flow regimes. The results achieved using LBM are in accordance to the experimental results from Van Dyke (1982).

Comparing LBM solutions to the expected results, flow around an infinite cylinder was shown to be satisfactory. The same vortices were observed for the LBM and the CFD in higher velocities. The difference observed for the lower velocities are due to differences in the pressure applied. There is no point in comparing velocity profiles in this case, since there is no analytical solution. Therefore, by also looking at the comparisons

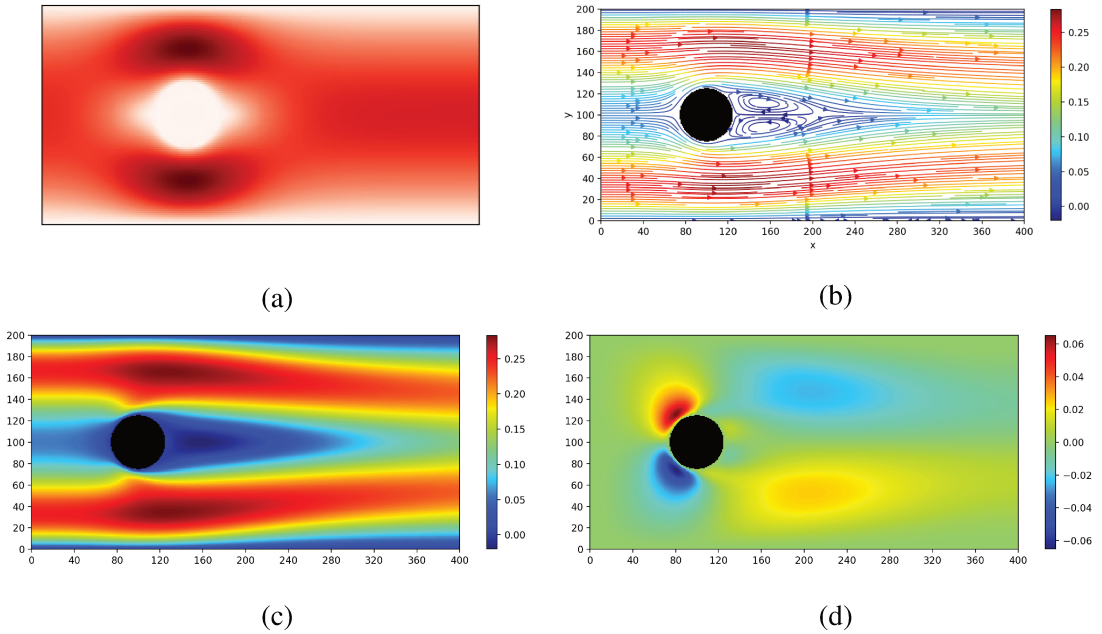


Figure 19 – Results for the Poiseulle flow around an infinite cylinder for a pressure drop with magnitude increased 10 times (Re ≈ 70): (a) absolute velocity; (b) flow lines; (c) horizontal component of velocity; (d) vertical component of velocity (d).

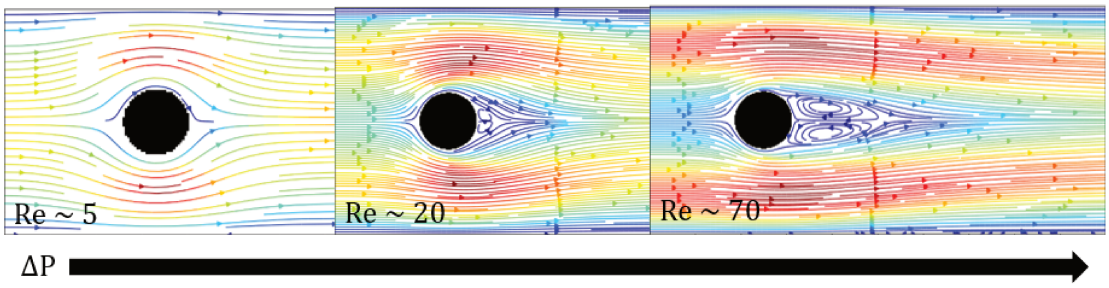


Figure 20 – Effect of increasing the pressure drop and Reynolds number.

towards CFD methods, LBM was considered able to reproduce the flow.

One notable thing was the effect of some terms into the equilibrium distribution function. Simplifications in Equation 4.13, introduced by removing the last two terms (Equation 5.14), have not affected considerably Poiseuille and Couette flows (results not shown). However, in the cylinder case, the vortices were not obtained unless those

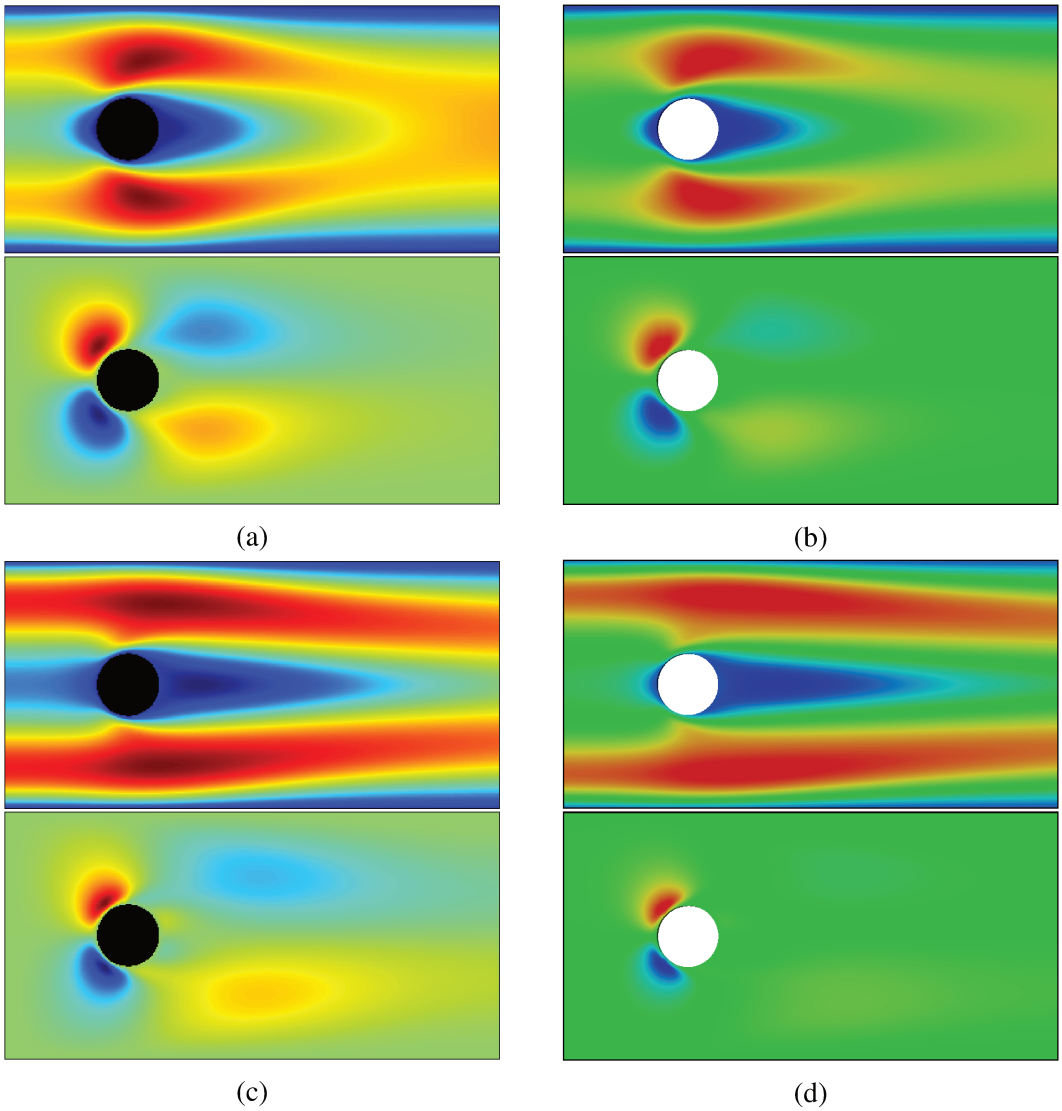


Figure 21 – Comparison between results obtained using the LBM and CFD: (a) LBM with low pressure drop (Re $\approx$ 5); (b) CFD with low pressure drop; (c) LBM with 10 times higher pressure drop (Re $\approx$ 20); (d) CFD with 10 times higher pressure drop. In all cases the images show the velocities in the x axis (top) and in the y axis (bottom).

terms were considered.

$$f_i^{\text{eq}} = w_i \rho \left(1 + \frac{\vec{u} \cdot \vec{c}_i}{c_s^2}\right) \tag{5.14}$$

Therefore, the removed terms can be considered as convective terms that lead

to the advection observed in the ADEs, and are a condition to vortices formation. So it becomes clear why the impact is minimized in laminar flows and becomes significant as these effects become relevant. Nonetheless, for better accuracy the full equilibrium distribution given by Equation 4.13 was preferred for all flows in this work.

5.9 Lid-Driven Cavity

The lid-driven cavity is a geometry in which flow is not induced by pressure difference. As in the case of Couette flow, the moving wall in one of the boundaries of the domain is what causes the movement. As all other boundaries are solid walls, the consequence is a recirculation within the cavity. The formation of a central vortex can be seen, with minor vortices depending on the velocity of the moving wall, which affects Re. Figure 22 shows the results obtained for a Reynolds number $\mathrm{Re} \approx 25$.

Qualitatively, the results in Figure 22 were also in agreement to the expected behavior. It is also a common method to validate the LBM and prove its functionality (HO *et al.*, 2009; PENG, 2011; PORTINARI, 2015). The velocity profile matches those found in the literature for this geometry (PERUMAL; DASS, 2011; SHANKAR; DESHPANDE, 2000). The results were also compared to CFD simulations, as presented in Figure 23.

The CFD simulation steps were set to $\Delta x = 1$ mm. Viscosity was calculated as indicated in Section 4.6. The velocity was also calaculated and specified as a fixed value at the top. The no-slip condition ($u = 0$) was used at the walls. Pressure was set to 0 within the whole computational domain, and pressure gradients were set to 0 at the boundaries.

The results imply the LBM is suitable to represent the flow in this geometry. These results corroborate and validate both the method and the code implemented to this moment within this project.

5.9.1 Lid-Driven Cavity With Infinite Cylinder

Other simulated situation was the union of the two geometries presented, consisting of a cavity containing a infinite cylinder. The LBM results were compared to

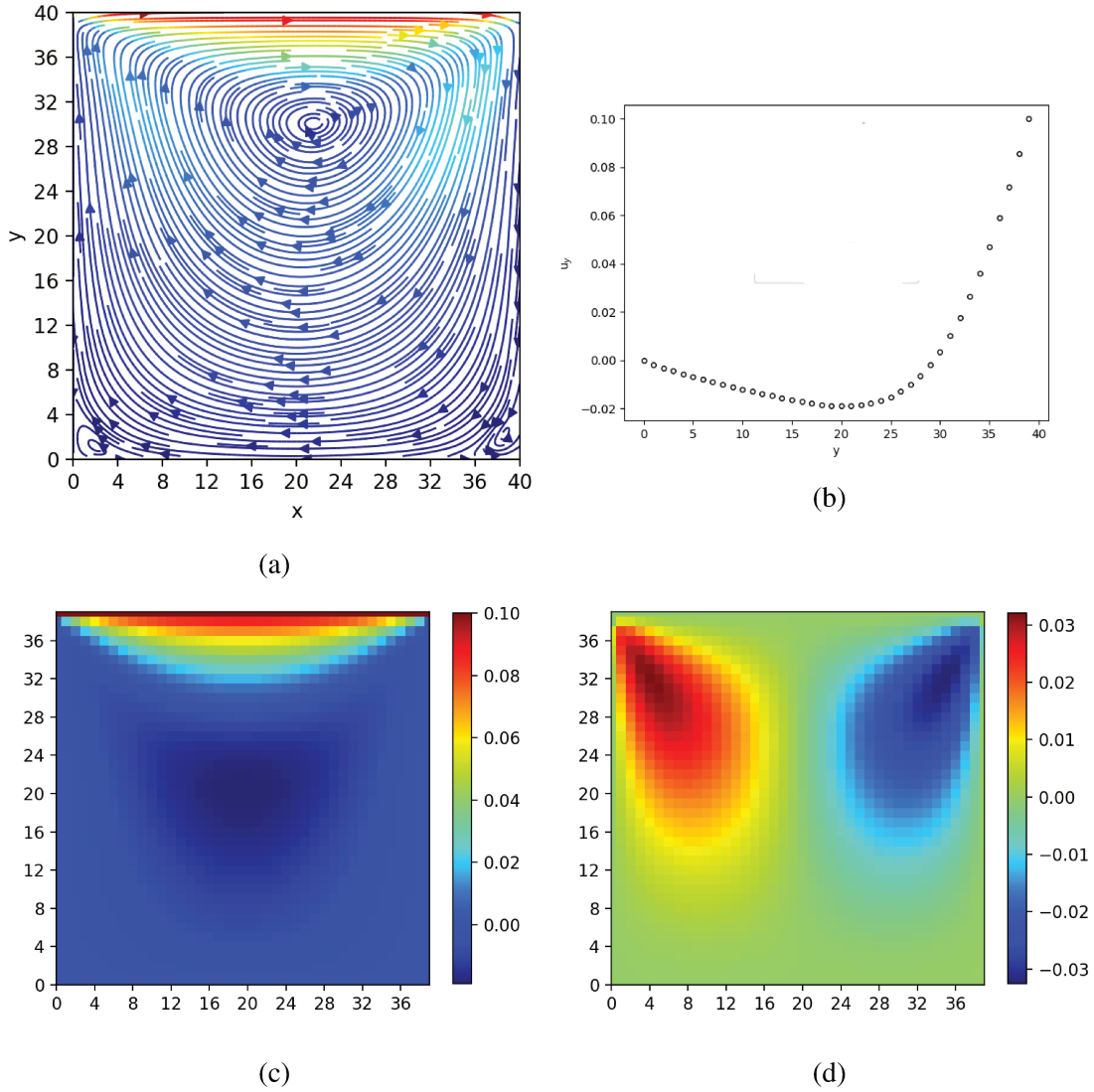
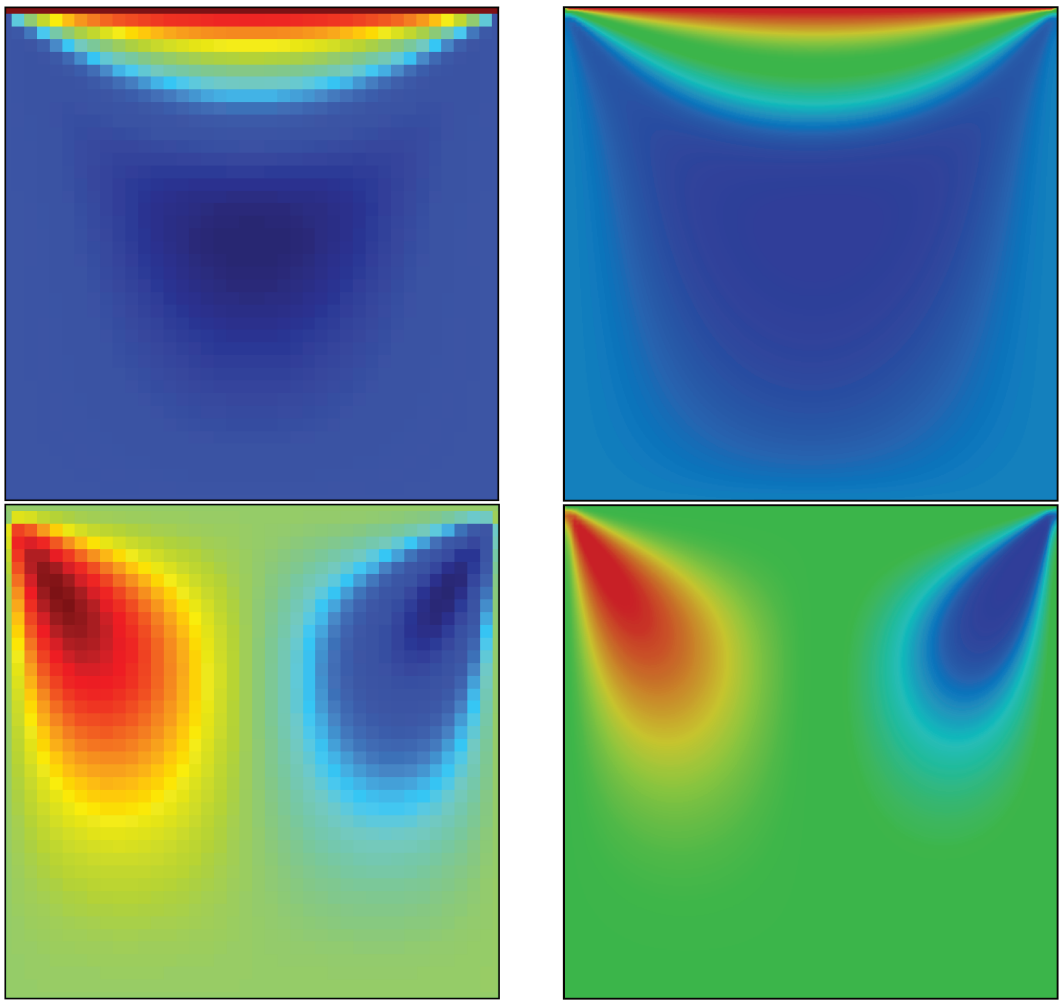


Figure 22 – Results for the Lid-Driven Cavity flow for a top moving lid with the maximum velocity observed in the images ($u = 0.1$ and Re ≈ 25): (a) flow lines; (b) velocity profile in the middle; (c) horizontal component of velocity; (d) vertical component of velocity (d).

CFD simulations (24), with the same conditions used in Section 5.9 applied. It is noticeable the laminar flow around the cylinder and the vortices in the bottom corner. Overall, the results were satisfactory, indicating the versatility of LBM to simulate traditional CFD problems.



(a) (b)

Figure 23 – Comparison between the results obtained for the Lid-Driven Cavity geometry using: (a) LBM; (b) CFD. The graphs show the velocities in the x axis (top) and y axis (bottom).

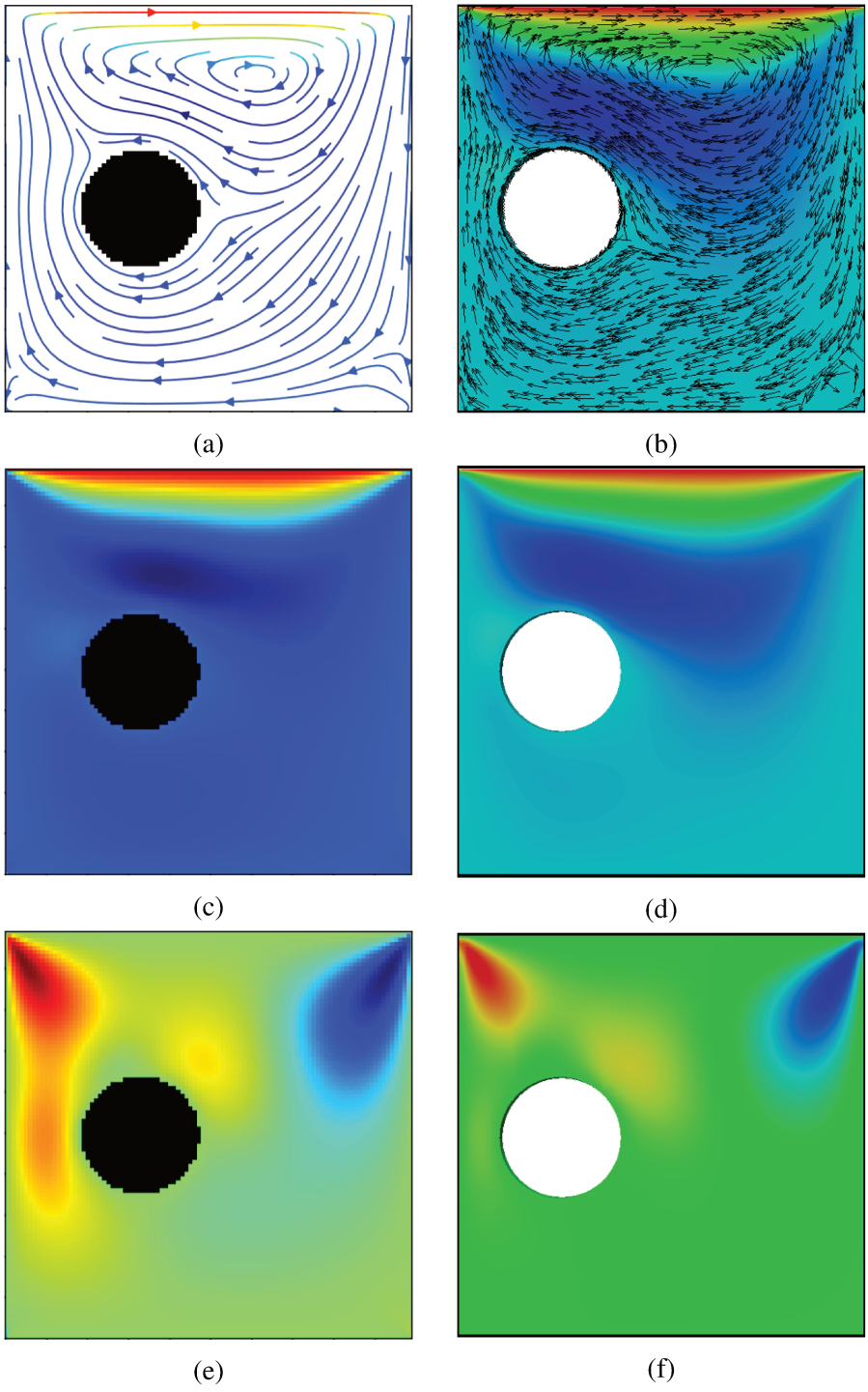


Figure 24 – Results for the Lid-Driven Cavity flow with a cylinder inserted in the cavity: simulated flow with (a) LBM and (b) CFD; velocity in the x axis for (c) LBM and (d) CFD simulations; velocity in the x axis for (e) LBM and (f) CFD simulations.

5.10 External Force Field

The LBE with the BGK collision operator and forces can be implemented as seen in Equations 4.9, 4.127, and 4.69:

$$f_i(\vec{x}+\vec{c}_i\Delta t, t+\Delta t) = \left(1-\frac{\Delta t}{\tau}\right) f_i(\vec{x},t) + \frac{\Delta t}{\tau} f_i^{\mathrm{eq}}(\vec{x},t) + S_i\Delta t$$

$$S_i = \left(1-\frac{\Delta t}{2\tau'}\right) F_i$$

$$F_i = w_i\left(\frac{\vec{c}_i}{c_s^2} + \frac{(\vec{c}_i\cdot\vec{u})\cdot\vec{c}_i}{c_s^4} - \frac{\vec{u}}{c_s^2}\right)\cdot\vec{F}$$

Using Guo-force scheme as presented in Sections 4.5, the equilibrium distribution function and collision operator remain unaltered, and the velocity is redefined as in Equation 4.127:

$$\rho\vec{u} = \sum_i (f_i\vec{c}_i) + \vec{F}\frac{\Delta t}{2}$$

External force fields can be implemented as body-forces that act with the same intensity in the whole domain, that is, when the value of $\vec{F}$ is the same in all nodes. This is the case for force driven flows and gravity. Other force fields that act locally exist. The value of $\vec{F}$ in this case depends on the position of each node, as can be the case for electric and magnetic fields, and adsorption forces.

The pressure gradient in the Poiseuille flow can be translated into a corresponding force calculated as indicated by Equation 5.8. As the results show (Figure 25), the velocity profile match the analytical solution, and the error is small. Compared to the results previously seen in Section 5.3, the body-force approach presents itself as a viable alternative.

This approach also eliminates the variation of density, as shown in Figure 26. However, it presents limitations, since the implementation requires more modifications in the code. It is also more difficult to implement in multicomponent flows, and some efficiency is lost, since the force term has to be calculated in each iteration for all nodes, instead of conditions calculated only on boundaries.

Another important example is gravity, which is usually neglected but can influence the results in some specific problems, and hence, ought to be implemented. How-

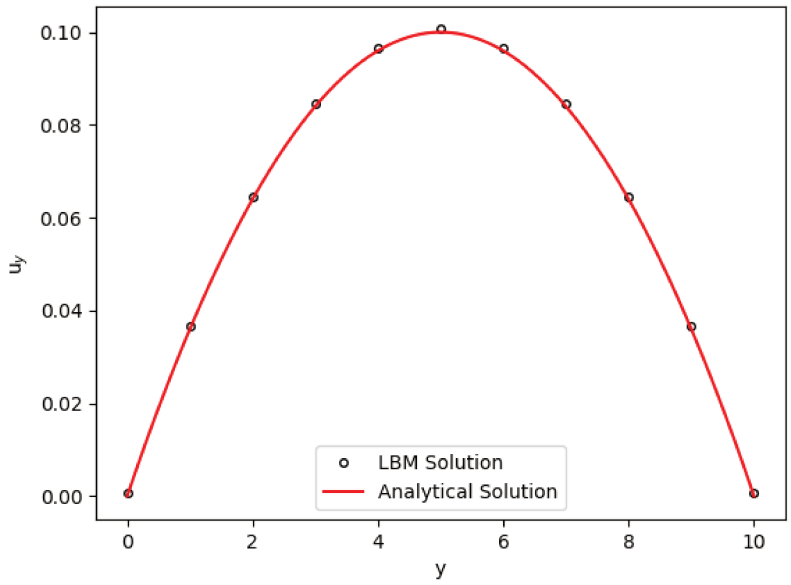
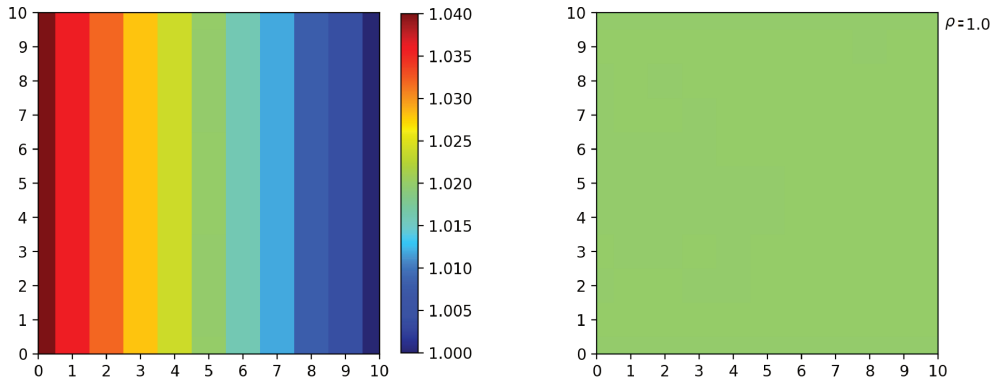


Figure 25 – Velocity profile of the Poiseuille flow obtained using the body-force approach.

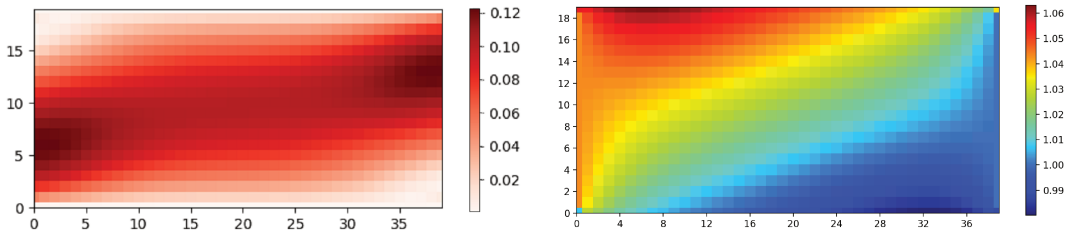


(a) (b)

Figure 26 – Density profiles obtained for the Poiseuille flow: (a) pressure difference approach; (b) body-force approach.

ever, the directions of the force and the flow can be distinct in this case, and no pressure drop is associated to the force field.

The body-force approach can be implemented in any direction, not only in the direction of the flow. Figure 27 shows a pressure driven flow by a pressure difference between the inlet and outlet in the horizontal direction submitted to a force in the vertical direction.



(a) (b)

Figure 27 – Pressure driven flow between infinite plates subjected to a perpendicular force field: (a) velocity; (b) density.

This represents a generic force field, since gravity usually has a much smaller effect. The force in the case presented, despite being small when compared to the pressure difference, was exaggerated so the results would be visible. In reality, gravitational effects are only visible for systems with a wide difference in height, in the scale of kilometers.

The effects of the force in the density profile and velocity profile are clear, attracting mass to the side the force is directed towards as the flow develops within a channel.

6 Porous Media Simulations

There are many works that acknowledge the importance of researching LBM in porous media since the applications are vast (PAN; HILPERT; MILLER, 2004; BOEK; VENTUROLI, 2010). The combination of multiphase and multicomponent flows with porous media has, thus, drawn some attention. Shouguang *et al.* (2019) studied phase transition in such medium, and Luo and Xu (2019) simulated heat and mass transfer in porous media, important steps in the direction of simulating real problems, despite the less realistic geometry.

There are several approaches for applying LBM to porous media, but the pore-scale simulations apply the already presented concepts regarding external forces and multiphasic or multicomponent flows. It is also possible to apply a generalized lattice Boltzmann equation for porous media considering the macroscopic properties, as performed by (QIU *et al.*, 2019), and Peng *et al.* (2018) in their study of gas absorption on coal reservoirs.

6.1 Generation of Meshes

To generate the random meshes used as a base for these simulations, the Ising model (FERRENBERG; XU; LANDAU, 2018) was used. The meshes generated using the method are random and resemble the pores found within rocks, as seen in Computed Tomography (CT) and microscopy images found in the literature, as in the works of Arab *et al.* (2009) and Louis, Baud and Wong (2007).

There are other algorithms used to generate computational porous media, but no other works were found using a combination of the Ising model and LBM. The option for a simulated porous media approach is the capacity of performing the whole process computationally, with no need for experiments.

The Ising model is based upon interactions between binary variables denominated spins, which are analogous to the magnetic spin, assuming a positive or negative value. The variables are arranged in a lattice and interact with their neighbors. The re-

sulting energy from this interaction is higher if the spins are different and lower if the spins have the same value. The total energy of such a system can be given as the sum of all binary interactions.

As the system tends to minimize the total energy, the spins tend to align. However, heat tends to disrupt this tendency and, as the temperature increases, the collective effects are gradually lost owing to the introduction of aleatory effects. This leads to the possibility of different phases appearing. If the temperature is further increased beyond a certain critical temperature (T_c), the spins are randomly distributed and no phase separation is seen.

One efficient approach to implement the Ising model is using Monte Carlo (MC) simulations. This method consist of changing the spin of any aleatory particle within the lattice and evaluating the resulting total energy for the system. If the resulting energy of the system is minimized with this modification, then the new configuration is maintained. Otherwise, if the energy increases, the alteration is maintained or discarded according to a pre-established probability. Since the chance of an alteration that increases the total energy be accepted is lower than an alteration that results in a lower energy, so that the system tends to a single-spin state (for small lattices and next to zero temperature), or to form regions with the same spin (phases) for temperature bellow T_c.

The Ising model is an important problem in statistical mechanics and can be used to represent phase separation, or magnetism in several materials (FERRENBERG; XU; LANDAU, 2018). The model can be implemented in any number of dimensions but, in this work, the bidimensional model was used since LBM codes were also implemented using bidimensional velocity sets. As stated, depending on the input values for the temperature bellow T_c, different phases form.

Considering one of these phases as solid and the other as empty space (pores filled with liquid), the structure becomes an artificial representation of a porous media grid, through which it is possible to simulate the behavior of pressure-driven flows.

These computational meshes are promising, since when compared to real porous media images similar structures were observed. The presented channels, dead-ends, obstructions, and constrictions. Figure 28 show examples of these grids.

These resulting grids depend on the input parameters for the Ising model (Tem-

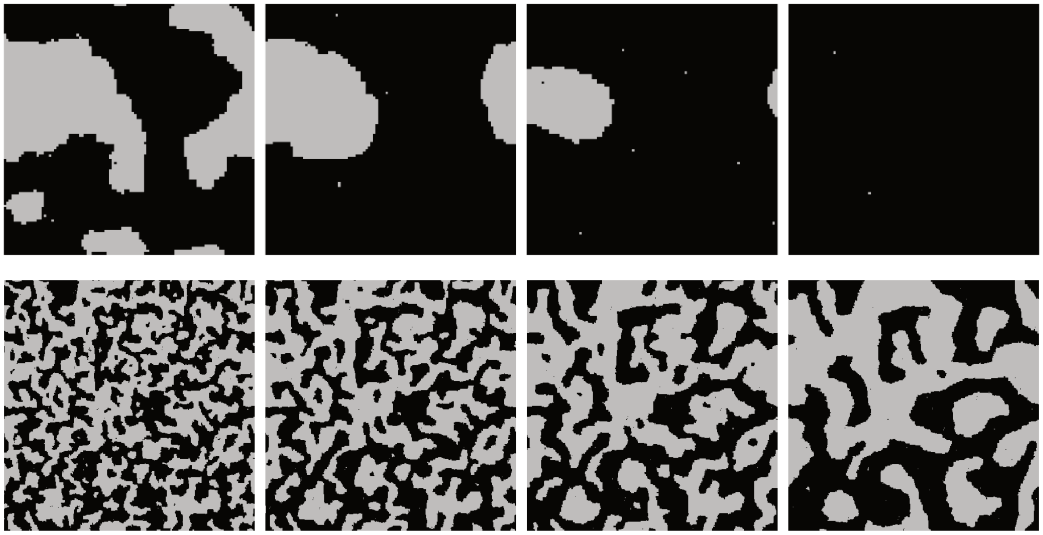


Figure 28 – Results for the Ising model with temperature set to 4 K. Meshes with 101x101 nodes (top) and 401x401 nodes (bottom). From left to right, the number of time steps elapsed in the simulations grow.

perature, size of the grid, and initial configuration), which results in interesting patterns. The meshes also change according to the number of time steps simulated, as the phases tend to separate. The best results were obtained before complete phase separation.

In some cases, the Ising model results in the formation of two contiguous phases with no channels or even the formation of a unique phase as seen in Figure 28. This happens for lower temperatures (next to 0 K), or small grids (100x100 points). High temperatures, above T_c do not separate phases and are unusable for the purposes of this work. Big grids (1000x1000 points) also do not present satisfactory results as artifacts appear on the grids.

The best results were obtained before complete phase separation, for intermediate grid sizes (around 400x400 points) and temperature a little below T_c (4 K).

6.2 LBM Simulations

The meshes obtained from the Ising model were submitted to LBM simulations (Figure 29) with one of the phases treated as solid, whereas the other was considered

fluid. The aim of these simulations was to emulate flow in porous media.

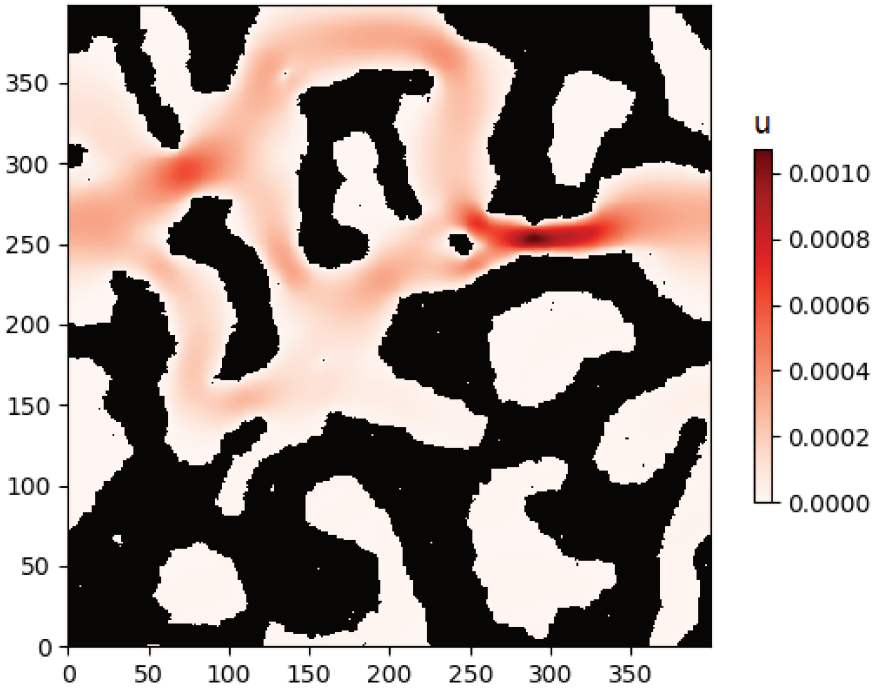


Figure 29 – Result of LBM simulation through porous media obtained using the Ising model. The grid contains 401x401 nodes, and the velocity is represented in red.

Only qualitative results could be achieved and interpreted, since the simulation results cannot be compared to analytical solutions, as in the case of Couette and Poiseuille flows. The results also cannot be compared to experimental results for this specific system, since the grid was generated for this specific purpose and is non-physical in nature.

The grid porosity can be calculated, and other microscopic properties can be obtained. These properties could be compared to real porous media that presents the same characteristics using universality laws, in an attempt to obtain equivalent experimental results. The relation between porosity, the geometry of the pores, and the macroscopic permeability could be explored. However, since this work focuses on LBM, a separate work focused on studying the universality laws that govern these rock structures and flow in porous media would be more adequate to access these effects.

As for the boundary conditions, the bounce-back (BB) method (Section 4.1.1) was used in the pore-walls. Non-equilibrium bounce-back (NEBB) boundary conditions (Section 4.1.6) were used to apply pressure, generating a pressure drop. The top and bottom ends of the computational domain were also treated as solid walls with the BB method when the original porous grid showed an opening in these regions.

As expected, the velocities are greater in places where constrictions are observed, and the profile resembles the profile found for Poiseuille flow inside the pores. The maximum velocity was much smaller when compared to the Poiseuille flow with the same pressure drop applied in the same size of the grid, as expected due to the constrictions and the smaller diameter of the channels.

The fluid was also observed to be almost stagnated in various points, especially inside the dead-ends, as the main flow occurs in the larger contiguous channels. Therefore, little interaction with the flow and little movement inside those cavities can be assumed, implying it is difficult to penetrate in those pores and extract anything from them. Nonetheless, these results depend on the porosity and geometry of the pores, since different meshes could be obtained with more channels and no dead-ends.

Flow inside a pore resembles the Poiseuille flow, but with curved and irregular walls. In fact, some equations that emulate the macroscopic behavior of fluids are based upon assumptions of flows through cylindrical pores.Therefore, despite being very different in geometry to those idealized flows, qualitatively it would be expected that those results still applied, as it was exactly the case.

6.3 Darcy's Law

For validation and comparison with macroscopic properties, it is also possible to analyze if the flow follows the Darcy's Law (Equation 6.1):

$$u = -\frac{K}{\mu}\Delta P \tag{6.1}$$

which establishes a linear relationship between the pressure drop applied ΔP and the apparent velocity u of the flow in a porous media with permeability K. μ is the dynamic viscosity of the fluid.

Thereafter, values of pressure and velocity can be plotted to verify the linearity between those two values and to verify if the linear regime established by Darcy's Law is observed. This regime is observed in laminar flows with low velocities, which is the case in the simulations performed.

Comparing different pressure drops and velocities, the relation between them showed to be linear (Figure 30), as expected according to Darcy's law (Equation 6.1) for a linear flow as is the present case.

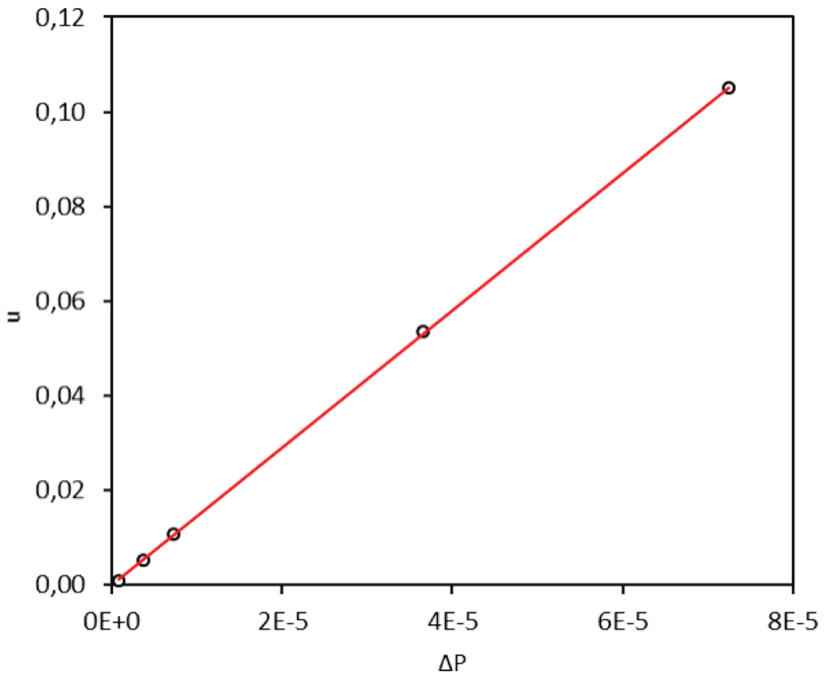


Figure 30 – Maximum flow velocity relation to the pressure difference applied to the system. Simulation results presented with linearization (in red).

The linear model fits the data very well with an error ($R^2 = 0.99996$). From this linear relation, it is possible to obtain a value to the permeability K of the grid, as in Equation 6.1. Such value could be compared to a measured value. However, as the meshes were created computationally, this does not add information in terms of drawing comparisons.

7 Multicomponent Flow

There are multiple approaches to model multiphase and multicomponent flows using the LBM (KRÜGER *et al.*, 2017; ZHANG *et al.*, 2019). One of the first methodologies to be developed was the color-gradient method proposed by Gunstensen *et al.* (1991). Despite being less frequently used, some works still use this method in suitable manners (FU *et al.*, 2018; BAKHSHIAN; HOSSEINI; SHOKRI, 2019). Nonetheless, the most common methods for multiphase and multicomponent flows are the pseudo-potential (PP) models introduced by Shan and Chen (1993), and the free-energy methods, first developed by Swift *et al.* (1996).

Despite the basic methodology being the same, modeling single-component multiphase flows is different than modeling flow with more than one component, but these cases can be integrated (KRÜGER *et al.*, 2017). The former is still much more common in terms of LBM simulations. Nonetheless, the principle applied in both cases is essentially the same. Forces are added to the LBM by adding a new discretized term to the equation. For the detailed steps, refer to the work of Krüger *et al.* (2017). This term modifies the collision step and may require a new equilibrium distribution function that accounts for its addition. Moreover, since this forcing term can alter the moments, the velocities are modified to include its effects.

The Shan and Chen (1993) model is based on the addition of microscopical interactions between the particles that constitute the fluid by implementing a pseudo-potential (PP), which is represented as a forcing term on the Lattice Boltzmann Equation (LBE). The work of Başağaoğlu and Succi (2010) shows an example of various distinct forces applied to LBM for representing different types of interactions based on a methodology close to that of Shan-Chen. These PPs based in the Shan-Chen model evolved with LBM, and recent studies still propose new boundary conditions and force terms to account for these interactions and generation of more accurate results in specific cases, as is the case of the recent study by Tao *et al.* (2019).

The easiest way to implement this method is to define different distribution functions f_i^α for each component or phase α. These functions are propagated independently

and related to each other by the PPs. Nonetheless, in the case of single-component multiphasic flows, a single distribution function may be used with an equation of state (EoS) that predicts phase separation (e.g.: the Peng-Robinson EoS) (KRÜGER *et al.*, 2017). The phase differs by a difference in density in this case. For multicomponent mixtures with low-density ratios, however, a function of the density associated with a relative concentration (order parameter) is preferred. In both cases, the separation occurs naturally due to the introduced PPs.

Another common methodology is the free-energy method (SWIFT *et al.*, 1996), which is based on the introduction of thermodynamically consistent free-energy functionals. This method also conserves local momentum, which does not occur in the Shan-Chen PP method. The free energies associated with the chemical potential of the species or phases in the fluid are used to construct functionals. Wen, Qiu and Shan (2019) also expanded the models to accommodate larger density ratios for the fluids using the definition of the chemical potentials. Properties such as surface tension and contact angle allow for modeling of interfaces within the fluid and the interactions between the fluid components and the walls (wetting properties). This approach differs from the PPs, because it considers macroscopic characteristics derived thermodynamically as input, in contrast to obtaining them from the LBM simulation. This became a popular approach owing to the thermodynamic consistency.

As previously stated, it is possible to simulate multiple phases with a single distribution function, as the cavitating flow implemented by Cai *et al.* (2018) demonstrates. The authors, however, concentrated on the study of single-component flows. In fact, as pointed by Akker (2018), most works published recently in the multiphase-multicomponent LBM focuses on single-component problems. These are simulated for one of the various applications that do not need detailed information on the compositions of each phase, constituted by a single component or by a particle representative of the mixture. The studies focusing on individual component concentrations are often treated as mass transfer problems, despite this distinction not being necessary, and have been also successfully implemented on a minor scale.

Particularly, various works focused on LBM studies for shale-gas simulation within shale reservoirs (YU *et al.*, 2017; PEREIRA, 2019; REN *et al.*, 2019), since LBM is robust for such applications and shale-gas extraction tends to be an important topic of

study in various countries. Pereira (2019) used a modified Shan-Chen PP to account for the presence of micropores of different scales present in the shale porous media. The application to oil reservoirs is also promising, as it is a topic no less important than the shale-gas studies for economic reasons.

Recent works have also been published for the diffusion of oxygen and the electrochemical reactions in proton exchange membrane (PEM) fuel cells (DENG *et al.*, 2019; HOU *et al.*, 2019) by exploring the distribution function of specific components, as well as drop coalescence (HOU *et al.*, 2018) within the same cells. Fang *et al.* (2019) also used LBM in pore-scale to study atomic layer deposition in porous electrodes. Guan and Novosselov (2019) applied the concept of internal forces to charged particles. Moreover, new amphiphilic PPs for the Shan-Chen model have also been proposed to simulate surfactants (MUKHERJEE; BERGHOUT; AKKER, 2019; WEI *et al.*, 2018; WEI *et al.*, 2019).

Despite the existence of various models for implementing internal forces in LBM, there is not a consensus for the adequate method, each presenting its own advantages and drawbacks. The Shan-Chen based methods are straightforward to implement and understand and allows for solubility calculations of the phases for the multicomponent cases. However, they are not derived from a thermodynamically consistent standpoint. They are, nonetheless, still preferred in various cases for its simplicity and can produce very accurate results if an adequate EoS (different from the usual assumed isothermal LBM EoS) is chosen. They are also preferred to represent forces resulting from microscopic interaction potentials.

The basic formulation of the SC method will be provided in the following section. As for the Free-Energy model, it will be explained in Section 7.2, albeit not studied in great detail.

7.1 Shan-Chen Pseudo-Potential Method

The Shan-Chen (SC) model for multicomponent fluids (SHAN; CHEN, 1993) is a microscopic model that considers the interaction between the molecules directly through a force term. These microscopic interactions are postulated as potentials generated by the molecules of the fluid that have the capacity to model homogeneous mixtures

as well as phase separation. In this approach, there is no need to provide macroscopic variables such as the surface tension or contact angle. Instead, the macroscopic behavior and consequent properties are generated from the microscopic interactions.

The force in the SC model is proportional to the local density of the components ρ_α. It was proposed to account for the interaction between a node and the neighboring nodes. The long range interactions are discarded in SC approach. Moreover, a term to account for the intensity of the interaction $G_{\alpha\beta}$ between two components α and β must be considered.

The SC force $\vec{F}_{SC}$ for the interaction between component α located in position $\vec{x}$ and component β in position $\vec{x}^*$ can, thus, be written as:

$$\vec{F}^{SC}_{\alpha\beta}(\vec{x},\vec{x}^*) = -\psi_\alpha(\vec{x})G_{\alpha\beta}(\vec{x},\vec{x}^*)\sum_i w_i\psi_\beta(\vec{x}^*_i)(\vec{x}-\vec{x}^*_i) \tag{7.1}$$

The direct link to the density is substituted by a pseudo-potential for the component α that is a function of the local density of the component α:

$$\psi_\alpha = \psi_\alpha(\rho_\alpha) \tag{7.2}$$

This is the most generic form for any two nodes in the defined positions. The total force that acts on a component α within the node at position $\vec{x}$ is, then, the sum of each SC force, for all positions $\vec{x}^*_j$ and all components k:

$$\vec{F}^{SC}_\alpha(\vec{x}) = \sum_k\sum_j \vec{F}^{SC}_{\alpha k}(\vec{x},\vec{x}^*_j) \tag{7.3}$$

Usually, the long-range interactions are not considered, and the interactions beyond the neighboring nodes can be considered. This assumption is supported by the fact that the microscopic interactions are usually very local and can be ignored beyond the vicinity of a molecule. The formalized assumption is:

$$\vec{x}^* - \vec{x} > \vec{c}_i\Delta t \Rightarrow G_{\alpha\beta}(\vec{x},\vec{x}^*) \to 0 \tag{7.4}$$

This assumption allows for the simplification of the SC force scheme:

$$\vec{F}^{SC}_\alpha(\vec{x},\vec{x}^*) = -\psi_\alpha(\vec{x})\sum_k G_{\alpha k}s(\vec{x},\vec{x}^*)\sum_i w_i\psi_k(\vec{x}^*_i)\vec{c}_i\Delta t \tag{7.5}$$

In this equation, $G_{\alpha k}$ is a scalar value that accounts for the intensity of the interaction between components α and k, and the term $s(\vec{x},\vec{x}^*)$ is such that it assumes the value 1 if $\vec{x}^*$ represents an immediate neighboring node of the node in $\vec{x}$, otherwise assuming the value 0 to indicate the absence of interactions.

The sum in Equation 7.5 is made for all k components in the mixture, including the possible interaction $G_{\alpha\alpha}$ of the component with itself. This interaction allows for phase separation in single-component fluids. This force is to be computed for all components within the mixture, and the interactions are modeled according the the value of $G_{\alpha k}$. Usually, if there is no phase change, the self-interaction values $G_{\alpha\alpha}$ can be tuned to 0.

Finally, as for the pseudo-potentials, the simplest approach is to use the density as:

$$\psi_k = \rho_k \tag{7.6}$$

However, the idea of applying pseudo-potentials is to eliminate numeric instability by fixing the value of ψ to a finite value, even for large or small densities. A very accepted and commonly used pseudo-potential is:

$$\psi_k = \rho_0 \left(1 - e^{-\rho_k/\rho_0}\right) \tag{7.7}$$

This form guarantees a potential between 0 and ρ_0 for all densities, where ρ_0 is a reference density usually set to 1 in simulation parameters.

The velocities are also modified. There is a velocity for each component given as:

$$\rho_k \vec{u}_k = \sum_i f_i^k \vec{c}_i \tag{7.8}$$

The densities ρ_k for each component k can be understood as concentrations for the components in the mixture, given as:

$$\rho_k = \sum_i f_i^k \tag{7.9}$$

The mixture as a whole can be described using a barycentric velocity $\vec{u}_b$, which tracks the motion of the fluid as a whole. This velocity is the one that describes the

motion of the mixture in the NSEs. The total density ρ can also be given by a sum of the component densities.

$$\rho\vec{u}_b = \sum_k \left(\sum_i f_i^k \vec{c}_i + \vec{F}_k^{SC} \frac{\Delta t}{2} \right) \qquad \rho = \sum_k \rho_k \tag{7.10}$$

As each component k has a corresponding distribution function f_i^k, there is an equilibrium distribution function $f_{i,k}^{\text{eq}} = f_i^{\text{eq}}(\rho_k, \vec{u}_k^{\text{eq}})$. The density for the equilibrium function is the density of each component as in Equation 7.9. As for the velocity, in the case of the original SC force algorithm, the velocity used for each component is given as:

$$\vec{u}_k^{\text{eq}} = \vec{u}' + \frac{\tau_k F_k^{SC}}{\rho_k} \tag{7.11}$$

where $\vec{u}'$ is a weighted average for the velocity common to all components:

$$\vec{u}' = \frac{\sum_k \left(\rho_k \vec{u}_k \frac{1}{\tau_k} \right)}{\sum_k \left(\rho_k \frac{1}{\tau_k} \right)} \tag{7.12}$$

The relaxation constant can also be different for the components depending on the viscosity of each one of them, resulting in terms τ_k.

Nonetheless, as previously explained in Section 4.5, other force schemes can be implemented. The use of Guo-force scheme can be done by using the barycentric velocity $\vec{u}_b$ to calculate the equilibrium function (SEGA *et al.*, 2013). The force for each component is also calculated as indicated in Equation 4.69, but using the barycentric velocity.

$$\vec{u}_k^{\text{eq}} = \vec{u}_b \tag{7.13}$$

$$F_i = w_i \left(\frac{\vec{c}_i}{c_s^2} + \frac{(\vec{c}_i \cdot \vec{u}_b) \cdot \vec{c}_i}{c_s^4} - \frac{\vec{u}_b}{c_s^2} \right) \cdot \vec{F} \tag{7.14}$$

The implementation of Guo forcing has an advantage, since the original SC method leads to a dependence of the surface tension with the viscosity. It is also simpler to implement, since no significant changes in the algorithm are needed.

7.2 Surface Thermodynamics and the Free-Energy Method

The free-energy (SWIFT *et al.*, 1996) is much more complex when compared to the SC method. It is based on the thermodynamics of the surface interactions, as well as the wetting conditions. For this approach, a free-energy functional ψ is constructed as:

$$\psi = \int_V (\psi_b + \psi_g) dV + \int_A \psi_s dA \tag{7.15}$$

The components ψ_b, ψ_g, and ψ_s are dependent of space and time. ψ_b represents the bulk free-energy and leads to a equation of state that allows for the coexistence of multiple phases and components. ψ_g represents the free-energy of the interface by tracking gradients, related to the surface tension of the fluid. ψ_s is a term for interaction between the fluid and a surface, related to the wetting conditions of the problem.

The implementation of these functionals is achieved with modifications made to the pressure tensor, which results in a pressure tensor term G_i added to the LBE:

$$f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) = f_i(\vec{x}, t) + \Omega_i(\vec{x}, t) + S_i(\vec{x}, t)\Delta t + G_i(\vec{x}, t)\Delta t \tag{7.16}$$

This pressure tensor is a discretized form of the pressure tensor, $\vec{\vec{P}}$, calculated as a result of a differential chemical potential, μ, that causes a thermodynamic force to appear:

$$\nabla \cdot \vec{\vec{P}} = \rho \nabla \mu \tag{7.17}$$

The specific form of the pressure tensor depends on the model used in the calculations. This term can be incorporated into the distribution function or simply be added as a forcing term as:

$$\vec{F} = -\nabla \cdot \vec{\vec{P}} + \nabla(\rho c_s^2) = -\rho \nabla \mu + \nabla(\rho c_s^2) \tag{7.18}$$

where the term $\nabla(\rho c_s^2)$ is the pressure (not pressure tensor) gradient.

Modifications can be made to these models to minimize and cancel errors, but this work will not delve further into this approach. The important notion is that the potential ψ can be calculated using macroscopic quantities, such as the surface tension and wetting conditions given by the contact angles.

7.3 Chemical Reactions

Chemical reactions and different chemical species can also be addressed by using multiple distribution functions. One general function f_i may be applied to ensure mass conservation and calculated as the simple single-component LBM method. In such approaches, there are functions g_i^α that represent the distribution of the particles of each component α, contrasting with the distribution of all particles f_i. On the resulting equations for the component distribution functions, it is possible to implement source and sinking terms related to the kinetics of chemical reactions.

These sinking and source terms may be of the form of reaction rates. However, it is an approach that do not contemplate the microscopic nature of the method and needs macroscopic quantities, which decreases the predictability of the method. To fully appreciate the mesoscopic nature of th LBM, reactional models based on the particle collision theory have been proposed presenting satisfatory results (BRESOLIN; OLIVEIRA, 2012; ABDOLLAHZADEH *et al.*, 2018).

A possible approach is to perform a multi-scale simulation to obtain the macroscopic parameters using microscopic models, then apply such parameters in LBM to see the macroscopic effects otherwise impossible to fully simulate in large scale. This can also be implemented for multicomponent flows with no reactions by calculating the surface tension between the two phases or components, as the wetting conditions to simulate fluid-solid interactions.

There are many prospects regarding reactive flows, especially combined with adsorption in porous media, as these may be used in simulations of catalysis and reactors, but it is not the focus of the present study. Nevertheless, momentum and energy transfer have important applications, even in the absence of any chemical reaction, as in the case of extraction of oil from reservoirs.

7.4 Binary Fluid Flow

The presented methods are the most common approaches in LBM. They are also necessary steps in understanding and applying the method for more complex cases. It also is notable the low number of works related to multicomponent flows and mass

transfer, when compared to other applications of the LBM in the literature. These are very important aspects that have the potential to contribute much to CFD since the LBM has several advantages despite being much newer and not yet as widespread as the traditional methods.

The chosen scheme was based on the the Shan-Chen pseudo-potential method explored in Section 7.1, using Guo-force as explained in Section 4.5. The the Guo-force scheme does not cause great modifications in the overall method. The velocity was modified to include the force effects and calculated for each fluid as in Equation 4.126 in Section 4.5. The barycentric velocity was obtained as indicated in Equation 7.10. The equilibrium was calculated using the individual components densities and the general barycentric velocity of the fluid. The force term F_i was also modified to include the barycentric velocity instead of the individual velocity for each component, as indicated in Equation 7.14.

The pseudo potentials $\psi(\rho)$ were set as the density, and as the original SC potential:

$$\psi(\rho) = \rho \tag{7.19}$$

$$\psi(\rho) = (1 - e^{\rho}) \tag{7.20}$$

This work focused on binary fluids, but the extension to more than two components is simple when a binary fluid is already implemented.

Moreover, this work proposed focusing on the study of binary fluids of non-interacting molecules (ideal gases). A hypothesis was made that the methods for binary fluids must present the same behavior as the single-component flow if there is no interaction. This is natural, since the self-interaction terms $G_{\alpha\alpha}$ in the Shan-Chen method (Section 7.1) are usually not considered unless phase separation is involved. Therefore, this work assumes a single population can be separated into two whilst maintaining its general properties. This should be a valid technique to validate and evaluate multi-component LBM methods.

We propose that a system with only non-interacting components could be simulated this way. This could also apply to cases when the binary interaction terms are equal, or very close, to the self-interaction terms. This implies there is no difference or

preference in the way each particle interacts with other particles of the same component or with particles of the other component. The main difference would be given by the boundary conditions, which do not need to be the same for both populations.

This is a good approximation for molecules that are very similar, in thermodynamic conditions of low pressure (under about 1 bar) and high temperatures (over about 300 K), in which the molecules may be approximated by an ideal gas. In these cases, the interactions between the molecules may be considered negligible.

Therefore, the binary interaction potential $G_{\alpha\beta}$, and the self-interaction potential $G_{\alpha\alpha}$, which represents the interaction of the molecules of one component with themselves, were both set to 0, indicating the absence of interactions.

An ideal gas mixture may be simulated by setting these potentials to 0. That is the same case for a single component fluid separated into two populations. It becomes clear that these populations when summed have to present the same results as a unique population with the total density.

This approach simplifies the simulations, and is a good approximation if the aforementioned condition of no interaction apply. In fact, interactions between the components were not considered in the scope of this work, focusing instead in how external forces can act distinctly in each component, affecting the result.

For static fluids, a periodic computational domain as presented in Section 4.1.3 was used. The density (or pressure) of each component was set in one of the boundaries using the anti-bounce-back method as explained in Section 4.1.7, which was better for setting the desired individual density at the boundary.

In the presented cases, the steady-state was desired. Therefore, no constraints were adopted for the mass allowed to enter or leave the domain as the simulation proceeded. The density profiles after reaching the equilibrium in the conditions set, including the resulting mass, were also of interest in this case.

When testing to obtain density profiles, the velocity u was substituted by the density ρ when testing the stop criteria given by Equation 5.1.

No simulation results are presented here. Nonetheless, this section is necessary when deriving the adsorption model based on the Shan-Chen method. The efects of

the non-interacting fluids are also used when simulating the adsorption of binary fluids, where the effects are apparent and the assumptions presented here are necessary. Moreover, a new model for an equilibrium function for multicomponent flows is derived in this work, based on the incompressible Zou-He equilibrium distribution function shown in Section 4.4. This equilibrium function will also be used when simulating the adsorption of binary fluids and comparing binary to single population adsorption.

7.5 Alternative Equilibrium Distribution for Binary Flows

To calculate binary non-interacting fluids, a new model for the equilibrium function was proposed and tested for adsorption. This alternative equilibrium distribution function was not based directly on other models from the literature, but is related to the incompressible model proposed by Zou *et al.* (1995) and the Shan-Chen method using Guo-force, as explained in Section 7.

In Section 4.4, it was shown how incompressibility can be incorporated to LBM by altering the equilibrium distribution function f_i^{eq}. In fact, several modifications are made to the equilibrium distribution function to accommodate different conditions for LBM.

A general form for f_i^{eq} is given in the form of Equation 4.122. For the simulation of multi-component flows, a new distribution function of this form was proposed in this study for each component α as:

$$f_{i,\alpha}^{\mathrm{eq}} = w_i \left(\rho_\alpha + a_\alpha \frac{\vec{U}\cdot\vec{c}_i}{c_s^2} + a_\alpha \frac{\left(\vec{U}\cdot\vec{c}_i\right)^2}{2c_s^4} - a_\alpha \frac{\vec{U}\cdot\vec{U}}{2c_s^2} \right) \tag{7.21}$$

With $a_\alpha = \frac{\rho_\alpha}{\rho}$, and $\vec{U} = \rho\vec{u}$ as the momentum density. This new equilibrium distribution considers the equilibrium for incompressible models in single components weighted by a factor that considers the individual components. To respect the Shan-Chen method with Guo-forces, the velocity is substituted by the barycentric velocity.

The final equation for the distribution function reads:

$$f_{i,\alpha}^{\mathrm{eq}} = w_i \left(\rho_\alpha + \frac{\rho_\alpha}{\rho} \frac{\vec{U}_b \cdot \vec{c}_i}{c_s^2} + \frac{\rho_\alpha}{\rho} \frac{\left(\vec{U}_b \cdot \vec{c}_i\right)^2}{2c_s^4} - \frac{\rho_\alpha}{\rho} \frac{\vec{U}_b \cdot \vec{U}_b}{2c_s^2} \right) \tag{7.22}$$

This new equilibrium was not created from association with macroscopic or microscopic models, but to facilitate the implementation. Nonetheless, it seemed to behave well for all cases studied. It is equivalent to multiply the incompressible distribution function given by Equation 4.122 for $\frac{\rho_\alpha}{\rho}$. This model was able to simulate the binary fluids recovering the behavior observed for single-populations. It was also observed to reduce numerical instability during the simulations.

The results presented in Section 8.3 will use the proposed model for the equilibrium distribution function. The results were satisfactory for the cases tested. This distribution function was not tested for interacting (real) fluids, but is assumed to work, since no great modifications were made in the general form of the equilibrium distribution.

8 Adsorption

8.1 Modeling Adsorption and Solid-Fluid Interactions

The last part, and the major contribution of this study, is the study of adsorption models. It is an important issue and still focuses on research (SHARMA; STRAKA; TAVARES, 2019).

Adsorption occurs when particles of the fluid attach to the walls, forming a layer of higher density, as seen in Figure 31a. The phenomenon can be modeled as an interaction between the fluid particles and the walls in the form of a force (GUO *et al.*, 2016; CAI *et al.*, 2018). Shi *et al.* (2019) used solid-wall interaction forces to simulate the adsorption of falling films. Zhou *et al.* (2015) also simulated pore-scale adsorption by implementing special boundary conditions on the distribution functions of the components instead of the usual approach.

Usually, only particles close to the wall are proposed to be affected, but it is possible to understand adsorption forces in the form of a potential that varies with distance, forming potential curves parallel to the surface. With such interpretation, the implementation of a lattice with nodes corresponding to these curves can be imagined as seen in Figure 31.

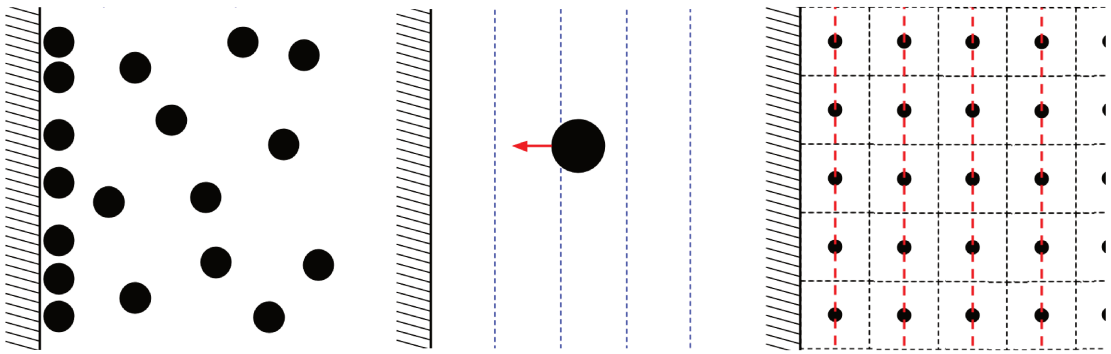


Figure 31 – Adsorption schemes: (a) adsorbed layers; (b) adsorption force and potential curves (blue); (c) adsorption potential curves (red) in lattice.

The modeling of adsorption can be made using the same approaches as the modeling of multicomponent flows. In the free-energy model, the interaction with the walls is given by the term ψ_s that composes the free-energy functional. As stated in Section 7.2, these calculations are not trivial.

A more straightforward approach is to model the adsorption by applying an adsorption force. An extension of the Shan-Chen (CS) pseudo-potential method to apply to solid nodes is possible. In this approach, the adsorption is modeled with a potential given by G'_α that represents the interaction between a component and a solid wall with density ρ_s. This force can be written as:

$$\vec{F}_\alpha^{\text{ads}}(\vec{x}) = -\psi_\alpha(\vec{x}) G_\alpha^{\text{ads}} s'(\vec{x}) \sum_i w_i \psi_s(\rho_s) \vec{c}_i \Delta t \tag{8.1}$$

The term $s'(\vec{x})$ includes the effect of the distance. In the same manner as stated in the multicomponent fluid, one possible assumption is the interaction is only significant to the neighboring nodes. In the original SC scheme, $s'(\vec{x})$ is set to zero, unless the neighboring node is a solid node. The result is a constant density in the bulk of the fluid, forming an absorbed layer only in the node attached to the wall, as can be seen in the work of Guo *et al.* (2016).

The pseudo-potential for the solid node is:

$$\psi_s = \psi(\rho_s) \tag{8.2}$$

The sum is also performed to all possible positions $\vec{x}_j^*$, resulting in a total adsorption force acting on component α given as:

$$\vec{F}_\alpha^{\text{ads}}(\vec{x}) = -\psi_\alpha(\vec{x}) G_\alpha^{\text{ads}} \sum_j s'(\vec{x}, \vec{x}_j^*) \sum_i w_i \psi_s(\vec{x}_j^*) \vec{c}_i \Delta t \tag{8.3}$$

If the pseudo-potential $\psi_s(\vec{x}_j^*)$ for the solid is constant, meaning the density of the solid is constant, then it can be absorbed into the value of G_α^{ads}.

The characterization of the problem and validation was performed by comparing the results obtained with those presented in Guo *et al.* (2016), for the pseudo-potential $\psi = \rho$.

The problem was implemented in a bidimensional grid using the D2Q9 velocity set. Nonetheless, by considering the adsorption in an infinite plate the problem becomes one-dimensional with the density profile varying with the distance to the plate. In terms of simulation, a 11x11 grid was used with a periodic boundary condition to grant the density was the same in all nodes of the same distance to the wall. The density on the boundary opposed to the plate was set to a constant value (ρ_∞) for each simulation, using the anti bounce-back (ABB) technique describes in Section 4.1.7.

For the boundary where adsorption occurs, in the case of a single component a simple bounce-back (BB) condition was enough to simulate adsorption. For a binary fluid, it was necessary to be more careful. The expected behavior was recovered using virtual nodes outside of the computational domain (Section 4.1.8), which are unaffected by forces, and performing the BB technique on these nodes.

The Guo forcing scheme was applied to convert $\vec{F}_\alpha^{\text{ads}}$ in the forcing terms that appear in LBM F_i^{ads}, as indicated in Section 4.5 for single-component adsorption. The adsorption of binary fluids was also performed as indicated in Section 7.1, using the barycentric fluid velocity.

The adsorption forces were implemented first for single populations. Then, the populations were separated in non-interacting parts and the adsorption was simulated again. The single population is shown to behave as the sum of the separated distributions.

As for the form of the adsorption force, a generic interaction value was used and varied to observe the effects on the resulting profile. Nonetheless, models to describe adsorption are also available, such as the potential proposed by Polanyi (1932) and proved to be efficient for several cases (KADLEC, 2001; PONCE *et al.*, 2011).

The assumption of the neighboring nodes explained in Section 7.1 that states the forces only act upon the immediate neighbors is still assumed to be true. However, the models for adsorption potentials can go beyond this immediate assumption. To implement another force, a dependence of the distance between the node and the wall could be observed. This does not affect the terms ψ_s and G_α^{ads}, since they are dependent only on the molecules present.

The force was rewritten as:

$$\vec{F}(\vec{x},\vec{x}_s) = \vec{F}D^{-n} \tag{8.4}$$

where $D = |\vec{x} - \vec{x}_s|$ represents the distance between the position $\vec{x}$ where the force acts and the position $\vec{x}_s$ where a solid node is located.

Clearly, in a real case, this would be summed for all solid node positions to determine the total force exerted in each point. n defines how the interaction decay with distance. As an example, for an electric or gravitational force, $n = 2$. Polanyi (1932) developed a theory based on long-range van der Waals forces interactions, which resulted in an adsorption potential. The Potential Theory for Absortion by Polanyi was later shown to be adequate to simulate adsorption in various systems (PONCE *et al.*, 2011; ABDOLLAHZADEH *et al.*, 2018).

The original potential can be found in the referred works, but an important thing is the dependence it establishes with the distance of the molecules from the wall. As seen in Guo *et al.* (2016), potential curves for adsorption can be seen, which agrees with the nature of the LBM by also dividing space in regions. Therefore, a dependence with distance can be assumed to an adsorption potential for LBM.

In the original works, the dependence of the potential with the distance is found to obey a relation $\Psi(D) \propto \frac{1}{D^3}$. As the force $\vec{F}$ corresponding to a potential Ψ is given by $\vec{F} = -\nabla\Psi$, the relation of the force with the distance can be found to be:

$$\vec{F}(D) \propto \frac{1}{D^4} \tag{8.5}$$

This relation for the dependence of forces with distance can be assumed to hold true for the nodes LBM. Since the distribution function represents the molecules within a node at a given position, it is natural to assume that the force will act on each individual molecule with this dependence. Therefore, the force term in LBM can be also related to the distance in the same manner, since it represents a force acting on the particles in that position. A scheme for plying adsorption potentials to a lattice can be seen in Figure 31

Thus, the force field can be discretized. It can be implemented using the scheme presented in Equation 8.1 with modifications of the term $s'(\vec{x})$ to accommodate this dependence to the distance. As previously explained in Section 7.1, the forces have

to be considered for each solid node and then summed. In the one-dimensional case, however, it reduces to a trivial dependence of distance $s'(D)$ without need to sum.

The Force in the exact position of a node can be calaulated by decomposing the distance shown in Equation 8.4 in terms of the lattice units as:

$$\vec{F} = \vec{F}D^{-n} = \vec{F}(N\Delta x)^{-n} \tag{8.6}$$

where N is the number of nodes between the position of the solid node and the node upon which the force acts.

If the assumption holds true that this relation is kept in the LBM as it is presented in Equation 8.6, then this dependence can be directly implemented for each node. In the case of a one-dimensional flow it is trivial, and the term s' for a $\Delta x = 1$ should read:

$$s' = \frac{1}{N^n} = \frac{1}{D^n} = \frac{1}{(x - x_s)^n} \tag{8.7}$$

By variating the factor n, therefore, it is possible to obtain various curves representing the density profile. The other part of the Polanyi potential is absorbed by G_α^{ads} and ψ_s. A complete derivation of the potential for the LBM could be made, however it has not been achieved at this point of the present study.

8.2 Single Component Adsorption

Adsorption can be implemented as a force that acts on neighboring nodes of a solid boundary node. In this section, the Shan-Chen based adsorption method was applied. The density profile was obtained using a value for $G_\alpha^{\text{ads}}\psi_s(\rho_s) = 0.1$. Isotherms showing the absorbed amount were calculated varying the bulk density (ρ_∞) for two different pseudo-potentials ($\psi(\rho) = \rho$ and $\psi(\rho) = 1 - e^{-\rho}$). Other isotherm was calculated for $\psi(\rho) = \rho$ varying the intensity of the force ($G_\alpha^{\text{ads}}\psi_s(\rho_s)$).

The results obtained for the density profile (Figure 32) were in accordance with the profiles present in the work of Guo *et al.* (2016). This is a good indication of a successful implementation. As the traditional SC adsorption force only applies to the neighboring nodes, the only nodes with noticeable differences in local density are those in contact with solid nodes ($y = 9$ in the figure).

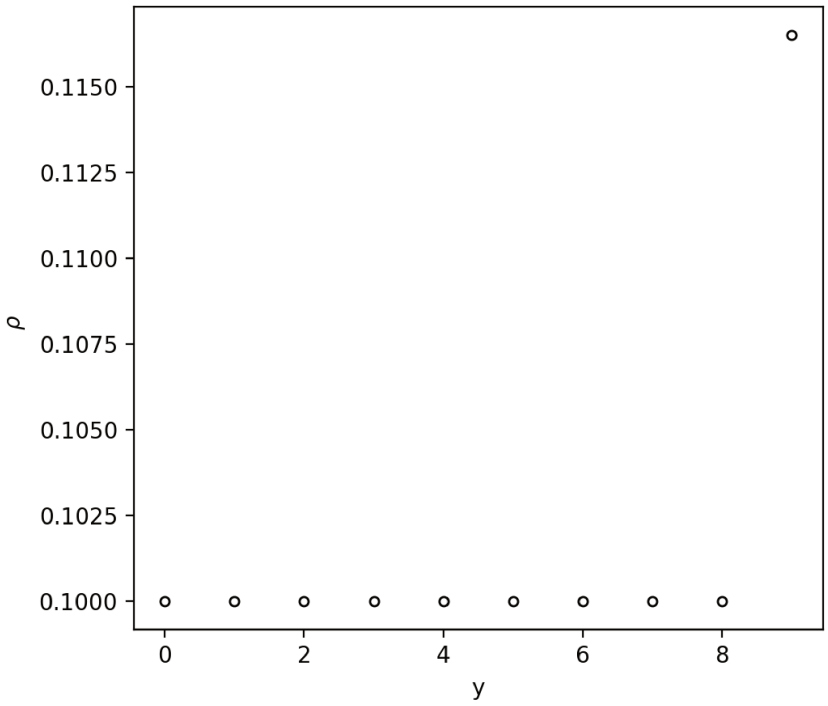
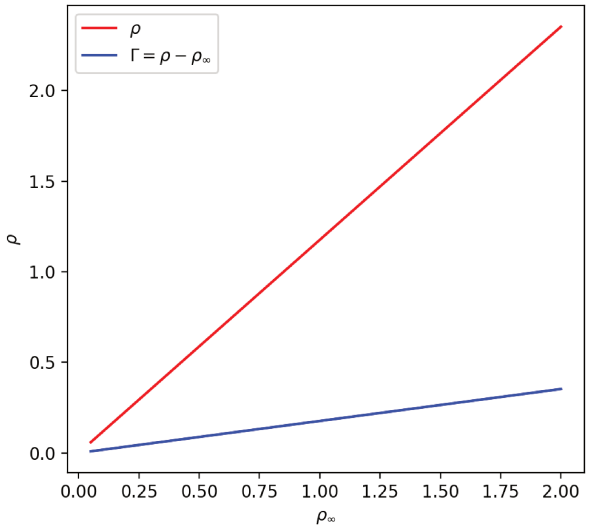


Figure 32 – Density profile of fluid submitted to adsorption force.

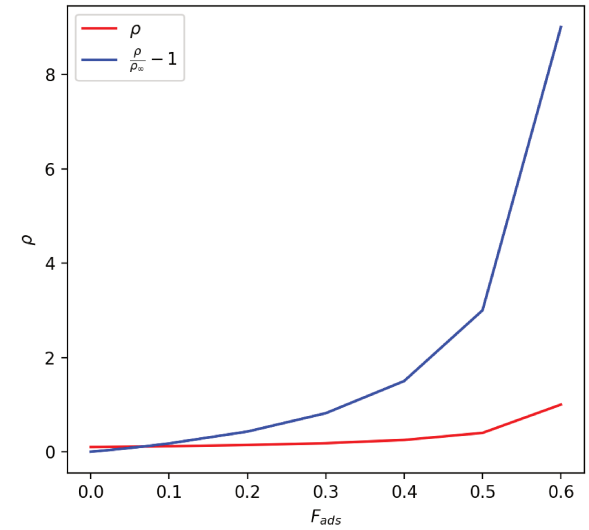
The isotherm obtained from the system for $\psi(\rho) = \rho$ show a relation between the density on the surface of the solid, or the absorbed amount ($\Gamma = \rho - \rho_\infty$), and a reference density several nodes away from the solid node, corresponding to the density of the bulk of the fluid (ρ_∞). Other isotherm shows the variation of fluid density on the wall (given in terms of ρ and $\frac{\rho}{\rho_\infty} - 1$), as interaction intensity ($G_\alpha^{\text{ads}} \psi_s(\rho_s)$) varies from 0 to 0.6. The results, shown in Figure 33, are also in good agreement with the work of Guo *et al.* (2016).

Moreover, testing different pseudo-potentials $\psi(\rho)$ used in the method resulted in different profiles and led to specific isotherms (Figure 34). Other more complex potentials yield more complex graphs, and the intensity of the force can also affect the results, as shown by Guo *et al.* (2016).

This force commonly used in the literature clearly acts solely on the direct vicinity of the solid nodes. Therefore, only one node is to have its density influenced by the



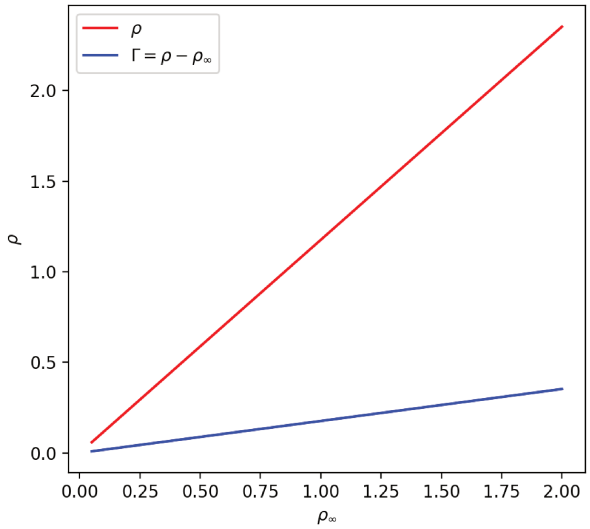
(a)



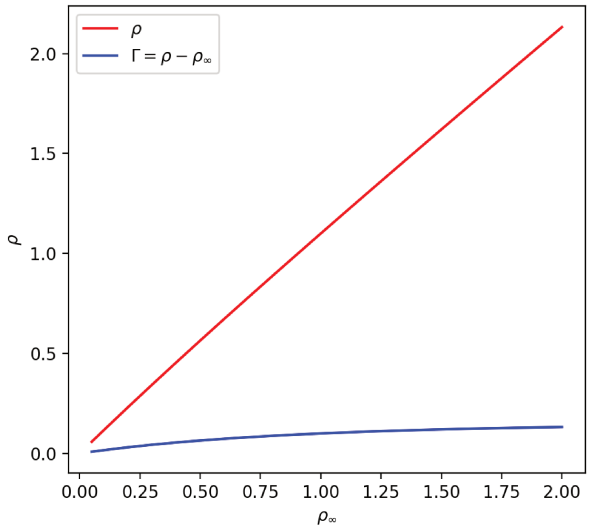
(b)

Figure 33 – Adsorption isotherms showing the variation of the density in the wall: (a) with the reference density of the bulk; (b) with the magnitude of the force.

adsorption force. This assumption is valid if the force is considered to decay fast with the distance from the wall. This may be a robust approximation for adsorption, since the



(a) $\psi(\rho) = \rho$



(b) $\psi(\rho) = 1 - e^{-\rho}$

Figure 34 – Adsorption isotherms for different models applied to the pseudo-potentials.

phenomenon can be interpreted as an adsorbed layer with the bulk mostly unaffected. Nonetheless, other force fields varying with the distance can be implemented. The force

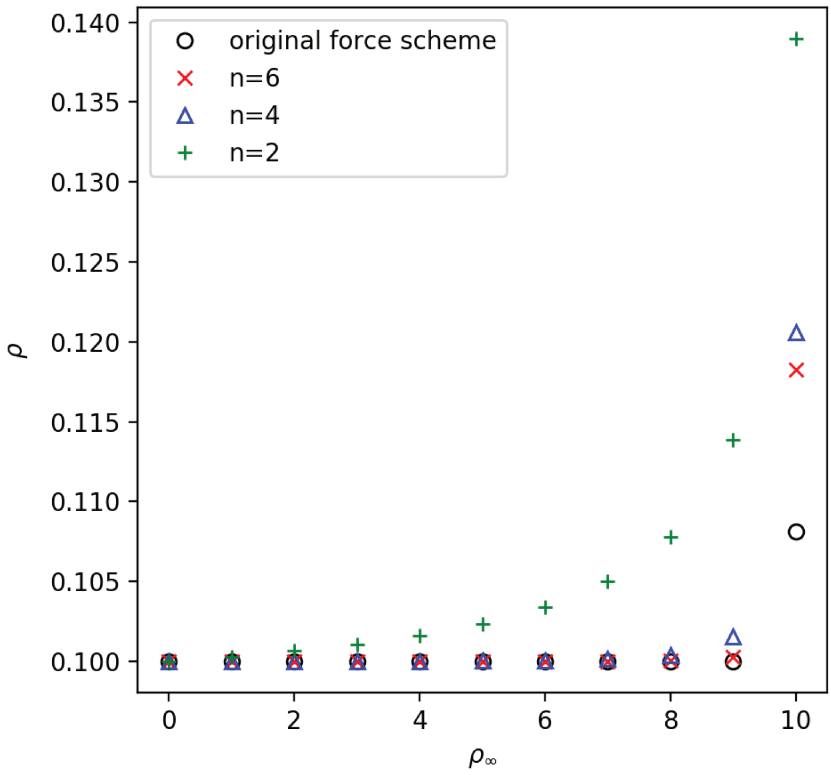


Figure 35 – Effect of the long-range force in the density profile.

F decays with the distance D by a factor of n as:

$$F(D) \propto \frac{1}{D^n} \tag{8.8}$$

Various potentials were tested in this manner, generating the profiles shown in Figure 35. The forces related to macroscopic interacting potentials usually decay by a factor of $n = 2$ with the distance, that being the case for the gravitational and electric forces. These forces are attractive or repulsive. In a microscopical level, however, the interactions become more complex. As an example, the Lennard-Jones potential for molecular interactions has both attractive and repulsive terms, each presenting a different decaying factor. The repulsive short-range potential decays by a factor of $n_1 = 12$ whereas the long-range attractive potential decays by a factor of $n_2 = 6$.

For adsorption, Polanyi (1932) developed a potential that decays with a factor $n = 4$. For the factor $n = 1$ the decayment generates a linear force. The factor $n = 2$, which relates to gravitational and electrical forces, shows a large distance affected by the force. Larger factors such as $n > 6$ tend to confine the force to the neighboring node, which validates the original hypothesis.

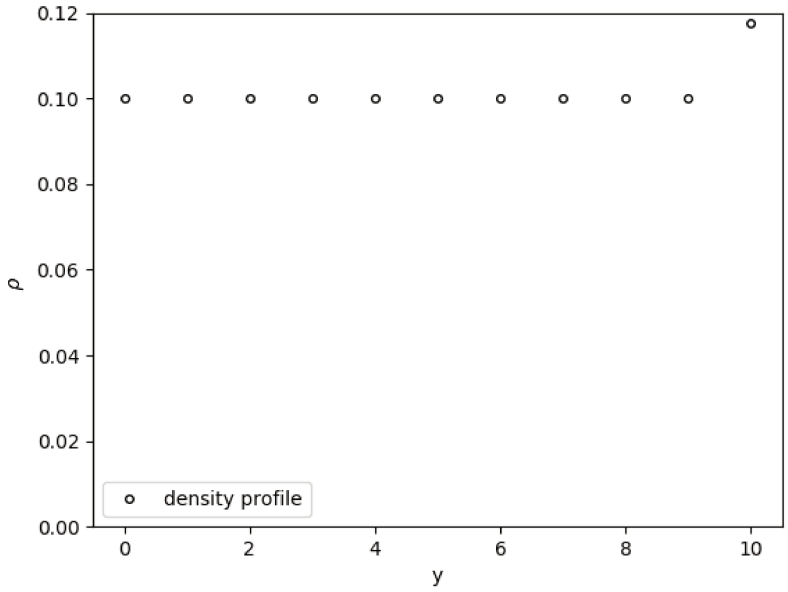
However, the factor $n = 4$, which is though to be the most adequate to reproduce the Polanyi adsorption potential, shows a short-range interaction with a visible effect on other nodes than the immediate neighbors. Despite this value being small, the adsorbed amount also varies from the original Shan-Chen scheme for the same magnitude in the interaction parameter. Therefore, the derivation of an interaction parameter considering the short-range approximation has to consider and accommodate these effects, and a potential based on the works of Polanyi (1932) would have to consider using a long-range interaction force.

The density profiles for $n = 4$ and $n = 6$ were similar. The values observed for these two potentials were similar. However, the potential that considers $n = 2$ has a much longer curve, and the absorbed amount is also much greater. Therefore, this potential is shown to be inadequate for these simulations. The Polanyi equivalent potential is given by $n = 4$, and is demonstrated to have a range of 3 nodes in the case presented. This potential should be the more adequate to simulate adsorption, since it does not model a single layer as the usual Shan-Chen based potential. Moreover, it has a microscopic associated potential which could be brought to the LBM. It could allow for more predictive and simulations for wider applications.

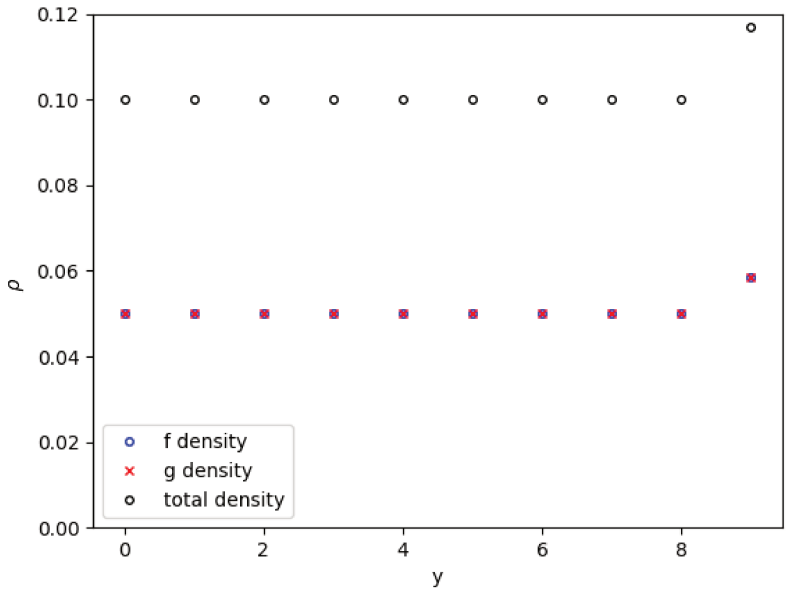
8.3 Binary Non-Interacting Fluid Adsorption

Finally, combining the observations made in Section 7.4 and 8.2, it was possible to apply the concept of adsorption for a binary fluid with non-interacting components.

Figure 36 shows that the implemented adsorption force is valid when separating the fluid into different distribution functions. In this case, the same interaction was applied to each population yielding the same results as the same force applied to a single population of summed density with minor errors.



(a)

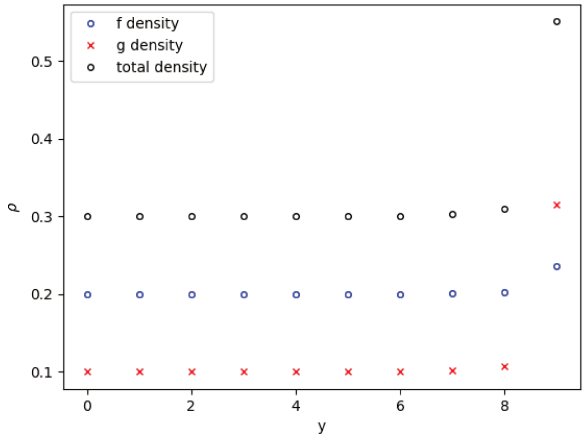


(b)

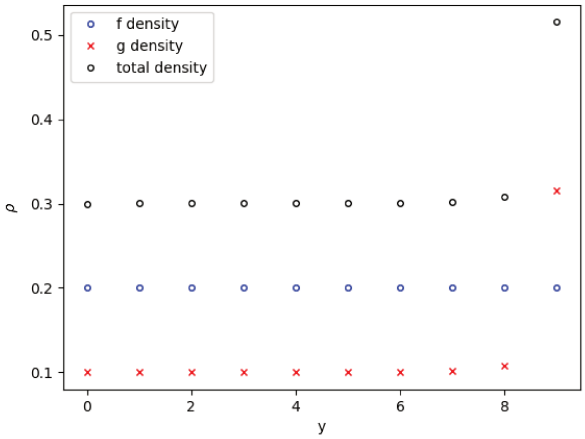
Figure 36 – Simulated adsorption using: (a) single population; (b) two populations.

This result points to the validation of the Shan-Chen methodology used to implement multi-component flows. The equilibrium distribution used to generate this result was presented in Section 7.5, also validating the function for this case.

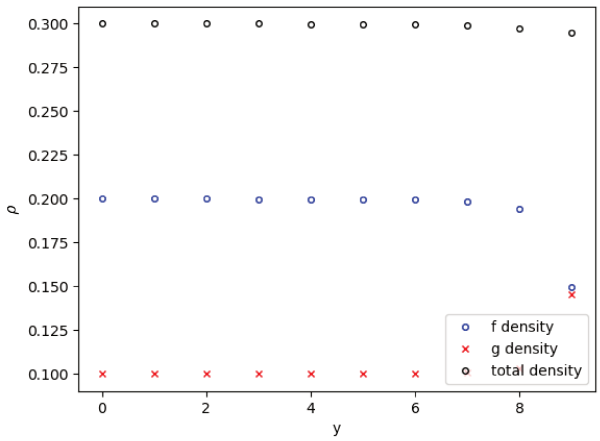
By applying different adsorption forces, however, the results start behaving in a more interesting manner. Yet, no effects of one population were felt by the presence of the other with the interaction potentials set to 0. The populations behave as if they were independent of one another, which is natural since no repulsion effects were presented nor limitations as the total adsorbed amount was applied. Figure 37 show examples of non-interacting fluid with different adsorption forces applied.



(a) Different intensity for each component



(b) Non-adsorbent component



(c) Repulsion force for one component

Figure 37 – Differential adsorption of binary non-interacting fluid.

9 Conclusions and Perspectives

The Lattice Boltzmann Method (LBM) was implemented, and several simulations were performed. The code was validated using the Poiseiulle, Couette, and combined flows. When compared to the analytical solutions, LBM presented good results overall. The techniques used proved to be satisfactory when simulating these flows, and the errors were small when compared to the analytical solutions for the scale of the problem (maximum velocity).

Moreover, when simulating more complex flows, LBM also presented good qualitative results for the Lid-Driven Cavity and Poiseuille flow around an infinite cylinder. These LBM simulations matched both experimental results and other CFD methods. No analytical solutions exist to these cases.

Forces also proved to be implemented successfully. The code was able to simulate uniform body-forces perpendicular to the flow, such as gravity. The substitution of the pressure drop for a body-force was also demonstrated as a manner to eliminate compressibility effects in the Poiseuille flow with satisfactory results. However, the limitations and problems of such an application make this approach difficult to implement. Among these difficulties are the necessity of implementing and calculating the force in each step, which turns the method much more expensive computationally.

Overall, the results imply the code was successfully implemented and tested. The chosen boundary conditions (bounce-back, virtual nodes, non-equilibrium bounce-back) showed to be adequate. Complex geometries were also shown to behave as expected, including convective effects which led to the formation of vortices when velocities were high enough.

As for the limitations, the code was not able to simulate high velocities and forces. Unit conversion is also a complicate matter, since the value of the relaxation constant and the velocities is dependent on both space and time lattice units. Thereafter, a strong coupling of spatial and time scales is observed, limiting the scope of problems that can be simulated without introducing modifications in the method. The units must be converted carefully to obtain the physical values associated.

Flow in porous media was also simulated in two dimensions. The meshes were obtained using the Ising model, which provided random grids containing channels and other various structures. This approach allows for the study of flow in microscopic scales through porous media, in cavities, channels, and constrictions. Interestingly, the fluid was shown to be almost stagnated in a great part of the computational domain, with flow occurring only in a small part of the computational domain, which implies difficulty to penetrate those cavities and extract the fluid contained within these places. Nonetheless, these results can vary depending on the porosity of the grid and the geometry of the pores in other porous media. The velocity of the flow through the porous media was shown to have a linear dependence with the Pressure difference applied, as expected of laminar flow in the Darcy regime and, therefore, in accordance with Darcy's Law.

The last part of the work focused on binary fluids and adsorption. Binary flows were implemented using a methodology based on the Shan-Chen model of pseudo-potentials and attractive forces. The Guo-force scheme has proven to be an adequate method when implementing these attractive forces. Two non-interacting populations were shown to behave as one denser population, which is expected of non-interacting molecules. This approach reassures the Shan-Chen model with Guo-forces do not deviate much of the expected behavior for the flow, which has to be respected.

To perform these calculations, an alternative hybrid equilibrium function model was proposed for calculating non-interacting binary fluids. It was obtained from an incompressible model for the equilibrium distribution function and the Shan-Chen using Guo forcing scheme. This alternative equilibrium distribution was shown to yield satisfactory results, at least qualitatively, for adsorption.

As for the adsorption process, the Shan-Chen model was successfully implemented, as the results suggest when compared to other works. Nonetheless, it was demonstrated how this model limits the adsorption, and how a more comprehensive scheme could be obtained from the Polanyi model. The dependence of the force with the distance was shown to affect the absorbed amount when an interaction potential of the same magnitude is applied. The Polanyi based force was also shown to affect at least two more neighboring nodes beyond the region where the Shan-Chen potential is effective. More effective adsorption models could be implemented following those models, since for an immovable solid, this dependence do not change during the computations.

Overall, the proposed objectives of successfully implementing the Lattice Boltzmann Method were fulfilled. The implementation for multicomponent fluids and adsorption forces was accomplished. This work initiates the exploration of the LBM, and can be expanded in future works to explore real cases, where the Lattice Boltzmann method is promising.

As suggestion future works, the method could be expanded to tridimensional lattices and to include effects associated with heat transfer. Another area of great interest is the implementation of chemical reactions, which are still minimal in the published works in the field. The combination of reactive flow, adsorption, and porous media can be an interesting topic, especially in chemical engineering, as it allows for the accurate simulation of chemical reactors and catalysis. Moreover, a field where the Lattice Boltzmann method is also promising is in the simulation of reservoirs for oil extraction.

Overall, multicomponent models for the Lattice Boltzmann Method are available, but they are not completely established, especially when adsorption and chemical reactions are involved. As for adsorption, few works focuses on the adsorption potential as a microscopic approach. This is an open field which could potentially be integrated to the Lattice Boltzmann Method in the future to perform more accurate and predictive simulations.

Bibliography

ABDOLLAHZADEH, Y.; MANSOURPOUR, Z.; MOQTADERI, H.; AJAYEBI, S. N.; MONTAZERI, M. M. A molecular collision based Lattice Boltzmann method for simulation of homogeneous and heterogeneous reactions. *Chemical Engineering Research and Design*, v. 136, p. 456 – 467, 2018. ISSN 0263-8762. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0263876218302892>. Cited 2 times on pages 110 and 118.

AKKER, H. E. V. den. Lattice Boltzmann simulations for multi-scale chemical engineering. *Current Opinion in Chemical Engineering*, v. 21, p. 67 – 75, 2018. ISSN 2211-3398. Energy and environmental engineering: reaction engineering and catalysis. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S2211339818300030>. Cited on page 104.

ARAB, M.; SEMMA, E.; PATEYRON, B.; GANAOUI, M. E. Determination of physical properties of porous materials by a lattice boltzmann approach. *Fluid Dynamics & Materials Processing*, v. 5, n. 2, p. 161–176, 2009. ISSN 1555-2578. Available from Internet: <http://www.techscience.com/fdmp/v5n2/24439>. Cited on page 97.

BAKHSHIAN, S.; HOSSEINI, S. A.; SHOKRI, N. Pore-scale characteristics of multiphase flow in heterogeneous porous media using the Lattice Boltzmann method. *Scientific Reports*, v. 9, n. 1, p. 3377, 2019. ISSN 2045-2322. Available from Internet: <https://doi.org/10.1038/s41598-019-39741-x>. Cited on page 103.

BAŞAĞAOĞLU, H.; SUCCI, S. Lattice-Boltzmann simulations of repulsive particle-particle and particle-wall interactions: Coughing and choking. *The Journal of Chemical Physics*, v. 132, n. 13, p. 134111, 2010. Available from Internet: <https://doi.org/10.1063/1.3374685>. Cited 2 times on pages 15 and 103.

BHATNAGAR, P. L.; GROSS, E. P.; KROOK, M. A model for collision processes in gases. i. small amplitude processes in charged and neutral one-component systems. *Phys. Rev.*, American Physical Society, v. 94, p. 511–525, May 1954. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRev.94.511>. Cited on page 26.

BOEK, E. S.; VENTUROLI, M. Lattice-Boltzmann studies of fluid flow in porous media with realistic rock geometries. *Computers & Mathematics with Applications*, v. 59, n. 7, p. 2305 – 2314, 2010. ISSN 0898-1221. Mesoscopic Methods in Engineering and Science. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0898122109006427>. Cited 2 times on pages 31 and 97.

BRESOLIN, C.; OLIVEIRA, A. An algorithm based on collision theory for the Lattice Boltzmann simulation of isothermal mass diffusion with chemical reaction. *Computer Physics Communications*, v. 183, n. 12, p. 2542 – 2549, 2012. ISSN 0010-4655. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0010465512002317>. Cited on page 110.

CAI, J.; HUAI, X.; LIU, B.; CUI, Z. Numerical prediction of thin liquid film near the solid wall for hydraulic cavitating flow in microchannel by a multiphase Lattice Boltzmann model. *International Journal of Heat and Mass Transfer*, v. 127, p. 107 – 115, 2018. ISSN 0017-9310. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0017931017348238>. Cited 2 times on pages 104 and 115.

CHAMPMARTIN, S.; AMBARI, A. Kinematics of a symmetrically confined cylindrical particle in a "stokes-type" regime. *Physics of Fluids*, v. 19, n. 7, p. 073303, 2007. Available from Internet: <https://doi.org/10.1063/1.2747659>. Cited on page 87.

CHAPMAN, S.; COWLING. *The Mathematical Theory of Non-uniform Gases: An Account of the Kinetic Theory of Viscosity, Thermal Conduction and Diffusion in Gases*. 2. ed. [S.l.]: Cambridge University Press, 1952. (Cambridge Mathematical Library). ISBN 9780521408448. Cited on page 56.

CHEN, S.; DOOLEN, G. D. Lattice Boltzmann method for fluid flows. *Annual Review of Fluid Mechanics*, v. 30, n. 1, p. 329–364, 1998. Available from Internet: <https://doi.org/10.1146/annurev.fluid.30.1.329>. Cited on page 14.

CHEN, S.; MARTíNEZ, D.; MEI, R. On boundary conditions in Lattice Boltzmann methods. *Physics of Fluids*, v. 8, n. 9, p. 2527–2536, 1996. Available from Internet: <https://doi.org/10.1063/1.869035>. Cited on page 40.

DENG, H.; HOU, Y.; CHEN, W.; PAN, F.; JIAO, K. Lattice Boltzmann simulation of oxygen diffusion and electrochemical reaction inside catalyst layer of pem fuel cells. *International Journal of Heat and Mass Transfer*, v. 143, p. 118538, 2019. ISSN 0017-9310. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0017931019329242>. Cited on page 105.

EGGELS, J. G. Direct and large-eddy simulation of turbulent fluid flow using the Lattice-Boltzmann scheme. *International Journal of Heat and Fluid Flow*, v. 17, n. 3, p. 307 – 323, 1996. ISSN 0142-727X. Available from Internet: <http://www.sciencedirect.com/science/article/pii/0142727X96000446>. Cited on page 37.

FANG, W.-Z.; TANG, Y.-Q.; BAN, C.; KANG, Q.; QIAO, R.; TAO, W.-Q. Atomic layer deposition in porous electrodes: A pore-scale modeling study. *Chemical Engineering Journal*, v. 378, p. 122099, 2019. ISSN 1385-8947. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S1385894719314937>. Cited on page 105.

FERRENBERG, A. M.; XU, J.; LANDAU, D. P. Pushing the limits of monte carlo simulations for the three-dimensional ising model. *Phys. Rev. E*, American Physical Society, v. 97, p. 043301, Apr 2018. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevE.97.043301>. Cited 2 times on pages 97 and 98.

FRISCH, U.; HASSLACHER, B.; POMEAU, Y. Lattice-gas automata for the navier-stokes equation. *Phys. Rev. Lett.*, American Physical Society, v. 56, p. 1505–1508, Apr 1986. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevLett.56.1505>. Cited on page 21.

FU, Y.; BAI, L.; ZHAO, S.; ZHANG, X.; JIN, Y.; CHENG, Y. Simulation of reactive mixing behaviors inside micro-droplets by a Lattice Boltzmann method. *Chemical Engineering Science*, v. 181, p. 79 – 89, 2018. ISSN 0009-2509. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0009250918300484>. Cited on page 103.

GEORGESCU, I. M.; ASHHAB, S.; NORI, F. Quantum simulation. *Rev. Mod. Phys.*, American Physical Society, v. 86, p. 153–185, Mar 2014. Available from Internet: <https://link.aps.org/doi/10.1103/RevModPhys.86.153>. Cited on page 20.

GINZBURG, I.; VERHAEGHE, F.; D'HUMIERES, D. Two-relaxation-time Lattice Boltzmann scheme: About parametrization, velocity, pressure and mixed boundary conditions. *Communications in computational physics*, Global science press, v. 3, n. 2, p. 427–478, 2008. Cited on page 34.

GRUNAU, D.; CHEN, S.; EGGERT, K. A Lattice Boltzmann model for multiphase fluid flows. *Physics of Fluids A: Fluid Dynamics*, v. 5, n. 10, p. 2557–2562, 1993. Available from Internet: <https://doi.org/10.1063/1.858769>. Cited on page 40.

GUAN, Y.; NOVOSSELOV, I. Two relaxation time Lattice Boltzmann method coupled to fast fourier transform poisson solver: Application to electroconvective flow. *Journal of Computational Physics*, v. 397, p. 108830, 2019. ISSN 0021-9991. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0021999119305145>. Cited on page 105.

GUNSTENSEN, A. K.; ROTHMAN, D. H.; ZALESKI, S.; ZANETTI, G. Lattice Boltzmann model of immiscible fluids. *Phys. Rev. A*, American Physical Society, v. 43, p. 4320–4327, Apr 1991. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevA.43.4320>. Cited on page 103.

GUO, L.; XIAO, L.; SHAN, X.; ZHANG, X. Modeling adsorption with Lattice Boltzmann equation. *Scientific Reports*, v. 6, n. 1, p. 27134, 2016. ISSN 2045-2322. Available from Internet: <https://doi.org/10.1038/srep27134>. Cited 5 times on pages 115, 116, 118, 119, and 120.

GUO, Z.; ZHAO, T. S. Lattice Boltzmann model for incompressible flows through porous media. *Phys. Rev. E*, American Physical Society, v. 66, p. 036304, Sep 2002. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevE.66.036304>. Cited on page 37.

GUO, Z.; ZHENG, C.; SHI, B. Discrete lattice effects on the forcing term in the Lattice Boltzmann method. *Phys. Rev. E*, American Physical Society, v. 65, p. 046308, Apr 2002. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevE.65.046308>. Cited on page 63.

HE, X.; LUO, L.-S. Theory of the Lattice Boltzmann method: From the Boltzmann equation to the Lattice Boltzmann equation. *Phys. Rev. E*, American Physical Society, v. 56, p. 6811–6817, Dec 1997. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevE.56.6811>. Cited on page 49.

HIRSCHFELDER, J. O.; CURTISS, C. F.; BIRD, R. B.; MAYER, M. G. *Molecular theory of gases and liquids*. [S.l.]: Wiley New York, 1954. v. 26. Cited on page 24.

HO, C.-F.; CHANG, C.; LIN, K.-H.; LIN, C.-A. Consistent boundary conditions for 2d and 3d Lattice Boltzmann simulations. *CMES: Computer Modeling in Engineering & Sciences*, v. 44, n. 2, p. 137–156, 2009. Available from Internet: <http://dx.doi.org/10.3970/cmes.2009.044.137>. Cited 5 times on pages 40, 46, 68, 85, and 90.

HOU, Y.; DENG, H.; DU, Q.; JIAO, K. Multi-component multi-phase Lattice Boltzmann modeling of droplet coalescence in flow channel of fuel cell. *Journal of Power Sources*, v. 393, p. 83 – 91, 2018. ISSN 0378-7753. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0378775318304695>. Cited on page 105.

HOU, Y.; DENG, H.; PAN, F.; CHEN, W.; DU, Q.; JIAO, K. Pore-scale investigation of catalyst layer ingredient and structure effect in proton exchange membrane fuel cell. *Applied Energy*, v. 253, p. 113561, 2019. ISSN 0306-2619. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0306261919312358>. Cited on page 105.

INAMURO, T.; YOSHINO, M.; OGINO, F. A non-slip boundary condition for Lattice Boltzmann simulations. *Physics of Fluids*, v. 7, n. 12, p. 2928–2930, 1995. Available from Internet: <https://doi.org/10.1063/1.868766>. Cited on page 40.

KADLEC, O. The history and present state of dubinin's theory of adsorption of vapours and gases on microporous solids. *Adsorption Science & Technology*, v. 19, n. 1, p. 1–24, 2001. Available from Internet: <https://doi.org/10.1260/0263617011493944>. Cited on page 117.

KIM, S. H.; PITSCH, H. A generalized periodic boundary condition for Lattice Boltzmann method simulation of a pressure driven flow in a periodic geometry. *Physics of Fluids*, v. 19, n. 10, p. 108101, 2007. Available from Internet: <https://doi.org/10.1063/1.2780194>. Cited on page 44.

KRÜGER, T.; KUSUMAATMAJA, H.; KUZMIN, A.; SHARDT, O.; SILVA, G.; VIGGEN, E. M. *The Lattice Boltzmann Method: Principles and Practice*. 1. ed. [S.l.]: Springer International Publishing, 2017. (Graduate Texts in Physics). ISBN 9783319446493. Cited 18 times on pages 13, 14, 19, 30, 31, 33, 36, 37, 38, 45, 48, 49, 62, 64, 66, 67, 103, and 104.

KUPERSHTOKH, A. L. New method of incorporating a body force term into the Lattice Boltzmann equation. In: *Proc. 5th International EHD Workshop, University of Poitiers, Poitiers, France*. [S.l.: s.n.], 2004. p. 241–246. Cited on page 64.

LADD, A. J. C.; VERBERG, R. Lattice-Boltzmann simulations of particle-fluid suspensions. *Journal of Statistical Physics*, v. 104, n. 5, p. 1191–1251, Sep 2001. ISSN 1572-9613. Available from Internet: <https://doi.org/10.1023/A:1010414013942>. Cited on page 37.

LATT, J.; CHOPARD, B.; MALASPINAS, O.; DEVILLE, M.; MICHLER, A. Straight velocity boundaries in the Lattice Boltzmann method. *Phys. Rev. E*, American Physical Society, v. 77, p. 056703, May 2008. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevE.77.056703>. Cited on page 40.

LOUIS, L.; BAUD, P.; WONG, T.-F. Characterization of pore-space heterogeneity in sandstone by x-ray computed tomography. *Geological Society, London, Special Publications*, Geological Society of London, v. 284, n. 1, p. 127–146, 2007. ISSN 0305-8719. Available from Internet: <https://sp.lyellcollection.org/content/284/1/127>. Cited on page 97.

LUO, Z.; XU, H. Numerical simulation of heat and mass transfer through microporous media with Lattice Boltzmann method. *Thermal Science and Engineering Progress*, v. 9, p. 44 – 51, 2019. ISSN 2451-9049. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S2451904918302282>. Cited 2 times on pages 34 and 97.

MAIER, R. S.; BERNARD, R. S.; GRUNAU, D. W. Boundary conditions for the Lattice Boltzmann method. *Physics of Fluids*, v. 8, n. 7, p. 1788–1801, 1996. Available from Internet: <https://doi.org/10.1063/1.868961>. Cited on page 40.

MISZTAL, M. K.; HERNANDEZ-GARCIA, A.; MATIN, R.; SøRENSEN, H. O.; MATHIESEN, J. Detailed analysis of the Lattice Boltzmann method on unstructured grids. *Journal of Computational Physics*, v. 297, p. 316 – 339, 2015. ISSN

0021-9991. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0021999115003538>. Cited 2 times on pages 31 and 53.

MOHAMAD, A. *Lattice Boltzmann Method: Fundamentals and Engineering Applications with Computer Codes*. [S.l.]: Springer London, 2011. (SpringerLink : Bücher). ISBN 9780857294555. Cited 5 times on pages 19, 21, 24, 26, and 36.

MUKHERJEE, S.; BERGHOUT, P.; AKKER, H. E. Van den. A Lattice Boltzmann approach to surfactant-laden emulsions. *AIChE Journal*, v. 65, n. 2, p. 811–828, 2019. Available from Internet: <https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.16451>. Cited on page 105.

NOURGALIEV, R.; DINH, T.; THEOFANOUS, T.; JOSEPH, D. The Lattice Boltzmann equation method: theoretical interpretation, numerics and implications. *International Journal of Multiphase Flow*, v. 29, n. 1, p. 117 – 169, 2003. ISSN 0301-9322. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0301932202001088>. Cited 2 times on pages 14 and 37.

PALPACELLI, S.; SUCCI, S. The quantum lattice Boltzmann equation: Recent developments. *Communications in Computational Physics*, v. 4, 11 2008. Cited on page 20.

PAN, C.; HILPERT, M.; MILLER, C. T. Lattice-Boltzmann simulation of two-phase flow in porous media. *Water Resources Research*, v. 40, n. 1, 2004. Available from Internet: <https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2003WR002120>. Cited on page 97.

PENG chen. *The Lattice Boltzmann method for fluid dynamics: theory and applications*. Dissertation (Masters) — Ecole Polytechnique Federale de Lausanne, 2011. Cited 5 times on pages 14, 21, 37, 68, and 90.

PENG, Z.; LIU, S.; TANG, S.; ZHAO, Y.; LI, Y. Multicomponent Lattice Boltzmann simulations of gas transport in a coal reservoir with dynamic adsorption. *Geofluids*, Hindawi, v. 2018, p. 13, 2018. Available from Internet: <https://doi.org/10.1155/2018/5169010>. Cited on page 97.

PEREIRA, G. Fluid flow, relative permeabilities and capillary pressure curves through heterogeneous porous media. *Applied Mathematical Modelling*, v. 75, p. 481 – 493, 2019. ISSN 0307-904X. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0307904X19303476>. Cited 2 times on pages 104 and 105.

PERUMAL, D. A.; DASS, A. K. Multiplicity of steady solutions in two-dimensional lid-driven cavity flows by Lattice Boltzmann method. *Computers & Mathematics with*

Applications, v. 61, n. 12, p. 3711–3721, 2011. ISSN 0898-1221. Mesoscopic Methods for Engineering and Science — Proceedings of ICMMES-09. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0898122110002427>. Cited on page 90.

PETERS, A.; MELCHIONNA, S.; KAXIRAS, E.; LäTT, J.; SIRCAR, J.; BERNASCHI, M.; BISON, M.; SUCCI, S. Multiscale simulation of cardiovascular flows on the ibm bluegene/p: Full heart-circulation system at red-blood cell resolution. In: *SC '10: Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*. [S.l.: s.n.], 2010. p. 1–10. ISSN 2167-4337. Cited on page 14.

POLANYI, M. Section iii.—theories of the adsorption of gases. a general survey and some additional remarks. introductory paper to section iii. *Trans. Faraday Soc.*, The Royal Society of Chemistry, v. 28, p. 316–333, 1932. Available from Internet: <http://dx.doi.org/10.1039/TF9322800316>. Cited 3 times on pages 117, 118, and 124.

PONCE, M.; MUNGUíA, L.; ESPARZA, M.; KORNHAUSER, I.; ROJAS, F. On scrutinizing the classical polanyi adsorption potential theory for vapour uptake occurring in the mesopores of curved shapes. *Adsorption Science & Technology*, v. 29, n. 6, p. 585–594, 2011. Available from Internet: <https://doi.org/10.1260/0263-6174.29.6.585>. Cited 2 times on pages 117 and 118.

POONOOSAMY, J.; WESTERWALBESLOH, C.; DEISSMANN, G.; MAHROUS, M.; CURTI, E.; CHURAKOV, S. V.; KLINKENBERG, M.; KOHLHEYER, D.; LIERES, E. von; BOSBACH, D.; PRASIANAKIS, N. I. A microfluidic experiment and pore scale modelling diagnostics for assessing mineral precipitation and dissolution in confined spaces. *Chemical Geology*, v. 528, p. 119264, 2019. ISSN 0009-2541. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0009254119303651>. Cited on page 15.

PORTINARI, M. *2D and 3D Verification an Validation of the Lattice Boltzmann Method*. Dissertation (Masters) — École Polytechnique de Montréal, 2015. Available from Internet: <https://publications.polymtl.ca/1927/>. Cited 3 times on pages 68, 85, and 90.

QIU, Z.; MA, Q.; ZHANG, Y.; YI, Y. Multiple-relaxation-time Lattice Boltzmann simulation for adsorption processes of carbon dioxide in a fixed-bed. *Engineering Computations*, Emerald Publishing Limited, v. 36, n. 3, p. 1021–1035, 2019. ISSN 0264-4401. Available from Internet: <https://doi.org/10.1108/EC-03-2018-0150>. Cited 2 times on pages 34 and 97.

REN, J.; ZHENG, Q.; GUO, P.; PENG, S.; WANG, Z.; DU, J. Pore-scale Lattice Boltzmann simulation of two-component shale gas flow. *Journal of Natural Gas Science*

and Engineering, v. 61, p. 46 – 70, 2019. ISSN 1875-5100. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S1875510018305043>. Cited on page 104.

ROSSI, N.; UBERTINI, S.; BELLA, G.; SUCCI, S. Unstructured Lattice Boltzmann method in three dimensions. *International Journal for Numerical Methods in Fluids*, v. 49, n. 6, p. 619–633, 2005. Available from Internet: <https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.1018>. Cited 2 times on pages 31 and 53.

SANDOVAL, D. W. L. *Revisiting grid refinement algorithms for the Lattice Boltzmann method*. Thesis (Doctoral) — Université de Genève, 12/18 2012. ID: unige:26414. Available from Internet: <https://nbn-resolving.org/urn:nbn:ch:unige-264141>. Cited 2 times on pages 31 and 53.

SEGA, M.; SBRAGAGLIA, M.; KANTOROVICH, S. S.; IVANOV, A. O. Mesoscale structures at complex fluid–fluid interfaces: a novel lattice Boltzmann/molecular dynamics coupling. *Soft Matter*, The Royal Society of Chemistry, v. 9, p. 10092–10107, 2013. Available from Internet: <http://dx.doi.org/10.1039/C3SM51556G>. Cited on page 108.

SHAN, X.; CHEN, H. Lattice Boltzmann model for simulating flows with multiple phases and components. *Phys. Rev. E*, American Physical Society, v. 47, p. 1815–1819, Mar 1993. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevE.47.1815>. Cited 5 times on pages 37, 48, 64, 103, and 105.

SHAN, X.; YUAN, X.-F.; CHEN, H. Kinetic theory representation of hydrodynamics: a way beyond the navier–stokes equation. *Journal of Fluid Mechanics*, Cambridge University Press, v. 550, p. 413–441, 2006. Cited on page 49.

SHANKAR, P. N.; DESHPANDE, M. D. Fluid mechanics in the driven cavity. *Annual Review of Fluid Mechanics*, v. 32, n. 1, p. 93–136, 2000. Available from Internet: <https://doi.org/10.1146/annurev.fluid.32.1.93>. Cited on page 90.

SHARMA, K. V.; STRAKA, R.; TAVARES, F. W. Lattice Boltzmann methods for industrial applications. *Industrial & Engineering Chemistry Research*, v. 58, n. 36, p. 16205–16234, 2019. Available from Internet: <https://doi.org/10.1021/acs.iecr.9b02008>. Cited 2 times on pages 14 and 115.

SHI, Y.; CHEN, G.; WANG, Q.; CHEN, Q. Simulation on falling film absorption based on Lattice Boltzmann method at moderate reynolds number. *International Journal of Heat and Mass Transfer*, v. 128, p. 991 – 998, 2019. ISSN 0017-9310. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0017931017353474>. Cited on page 115.

SHOUGUANG, Y.; LUOBIN, D.; KAI, Z.; JIANGBANG, Z.; ZHESHU, M.; XINWANG, J. Simulation of phase transition process in reconstructed porous medium based on Lattice Boltzmann method. *Thermal Science*, v. 23, n. 1, p. 169 – 177, 2019. ISSN 03549836. Available from Internet: <http://search.ebscohost.com/login.aspx?direct=true&db=asx&AN=135202959&lang=pt-br&site=eds-live&scope=site>. Cited on page 97.

SKORDOS, P. A. Initial and boundary conditions for the Lattice Boltzmann method. *Phys. Rev. E*, American Physical Society, v. 48, p. 4823–4842, Dec 1993. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevE.48.4823>. Cited on page 40.

SUCCI, S. *The Lattice Boltzmann Equation: For Fluid Dynamics and Beyond*. [S.l.]: Oxford university press, 2001. (Numerical Mathematics and Scientific Computation). ISBN 9780198503989. Cited 2 times on pages 14 and 37.

SWIFT, M. R.; ORLANDINI, E.; OSBORN, W. R.; YEOMANS, J. M. Lattice Boltzmann simulations of liquid-gas and binary fluid systems. *Phys. Rev. E*, American Physical Society, v. 54, p. 5041–5052, Nov 1996. Available from Internet: <https://link.aps.org/doi/10.1103/PhysRevE.54.5041>. Cited 3 times on pages 103, 104, and 109.

TAO, S.; HE, Q.; CHEN, J.; CHEN, B.; YANG, G.; WU, Z. A non-iterative immersed boundary-Lattice Boltzmann method with boundary condition enforced for fluid–solid flows. *Applied Mathematical Modelling*, v. 76, p. 362 – 379, 2019. ISSN 0307-904X. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0307904X19303889>. Cited 2 times on pages 34 and 103.

UBERTINI, S.; SUCCI, S. Recent advances of Lattice Boltzmann techniques on unstructured grids. *Progress in Computational Fluid Dynamics, an International Journal*, v. 5, n. 1-2, p. 85–96, 2005. Available from Internet: <https://www.inderscienceonline.com/doi/abs/10.1504/PCFD.2005.005820>. Cited 2 times on pages 31 and 53.

Van Dyke, M. *An album of fluid motion*. Parabolic Press, 1982. (An Album of Fluid Motion). Available from Internet: <https://books.google.com.br/books?id=rWoeAQAAIAAJ>. Cited on page 87.

WANG, Y.; SHU, C.; HUANG, H.; TEO, C. Multiphase Lattice Boltzmann flux solver for incompressible multiphase flows with large density ratio. *Journal of Computational Physics*, v. 280, p. 404 – 423, 2015. ISSN 0021-9991. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0021999114006718>. Cited on page 15.

WEI, B.; HOU, J.; SUKOP, M. C.; LIU, H. Pore scale study of amphiphilic fluids flow using the Lattice Boltzmann model. *International Journal of Heat and Mass Transfer*,

v. 139, p. 725 – 735, 2019. ISSN 0017-9310. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0017931018346957>. Cited on page 105.

WEI, B.; HOU, J.; WU, D.; WANG, H.; LIU, H. *et al.* Pore scale simulation of surfactant flooding by Lattice Boltzmann method. In: SOCIETY OF PETROLEUM ENGINEERS. *SPE International Heavy Oil Conference and Exhibition*. [S.l.], 2018. Cited on page 105.

WEN, B.; QIU, W.; SHAN, X. Chemical-potential multiphase Lattice Boltzmann method with superlarge density ratios. *arXiv e-prints*, p. arXiv:1906.09530, Jun 2019. Cited on page 104.

YU, H.; CHEN, J.; ZHU, Y.; WANG, F.; WU, H. Multiscale transport mechanism of shale gas in micro/nano-pores. *International Journal of Heat and Mass Transfer*, v. 111, p. 1172 – 1180, 2017. ISSN 0017-9310. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0017931017302326>. Cited on page 104.

ZHANG, C.; FAKHARI, A.; LI, J.; LUO, L.-S.; QIAN, T. A comparative study of interface-conforming ale-fe scheme and diffuse interface amr-lb scheme for interfacial dynamics. *Journal of Computational Physics*, v. 395, p. 602 – 619, 2019. ISSN 0021-9991. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0021999119304565>. Cited on page 103.

ZHANG, J.; JOHNSON, P. C.; POPEL, A. S. Red blood cell aggregation and dissociation in shear flows simulated by Lattice Boltzmann method. *Journal of Biomechanics*, v. 41, n. 1, p. 47 – 55, 2008. ISSN 0021-9290. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0021929007003363>. Cited on page 14.

ZHOU, L.; QU, Z.; CHEN, L.; TAO, W. Lattice Boltzmann simulation of gas–solid adsorption processes at pore scale level. *Journal of Computational Physics*, v. 300, p. 800 – 813, 2015. ISSN 0021-9991. Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0021999115005331>. Cited on page 115.

ZOU, Q.; HE, X. On pressure and velocity boundary conditions for the Lattice Boltzmann bgk model. *Physics of Fluids*, v. 9, n. 6, p. 1591–1598, 1997. Available from Internet: <https://doi.org/10.1063/1.869307>. Cited 2 times on pages 40 and 46.

ZOU, Q.; HOU, S.; CHEN, S.; DOOLEN, G. D. A improved incompressible Lattice Boltzmann model for time-independent flows. *Journal of Statistical Physics*, v. 81, n. 1, p. 35–48, Oct 1995. ISSN 1572-9613. Available from Internet: <https://doi.org/10.1007/BF02179966>. Cited 4 times on pages 37, 62, 69, and 113.

Appendix

APPENDIX A – LBM Code

The ensuing lines represent the main part of the LBM code. The code uses auxiliary files *input.txt* and *functions.py*. File *input.txt* contains the information and input necessary to the code, including selecting the mode, defining the size of the grid, and the magnitude of forces and pressures. Grids can be implemented using a *grid.xyz* file. The most important auxiliary functions used in the code can be found in Appendix B. Unnecessary parts of the program related to plotting the results are omitted. The output is written in an external *output.txt* file.

```
######################################################################
#                                                                    #
#      LBM code createded by Lucas Giuliano Murdiga de Moraes        #
#              State Univeristy of Campinas (UNICAMP)                #
#                                                                    #
######################################################################

#SET TIMER###########################################################
import time
init = time.clock()
print('\n***RUNNING***\n\n')
######################################################################

#IMPORT FUNCTIONS####################################################
import functions as f
#Python libraries---------------------------------------------------#
import os
import shutil
import numpy as np
from matplotlib import pyplot as plt
######################################################################
```

```
#CHECK FOR INPUT FILE################################################
cwdpth = os.getcwd()

#Check if inputfile.txt exists in folder
if os.path.exists("./input.txt"):
    pass
elif os.path.exists("../input.txt"):
    os.chdir('..')
else:
    if input('***ERROR***\ninputfile.txt not found\n'\
             'provide folder path? [Y/n]') in {'Y','y','1',True}:
        os.chdir(input('Path:'))
    else:
        exit('process finished - no input file')

inputfile = open("input.txt","r")
outfile = open("output.txt","w")

print('currentdirectory:'+os.getcwd()+'\n')
#####################################################################

#READ EXTERNAL FILES TO SELECT MODE FOR THE PROGRAM TO RUN##########
print("reading files\n")

#All modes available are here
modelist = {'ps':'poiseuille',
            'ct':'couette',
            'ldc':'lid-driven cavity',
            'file':'grid from file',
            'bx':'testing box',
            'psf':'poiseuille driven by body force',
            'ads':'adsorption'}
```

```
#Grid control variables-----------------------------------------#
gr    = False

#Mode control variable. Accepted values are [ps|ct|ldc|file|define]
inputfile.readline()
mode = f.read(inputfile,1)

if mode not in modelist:
    print('mode not defined: using standard poiseuille')
    mode = 'ps'
elif mode == 'file':
    if os.path.exists("./grid.xyz"):
        gridfile = open("grid.xyz","r")
        gr = True
    else:
        print('no grid file found: using standard poiseuille')
        mode = 'ps'
else:
    print('using preset: %s' %modelist[mode])
#####################################################################

#GETING THE GENERAL VARIABLES FROM FILES#############################
print("importing set conditions")

#define path to save images and create necessary folder
if os.path.exists("Results") == False:
os.mkdir("Results/")

#set path and folders-------------------------------------------#
#set a variable to define if a new folder is to be created
nf == True
```

```
#creating a new sequantial foder
if nf == True:
    num2 = 0
    while os.path.exists\
                    ("Results/images"+str(int(num2)).zfill(5)+"/"):
        num2 += 1
    path = "Results/images"+str(int(num2)).zfill(5)+"/"

    if os.path.exists(path) == False:
    os.mkdir(path)

#if not creating a new folder the results will be saved in cwd
else:
    num = 0
    path = ""

print('saving in: '+path)

#other general variables-----------------------------------------#
tstep = bool(int(f.read(inputfile,1)))
tstamp = bool(int(f.read(inputfile,1)))
init = bool(int(f.read(inputfile,1)))

#read separation line
inputfile.readline()

#get geometry----------------------------------------------------#
man = bool(int(f.read(inputfile,1)))

if man == 0:
    c = f.read(inputfile,1,2)
    b = f.read(inputfile,1,2)
    l = f.read(inputfile,1,2)
```

```
    r = f.read(inputfile,1,2)
#manual input
else:
    inputfile.readline()
    inputfile.readline()
    inputfile.readline()
    inputfile.readline()
    c = input('top wall: ')
    b = input('bottom wall: ')
    l = input('left wall: ')
    r = input('right wall: ')

#when using the testing box mode
if (mode == 'bx'):
c = [1]
b = [1]
l = [1]
r = [1]

#forcing term-----------------------------------------------------#
#changing uf and DF sobrescribe Fx
uf = 0
DF = 0
Fx=Fy=0
force = f.read(inputfile,1,2)
if force[0] == False:
    pass

if force[0] == 'umax':
    uf = float(force[1])
elif force[0] == 'DP':
    DF = float(force[1])
else:
```

```
    Fx = float(force[0])
    Fy = float(force[1])

inputfile.readline()

#other general variables-----------------------------------------#
tmax = float(f.read(inputfile,1))
tcheck = float(f.read(inputfile,1))
tol = float(f.read(inputfile,1))
scale = int(f.read(inputfile,1))

nx = int(f.read(inputfile,1))
NX = nx
ny = int(f.read(inputfile,1))
NY = ny

if c == -1:
ny = ny+2
if l == -1:
nx = nx+2

tau = float(f.read(inputfile,1))
if tau==0:
    tau = (3/16)**(1/2)+0.5
    itau = 1/tau

inputfile.readline()

#see if there is a cylinder
cylinder = bool(int(f.read(inputfile,1)))
#################################################################

#CREATE GRID#####################################################
```

```
print("Creating grid")
#from file
if gr:
    grid = f.gridf(gridfile)
    nx,ny = grid.shape
    grid = 1 - grid

#based on the geometry input
else:
    grid = f.setgrid(nx,ny,cylinder,inputfile,c,b,l,r,)
#####################################################################

#INITIALIZE THE VECTORS AND VARIABLES#################################
print("Initializing")
q = 9
t = 0
flag = 0

#Macroscopic variables vectors
rho = np.zeros((nx,ny))
ux = np.zeros((nx,ny))
uy = np.zeros((nx,ny))
u = np.zeros((nx,ny), dtype='longdouble')

rho = rho + 1

#Distribuction vectors
fxy = np.zeros((nx,ny,q))
feq = np.zeros((nx,ny,q))

#Auxiliar variables and vectors
w  = [4/9,1/9,1/9,1/9,1/9,1/36,1/36,1/36,1/36]
cx = [  0,  1,  0, -1,  0,   1,  -1,  -1,   1]
```

```
cy = [  0,  0,  1,  0, -1,   1,   1,  -1,  -1]

#Define position of the nodes
x = np.zeros(nx)
y = np.zeros(ny)
for i in range(nx):
    x[i]=(i)/scale
for i in range(ny):
    y[i]=(i)/scale

#Initialize the velocities from file
if init:
    rho,ux,uy = f.init(rho,ux,uy,nx,ny)

#Initializing equilibrium and velocities
u[:,:] = (ux[:,:]*ux[:,:] + uy[:,:]*uy[:,:])**0.5
feq = f.equilibrium(feq,q,rho,ux,uy,w,cx,cy)
fxy = feq
t += 1

umax=0.0
rho_in=rho_out=1
rho_top=rho_bot=1

#Initialize contour conditions
for a in (c,b,l,r):
    if a[0] == 'umax':
        umax = float(a[1])
        if umax == 0:
            umax = float(input('umax: '))
        rho_in,rho_out,DP = f.vn(umax,tau,scale,nx,ny)
    elif a[0] == 'DP'
        DP = float(a[1])
```

```
        if DP = 0:
            umax = float(input('DP: '))
        rho_in = rho_out + 3*DP*(nx-1)

#Initialize population for error
e = u.copy()

#Output of initial macroscopic conditions
if tstep:
    outfile.write('0\n')
    for i in range (nx):
        for j in range (ny):
            outfile.write('%d %d %0.5f %0.5f %0.5f %0.5f\n'\
                      %(i,j,rho[i,j],u[i,j],ux[i,j],uy[i,j]))
#####################################################################

#Stablishing the correct contour conditions for each case###########

#start with normal bounce-back (testing box)
top = f.bb_yt
bottom = f.bb_yb
left = f.bb_xe
right = f.bb_xd
corner = f.corner

#special corners for adsorption
if mode == 'ads':
    corner = f.corner2

#changes for Poiseuille and Couette
if (mode == 'ps' or mode == 'ct'):
    left = f.zouhe_e
    right = f. zouhe_d
```

```
    print('left and right: zouhe for pressure')
    if mode == 'ct':
        top = f.zh_top
        print('top: zouhe for u')
        rho_in=rho_out

#lid-driven cavity
if mode == 'ldc':
    top = f.zh_top
    rho_in = rho_out
    ('top: zouhe for u')

if (mode == 'psf' or mode == 'ads'):
    left = f.fvn
    right = f.null
    print('left and right: virtual nodes')

if mode == 'ads':
    bottom = f.abb_bottom
    print('bottom: defined density - abb method')
    print(inputfile.readline())
    rho_bot=float(f.read(inputfile,1))

#read remaining lines in input file in case no cylinder is present
if cylinder == 0:
    inputfile.readline()
    inputfile.readline()
    inputfile.readline()
    inputfile.readline()
    inputfile.readline()
#####################################################################

#FORCE TERM##########################################################
```

```
#phi gives dependence of the force with distance to the wall
#for gravity phi = 1 (equal in all domain)
#phi is dependent of distance according to r^n
phi = np.zeros(ny)

if (mode == 'ads'):
    ads = f.read(inputfile,1,2)
    ads = bool(int(ads[0])),int(ads[1])

if mode != 'ads':
    ads = [1,0]

if ads[0] == False:
    phi[-1] = 1/((ny-i)**int(ads[1]))
    else:
        for i in range(ny):
            phi[i] = 1/((ny-i)**int(ads[1]))


if mode == 'psf':
#setting horizontal force equivalent to a pressure drop or max vel
#change uf to add force equivalent to max velocity
if uf != 0:
    rho_inf,rho_outf,DF = f.vn(uf,tau,scale,nx,ny)

#change DF directly to add force equivalent to a pressure drop
if DF != 0:
    Fx = DF

F = [Fx,Fy]
print('Fx,Fy: ',Fx,Fy)
```

```
Fi = np.zeros((nx,ny,q))

#psi is the pseudo-potential, change for multicomponent flow
psi = 1

for i in range(nx):
    for j in range(ny):
        for k in range(q):
            Fi[i,j,k]=w[k]*3*(cx[k]*Fx+cy[k]*Fy)*phi[j]*psi

S = (1-0.5*itau)*Fi
#####################################################################

#MAIN LOOP###########################################################
print("Calculating")

chi = 0
e = rho.copy()

print('top:     ',top)
print('bottom:  ',bottom)
print('left:    ',left)
print('right:   ',right)
print('corners: ',corner)

while flag == 0 and t <= tmax:

    #collision and propagation
    S = (1-0.5*itau)*Fi
    f0 = fxy.copy()
    fxy = f.colision(itau,fxy,feq,S)
    fxy = f.propagation(fxy,nx,ny,q)
```

```
    #boundary conditions
    fxy,uy = top  (fxy,nx,-1,uy,rho,rho_top,rho_bot,umax,Fx,Fy,f0,w)
    fxy,uy = bottom(fxy,nx,0,uy,rho,rho_top,rho_bot,umax,Fx,Fy,f0,w)
    fxy,ux = left (fxy,ny, 0,ux,rho, rho_in,rho_out,umax,Fx,Fy, 0,0)
    fxy,ux = right(fxy,ny,-1,ux,rho, rho_in,rho_out,umax,Fx,Fy, 0,0)
    fxy = corner(mode,fxy,nx,ny,rho_in,rho_out,umax,rho)

    #bounce-back within the domain
    fxy = f.bb(fxy,nx,ny,grid)

    #macroscopic variables
    rho,ux,uy,Fi = f.update(fxy,q,nx,ny,Fi,Fx,Fy,w,cx,cy,phi)
    u = (ux*ux + uy*uy)**0.5

    #equilibrium distribution
    feq = f.equilibrium(feq,q,rho,ux,uy,w,cx,cy,u)

    #checking stationary state
    if t%tcheck == 0:
    flag,e,err = f.check(u,e,tol,nx,ny,rho,chi)

#output of macroscopic variables and images within loop if necessary

    #print information as the code runs
    if t%2000 == 0:
    print('Calculating')
    print('error: %e' %err)

    #update time variable
    t += 1

#downgrade time variable to compensate for the last upgrade
t -= 1
```

```
#####################################################################

#ENDING
#####################################################################
print("Shutting Down")

inputfile.close()

outfile.write('t %d' %t)
outfile.write('%0.2f\n' %tau)
for i in range(nx):
    for j in range(ny):
        outfile.write('%d %d %0.5f %0.5f %0.5f %0.5f\n'\
                      %(i, j, rho[i,j], u[i,j], ux[i,j], uy[i,j]))
shutil.copy('output.txt', path+'output.txt')

if gr:
gridfile.close()

print('process finished in %d seconds' %(time.clock()-init))
outfile.write('process finished in %d seconds' %(time.clock()-init))
outfile.close()
```

APPENDIX B – Auxiliary Functions

```
#######################################################################
#                                                                     #
#      LBM code createded by Lucas Giuliano Murdiga de Moraes         #
#              State Univeristy of Campinas (UNICAMP)                 #
#                  ***auxiliary functions file***                     #
#                                                                     #
#######################################################################

import numpy as np
from numba import jit
from math import exp

#AUXILIARY############################################################

@jit
def read(file, position, pos2=None):
    read = file.readline()
    read = read.split()
    var = read[position]
    if pos2 and pos2 < len(read):
        var2=read[pos2]
        return var,var2
    return var

@jit
def setgrid(nx, ny, flag, inputfile, t=[0], b=[0], l=[0], r=[0]):
    grid = np.zeros((nx, ny))
    grid[0, :] = 1 if l[0]==1 else 0
    grid[-1, :] = 1 if r[0]==1 else 0
```

```
    grid[:, 0] = 1 if b[0]==1 else 0
    grid[:, -1] = 1 if t[0]==1 else 0
    if flag:
        grid = cYlinder(grid, nx, ny, inputfile)
    return grid

@jit
def gridf(file, nx=0, ny=0, sq=1):
    #use total number of points in square grid if no nx and ny
    tot = int(file.readline())
    if sq:
        nx = int(tot ** (0.5))
        ny = int(nx)
    grid = np.zeros((nx, ny))
    for i in range(nx):
        for j in range(ny):
            grid[i, j] = read(file, -1)
    return grid

@jit
def cylinder(grid, nx, ny, inputfile):
    aux = read(inputfile, 1)
    if aux == 0:
        px = float(read(inputfile, 1))
        py = float(read(inputfile, 1))
        rad = int(read(inputfile, 1))
    else:
        px = float(nx / float(read(inputfile, 1)))-0.5
        py = float(ny / float(read(inputfile, 1)))-0.5
        if aux == 'y':
            rad = ny / float(read(inputfile, 1))
        elif aux == 'x':
            rad = nx / float(read(inputfile, 1))
```

```
        else:
            print('error: cylinder variables not defined')
            return grid
    obst = grid.copy()
    for i in range(nx):
        for j in range(ny):
            obst[i, j] = (i-px)*(i-px) + (j-py)*(j-py) <= rad*rad
    return grid + obst

@jit
def vn(u, tau, scale, nx, ny):
    umax = u / scale
    visc = (2 * tau - 1) / 6
    Re = (ny-1) * umax / visc
    DP = (8 * visc * umax) / ((ny-1) ** 2)
    rho_out = 1
    rho_in = rho_out + 3 * DP * (nx-1)
    return rho_in, rho_out, DP

#MAIN###########################################################

@jit
def equilibrium(feq, q, rho, ux, uy, w, cx, cy, u):
    for k in range(q):
        feq[:,:,k] = w[k]*(rho[:,:]+3*(ux[:,:]*cx[k]+uy[:,:]*cy[k])\
                     +(9/2)*(ux[:,:]*cx[k]+uy[:,:]*cy[k])**2\
                     -(3/2)*u[:,:]**2)
    return feq

@jit
def colision(itau, f, feq, S):
    f0 = f.copy()
    theta = 1 - itau
```

```
    f0[:,:] = theta*f[:,:] + itau*feq[:,:] + S
    return f0

@jit
def propagation(f, nx, ny, q, t=0,b=0,l=0,r=0):
    f0 = f.copy()
    xmin=0
    ymin=0
    xmax=nx-1
    ymax=ny-1
    for x in range(nx):
        for y in range(ny):
            a=1
            b=1
            c=1
            d=1
            if x == xmin:
                a=0
            elif x == xmax:
                b=0
            if y == ymin:
                c=0
            elif y == ymax:
                d=0
            f0[x, y, 1] = f[x-a, y, 1]
            f0[x, y, 3] = f[x+b, y, 3]
            f0[x, y, 2] = f[x, y-c, 2]
            f0[x, y, 4] = f[x, y+d, 4]
            f0[x, y, 5] = f[x-a, y-c, 5]
            f0[x, y, 6] = f[x+b, y-c, 6]
            f0[x, y, 7] = f[x+b, y+d, 7]
            f0[x, y, 8] = f[x-a, y+d, 8]
    return f0
```

```
@jit
def bb_yt(f,nx,j,uy,rho,b,c,d,e,k,g,h):
    for i in range(1,nx-1):
        f[i, j, 4] = f[i, j, 2]
        f[i, j, 7] = f[i, j, 5]
        f[i, j, 8] = f[i, j, 6]
    return f,uy

@jit
def bb_yb(f,nx,j,uy,rho,b,c,d,e,k,g,h):
    for i in range(1,nx-1):
        f[i, j, 2] = f[i, j, 4]
        f[i, j, 5] = f[i, j, 7]
        f[i, j, 6] = f[i, j, 8]
    return f,uy

@jit
def bb_xe(f,ny,i,ux,a,b,c,d,e,k,g,h):
    for j in range(1, ny - 1):
        f[i, j, 1] = f[i, j, 3]
        f[i, j, 5] = f[i, j, 7]
        f[i, j, 8] = f[i, j, 6]
    return f,ux

@jit
def bb_xd(f,ny,i,ux,a,b,c,d,e,k,g,h):
    for j in range(1, ny - 1):
        f[i, j, 3] = f[i, j, 1]
        f[i, j, 7] = f[i, j, 5]
        f[i, j, 6] = f[i, j, 8]
    return f,ux
```

```
@jit
def update(f, q, nx, ny, F, Fx, Fy, w, cx, cy, s):
    rho = np.zeros((nx,ny))
    ux = rho.copy()
    uy = rho.copy()
    for k in range(q):
        rho[:,:] = rho[:,:] + f[:,:,k]
        r = rho.copy()
    #Shan-Chen pseudo-potential
    psi = 1 - np.exp(-np.abs(rho))
    for i in range(nx):
        for j in range(ny):
            for k in range(q):
                F[i,j,k]=w[k]*3*(cx[k]*Fx+cy[k]*Fy)*s[j]*psi[i,j]
#MAIN###########################################################
    ux[:,:]=(np.sum(f[:,:,[1,5,8]],axis=2)-\
            np.sum(f[:,:,[3,6,7]],axis=2))+\
            (np.sum(F[:,:,[1,5,8]],axis=2)-\
            np.sum(F[:,:,[3,6,7]],axis=2))*0.5
    uy[:,:]=(np.sum(f[:,:,[2,5,6]],axis=2)-\
            np.sum(f[:,:,[4,7,8]],axis=2))+\
            (np.sum(F[:,:,[2,5,6]],axis=2)-\
            np.sum(F[:,:,[4,7,8]],axis=2))*0.5
return rho, ux, uy, F

@jit
def check(u, e, tol, nx, ny, rho, chi):
    var = 0
    if chi == 1:
        a = rho
    else:
        a = u
    for i in range(nx):
```

```
        for j in range(ny):
            var += abs(a[i, j] - e[i, j])
    if var < tol:
        print('final error: %e' %var)
        return 1, e, var
    e = u.copy()
    return 0, e, var

@jit
def zouhe_e(f,ny,x,ux,a,rho,b,c,d,e,g,h):
    r = 1/rho
    for i in range(1,ny-1):
        ux[x,i] = 1-(f[x,i,0]+f[x,i,2]+f[x,i,4]+\
                  2*(f[x,i,3]+f[x,i,6]+f[x,i,7]))*r
        f[x,i,1]=f[x,i,3]+(2/3)*rho*ux[x,i]
        f[x,i,5]=f[x,i,7]-0.5*(f[x,i,2]-f[x,i,4])+(1/6)*rho*ux[x,i]
        f[x,i,8]=f[x,i,6]+0.5*(f[x,i,2]-f[x,i,4])+(1/6)*rho*ux[x,i]
    return f,ux

@jit
def zouhe_d(f,ny,x,ux,a,b,rho,c,d,e,g,h):
    r = 1/rho
    for i in range(1,ny-1):
        ux[x,i] = (f[x,i,0]+f[x,i,2]+f[x,i,4]+\
                  2*(f[x,i,1]+f[x,i,5]+f[x,i,8]))*r-1
        f[x,i,3]=f[x,i,1]-(2/3)*rho*ux[x,i]
        f[x,i,7]=f[x,i,5]+0.5*(f[x,i,2]-f[x,i,4])-(1/6)*rho*ux[x,i]
        f[x,i,6]=f[x,i,8]-0.5*(f[x,i,2]-f[x,i,4])-(1/6)*rho*ux[x,i]
    return f,ux

@jit
def zh_top(f,nx,j,uy,a,b,c,u,d,e,g,h):
    for i in range(1,nx-1):
```

```
        f[i,j,1] = f[i,j,3] + (2/3)*u
        f[i,j,4] = f[i,j,2]
        f[i,j,7] = f[i,j,5] - (1/6)*u
        f[i,j,8] = f[i,j,6] + (1/6)*u
    return f,uy

@jit
def abb_bottom(f,nx,j,uy,a,rho_top,rho_bot,umax,Fx,Fy,f0,w):
    for i in range(1,nx-1):
        f[i, j, 2] = -f[i, j, 4] + 2*w[4]*rho_bot
        f[i, j, 5] = -f[i, j, 7] + 2*w[7]*rho_bot
        f[i, j, 6] = -f[i, j, 8] + 2*w[8]*rho_bot
    return f,uy

@jit
def fvn(fxy,a,b,ux,c,d,e,f,g,h,i,j):
    f0 = fxy.copy()
    f0[0 ,:,[1,5,8]] = fxy[-1,:,[1,5,8]]
    f0[-1,:,[3,6,7]] = fxy[0 ,:,[3,6,7]]
    return f0,ux

def null(fxy,a,b,ux,c,d,e,f,g,h,i,j):
    return fxy,ux
```

APPENDIX C – LBM Code II

The ensuing lines correspond to the altered main part of the program to include two distinct distribution functions to simulate binary fluid flows.

```
#MAIN LOOP###########################################################
print("Calculating")

while flag == 0 and t <= tmax:

    #force terms and collision-------------------------------------#
    S_f = (1-0.5*itau)*Fi_f
    f0 = fxy.copy()
    fxy = f.colision(itau,fxy,feq,S_f)

    S_g = (1-0.5*itau)*Fi_g
    g0 = gxy.copy()
    gxy = f.colision(itau,gxy,geq,S_g)

    fxy = f.propagation(fxy,nx,ny,q)
    gxy = f.propagation(gxy,nx,ny,q)

    #boundaries----------------------------------------------------#
    fxy,uy=top   (fxy,nx,-1,uy,rh_f,rh_ft,rh_fb,umax,Fx_f,Fy_f,f0,w)
    fxy,uy=bottom(fxy,nx, 0,uy,rh_f,rh_ft,rh_fb,umax,Fx_f,Fy_f,f0,w)
    fxy,ux=left  (fxy,ny, 0,ux,rh_f,rh_fi,rh_fo,umax,Fx_f,Fy_f, 0,0)
    fxy,ux=right (fxy,ny,-1,ux,rh_f,rh_fi,rh_fo,umax,Fx_f,Fy_f, 0,0)
    fxy = corner (mode,fxy,nx,ny,rh_fi,rh_fo,umax,rho_f)
    fxy = f.bb(fxy,nx,ny,grid)

    gxy,uy=top   (gxy,nx,-1,uy,rh_g,rh_gt,rh_gb,umax,Fx_g,Fy_g,g0,w)
```

```
gxy,uy=bottom(gxy,nx, 0,uy,rh_g,rh_gt,rh_gb,umax,Fx_g,Fy_g,g0,w)
gxy,ux=left  (gxy,ny, 0,ux,rh_g,rh_gi,rh_go,umax,Fx_g,Fy_g, 0,0)
gxy,ux=right (gxy,ny,-1,ux,rh_g,rh_gi,rh_go,umax,Fx_g,Fy_g, 0,0)
gxy = corner (mode,gxy,nx,ny,rh_gi,rh_go,umax,rho_g)
gxy = f.bb(gxy,nx,ny,grid)

#upate velocities and desities---------------------------------#
rh_f,ux_f,uy_f,Hx_f,Hy_f =\
   f.update_ru(fxy,q,nx,ny,Fi_f,Fx_f,Fy_f,w,cx,cy,phi,ux_f,uy_f)
rh_g,ux_g,uy_g,Hx_g,Hy_g =\
   f.update_ru(gxy,q,nx,ny,Fi_g,Fx_g,Fy_g,w,cx,cy,phi,ux_g,uy_g)

rho = rh_f + rh_g

#check to avoid overflow error when dividing-------------------#
for i in range(nx):
    for j in range(ny):
        ux_f[i,j] = ux_f[i,j] if abs(ux_f[i,j]) > 1e-8 else 0
        uy_f[i,j] = uy_f[i,j] if abs(uy_f[i,j]) > 1e-8 else 0
        ux_g[i,j] = ux_g[i,j] if abs(ux_g[i,j]) > 1e-8 else 0
        uy_g[i,j] = uy_g[i,j] if abs(uy_g[i,j]) > 1e-8 else 0

#baricentric velocity------------------------------------------#
ux = (ux_f + ux_g)
uy = (uy_f + uy_g)
for i in range(nx):
    for j in range(ny):
        ux[i,j] = ux[i,j] if abs(ux[i,j]) > 1e-8 else 0
        uy[i,j] = uy[i,j] if abs(uy[i,j]) > 1e-8 else 0
        u[i,j] = (ux[i,j]**2 + uy[i,j]**2)**0.5

#update force terms with barycentric velocity------------------#
Fi_f = f.update_F(fxy,q,nx,ny,Fi_f,Hx_f,Hy_f,w,cx,cy,phi,ux,uy)
```

```
    Fi_g = f.update_F(gxy,q,nx,ny,Fi_g,Hx_g,Hy_g,w,cx,cy,phi,ux,uy)

    #division for the equilibrium distribution---------------------#
    for i in range(nx):
    for j in range(ny):
    xf[i,j]=rh_f[i,j]/rho[i,j]\
    if abs(rh_f[i,j]) > 1e-10 else 0
    xg[i,j]=rh_g[i,j]/rho[i,j]\
    if abs(rh_g[i,j]) > 1e-10 else 0

    #calculate equilibrium distributions--------------------------#
    geq = f.equilibrium2(geq,q,rho_g,ux,uy,w,cx,cy,xg)
    feq = f.equilibrium2(feq,q,rho_f,ux,uy,w,cx,cy,xf)

    #check for stationary state-----------------------------------#
    if t%tcheck == 0:
        flag,e,err = f.check(u,e,tol,nx,ny,rho,chi)

    if t%2000 == 0:
        print('Calculating')
        print('error: %e' %err)

    t += 1

t -= 1
```