Wind Flow Simulation around Buildings using the Lattice Boltzmann Method

Angel Agustin Cesar Peinado Bravo

Master-Thesis
Wind Flow Simulation around Buildings using the Lattice Boltzmann Method

Angel Agustin Cesar Peinado Bravo
Master-Thesis

Aufgabensteller: Prof. Dr. U. Rüde
Betreuer: M. Sc. Martin Bauer
Erklärung:

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Der Universität Erlangen-Nürnberg, vertreten durch den Lehrstuhl für Systemsimulation (Informatik 10), wird für Zwecke der Forschung und Lehre ein einfaches, kostenloses, zeitlich und örtlich unbeschränktes Nutzungsrecht an den Arbeitsergebnissen der Master-Thesis einschließlich etwaiger Schutzrechte und Urheberrechte eingeräumt.

ACKNOWLEDGEMENTS

First of all, I would like to convey my sincere gratitude to my supervisor Prof. Ulrich Rüde and the company Framatome for providing me the opportunity to work on this exciting and challenging topic of environmental flows using the lattice Boltzmann method with the aim to real-time simulations.

Furthermore, I would like to express my deepest gratitude to my advisor Martin Bauer for his continuous support and encouragement during the development of my master thesis. In particular, I would like to thanks his patience and enthusiasm for sharing this knowledge on the subject of lattice Boltzmann methods and lbmpy.

A special thanks go to Simon Grams for his support and for taking the time out from his busy schedule to provide his valuable feedback throughout my thesis work. In particular, the discussion with Dominik Geuss about the lattice Boltzmann method and its implementation in industrial scenarios.

I am particularly grateful to my family and friends, who have been with me during this time and encouraged and supported me in the right moments.
ABSTRACT

Author: Angel Agustin Cesar Peinado Bravo
Title: Wind Flow Simulation around Buildings using the Lattice Boltzmann Method
Institution: Friedrich-Alexander-Universität Erlangen-Nürnberg
Supervisor: Prof. Dr. U. Rüde
Advisor: M. Sc. Martin Bauer
Degree: Master of Science in Computational Engineering
Year: 2019

Real-time simulations of environmental flows of pollutants near cities and facilities are of great interest for national safety, governmental institution and companies. These institutions require the ability to predict the dispersion of toxic and dangerous pollutants near populated areas, not only to monitor emissions in diverse type of facilities but also in case of an accident to elaborate an optimal evacuation plan and reduce human and material costs. A clear example was the eruption of Eyjafjallajkull in 2010, which had a significant impact in the surrounding area (evacuation of people) and on airlines not only in Iceland but also in western and northern Europe.

The lattice Boltzmann method (LBM) has the potential to perform real-time simulations due to its highly parallelizable nature and its suitability to be implemented in GPU clusters and supercomputers. However, its original formulation, single relaxation-time, presented several numerical instabilities when the flow scales are not highly resolved; thus, impeding the simulation of highly turbulent flows in a real-time framework. To overcome this problem, several variations of the original LBM have been developed with great results, in particular, the sub-grid scale models.

In the present thesis, the coherent structure model (CSM) is explored and implemented in the lbmpy, a module of pystencils and extension of waLberla. Through various two- and three-dimensional benchmarks, the CSM is validated and compared with other LBMs, such as the entropic models (KBC) and the multi-relaxation time (MRT), and experimental data. At last, the different LBMs are verified for their application in environmental flows, through diverse benchmarks and an industrial
application with experimental and traditional CFD methods for comparison. Thus, a significant outcome of the thesis is an analysis of the different LBMs for highly turbulent flows with a particular interest in real-time simulations of environmental flows.
To my Mother and Father.

Of whose authentic gray hairs

I am one of the principal causes.

Cuando creíamos que teníamos todas las respuestas

de pronto cambiaron todas las preguntas

Mario Benedetti
WIND FLOW SIMULATION AROUND BUILDINGS USING THE LATTICE BOLTZMANN METHOD

List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 Introduction

1.1 Motivation

1.2 Outline of the Thesis

2 Lattice Boltzmann Method

2.1 Lattice Gas Automata

2.2 Lattice Boltzmann Method

2.3 LBM to Navier Stokes Equation

2.4 Lattice units to macroscopic scale

2.5 Accuracy and stability

2.6 Different LBMs

2.6.1 Multi-relaxation time (MRT)

2.6.2 Entropic multiple-relaxation time

2.6.3 Large eddy simulation (LES) - LBM-LES

3 The coherent structure model (CSM)

3.1 Large eddy simulation (LES)

3.2 Smagorinsky model - LBM-LES

3.3 Coherent Structure

4 CSM implementation in lbmpy

4.1 pystencils

4.2 lbmpy

4.3 Implementation of CSM in lbmpy

4.4 Usage of CSM in lbmpy
LIST OF TABLES

5.1 QUADRO P5000 ................................................................. 45
5.2 Kida Vortex - Resume Table for LBMs D3Q27: Domain 300x300x300 .......... 48
5.3 Kida Vortex - Compressible LBM-CSM D3Q27 ................................. 49
5.4 Kida Vortex - Compressible LBM-NRCSM D3Q27 ............................ 51
5.5 Turbulent Channel flow - Comparison of LBMs D3Q27 with $\delta = 50$ .... 58
5.6 Turbulent Channel flow - Compressible LBM-CSM D3Q27 ..................... 59
5.7 Turbulent Channel flow - Compressible LBM-NRCSM D3Q27 ................ 62

6.1 Resume Table for LBM Methods: Domain 1920x320 (LxH) .................... 68
6.2 Square Cylinder - incompressible LBM CSM D2Q9 ............................ 69
6.3 Square Cylinder - Compressible LBM CSM D2Q9 ............................... 71
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>LBM-SCALES</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>HPP</td>
<td>7</td>
</tr>
<tr>
<td>2.2</td>
<td>FHP</td>
<td>8</td>
</tr>
<tr>
<td>2.3</td>
<td>LBM-D2Q9</td>
<td>11</td>
</tr>
<tr>
<td>4.1</td>
<td>Jacobi Stencil</td>
<td>33</td>
</tr>
<tr>
<td>5.1</td>
<td>Kida Vortex - CSM</td>
<td>50</td>
</tr>
<tr>
<td>5.2</td>
<td>Kida Vortex - CSM</td>
<td>50</td>
</tr>
<tr>
<td>5.3</td>
<td>Kida Vortex - NRCSM</td>
<td>52</td>
</tr>
<tr>
<td>5.4</td>
<td>Kida Vortex - NRCSM-FD</td>
<td>52</td>
</tr>
<tr>
<td>5.5</td>
<td>Kida vortex Max Re CSM D3Q27</td>
<td>53</td>
</tr>
<tr>
<td>5.6</td>
<td>Kida vortex Max Re NRCSM D3Q27</td>
<td>53</td>
</tr>
<tr>
<td>5.7</td>
<td>Kida vortex Max Re TRT-KBC</td>
<td>54</td>
</tr>
<tr>
<td>5.8</td>
<td>Kida vortex Max Re CSM D3Q19</td>
<td>54</td>
</tr>
<tr>
<td>5.9</td>
<td>Turbulent Channel Flow CSM D3Q27 - H=64</td>
<td>60</td>
</tr>
<tr>
<td>5.10</td>
<td>Turbulent Channel Flow CSM-FD D3Q27 - H=64</td>
<td>60</td>
</tr>
<tr>
<td>5.11</td>
<td>Turbulent Channel Flow CSM D3Q27 - H=100</td>
<td>61</td>
</tr>
<tr>
<td>5.12</td>
<td>Turbulent Channel Flow CSM-FD D3Q27 - H=100</td>
<td>61</td>
</tr>
<tr>
<td>5.13</td>
<td>Turbulent Channel Flow CSM D3Q27 - H=64</td>
<td>62</td>
</tr>
<tr>
<td>5.14</td>
<td>Turbulent Channel Flow CSM-FD D3Q27 - H=64</td>
<td>63</td>
</tr>
<tr>
<td>5.15</td>
<td>Turbulent Channel Flow CSM D3Q27 - H=100</td>
<td>63</td>
</tr>
<tr>
<td>5.16</td>
<td>Turbulent Channel Flow CSM-FD D3Q27 - H=100</td>
<td>64</td>
</tr>
<tr>
<td>6.1</td>
<td>Geometry of Square Cylinder</td>
<td>66</td>
</tr>
<tr>
<td>6.2</td>
<td>Square Cylinder LBM CSM - Vortex formation</td>
<td>67</td>
</tr>
<tr>
<td>6.3</td>
<td>Square Cylinder - LBM CSM</td>
<td>70</td>
</tr>
<tr>
<td>6.4</td>
<td>Square Cylinder - LBM CSM-FD</td>
<td>70</td>
</tr>
<tr>
<td>6.5</td>
<td>Square Cylinder - LBM NRCSM</td>
<td>72</td>
</tr>
</tbody>
</table>
6.6 Square Cylinder - LBM NRCSM-FD ................................. 72
6.7 Geometry of wind tunnel test ...................................... 73
6.8 StarCCM+ wind tunnel mesh ........................................ 75
6.9 Wind tunnel test - shock waves with CSM ...................... 76
6.10 Wind tunnel test - coarse grid refinement ...................... 77
6.11 Wind tunnel test - middle grid refinement ..................... 77
6.12 Wind tunnel test - Inlet profile ................................. 78
6.13 Wind tunnel test with inlet boundary condition - shock waves ............................. 78
6.14 Wind tunnel test with inlet boundary condition - shock waves and singularities ... 79
6.15 TSN Geometry ...................................................... 79
6.16 TSN-LBM Plane XY ................................................ 80
6.17 TSN-LBM Plane XZ ................................................ 80
6.18 TSN-LBM Initial velocity .......................................... 81
6.19 TSN-LBM shock waves ............................................ 81
6.20 TSN-LBM shock waves ............................................ 82
6.21 TSN-LBM Simulation ............................................... 82
CHAPTER 1
INTRODUCTION

Fluid dynamics and its physical manifestation has different impacts on our daily lives, from the forecast of the weather, to bring an umbrella or not, to thermonuclear combustion in white dwarfs; thus, arising in humanity the interest to understand and being able to predict their behavior.

Nevertheless, the study of fluid dynamics is a complex and challenging area where the Navier Stokes equation provides the fundamental basis of the continuum description. However, the Navier Stokes equations are only solvable for simplistic situations and it is part of the millennium problems.

The most exciting phenomena arise with high Reynolds numbers, where the dissipative forces are small compared with the inertial forces. In this regime, the flow becomes turbulent, such that the motion of the fluid appears chaotic and unpredictable, making an analytical description more complex. With the increase of computational power, numerical simulations of fluid flow have become a valuable tool to understand and predict its behavior, resulting in the field of Computational Fluid Dynamics (CFD).

For several decades, the application of CFD to study environmental flows, characterized by high Reynolds numbers, has increased, for example, the study of air quality in urban areas. Nowadays, due to more strict regulations of diverse types of pollution and concern to protect the environment, companies and national organization have increased their interest to model, with reasonable accuracy, the dispersion of pollutant from a source, such as a nuclear power plant. However, the application of CFD in wind flow simulations around the building is particularly challenging, due to the high Reynolds numbers in these applications, the complex nature of the 3D flow field, numerical difficulties associated with flow at sharp corners and the difficult inflow boundary conditions [5].

Traditional CFD methods have predominated the research field of wind flow simulations, mainly by the Reynolds averaged Navier-Stokes (RANS) and the large eddy simulation (LES). The first efforts to evaluate wind flows around buildings using RANS were made in the 1970s by Yamada and
Meroney, and LES in the 1990s by Murakami, Hibi, and others. Even though LES is capable to model the inherently transient features of the flow such as vortex shedding in the wake in contrast to RANS, RANS has predominated this field for several reasons. One of the main reasons is the low computational cost of RANS compared to LES and the low accuracy requirement of environmental flows (errors of more than 25% might be acceptable) [5].

Even though RANS and LES have presented great results for wind flow simulations in buildings (indoors and outdoors simulations), the required computational time to simulate one second of the flow is vastly long. Thus, it is almost impossible to perform real-time simulations within reasonable accuracy. Nowadays, the need arises for companies and national organizations to monitor smoke and dispersion of highly toxic pollutants resulting from disasters in real time ("Emergency Preparedness").

This brings up the requirement to develop efficient, robust, and accurate solvers independent of complexity and domain characteristic of the flow within a real-time framework, which is an immensely exciting topic.

1.1 MOTIVATION

In recent years, besides classic CFD approaches based on the Navier-Stokes equations (macroscopic scale), a new method called the lattice Boltzmann method (LBM) has been established and further developed and is now a promising alternative to classic CFD [21]. Unlike the traditional CFD methods, the LBM is based on microscopic models and mesoscopic kinetic equations, laying between the microscopic and macroscopic world as shown in the figure 2.3. Moreover, its highly parallelizable nature has been one of the most promising characteristics of the LBM, being suitable for real-time simulations and its implementation on GPU clusters and supercomputers [21].

The first or original LBM formulation is the single relaxation-time (SRT) or LBGK [21]. The SRT method present diverse shortcomings to simulate highly turbulent flows with an affordable resolution, which will be detailed later. To overcome these shortcomings, several improvements and variations of the original LBM were developed. One alternative to the SRT is the multi relaxation-time method (MRT), which takes advantage of different relaxation rates in a specific moment space to stabilize and improve the accuracy of the simulation [21]. However, the increment of the variables (relaxation rates and moment spaces) on the MRT increases the complexity of the selection
of parameters. Another approach to improve the LBM is the entropic model [3, 9, 29], where one of the subclasses with great results is the KBC models [6], and the application of sub-grid scale models (SGS) to the kinetic equation, the introduction of turbulent viscosity and spatial relaxation [17, 20, 27, 19]. This last approach has gained popularity due to its fast implementation and relation with traditional schemes, such as RANS and large eddy simulation (LES). Furthermore, between the different LBM-SGS models, the coherent structure model (CSM) [19, 25] has presented excellent outcomes for the simulation of environmental flows due to its local parameter evaluation in contrast to dynamic Smagorinsky models.

On the other hand, there are not many studies on how the different LBM behave and which one is the most suitable for specific applications, especially for environmental flows. Moreover, the fact that the LBM and the traditional CFD methods belong to different scales, mesoscopic and macroscopic scale, respectively, generate a communication barrier between users of both methods. Due to the shortcomings mentioned, a barrier appears between interested groups and researchers and increase the difficulty to use the LBM in industry, where the usage of traditional CFD methods predominate.

For this reason, various variations of the LBM are explored in the present thesis and compared with the focus on the simulation of environmental flows.

As a first step, the lbmpy is selected as our framework [10]. The lbmpy is a module in the pystencils code generator [11], developed at the chair of Informatik 10 Friedrich-Alexander-Universität Erlangen-Nürnberg. lbmpy is a package to generate fast C code implementations through symbolic representation (SymPy) and meta-programming. Moreover, lbmpy can generate kernles in other
platforms as C(++) for CPUs and CUDA for NVIDIA GPUs, which can be integrated into existing C/C++ frameworks as waLBerla.

`lbmpy` has different LBMs, but it does not have implemented the CSM, which looks promising for our target application. Thus, as a first task, the CSM is implemented in `lbmpy`. Once the CSM is implemented in `lbmpy`, diverse benchmarks are performed to verify the correct implementation. Then, the CSM is compared not only with other LBM variants, but also with experimental data and other numerical methods. In this way, a general comparison with different variations of the original LBM is possible with regards to accuracy and performance.

Finally, the CSM and the different LBM variants are validated for environmental flows. The wind flow around a building is simulated with a suitable LBM and compared with a simulation using the software StarCCM+, a traditional and commercial CFD program. Therefore, a significant outcome of the thesis is the analysis of the different LBM for high turbulent flows with a particular focus on wind flow simulations towards real-time simulations.

### 1.2 OUTLINE OF THE THESIS

- Chapter 1: Motivation and outline of the master thesis.
- Chapter 2: The lattice Boltzmann method is introduced from a historical point of view, from the early lattice gas automata to the basic lattice Bhatnagar-Gross-Krook model (LBGK) or single relaxation time (SRT) is explained in detail. Besides, the different variants of the LBM are briefly described.
- Chapter 3: The coherent structure model (CSM) is explained in detail. Starting from the large eddy simulation (LES) theory, the Smagorinsky model for LBM (LBM-LES), and lastly the CSM.
- Chapter 4: The CSM implementation in the `lbmpy` module is described in detail.
- Chapter 5: The CSM implementation is verified with the 3D Kida Vortex and the classic turbulent channel flow benchmarks.
- Chapter 6: The CSM and the different LBM models are validated for environmental flows, addressing a classic square cylinder simulation, a wind tunnel test, and a wind flow simulation around a nuclear power plant (industrial application).
CHAPTER 2
LATTICE BOLTZMANN METHOD

Thanks to the evolution of computers and the increase of computing power, the field of computational fluid dynamics (CFD) opens the door to understand in more detail the physical phenomena around us. However, we are not yet at the technological point to be able to simulate fully resolved flows for industrial applications through direct numerical simulations (DNS) for highly turbulent flows. This limit is mostly due to the high computational resources needed. As an alternative to DNS, the continuum equations are often filtered and the sub-grid scales are modeled, resulting in the field of turbulence modeling. The most common turbulence models are based on eddy-viscosity models, Reynolds-stress models, large eddy simulation (LES), and probability density function (PDF). These methods discretize the macroscopic continuum equation, dealing with the Navier-Stokes equation and its non-linear terms.

Moreover, the field of environmental flows and particularly building simulations are particularly challenging. Due to its complex nature, characterized by high Reynolds numbers, complex flow phenomena such as impingement flows, separation and vortex shedding, sharp corners as well as complex boundary conditions at the inflow. In order to overcome these problems, it is required to use high grid resolutions near the walls and accurate wall functions, discretization schemes near the corners, and inflow and outflow boundaries that reflect reality. Thus, this is a complicated and challenging scenario for CFD engineers.

There exist diverse CFD techniques to simulate flows around buildings, where the two most popular approaches by far are LES and Reynolds-averaged Navier-Stokes simulations (RANS) [5]. Each method has its advantages and disadvantages. LES has the potential to provide more accurate solutions and more reliable results than RANS simulations but with a higher simulation complexity and computational cost, thus less suitable for real-time simulations.

An alternative to the traditional CFD methods for building simulation is the LBM. The LBM is based on the Boltzmann equation, which describes the dynamics of particles on a mesoscopic scale.
In other words, LBM is a microscopic approach that incorporates the essential characteristics of the physical microscopic processes so that the macroscopic averaged properties obey the desired macroscopic equation.

2.1 LATTICE GAS AUTOMATA

The LBM has its base in the lattice gas automata (LGA), which can be explained as a simple molecular dynamics method for gas simulations. Thus, the LGAs is a microscopic method with the capability to recover macroscopic quantities from it, making it possible to simulate fluid dynamics. Moreover, the gas is modeled as a conjunction of hard spheres or particles with a discrete set of possible velocities for each particle, a Boolean occupation number (for each node) and a specific set of rules for the collision between particles, in this way conserving mass and momentum. Their evolution equation is the following:

\[
 n_i(x_i^t + c_i^t \Delta t, t + \Delta t) - n_i(x_i^t, t) = \Omega_i (f(x_i^t, t)) \tag{2.1}
\]

where \( n_i(x_i^t, c_i^t, t) \) is the boolean occupation number, \( c_i^t \) is the velocity vector, \( \Delta t \) is the time step (equal to one for LGA), and \( \Omega \) is the collision operator, which is dependent of \( n_i(x_i^t, t) \). This means that for the LG the evolution equation is:

\[
 n_i(x_i^t + c_i^t, t + 1) - n_i(x_i^t, t) = \Omega_i (f(x_i^t, t)) \tag{2.2}
\]

One of the basic LGAs is the HPP model; which was proposed by Hardy, Pomeau, and de Pazzis and named by its creator [15]. In this method, the grid is two-dimensional and cartesian; thus, each node in the grid has only four neighbors, and the particles have only four possible velocities vectors, \( c_1 = (1, 0), \ c_2 = (0, 1), \ c_3 = (-1, 0), \) and \( c_4 = (0, -1) \), as shown in the figure 2.1. Each particle travels from node to node according to their respective velocity vector, when two or more particles meet in a node a collision happens. To ensure momentum and mass conservation, the number of particles and total velocity must be the same before and after the collision. When two particles collide head-on, the particles change their velocity 90 degrees in the counterclockwise direction. However, when three or more particles collide, there is no change in their velocity (no change in their configuration) [30].
An appealing aspect of the HPP model is that the collision operator is deterministic, meaning that each collision has only one possible outcome; thus, making possible that a simulation can be reversed from any time step to its original state (time reversal invariance). Besides, the state of each node in the grid can be entirely described by four bits, where every bit represents the absence or presence of the particle in the four possible velocity vectors. At last, the events in every node are not correlated with the simultaneous events occurring at other nodes; thus the system has inherent parallel nature [30].

On the other hand, the HPP model present diverse disadvantages. Due to its microscopic scope, it can never reach a complete equilibrium state. In other words, the system will never reach by itself a state where the system is identical from one step to the next one. This problem happens due to its time reversal invariance characteristic, were to reach a complete equilibrium state all states must be equal. Thus, in every time step, the state of the system in the microscopic scale changes. In order to recover the macroscopic quantities, space and time averages are required, but there will always be statical noise, even with large averaging domains. Moreover, this method lacks rotational invariance, which means that its behavior becomes anisotropic, and fails to behave under the Navier-Stokes equation in the macroscopic scale [30].

For this reason, the HPP model was abandoned in the late 80s. In 1986 Frisch, Hasslacher and Pomeu suggested the FHP model (also named by the inventors) [13]. This method moved from a square lattice to a hexagonal lattice with six velocity vectors, $\vec{c}_1 = (1, 0)$, $\vec{c}_2 = (1/2, \sqrt{3}/2)$, $\vec{c}_3 = (-1/2, \sqrt{3}/2)$, $\vec{c}_4 = (-1, 0)$, $\vec{c}_5 = (-1/2, -\sqrt{3}/2)$, and $\vec{c}_6 = (1/2, -\sqrt{3}/2)$, as shown in the figure 2.2.
Another significant change is the collision operator for two particles since there exist two possible configurations that conserve mass and momentum. In order to choose the next configuration, it is resolved randomly with equal probabilities. Due to the incorporation of a stochastic process, the FHP method loses the time reversal invariance characteristic from the HPP. Besides, for three particles collision is resolved by reflection, for all other collisions, there is no change in configuration [30].

The hexagonal configuration of the FHP provides the rotational invariance required to recover the incompressible Navier-Stokes equation. Moreover, the FHP method retains the advantages of the HPP, such as small storage and high parallelism [30].

However, the FHP can not overcome the statical noise in pursue to recover the macroscopic quantities. The reasons are the random fluctuations in the microscopic scale that disappear in the continuum limit [30].

As a way to keep improving the FHP, Frisch proposed the replacement of the boolean particle with distribution functions, which represent its ensemble average:

\[ f_i = \bar{n}_i \]  

(2.3)

If we replace the boolean particle with distribution functions in the equation (2.2) we obtain the
following equation:

\[ f_i(\vec{x}_i + \vec{c}_i \Delta t, t + \Delta t) - f_i(\vec{x}_i, t) = \Omega_i(f(\vec{x}_i, t)) \quad (2.4) \]

### 2.2 LATTICE BOLTZMANN METHOD

The idea or method proposed by Frisch can be achieved also through the Boltzmann equation from statistical mechanics. The Boltzmann transport equation, which is the first-order Taylor expansion of the kinetic Boltzmann equation:

\[
\frac{\partial f(\vec{x}, \vec{c}, t)}{\partial t} + \vec{c} \cdot \nabla_x f(\vec{x}, \vec{c}, t) + \vec{F} \cdot \nabla_p f(\vec{x}, \vec{c}, t) = \Omega \quad (2.5)
\]

where the distribution function \( f(\vec{x}, \vec{c}, t) \) depends not only on the position \( \vec{x} \) and time \( t \) but also on the momentum \( \vec{p} \). The momentum can also be expressed as \( \vec{p} = m \times \vec{c} \), where \( \vec{c} \) is the velocity vector. Besides, \( F \) is an external force applied at \( \vec{x} \) and \( d\vec{p} = \vec{F} dt \).

The lattice Boltzmann equation (2.4) can be seen as a further discretization of equation (2.5). In this case, the external forces are neglected, and the mass is normalized to one, thus \( \vec{p} = \vec{c} \). Then we discretize the velocity space to a set of vectors that cover the entire space as the HPP and FHP models, figures 2.1 and 2.2. With the discretization of the velocity space, we can simplify the representation of the distribution function \( f(\vec{x}, \vec{c}, t) \) to \( f_i(\vec{x}, t) \) without loss of information and also the collision operator \( \Omega \) to \( \Omega_i(\vec{x}, t) \). After further discretization we are left with:

\[
\frac{\partial f_i(\vec{x}, t)}{\partial t} + \vec{c}_i \cdot \nabla_x f_i(\vec{x}, t) = \Omega_i(f(\vec{x}_i, t)) \quad (2.6)
\]

To obtain the lattice Boltzmann equation, we apply a Euler time step and an upwind spatial discretization to the continuum Boltzmann equation (2.2):

\[
\frac{f_i(\vec{x}, t + \Delta t) - f_i(\vec{x}, t) + f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) - f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t)}{\Delta t} = \Omega_i \quad (2.7)
\]

Setting up the time and grid spacing to unity we can obtain:

\[
f_i(\vec{x} + \vec{c}_i \Delta t, t + \Delta t) - f_i(\vec{x}, t) = \Omega_i(f(\vec{x}_i, t)) \quad (2.8)
\]
In first sight, if particles collide in our system, after some time, they will reach the equilibrium state. The time to reach the equilibrium state depends on which type of collision we assume and relaxation parameters. In 1989 Higuera [16] proposed the replacement of Boltzmann’s collision operator by a linearized collision operator. Moreover, diverse groups proposed the replacement of the ”exact” collision for a simplified version that would allow, in a controllable way, to reach the equilibrium state. On the other hand, the equilibrium state was selected in such a way to be only dependable on local fluid variables which can be obtained from the actual values of the distribution functions and satisfy the macroscopic properties of the fluid.

In the 90s several researchers proposed further simplifications of the LBM collision operator. In 1992 Chen [8] and Qian [28] proposed the use of a single-relaxation-time approximation using a collision operator similar to the one proposed by Bhatnagar, Gross, and Krook for the Boltzmann equation in 1954 [4]. Thus, the name of the LBGK operator, which is given by:

$$\Omega_i(f(\vec{x}_i, t)) = -\frac{f_i(\vec{x}, t) - f_i^{eq}(\vec{x}, t)}{\tau}$$

(2.9)

where $\tau$ controls the approach to equilibrium, a free parameter known as the relaxation time, and $f^{eq}$ is the equilibrium distribution of particles.

The equilibrium distribution of particles $f^{eq}$ can be derived from the Maxwell-Boltzmann velocity distribution from statistical mechanics. By assuming ideal gas law, the isothermal ideal gas pressure relation, and finally a low Mach number, allowing us to truncate the Taylor expansion to the second order without loss of information [30], we arrive at the following equation:

$$f_i^{eq} = W_i \rho \left[ 1 + \frac{\vec{u} \cdot \vec{c}_i}{c_s^2} + \frac{(\vec{u} \cdot \vec{c}_i)^2}{2c_s^4} - \frac{\vec{u} \cdot \vec{u}}{2c_s^2} \right]$$

(2.10)

where $W_i$ are the weight of the velocity vector, $\vec{u}$ is the mean velocity and $c_s$ is the speed of sound in lattice units. Applying all of this we arrive at the fully discrete lattice Boltzmann equation:

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

(2.11)
Besides, the LBM can be based on diverse possible lattice configurations, but not all of them are appropriate. The different lattice configurations must meet certain requirements to give sufficiently isotropic behavior to recover the Navier-Stokes equation in the macroscopic scale. Considering the LBGK operator, those are the following [30]:

\[
\sum_i W_i = 1 \quad (2.12)
\]
\[
\sum_i W_i c_{i\alpha} = 0 \quad (2.13)
\]
\[
\sum_i W_i c_{i\alpha} c_{i\beta} = c_s^2 \delta_{\alpha\beta} \quad (2.14)
\]
\[
\sum_i W_i c_{i\alpha} c_{i\beta} c_{i\gamma} = 0 \quad (2.15)
\]
\[
\sum_i W_i c_{i\alpha} c_{i\beta} c_{i\gamma} c_{i\delta} = c_s^4 \left( \delta_{\alpha\beta} \delta_{\gamma\delta} + \delta_{\alpha\gamma} \delta_{\beta\delta} + \delta_{\alpha\delta} \delta_{\beta\gamma} \right) \quad (2.16)
\]
\[
\sum_i W_i c_{i\alpha} c_{i\beta} c_{i\gamma} c_{i\delta} c_{i\epsilon} = 0 \quad (2.17)
\]

For the different lattice configurations or LBM stencil, we have to define its \(d\)-dimensionality and the number of lattice vectors \(q\) (\(D_dQ_q\) nomenclature). The most common lattice configurations are the D2Q9, D3Q19 and D3Q27, where the speed of sound \((c_s)\) have the same value of \(1/\sqrt{3}\). As an example, we have the D2Q9 lattice in the figure 2.3 with the following characteristics:

\[
W_i = \begin{cases} 
  W_0, & \text{for } i = 0, \text{ the rest vector} \\
  W_s, & \text{for } i = 1, 2, 3, 4, \text{ the short vectors} \\
  W_l, & \text{for } i = 5, 6, 7, 8, \text{ the long vectors}
\end{cases} \quad (2.18)
\]

Figure 2.3: D2Q9 lattice (a) Discrete velocity vectors, (b) interaction D2Q9 with neighbors
From the conditions (2.12), (2.13), (2.14), and (2.15), we have that:

\begin{align*}
W_0 + 4W_s + 4W_l &= 1 \quad (2.19) \\
2W_s + 4W_l &= c_s^2 \quad (2.20) \\
2W_s + 4W_l &= 3c_s^4 \quad (2.21) \\
4W_l &= c_s^4 \quad (2.22)
\end{align*}

The only configuration that satisfy the conditions (2.19), (2.20), (2.21), and (2.22) are \( W_0 = 4/9, W_s = 1/9, W_l = 1/36, \) and \( c_s = 1/\sqrt{3}. \)

### 2.3 LBM TO NAVIER STOKES EQUATION

In the first place, the discrete lattice Boltzmann equation conserves the mass and the momentum in the collision step:

\begin{align*}
\rho(\vec{x}, t) &= \sum_i f_i(\vec{x}, t) \quad (2.23) \\
\rho \vec{u}(\vec{x}, t) &= \sum_i \vec{c}_i f_i(\vec{x}, t) \quad (2.24)
\end{align*}

To derive the Navier Stokes equation, the procedure done by Hou [17] is followed with \( c_s = 1/\sqrt{3}. \) The following step will describe the basis of the derivation.

In the first place, the Chapman-Enskog procedure is required, assuming a multi-scale expansion of time and space derivatives in the small parameter \( \epsilon: \)

\[ \frac{\partial}{\partial l} = \epsilon \frac{\partial}{\partial l_1} + \epsilon^2 \frac{\partial}{\partial l_2} + ... \]
\[ \nabla = \epsilon \nabla_1 + \epsilon^2 \nabla_2 + ... \]

We also need the expansion of the distribution function:

\[ f_i = f_i^{(0)} + \epsilon f_i^{(1)} + \epsilon^2 f_i^{(2)} + ... \]
The first and second order in \( \epsilon \) on the continuum Boltzmann equation (2.5) [17] are needed. Besides, a low Knudsen number \((Kn)\) is assumed, which implies that the mean free path of a molecule is small enough such that the continuum assumption of fluid mechanics is valid.

Moreover, the expansion parameter \( \epsilon \) can be interpreted as a perturbation of \( f_i \) around the equilibrium distribution \( f_i^{(eq)} \) with the Knudsen number as the expansion parameter. Thus, \( \epsilon \) can be regarded as the Knudsen number \((\epsilon = Kn)\):

\[
- \frac{\left( f_i - f_i^{(eq)} \right)}{\tau} = -\frac{1}{\tau} \left( f_i^{(1)} + \epsilon f_i^{(2)} + ... \right)
\]  

(2.25)

Since the mass and the momentum are conserved in collisions and the summations over nonequilibrium populations are zero, we can replace the above equations into (2.8):

\[
\frac{\partial f_i^{(0)}}{\partial t_1} + \vec{c}_i \cdot \nabla_1 f_i^{(0)} = -\frac{1}{\tau} f_i^{(1)}
\]

(2.26)

\[
\frac{\partial f_i^{(0)}}{\partial t_2} + \left( \frac{\partial}{\partial t_1} + \vec{c}_i \cdot \nabla_1 \right) \left( 1 - \frac{1}{2\tau} \right) f_i^{(1)} = -\frac{f_i^{(2)}}{\tau}
\]

(2.27)

When the equations (2.26) and (2.27) are summed over the \( i \) velocities, the continuity equation to second order in \( \epsilon \) is obtained:

\[
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0
\]

(2.28)

The momentum equation to second order in \( \epsilon \) is derived by multiplying the equation (2.28) with the velocities \( \vec{c}_i \) and then summing over the velocities:

\[
\frac{\partial (\rho \vec{c})}{\partial t} + \nabla \cdot \left( \Pi^{(0)} + \left( 1 - \frac{1}{2\tau} \right) \Pi^{(1)} \right) = 0
\]

(2.29)

where \( \Pi^{(0)} \) and \( \Pi^{(1)} \) are the momentum flux tensor:

\[
\Pi_{\alpha \beta}^{(0)} = \sum_i c_{i\alpha} c_{i\beta} f_i^{(0)}
\]

(2.30)
\[ \Pi_{\alpha\beta}^{(1)} = \sum_i c_{i\alpha}c_{i\beta}f_i^{(1)} \] (2.31)

By selecting a lattice geometry and equilibrium distributional function form and then matching the moments of the distributional functional with the Navier Stokes equation, we arrive at:

\[ \Pi_{\alpha\beta}^{(0)} = \rho \sigma_{\alpha\beta} + \rho u_{\alpha} u_{\beta} \] (2.32)
\[ \Pi_{\alpha\beta}^{(1)} = -\frac{2\rho r}{3} S_{\alpha\beta} \] (2.33)

Now, applying the equations (2.32) and (2.32) to (2.29) we obtain:

\[ \rho \frac{\partial u_{\alpha}}{\partial t} + \rho u_{\beta} \frac{\partial u_{\alpha}}{\partial x_{\beta}} = -\frac{\partial P}{\partial x_{\alpha}} + \frac{\partial}{\partial x_{\beta}} \left( \mu \left( \frac{\partial u_{\beta}}{\partial x_{\alpha}} + \frac{\partial u_{\alpha}}{\partial x_{\beta}} \right) \right) \] (2.34)

where \( \mu \) is the shear viscosity, which leads to the kinematic viscosity.

\[ \mu = \frac{2\tau - 1}{6} \rho \] (2.35)
\[ \nu = \frac{2\tau - 1}{6} \] (2.36)

If other configuration with \( c_s \) different than \( 1/\sqrt{3} \) is used, then the equation (2.36) would take the form:

\[ \nu = \frac{2\tau - 1}{2} c_s^2 \] (2.37)

### 2.4 LATTICE UNITS TO MACROSCOPIC SCALE

There are two different procedures to convert physical units to lattice units. The first one is to convert directly between them, and the other is through dimensionless units, such as the Reynolds number.

In the first type of conversion, the physical units are related to the lattice units through the time step \( \Delta t \) and the node spacing \( \Delta x \). For example, the physical speed of sound is related as follows [30]:

\[ c_{s,\text{physical}} = c_{s,\text{lattice}} \frac{\Delta x}{\Delta t} \] (2.38)
Applying this procedure to the viscosity and considering the equation (2.37) and (2.38):

\[
\nu_{\text{physical}} = \nu_{\text{lattice}} \frac{\Delta x^2}{\Delta t} = c_{s,\text{lattice}}^2 \frac{2\tau - 1}{2} \frac{\Delta x^2}{\Delta t} = c_{s,\text{physical}}^2 \frac{2\tau - 1}{2} \Delta t
\]  

(2.39)

Thus, we can relate the time step and the node spacing with the relaxation rate by using the equation.

\[
\Delta t = \frac{\nu_{\text{physical}}}{c_{s,\text{physical}}^2 (\tau - 1/2)}
\]  

(2.40)

We can use the same procedure to get an equation for the node spacing:

\[
\Delta x = \frac{\nu_{\text{physical}}}{c_{s,\text{physical}} c_{s,\text{lattice}} (\tau - 1/2)}
\]  

(2.41)

This method has as a free parameter \( \tau \), and the variables \( \nu_{\text{physical}} \) and \( c_{s,\text{physical}} \) are given by the system.

The second method is the most used due to its simplicity and the possibility to analyze the accuracy of the selected LBM. In the first instance, the density, node spacing, and time step are set to one in lattice units. Thus, the characteristic length and velocity are set as:

\[
C_x = \frac{L}{N \Delta x} = \frac{L}{N}
\]  

(2.42)

\[
C_u = \frac{U}{|u|}
\]  

(2.43)

where \( N \) is the number of lattices that represents the characteristic length \( L \), \( |u| \) is the characteristic lattice velocity, \( U \) is the characteristic physical velocity, and \( C_x \) and \( C_u \) are the conversion factor for space and velocity, respectively. With this information, we can set the conversion factor of time:

\[
C_t = \frac{C_x}{C_u}
\]  

(2.44)

Using this procedure, we can relate the Reynolds number, which has to be identical in both systems, and by using equation (2.36) we obtain:
\[ Re = \frac{UL}{\nu} = \frac{|u|N}{\nu_{\text{lattice}}} = 6|u|N \tau - \frac{1}{2} \]  

(2.45)

### 2.5 ACCURACY AND STABILITY

By employing the equation (2.45), we can observe that there is an infinite number of combinations to obtain the physical properties in lattice units through \( N, \tau, \) and \(|u|\). Thus, we might arrive at the question: how do we choose these parameters? To answer this question, we need to understand the limitations and primary characteristics of the original LBM, from now on referenced as the SRT method.

For simplicity, we discuss the stability of the SRT without considering boundaries as Krüger explained in his book [21]. With this, we will have important insight into the limit values of \(|u|\) and \(\tau\).

A sufficient stability condition is the non-negativity of all equilibrium populations; this condition holds for any value of \(\tau/\Delta t > 1/2\) as long as \(f_i^{eq} \geq 0\), for all \(i\). The main problem from this sufficient stability condition is that the equilibrium populations are a complicated function of the velocity \((\overrightarrow{u})\). For simplicity, the maximum achievable velocity magnitude \(|u_{\text{max}}|\) is used. For the LBGK collision operator with the equilibrium function (2.10), we can write it as (for D2Q9, D3Q15, D3Q19, and D3Q27):

\[ |u| \geq \sqrt{\frac{1}{3}} \frac{\Delta x}{\Delta t} \approx 0.577 \]  

(2.46)

However, this stability condition is only guaranteed in the bulk if \(\tau/\Delta t \geq 1\). For other relaxation times, the stability condition function is more complicated. As a rule of thumb for \(\tau/\Delta t < 0.55\), one can use the following equation [21].

\[ |u| \leq 8 \left( \frac{\tau}{\Delta t} - \frac{1}{2} \right) \frac{\Delta x}{\Delta t}, \quad \frac{\tau}{\Delta t} < 0.55 \]  

(2.47)

or

\[ \tau \geq \left( \frac{\Delta t}{\Delta x} \frac{|u|}{8} + \frac{1}{2} \right) \Delta t \]  

(2.48)

Moreover, Krüger presents in his book [21], a table with the requirements to have good accuracy,
stability, and efficiency of the simulation. From this table, we can have the following recommendations:

\[
0.5 < \tau \leq 1 \quad (2.49)
\]
\[
|u_{max}| \leq 0.1 \quad (2.50)
\]

These equations give an idea of the parameters to select but do not guarantee the stability of the simulation, since it is considered in the bulk of the fluid and not near the boundaries. However, we can use them to get an approximate maximum Reynolds number for a discretization. For example, with \(N = 100\) and \(|u| = 0.1\) and the maximum Reynolds number for stable simulations, can be estimated \(Re_{max} = \frac{6N|u|}{\tau_{min}^{-1/2}} = \frac{60}{0.0125} = 4800\).

The maximum Re with a high probability to be stable is 4,800.0 for the configuration selected and lay below our target. Thus, the SRT cannot fulfill the need for highly turbulent flow simulations within reasonable computational resources. Thus, diverse research group has improved the SRT and there exist several LBM variants.

### 2.6 DIFFERENT LBMS

To overcome the stability limitation of the original LBM, diverse research groups have developed variations of it where the most prominent are the multi-relaxation time (MRT), the entropic-MRT models (KBC), and the sub-grid scale models (SGS).

#### 2.6.1 Multi-relaxation time (MRT)

The idea behind the MRT method is to introduce different relaxation rates \(\omega = 1/\tau\). Thus, the collision step can be improved to ensure better accuracy and stability of the LBM simulation.

The first idea was to relax each particle population in the collision step. However, this method is not so precise and requires more constraints to satisfy the conservation laws for density and momentum. This method, which is called MRT-L\[21\], is quite complicated and will not be treated here.

The second idea was to map the population into moment space or kinetic space. For a LBM stencil \(DdQq\), the moment space has \(q\)-dimensions and each moment can be represented as any linear com-
bination of the particle distribution functions. Each of these moments can be relaxed individually and be controlled to ensure the stability and accuracy of the system.

Following this idea, we can define the map to the moment space as a summation of populations in a $DdQq$ velocity set through a $q \times q$ matrix as:

$$m_k = \sum_{i=0}^{q-1} M_{ki} f_i \quad \text{for} \quad k = 0, ..., q - 1 \quad (2.51)$$

We can rewrite equation (2.51) on a vector-matrix form $m = Mf$, where:

$$m = \begin{pmatrix} m_0 \\ \vdots \\ m_{q-1} \end{pmatrix}, \quad M = \begin{pmatrix} M_{0,0} & \cdots & M_{0,q-1} \\ \vdots & \ddots & \vdots \\ M_{q-1,0} & \cdots & M_{q-1,q-1} \end{pmatrix}, \quad f = \begin{pmatrix} f_0 \\ \vdots \\ f_{q-1} \end{pmatrix} \quad (2.52)$$

If we look closely to the transformation matrix $M$, we can observe that the SRT is a particular case of the MRT, where all the relaxation rates are chosen to be equal.

With a careful selection of the transformation matrix $M$, the obtained moments can represent specific hydrodynamic moments. Therefore, it is possible to affect those hydrodynamic moments individually by choosing different relaxation rates. The relaxation rates are introduced in a diagonal matrix form, containing the reciprocals of the relaxation time for each moment (relaxation matrix $S$). This will lead to the following equation:

$$f_i(x + c_i + \Delta t, t + \Delta t) - f_i(x, t) = -(M^{-1}SM)_{ij} [f_j(x, t) - f_{eq}^j(x, t)] \quad (2.53)$$

The most challenging and essential part of the MRT is the selection of moments. At first we must ensure the existence of the inverse of the matrix $M$. One primary approach is the usage of the Gram-Schmidt procedure, which generates an orthogonal set of linear independent vectors. Moreover, each moment is uniquely defined, and the group of moments can be relaxed independently, giving them enough flexibility to pursue stability and accuracy [21].

One class of such alternative basis resemble the natural moments and hydrodynamic behavior. In other words, each moment corresponds to measurable physical quantities related to the hydrodynamics.
namic behavior of the fluid. We choose D2Q9 stencil as an example. The natural moments for the D2Q9 are:

\[ \rho = m_{00} : \text{Normalization to the density} \]
\[ u_x = m_{10} : \text{Component of the momentum} \]
\[ u_y = m_{01} : \text{Component of the momentum} \]
\[ T = m_{20} + m_{02} : \text{Trace of the pressure tensor at unit density} \]
\[ N = m_{20} - m_{02} : \text{Normal stress difference at unit density} \]
\[ P_{xy} = m_{11} : \text{Off-diagonal component of the pressure tensor} \]
\[ Q_{xyy} = m_{12} : \text{Linear independent third order moment} \]
\[ Q_{yxx} = m_{21} : \text{Linear independent third order moment} \]
\[ A = m_{22} : \text{Lack of physical meaning} \]

As it can be appreciated, the first six moments are hydrodynamically relevant, and the other three do not have physical relevant meaning for isothermal flows (the last three moments are related to heat transport). The main gain from the moment spaces is the possibility to modify the physics of the model or to increase the stability of the model through different relaxation rates. However, we have several degrees of freedom for relaxation rates. One alternative is to have three relaxation rates for three different groups of moments, moments that affect the viscosity \((P_{xy}, N)\), moments that affect the bulk viscosity \((T, N)\), and other moments. This approach is from now on, called MRT3. One stable solution for the MRT3 is to relax the moments that affect the viscosity and set one to the others (no relaxation), this strategy will be used in our simulations.

### 2.6.2 Entropic multiple-relaxation time

The entropic multiple-relaxation time method combines the advantages of MRT and entropic lattice Boltzmann model [3]. These models are also known as KBC model [6].

At first, we must understand the principle of the entropic models has its origin in the second law of thermodynamics. The second law of thermodynamics basically states that any natural system is driven to an equilibrium state (state of maximum entropy). The kinetic energy drives the system out of equilibrium, while the collision of particles pulls it back to equilibrium, where entropy is locally maximized (Boltzmann’s H-theorem).
With a base on the second law of thermodynamics and the H-Theorem, the entropic models aim to recover these properties. Thus, making the simulations more stable and accurate without the need to resolve all the scales.

The first difference of the entropic models versus the LBGK is the redefinition of the equilibrium state. The entropic models define the local equilibrium through a suitable $H$-function, based in the H-Theorem:

$$ H[f] = -S[f] = \sum_i f_i \ln\left(\frac{f_i}{W_i}\right) $$

With a suitable $H$-function, the equilibrium is obtained by minimizing the function under the constraints of mass and momentum conservation.

The other difference is the definition of a mirror state in order to ensure the discrete-time H-Theorem.

$$ f_i^{\text{mirr}} = \alpha f_i^{eq} + (1 - \alpha) f_i $$

where $\alpha$ is the solution of the isentropic constraint, $H[f^{\text{mirr}}] = H[f]$, and ensure that entropy is always growing. More details can be found in the work of Karlin [3].

Meanwhile, the KBC represents the particle distribution function as a function of moments:

$$ f_i = k_i + s_i + h_i $$

where $k_i$ represent the kinetic part (locally conserved fields), $s_i$ represents the shear part (deviatoric stress tensor and other non-conserved moments), and $h_i$ represents the high-order moments (remaining moments not included in $s_i$).

With this representation, we can obtain a mirror state[6]:

$$ f_i^{\text{mirror}} = k_i + [2s_i^{eq} - s_i] + [(1 - \gamma)h_i + \gamma h_i^{eq}] $$
where $h_i^{eq}$ and $s_i^{eq}$ are calculated at the local equilibrium, and $\gamma$ is a spatial stabilizer under entropy control. Thus, $\gamma$ is an entropic stabilizer which self-adapts to a value given the maximum entropy condition and is calculated every time step on each lattice. More details can be found at [6].

As proposed in [6] there are different models for KBC according to the selected moments for the $k$–, $s$– and $h$– parts of the populations. The $k$–part contains the locally conserved moment; in other words, the moments related to $\rho$ and $u$. The $s$–part includes a combination of the (deviatoric) stress tensor moments. At last, the $h$–part contains the remaining moments.

Following the work of Karlin [6], there exist four different KBC models according to moments that belong to the $s$-part. As guidance, for the D2Q9:

- $k_i$: kinetic part has the moments $\rho, u_x$ and $u_y$
- $D$: (deviatoric) stress tensor $[N, P_{xy}]$
- $T$: trace of the stress tensor $[T]$
- $Q$: higher-order moments $[Q_{xyy}, Q_{yxx}, A]$

The moment $A$ always belongs to the $h_i$. The different KBC models have the following configuration for the $s_i$ part:

- KBC-N1: $[D]$
- KBC-N2: $[D, T]$
- KBC-N3: $[D, Q]$
- KBC-N4: $[D, T, Q]$

2.6.3 Large eddy simulation (LES) - LBM-LES

The concept of large eddy simulation (LES) model belongs to the traditional CFD method. LES was initially proposed in 1963 by Joseph Smagorinsky. The basic idea is to apply low-pass filtering to the Navier-Stokes equation to eliminate the small scales of the solution. This filtering leads to transformed equations which solve a filtered velocity field. In this way, LES models resolve the
large scales solutions, through the filtering function, and the smaller scales are modeled through a subgrid-scale model (SGS). Hence, reducing the computational cost without the loss of accuracy. This concept can also be applied to LBM.

Hou [17] developed an approach for LES in the LBM by applying the Smagorinsky subgrid scale turbulent model for 2D with the SRT and Krafczyk [20] for 3D with the MRT. Its main advantage is the local calculation of the stress tensor for the turbulent viscosity. However, the Smagorinsky model faces difficulties in simulating the flow near the walls accurately in the transitional region, due to the assumption of a constant Smagorinsky coefficient for the calculation of the turbulent viscosity.

There are different ways to overcome the near the wall problematic such as performing semi-empirical damping functions[26], dynamic procedures[27], and shear-improved dissipative modeling[18]. These models lead to a better representation of the turbulence flow near the wall but increase the computational cost of the simulation and complexity of the code. Luckily, there exist other methods that avoid the usage of the damping function or dynamics procedures such as the Dynamic Smagorinsky[14], the wall-adapting local eddy-viscosity (WALE)[23], and Vreman models[31]. These methods have a reduced increase in the computational cost of the simulation compared with the previous ones.

An attractive alternative is the coherent structure model (CSM), where Kobayashi is one of the leading researches[19]. The main advantage of the CSM is the local determination of the SGS model parameters, faster computation, and no need for wall functions. These main advantages are the main reason to implement the CSM and its variants in the lbmpy platform.
CHAPTER 3
THE COHERENT STRUCTURE MODEL (CSM)

To be able to understand the theory behind the coherent structure model (CSM), we must recall the large eddy simulation (LES) theory. Besides, this insight will provide a base to understand how LES can be applied to LBM, and open the door to apply different LES variants to LBMs.

The significant implication of using the LES approximation to LBM is the possibility to modify the viscosity at every lattice. Thus, the dynamic of the particle distribution function has a space-dependent relaxation. This approach improves the stability of the LBM for high Reynolds numbers, which is our goal.

3.1 LARGE EDDY SIMULATION (LES)

In the large eddy simulation (LES), a spatial filtering operation is introduced:

\[ \bar{w}(x) = \int w(x)G(x, x')dx' \] (3.1)

where \( w \) can be any physical quantity such as density and velocity; \( G \) is a given spatial filter function and the integral is extended over the entire domain. There exist diverse filter functions to apply. For most finite difference methods, the box filter is assumed and is defined as follows:

\[ G_i(x_i, x'_i) = \begin{cases} \frac{1}{\Delta_i}, & \text{for } |x_i - x'_i| < \frac{\Delta_i}{2} \\ 0, & \text{otherwise} \end{cases} \] (3.2)

Applying the same filter operation to the Navier-Stokes equation for incompressible flows, we have:

\[ \frac{\delta \bar{p}_i}{\delta x_i} = 0 \] (3.3)

\[ \frac{\delta \bar{p}_i}{\delta t} + \bar{u}_j \frac{\delta \bar{p}_i}{\delta x_j} = -\frac{1}{\rho} \frac{\delta p}{\delta x_i} - \frac{\delta \tau_{ij}^{uv}}{\delta x_j} + \frac{\delta}{\delta x_j} \left( \nu \left[ \frac{\delta \bar{p}_i}{\delta x_j} + \frac{\delta \bar{p}_j}{\delta x_i} \right] \right) \] (3.4)
where $\tau_{ij}^v$ are the Reynolds stresses, which represent the effects of the unresolved scales on the resolved scales:

$$
\tau_{ij}^v = \bar{u}_i \bar{u}_j - \bar{u}_i \bar{u}_j \quad (3.5)
$$

For laminar flows the Reynolds stress ($\tau_{ij}^v$) is negligible, but for turbulent flows, it is significant and must be modeled. By using the Boussinesq approximation, we obtain the following:

$$
\tau_{ij}^v = -2\nu_{SGS} \overline{S_{ij}} \quad (3.6)
$$

where the term $\overline{S_{ij}}$ is the filtered strain rate tensor, and $\tau_{ij}^v$ is the eddy viscosity. The strain tensor has the following formulation:

$$
\overline{S_{ij}} = \frac{1}{2} \left( \frac{\delta \bar{u}_j}{\delta x_i} + \frac{\delta \bar{u}_i}{\delta x_j} \right) \quad (3.7)
$$

Now, as the LES model is defined, the new question is how to modify the Lattice Boltzmann method to simulate the filtered density and velocity equations. To perform this, let us introduce the filtered particle distribution $\overline{f_i}(x)$, defined as follows:

$$
\overline{f_i}(x) = \int f_i(x) G(x, x') dx' \quad (3.8)
$$

and modify the microscopic equation (2.2) to be a kinetic equation for the filtered particle distribution function:

$$
\frac{\delta \overline{f_i}}{\delta t} + c_i \cdot \nabla \overline{f_i} = \overline{\Omega_i} \quad (3.9)
$$

by applying a finite difference approach, we obtain:

$$
\overline{f_i}(x + e_i, t + 1) = \overline{f_i}(x, t) + \overline{\Omega_i(f(x, t))} \quad (3.10)
$$

One approach to solve the filtered function is to link the collision steps with some local information.
and abandon the single-relaxation approximation. This procedure is performed by Hou [17], leading to the total viscosity equation:

$$\nu_{\text{total}} = \nu_0 + \nu_{\text{SGS}}$$  \hspace{1cm} (3.11)$$

where the $\nu_{\text{total}}$ is the total viscosity, $\nu_0$ is the physical kinematic viscosity (bulk viscosity), and $\nu_{\text{SGS}}$ is the eddy-viscosity term. In this way, we can express the relaxation time, $\tau$ as function of the total viscosity:

$$\frac{2\tau_{\text{total}} - 1}{6} = \nu_{\text{total}}$$  \hspace{1cm} (3.12)$$

therefore:

$$\tau_{\text{total}} = 3\nu_{\text{total}} + \frac{1}{2}$$  \hspace{1cm} (3.13)$$

or

$$\tau_{\text{total}} = 3(\nu_0 + \nu_{\text{SGS}}) + \frac{1}{2}$$  \hspace{1cm} (3.14)$$

To define the $\nu_{\text{SGS}}$, we need to select a subgrid-scale model (SGS).

### 3.2 SMAGORINSKY MODEL- LBM-LES

The best-known eddy-viscosity model bears the name of Joseph Smagorinsky. In the Smagorinsky model, the eddy viscosity is expressed as the characteristic scale $\Delta$ times a velocity scale $\Delta|\mathbf{S}|$ where $\mathbf{S}$ represents a Galilean invariant estimation of velocity differences over length-scales of order $\Delta$. The model contains a dimensionless empirical parameter (the so-called Smagorinsky coefficient $c_s$); therefore, the eddy viscosity takes the following form:

$$\nu_{\text{SGS}}^{\text{smagorinsky}} = (c_s \Delta)^2 |\mathbf{S}|$$  \hspace{1cm} (3.15)$$

and in consequence:

$$\tau_{ij}^{\text{smagorinsky}} = -2(c_s \Delta)^2 |\mathbf{S}_i|\mathbf{S}_j$$  \hspace{1cm} (3.16)$$
where \(|\mathbf{S}| = \sqrt{2S_{ij}S_{ij}}\) is the magnitude of the large scale strain rate tensor.

Thus, the relaxation rate should be locally adjusted, depending on the local magnitude of the large-scale strain rate tensor. The constant \(c_s\) can be modeled dynamically, leading to the dynamic Smagorinsky model [14].

One main advantage of the LBM with the Smagorinsky model is the local calculation of the strain rate tensor, which is obtained from the nonequilibrium properties of the filtered particle distribution:

\[
S_{ij} = -\frac{3\tau_{\text{total}}}{2\rho} \Pi_{ij}^{(\text{neq})}
\]

where \(\Pi_{ij}^{(\text{neq})}\) is the second order moment tensor of the non-equilibrium part of the distribution functions \(f^{(\text{neq})} = f - f^{(\text{eq})}\) and can be computed as

\[
\Pi_{ij}^{(\text{neq})} = \sum_q c_q c_q f_q^{(\text{neq})}
\]

This means that we have the following:

\[
|\mathbf{S}| = \frac{3\Pi}{2\rho} \frac{1}{\tau_{\text{total}}} = \frac{3\Pi}{2\rho} \frac{1}{3(\nu_0 + (c_s \Delta)^2 |\mathbf{S}|)} + 1/2
\]

where \(\Pi\) is the intensity of \(\Pi_{ij}^{(\text{neq})}\).

Solving for the strain rate tensor \(|\mathbf{S}|\) considering incompressible flow (\(\rho = 1\)) and the filter size equal to a lattice unit (\(\Delta = 1\)):

\[
|\mathbf{S}| = \frac{1}{12c_s^2} \left( -6\nu_0 + \sqrt{72c_s^2 \Pi + 36\nu_0^2 + 12\nu_0 + 1} - 1 \right)
\]

replacing \(\tau_0 = 3\nu_0 + \frac{1}{2}\):

\[
|\mathbf{S}| = -\frac{\tau_0}{6c_s^2} + \frac{1}{12c_s^2} \sqrt{72c_s^2 \Pi + 4\tau_0^2}
\]
finally the total relaxation time is:

$$\tau_{total} = \frac{\tau_0}{2} + \frac{1}{2} \sqrt{18c_s^2 \Pi + \tau_0^2}$$  \hspace{1cm} (3.22)$$

The LBM with the Smagorinsky model, from now named as LBM-LES, has the great advantage to calculate the strain tensor locally. Thus, not reducing the high-parallelizable nature of the LBM and its performance. However, the fact that the Smagorinsky constant is unique in the whole domain introducing the difficulty to choose the right one. Moreover, the Smagorinsky constant is dependent on the geometry.

### 3.3 COHERENT STRUCTURE

The coherent structure model [19] has a spatial dynamic constant $c_s$ calculated using a function of the velocity gradient tensors, in contrast to the LBM-LES. This function is composed of a model parameter $C_{csm}$ and a coherent structure function $F_{cs}$.

$$\nu_{SGS}^{csm} = C_{csm} \Delta^2 |\mathbf{S}|$$

$$F_{cs} = \frac{Q}{E} \hspace{1cm} (-1 \leq F_{cs} \leq 1) \hspace{1cm} (3.23)$$

Where $Q$ is the second invariant of the grid scale flow field normalized and $E$ is the magnitude of a velocity gradient tensor, which plays a role in wall-damping near the wall boundary:

$$Q = \frac{1}{2} (\overline{W_{ij} W_{ij}} - \overline{S_{ij} S_{ij}}) = \frac{1}{2} \left( \frac{\delta u_j}{\delta x_i} \frac{\delta u_i}{\delta x_j} \right)$$  \hspace{1cm} (3.24)$$

$$E = \frac{1}{2} (\overline{W_{ij} W_{ij}} + \overline{S_{ij} S_{ij}}) = \frac{1}{2} \left( \frac{\delta u_j}{\delta x_i} \right)^2$$  \hspace{1cm} (3.25)$$

where $\overline{S_{ij}}$ and $\overline{W_{ij}}$ are the strain rate tensor and the vorticity tensor in a grid scale flowfield, respectively.

There exist two variants in the coherent structure models, non-rotational coherent structure model (NRCSM) and coherent structure model (CSM), which takes into account an energy suppression
function \( F_{\Omega} \).

NRCSM:

\[
C_{NRCSM} = C'|F_{cs}|^{3/2}, \quad C' = \frac{1}{20}
\]  

CSM:

\[
C_{CSM} = C''|F_{cs}|^{3/2}F_{\Omega}, \quad C'' = \frac{1}{22}, \quad F_{\Omega} = 1 - F_{CS} \quad (0 \leq F_{\Omega} \leq 2)
\]  

The primary challenge for the implementation of the coherent structure model is the computation of the vorticity tensor \( \overline{W_{ij}} \). As explained in the Smagorinsky model, we can obtain the strain rate tensor by the second order moment tensor of the non-equilibrium part of the distribution functions, but not the vorticity tensor.

Since the LBM present a relatively small step, we can assume that the change of the gradients between two steps is not significant. Thus, the impact in the model parameter \( C_{csm} \) is small. We have the option to recover relevant information from the previous time step and the current time step or purely from the previous time step for the construction of the velocity gradient tensor.

With this concept, we are able to obtain the different gradients of the velocity gradient tensor from the strain tensor at the current time step and complete them with information of the previous time step. As an example, a 2D scenario will be employed.

The velocity gradient tensor is defined as:

\[
\nabla u = \overline{S_{ij}} + \overline{W_{ij}}
\]  

\[
\begin{bmatrix}
\delta_x \overline{u_x} & \delta_y \overline{u_x} \\
\delta_x \overline{u_y} & \delta_y \overline{u_y}
\end{bmatrix} = 
\begin{bmatrix}
S_{xx} & S_{xy} \\
S_{xy} & S_{yy}
\end{bmatrix} + 
\begin{bmatrix}
0 & \omega \\
-\omega & 0
\end{bmatrix}
\]  

where:

\[
S_{xy} = \delta_x \overline{u_y} + \delta_y \overline{u_x}
\]  

\[
\omega = \delta_x \overline{u_y} - \delta_y \overline{u_x}
\]
Since we have the values of the strain tensor at the current time step for the calculation of its intensity, we can recover the gradients from its diagonal. However, we still need to obtain the values of $\delta_x u_y$ or $\delta_y u_x$ to complete the velocity gradient tensor. To do this, we can perform a finite difference approach using the velocity values from the previous time step, which are already calculated. In this case, we use a second order approach (central difference scheme):

$$
\delta_x u_y = \frac{u_y^E - u_y^W}{2\Delta x}, \quad \delta_y u_x = \frac{u_x^N - u_x^S}{2\Delta y}, \quad \Delta_x = \Delta_y = 1
$$

With this information, we can calculate $\overline{Q}$ and $\overline{E}$ directly. We must note that $\overline{Q}$ and $\overline{E}$ are obtained using the velocity values of the previous step and $\overline{S}_{ij}$ at the current time step. With this information, we can obtain the coherent structure function $F_{cs}$ and finally, the model parameter $C_{csm}$.

This approach takes into consideration that the $\Delta t$ is small in the LBM and the change of the gradient between two-time steps is not that significant, and the impact in the model parameter $C_{csm}$ is small.

Another approach is to calculate all the values of the velocity gradient tensor in the previous step time step by finite difference or by using another interpolation technique. One must take into account that these approaches may impact the performance of the code and make it slower.

For the purpose of this thesis, we implement a full finite difference method and identify it with FD letters on its LBM name. Thus, we can compare the two approaches and get an insight into the efficiency of the different approaches.
CHAPTER 4
CSM IMPLEMENTATION IN LBMPY

After understanding the CSM and NRCSM, the implementation of them in an LBM framework can be done. At first, the definition of which framework is the most suitable for our application is required.

There exist various LBM frameworks, such as Palabos and OpenLB, with great support for parallel programming on CPUs (single node OpenMP or multiple nodes MPI). However, not all of the LBM frameworks support GPUs (CUDA or OpenCL).

The framework waLBerla is one of the few that support CUDA, OpenMP, and MPI parallelization, and is therefore used for further investigations [12]. The framework of waLBerla is being developed at the Friedrich-Alexander-Universität Erlangen-Nürnberg at the chair of Informatik 10. Besides waLBerla, the chair of Informatik 10, is currently developing lbmpy [10] as an extension to it. lbmpy uses code generation based on the pystencils package [11].

lbmpy was developed to overcome a significant problem with all the LBM frameworks: it is almost impossible to have a generic and fast LBM code in a language like C++. The main problem arises due to the large number of different LBMs, such as variants with different collision operators, stencil, entropic models, turbulence models, different moment space transformations, and other characteristics of the LBM, not even to mention different hardware optimizations. Even with the possibility to realize abstractions (for example, static polymorphism), it is hard to write a readable and easy to maintain code. Thus, the possibility to use metaprogramming to generate LBM kernels is attractive.

4.1 PYSTENCILS

pystencils [11] is a code generator to speed up computations on numpy arrays [1] with particular emphasis in stencil codes. pystencils generates a C code, compiles it, and let us call the compiled C
function in Python as a native function of it. Besides, it can compile in different languages, such as CUDA for GPUs, with different optimizations and specifications.

One of the most attractive characteristics of pystencils is the symbolic representation of stencil codes. For this purpose, pystencils use the symbolic mathematics library of Python (SymPy) [2], which is similar to Mathematica. SymPy is a computer algebra system (CAS) that manipulates mathematical expressions in symbolic form. Thus, it can manipulate symbolic expression with variables, where mathematical objects are represented exactly. For example, we can represent a mathematical expression as:

```python
>>> import sympy as sp
>>> x, y = sp.symbols('x y')
>>> expr = x**2 + 2*y*x
x**2 + 2*y*x

Where each variable are defined as symbols in SymPy. Besides, the mathematical expression can be further manipulated; for example, we can add $y^2$:

```python
>>> expr + y**2
x**2 + 2*y*x + y**2
```

We can also expand and factorize the mathematical expression:

```python
>>> sp.factor(expr + y**2)
(x + y)**2
```

Moreover, we can replace a variable for another mathematical expression or by a real number:

```python
>>> expr.subs(y, cos(x))
x**2 + 2*x*cos(x)
```
Besides, *pystencils* use SymPy to manipulate stencil codes with the addition of **Fields**. **Fields** represent a multidimensional array, where some dimensional are spatial and others index or stored values on the cell. Thus, **Fields** allows a spatial interaction between neighbors in the different indexes.

As an example, how to solve a 2D steady heat equation with *pystencils* will be explained. First of all, the steady heat equation has the following equation:

\[ \nabla^2 T = \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \]  

(4.1)

Using a central difference scheme to discretize equation (4.1), we obtain:

\[ \frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{\Delta x^2} + \frac{T_{i,j-1} - 2T_{i,j} + T_{i,j+1}}{\Delta y^2} = 0 \]  

(4.2)

Where \( i \) and \( j \) are indexed along the \( x \) and \( y \), respectively. Considering equal spacing in \( x \) and \( y \) direction (\( \Delta x = \Delta y \)) and clearing equation (4.2) for \( T_{i,j} \):

\[ T_{i,j} = \frac{T_{i-1,j} + T_{i+1,j} + T_{i,j-1} + T_{i,j+1}}{4} \]  

(4.3)

Thus, we obtain the Jacobi stencil, where the update rule uses the four neighbors (North, South, East and West) as shown in the figure 4.1:

For the implementation in *pystencils*, we require to specify the domain (100 x 100) cells randomly initialized:

```python
>>> from pystencils.session import *
>>> input_array = np.random.rand(100,100)
>>> output_array = np.zeros_like(input_array)
```

Now we can define our **Fields** for the input and output array. In the first place, the spatial dimensions are equal to the domain, and it has only one index (temperature). We have two ways to do this:
The first command creates the field directly from the numpy array, without the need to specify zero indexes or if it is 2D or 3D. The second specifies the name of the field (DST), no indexes, and that is 2D, besides it specify that the fields refer to the numpy array (output array).

Afterwards, we need to create the update rule (stencil code):

```python
>>> jacobi = ps.Assignment(dst[0,0](0),
   (src[1,0]+src[-1,0]+src[0,1]+src[0,-1])*0.25)
```

As observed, the Fields locate their neighbors by a compass, where west or east is the first index, north and south is the second index, and the third index represents top or bottom.

The last step is to create the kernel and execute it:

```python
>>> kernel = ps.create_kernel(jacobi).compile()
>>> kernel(src=input_array, dst=output_array)
```
4.2 LBMPY

lbmpy [10] is a module that makes heavy use of pystencil [11]. It presents diverse predefined 2D and 3D scenarios for fast implementation, for example, the lid-driven cavity, full periodic flows, channel flow with extension to a pipe. Besides, different tools to modify the boundaries conditions, for example, locate obstacles in the domain. However, one can aim to generate a LBM scenario from scratch and modify it as required.

The different variables of the LBM, such as mean velocity, particle distribution functions, and density, are treated as Fields and can be accessed and modified. On the other hand, the collision operators are treated symbolically and can be modified. Thus, allowing versatility in the generation of scenarios with different LBM models.

The generated kernel can also be compiled in CUDA and OpenMP with different optimization options. Thus, the kernels can make full use of the hardware and have a higher performance. Besides, the generated compiled kernel can be used in other frameworks as waLBerla.

lbmpy has different LBM implemented, such as the SRT, MRT, LBM-LES with Smagorinsky turbulence modeling, and KBC models. Since lbmpy is a code generator, it is possible to generate benchmarks with the same geometry with the different LBM models and run it in the Python. Thus, lbmpy has great advantages compared to other LBM frameworks.

Besides, the LBM kernel function is created in four steps:

- The creation or definition of the method, where it is specified, for example, how are the collision threatened (relaxation rates and moment space), which stencil are we using, between others.

- Collision and update rule, here the collision are specified and update rules symbolically. Thus, in this step, the collision rules are modified, if specified, for example, for the Smagorinsky model with an SRT.

- Creation of the AST, create the structure of the kernel, considering all the information specified.
• Creation of the function, compile process according to the optimization parameters specified, for example in CUDA for NVIDIA GPUs.

Thus, to implement the CSM and NRCSM, we work in the collision and update rule step, with the symbolic collision rules.

In order to understand more how lbmpy works, we are going to make an example. In this case, we will use a 2D lid cavity flow with the flowing characteristics:

• 100 lattice per side, thus domain size (100, 100)
• Stencil D2Q9
• Method SRT
• Relaxation rate of 1.98 ($\omega_0 = 1.98$)
• Lid velocity of 0.025 (in lattice units)
• Optimization in a GPU with a floating point precision

The first step is to create the method with the stencil, the LBM to use, and the relaxation rate:

```python
>>> from lbmpy.session import *

>>> method = create_lb_method(method='srt', stencil='D2Q9', relaxation_rate=1.98)
```

method has different properties, which we can verify. For example, if we want to know which moments are we using:

```python
>>> method.moments
(1, x, y, x^2, y^2, xy, x^2 y, y^2 x, x^2 y^2)
```

or the relaxation rates:
>>> method.relaxation_rates
(1.98, 1.98, 1.98, 1.98, 1.98, 1.98, 1.98, 1.98)

We can also obtain the stencil and moment matrix.

The next step is to create the collision and update rule:

>>> collision_rule = create_lb_collision_rule(lb_method=method)

collision_rule contain the collision rules in a symbolic way (expressions). Thus, we can modify the expressions, by replacing variables or by adding new expressions.

The next step is to create the LBM function:

>>> lb_step=create_lid_driven_cavity(domain_size=(100,100),
          collision_rule=collision_rule, lid_velocity=0.025,
          optimization = {'target': 'gpu',
          'double_precision' : False})

Once the LBM object is created we can run it with:

>>> lb_step.benchmark_run(100);

A straightforward method to create the LBM function is to call directly:

>>> lb_step=create_lid_driven_cavity(domain_size=(100,100),
          method='srt', stencil='D2Q9', relaxation_rate=1.98,
          lid_velocity=0.025,optimization = {'target': 'gpu',
          'double_precision' : False})
4.3 IMPLEMENTATION OF CSM IN LBMPY

For the implementation in lbmpy, we need to modify the collision operator and integrate the spatial relaxation of the CSM. Following the chapter three, we can get the equation for the viscosity for the CSM and NRCSM.

\[

\nu_{\text{total}} = \nu_0 + \nu_{\text{NRCSM}} = \nu_0 + C_{\text{NRCSM}} (\Delta)^2 |S| = \nu_0 + \frac{1}{20} \left( \frac{Q}{E} \right)^{3/2} (\Delta)^2 |S| \quad (4.4)
\]

\[

\nu_{\text{total}} = \nu_0 + \nu_{\text{CSM}} = \nu_0 + C_{\text{CSM}} (\Delta)^2 |S| = \nu_0 + \frac{1}{22} \left( \frac{Q}{E} \right)^{3/2} \left( 1 - \frac{Q}{E} \right) (\Delta)^2 |S| \quad (4.5)
\]

where \( \nu_0 \) is the bulk viscosity, and the filter length \( \Delta \) is chosen as one lattice.

From the equations 4.4 and 4.5, one can observe that it is needed to calculate the second invariant \( (Q) \) and velocity gradient \( (E) \), which depends on the strain tensor \( (S_{ij}) \) and vorticity tensor \( (\Omega_{ij}) \), and the intensity of the local strain tensor \( (S) \).

As a first step, the strain tensor and the vorticity tensor are calculated. Luckily the LBM can obtain the strain tensor from the non-equilibrium part of the distribution function, from equation 2.33:

\[

\overline{S_{ij}} = -\frac{3}{2\rho(0)} \Pi_{ij}^{(\text{neq})} = -\frac{3\omega}{2\rho(0)} \Pi_{ij}^{(\text{neq})} \quad (4.6)
\]

where \( \omega = 1/\tau \) is the relaxation rate, \( \rho(0) \) is the density, which take value of one for incompressible flows, and \( \Pi_{ij}^{(\text{new})} \) is the second order moment tensor of the non-equilibrium part of the distribution functions \( f^{(\text{neq})} = f - f^{(\text{eq})} \). Thus:

\[

\Pi_{ij}^{(\text{new})} = \sum_q c_{qij} f_q^{(\text{neq})} = \sum_q c_{qij} \left( f^{(\text{eq})} - f^{(\text{neq})} \right) \quad (4.7)
\]

However, the vorticity tensor cannot be computed locally. Thus, it is required to do a non-local method to obtain the vorticity tensor, as explained in chapter three. Therefore, we have to access to the Field that will store the mean velocity in every iteration.

As a first step, we explore the possibility to calculate the total relaxation time in base of the known variables. Since we can take advantage of the symbolic representation of mathematical expressions
of SymPy, we will use it. Thus, the relevant variables for the equations are created by:

```python
>>> from lbmpy.session import *
>>> from lbmpy.relaxationrates import *
>>> τ₀, ω = sp.symbols("tau_0 omega", positive=True, real=True)
>>> ν₀, C_csm, S, Π = sp.symbols("nu_0, C_CSM, |S|, Pi",
positive=True, real=True)
```

If the flow is considered incompressible, the density is equal to one ($\rho = 1$). Thus, from equation 4.6 the intensity of the filtered strain tensor ($\mathbf{S}$) is equal to:

$$|\mathbf{S}| = \frac{3\Pi}{2}\omega$$

(4.8)

Note that the minus sign is taken out since the absolute value of both tensors is taken. Then, we can symbolically create equation 4.8 by:

```python
>>> Seq = sp.Eq(S, 3*ω/2*Π)
```

Since the filter length is equal to one, the equation 4.5 is reduced to:

$$\nu_{total} = \nu_0 + \nu_{CSM} = \nu_0 + C_{CSM}|\mathbf{S}|$$

(4.9)

By applying equation 3.14 in 4.9, the relaxation time and relaxation rate are obtained:

$$\tau_{total} = \frac{6C_{CSM}|\mathbf{S}| + \nu_0 + 1}{2}$$

(4.10)

$$\omega_{total} = \frac{2}{6C_{CSM}|\mathbf{S}| + \nu_0 + 1}$$

(4.11)

By applying equation 4.11 in 4.8, an equation for the intensity of the filtered strain tensor ($\mathbf{S}$) with known values is obtained. Note that $\nu_0$ is obtained from the original relaxation time $\tau_0$, and since the $C_{CSM}$ is obtained with the assumption of chapter three, it is a known value. Besides, we use the python function `relaxation_rate_from_lattice_viscosity` from `lbmpy` to obtain equation 4.11.
\[ |\mathbf{S}| = \frac{3\Pi}{6C_{CSM}|\mathbf{S}| + \nu_0 + 1} \] (4.12)

Then, equation 4.12 is created symbolically by:

```python
>>> Seq2 = Seq . subs ( \omega,
       relaxation_rate_from_lattice_viscosity ( \nu_0 + C_{csm}*S ) )
```

```python
>>> Seq2
|\mathbf{S}| = \frac{3\Pi}{6C_{CSM}|\mathbf{S}| + \nu_0 + 1}
```

A significant advantage of SymPy is that it can solve equation symbolically, without the need to rewrite the equation. Thus, the equation 4.12 can be solved in SymPy and the equation for the intensity of the filtered strain tensor \(|\mathbf{S}|\) is:

```python
>>> solveRes = sp . solve ( Seq2, S )
>>> SVal = solveRes [ 0 ]
>>> SVal
\frac{1}{12C_{CSM}} (-6\nu_0 + \sqrt{72C_{CSM}\Pi + 36\nu_0^2 + 12\nu_0 + 1} - 1)
```

The solution can be more straightforward \(\nu_0\) is replaced by \(\tau_0\), using equation 2.35 (python function `lattice_viscosity_from_relaxation_rate` from `lmbpy`):

```python
>>> SVal = Sval . subs ( \nu_0,
       lattice_viscosity_from_relaxation_rate (1/\tau_0)) . expand ()
>>> SVal
-\frac{\tau_0}{6C_{CSM}} + \frac{1}{12C_{CSM}} \sqrt{72C_{CSM}\Pi + 4\tau_0^2}
```

After solving the intensity of the filtered strain tensor \(|\mathbf{S}|\), the total relaxation time \((\tau_{total})\) from the equation 4.9 can be computed symbolically:

```python
>>> \tau_{total} = 1/(relaxation_rate_from_lattice_viscosity (}
```
\[ \text{lattice_viscosity_from_relaxation_rate}(1/\tau_0) + C_{\text{csm}}*\text{SVal}) \]. \text{cancel()} \]

\[
\tau_{\text{total}} = \frac{\tau_0}{2} + \frac{1}{2}\sqrt{18C_{\text{CSM}}\Pi + \tau_0^2}
\]

Thus, total relaxation time depends only on the \( C_{\text{CSM}}, \tau_0, \) and \( \Pi \). However, those variables have to be calculated in every step, and the following functions are defined to calculate the second order moment tensor \( \Pi^{(neq)}_{ij} \) and the Frobenius norm for the intensity of the filtered strain, vorticity, and velocity gradient tensors.

```python
def second_order_moment_tensor(function_values, stencil):
    assert len(function_values) == len(stencil)
    dim = len(stencil[0])
    return sp.Matrix(dim, dim, lambda i, j: sum(c[i] * c[j] * f for f, c in zip(function_values, stencil)))

def frobenius_norm(matrix, factor=1):
    return sp.sqrt(sum(i*i for i in matrix) * factor)
```

Several variables are needed to be created, such as the vorticity tensor and the velocity gradient tensor.

Afterwards, the equations detailed in chapter three for the CSM are created in a function coherent_structure_equations. Those equations will complement the collision rules of the desired LBM, such as SRT. The function has as arguments the relaxation rates in the bulk and total, the method, and the Field with the mean velocities. With this information, it performs the following:

- Creates the non-equilibrium part of the distribution functions, the second order moment tensor, and the vorticity tensor with the same shape of SIJ.
- The strain tensor is created, and the elements of the vorticity
- The second invariant and the velocity gradient tensor is created.
- The velocity gradient matrix is calculated with values according to chapter three for full finite
difference or our approximation by adding equations to the collision rule. In this case, our method is used.

- The equation for the CSM model are added to the collision rule, relaxation time in the bulk \((\tau_0)\), the intensity of the velocity gradient tensor \((E)\), intensity of the second invariant \((Q)\), the \((FCS)\) variable with its restriction, the intensity of second-order moment tensor \((II)\), the \((C_{CSM})\) constant, and the total relaxation rate \((\omega_{total})\).

The same procedure can be done for the NRCSM and the method with the full finite approximation for NRCSM and CSM.

### 4.4 USAGE OF CSM IN LBMPY

In order to use the CSM and NRCSM in an LBM model, the following steps need to be done:

- Create the Field to store the mean velocities and name it as "vel_out"
- Create an LBM method
- Get the collision rules and modify them (add the collision rules for the CSM or NRCSM with or without full finite approximation)
- Create an LBM function in lbmpy and add the collision rules. Besides, specify the characteristics of the simulation, and specify that in every step, the mean velocity has to be computed and named with the same name of the Field previously created.

In order to understand how to add the CSM model, we are going to use the previous 2D lid cavity simulation and add the CSM collision rules.

As a first step, we need to get access to the velocity field through:

```python
>>> dh = create_data_handling(domain_size=(100,100))
>>> vel_field = dh.add_array("vel_out", values_per_cell=2, gpu=True, cpu=True)
```

In the next step, we need to create the method with a symbolic relaxation rate \(\omega\):
method = create_lb_method(method='srt', stencil='D2Q9',
                           relaxation_rate=\omega)

Then, we create the collision and update rules:

>>> collision_rule = create_lb_collision_rule(lb_method=method)

Now we can modify the collision rules:

>>> collision_rule = collision_rule.new_with_substitutions({\omega: \omega_{total}})
>>> collision_rule.subexpressions+=coherent_structure_equations(\omega,
                                                             \omega_{total}, method, vel_field)
 collision_rule.topological_sort(sort_subexpressions=True,
                              sort_main_assignments=False)

At last, the LBM scenario of the lid cavity is created:

>>> lb_step = create_lid_driven_cavity(data_handling=dh,
                                    collision_rule=collision_rule, lid_velocity=0.025,
                                    kernel_params={"omega": relaxation_rate},
                                    velocity_data_name='vel_out',
                                    compute_velocity_in_every_step=True,
                                    optimization = {'target': 'gpu',
                                                   'double_precision': False})

Once the LBM scenario is created, it can be run as follows to obtain various parameters as performance in MLUPs and plots of the velocity field magnitude:

>>> lb_step.benchmark_run(60000);
>>> plt.vector_field_magnitude(lb_step.velocity[:,:,])
However, those steps are quite long and laborious. Thus, the CSM and NRCSM with and without full finite difference (FD) has been implemented and can be called as following:

```python
>>> lb_step = create_lid_driven_cavity(data_handling=dh,
                                       coherent_structure=vel_field, CSM=True, FD=False,
                                       lid_velocity=0.025, method='srt', relaxation_rate=1.98,#
                                       velocity_data_name='vel_out', compute_velocity_in_every_step=True,
                                       optimization = {'target' : 'gpu',
                                           'double_precision' : False})
```

where the **Field** of mean velocity is passed with the **coherent_structure**, the argument CSM specify the usage of CSM (True) or NRCSM (False), and the argument FD defines the usage of the full finite difference approximation.

The CSM and NRCSM with or without finite difference approximation has been implemented in the file of *turbulence_models.py* in the *lbmpy*. 
CHAPTER 5
SIMULATIONS AND VERIFICATION OF THE CSM

After the implementation of the CSM and NRCSM in lbmpy, it is necessary to verify its implementation and to perform some benchmarks. As one of the main goals of this thesis is the simulation of turbulent flows, the two main benchmarks will address turbulent and wall bounded flows. For the benchmarks, besides CSM and NRCSM, the different LBMs presented and discussed before will be also used for the calculations and serve as a basis for further comparison.

The first benchmark is the Kida vortex simulation [6], which represents the turbulence decay a full periodic flow with a deterministic initial conditions (under-resolved turbulence setup), this means without walls or force. In this benchmark, the CSM and NRCSM are investigated with particular emphasis on numerical stability and performance. The different CSM and NRCSM are compared with the KBC, MRT, and the LBM-LES (Smagorinsky).

The second benchmark is the standard turbulent channel flow [22], which presents a wall with a non-slip condition and force drive. This benchmark addresses the interaction directly with wall-bounded flows and moderate to high Reynolds numbers.

Moreover, all the simulations are performed with an NVIDIA Quadro P500 with the following characteristics:

5.1 EVALUATION PARAMETERS

The LBM simulations are compared regarding accuracy and performance. In this way, we can rank different LBM and choose the correct model for our purposes. For example, for wind engineering and outdoor flow simulations, deviations in accuracy between 10% and 25% for the main characteristics of the flow are acceptable for a low computing cost [5]; especially, if the aim is real-time simulations, the computing cost is relevant, and a rough accuracy is acceptable. However, for other applications, such as aerospace engineering, the main target is to deviations in accuracy of less than 5%[5].
The accuracy gives us the difference between calculated values and the real physical results. In our case, we compare the calculated values with values from experiments (experimental data), analytical solutions, or high resolved simulations, such as DNS or high resolved LBM simulations, where the smallest turbulent scales are calculated. For this purpose, we define the absolute error ($\epsilon$), the relative error ($\eta$), and the percentage error ($\%\text{error}_{\text{avg}}$) as:

$$\epsilon = |v - v_{\text{sim}}| \quad (5.1)$$

$$\eta = \frac{\epsilon}{|v|} = \frac{|v - v_{\text{sim}}|}{|v|} \quad (5.2)$$

$$\%\text{error}_{\text{avg}} = 100\% \times \eta = 100\% \times \frac{|v - v_{\text{sim}}|}{|v|} \quad (5.3)$$

Where $v$ is the true value, and $v_{\text{sim}}$ is the simulation outcome. Since we can obtain different relative errors, according to our measuring points (such as time or position), we can define for example a maximum relative error ($\eta_{\text{max}}$) and an average relative error ($\eta_{\text{avg}}$). This procedure can also be done for ($\epsilon$ and $\%\text{error}_{\text{avg}}$).

On the other hand, the performance allows us to compare the different LBM regarding the speed of the code with a specific metric. The most common metric for LBM codes is million lattices units per second (MLUPs):

$$MLUPs = \frac{lattice \ updates}{10^6 \times s} \quad (5.4)$$
This metric measures the number of lattice units/cells that are updated per second. In case we need to compute the amount of time required to simulate one step, we need to divide it by the total number of cells in the domain of the simulation \((N = N_x \times N_y \times N_z)\), and for the real-time rate with the following formula:

\[
RT_{rate} = \frac{C_t \times MLUPs}{N}
\]

The performance of the code depends on different parameters such as the compiler, OS parameters, BIOS settings, memory interface, size and speed, the architecture used, and others. In our simulations, all of them are run in the NVIDIA GPU (QUADRO P5000) with a single-precision floating-point.

### 5.2 KIDA VORTEX

The Kida Vortex is a classic 3D benchmark flow which evolves from a simple deterministic and symmetric initial conditions to a state which resembles a fully developed turbulent flow. The initial conditions for the flow field are given by [6]:

\[
\begin{align*}
    u_x(x, y, z) &= U_0 \sin(x)(\cos(3y)\cos(z) - \cos(y)\cos(3z)) \\
    u_y(x, y, z) &= U_0 \sin(y)(\cos(3z)\cos(x) - \cos(z)\cos(3x)) \\
    u_z(x, y, z) &= U_0 \sin(z)(\cos(3x)\cos(y) - \cos(x)\cos(3y))
\end{align*}
\]

where \(x, y, z \in [0, 2\pi]\) and all the directions are imposed with periodic boundary conditions. Besides, the Reynolds number in the Kida vortex is defined as:

\[
Re = \frac{U_0 N}{\nu}
\]

where \(N\) is the domain size.

The primary parameters of comparison percentage errors \((%\text{error}_{avg})\) of the enstrophy \((\Omega)\) and kinetic energy evolution \((k)\), defined as:

\[
\Omega = \frac{1}{2} \int \omega^2 \]
where vorticity ($\omega$):

$$\omega = \nabla \times u$$  \hspace{1cm} (5.9)

here $\overline{u}$ represents the average velocity and $u'$ the fluctuating part of the velocity:

$$u = \overline{u} + u'$$  \hspace{1cm} (5.10)

for the Kida vortex, $\overline{u} = 0$.

The main parameter of comparison is the enstropy, in specific the enstropy peak. The enstropy reflects the dissipative effects that arise from the vortex formation activity. The Kida vortex presents a particular enstropy evolution, increasing until the maximum vortex formation (enstropy peak) and then decaying. The enstropy peak indicates large gradients, which are numerically challenging, and is a good indicator for stability and accuracy of the simulations.

Thus, the main parameter of comparison is percentage error in the enstropy peak. Other parameter is the percentage error average in the whole time ($\%\text{error}_{\text{avg}}$).

The first set of simulations are set up with $U_0 = 0.1$, Reynolds number of 6,000, and diverse domain sizes ($N \in [50, 100, 200, 300]$). The outcomes of the simulations are contrasted versus the resolved LBGK simulation of Bösch with a Reynolds number of 6,000 [6], where the smallest turbulence scales are resolved.

5.2.1 General Outcome of the Simulations

The table 5.2 presents an overview of all the methods with the finest resolution simulated, using the D3Q27 stencil, and a lattice velocity of 0.1 ($U_0$).

The CSM and NRCSM are located on the top of the table, followed by the entropic models, regarding the accuracy. However, the fastest LBM is the LBM-LES, lead by the Smagorinsky model and then by the CSM and NRCSM.
Table 5.2: Kida Vortex - Resume Table for LBMs D3Q27: Domain 300x300x300

<table>
<thead>
<tr>
<th>Method</th>
<th>Compressible</th>
<th>(%\text{error}<em>{avg}(\Omega</em>{peak}))</th>
<th>(%\text{error}_{avg})</th>
<th>MLUPs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>(\Omega)</td>
<td>(k)</td>
<td>(\Omega)</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>True</td>
<td>8.80</td>
<td>0.86</td>
<td>9.35</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>True</td>
<td>8.98</td>
<td>0.84</td>
<td>9.78</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>True</td>
<td>9.04</td>
<td>0.83</td>
<td>9.13</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>False</td>
<td>9.32</td>
<td>0.71</td>
<td>7.60</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>False</td>
<td>9.49</td>
<td>0.69</td>
<td>7.78</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>False</td>
<td>9.56</td>
<td>0.68</td>
<td>7.24</td>
</tr>
<tr>
<td>TRT-KBC-N1</td>
<td>True</td>
<td>9.79</td>
<td>0.81</td>
<td>8.69</td>
</tr>
<tr>
<td>TRT-KBC-N3</td>
<td>True</td>
<td>9.79</td>
<td>0.82</td>
<td>8.61</td>
</tr>
<tr>
<td>TRT-KBC-N4</td>
<td>True</td>
<td>9.79</td>
<td>0.83</td>
<td>8.59</td>
</tr>
<tr>
<td>TRT-KBC-N2</td>
<td>True</td>
<td>9.80</td>
<td>0.83</td>
<td>8.66</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>True</td>
<td>10.62</td>
<td>0.60</td>
<td>9.89</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>False</td>
<td>11.11</td>
<td>0.45</td>
<td>8.23</td>
</tr>
<tr>
<td>SRT-LES-0.1</td>
<td>True</td>
<td>13.13</td>
<td>0.15</td>
<td>11.15</td>
</tr>
<tr>
<td>SRT-LES-0.1</td>
<td>False</td>
<td>12.70</td>
<td>0.30</td>
<td>11.14</td>
</tr>
<tr>
<td>MRT3</td>
<td>False</td>
<td>19.37</td>
<td>0.12</td>
<td>18.50</td>
</tr>
</tbody>
</table>

SRT: Simple-relaxation time; MRT3: Multi-relaxation time with relaxation in moment of viscosity; CSM: Coherent structure model; NRCSM: Non-rotational coherent structure model; LES: Smagorinsky model; Entropic: entropic model; TRT-KBC-N#: the different KBC models; FD: Finite difference method to obtain the CSM or NRCSM dynamic constant.
5.2.2 LBM CSM

Taking a closer look into the CSM model, one can observe that it is stable with meshes higher than 100 lattices per side and maintains a better accuracy compared to other LBMs. However, it cannot attain to coarser meshes with domain size of 50 lattices.

The two different approaches followed in the master thesis (with and without finite difference approach) have an excellent agreement even within coarser meshes (larger $\Delta t$), as to seen in table 5.3.

Table 5.3: Kida Vortex - Compressible LBM-CSM D3Q27

<table>
<thead>
<tr>
<th>Method</th>
<th>Domain Size</th>
<th>$%\text{error}<em>{\text{avg}}(\Omega</em>{\text{peak}})$</th>
<th>$%\text{error}_{\text{avg}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRT-CSM</td>
<td>300</td>
<td>8.80 0.86</td>
<td>9.35 2.75</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>300</td>
<td>9.04 0.83</td>
<td>9.13 2.57</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>200</td>
<td>19.16 0.66</td>
<td>15.09 4.84</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>200</td>
<td>19.54 0.59</td>
<td>14.04 4.13</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>100</td>
<td>41.66 2.80</td>
<td>30.10 5.20</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>100</td>
<td>42.56 3.07</td>
<td>29.93 7.31</td>
</tr>
</tbody>
</table>

In the following figures 5.1 and 5.2, the outcomes for the CSM with and without the finite difference approach are plotted, and they are contrasted with the outcomes of the resolved LBGK and the KBC with a resolution of 200 lattices per side of Bösch [6]. Both methods present a good agreement with the evolution of the kinetic energy and enstrophy.

5.2.3 LBM NRCSM

In contrast to the CSM, the NRCSM is stable with a coarse mesh with 50 lattices as domain size with the finite difference approximation. However, the results for NRCSM with and without finite difference vary slightly, as observed in the table 5.4. Moreover, the NRCSM-FD has lower accuracy than the KBC models but higher than LBM-LES and MRT3.
Figure 5.1: Lattice Boltzmann with coherent structure model without finite difference approximation

Figure 5.2: Lattice Boltzmann with coherent structure model with finite difference approximation
Table 5.4: Kida Vortex - Compressible LBM-NRCSM D3Q27

<table>
<thead>
<tr>
<th>Method</th>
<th>Domain Size</th>
<th>$%<em>{error</em>{avg}}(\Omega_{peak})$ $\Omega$</th>
<th>$k$</th>
<th>$%<em>{error</em>{avg}}$ $\Omega$</th>
<th>$k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRT-NRCSM</td>
<td>300</td>
<td>8.98</td>
<td>0.84</td>
<td>9.78</td>
<td>1.98</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>300</td>
<td>10.62</td>
<td>0.60</td>
<td>9.89</td>
<td>3.88</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>200</td>
<td>19.48</td>
<td>0.61</td>
<td>13.50</td>
<td>1.32</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>200</td>
<td>21.84</td>
<td>0.16</td>
<td>17.20</td>
<td>2.01</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>100</td>
<td>43.00</td>
<td>3.11</td>
<td>30.22</td>
<td>5.59</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>100</td>
<td>47.82</td>
<td>4.59</td>
<td>34.92</td>
<td>7.90</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>50</td>
<td>60.97</td>
<td>4.21</td>
<td>48.24</td>
<td>23.75</td>
</tr>
</tbody>
</table>

As in the CSM, the NRCSM with and without the finite difference approach results are plotted in the figures 5.3 and 5.4, and they are contrasted with the outcomes of the resolved LBGK and the KBC with a resolution of 200 (lattices per side) of Bösch [6]. Both methods present a good agreement with the evolution of the kinetic energy and enstropy.

### 5.2.4 Stability Analysis

Besides the simulations to assess the stability and accuracy of the CSM and NRCSM, a set of simulations has been performed to verify the maximum Reynolds number that could be achieved by this LBM model. For this purpose, the mesh is set to 100 lattices per side and ran until a Reynolds number of 100,00.00, which is the expected Reynolds number for building flow simulations, with a lattice configuration of D3Q19 and D3Q27.

The CSM and NRCSM are stable in the whole range using the D3Q27 stencil and present a constant increase of the maximum enstropy versus the Reynolds number, as shown in the figures 5.5 and 5.6. However, there exist different values of the maximum enstropy between the CSM and NRCSM, due to the assumption of a non-rotational structure in the NRCSM.

One significant observation in the KBC model is that the increase in the maximum enstropy is not as smooth as expected (presence of oscillations); this can be seen in the figure 5.7.
Figure 5.3: Lattice Boltzmann with non-rotational coherent structure model without finite difference approximation

Figure 5.4: Lattice Boltzmann with non-rotational coherent structure model with finite difference approximation
On the other hand, with a stencil of D3Q19, the NRCSM is stable and with constant increase of the maximum enstrophy in the whole range; however, the CSM is not stable in the whole range for incompressible flows. This behavior can be appreciated in the figure 5.8.

5.2.5 General remark

As can be appreciated, the CSM and NRCSM have high accuracy and stability with the D3Q27 stencil in comparison to the D3Q19. This behavior agrees with the general recommendation to use the stencil D3Q27 for turbulent flows in the LBM methods. Moreover, CSM and NRCSM have a higher performance and faster calculation than the KBC models but slower than the LBM LES (Smagorinsky model).

![Figure 5.5: Maximum enstropy for different Reynolds number for CSM D3Q27](image1)

![Figure 5.6: Maximum enstropy for different Reynolds number for NRCSM D3Q27](image2)
Figure 5.7: Maximum enstrophy for different Reynolds number for KBC models

Figure 5.8: Maximum enstrophy for different Reynolds number for CSM D3Q19
5.3 TURBULENT CHANNEL FLOW

The turbulent channel flow with $Re_τ = 180$ is chosen as a second benchmark for the CSM and the NRCSM. This standard benchmark represents a homogeneous wall-bounded turbulent flow, and gives insight if the CSM and NRCSM can deliver practical results for low-order statistics with coarse meshes. Moreover, the results are compared with the DNS simulation of Moser [22] and with the LBM-LES (Smagorinsky model).

The turbulent channel flow can be characterized by three different Reynolds numbers, based on the bulk mean velocity ($u_m$) and height of the channel flow ($H$), centered line velocity ($u_c$) and half-height of the channel flow ($δ$) or the wall friction velocity ($u_τ$) and the half-height of the channel flow ($δ$).

$$Re = \frac{u_m H}{ν} , \quad Re_c = \frac{u_c δ}{ν} \quad \text{and} \quad Re_τ = \frac{u_τ δ}{ν}$$  \hspace{1cm} (5.11)

Moreover, there are two different approaches to get the time of an eddy turn over to the get the statistical characteristic of the flow, based on the centered line velocity ($u_c$) or the wall friction velocity ($u_τ$).

$$T_{eddyTurnover} = \frac{δ}{u_c} \quad \text{and} \quad T_{eddyTurnover} = \frac{δ}{u_τ}$$  \hspace{1cm} (5.12)

The definition of which characteristic numbers or parameters are essential for the comparison between DNS simulations and different LBMs. For this purpose, the Reynolds number and the turn over eddy time regarding the wall friction velocity are selected.

The wall friction velocity is defined, through the shear stress at the wall in the normal direction ($τ_w$), as:

$$u_τ = \sqrt{\frac{τ_w}{ρ}} \quad \text{where} \quad τ_w = ρν \frac{∂u_x}{∂y} \bigg|_{y=0}$$  \hspace{1cm} (5.13)

For the initialization of the LBM, a pressure driver or a driven force has to be set. Thus, a constant body force can be related to a pressure drop by:
\[
F = \frac{dp}{dx} = \frac{u_\tau^2 \rho}{\delta} = \frac{Re_\tau^2 \nu^2}{\delta^3}
\] (5.14)

However, this procedure gives an approximation, and the effective Reynolds number has to be checked a posteriori. There are two ways to do this procedure by measuring the wall shear stresses from a resolved simulation, or by using the log-law of the wall at the centerline:

\[
\frac{u_c}{u_\tau} = \frac{1}{\kappa} \cdot \ln(Re_\tau) + C
\] (5.15)

with the von Karman constant \(\kappa = 0.41\) and \(C = 5.5\).

Moreover, the flow field can be initialized by using the log-law at the wall with additional perturbations. Thus, the initial conditions are set up as [32]:

\[
u = \overline{\nu} + \nu'
\]

(5.16)

\[
u = \overline{\nu} + \nu'
\]

(5.17)

\[
u = \overline{\nu} + \nu'
\]

(5.18)

where \(\overline{\nu} = \overline{\nu} = 0\) and the fluctuations are set as \(u' = v' = w' = u_c(\tau_{rand} - 0.5), \tau_{rand} \in [0, 1]\).

In the streamwise direction we obtain \(\overline{u}\) with:

\[
u^+ = \begin{cases} y^+, & y^+ \leq y_\omega \\ \ln(y^+) / (\kappa + C), & y^+ > y_\omega \end{cases}
\] (5.19)

Where the superscript (+) indicates quantities measured with the "wall units" (non-dimensional), and \(y_\omega = 11.6\) is chosen as the inflection point for the values of the von Karman constant \(\kappa = 0.41\) and \(C = 5.5\). Here the "wall units" are the viscous length (\(l_\tau\)) and the wall friction velocity (\(u_\tau\)):

\[
l_\tau = \frac{\nu}{u_\tau}
\] (5.20)

In other words, \(u^+ = \overline{\nu} / u_\tau\) and \(y^+ = y / l_\tau\).

Another essential parameter for the simulation is the non-dimensional grid spacing, which gave an idea of the resolution of the simulation. If we want to resolve the smallest eddies, we need to have a
resolution smaller than the Kolgomorov scale. For a fully developed turbulent flow $\eta^+$ is 1.5 in the region near the wall. Since we are using a uniform mesh with $\Delta = 1$, we would need to satisfy the following relationship:

$$\Delta^+ = \frac{\Delta}{l_\tau} = \frac{Re_\tau}{\delta} < \eta^+ = 1.5$$

(5.21)

for a $Re_\tau = 180$, the half-height ($\delta$) of the channel should be larger than 120. However, this is not our case, and the value will give an idea of how coarse our mesh is compared with a fully resolved DNS simulation. In the DNS simulation of Moser [22], a non-uniform grid with values of $\Delta \approx 0.1$ near the wall is used.

The domain characteristics are $20\delta$ in the streamwise direction and $3\delta$ in the periodic transverse direction, this means $[20\delta, 3\delta, 2\delta]$ with $\delta \in [16, 32, 50]$.

The simulation is run until the transient phase is overcome, where the main flow characteristics does not change considerably $\approx 20T_{eddyTurnover}$. Afterwards the data is collected for a period of $400T_{eddyTurnover}$. Where the the velocity information is recorded every $0.01T_{eddyTurnover}$ and the space and planar average for the statistical study is performed. In this way, enough information is recorded for a correct statistical study.

On the other hand, to be able to compare the DNS results of Moser [22], the average relative error of the mean velocity in the streamwise direction $(u^+)$, and the root means square (r.m.s) velocity fluctuation in the three directions($u_{rms}^+, v_{rms}^+$ and $w_{rms}^+$) are used. Furthermore, the relative error of the centered line velocity is also presented. Finally, as a comparison within the different LBM models, the average relative error of the mean velocity in the streamwise direction $(\pi^+)$ is prioritized.

### 5.3.1 General Outcome of the Simulations

The simulations are the CSM, the NRCSM, and the LBM-LES (Smagorinsky) with the D3Q27 stencil, and a lattice velocity of 0.1. The results with the mesh of $\delta = 50$ can be observed in the table 5.5. For the specified mesh, the exact outcome of the equation (5.15) has been used, defining the relative error regarding the centered line velocity.
As a first insight, the CSM and NRCSM are located first in the table, regarding the average relative error of the mean velocity. However, the LBM-LES has a better speed than the CSM and NRCSM models, with almost the double performance.

Table 5.5: Turbulent Channel flow - Comparison of LBMs D3Q27 with $\delta = 50$

<table>
<thead>
<tr>
<th>Method</th>
<th>Compr.</th>
<th>$%error_{avg}$</th>
<th>MLUPs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$u_c$ $u^+$ $u_{rms}^+$ $v_{rms}^+$ $w_{rms}^+$</td>
<td></td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>True</td>
<td>2.98 0.88 2.18 47.95 20.42</td>
<td>140.41</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>True</td>
<td>3.08 0.91 0.96 48.17 20.19</td>
<td>142.90</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>False</td>
<td>3.53 0.94 1.22 46.37 20.55</td>
<td>163.52</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>False</td>
<td>3.51 1.05 1.26 47.77 20.17</td>
<td>158.31</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>True</td>
<td>2.27 1.08 1.18 49.08 19.52</td>
<td>130.52</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>False</td>
<td>2.83 1.18 2.14 49.12 20.38</td>
<td>161.52</td>
</tr>
<tr>
<td>SRT-LES-0.1</td>
<td>True</td>
<td>1.44 1.71 3.30 46.86 21.17</td>
<td>303.23</td>
</tr>
<tr>
<td>SRT-LES-0.1</td>
<td>False</td>
<td>0.63 1.78 2.30 46.48 21.00</td>
<td>306.38</td>
</tr>
<tr>
<td>SRT-LES-0.12</td>
<td>True</td>
<td>2.44 1.87 2.47 48.70 20.51</td>
<td>303.24</td>
</tr>
<tr>
<td>SRT-LES-0.12</td>
<td>False</td>
<td>0.17 2.00 3.05 46.99 20.86</td>
<td>306.42</td>
</tr>
<tr>
<td>SRT-LES-0.15</td>
<td>True</td>
<td>3.62 2.29 3.42 47.45 21.34</td>
<td>303.22</td>
</tr>
<tr>
<td>SRT-LES-0.15</td>
<td>False</td>
<td>1.71 2.42 3.29 46.14 20.94</td>
<td>306.43</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>True</td>
<td>4.75 3.38 6.14 46.96 22.23</td>
<td>125.74</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>False</td>
<td>4.40 3.43 6.59 46.25 22.04</td>
<td>161.52</td>
</tr>
</tbody>
</table>

SRT: Simple-relaxation time; MRT3: Multi-relaxation time with relaxation in the first moments; CSM: Coherent structure model; NRCSM: Non-rotational coherent structure model; LES: Smagorinsky model; FD: Finite difference method to obtain the CSM or NRCSM dynamic constant

5.3.2 LBM CSM

Taking a closer look into the CSM model, one can observe that it is stable with coarse meshes with $\delta = 16$, as illustrated in the table 5.6.
Table 5.6: Turbulent Channel flow - Compressible LBM-CSM D3Q27

<table>
<thead>
<tr>
<th>Method</th>
<th>$\delta$</th>
<th>$u_c$</th>
<th>$u_rms^+$</th>
<th>$v_rms^+$</th>
<th>$w_rms^+$</th>
<th>MLUPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRT-CSM-FD</td>
<td>50</td>
<td>3.08</td>
<td>0.91</td>
<td>0.96</td>
<td>48.17</td>
<td>20.19</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>50</td>
<td>2.27</td>
<td>1.08</td>
<td>1.18</td>
<td>49.08</td>
<td>19.52</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>32</td>
<td>9.87</td>
<td>1.40</td>
<td>5.35</td>
<td>44.64</td>
<td>24.79</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>32</td>
<td>4.38</td>
<td>1.91</td>
<td>2.36</td>
<td>46.16</td>
<td>20.32</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>16</td>
<td>0.86</td>
<td>10.75</td>
<td>53.65</td>
<td>93.54</td>
<td>48.68</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>16</td>
<td>5.39</td>
<td>13.68</td>
<td>95.78</td>
<td>158.99</td>
<td>91.59</td>
</tr>
</tbody>
</table>

However, a coarse mesh with $\delta = 16$ cannot resolve the eddies and presents average high relative errors in the r.m.s of the velocity fluctuations. This effect is due to a large grid spacing ($\Delta^+ = 11.25$), which means that there is no point in the viscous sub-layer (before the inflection point of the log-law of the wall).

Meanwhile, considering a medium mesh with $\delta = 32$ and $\Delta^+ = 5.625$ (two points in the viscous sub-layer region), the outcomes are good, and the average relative errors are lower than 10%. The outcomes of the medium mesh can be appreciated in the figure 5.9 and 5.10. At last, the most refined mesh with $\delta = 50$ and $\Delta^+ = 3.6$ (three points in the viscous sub-layer region) shows average relative errors below than 5% as presented in the figures 5.11 and 5.12.

5.3.3 LBM NRCSM

Taking a closer look into the NRCSM model in the table 5.7, one can observe that it is stable with coarse meshes with $\delta = 16$ and with better outcomes than the CSM model.

However, a coarse mesh with $\delta = 16$ cannot resolve the eddies and present average high relative errors in the r.m.s of the velocity fluctuations. This effect is due to a large grid spacing ($\Delta^+ = 11.25$), which means that there is no point in the viscous sub-layer (before the inflection point of the log-law of the wall) and therefore not taken in consideration.
Figure 5.9: Average stream-wise velocity and r.m.s. velocity fluctuations in wall units for the turbulent channel flow at Re = 180 using the CSM (δ=32)

Figure 5.10: Average stream-wise velocity and r.m.s. velocity fluctuations in wall units for the turbulent channel flow at Re = 180 using the CSM with FD (δ=32)
Figure 5.11: Average stream-wise velocity and r.m.s. velocity fluctuations in wall units for the turbulent channel flow at Re = 180 using the CSM (δ=50)

Figure 5.12: Average stream-wise velocity and r.m.s. velocity fluctuations in wall units for the turbulent channel flow at Re = 180 using the CSM with FD (δ=50)
Table 5.7: Turbulent Channel flow - Compressible LBM-NRCSM D3Q27

<table>
<thead>
<tr>
<th>Method</th>
<th>δ</th>
<th>$u_c$</th>
<th>$\bar{u}^+$</th>
<th>$u_{rms}^+$</th>
<th>$v_{rms}^+$</th>
<th>$w_{rms}^+$</th>
<th>MLUPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRT-NRCSM-FD</td>
<td>50</td>
<td>2.98</td>
<td>0.74</td>
<td>2.06</td>
<td>2.27</td>
<td>1.64</td>
<td>140.41</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>50</td>
<td>4.75</td>
<td>2.22</td>
<td>7.01</td>
<td>7.30</td>
<td>5.05</td>
<td>125.74</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>32</td>
<td>4.65</td>
<td>1.38</td>
<td>2.76</td>
<td>1.79</td>
<td>1.55</td>
<td>131.56</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>32</td>
<td>3.23</td>
<td>3.50</td>
<td>15.96</td>
<td>13.91</td>
<td>8.94</td>
<td>125.26</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>16</td>
<td>1.30</td>
<td>8.15</td>
<td>37.63</td>
<td>29.31</td>
<td>19.25</td>
<td>100.13</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>16</td>
<td>0.87</td>
<td>10.36</td>
<td>41.61</td>
<td>45.14</td>
<td>34.40</td>
<td>105.48</td>
</tr>
</tbody>
</table>

Meanwhile, considering a medium mesh with $\delta = 32$ and $\Delta^+ = 5.625$ (two points in the viscous sub-layer region), the outcomes are good, and the average relative errors are lower than 15% (figures 5.13 and 5.14). At last the most refined mesh with $\delta = 50$ and $\Delta^+ = 3.6$ (three points in the viscous sub-layer region) present average relative errors are lower than 8% and are presented in the figures 5.15 and 5.16.

Figure 5.13: Average stream-wise velocity and r.m.s. velocity fluctuations in wall units for the turbulent channel flow at $Re = 180$ using the NRCSM ($\delta=32$)
Figure 5.14: Average stream-wise velocity and r.m.s. velocity fluctuations in wall units for the turbulent channel flow at \( Re = 180 \) using the NRCSM with FD (\( \delta = 32 \))

Figure 5.15: Average stream-wise velocity and r.m.s. velocity fluctuations in wall units for the turbulent channel flow at \( Re = 180 \) using the NRCSM (\( \delta = 50 \))
Figure 5.16: Average stream-wise velocity and r.m.s. velocity fluctuations in wall units for the turbulent channel flow at \( \text{Re} = 180 \) using the NRCSM with FD (\( \delta = 50 \)).
CHAPTER 6
TOWARDS WIND FLOW SIMULATIONS

In the present chapter, the CSM, NRCSM, and some of the LBM models are applied to two standard benchmarks for environmental flows and the simulation of a nuclear power plant. In the present chapter, the question of whether the diverse LBM models can be applied to simulate environmental flow within a real-time simulations framework is addressed.

The first benchmark is a square cylinder in a 2D channel. The outcomes are compared with FVM methods and simulations results as in the work of Breuer [7] and bring the first insight into the usage of inlet and outlet boundary conditions in a moderate Reynolds number flow.

The second benchmark address a wind flow around an isolated building. This specific benchmark is considered as one of the primary benchmarks for outdoor building simulation applications, which set the foundations for the best practice guidelines for RANS and LES simulations [5]. The outcomes are compared with a RANS simulation using the STAR-CCM+ commercial software and the experimental data from the Institute of Advanced Industrial Science and Technology (AIST) in Japan [24].

At last, the simulation of the nuclear power plant is performed, addressing a possible industrial application of the LBM for real-time simulations. The outcomes are compared with simulation results provided by the company Framatome.

6.1 SQUARE CYLINDER

The square cylinder benchmark is the first 2D benchmark that tackles the simulation of wind flow around buildings and building aerodynamics. This benchmark studies includes the different flow regimes, from laminar to turbulent flow, and in our case the first test of the inlet (as prescribed velocity field) and outlet (as fixed pressure $\nabla P = 0$) boundary condition of the LBM.

This benchmark is characterized by a confined channel flow, through walls at the bottom and top
(non-slip boundary condition), an inlet with a prescribed flow, the outlet as zero pressure gradient
(fixed pressure boundary condition), and the square cylinder as an obstacle (non-slip boundary con-
dition) as shown in the figure 6.7.

![Figure 6.1: Geometry of Square Cylinder](image)

An important parameter is the blockage ratio, which is the relation between the size of the obstacle
\((D)\) and the height of the channel flow or domain size \((H)\):

\[
B = \frac{D}{H} \tag{6.1}
\]

Following the work of Breuer [7], a blockage ratio of \(1/8\) and for the geometry \(L = 6 \times H\) and
\(l = 3 \times H\) are used. Besides, the inflow is defined as a prescribed developed Poiseuille flow with a
maximum lattice velocity of 0.025:

\[
u_y = u_{\text{max}} \left(1 - \frac{y^2}{H^2}\right) \tag{6.2}
\]

For comparison, the result of the traditional CFD methods are used from Breuer [7] regarding the
unsteady regime from 60 to 300 Reynolds number. Where the Reynolds number is defined as:

\[
Re = \frac{Du_{\text{max}}}{\nu} \tag{6.3}
\]

The main parameters to compare are the Strouhal number \((St)\), calculated from the cylinder diam-
ter \((D)\), the measured frequency of the vortex shedding \((f)\) and the maximum velocity at the inflow
plane \(u_{\text{max}}\):

\[
St = \frac{fD}{u_{\text{max}}} \tag{6.4}
\]
A time series of the lift coefficient ($C_l$) is used to calculate the vortex shedding ($f$) by spectral analysis, this parameter will give us an idea of how well the eddies and the recirculation region are developed in the simulation side walls. Moreover, an essential characteristic of the Strouhal number evolution versus the Reynolds numbers is a maximum peak around Reynolds 150, followed by decay and a plateau. This phenomenon occurs due to the movement of the separation point from the trailing edge to the leading edge of the square cylinder. Thus, the correct behavior is affected by the resolution in the vicinity of the body [7].

Meanwhile, the drag coefficient ($C_d$) is used to compare the accuracy of the results. The drag coefficient gives us an idea of how well the pressure around the square cylinder, and the separation size is developed. In the unsteady regimen ($60; Re; 300$) the near-wake becomes unstable, and the shear layers adopt a sinusoidal oscillation shape [7]. However, due to the resolution near the body play a crucial role in the calculation of the pressure.

For comparison, the maximum ($\%error_{max}$) and average relative error ($\%error_{avg}$) are used.

### 6.1.1 General Outcome of the Simulations

The primary outcomes are presented in the table 6.1 for the highest resolution of $H = 320$ or $D = 40$. As a first insight, the CSM and NRCSM present excellent outcomes regarding the accuracy of the drag coefficient, lying in between the results of LBM-LES and the MRT3. However, LBM-LES has the highest computational performance, followed by the MRT3. Besides, in figure 6.2 can be observed the vortex formations for a Reynolds number of 200.

![Figure 6.2: Square Cylinder LBM-LES CSM - Vortex formation](image)
<table>
<thead>
<tr>
<th>Method</th>
<th>H</th>
<th>$\text{%error}_{\text{avg}}$</th>
<th>$\text{%error}_{\text{max}}$</th>
<th>MLUPs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$St$</td>
<td>$Cd$</td>
<td>$St$</td>
<td>$Cd$</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>False</td>
<td>1.02</td>
<td>1.12</td>
<td>2.23</td>
</tr>
<tr>
<td>SRT-LES-0.1</td>
<td>False</td>
<td>1.02</td>
<td>1.21</td>
<td>2.34</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>True</td>
<td>1.02</td>
<td>1.24</td>
<td>2.56</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>False</td>
<td>1.02</td>
<td>1.29</td>
<td>2.32</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>False</td>
<td>1.02</td>
<td>1.34</td>
<td>2.29</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>True</td>
<td>1.02</td>
<td>1.35</td>
<td>2.65</td>
</tr>
<tr>
<td>MRT3</td>
<td>False</td>
<td>1.02</td>
<td>1.42</td>
<td>2.66</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>True</td>
<td>1.02</td>
<td>1.45</td>
<td>2.61</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>False</td>
<td>1.02</td>
<td>1.46</td>
<td>2.31</td>
</tr>
<tr>
<td>SRT-LES-0.1</td>
<td>True</td>
<td>1.40</td>
<td>1.51</td>
<td>2.77</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>True</td>
<td>1.02</td>
<td>1.58</td>
<td>2.63</td>
</tr>
<tr>
<td>TRT-KBC-N1</td>
<td>True</td>
<td>0.98</td>
<td>1.63</td>
<td>2.21</td>
</tr>
<tr>
<td>TRT-KBC-N2</td>
<td>True</td>
<td>1.86</td>
<td>1.85</td>
<td>5.79</td>
</tr>
<tr>
<td>TRT-KBC-N4</td>
<td>True</td>
<td>1.54</td>
<td>3.45</td>
<td>3.75</td>
</tr>
<tr>
<td>TRT-KBC-N3</td>
<td>True</td>
<td>1.71</td>
<td>3.60</td>
<td>3.75</td>
</tr>
</tbody>
</table>

SRT: Simple-relaxation time; MRT3: Multi-relaxation time with relaxation in the first moments; CSM: Coherent structure model; NRCSM: Non-rotational coherent structure model; LES: Smagorinsky model; Entropic: entropic model; TRT-KBC-N#: the different KBC models; FD: Finite difference method to obtain the CSM or NRCSM dynamic constant.
6.1.2 LBM CSM

The LBM with the coherent structure model (CSM) presents good results in general. The results for different mesh resolutions can be observed in the table 6.2 for incompressible flow.

<table>
<thead>
<tr>
<th>Method</th>
<th>$H$</th>
<th>$%\text{error}_{\text{avg}}$</th>
<th>$%\text{error}_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$St$</td>
<td>$Cd$</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>320</td>
<td>1.02</td>
<td>1.12</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>320</td>
<td>1.02</td>
<td>1.46</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>160</td>
<td>2.67</td>
<td>2.01</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>160</td>
<td>2.60</td>
<td>2.41</td>
</tr>
<tr>
<td>SRT-CSM</td>
<td>80</td>
<td>6.57</td>
<td>4.40</td>
</tr>
<tr>
<td>SRT-CSM-FD</td>
<td>80</td>
<td>5.82</td>
<td>6.79</td>
</tr>
</tbody>
</table>

The CSM model presents good results in general. However, for the coarse mesh of $H = 80$ ($D = 10$), the development of the Strouhal number ($St$) is not the desired. This effect can be explained by the low resolution near the square cylinder, which lead to a low resolution of the separation point movement at Reynolds of 150. Thus, the coarsest grid should not be discussed seriously.

On the other hand, the behavior of the drag coefficient ($Cd$) and Strouhal number ($St$) is similar using both methods as observed in the figures 6.3 and 6.4 for the fine mesh. The discrepancies of the CSMs with the FVM are due to a low resolution near square cylinder and would be improved by a finer mesh near it.

Besides, the CSM with and without finite difference presents deviations between each other for the coarsest grid, as observed in figures 6.3 and 6.4. This effect might be due to a large time step, which contradict the assumption of our method.
Figure 6.3: Square Cylinder - CSM

Figure 6.4: Square Cylinder - CSM-FD
6.1.3 LBM NRCSM

The LBM with the non-rotational coherent structure model (NRCSM) presents good results in general. The results for different mesh resolutions can be observed in the table 6.3 for incompressible flow.

<table>
<thead>
<tr>
<th>Method</th>
<th>H</th>
<th>$%error_{avg}$</th>
<th>$%error_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$St$</td>
<td>$Cd$</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>320</td>
<td>1.02</td>
<td>1.29</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>320</td>
<td>1.02</td>
<td>1.34</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>160</td>
<td>3.05</td>
<td>2.51</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>160</td>
<td>2.60</td>
<td>2.24</td>
</tr>
<tr>
<td>SRT-NRCSM</td>
<td>80</td>
<td>7.15</td>
<td>5.45</td>
</tr>
<tr>
<td>SRT-NRCSM-FD</td>
<td>80</td>
<td>5.71</td>
<td>6.32</td>
</tr>
</tbody>
</table>

The NRCSM model presents good results in general. As in the CSM, for the coarse mesh of $H = 80$ ($D = 10$), the Strouhal number does not behave as expected, due to a low resolution near the body. Thus, the coarse grid should not be taken into consideration for the performance of the NRCSM.

Besides, the behavior of the drag coefficient ($Cd$) and Strouhal number ($St$) are similar with and without full finite difference approach. However, there are higher discrepancies compared with the CSM, but the difference gets reduced if we refine the mesh, as observed in the figure 6.5 and 6.6. Those discrepancies between our simulation and the FVM are due to the resolution near the square cylinder. Thus, we can overcome, then if we apply a finer mesh.

On the other hand, NRCSM compared with the CSM is less accurate, but the general behaviour of the unsteady flow is good.
Figure 6.5: Square Cylinder - LBM NRCSM

Figure 6.6: LBM NRCSM-FD
6.2 WIND TUNNEL TEST

The wind tunnel test is a 3D benchmark that tackles simulations of wind flow around an isolated building and in specific environmental flows with high Reynolds number. This specific type of benchmark has been studied since the '80s with the traditional CFD methods (RANS and LES) and experimental studies. In our case, we will use experimental data from a wind tunnel test released from the National Institute of Advanced Industrial Science and Technology (AIST) in Japan [24]. Besides, it will also be compared with runs in the commercial software StarCCM+.

This benchmark has different possibilities to set the boundaries. The floor and the building are set with non-slip boundary condition and the sides with a periodic boundary condition. On the other hand, the inlet, outlet, and top can be set in different ways. The basic geometry of the wind tunnel can be appreciated in the figure 6.7.

![Figure 6.7: Geometry of wind tunnel test](image)

Following the practice of the traditional CFD methods, we could set the inlet as a prescribed velocity field (for example, the power law), the outlet as a zero pressure gradient (constant density for LBMs), and the top as a slip boundary condition. However, this configuration requires a significant distance from the inlet to the building, to develop the flow and turbulence. Moreover, the distance from the building to the outlet has to be long enough to develop the eddies. There are several
recommendations for this type of configuration, however, these are only applicable to RANS or LES.

Another approach is to set the inlet and outlet with a periodic boundary condition, and the top with a prescribed velocity. This type of approach is attractive since it requires smaller domain sizes and avoids the complex boundary conditions. However, the scenario has to be set such in a way that the inlet does not lose information or significant difference with the ideal inlet flow field. This scenario will be used for the present thesis.

For environmental flow, it is common to use the power law as a reference to the developed flow field without obstacles:

\[ u(z) = u_s \left( \frac{z}{z_s} \right)^{\frac{1}{7}} \]  \hspace{1cm} (6.5)

where \( u_s \) is the wind velocity coefficient and \( z_s \) is the ground roughness. In our simulations, the power law will be used to initialize the flow field and will serve to set the top velocity.

On the other hand, the characteristic length of the wind tunnel test is the height of the building \( (h) \), and the characteristic velocity is the reference wind speed \( (u_s) \). From this physical characteristic we can obtain the Reynolds number and then recover the parameters required for our LBM setup.

The wind tunnel test from AIST has a height \( (h) \) of 0.1 meters, the wind velocity coefficient \( (u_s) \) is 2.14 m/s, and the ground roughness \( (z_s) \) is 0.5 m. Thus, considering air at 25°C, the Reynolds numbers is around 14,000.00.

For the LBM simulation, we select a lattice velocity of 0.05 as the characteristic velocity. Moreover, the domain of the simulation is characterized by the height of the building \( ([\alpha_x h, \alpha_y h, \alpha_z]) \), and different domain configurations are explored in the thesis.

6.2.1 StarCCM+

For the simulation in StarCCM+, the boundary characteristics were set as explained before for traditional CFD methods, while the domain characteristic was set according to environmental flow recommendations. However, these parameters are different from the set up of the nuclear power plant (NPR) of Framatome since we aim to simulate a basic benchmark with basic StarCCM++
recommendations in turbulence modeling.

The distance from the inlet to the building is ten times the height of the building \( l \), and the distance from the outlet to the building is 15 times the height of the building. The height \( H \) and width \( W \) of the domain are set up to be ten times the height of the building. Thus, the domain of the simulation is \([2.5, 1.0, 1.0] \) m.

For the grid generation, a trimmer cell mesh (cartesian mesh) with surface and prism layer mesher has been used with a base size of 0.025m. Moreover, a refinement region has been defined near the building with a base size of 0.00625 m, as it can be seen in the figure 6.8.

![Figure 6.8: Mesh of StarCCM+ simulation](image)

The simulation with Star CCM+ has been done using the k-epsilon model, one of the most basic turbulence model. Besides, there has not been special treatment on the walls, which could improve the accuracy of the simulations.

### 6.2.2 LBM simulations

The central problematic of the simulation has been the memory capacity of the NVIDIA Quadro P5000, which prevents the usage of fine meshes on larger domains. The simulations performed using the CSM and NRCSM were not stable for the different configurations tested. The main characteristic of the instabilities is the presence of shock waves near the building, as observed in the figure 6.9.

The observed shockwaves propagate and generate instabilities in the corners of the building, making the simulation unstable. To overcome this problem, we could reduce the velocity profile or use a previous stable simulation as initialization. However, those possibilities were not explored in the scope of this thesis.
Figure 6.9: Velocity magnitude at middle plane XZ, presence of shockwaves near the building

On the other hand, the LBM-LES with Smagorinsky is stable within the different resolutions and configuration. Thus, those outcomes will be presented.

As a first step, a moderate domain with the following characteristics has been simulated:

- Distance from inlet to the building (l): five times h
- Distance from the building to the outlet: thirty times h
- Distance from the building to the top: twelve times h
- Distance from the building to the sides: two times h

Thus, the domain shape is $[35h, 5h, 13h]$. In these simulations, a short period of stability has been used, in this case one second of simulation time, and for a period of eighteen seconds of simulation time several data has been taken. For comparison, we use the middle plane in the XZ plane (profile of the building) and the a XY plane at the middle of the height of the building. In the figures 6.10 and 6.11 we can see the results for a coarse grid and a middle grid refinement.

The outcomes of the simulations are acceptable for a coarse and middle grid refinement. Even though there are some variations at the inlet and the profile is not entirely conserved as shown in the figure 6.12.
Figure 6.10: Average stream velocity $\bar{u}$ m/s at middle plane XZ and at plane XY at $z = 0.05m$ with $h=10$

Figure 6.11: Average stream velocity $\bar{u}$ m/s at middle plane XZ and at plane XY at $z = 0.05m$ with $h=20$
However, if the simulation runs more time the velocity profile at the inlet decay, as observed in the figure 6.12. To overcome this problem, a possible solution is to add an external force to compensate the dissipation or to use an inlet boundary condition with a prescribed velocity profile.

As a test, an inlet boundary condition with a prescribed velocity profile according to the power law, equation (6.5), has been used. However, this led to instabilities, especially in the corners of the domain where different shock waves were observed as shown in the figures 6.13 and 6.14.

Thus, to have better accuracy of the wind tunnel test, it is necessary to explore the implementation of a force to compensate the dissipation and the boundary condition. Also, the alternative to use a pre-calculated scenario for the initialization is attractive to reduce the shockwaves near the building in the initialization. Thus, a continuous simulation with change of a force is attractive to run a real-time simulation with variations in the wind speed.
6.3 NUCLEAR POWER PLANT SIMULATION

The last simulation performed is an industrial application; in this case, the simulation of wind around a nuclear power plant (NPP). For this purpose, the geometry of the EPR (European Pressurised Reactor) has been provided by the company Framatome. The EPR is a third generation pressurised water reactor design, which has been developed mainly by Framatome (part of Areva), Electricité de France (EDF), and Siemens. The building arrangements can be seen in figure 6.15.

The EPR presents a chimney of 34 meters height. The buildings extent to 69 meters width and 104 meters length. As a first trial, a coarse grid is used to evaluate the stability of LBM, where one
lattice is equal to one meter in real life. Thus, the characteristic length is the height of the chimney, and the domain of the LBM domain is [408, 272, 340].

For the boundary conditions, a periodic boundary is set in the width and length direction (stream-wise direction). Besides, the top is treated as in the wind tunnel test simulation with prescribed velocity according to the power law.

`lbmpy` provide different tools to include the geometry of the nuclear power plant to the LBM domain, even though this procedure is quite extensive and laborious. However, waLBerla has different tools to provide support on this point.

For this thesis, the main building, as shown in the figures 6.17 and 6.16 has been implemented.

![Figure 6.16: TSN implemented in `lbmpy` - middle plane XY](image)

![Figure 6.17: TSN implemented in `lbmpy` - middle lane XY](image)

For the initial condition, the power law with one m/s of wind velocity coefficient and ground roughness of one meter is used, this can be appreciated in the figure 6.18.
6.3.1 LBM simulation

For the LBM simulation, the SRT with a Smagorinsky constant of 0.15 has been used, which have shown good performance in the wind tunnel test. However, the presence of shock waves makes the simulation unstable as presented in the figures 6.19 and 6.20.

In order to improve the simulation and make it stable, a different approach has been used. This approach is quite common for CFD traditional simulations. The approach consists of slowly incrementing the driven force of the fluid, in our case the prescribed velocity at the top, and make the simulation stable in the first steps. This approach makes the simulation more stable in the first steps, reducing the shock waves. After several time steps 24,000.00 approximately the simulation is still stable as it can be seen in figure 6.21.
Another possibility is to increase the viscosity in the first steps to reduce the shock waves near the EPR or to use the velocity of a pre-calculated scenario as initial conditions. However, these two ideas have not been explored yet.
CHAPTER 7
CONCLUSION

The theory behind the lattice Boltzmann method and the different approaches to improve the stability and accuracy of the original LBM has been discussed briefly for the simulation of highly turbulent flows. From these approaches, the sub-grid scale approach is the most promising, due to their high performance and the spatial relaxation through turbulence viscosity. Besides, from the different sub-grid scales models, the coherent structure model (CSM and NRCSM) is one of the most promising for the simulation of environmental flows due to its dynamic coefficient, and it has been implemented in the lbmpy.

The implemented model has been validated and compared not only with other LBMs but also with numerical and experimental results. Moreover, two different approaches to calculate the dynamical constant of the CSM and NRCSM has been implemented and tested (a direct approach and a self-developed alternative).

The CSM and NRCSM with the two variant to calculated the dynamic constant provided excellent results with respect accuracy and performance. However, limits of these models concerning industrial applications were encountered, namely shock-waves near bodies during the initialization. Thus, the Smagorinsky sub-grid scale model is applied for the envisaged industrial test case of simulation a wind flow around a nuclear power plant. Besides, different initial conditions, a ramp-up of the flow speed, led to stable results, and demonstrated the capability of the LBM to perform real-time simulations around buildings.
BIBLIOGRAPHY


